

Title : Cricket Manager

Submitted in partial fulfilment of the requirements

of the degree of

Bachelor of Engineering in
Artificial Intelligence and Data Science

by

Arunim Chakraborty(7)

Kshitij Shidore(51)

Rupesh Dhirwani(10)

Siddhant Dongre(12)

under the guidance of

Supervisor (s):

Dr. Anjali Yeole



Department of Artificial Intelligence and Data Science



Vivekanand Education Society's

Institute of Technology

(Affiliated to University of Mumbai, Approved by AICTE & Recognized by Govt. of Maharashtra)

Department of Artificial Intelligence and Data Science

CERTIFICATE

This is to certify that **Mr Arunim Chakraborty, Mr Kshitij Shidore, Mr Rupesh Dhirwani and Mr Siddhant Dongre** of Second Year of Artificial Intelligence and Data Science studying under the University of Mumbai have satisfactorily presented the Mini Project entitled **Cricket Manager** as a part of the MINI-PROJECT for Semester-III under the guidance of **Dr. Anjali Yeole** in the year 2021-2022.

Date: 18/12/2021

(Name and sign)
Head of Department

(Name and sign)
Supervisor/Guide



Vivekanand Education Society's

Institute of Technology

(Affiliated to University of Mumbai, Approved by AICTE & Recognized by Govt. of Maharashtra)

Department of Artificial Intelligence and Data Science

DECLARATION

We, **Arunim Chakraborty, Kshitij Shidore, Rupesh Dhirwani and Siddhant Dongre** from **D6AD**, declare that this project represents our ideas in our own words without plagiarism and wherever others' ideas or words have been included, we have adequately cited and referenced the original sources.

We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our project work.

We declare that we have maintained a minimum 75% attendance, as per the University of Mumbai norms.

We understand that any violation of the above will be cause for disciplinary action by the Institute.

Yours Faithfully

1. _____ Arunim Chakraborty_16/12/21_____

2. _____ Kshitij Shidore_16/12/21_____

3. _____ Rupesh Dhirwani_16/12/21_____

4. _____ Siddhant Dongre_16/12/21_____



Vivekanand Education Society's

Institute of Technology

(Affiliated to University of Mumbai, Approved by AICTE & Recognized by Govt. of Maharashtra)

Table of Contents

Abstract

List of Tables

List of Figures

1. Introduction	Pg No
1.1. Introduction	5
1.2. Problem Statement	5
1.3. Objectives	5
1.4. Scope	5
2. Literature Survey	6
2.1 Literature/Techniques studied	
2.2 Papers/Findings	
3. Analysis and Design	7-11
3.1 Analysis of the system	
3.2 Algorithm	11 -14
3.3 Design of the proposed system	14 -16
4. Project Sample	16 -17
5. Future Work	17
5.1 Small Nuances	18
6. Conclusion	18
References	18

Introduction

Cricket is the second most popular sport in the world. Given its popularity, it's no surprise that it is also one of those topics upon which numerous games are created. However, there is a severe lack in the simulation part of the pack and an even bigger deficit where realism of the simulations is involved.

This is where our game, Cricket Manager comes in. It is a game that gives the player a sense of environment in which he/she can make and manage his/her own cricket team where a player will be able to handle a team in a popular sports environment, granting full control over finances, tactics, and backroom management. It allows the player to manage his/her own virtual cricket team against an AI as a competitor virtually.

Problem Statement

To create a Cricket Manager game which is able to simulate the realistic aspect of the sport,

Objectives

- Create a backend code that is able to simulate the various intricacies of the sport
- Creating an AI that will respond to the User's tactics almost like a real opposition captain.
- Creating a database of all players from all teams with the different attributes.
- Creating an Interactive GUI using which users can select their tactics and play the game
- Implementing the graphics part of the game for example the cricket field, fielders, and balls, etc.

Scope

The game itself is highly strategic and thus requires a lot of logical thinking and strategizing on the player's part. This provides users with a platform to showcase their strategic ideas and critical thinking.

