Crick-IT

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

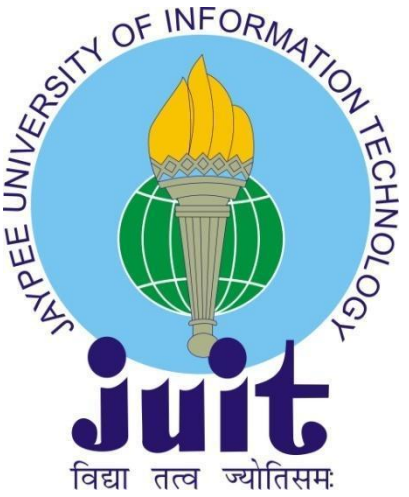
in

Computer Science and Engineering/Information Technology

By
Kuldeep Mishra 191263

Under the supervision of

Dr. Yugal Kumar
to



Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology
Waknaghat, Solan-173234, Himachal Pradesh

Certificate

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Crick-IT"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2023 to May 2023 under the supervision of **(Dr. Yugal Kumar)** (Associate Professor).
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Student Signature
Kuldeep Mishra, 191263.

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Supervisor Signature
Supervisor Name - Dr. Yugal Kumar
Designation - Associate Professor
Department name - CSE
Dated: 27/04/2023

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: .....

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: _____ **Department:** _____ **Enrolment No** _____

Contact No. _____ **E-mail.** _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at .....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

<table>
<tr><th>Copy Received on</th><th>Excluded</th><th>Similarity Index (%)</th><th colspan="2">Generated Plagiarism Report Details (Title, Abstract & Chapters)</th></tr>
<tr><td rowspan="2"></td><td rowspan="4">• All Preliminary Pages
• Bibliography/Images/Quotes
• 14 Words String</td><td rowspan="2"></td><td>Word Counts</td><td></td></tr>
<tr><td rowspan="2">Character Counts</td><td rowspan="2"></td></tr>
<tr><td rowspan="2">Report Generated on</td><td rowspan="2">Submission ID</td></tr>
<tr><td></td><td>Total Pages Scanned</td><td></td></tr>
<tr><td></td><td></td><td></td><td>File Size</td><td></td></tr>
</table>

Checked by
Name & Signature **Librarian**

.....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

First, I express my heartiest thanks and gratefulness to Almighty God for His divine blessing to make it possible to complete the project work successfully.

I am really grateful and wish my profound indebtedness to Supervisor **Dr. Yugal Kumar, Associate Professor ,** Department of CSE Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of my supervisor in the field of "Machine Learning" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, and reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to **Dr. Yugal Kumar**, Department of CSE, for his kind help in finishing my project.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non- instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Kuldeep Mishra

191263

Table Of Content

Sr. No.	Title	Page No.
1	Certificate	I
2	Plagiarism Certificate	II
3	Acknowledgement	III
4	Table of Content	IV
5	List of Abbreviations	V
6	List of Figures	VI
7	Abstract	VII
8	CHAPTER 1 - Introduction	1-20
9	CHAPTER 2 - Literature Survey	21-22
10	CHAPTER 3 - System Design,Algorithm	23-40
11	CHAPTER 4 - Performance Analysis	41-56
12	CHAPTER 5 - Conclusion	57-58
13	REFERENCES	59
14	APPENDICES	60-66

List of Abbreviations

Sr. No.	Abbreviations	Full Form
1	ML	Machine learning
2	DL	Deep learning
3	WHO	World Health Organization
4	CNNs	Convolutional neural networks
5	AUC-ROC	Area under the receiver operating characteristic curve
6	mAP	Mean average precision
7	GAP	Global Average Pooling
8	HSI	Hyperspectral imaging
9	NIRS	Near-infrared spectroscopy
10	AI	Artificial intelligence

List Of Figures

Sr. No.	Fig. No.	Description
1	1.1	Workflow of Scraping Data to Feature Extraction
2	1.2	Feature Extraction Workflow
3	1.3	ML Workflow of Project
4	3.1	Web Scraping for List of Players
5	3.2	Web Scraping for Player Details
6	3.3	Software Workflow
7	3.4	Algorithm 1
8	3.5	Trained Model Result
9	3.6	Algorithm 2
10	3.7	Fetch Player Code Snippet 1
11	3.8	Fetch Player Code Snippet 2
12	3.9	Fetch Player Code Snippet 3
13	3.10	Drag & Drop Code Snippet 1
14	3.11	Drag & Drop Code Snippet 2
15	3.12	Drag & Drop Code Snippet 3
16	3.13	Simulation UI Code Snippet 1

17	3.14	Simulation UI Code Snippet 2
18	3.15	Simulation UI Code Snippet 3
19	3.16	Simulation UI Code Snippet 4
20	3.17	Simulation UI Code Snippet 5
21	3.18	Database Schema
22	3.19	Dataset CSV
23	3.20	Database UI - Players
24	3.21	Database UI Player Schema
25	4.1	Graph(Loss) for visualization
26	4.2	Showing Early Stopping of the Model
27	4.3	UI - Home Page
28	4.4	UI - Start Match Page
29	4.5	UI - Batting Lineup Page
30	4.6	UI - Simulation of Match
31	4.7	UI - Team Setup
32	4.8	UI - Match Summary

Abstract

Cricket is played in almost every corner of India, whether it be rural or urban, and is favored by both young and elderly people. Unlike any other sport, it connects billions of people in India. The media frequently covers cricket. The stakes are high in terms of both money and fame. Technology has changed the game significantly over the past few years. With streaming media, competitions, accessible live cricket viewing on mobile devices, and more, audiences are spoiled for choice.

The amount of runs a batsman scores, the number of wickets a bowler takes, the number of games a cricket team wins, the number of times a batsman reacts a specific manner to a particular bowling assault, etc. are all important statistics in the game of cricket. It's a significant thing to be able to use strong analytics tools, driven by numerical computing software like NumPy, to analyze cricketing data for the sake of both increasing performance and researching commercial potential, the general market, and the economics of cricket. Cricket analytics offers fascinating perspectives on the game as well as forecasting information on game results.

So, the main purpose of the project is to analyze cricket team performance, player behavior, game strategies, and game performance. It can be used to improve team performance in multiple ways. The key aim will be to provide a complete software for cricket team organizations to plan according to the statistics.

The project is intended for cricket team organizations around the world. The software will be useful for organizations to make strategies for the upcoming competitions, and to provide a match simulation facility. Cricket is one of the

most played sports around the world. According to the International Cricket Council (ICC), the sport is played in over 120 countries.

The software can be used not only for international cricket but also for local tournaments, which can increase the popularity of the software significantly. The software will be aiming at highlighting the strengths and weaknesses of the player. This will help the player to eliminate his weaknesses. For example, if a player is bad at facing short balls, he can improve his skills. The selectors will also have a good idea about every single player, for example, which player is to be selected for a bouncy or a dead pitch.

Using web scraping we built a dataset of players, weather conditions, pitches and information about various different stadiums. Using dimensionality reduction only the necessary attributes were kept which included their playing style and performance statistics. After that, we wrote the algorithms to simulate a match between the two teams to check out the performance of the players.

Our aim is to serve a full-fleshed working software for the end-users to simulate matches between their desired teams to select their playing 11 for the tournament.

Chapter-1

INTRODUCTION

1.1 Introduction

Crick-IT is an online fantasy sports game that revolves around cricket. In this game, participants create a virtual team consisting of actual cricket players, and points are given based on the performance of the selected players in real-life matches. The primary goal of the players is to gather the most points and secure the top spot on the leaderboard to emerge as the winner of the tournament.

The batting and bowling orders are critical components of cricket strategy and form the foundation of a Crick-IT match. Slight adjustments to the sequence can alter the game's outcome. A squad of 11 players and 3 substitutes are chosen from the participants, and there are no budget constraints or specific requirements for the number of batters, bowlers, or all-rounders. Fantasy programs suggest lineups that include five hitters, one wicketkeeper, two all-rounders, and three bowlers, but any player type is acceptable. Crick-IT tournaments may use Test cricket, Twenty20 cricket, and One Day International cricket, which are international formats played in daily games and rounds.

In India, Crick-IT is classified as a "game of skill" like fantasy sports in the United States. The game's character is determined by the dominant element, whether it be "skill" or "chance." According to the Indian Supreme Court, skill involves superior knowledge, training, focus, experience, and dexterity. Horse racing was considered a game of skill under the Tamil Nadu Gaming Act's

Section 11, which exempts games of "mere skill." Cricket is not as popular as fantasy (American) football in the United States due to the nation's poor understanding of the sport. However, the Unlawful Internet Gambling Enforcement Act of 2006 defines and exempts fantasy sports for money, considering them to reflect the relative knowledge or skill of the participants.

Crick-IT is a hybrid of cricket and video games, and live performances of players can be watched in real-time. The game's outcome is solely dependent on a player's prior performance, the opposition, the pitch, and the position they are playing in, and therefore cannot be altered.

The Game

Cricket is a team sport consisting of two teams, with 11 players on each side. The captain who wins the toss decides whether his team will bat or bowl first. If the team decides to bat first, their goal is to score as many runs as possible while preventing the opposing team from reaching their score.

There are numerous different ways to play cricket, but test cricket and one-day cricket are the most common. Each team bats twice throughout the five-day TEST cricket match, if time allows. The most common style is ONE DAY, which gives each side 300 balls to score runs. In the same number of balls, the other team tries to outscore them.

The players serve three purposes:

- BATSMEN
- BOWLING
- FIELDING

BATSMAN: A batsman is a person who scores runs off the balls that the bowler bowls.

BOWLER: A bowler is a person who throws a ball and attempts to "out" a batsman (dismissed from the ground).

FIELDER: Players (10) known as fielders help the bowler accomplish his objective and keep the batsmen from scoring runs.

A great batsman is made or determined by several things. A natural talent, a skilled teacher (Achrekar, who tutored Sachin Tendulkar), and undoubtedly performing at the appropriate moment are all certainties.

1.2 Problem Statement

The amount of runs a batsman scores, the number of wickets a bowler takes, the number of games a cricket team wins, the number of times a batsman reacts a specific manner to a particular bowling assault, etc. are all important statistics in the game of cricket. It's a significant thing to be able to use strong analytics tools, driven by numerical computing software like NumPy, to analyze cricketing data for the sake of both increasing performance and researching commercial potential, the general market, and the economics of cricket. Cricket analytics offers fascinating perspectives on the game as well as forecasting information on game results.

So, the main purpose of the project is to analyze cricket team performance, player behavior, game strategies, and game performance. It can be used to improve team performance in multiple ways. The key aim will be to provide a complete software for cricket team organizations to plan according to the statistics.

The project is intended for cricket team organizations around the world. The software will be useful for organizations to make strategies for the upcoming competitions, and to provide a match simulation facility. Cricket is one of the most played sports around the world. According to the International Cricket Council (ICC), the sport is played in over 120 countries.

The software can be used not only for international cricket but also for local tournaments, which can increase the popularity of the software significantly.

The software will be aiming at highlighting the strengths and weaknesses of the player. This will help the player to eliminate his weaknesses. For example, if a player is bad at facing short balls, he can improve his skills. The selectors will also have a good idea about each and every single player, for example, which player is to be selected for a bouncy or a dead pitch.

Using web scraping we built a dataset of players, weather conditions, pitches and information about various different stadiums. Using dimensionality reduction only the necessary attributes were kept which included their playing style and performance statistics.

1.3 Objectives

The main objective of the project is to create a database of players, teams, and stadiums. And then use the data from the database to create teams and simulate live matches.

We started with the creation of a python script that went to cricketing websites such as CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI, etc, and fetched the cricketer's name using web scraping. The list of players' names was stored in a file. We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

This list of players' names was further used to fetch the details of the players. We used the list of players' names to fetch information about the players such as their age, player type, Batting Hand, Bowling Style, Runs Scored, Wickets Taken, and Comfort (Batting & Bowling) using web scraping. We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

The following websites were used to perform web scraping: CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI. After fetching the required information for every player, a player object was created for every player and every player object went through another python script which validated the object if none of the required information was missing from it.

Every validated player object was stored in the database (SQLite3). We then used a regression algorithm and k-means algorithm to fetch the details such as the experience of the players, their stamina, their batting and bowling aggression.

In order to simulate a live cricket match two individuals/managers have to create their respective teams. The creation of the team and selection of players is solely based upon the auction. Both managers are given a fixed amount of virtual money to purchase the players of their choice. This process should be done smartly because if we spent a huge sum of money on a single player then we won't be able to spend much on the rest of the other players. Cricket is a game of a team and not a single individual. Therefore, the entire process of the auction should be based on statistics and other data analysis components so as to create the best team to compete with others.

Once the teams are formed, we select the start match option from the UI. After this, two teams are selected which have to compete with each other. Now toss is performed in order to have an unbiased game. The team that wins the toss gets to choose whether they want to do batting or fielding.

At last, we can see the real simulator that shows real-time score updates after every over is bowled. We can see the individual player's score, boundaries scored, number of wickets fallen till then, run rate, etc. We can also clearly see which player has the strike and which bowler is bowling. At the end of each innings, we get to see the match summary. After the end of the second innings, we get to know which team has won the match. If the game ends in a tie, the game is extended further for a super over. Rules of the super over are according to international standards. When the game finally gets over, we return to the home page.

1.4 Methodology

The project begins with scraping the data from the internet and then creating the desired dataset from it. After that dimensionality reduction was performed in order to increase the accuracy of our model. Finally, we trained and evaluated our model.