Literature Survey

One of Cricket Manager's biggest strength has always been its realistic outcomes. We have obtained ball by ball data of the entire IPL 2016

(FIG 1)KAGGLE DATA BASE

	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	bowling_over	ball	batsman	non_striker	bowler	is_super	wide_runs	bye_runs	legbye_runs	noball_runs	penalty_runs	batsman_runs	extra_runs	total_runs	player_dism	dismissal	fielder		
2	Royal Chi	1	1	DA Warner	S Dhawa	TS Mills	0	0	0	0	0	0	0	0	0	0			
3	Royal Chi	1	2	DA Warner	S Dhawa	TS Mills	0	0	0	0	0	0	0	0	0	0			
4	Royal Chi	1	3	DA Warner	S Dhawa	TS Mills	0	0	0	0	0	0	4	0	4				
5	Royal Chi	1	4	DA Warner	S Dhawa	TS Mills	0	0	0	0	0	0	0	0	0				
6	Royal Chi	1	5	DA Warner	S Dhawa	TS Mills	0	2	0	0	0	0	0	2	2				
7	Royal Chi	1	6	S Dhawan	DA Warn	TS Mills	0	0	0	0	0	0	0	0	0				
8	Royal Chi	1	7	S Dhawan	DA Warn	TS Mills	0	0	0	1	0	0	0	1	1				
9	Royal Chi	2	1	S Dhawan	DA Warn	A Choudh	0	0	0	0	0	0	1	0	1				
10	Royal Chi	2	2	DA Warner	S Dhawa	A Choudh	0	0	0	0	0	0	4	0	4				
11	Royal Chi	2	3	DA Warner	S Dhawa	A Choudh	0	0	0	0	1	0	0	1	1				
12	Royal Chi	2	4	DA Warner	S Dhawa	A Choudh	0	0	0	0	0	0	6	0	6				
13	Royal Chi	2	5	DA Warner	S Dhawa	A Choudh	0	0	0	0	0	0	0	0	0	DA Warner	caught	Mandeep Singh	
14	Royal Chi	2	6	MC Henriq	S Dhawa	A Choudh	0	0	0	0	0	0	0	0	0				
15	Royal Chi	2	7	MC Henriq	S Dhawa	A Choudh	0	0	0	0	0	0	4	0	4				
16	Royal Chi	3	1	S Dhawan	MC Henri	TS Mills	0	0	0	0	0	0	1	0	1				
17	Royal Chi	3	2	MC Henriq	S Dhawa	TS Mills	0	0	0	0	0	0	0	0	0				
18	Royal Chi	3	3	MC Henriq	S Dhawa	TS Mills	0	0	0	0	0	0	0	0	0				
19	Royal Chi	3	4	MC Henriq	S Dhawa	TS Mills	0	0	0	0	0	0	3	0	3				
20	Royal Chi	3	5	S Dhawan	MC Henri	TS Mills	0	0	0	0	0	0	1	0	1				
21	Royal Chi	3	6	MC Henriq	S Dhawa	TS Mills	0	0	0	0	0	0	1	0	1				
22	Royal Chi	4	1	MC Henriq	S Dhawa	YS Chaha	0	0	0	0	0	0	0	0	0				
23	Royal Chi	4	2	MC Henriq	S Dhawa	YS Chaha	0	0	0	0	0	0	1	0	1				
24	Royal Chi	4	3	S Dhawan	MC Henri	YS Chaha	0	0	0	0	0	0	0	0	0				
25	Royal Chi	4	4	S Dhawan	MC Henri	YS Chaha	0	0	0	0	0	0	1	0	1				
26	Royal Chi	4	5	MC Henriq	S Dhawa	YS Chaha	0	0	0	0	0	0	1	0	1				
27	Royal Chi	4	6	S Dhawan	MC Henri	YS Chaha	0	0	0	0	0	0	1	0	1				
28	Royal Chi	5	1	S Dhawan	MC Henri	S Aravind	0	0	0	0	0	0	1	0	1				

Analyzing this entire data, we obtained some valuable insights into how we can make our game hyper-realistic.

Despite the fact that our game is cricket based, a game from another completely different sport has been a constant inspiration. The game is Football Manager. Football Manager is yet another game that mirrors the real football game on a giant scale. The level of details, complexity and AI – part in it is beyond the scope of our project. But it has laid some very basic foundations for our project.

Our aim is to add more details into the game such as AI that would compete against the player, also we will look to add more data-base into the game. Simultaneously, there would be more options other than aggression, defensive playing of the game.

There we would also be adding finance (to purchase players from the points), tactics and strategy (for the outcomes that has to be decided by the player against the Artificial Intelligence), Player Health (an important aspect that has to be considered for the player's fitness).

We have also studied the conventional cricket games such as EA CRICKET 7, DON BRADMAN CRICKET and also the REAL SPORT, TOP ELEVEN (Cricket Manager). Also we have studied their variable declaring skills such as batsman and bowler skill and how they work with the backend, we have also studied various cricket data sets from KAGGLE website.

Analysis and Design

(FIG 2) PROBABILITIES OF DIFFERENT OUTCOMES

	freq	runs	prob
0run	61148		0.40653131
1run	55497	55497	0.36896167
2run	9705	19410	0.06452192
3run	509	1527	0.00338399
4run	17032	68128	0.11323414
6run	6523	39138	0.04336697
total	150414	183700	1
dismissals	7438		0.04945018

As can be seen from the above picture, we recorded the frequency of each event from the excel data we had obtained and calculated the probability of each event. We then created variable step ladder which basically relates between which values of dice variable, each outcome should occur.

(FIG 3) RANGES FOR THE DICE VALUE

runs	high	low
0	0	-137
1	124	1
2	146	125
3	147	147
4	185	148
6	200	186
out	-120	-137

We then created variable step ladder which basically relates between which values of dice variable, each outcome should occur; as can be seen in the picture above. For example, should the dice value (an algebraic function between Batsman Skill, Bowler Skill and a random luck factor variable) assume a value between 125 and 146, two runs will be added to the score as can be seen from the picture above.

(FIG 4) DICE VARIABLE

```
int dice = 2*bat_skill - 2*bowl_skill + luck;
```

Our entire backend code is built around this one simple equation where bat_skill is the Batsman Skill and bowl_skill the bowler skill both of which are to be taken from a predefined Database.

This dice is fed into the next part of our algorithm that is the if else statements that determine the outcome.

(FIG 5) BACKEND ALGORITHM

```
if(dice<=hout){
    out(true);
}
if (dice >= 10 && dice <= h0) {
    run0();
}
if (dice >= 11 && dice <= h1) {
    run1();
}
if (dice >= 12 && dice <= h2) {
    run2();
}
if(dice==13){
    run3();
}
if (dice >= 14 && dice <= h4) {
    run4();
}
if (dice >= 16) {
    run6();
}
```


From these if else statements the various outcome methods are called which get executed accordingly.

(FIG 6) OUTCOME FUNCTIONS

```
int run0(){
    System.out.println("0 Runs");
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "0";
    return(0);
}

int run1(){
    System.out.println("1 Run");
    runs++;
    game_runs++;
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "1";
    return(1);
}

int run2(){
    System.out.println("2 Run");
    runs+=2;
    game_runs+=2;
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "2";
    return(2);
}

int run3() {
    System.out.println("3 Run");
    runs += 3;
    game_runs+=3;
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "3";
    return(3);
}
```

```

int run4() {
    System.out.println("4 Run");
    runs += 4;
    game_runs+=4;
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "4";
    return(4);
}

int run6() {
    System.out.println("6 Run");
    runs += 6;
    game_runs+=6;
    balls++;
    game_balls++;
    imdt_outcome[game_balls] = "6";
    return(6);
}

```

This is the main backend part of the algorithm.