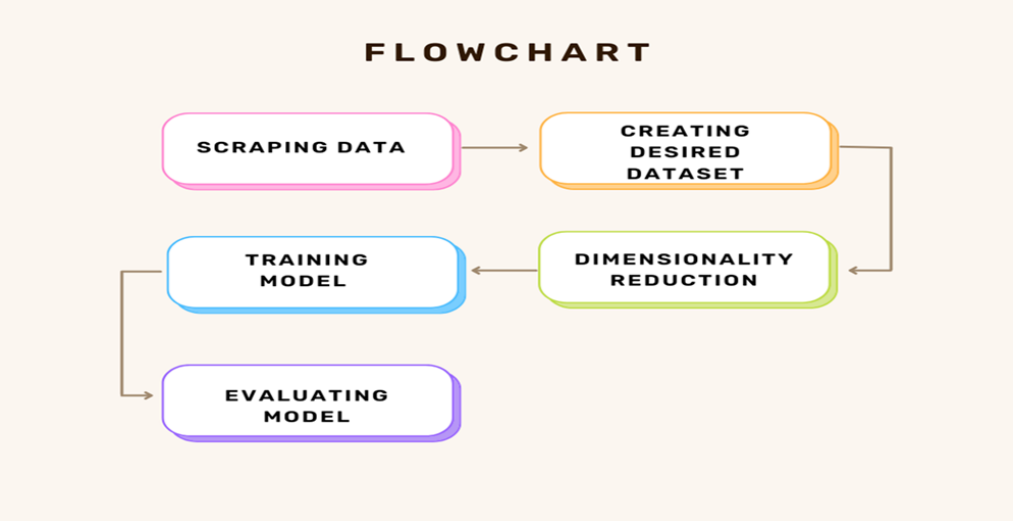


Fig 1.1: Workflow of Scraping Data to Feature Extraction

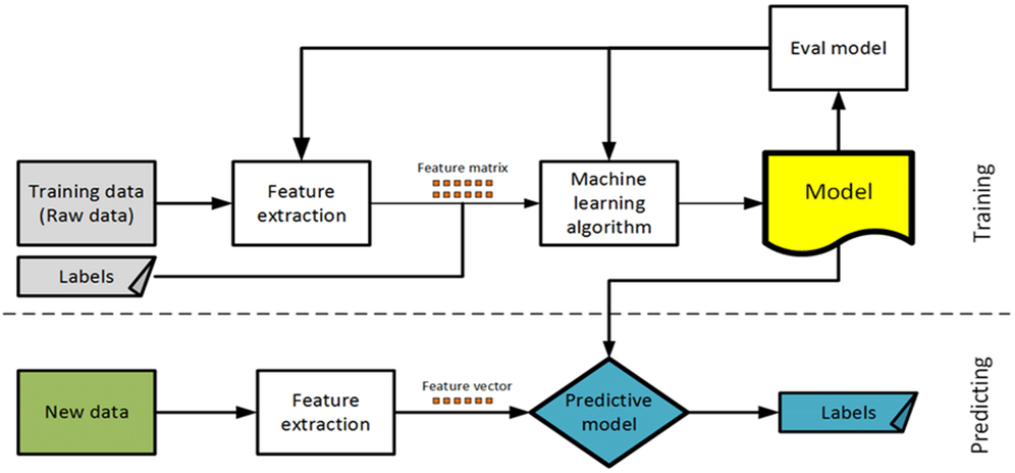


Fig 1.2: Feature Extraction Workflow

1.5 Motivation for the Work

Since we first learned about the game, India has been obsessed with cricket. We have cherished the game and the stars who have captured our hearts for many years. But over the past few years, the real game has given way to the virtual. Today, we are the players—or, more accurately, the selectors.

Crick-IT was designed to be similar to playing a game in a virtual environment with your team adhering to the same rules as the teams you see playing on the field.

For cricket enthusiasts, Crick-IT is a terrific resource. It provides real-time interaction with their favored teams and players and inspires you to start playing the game. People from all walks of life have come together to create fantasy teams and compete for points based on the performance of their selections. On online platforms, people create private leagues and engage in competition. Crick-IT enables you to participate in the game directly, going

beyond merely viewing it on a virtual screen. You can choose to suggest your friends, and you receive points for each referral.

A hundred fantasy game websites are readily apparent. The market has everyone's attention since real money is at stake. Currently, 11 Wickets is one such app that includes the most well-liked games, including Basketball, Football, Kabaddi, and Cricket. It is very simple and you can create your team using just your Android phone. You can then join any match while it is happening and match the performances of the players you chose. If the players you choose perform well in that match, you stand a chance to gain more points, and the total number of points awarded to each player determines your ranking.

The program that enables you to play the game in your preferred manner is simple to install, and Crick-IT is now gaining greatest popularity.

The organizer can simulate matches in Crick-IT in order to plan their strategies for forthcoming competitions/matches. They may simulate games and select the amount of overs at their discretion. The stats of each player would be shown when the simulation was over. For instance, the quantity of goals, dot balls, runs scored overall, etc.

The user will choose two suitable teams of his or her choosing for the game simulation. The user must now select the pitch and weather conditions at his or her own discretion. The user will be shown the outcome of the game along with the statistics after the simulation has taken place. Both the player's personal and professional information will be displayed on the Player Dashboard. The player has the opportunity to assess his or her performance, strengths, and weaknesses. The player will also receive advice regarding the skills that need improvement.

The Admin Dashboard will be accessed by the managers/selectors. The managers can view the list of players, their profile, their performance stats,

etc. They will also get the opportunity to use the features like a game simulator, team predictor, pitch analyzer, etc.

Crick-IT is a hybrid of Crick-IT and video games where we can watch the players' live performances in real-time with just one restriction: we cannot alter their performances because each player's performance is solely dependent on their prior performances, the opposition, the pitch (ground), and the position they are playing in.

The software will be aiming at highlighting the strengths and weaknesses of the player. This will help the player to eliminate his weaknesses. For example, if a player is bad at facing short balls, he can improve his skills. The selectors will also have a good idea about each and every single player, for example, which player is to be selected for a bouncy or a dead pitch.

So, the main purpose of the project is to analyze cricket team performance, player behavior, game strategies, and game performance. It can be used to improve team performance in multiple ways. The key aim will be to provide a complete software for cricket team organizations to plan according to the statistics.

1.6 Software Requirements

1.6.1. PYTHON

Python is a widely used computer programming language that is utilized for various purposes such as building websites, developing software, automating tasks, and performing data analysis. As a general-purpose language, Python can be used to create various types of programs and is not specific to any particular problem. This versatility and user-friendliness make it one of the most popular programming languages in use today. Python plays a crucial role in web development, including sending and receiving data to and from servers,

processing data, communicating with databases, and ensuring security. Python is typically used for developing the backend of websites and applications. Python's web development frameworks such as Django and Flask are popular choices for building web applications. They offer features such as URL routing and other functionalities to facilitate web development.

1.6.2. NUMPY

NumPy, or "numerical Python," is a Python module that enables the computation and manipulation of both multidimensional and one-dimensional array elements. It includes a high-performance multidimensional array object and tools for interacting with it. By fusing the traits of two different modules—Nummarray and the ancestor module Numeric-Travis Oliphant developed the NumPy package in 2005. Along with other intricate data structures, NumPy implements multidimensional arrays, matrices, and more. These data structures aid in the most effective computation of matrices and arrays. You can perform logical and mathematical operations on arrays using NumPy. Numpy is compatible with and used by many other well-known Python packages, such as pandas and matplotlib.

1.6.3. DJANGO

Django is a popular Python web framework that simplifies the process of building web applications. With Django, developers can focus on building their applications, while the framework handles the heavy lifting. One of the core principles of Django is component reusability, and it offers a range of features such as login systems, database connectivity, and CRUD operations out-of-the-box. Furthermore, Django provides a convenient way to manage navigation between pages of a website. Whenever a user requests a URL, Django identifies the corresponding view to be sent to her URL. This process is implemented in the urls.py file.

Django follows the MVT (Model View Template) design pattern.

· **Model -** The data you want to present, typically from a database.

· **Views -** Request handlers that return relevant templates and content based on user requests.

· **Template -** A text file (such as an HTML file) that contains the layout of a web page, including the logic for displaying data.

1.6.4. BEAUTIFUL SOUP

Beautiful Soup is a Python package that can extract data from HTML and XML files. The library works in tandem with your preferred parser and provides easy-to-use methods for navigating, searching, and manipulating the parse tree. By using Beautiful Soup, developers can save a significant amount of time and effort.

The package offers comprehensive tutorials covering all of the key aspects of Beautiful Soup 4. These tutorials include examples of how the library can be used, how it operates, how to use it effectively, how to achieve the desired results, and what to do if the output doesn't meet expectations.

1.6.5. JAVASCRIPT

JavaScript is a dynamic computer programming language that allows for client-side scripts to create dynamic web pages and interact with users. It is commonly used as a component of web pages and is an interpreted programming language that offers object-oriented capabilities.

Initially known as LiveScript, JavaScript was renamed by Netscape to JavaScript, possibly due to the popularity of Java at the time. The language

was first introduced in Netscape 2.0 in 1995 and has since been embedded in web browsers such as Internet Explorer and others.

JavaScript offers several advantages, including reduced server traffic due to the ability to check user input before page submission, immediate feedback for visitors without the need to reload the page, enhanced interactivity with the ability to design interfaces that respond to user actions, and the ability to create richer interfaces using sliders and drag-and-drop elements.

1.6.6. AXIOS

Axios is a promise-based HTTP client for both browser and Node.js environments, known for its isomorphic nature. This means that developers can use the same codebase in both environments. When running in a browser, Axios uses XMLHttpRequests, while in Node.js, it uses the built-in http module.

In contrast to other HTTP clients such as Fetch, which requires a two-step procedure for working with JSON data, Axios directly returns the data object from the API response, making it easier and more convenient for developers to work with JSON data.

1.6.7. JUPYTER NOTEBOOK

Jupyter Notebook is a web tool that allows users to create and share documents containing live code, equations, visualizations, and text. The project is free and open-source, and its maintenance is overseen by the staff of Project Jupyter.

Originally, Jupyter Notebooks were known as IPython Notebooks and were part of the IPython project. The project eventually evolved and expanded to

support other programming languages such as Julia and R, resulting in the name Jupyter, which stands for Julia, Python, and R. Although there are over 100 additional kernels available, Jupyter comes with the IPython kernel that supports Python programming.

1.6.8. VS CODE

Visual Studio Code, commonly known as VS Code, is a source-code editor created by Microsoft for Windows, Linux, and macOS platforms. It is built using the Electron Framework and features debugging support, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

Users can customize VS Code's theme, keyboard shortcuts, options, and extensions to add more functionality. The editor supports numerous programming languages, including Java, JavaScript, Go, Node.js, Python, C++, C, Rust, and Fortran.

Unlike traditional project-based editors, VS Code allows users to open one or more directories, which can be saved as workspaces for future use. This means that it can be used as a language-agnostic code editor for any language. Additionally, unwanted files and folders can be excluded from the project tree through the settings.

1.6.9. GIT

Git is a version control system used for source code management that allows multiple developers to collaborate on non-linear development projects. It is a free and open-source tool that can manage small to large projects effectively. With Git, changes to digital assets are logged, making it a valuable tool for tracking changes in software development. Git uses a distributed version

control technology that facilitates collaboration among several developers and supports tens of thousands of parallel branches, making nonlinear evolution easier.

1.6.10. WEB BROWSER

Web browsers are software applications that enable users to access websites by requesting and retrieving files from web servers, then rendering the pages on their devices. They are utilized on a range of devices including personal computers, laptops, tablets, and smartphones, with an estimated 4.9 billion users in 2020. Google Chrome is currently the most widely used browser, accounting for 65% of the global market share across all devices, with Safari in second place with 18%.

It is important to note that while web browsers and search engines are often confused, they are not the same thing. A search engine is a website that provides links to other websites, while a web browser is necessary to connect to a website's server and display its pages.

1.6.11. JINJA

Jinja is a fast, expressive, and customizable templating engine that allows placeholders with Python-like code to be written in the template. The data provided to the template is used to render the final document.

Jinja offers inheritance and inclusion of templates, as well as the ability to define and import macros. Auto escaping can be used in HTML templates to prevent cross-site scripting (XSS) attacks from untrusted user input. Furthermore, untrusted template rendering is safe within a sandboxed environment. Jinja supports async functions, which can automatically handle

sync and async functions without additional syntax, and Babel provides I18N support.

Templates can be precompiled or just-in-time compiled into cached and efficient Python code, and exceptions in templates point to the appropriate line for debugging. Finally, Jinja also offers flexible tests, filters, functions, and syntax.

1.6.12. PHOTOSHOP

Photoshop is a software program used for raster graphic design and photo editing that allows users to produce, modify, and manipulate various graphics and digital art. It also enables the import of images in various file types and the creation and editing of multi-layered raster pictures. Developed by Adobe Systems, Photoshop is available for both Windows and MacOS. Its layering capabilities provide depth and flexibility to the design and editing process, while its robust editing tools allow for nearly limitless possibilities when used in combination.

1.6.13. jQuery

jQuery is a JavaScript library that offers a wide range of features and is both fast and lightweight. It simplifies tasks such as HTML document manipulation and traversal, event handling, animation, and Ajax with an intuitive API that is compatible with a wide range of browsers. The library's versatility and extensibility have revolutionized the way millions of people write JavaScript today. By using jQuery, common tasks that would require multiple lines of JavaScript code can be performed by simply calling methods, making it a "write less, do more" library. jQuery also simplifies complex JavaScript features such as AJAX calls.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

1.7. PROPOSED APPROACH

The project workflow can be represented by the following steps:

1. Web scraping of players' information
2. Web scraping of player details using their names
3. Dataset validation
4. Additional feature extraction
5. Creation of teams using the dataset
6. Development of an algorithm to simulate the match
7. Evaluation of the algorithm.

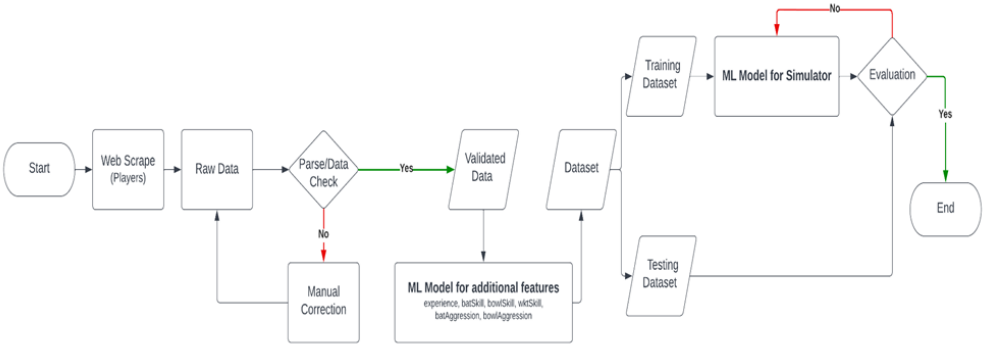


Fig 1.3: ML Workflow of Project

- We started with the creation of a python script that went to cricketing websites such as CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI, etc, and fetched the cricketer's name using web scraping.