FIG 7



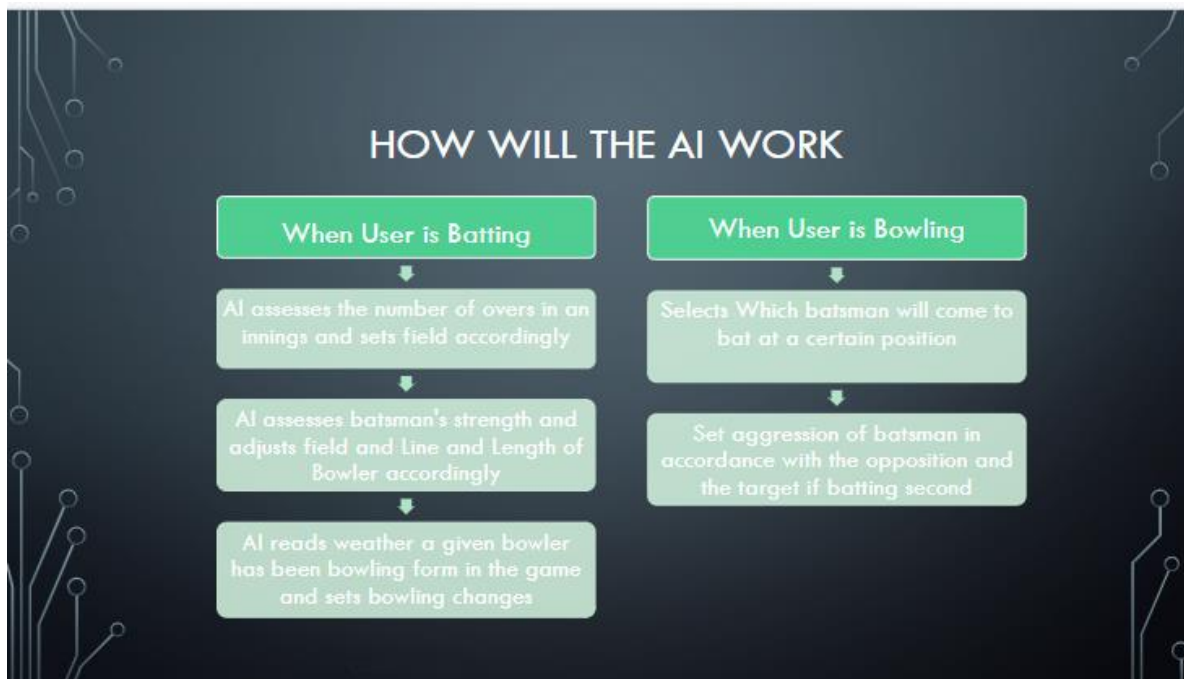


FIGURE 8.

Algorithm

```
// use byte instead of int (-127 to 127)
Variables
1. Here we have declared the variables for the program & database
2. We are going to declare the variables in an interface which is going to inherit in the backend code.
1 luck_factor
2 bat_form
3 bowl_form
4 pitch_condition
5 overhead_condition
6 strength
7 bat_agression
8 field
9 line
10 length
11 skills
11.1 batting
11.1.1 front_foot //out of 100
11.1.2 back_foot
11.1.3 off_side
11.1.4 leg_side
11.1.5 against_pace
11.1.6 against_spin
11.1.7 left_or_right
11.1.8 natural_agression
```

```

    11.1.9 batsman_traits
11.2 bowling
    11.2.1 pace
    11.2.2 left_or_right
    11.2.3 bowling_type
    11.2.4 swing
    11.2.5 seam
    11.2.6 spin
    11.2.7 drift
    11.2.8 accuracy
    11.2.9 bowling_traits
12 endurance/stamina
13 salary
14 run_dice
15 field_skill
16 batsman_runs[] //array
17 total_runs

```

Functions

1. Here we have declared the functions we are going to use in our code and going to inherit in an interface.

```

1 out_or_not_out
2 runs
3 over
4 no_ball
5 wide_ball
6 out_ways //class
    6.1 catchOut
    6.2 bowled
    6.3 lbw
    6.4 run_out
7 strike_rotate
8 bowler_change
9 field/field_change

```

```

    bat_skill = (front_foot or back_foot) + (off_side or leg_side) +
against_pace or against_spin; //depending on bowler type
    bowl_skill = seam or spin + swing or drift + accuracy; //depending on
bowler type
    //double dice_out = bat - bowl + luck; //weather batsman is out or not out
    double run_dice = bat - ball + bat_frm - bowl_frm + luck_number; //for
general luck of batsman

```

```

public void out(boolean b){
    if(b == true){
        System.out.println("OUT!!!");
        System.out.println("Runs Scored " + runs);
    }
}

```

```

        else{
            runs(bat_skill, bowl_skill, bat_agg, bowl_agg, bat_form,
bowl_form);
        }
    }

    luck -5 to 25
    if(run-dice <-3){
        out(true);
    }
    if(run_dice<=0){

        System.out.println("Play and miss");

    }
    else if(run_dice>0 && run_dice<=3){

        System.out.println("Dot ball");

    }
    else if(run_dice>3 && run_dice<=7){

        System.out.println("1 Run");
        runs++;
    }
    else if(run_dice>7 && run_dice<=9){

        System.out.println("2 Runs");
        runs+=2;
    }
    else if(run_dice>9 && run_dice<=10){

        System.out.println("3 Runs");
        runs+=3;

    }
    else if(run_dice>10 && run_dice<=12){

        System.out.println("4 Runs!!!!");
        runs+=4;

    }
    else if(run_dice>12){
        System.out.println("6 RUNS!!!!!!");
        runs+=6;
    }
    else{
        System.out.println("System Error!!");
    }
}

```

```

        return;
    }

    field{
        if (aggressive){
            luck == lower limit decrease and upper limit increase;
            side == offside or legside{
                out or runs on side; //mostly
            }
        }
    }

    if (side1 fielder on boundary > side2 fielder on boundary){
        field side1 defense;
        if (side1 != line){
            luck == -2 - 40;
        }
    }

    bat_agression{
        if bat_agression is high{
            luck == lower limit decrease and upper limit increase;
        }
    }

    form{
        (total_runs in last 5 matches + strike rate)
    }

    //ask how to set custom range

```

ALGORITHM DESCRIPTION

We determine the dice variable which is batsman skill minus bowler skill plus luck factor which is a random variable between a certain range. This dice variable is then fed into an if else control structure which determines the outcome according to the value of the dice variable. Accordingly functions for different outcomes like run0, run4, out (true) are called.

We first determine the length of the ball. Upon that we then decide weather batsman will play on the front foot or the back foot. Similarly, we determine the line of the ball and decide weather batsman is going to play on offside or onside. Then we match the corresponding skills to determine the batsman skill. For bowlers, its more or less constant with swing and seam being characteristics of pacers and spin and drift being that of spinners.

For field and batsman aggression, higher aggression means wider range for the luck factor while lower aggressiveness mean narrower range for luck factor.

Design

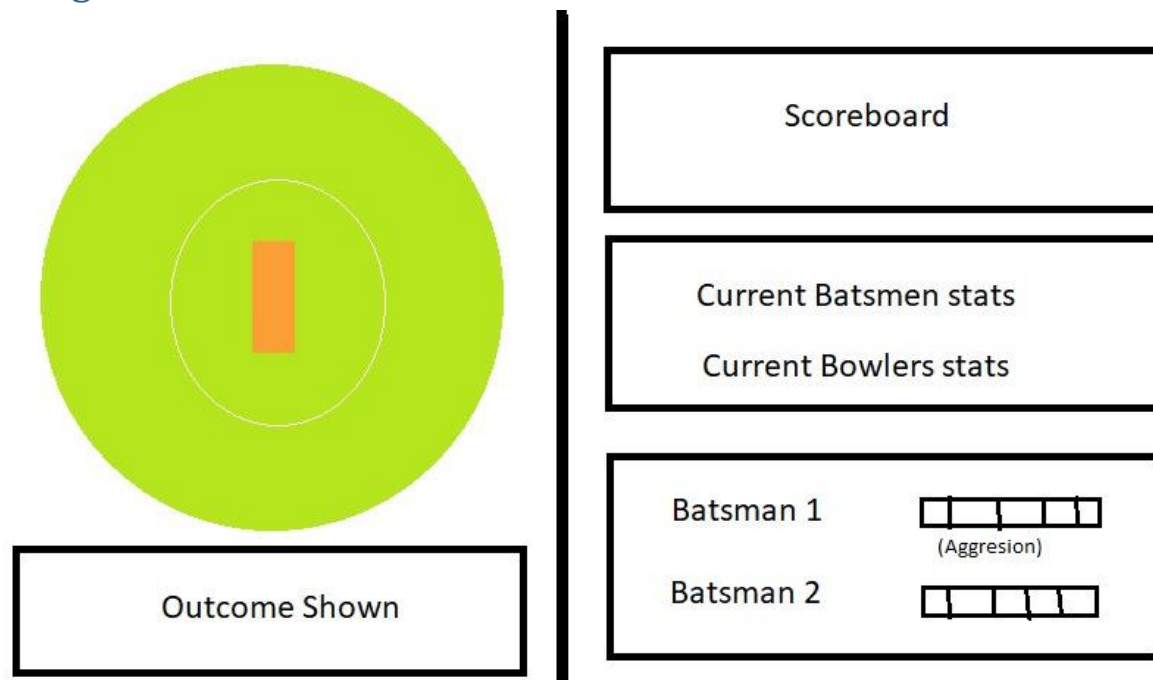


FIGURE 9 – UI when user's team will be batting

The above picture is the UI when the user's selected team is batting. The user can see the scoreboard on the Right-hand side along with the stats of both bowler and batsman. In the lower right side, the user can set aggression of the two batsmen according to his or her wishes. The ground on the left-hand side shows a simulation of the cricket ground with animations like ball running to the boundary, fielders chasing it, etc. We have planned to implement the ground part using Java OpenGL wrapper library or JOGL in short.