- The list of players' names was stored in a file.

- We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

- This list of players' names was further used to fetch the details of the players.

- We used the list of players' names to fetch information about the players such as their age, player type, Batting Hand, Bowling Style, Runs Scored, Wickets Taken, and Comfort (Batting & Bowling) using web scraping.

- We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

- The following websites were used to perform web scraping: CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI.

- After fetching the required information for every player, a player object was created for every player and every player object went through another python script which validated the object if none of the required information was missing from it.

- Every validated player object was stored in the database (SQLite3)

- We then used a regression algorithm and k-means algorithm to fetch the details such as the experience of the players, their stamina, their batting and bowling aggression.

- In order to simulate a live cricket match two individuals/managers have to create their respective teams. The creation of the team and selection of players is solely based upon the auction. Both managers are given a fixed amount of virtual money to purchase the players of their choice. This process should be done smartly because if we spent a huge sum of money on a single player then we won't be able to spend much on the rest of the other players. Cricket is a game of a team and not a single individual. Therefore, the entire process of the auction should be based on statistics and other data analysis components so as to create the best team to compete with others.

- Once the teams are formed, we select the start match option from the UI. After this, two teams are selected which have to compete with each other.

- Now toss is performed in order to have an unbiased game. The team that wins the toss gets to choose whether they want to do batting or fielding.

- At last, we can see the real simulator that shows real-time score updates after every over is bowled. We can see the individual player's score, boundaries scored, number of wickets fallen till then, run rate, etc. We can also clearly see which player has the strike and which bowler is bowling.

- At the end of each innings, we get to see the match summary. After the end of the second innings, we get to know which team has won the match.

- If the game ends in a tie, the game is extended further for a super over. Rules of the super over are according to international standards.

- When the game finally gets over, we return to the home page.

1.7.1. Feature Extraction

The quantitative parameter of resolution changes and irregularities, which are not visible to the unaided eye, are extracted using the quantitative analytical technique known as "feature extraction." Feature extraction aims to discover abnormalities. We must extract specific data from images in order to classify photos using a classifier that needs to be trained on these characteristics.

A Train and Test set should be made.

The following step is to separate the data into training and testing.

Training Records

The training dataset's results serve as the algorithm's classroom. Each observation in supervised machine learning is made up of a recorded variable, an observed outcome variable, and maybe more.

Test Results

A test set is utilized to measure a model's efficacy based on a specific performance metric. It is crucial to ensure that the test set doesn't include any data from the training set. If the test set comprises training set observations, it becomes challenging to determine whether the algorithm has genuinely generalized or has merely memorized the examples from the training set.

A summarizing programme would be able to do a job with recent data swiftly and efficiently. In contrast, if a computer memorizes the training data by learning an overly complicated model, it may be able to reliably forecast the replies control variables for the training set, but it will not be able to predict the answers crucial factors.

Portion of Data	Explanation	Split Chosen
Training Data	A subset of the data used to train the model.	90%
Testing Data	A subset of the data used to assess the model's performance during training and excitable adjustment.	10%

Chapter-2

LITERATURE SURVEY

Cricket is a highly popular sport today that could benefit from the use of machine learning and artificial intelligence (AI) to increase accuracy. With more games being played, there is a growing amount of information on cricket games and players, making it an ideal domain for big data analytics. There are various advantages of using this data wisely, such as selecting team members, predicting the outcome of sporting events, and making a wide range of other future predictions using machine learning models or big data techniques. To demonstrate the potential of this approach, we present a study on predicting the outcome of an Indian Premier League (IPL) cricket match using a supervised learning method based on team composition. Our research shows that the relative strength of the competing teams is a key factor in determining the winner. To model team strength, we develop models of individual player performance based on their recent batting and bowling statistics.

Cricket has quickly advanced, making it a topic that all sports analysts find to be very fascinating. The informational indexes are still ambiguous and contradictory, though. In spite of thorough research, they were unable to predict the match's outcome with absolute certainty. The winner has been predicted using methods like SVM, Naive Bayes, KNN, and logistic regression, among others. Additionally, the ball-by-ball information as well as different rules were applied to the data that was gathered from the matches, which was obtained from websites like Kaggle, Cricsheet, etc. Training data comprised 80% of the sorted data, while testing data made up the remaining 20%. The two main programmes used were TensorFlow and Python. A confusion matrix came to an end with the performance measures.

In the paper “**Cricket Match Analytics Using the Big Data Approach**” This study draws two main conclusions. Firstly, the Spark framework is more effective than the conventional framework. Secondly, the linear regression model is the most accurate for predicting a cricket match's overall score. The study presents a model that can predict the winner of a cricket match based on the input circumstances of the game. The study demonstrates that the linear regression model offers the highest degree of accuracy. The prediction analysis shows that the model, utilizing the Spark machine learning framework, provides 96% accuracy.

The paper titled "**Cricket Analytics and Predictor**" allows users to register or log in. Upon successful registration and login, users have access to two models: a descriptive model that displays player data and a predictive model that forecasts the winning percentage of the user's selected team. In addition to the models, users can also read the latest news and tweets on the website. The website is an official site, and the owner benefits from accessing the IPL match details, user predictions, winning percentages, and player statistics. The coach benefits from utilizing prior data to prioritize players.

In the paper “**Dimensionality Reduction: A Comparative Review**” An analysis and comparison of dimensionality reduction methods are presented in the work. Because of the results, we can say that classic PCA still outperforms nonlinear approaches for dimensionality reduction, despite their enormous variation. Future nonlinear dimensionality reduction methods that don't rely on the data manifold's local qualities are something we expect to develop. The emergence of nonlocal dimensionality reduction methods with well-optimized objective functions, such (Kernel) PCA and autoencoders, is another change in direction that we predict.

Chapter-3

SYSTEM DEVELOPMENT

This chapter's goal is to give readers the theoretical context they need to understand the information in the report. The segmentation task is introduced along with the fundamental elements of a segmentation network. In addition, this chapter offers metrics for evaluating the networks and earlier studies on the topic.

3.1.1. Workflow

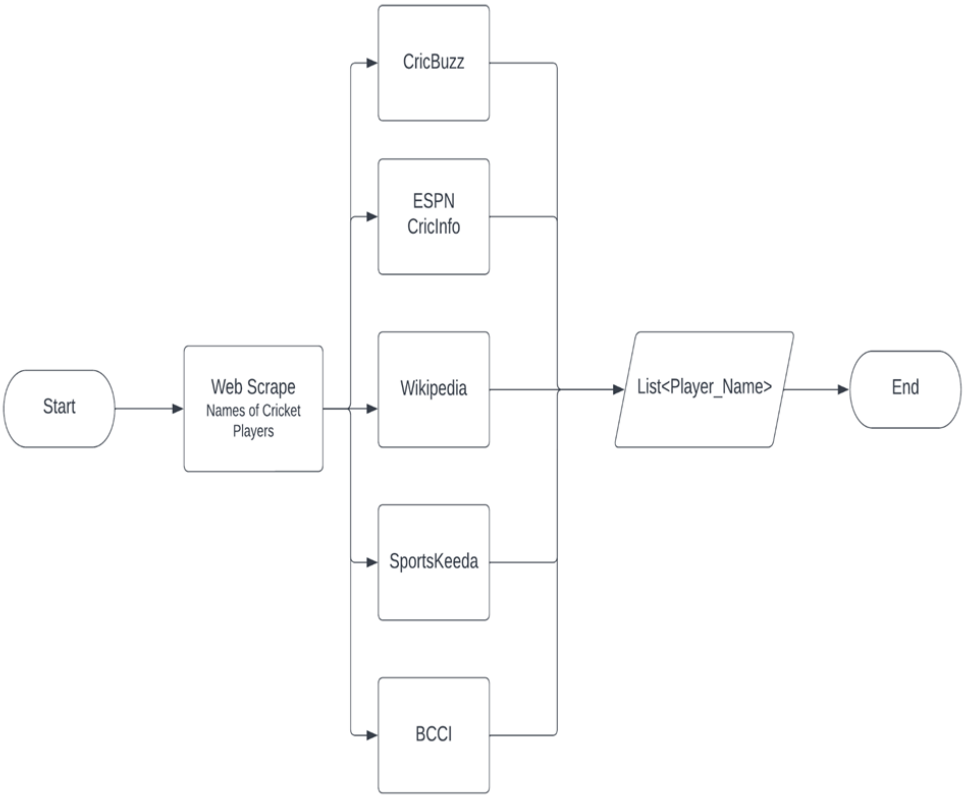


Fig 3.1: Web Scraping for List of Players

- We started with the creation of a python script that went to cricketing websites such as CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI, etc, and fetched the cricketer's name using web scraping.

- The list of players' names was stored in a file.

- We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

- This list of players' names was further used to fetch the details of the players.

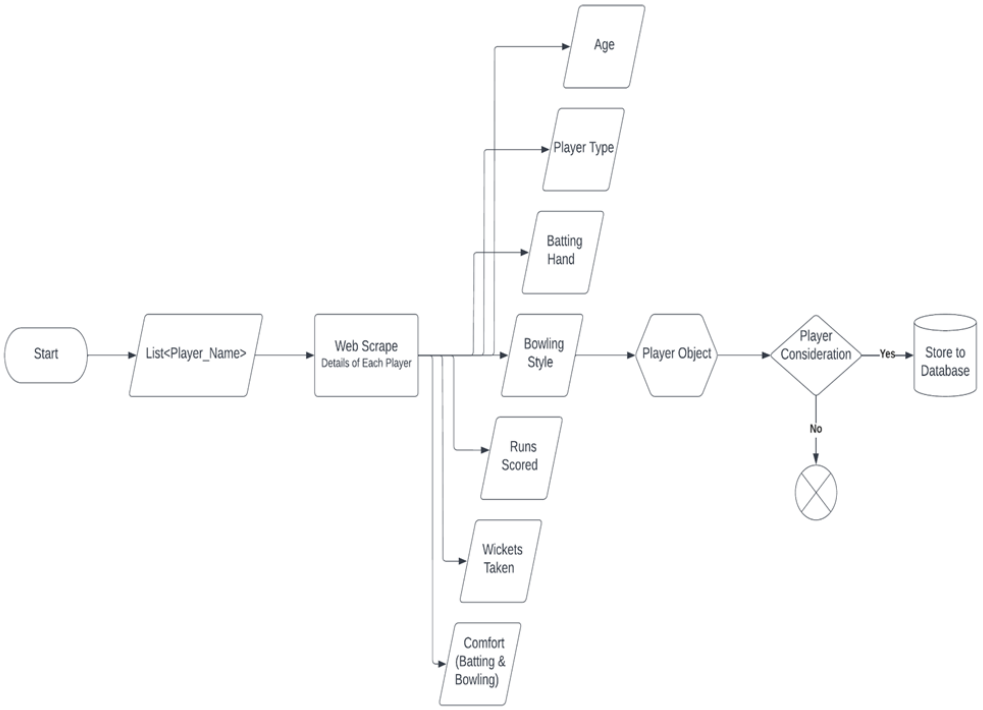


Fig 3.2: Web Scraping for Player Details

- We used the list of players' names to fetch information about the players such as their age, player type, Batting Hand, Bowling Style, Runs Scored, Wickets Taken, and Comfort (Batting & Bowling) using web scraping.

- We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

- The following websites were used to perform web scraping: CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI.

- After fetching the required information for every player, a player object was created for every player and every player object went through another python script which validated the object if none of the required information was missing from it.

- Every validated player object was stored in the database (SQLite3)

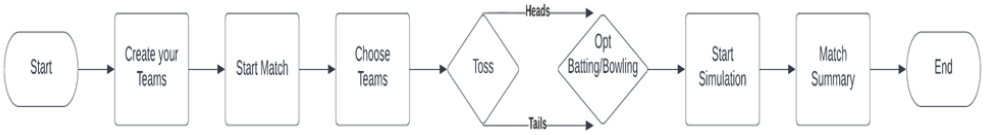


Fig 3.3: Software Workflow

- In order to simulate a live cricket match two individuals/managers have to create their respective teams. The creation of the team and selection of players is solely based upon the auction. Both managers are given a fixed amount of virtual money to purchase the players of their choice. This process should be done smartly because if we spent a huge sum of money on a single player then we won't be able to spend much on the rest of the other players. Cricket is a game of a team and not a single individual. Therefore the entire process of the

auction should be based on statistics and other data analysis components so as to create the best team to compete with others.

Let us give you an example of the IPL auction:

The IPL auction took place in Bengaluru over the weekend and showed both financial strength and hope for the league. Despite the challenges posed by two pandemic years, the IPL remains economically robust due to its marketing viability and strong financial foundation. The 10 franchises spent almost ₹552 crores, with 204 players going under the hammer, including 67 overseas cricketers. With the addition of two new teams and other squads rebuilding, team cultures were reflected in their choices. Mumbai Indians spent ₹8 crores on injured England speedster Jofra Archer, while Chennai Super Kings bought back core members. The auction also served as a mirror, showing the ruthless nature of the 'performance-age-marketability-availability-economics' matrix. Notably, Suresh Raina, Steve Smith, Aaron Finch, and Eoin Morgan were left out, revealing the importance of the matrix. The IPL's financial muscle has been demonstrated by the ability to continue despite pandemic-related challenges, unlike the Ranji Trophy, which had to take a break. Despite its success, a women's IPL has yet to be fully realized. The absence of Pakistan cricketers from the IPL is a reminder that the sport is not immune to diplomatic pressures.

- Once the teams are formed, we select the start match option from the UI. After this, two teams are selected which have to compete with each other.

- Now toss is performed in order to have an unbiased game. The team that wins the toss gets to choose whether they want to do batting or fielding.

- At last, we can see the real simulator that shows real-time score updates after every over is bowled. We can see the individual player’s score,

boundaries scored, number of wickets fallen till then, run rate, etc. We can also clearly see which player has the strike and which bowler is bowling.

- At the end of each innings, we get to see the match summary. After the end of the second innings, we get to know which team has won the match.

- If the game ends in a tie, the game is extended further for a super over. Rules of the super over are according to international standards.

- When the game finally gets over, we return to the home page.