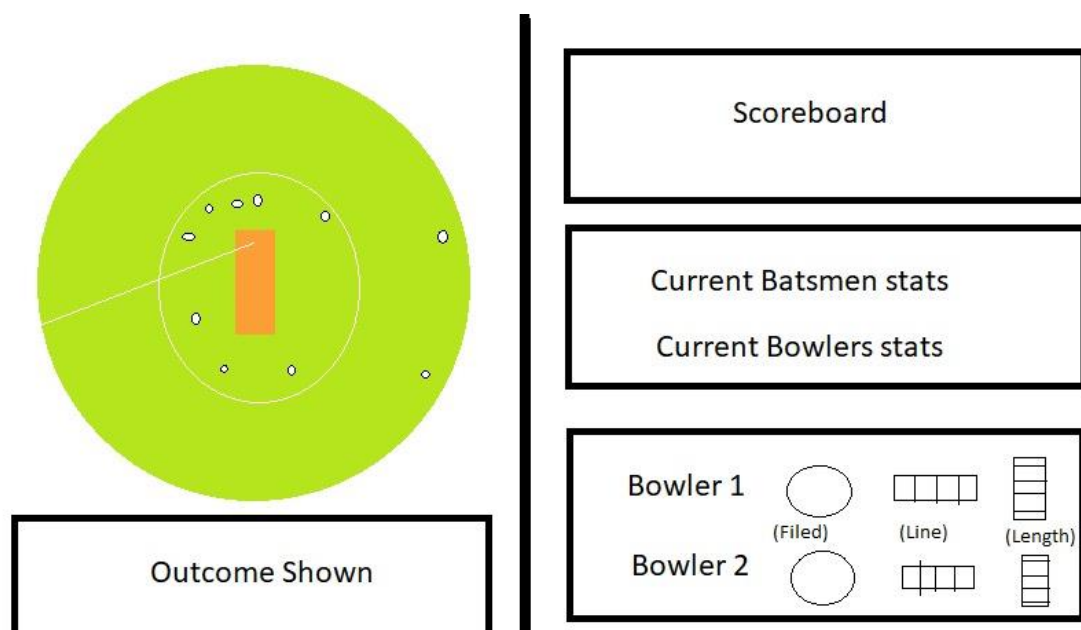


Figure 10 – UI when user's team will be bowling

The picture above represents the UI when the user's selected team will be bowling. As usual we can see the Scoreboard and the current stats on the right side. In the lower right side, the user can select the field placement, Line and the length of the bowler according to his wishes. After the end of each over the user will also have the option of selecting which bowler would bowl.



FIGURE 11

Project Sample

The pictures below represent the current working state of our program.