3.1.2. ALGORITHMS

match.py > Match > setScorecard

```
from Team import Team
from Stadium import Stadium
from pmt import *
import random
import copy

class Match:
    def __init__(self, batTeam: Team, bowlTeam: Team, currGround: Stadium, overs=20, target=150, innings=1):  # this function is the main function
        self.superOverScore = {}
        self.perBallResults = {}
        self.perBallResults1 = {}
        self.perBallResults2 = {}
        self.perOverRuns = {}
        self.totalRuns = 0
        self.wickets = 0
        self.balls = 0
        self.requiredRunRate = 0
        self.actualReqRunRate = 0
        self.currentRunRate = 0
        self.frequency = {}  # testing
        self.bfrequency = {} # testing
        self.extras = {"wd": 0, "nb": 0, "total": 0}
        self.boundaries = {6:0, 4:0, 3:0, 2:0, 1:0, 0:0}

        self.battingTeam = batTeam
        self.bowlingTeam = bowlTeam
        self.battingLineup = batTeam.battingLineup
        self.bowlingLineup = bowlTeam.bowlingLineup

        self.striker = self.battingLineup.pop(0)
        self.battingTeam.batsmanBin.append(self.striker)
        self.nonStriker = self.battingLineup.pop(0)
        self.battingTeam.batsmanBin.append(self.nonStriker)
        self.bowler = self.bowlingLineup.pop(0)
        self.bowlingTeam.bowlerBin.append(self.bowler)
```

Fig 3.4: Algorithm 1

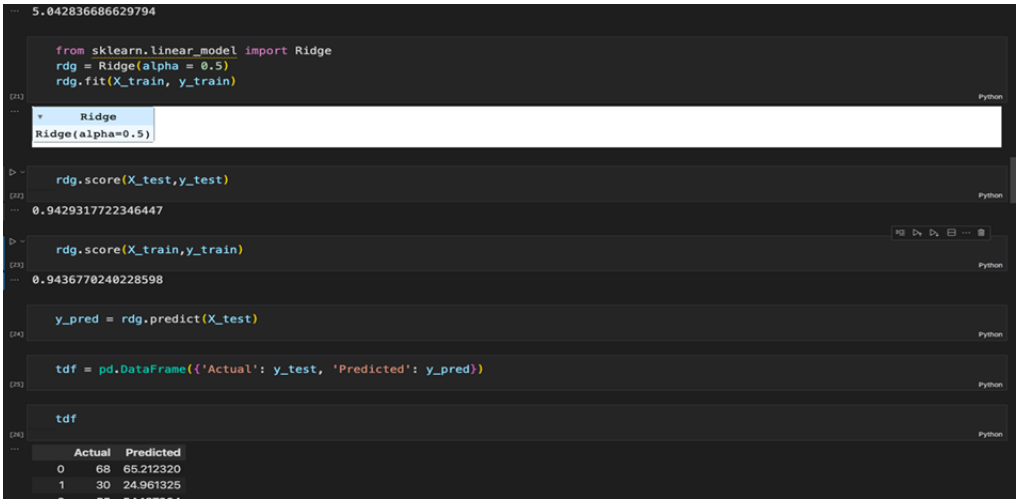


Fig 3.5: Trained Model Result

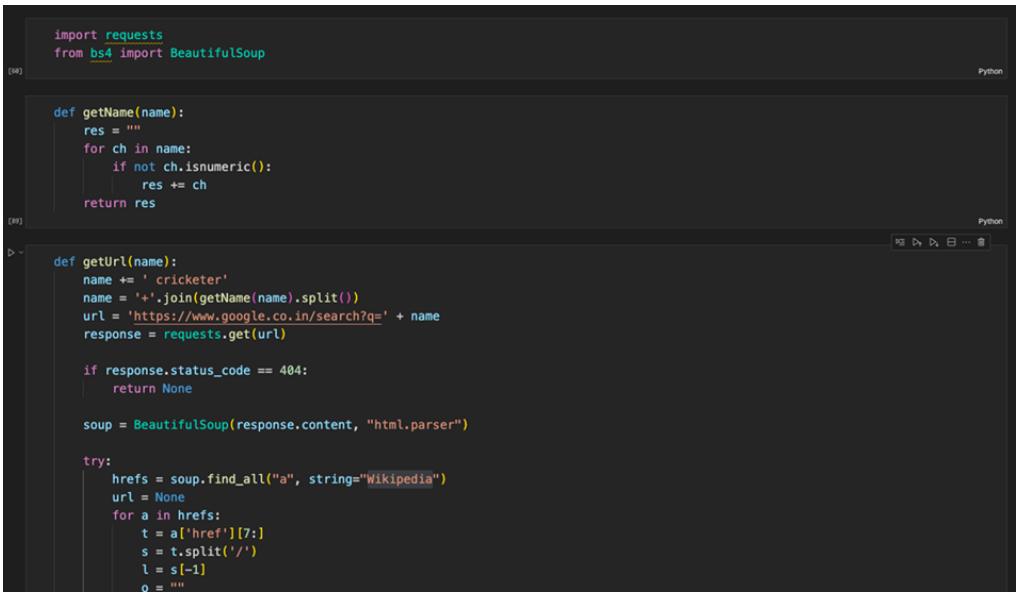
```
def updateProjectedScore(self):
    if self.innings != 1 or self.currentRunRate == 0:
        return
    if self.balls in [60, 90] or self.requiredRunRate <= 0:
        self.target = self.currentRunRate * 20

def updateRunRate(self):
    try:
        self.requiredRunRate = ((self.target-self.totalRuns)/(120-self.balls))*6
    except:
        self.requiredRunRate = (self.target-self.totalRuns)
    if self.innings == 1:
        try:
            self.actualReqRunRate = ((self.ground.averageRuns-self.totalRuns)/(120-self.balls))*6
        except:
            self.actualReqRunRate = (self.ground.averageRuns-self.totalRuns)
    else:
        self.actualReqRunRate = self.requiredRunRate
    try:
        self.currentRunRate = (self.totalRuns/self.balls)*6
    except:
        self.currentRunRate = 0

def startMatch(self):  # this function is the main function
    self.updateRunRate()
    self.setupPlayers()  # setting up some values which need to be set when player comes
    for over in range(self.overs):
        if self.target < self.totalRuns and self.innings != 1:
                break
        if self.wickets == 10:
                break
        ball = 1
        prevOverRuns = self.totalRuns  # testing
        prevOverWickets = self.wickets  # testing
        while ball != 7:
            self.updateProjectedScore() # this will only run after 10 and 15 overs.   **************NEW******************
```

Fig 3.6: Algorithm 2

3.1.3. FETCH LIST OF PLAYERS



```
import requests
from bs4 import BeautifulSoup
```

```
def getName(name):
    res = ""
    for ch in name:
        if not ch.isnumeric():
            res += ch
    return res
```

```
def getUrl(name):
    name += ' cricketer'
    name = '+'.join(getName(name).split())
    url = 'https://www.google.co.in/search?q=' + name
    response = requests.get(url)

    if response.status_code == 404:
        return None

    soup = BeautifulSoup(response.content, "html.parser")

    try:
        hrefs = soup.find_all("a", string="Wikipedia")
        url = None
        for a in hrefs:
            t = a['href'][7:]
            s = t.split('/')
            l = s[-1]
            o = ""
```

Fig 3.7: Fetch Player Code Snippet 1

The list of players' names will be stored in a file. We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL.

We used the list of players' names to fetch information about the players such as their age, player type, Batting Hand, Bowling Style, Runs Scored, Wickets Taken, and Comfort (Batting & Bowling) using web scraping.

We used the Request Module of Python along with the Beautiful Soup Library available for Python to web scrape the players from the particular URL. The following websites were used to perform web scraping: CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI.

```python
def fetch(name: str):
    url = getUrl(name)

    if not url:
        return None

    response = requests.get(url)

    if response.status_code == 404:
        return None

    try:
        soup = BeautifulSoup(response.content, "html.parser")
        age = getAge(soup)
        ptype = getType(soup)
        style = getStyle(soup)
        return {'age': age, 'ptype': ptype, 'style': style}
    except:
        return None
```

[106] Python

```python
name = 'Xavier Marshall'
print(fetch(name))
```

[123] Python

```
{'age': 36, 'ptype': None, 'style': {'batting': 'Right-handed', 'bowling': 'Right-arm offbreak'}}
```

Fig 3.8: Fetch Player Code Snippet 2

```python
import requests
from bs4 import BeautifulSoup
```

[2] Python

```python
def fetch():
    url = 'https://www.cricketcountry.com/players/'
    response = requests.get('https://www.cricketcountry.com/players/')

    if response.status_code == 404:
        return None

    try:
        soup = BeautifulSoup(response.content, "html.parser")
        content = soup.find_all("a")
        players = set()
        for i in range(len(content)):
            if content[i]['href'][0:39] == 'https://www.cricketcountry.com/players/' and content[i]['href']
                players.add(content[i].text.strip())
        return players
    except:
        return None
```

[7] Python

```python
players = fetch()
```

[8] Python

Fig 3.9: Fetch Player Code Snippet 3

We started with the creation of a python script that went to cricketing websites such as CricBuzz, ESPN CricInfo, Wikipedia, SportsKeeda, and BCCI, etc, and fetched the cricketer's name using web scraping.

3.1.4. DRAG & DROP FEATURE

```
$(".bowling").hide()
$("#save-btn").hide()
const players = $(".player")
const lists = $(".list")
let draggedItem = null
for(var i=0; i<players.length; i++){
    const player = players[i]

    player.addEventListener('dragstart', function(){
        draggedItem = this
        setTimeout(function(){
            player.style.display = 'none'
        }, 10)
    })
    player.addEventListener('dragend', function(){
        setTimeout(function(){
            draggedItem.style.display = 'block'
            draggedItem = null
        }, 10)
    })
```

Fig 3.10: Drag & Drop Code Snippet 1

It is a good interface solution to use drag and drop. Taking something and dragging it and dropping it is an obvious and simple way to complete many tasks, from ordering something to copying and moving documents (as in file managers).

A section about drag and drop is included in the current HTML standard, along with special events like dragstart, dragend, and others.

By handling drag-and-dropping files from the OS file manager into the browser window, for example, these events enable us to support special types of drag-and-drops. After that, JavaScript can access the data in those files.

```
for(var j=0; j<lists.length; j++){
    const list = lists[j]

    list.addEventListener('dragover', function(e){
        e.preventDefault()
    })
    list.addEventListener('dragenter', function(e){
        e.preventDefault()
    })
    list.addEventListener('drop', function(e){
        if(ct % 14 == 0){
            var flg = 0
            for(var t=0; t<options.length; t++){
                if(options[t] == draggedItem.innerHTML){
                    flg += 1
                }
            }
            if(flg == 0){
                $(".wkt")[0].innerHTML += `<option value="${draggedItem.innerHTML}">${draggedItem.
                innerHTML}</option>`
                $(".cpt")[0].innerHTML += `<option value="${draggedItem.innerHTML}">${draggedItem.
                innerHTML}</option>`
                options.push(draggedItem.innerHTML)
            }
```

Fig 3.11: Drag & Drop Code Snippet 2

```
var players_list = []
$("#get-btn").click(function(){
    select = $(".selected .player")
    if(select.length == 11){
        $(".end").css("padding-left", "0")
        $(".main").hide()
        $(".bowling").show()
        for(var i=0; i<select.length; i++){
            $(".bowling .bowlers-available")[0].innerHTML += `<div class="pbowler"><div class="name">${select[i].innerHTML}</div><div class="sum-sub"></div><div
            class="input-group mb-3" style="width: 10%;">
                <input type="number" id="${select[i].innerHTML}" class="form-control" placeholder="" value="0" name="${select[i].innerHTML}" aria-label="Example
                text with button addon" aria-describedby="button-addon1" style="background:transparent; text-align:center;color:white;font-weight:bold;
                border:none;border-bottom:2px solid white;border-radius:0px;font-family: 'Montserrat', sans-serif;outline:none;">
            </div></div>`
            players_list.push(select[i].innerHTML)
            $("#get-btn").hide()
            $("#save-btn").show()
        }

        for(var w=0; w<players_list.length; w++){
            var temp = ""
            for(var z=0; z<players_list[w].length; z++){
                if(players_list[w][z] == " "){
                    temp += "-"
                }else{
                    temp += players_list[w][z]
                }
            }
            console.log(temp)
            $(".bowling")[0].innerHTML += `<input type="hidden" name="player${w+1}"  value=${temp}>`
        }

    }else{
        alert("Firstly, select 11 players.")
    }
})
```

Fig 3.12: Drag & Drop Code Snippet 3

By default, only text and images are draggable. You just need to keep the mouse button down when dragging a picture. You must choose some text and drag it similarly to how you would drag a picture in order to move the text. The dragstart event fires on the draggable element you are dragging as soon as you start moving the mouse while holding down a mouse button. When an element cannot be dropped on another element, the pointer changes to a no-drop sign (a circle with a line across it).

3.1.5. SIMULATION OF GAME

```
var data1234 = ""
var team12 = ""
function handleboard(data, team){
    count = 1
    $(".batting-team1")[0].innerHTML = `<div class="scr1">Score - <span class='bolder'>${data[team][1]['score']}/${data[team][1]['wickets']} (${data[team][1]
    ['overs']})</span></div>`
    while(count != 11){
        for(p in data[team][1]['batting']){
            if(count == data[team][1]['batting'][p]['number']){
                $(".batting-team1")[0].innerHTML += `<div class="perbatsman">
                    <div class="name">${p}</div>
                    <div class="scr">${data[team][1]['batting'][p]['runs']}(${data[team][1]['batting'][p]['ballsfaced']})</div>
                </div>`
            }
        }
        count += 1
    }
    for(bowler in data[team][1]['bowling']){
        console.log(data[team][1]['bowling'][bowler])
        $(".bowling-team2")[0].innerHTML += `<div class="perbowler">
            <div class="name">${bowler}</div>
            <div class="scr">${data[team][1]['bowling'][bowler]['runsgiven']}/${data[team][1]['bowling'][bowler]['wickets']}(${data[team][1]['bowling'][bowler]
            ['ballsbowled']})</div>
        </div>`
    }
    count = 1
    $(".batting-team2")[0].innerHTML = `<div class="scr1">Score - <span class='bolder'>${data[team][2]['score']}/${data[team][2]['wickets']} (${data[team][2]
    ['overs']})</span></div>`
    while(count != 11){
        for(p in data[team][2]['batting']){
            if(count == data[team][2]['batting'][p]['number']){
                $(".batting-team2")[0].innerHTML += `<div class="perbatsman">
                    <div class="name">${p}</div>
                    <div class="scr">${data[team][2]['batting'][p]['runs']}(${data[team][2]['batting'][p]['ballsfaced']})</div>
                </div>`
            }
```

Fig 3.13: Simulation UI Code Snippet 1

```
    for(bowler in data[team][2]['bowling']){
        console.log(data[team][2]['bowling'][bowler])
        $(".bowling-team1")[0].innerHTML += `<div class="perbowler">
            <div class="name">${bowler}</div>
            <div class="scr">${data[team][2]['bowling'][bowler]['runsgiven']}/${data[team][2]['bowling'][bowler]['wickets']}(${data[team][2]['bowling'][bowler]
            ['ballsbowled']})</div>
        </div>`
    }
}

$(".after-match").hide()
$("#start2nd").hide()
$("#matchSummary").hide()

var team1 = "{{team1.name}}"
var team2 = "{{team2.name}}"
var team1_print_name = "{{team1_print_name}}"
var team2_print_name = "{{team2_print_name}}"

if(team1_print_name.length < 6){
    $(".team-name").addClass("team-name-addon")
}

$(".team-name")[0].innerHTML = team1_print_name
$(".scoreboard")[0].innerHTML = String("0/0") + String("  - (") + String("0.0") + String(")")
$(".target")[0].innerHTML = "1<sup>ST</sup> INNINGS"

var data = NaN

$.ajax({
    type: 'GET',
    url: 'scorecard_json',
    success: function(response){
        data = response
    },
    error: function(response){
```

Fig 3.14: Simulation UI Code Snippet 2

We have used the ajax tool to fetch the result of each ball faced between the particular batsman and bowler. The result of each ball depends on pitch conditions, batsman skills, bowler skills, wicketkeeper skills, stamina of batsman and bowler, experience of batsman and bowler.

Fig 3.15: Simulation UI Code Snippet 3

We can see the individual player's score, boundaries scored, number of wickets fallen till then, run rate, etc. We can also clearly see which player has the strike and which bowler is bowling.

Fig 3.16: Simulation UI Code Snippet 4

```
bowler = firstInning[ball]['bowler']
fullname = bowler['name'].split(' ')
firstName = fullname[0]
lastname = fullname[fullname.length-1]
runs_wickets = String(bowler['runs']) + "/" + String(bowler['wickets'])
bowler_over = String(Math.floor(bowler['balls']/6)) + String(".") + String(Math.round(bowler['balls']%6))
bname = firstName[0].toUpperCase() + "." + lastname
$.ajax({
    type: 'GET',
    url: `http://127.0.0.1:8000/match/start/${team1}/${team2}/get_player_json/${bowler['name']}/`,
    success: function(response){
        $(".bowler")[0].innerHTML = `<img class="bowl" src="${response['img_href']}" alt="" srcset="" height="200px" width="215px"><div
        id="bowlscore">${bname}<br>${runs_wickets}<br>${bowler_over}</div>`
    },
    error: function(response){
        console.log('an error occured')
    }
})

if((Math.round(((ball-1)%6)) == 0)){
    faulty = 1
    results = []
    dictionary = {}
}

if(firstInning[ball]['result'] == "W"){
    dictionary[faulty] = "dot wicket"
    faulty += 1
    results.push(firstInning[ball]['result'])
}else if((firstInning[ball]['result'] == "6")||(firstInning[ball]['result'] == "4")){
    dictionary[faulty] = "dot six"
    faulty += 1
    results.push(firstInning[ball]['result'])
}else{
    dictionary[faulty] = "dot"
    faulty += 1
    results.push(firstInning[ball]['result'])
```

Fig 3.17: Simulation UI Code Snippet 5

The above code snippets help us to simulate the match in the frontend of the software. The frontend is fully responsive i.e., it works well on mobile devices as well as on laptops and personal computers too.

Ajax helps us to simulate the match in real time. It fetches the result of each ball by requesting the particular URLs to fetch the desired outputs.

At the end of each innings, we get to see the match summary. After the end of the second innings, we get to know which team has won the match. If the game ends in a tie, the game is extended further for a super over. Rules of the super over are according to international standards. When the game finally gets over, we return to the home page.

3.2. DATABASE SCHEMA

We have used the dbSqlite3 database for the project.

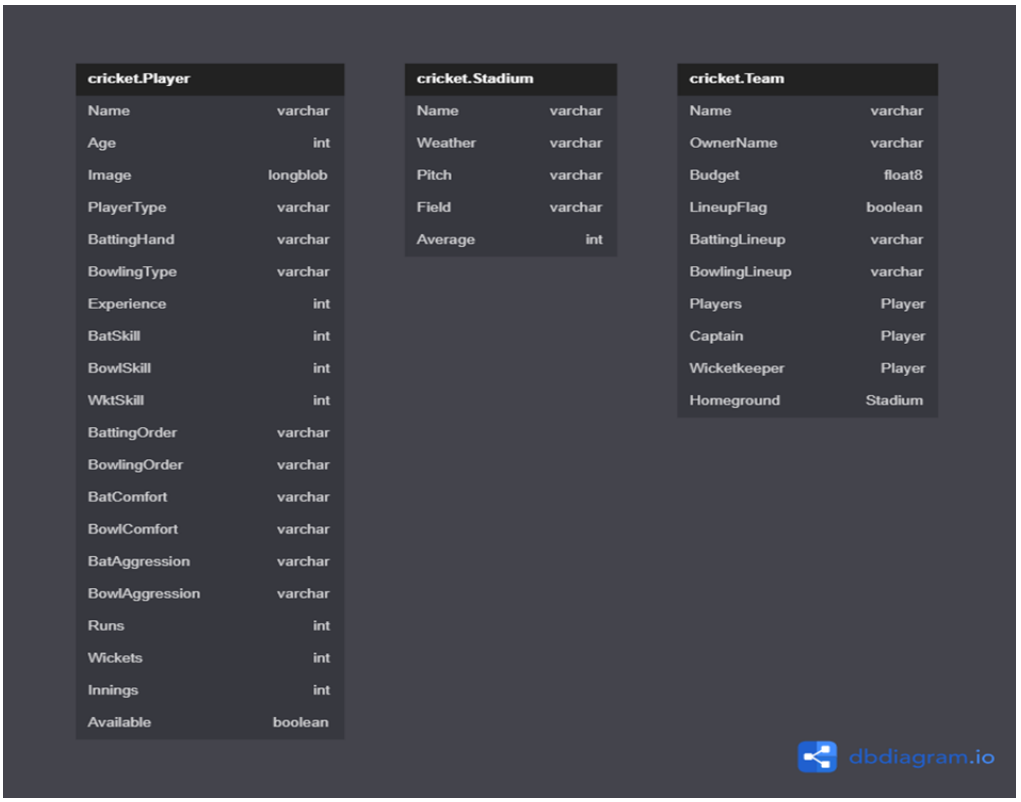


Fig 3.18: Database Schema

The above figure shows the Database of our software. We have used SQLite3 as our database to store the date data of every single player. The database mainly contains three tables Player, Stadium and Team:

- Player table contains all the necessary details about every individual player such as name, age, image, player type, batting hand, bowling type, experience, bat skill, bowl skill, wicket skill, batting order, bowling order, bat comfort, bowl comfort, batting aggression, bowling aggression, runs scored, wickets taken, total innings played and whether the player is available or not

- Stadium table contains most of the necessary information about the stadium such as the name of the stadium, weather conditions, pitch conditions, field type and average runs scored in that particular stadium.

- Team table contains all the necessary information about the entire team such as name of the team, its owner, budget associated with the team, line up flag, batting line up, bowling line up, players information, captain information, wicketkeeper information and the team's home ground.

3.3. DATASET

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
name	age	ptype	battingStyle	bowlingStyle	innings	runsScored	ballsFaced	50's	100's	wickets	ballsBowled	runsGiven	5's	10's	exp
P1	39	allrounder	right hand	left hand	78	2233	2102	7	6	82	538	734	2	0	30
P2	21	allrounder	right hand	right hand	41	1335	1295	6	3	46	1386	1609	1	2	22
P3	27	bowler	left hand	left hand	41	80	94	0	0	107	3523	4369	1	7	39
P4	28	wicketkeepe	left hand		138	6405	5965	16	29	0	64	223	0	0	51
P5	19	wicketkeepe	right hand		21	788	790	2	4	3	3	8	0	0	14
P6	17	batsman	right hand	right hand	22	560	553	2	1	1	97	216	0	0	10
P7	32	allrounder	left hand	left hand	182	5034	4793	17	16	207	1677	2435	18	1	57
P8	33	wicketkeepe	left hand		64	3129	3005	13	11	0	14	32	0	0	37
P9	33	allrounder	right hand	right hand	226	6221	5937	28	16	234	6585	8615	15	2	67
P10	24	bowler	right hand	left hand	61	174	216	0	0	141	4042	4106	3	5	45
P11	39	bowler	right hand	right hand	6	28	21	0	0	14	137	163	1	0	4
P12	34	bowler	left hand	right hand	70	203	256	0	0	197	5586	6361	0	0	42
P13	27	batsman	left hand	right hand	78	3250	3286	10	14	2	30	57	0	0	38
P14	24	batsman	right hand	left hand	66	2343	2206	10	7	2	94	200	0	0	30
P15	18	wicketkeepe	right hand		24	1296	1262	1	7	0	25	30	0	0	18
P16	27	bowler	left hand	left hand	60	217	258	0	0	142	1513	1721	9	1	39
P17	32	batsman	right hand	right hand	50	2009	1952	7	7	0	74	117	0	0	29
P18	27	batsman	left hand	left hand	134	4925	4713	13	20	2	7	17	0	0	47
P19	39	allrounder	left hand	left hand	232	7285	6919	29	24	461	15266	16538	17	14	79
P20	18	batsman	right hand	left hand	6	523	515	3	2	1	29	87	0	0	10
P21	19	bowler	left hand	right hand	20	57	85	0	0	35	1090	1336	4	1	14
P22	40	wicketkeepe	right hand		142	5905	5680	18	21	2	59	77	0	0	47
P23	18	allrounder	right hand	right hand	10	296	301	2	0	19	559	679	0	0	6
P24	22	allrounder	left hand	right hand	72	1908	1869	10	4	130	1340	1366	3	4	37
P25	34	batsman	right hand	right hand	164	6930	6859	25	28	0	65	106	0	0	56
P26	30	bowler	right hand	right hand	28	33	72	0	0	54	1793	1806	1	3	21
P27	31	bowler	right hand	left hand	196	503	640	0	0	453	4606	5216	56	10	63
P28	26	allrounder	left hand	right hand	84	3067	3012	18	7	114	1048	1214	10	1	42
P29	40	wicketkeepe	right hand		332	13650	13307	60	49	1	39	156	0	0	87
P30	38	allrounder	left hand	left hand	297	7793	7596	38	16	531	6889	9742	64	15	77
P31	25	allrounder	right hand	right hand	30	808	768	4	2	34	1362	1884	0	1	15
P32	22	allrounder	right hand	right hand	63	1661	1654	9	3	78	2466	2954	5	4	33
P33	23	wicketkeepe	left hand		41	1467	1425	6	5	0	0	0	0	0	21
P34	28	batsman	left hand	right hand	11	481	476	5	0	1	51	121	0	0	8
P35	26	allrounder	right hand	left hand	78	1634	1680	7	3	135	3159	4628	10	1	39
P36	36	batsman	right hand	left hand	93	3859	3755	13	13	1	36	48	0	0	39
P37	31	bowler	left hand	left hand	160	474	586	0	0	246	5092	5284	4	8	59
P38	26	allrounder	right hand	left hand	29	989	994	6	2	30	647	873	3	1	18
P39	20	bowler	right hand	left hand	26	82	107	0	0	70	1166	1233	8	2	24

Fig 3.19: Dataset CSV

This is the generated dataset of every player that contains necessary information such as name, age, player type, batting style, total innings played, runs scored, balls faced, total half centuries, centuries scored, wickets taken, balls bowled, runs given, and total experience level.

The image above shows the practical implementation of DBSQlite3 database software which is provided by Django. It is similar to various other database management systems such as Microsoft Excel, MySQL, PostGres SQL, MongoDB, etc.

The reason we chose SQLite is because it is available for free and can further be used at production level. It predominantly comes integrated with the Django application which we have used in the creation of web pages.

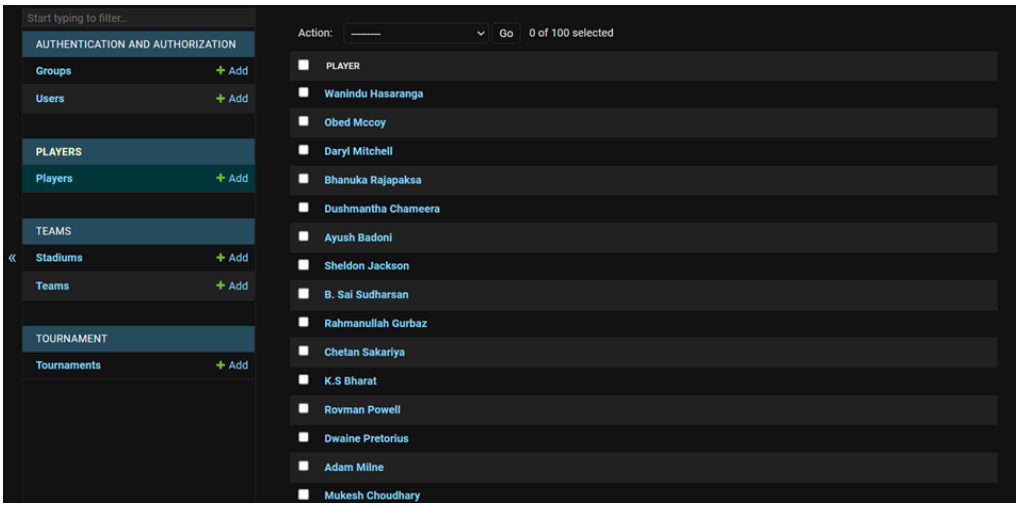


Fig 3.20: Database UI - Players

The image displayed above shows the UI of the DBMS software (SQLite3) which is integrated with the Django application. This page can only be accessed by the administrator. We can click on the players' name and see the details of every player.

The reason we chose SQLite is because it is available for free and can further be used at production level. It predominantly comes integrated with the Django application which we have used in the creation of web pages.

We can delete the players from the list or add new players. Along with these features we can also add / delete stadiums, teams, groups, users, administrators and create / delete tournaments. The UI of the above page is user friendly and can be used by any new administrator / user.

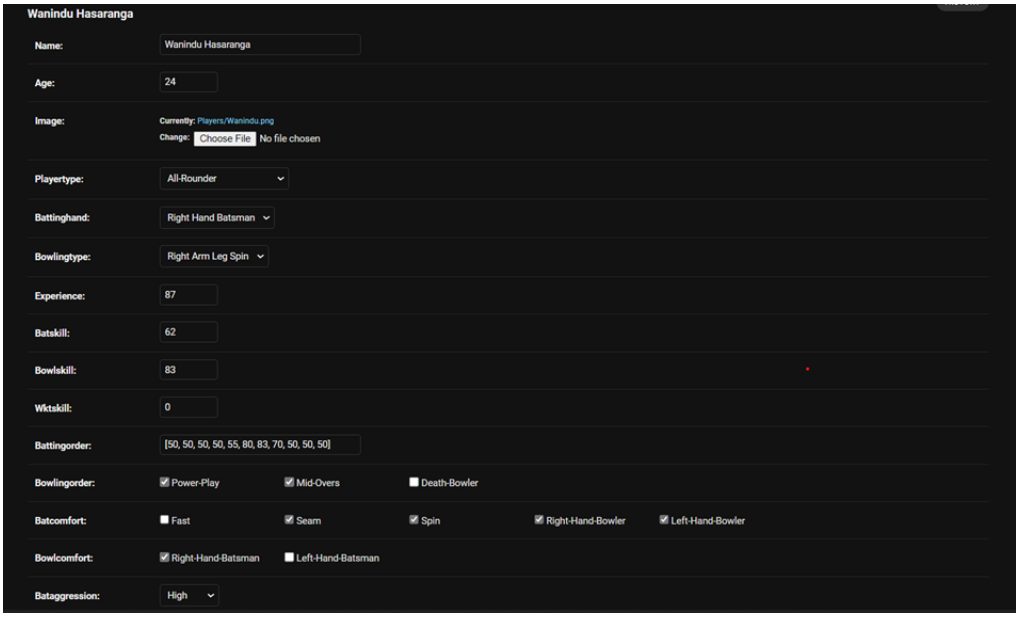


Fig 3.21: Database UI Player Schema

Once we click on any player name from the previous page, we reach onto the page which is particularly dedicated to every player. Here we can edit the entire details of the player. This page consists of various fields such as name, age, image of the player, player type, batting hand, bowling hand, bowling type, experience, bat skill, bowl skill, wicket skill, batting order, bowling order, bat comfort, bowl comfort and bat aggression.

All the mentioned information about the player is very important because it is going to affect his performance. For example, if the player loves batting at number 3 but during the match, we send him at a lower or upper order then his

performance is going to degrade drastically and so eventually the entire team will suffer. Bat comfort means which kind of bowler can that particular player play easily. All these details when combined together can make a player strong or weak. Though we can edit anything we want but in order to make our simulator resemble the real one we did a thorough research and filled every detail of the player genuinely so that we can simulate the real time environment and our audience can get the real excitement during the simulation.

Chapter-4

PERFORMANCE ANALYSIS

Performance analysis is an important aspect of developing accurate and efficient machine learning and deep learning models for predicting the outcome of every ball. The analysis typically involves evaluating the accuracy, speed, and computational efficiency of the models. Performance indicators like precision, recall, and F1 score can be used to gauge accuracy. Recall provides the proportion of correctly predicted outcomes of the ball being bowled in the dataset, whereas precision shows the proportion of correctly predicted outcomes out of all predicted outcomes/scores. A harmonic mean of memory and precision makes up the F1 score.
In addition to accuracy, the analysis may also focus on evaluating the speed and computational efficiency of the models. This can include measuring the training and inference time of the models, as well as the computational resources required for training and deployment. Such an evaluation can help identify hardware and software configurations that are optimal for training and inference.

Precision, recall, and F1 score can all be employed as measurements for accuracy. Recall represents the percentage of accurate forecasts for the target class among all positive predictions, whereas precision measures the percentage of accurate predictions for the target class among all positive actual results. The harmonic mean of recall and precision is the F1 score.

In addition to accuracy, speed and computational efficiency are also essential metrics to evaluate the system's performance. Training and inference time, as well as computational resources required for training and deployment, can be

measured to determine the system's efficiency. Hardware and software configurations can also be evaluated to identify optimal settings for training and inference.

Assessing the system's adaptability to new datasets is crucial as well. Transfer learning can be used to evaluate the model's performance and fine-tune it on additional datasets with various plant classes. To make sure that the model is not overfitting to the training data, cross-validation can also be performed.

Overall, a comprehensive performance analysis can help optimize the predicted result for accurate and efficient plant identification while minimizing computational resources and training time.

1. Accuracy: It can be computed by dividing the number of samples that were successfully categorized by the overall number of samples in the dataset. It reflects the percentage of correctly predicted outcomes. High accuracy indicates that the system can correctly predict different types of results of a ball being bowled.

2. Precision and Recall: Two crucial variables that are frequently employed in classification issues are recall and precision. Recall is the percentage of true positives among all real positive samples, whereas precision measures the percentage of true positives among all positive forecasts. These measurements can be used to assess the system's potential biases and to make performance-enhancing adjustments. Additionally, a single indicator of the system's total effectiveness can be found in the F1 score, which is the harmonic mean of precision and recall. For performance analysis in predicting the score/outcome per ball, other metrics including accuracy, confusion matrix, and receiver operating characteristic (ROC) curve can also be used.

3. F1 Score: The harmonic mean of recall and precision, known as the F1 score, can be used to assess the system's overall performance. It strikes a compromise between recall and precision, and a high F1 score shows that the system is operating effectively. The F1 score is determined by multiplying two by (precision + recall) / (precision + recall).

4. Confusion Matrix: A confusion matrix is a useful tool to evaluate the performance of a classification model. It provides a detailed breakdown of the number of correctly and incorrectly classified samples for each class. This can help identify which classes the system is struggling with and may require additional training or fine-tuning. The matrix is typically represented as a table where each row represents the actual class and each column represents the predicted class. The cells in the table contain the number of samples that belong to a particular class and were classified correctly or incorrectly. We can determine different performance indicators, including accuracy, precision, recall, and F1 score, from the analysis of the confusion matrix, which can aid in further system optimization.

5. Training and Validation Loss: Training accuracy and validation accuracy are two evaluation metrics that may be used to continuously monitor the training process and make sure the system is not overfitting or underfitting (Fig 2.5 shows Graph (Loss) for visualization). When a system performs exceptionally well on training data but poorly on new data, this is a sign of overfitting since it means the system has memorized the training data rather than discovering its underlying patterns. Underfitting, on the other hand, happens when the system is overly straightforward and fails to recognise the key patterns in the data, leading to subpar performance on both the training and validation sets. The performance of the model's generalization can be

enhanced by adjusting the hyperparameters while keeping an eye on these measures.

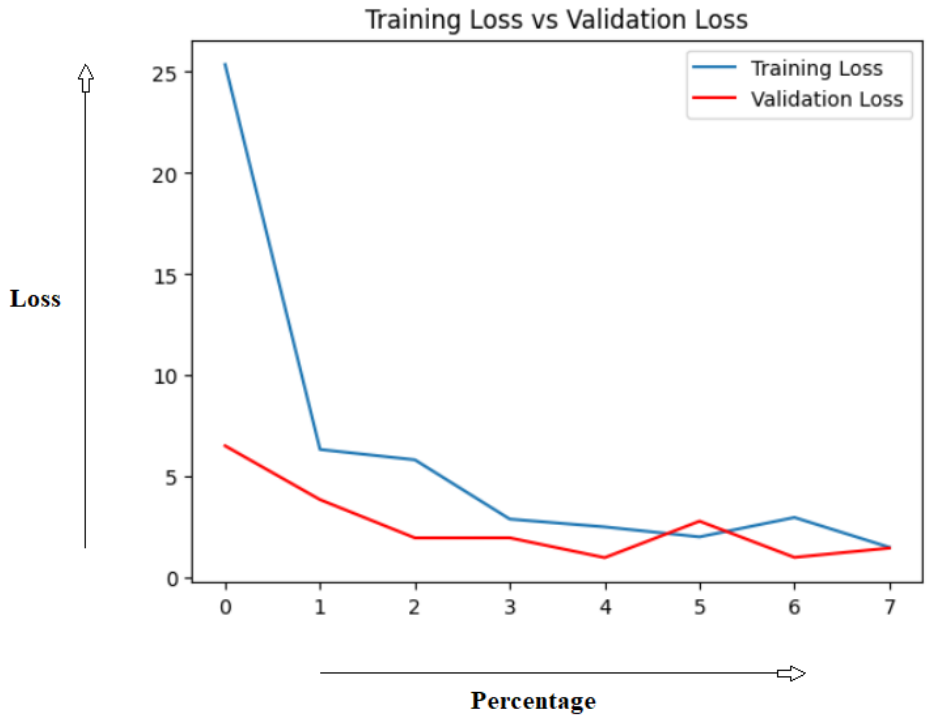


Fig 4.1 : Graph(Loss) for visualization

Overall, it is essential to perform a thorough performance analysis to evaluate the system's ability to accurately classify medicinal plant species using ML & DL techniques. The analysis should include various metrics such as accuracy, precision, recall, F1 score, and a confusion matrix to identify any potential biases or weaknesses in the system. Monitoring the training process is also crucial to ensure that the system is not overfitting or underfitting. A comprehensive performance analysis can help optimize the system for

accurate and efficient plant detection while minimizing computational resources and training time.

```
Epoch 1/50
16/16 [==============================] - ETA: 0s - loss: 25.3637 - accuracy: 0.2578
Epoch 1: val_accuracy improved from -inf to 0.64844, saving model to best_model.h5
16/16 [==============================] - 44s 3s/step - loss: 25.3637 - accuracy: 0.2578 - val_loss: 6.5174 - val_accuracy: 0.6484
Epoch 2/50
16/16 [==============================] - ETA: 0s - loss: 6.3381 - accuracy: 0.7070
Epoch 2: val_accuracy improved from 0.64844 to 0.82422, saving model to best_model.h5
16/16 [==============================] - 39s 3s/step - loss: 6.3381 - accuracy: 0.7070 - val_loss: 3.8570 - val_accuracy: 0.8242
Epoch 3/50
16/16 [==============================] - ETA: 0s - loss: 5.8253 - accuracy: 0.7715
Epoch 3: val_accuracy improved from 0.82422 to 0.86523, saving model to best_model.h5
16/16 [==============================] - 46s 3s/step - loss: 5.8253 - accuracy: 0.7715 - val_loss: 1.9669 - val_accuracy: 0.8652
Epoch 4/50
16/16 [==============================] - ETA: 0s - loss: 2.8958 - accuracy: 0.8281
Epoch 4: val_accuracy improved from 0.86523 to 0.88867, saving model to best_model.h5
16/16 [==============================] - 33s 2s/step - loss: 2.8958 - accuracy: 0.8281 - val_loss: 1.9701 - val_accuracy: 0.8887
Epoch 5/50
16/16 [==============================] - ETA: 0s - loss: 2.5106 - accuracy: 0.8535
Epoch 5: val_accuracy improved from 0.88867 to 0.94141, saving model to best_model.h5
16/16 [==============================] - 39s 3s/step - loss: 2.5106 - accuracy: 0.8535 - val_loss: 0.9900 - val_accuracy: 0.9414
Epoch 6/50
16/16 [==============================] - ETA: 0s - loss: 2.0167 - accuracy: 0.8848
Epoch 6: val_accuracy did not improve from 0.94141
16/16 [==============================] - 39s 3s/step - loss: 2.0167 - accuracy: 0.8848 - val_loss: 2.7927 - val_accuracy: 0.8691
Epoch 7/50
16/16 [==============================] - ETA: 0s - loss: 2.9767 - accuracy: 0.8633
Epoch 7: val_accuracy did not improve from 0.94141
16/16 [==============================] - 39s 2s/step - loss: 2.9767 - accuracy: 0.8633 - val_loss: 1.0038 - val_accuracy: 0.9258
Epoch 8/50
16/16 [==============================] - ETA: 0s - loss: 1.5036 - accuracy: 0.9102
Epoch 8: val_accuracy did not improve from 0.94141
16/16 [==============================] - 38s 2s/step - loss: 1.5036 - accuracy: 0.9102 - val_loss: 1.4638 - val_accuracy: 0.9355
Epoch 8: early stopping
```

Fig 4.2 : Showing Early stopping of model.

4.1. UI/UX – ANALYSIS

4.1.1. HOME PAGE

Fig 4.3: UI - Home Page

This is the home page of our project. In the right section it has an animation which lists the charming colors of red and blue. Our team's name (D Force) is also displayed along with the animation of a batsman.

The left side consists of various sections:

· **View-Players:** We can see the entire list of players present in the database. We can also list the players present in the respective team. We would also be able to see the statistics of every player, their strengths and weaknesses, etc.

· **Start Tournament:** Once we have two or more teams we can create a tournament between those teams, we would be able to select the number of matches per team along with the selection of opponent per team for every match. After the creation of the tournament, we would be shown the entire roadmap of every team which would be used later to simulate their game for

every opponent. At the end of every match, we would be shown the points table which will help every team to see their standings and the number of matches they need to win in order to qualify for the semi-finals and finals.

· **Start Match:** After the creation of tournament matches / at least two teams, we can visit this section to simulate a match between two teams. 11 players should be selected by both the teams along with the selection of the captain, wicketkeeper, batting and bowling preferences. Once we set up the playing 11, we head towards toss. The team which wins the toss gets to choose between batting and bowling. After the entire set up we can start the match and see the live action ball by ball. Runs scored, wickets fall, strike rate of the batsman and much more data will be displayed. At the end of both the innings we would be shown the match summary which can be used to improve the team.

· **Add Team:** This section will help the users / managers to create a team from the pool of remaining players which aren't already present in any other team. Creation of a team can be done either by not paying any fees or by participating in the virtual auction, where every team manager would be provided with a fixed virtual amount which they can use to purchase the players of their choice. All the managers should come up with proper strategies as it is said if you win the auction, you have won half of the tournament. Every single player can affect the performance of the team. Therefore, the auction part becomes very important in order to win the matches as well as the tournament.

· **Refresh:** This button is useful if somehow we experience any glitch in the webpage or our webpage starts functioning abnormally.

4.1.2. START MATCH PAGE

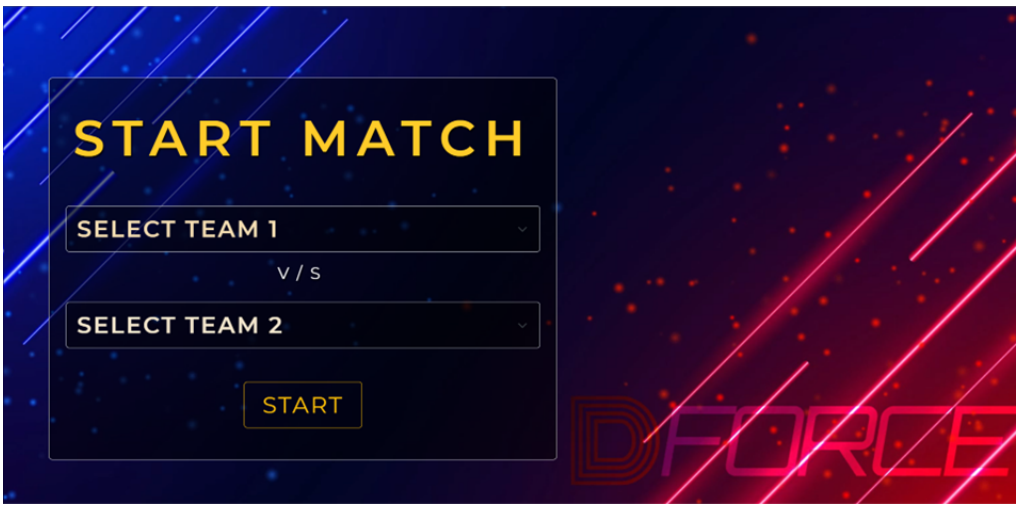


Fig 4.4: UI - Start Match Page

The image shown above is a part of the start match section from the home page. We can select the two teams which we had created in the add team section and head towards the toss after the selection of playing 11, captain, wicketkeeper, bowling and batting preferences. Once we are done with toss, the team winning the toss gets to choose from the two, batting and fielding.

Finally, we can simulate the match between two teams and see the live action ball by ball. Runs scored, wickets fall, strike rate of the batsman and much more data will be displayed. At the end of both the innings we would be shown the match summary which can be used to improve the team.

4.1.3. SET BATTING LINEUP

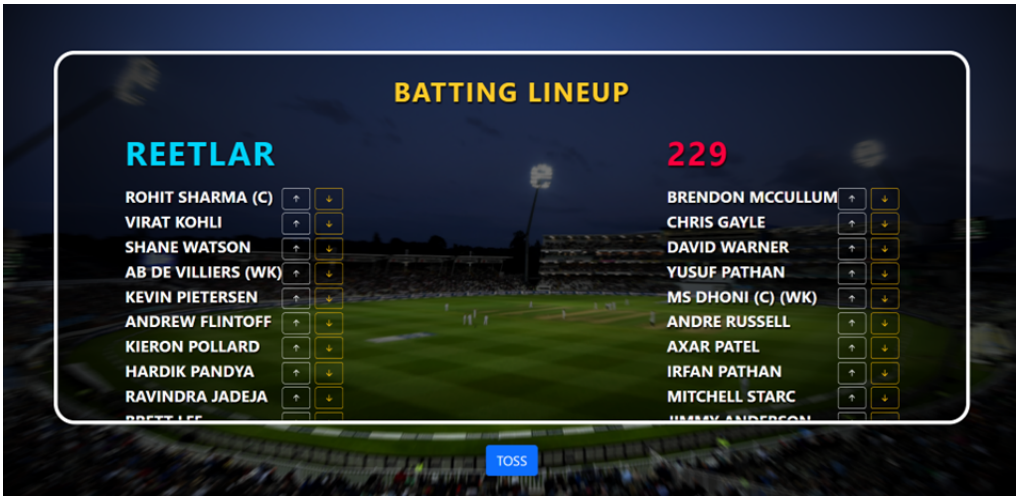


Fig 4.5: UI - Batting Lineup Page

The image shown above shows the batting lineup page which is a part of the Start Match section. This particular webpage is shown once we have selected the playing 11 along with the captain and wicketkeeper.

Here we can change the batting order. The person whose name comes first in the batting order list will get to bat first. The two arrows after the name of the player are used for this specific purpose, i.e., to change the lineup. This is the last time when we can make any changes in our team. After this we head towards the toss and finally after the toss, we can simulate the match between the two teams.

4.1.4. SIMULATION PAGE

Fig 4.6: UI - Simulation of Match

Once we are done with toss, we are directed to this page which shows the live simulation of the match. On top right we can see the picture of two batsmen playing along with the runs scored and total balls faced by them.

At the bottom right we can see the image of the bowler who is currently bowling. Total runs given by him as well as the total wickets taken by him is also shown. At the end of the image, we are shown the current ball of the over.

On top left we are shown various information such as the name of the team currently batting, their total score along with the total wickets fallen till then. There is a section named per ball where we can see the entire settings for every ball being bowled.

4.1.5. SETUP TEAM

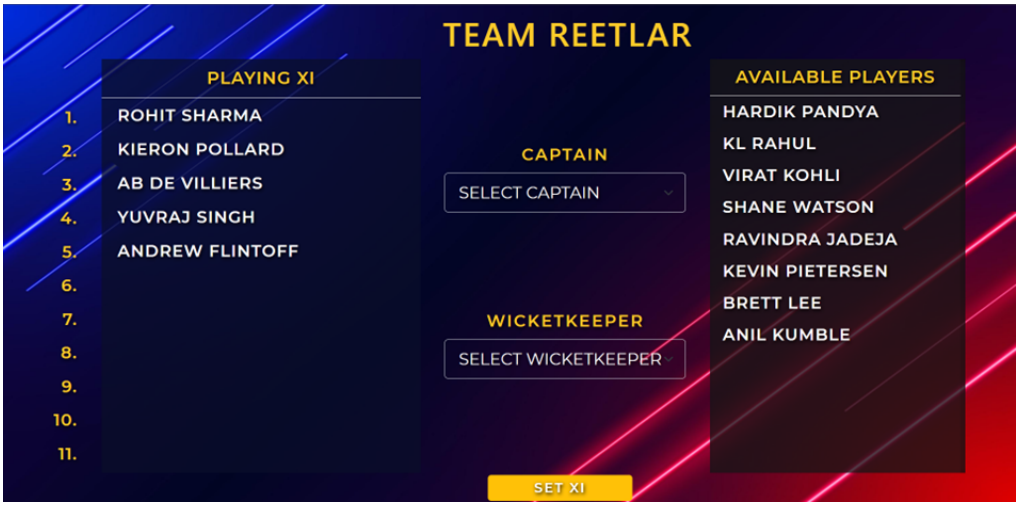


Fig 4.7: UI - Team Setup

The Set XI page is also a part of the Start Match section. Here we can select our playing 11 from the list of available players. One can simply drag and drop the player from the Available Players section to the Playing XI section.

This is how we get to choose our playing 11. Once we have selected 11 players from the pool of available players, we have to choose the captain and wicket keeper for the team.

The selection of captain and wicketkeeper should also be done wisely as their captaincy and wicket keeping skills will matter a lot during the match. Once we are done with the entire selection, we have to click on SET XI to finalize the team.

4.1.6. SUMMARY OF THE MATCH

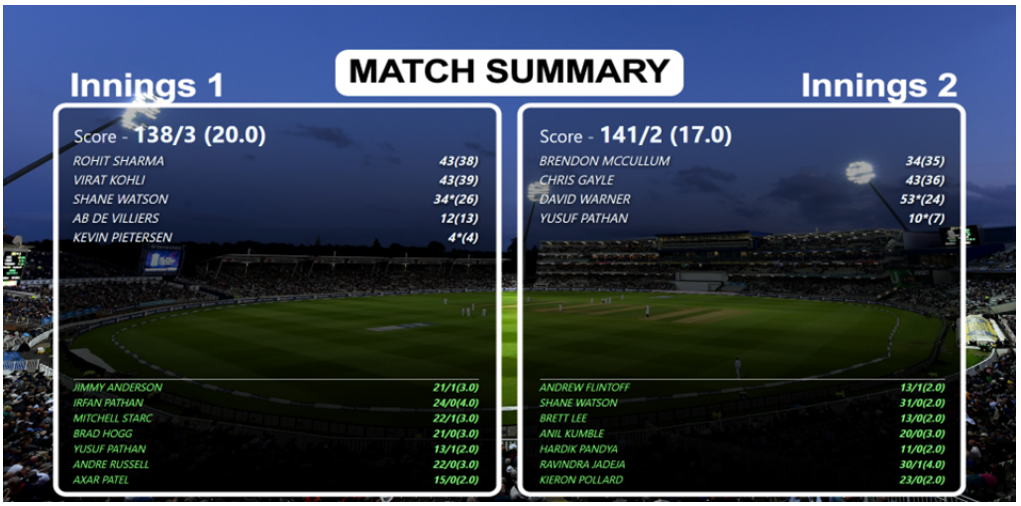


Fig 4.8: UI - Match Summary

At the end of both the innings the entire match summary of both the innings is shown. We can see individual scores and balls faced by them. We are also shown the performance of every bowler along with total runs conceived, total wickets taken and number of overs bowled by them.

The match summary can be very useful if studied properly as it can help to improve the team for upcoming matches. It can also suggest the correct batting order in order to acquire maximum results from every player.

4.2. OTHER REQUIREMENTS

4.2.1. SYSTEM REQUIREMENTS

- 3.3 GHz Processor
- Ryzen 9 5500 HS
- 1 TB Solid State Drive

- 16 GB DDR5 RAM
- Ubuntu Operating System
- RTX 3050i Graphic Card
- 6 Core
- 1920 x 1200 Resolution
- At Least 16-bit
- At Least 64 mb of video memory
- MediaTek MT7921 Wi-Fi 6 802.11ax PCIe Adapter

4.2.2. NON-FUNCTIONAL REQUIREMENTS

- **Reliability**

The system must have a reliable and robust structure to provide its functionalities. When a customer requests changes, the system should clearly indicate these changes. Any modifications made by the programmer must be approved by the project leader and the test designer.

- **Maintainability**

The system's monitoring and maintenance should be simple and straightforward in its approach. It is essential to avoid running too many jobs on different machines, which can make it difficult to monitor if the jobs are running without any errors.

- **Performance**

The system will be used by multiple employees simultaneously. As the system will be hosted on a single web server with a single database server in the background, performance becomes a significant concern. The system should not crash when multiple users use it simultaneously. It should allow quick accessibility to all its users. For example, if two test specialists are simultaneously trying to report the presence of a bug, there should be no inconsistency during the process.

- **Portability**

The system should be easily transferable to another server in case the current web server encounters technical issues or malfunctions. This will require the system to be easily portable and transferable to another server without much downtime or data loss.

- **Scalability**

The system should be flexible enough to incorporate additional functionalities in the future. A common interface should be available that can accommodate the new features.

- **Flexibility**

Flexibility refers to a system's ability to adapt to varying situations, circumstances, and modifications in business policies and procedures. An adaptable system is easy to reconfigure or modify based on different user and system requirements. To enhance adaptability, the system should have a deliberate separation of concerns between its management and engine components. This approach ensures that only a small portion of the system is affected when there are changes in policies or rules.

1.1.2. PRODUCT REQUIREMENTS

- **Correctness**

The system adhered to a clearly defined set of procedures and guidelines to initiate a dialogue with the user, and utilized a pre-trained classification model to process the information. In addition, comprehensive testing was conducted to ensure the accuracy of the data.

- **Modularity**

The product is divided into multiple modules, each with clearly defined interfaces to enhance its flexibility. The software is optimized for overall performance to provide users with accurate and relevant results in a timely manner. Additionally, nonfunctional requirements, also known as system qualities, are considered in the development process.

- **Basic Operational**

The eight primary functions of systems engineering are performed by customers, with emphasis on the operator as the key customer. The operational requirements define the basic needs and should include the following points at a minimum:-

Mission profile or scenario: This describes the procedures required to achieve the mission objectives and determines the effectiveness and efficiency of the system.

Performance and related parameters: This identifies the critical system parameters needed to accomplish the mission.

Utilization environments: This provides an overview of the system usage and determines the appropriate environments for effective system operation.

Operational life cycle: This defines the system lifetime, including the design, development, deployment, and retirement phases.

Chapter-5

CONCLUSIONS

5.1 Conclusions

In today's world people cannot go to play cricket or watch any live cricket match whenever they want. This is due to their busy schedules and low availability of cricket grounds. In order to provide the real excitement and experience of a cricket match we have built the Crick-IT software which is a combination of a fantasy cricket simulator and cricket auction. The best part of the software is that it is user friendly and can be accessed by even those who have a very little knowledge about computers. We have built the entire software in such a way that once you start using it there is no going back and you will get addicted to it. It is pretty close to resembling the feel of an actual auction and match. It has already been used by most of our friends and by seeing the thrill and excitement in their eyes we have a strong belief that we have achieved what we always wished for this project.

The fantasy games which are already available in the market give us the option to simulate only the live matches and create our team only according to those live matches. But the Crick-IT software provides us the flexibility to create teams according to our choice. We can select any player from any country and put them in our team. It is quite similar to the IPL.

Now if we particularly talk about our project, it has got everything a cricket enthusiast needs. It has got all the thrill and excitement. We can add new teams or modify the existing ones. We can add as well as edit existing players. We can even perform a live auction. Most importantly we can simulate live matches. The match summary which is displayed at the end of every match can be used to improve the team and plan our strategies accordingly.
Our software has endless possibilities and usage. For example, the managers of different franchises can use the analytics to create teams according to their choice and simulate matches in order to predict the correct playing 11 for the actual real game.

5.2 Future Scope

In the near future we are planning to make Crick-IT a cross platform application. We would be adding a video simulation which will display the live in action of the entire match ball by ball. We would be able to see the players scoring runs, wickets falling, and everything as if we were watching a live cricket match.

At the end of the match, we would also be able to see the match highlights which will include all the major details of the entire match. We are also aiming at improving the graphics. With the use of machine learning and artificial intelligence, the prediction of the simulator would be improved. Insights of every player will also be provided so that every player would get to know their weaknesses and would eventually work upon it in order to improve his performance.

We are also planning to add a virtual cricket game with high end graphics. With the introduction of add, we may also generate revenue.

REFERENCES

A) Conferences and Conference Proceedings

[1] Mazhar Javed Awan, Syed Arbaz Haider Gilani, Hamza Ramzan, Haitham Nobanee, Awais Yasin, Azlan Mohd Zain, and Rabia Javed, "Cricket Match Analytics Using the Big Data Approach" Oxford Centre for Islamic Studies, University of Oxford, Marston Road, Headington, Oxford OX3 0EE, UK 2021, 10, 2350

[2] Mr. Suyash Mahajan, Ms. Gunjan Kandhari, Ms. Salma Shaikh, Ms. Rutuja Pawar, Mr. Jash Vora, and Ms. A. R. Deshpande, "Cricket Analytics and Predictor" Walchand Institute of Technology, Solapur, India 2020

[3] T. B. Swartz, P. S. Gill, D. Beaudoin, and B. M. Desilva, Optimal batting orders in one - day cricket, Computers & Operations Research, vol.33, no.7, pp.1939 – 1950, 2006

[4] L.J.P. van der Maaten, E.O. Postma, H.J. van den Herik, "Dimensionality Reduction: A Comparative Review" MICC, Maastricht University, P.O. Box 616, 6200 MD Maastricht, The Netherlands

B) Journals/periodicals

[1] Geddam Jaishankar Harshit and Rajkumar S, ―A Review Paper on Cricket Predictions Using Various Machine Learning Algorithms and Comparisons Among Them, ‖ International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol. 45, no. 98, pp. 27 –32.

APPENDICES

Code:

Cricket_Auction_League > urls.py

```python
"""
from django.contrib import admin
from django.urls import path, include
from django.urls import re_path as url
from django.contrib.staticfiles.urls import static
from django.contrib.staticfiles.urls import staticfiles_urlpatterns
from django.conf import settings

admin.site.site_header = "DForce CRICKET SIMULATOR"
admin.site.site_title = "DForce Admin Portal"
admin.site.index_title = "Welcome to DForce Cricket Simulator"

urlpatterns = [
    url('', include('Home.urls')),
    url('players/', include('Players.urls')),
    url('team/', include('Teams.urls')),
    url('toss/', include('Toss.urls')),
    url('match/', include('Match.urls')),
    url('tournament/', include('Tournament.urls')),
    path('admin/', admin.site.urls),
]
urlpatterns += staticfiles_urlpatterns()
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

```python
def nextInnings(self):
    self.target = self.totalRuns
    self.perOverRuns = {}
    self.perBallResults = {}
    self.totalRuns = 0
    self.wickets = 0
    self.balls = 0
    self.requiredRunRate = 0
    self.currentRunRate = 0
    self.frequency = {}  # testing
    self.bfrequency = {} # testing
    self.extras = {"wd": 0, "nb": 0, "total": 0}
    self.boundaries = {6:0, 4:0, 3:0, 2:0, 1:0, 0:0}

    self.battingTeam, self.bowlingTeam = self.bowlingTeam, self.battingTeam
    self.battingLineup = self.battingTeam.battingLineup
    self.bowlingLineup = self.bowlingTeam.bowlingLineup

    self.striker = self.battingLineup.pop(0)
    self.battingTeam.batsmanBin.append(self.striker)
    self.nonStriker = self.battingLineup.pop(0)
    self.battingTeam.batsmanBin.append(self.nonStriker)
    self.bowler = self.bowlingLineup.pop(0)
    self.bowlingTeam.bowlerBin.append(self.bowler)

    self.alreadyBowled = []
    self.alreadyBowled.append(self.bowler.name)
    self.alreadyBat = []

    self.win = {"bowler":0, "batsman":0}
    self.batComeback = False
```

```
def updateRunRate(self):
    try:
        self.requiredRunRate = ((self.target-self.totalRuns)/(120-self.balls))*6
    except:
        self.requiredRunRate = (self.target-self.totalRuns)
    if self.innings == 1:
        try:
            self.actualReqRunRate = ((self.ground.averageRuns-self.totalRuns)/(120-self.balls))*6
        except:
            self.actualReqRunRate = (self.ground.averageRuns-self.totalRuns)
    else:
        self.actualReqRunRate = self.requiredRunRate
    try:
        self.currentRunRate = (self.totalRuns/self.balls)*6
    except:
        self.currentRunRate = 0

def startMatch(self):  # this function is the main function
    self.updateRunRate()
    self.setupPlayers()  # setting up some values which need to be set when player comes
    for over in range(self.overs):
        if self.target < self.totalRuns and self.innings != 1:
            break
        if self.wickets == 10:
            break
        ball = 1
        prevOverRuns = self.totalRuns  # testing
        prevOverWickets = self.wickets  # testing
        while ball != 7:
            self.updateProjectedScore() # this will only run after 10 and 15 overs.   *************NEW****
            if self.target < self.totalRuns and self.innings != 1:
```

```
# innings 2
print("\n\nInnings 2:\n")
target = total
self.battingTeam, self.bowlingTeam = self.bowlingTeam, self.battingTeam
tempbatline = self.selectTop3Batsman()
striker = tempbatline[0]
nonStriker = tempbatline[1]
bowler = self.selectTopBowler()
total = 0
wickets = 0
for i in range(6):
    result = random.choice([0, 1, 2, 3, 4, 6, "w"])
    if result == "w":
        wickets += 1
        striker = tempbatline[2]
    else:
        if result in [1, 3]:
            striker, nonStriker = nonStriker, striker
        total += result

    print("Result:", result)
    print("Striker:",striker, " Non-Striker:", nonStriker, " Bowler:", bowler)
    print("Score:", total, "/", wickets)
    perball2[i+1] = {
    "score":str(total)+"/"+str(wickets),
    "striker":{"name": striker.name, "runs": striker.runs, "balls": striker.ballsFaced},
    "nonstriker":{"name": nonStriker.name, "runs": nonStriker.runs, "balls": nonStriker.ballsFaced},
    "bowler":{"name": bowler.name, "runs": bowler.runsGiven, "balls": bowler.ballsBowled, "wickets": bo
    "result": result}
    if wickets == 2:
        break
```

```python
def nextOver(self):
    self.striker, self.nonStriker = self.nonStriker, self.striker  # changing strike
    self.bowler = self.bowlingLineup.pop(0)  # next bowler
    if self.bowler.name not in self.alreadyBowled:
        self.alreadyBowled.append(self.bowler.name)
        self.bowlingTeam.bowlerBin.append(self.bowler)
        self.setupNewBowler()
    else:
        crrOver = int(self.balls / 6)
        self.bowler.overWiseComfort(crrOver)
        self.bowler.updateBowlComfort(self.striker)
    self.striker.updateBatComfort(self.bowler)
    self.nonStriker.updateBatComfort(self.bowler)

def setupNewBowler(self):
    # for stamina:
    self.bowler.setInitialStamina()
    # for comfort:
    if self.bowlingTeam.homeGround.name == self.ground.name:
        homeGround = True
    else:
        homeGround = False
    crrOver = int(self.balls / 6) + 1
    self.bowler.setInitialBowlComfort(crrOver, homeGround)
    self.bowler.updateBowlComfort(self.striker)
    # for pressure:
    self.bowler.setInitialPressure()
    # for confidence:
    self.bowler.setBowlerConfidence()
```

```python
def batPowerPlay(self, relativeDiff, agg):
    result = 0

    if agg == "super":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 1, 1, 2, 4])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 2, 2, 4, 4])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 2, 4, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 2, 4, 4, 6])
        elif 10 < relativeDiff <= 15:
            result = random.choice([1, 2, 4, 6])
        elif 15 < relativeDiff <= 20:
            result = random.choice([2, 4, 4, 6])
        elif 20 < relativeDiff <= 25:
            result = random.choice([2, 4, 4, 4, 1, 4, 6, 6, 4])
        else:
            result = random.choice([2, 4, 4, 4, 2, 1, 6, 6, 6, 6, 4])

    elif agg == "high":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 0, 1, 1, 1, 2])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 1, 2, 2, 4])
        elif 5 < relativeDiff <= 9:
```

```
    elif agg == "normal":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 0, 1, 1, 1, 2])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 0, 1, 1, 2, 2, 3, 4])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 2, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 2, 2, 4, 6])
        elif 10 < relativeDiff <= 15:
            result = random.choice([0, 1, 2, 2, 4, 6])
        elif 15 < relativeDiff <= 20:
            result = random.choice([1, 2, 4, 4, 6])
        elif 20 < relativeDiff <= 25:
            result = random.choice([2, 4, 4, 2, 1, 4, 6, 4])
        else:
            result = random.choice([2, 4, 4, 2, 1, 6, 6, 4])

    elif agg == "low":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 0, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 0, 1, 1, 1, 2])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 0, 1, 1, 2, 2, 3])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 2, 2, 1, 1, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 2, 2, 2, 4])
```

```
def batMiddle(self, relativeDiff, agg):
    result = 0
    if agg == "super":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 1, 1, 2, 3])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 2, 2, 4])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 2, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 2, 4, 6])
        elif 10 < relativeDiff <= 15:
            result = random.choice([0, 2, 2, 4, 4, 6])
        elif 15 < relativeDiff <= 20:
            result = random.choice([1, 2, 2, 4, 4, 6, 6])
        elif 20 < relativeDiff <= 25:
            result = random.choice([2, 4, 4, 6])
        elif 25 < relativeDiff <= 30:
            result = random.choice([2, 4, 4, 4, 2, 1, 6, 4, 6, 4])
        else:
            result = random.choice([2, 4, 4, 4, 2, 1, 6, 6, 6, 6, 4])

    elif agg == "high":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 1, 1, 2, 3])
        elif 2 < relativeDiff <= 5:
```

```
    elif agg == "normal":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 0, 1, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 1, 1, 2, 2, 3])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 2, 2, 1, 2])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 1, 2, 2, 2, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 2, 3, 4])
        elif 10 < relativeDiff <= 15:
            result = random.choice([0, 1, 2, 2, 4, 4])
        elif 15 < relativeDiff <= 20:
            result = random.choice([1, 2, 2, 2, 4, 4, 6])
        elif 20 < relativeDiff <= 25:
            result = random.choice([1, 2, 2, 4, 4, 6])
        elif 25 < relativeDiff <= 30:
            result = random.choice([2, 4, 4, 2, 1, 4, 6, 4])
        else:
            result = random.choice([2, 4, 4, 2, 1, 6, 6, 4])

    elif agg == "low":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 1, 1, 1, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 1, 1, 2, 2, 2, 3])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 2, 2])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 1, 2, 2, 2])
```

```
def batDeath(self, relativeDiff, agg):
    result = 0
    if agg == "super":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 2, 4])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 2, 2, 2, 4, 6])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 4, 6])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 4, 4, 6])
        elif 10 < relativeDiff <= 15:
            result = random.choice([0, 2, 4, 4, 6, 4, 4, 6])
        else:
            result = random.choice([2, 4, 4, 4, 2, 6, 6, 6, 6, 4])

    elif agg == "high":
        if 0 < relativeDiff <= 1:
            result = random.choice([0, 0, 1, 2])
        elif 1 < relativeDiff <= 2:
            result = random.choice([0, 1, 2, 2, 3])
        elif 2 < relativeDiff <= 5:
            result = random.choice([0, 1, 1, 2, 2, 2, 4, 6])
        elif 5 < relativeDiff <= 9:
            result = random.choice([0, 1, 2, 4])
        elif 9 < relativeDiff <= 10:
            result = random.choice([0, 1, 2, 4, 4, 6])
        elif 10 < relativeDiff <= 15:
            result = random.choice([0, 4, 1, 2, 6, 2, 4, 4, 6])
```

Linear Regression:

Linear regression is a statistical technique that is used to study the relationship between a dependent variable and one or more independent variables. It is a widely used technique in data analysis and predictive modeling.

The basic idea of linear regression is to fit a straight line through a set of data points in order to describe the relationship between the dependent variable and

one or more independent variables. The line is determined by estimating the slope and intercept of the line that best fits the data.

The equation for a simple linear regression model with one independent variable can be written as:

$y = \beta_0 + \beta_1 x + \varepsilon$

where:

- y is the dependent variable
- x is the independent variable
- β_0 is the intercept (the value of y when x = 0)
- β_1 is the slope (the change in y for a one-unit change in x)
- ε is the error term (the difference between the predicted value of y and the actual value of y)

In multiple linear regression models with more than one independent variable, the equation becomes:

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_r x_r + \varepsilon$

where:

- y is the dependent variable
- $x_1, x_2, ..., x_r$ are the independent variables
- β_0 is the intercept
- $\beta_1, \beta_2, ..., \beta_r$ are the slopes for each independent variable
- ε is the error term

The objective of linear regression is to estimate the values of $\beta_0, \beta_1, \beta_2, ..., \beta_r$ that minimize the sum of the squared errors between the predicted and actual values of y. This is called the least squares method.

Once the model has been estimated, it can be used to make predictions by plugging in values for the independent variables. The model can also be used to test hypotheses about the relationship between the dependent variable and the independent variables, and to determine the strength and direction of the relationship.

There are several assumptions that must be met in order for linear regression to be valid, including linearity, independence of errors, normality of errors, and equal variance of errors. Violations of these assumptions can lead to biased estimates and incorrect conclusions.

Linear regression can be performed using a variety of software packages, including Excel, R, Python, and SAS. In addition to simple and multiple linear regression, there are also more advanced techniques such as polynomial regression, ridge regression, and logistic regression that can be used to model more complex relationships between variables.