Field Aggression Batsman Aggression

Score 32/1 Overs - 4.0

Ball 1 - 0

Ball 2 - 1

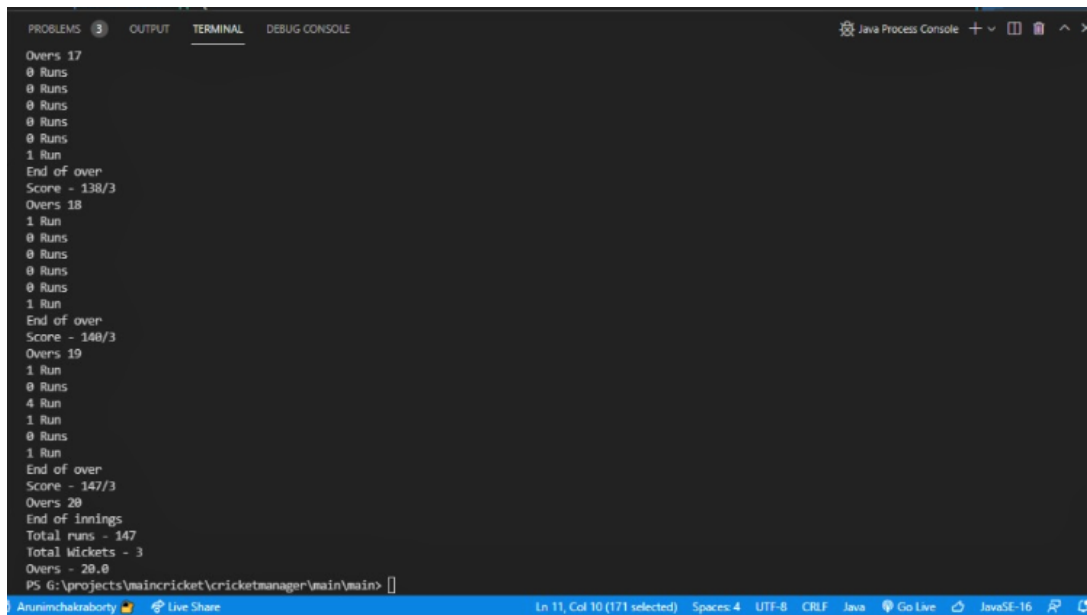
Ball 3 - 1

Ball 4 - 0

Ball 5 - 4

Ball 6 - 1

FIGURE 12 – Project Sample UI



The terminal window displays the following text:

```
Overs 17
0 Runs
0 Runs
0 Runs
0 Runs
0 Runs
1 Run
End of over
Score - 138/3
Overs 18
1 Run
0 Runs
0 Runs
0 Runs
0 Runs
1 Run
End of over
Score - 146/3
Overs 19
1 Run
0 Runs
4 Run
1 Run
0 Runs
1 Run
End of over
Score - 147/3
Overs 20
End of innings
Total runs - 147
Total Wickets - 3
Overs - 20.0
PS G:\projects\maincricket\cricketmanager\main\main>
```

The terminal interface includes tabs for PROBLEMS, OUTPUT, TERMINAL, and DEBUG CONSOLE. The status bar at the bottom shows 'Ln 11, Col 10 (171 selected)', 'Spaces: 4', 'UTF-8', 'CRLF', 'Java', 'Go Live', 'JavaSE-16', and a 'Live Share' button.

FIGURE 13 – Outcomes Shown on the terminal

Further Upgrades

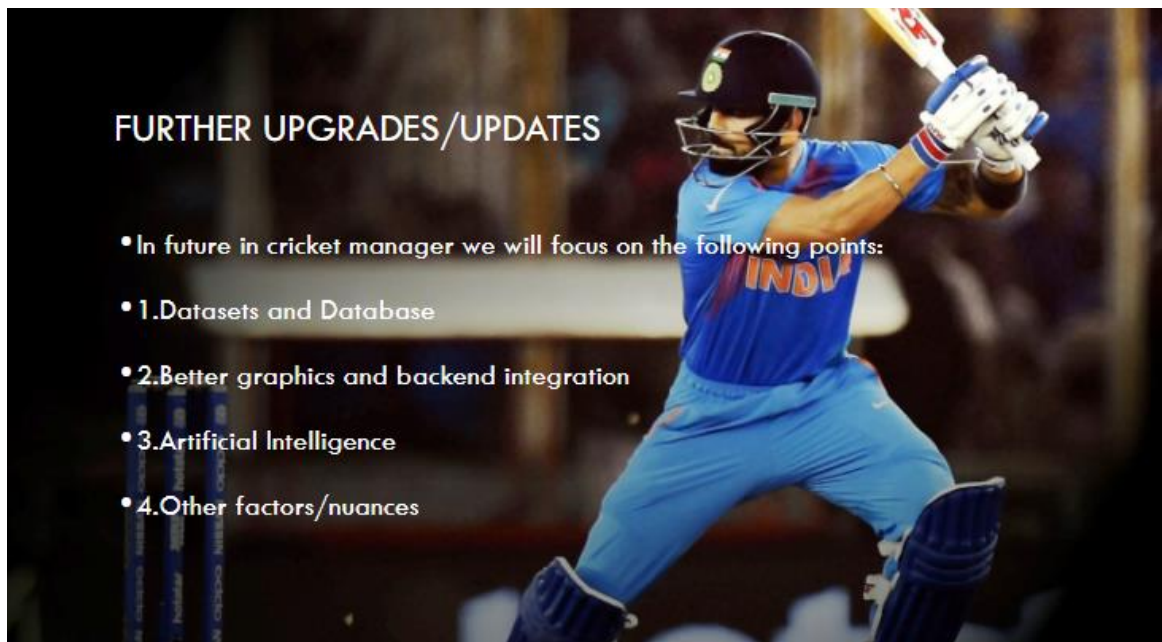


FIGURE 14 – Future Upgrades

Other Small Nuances

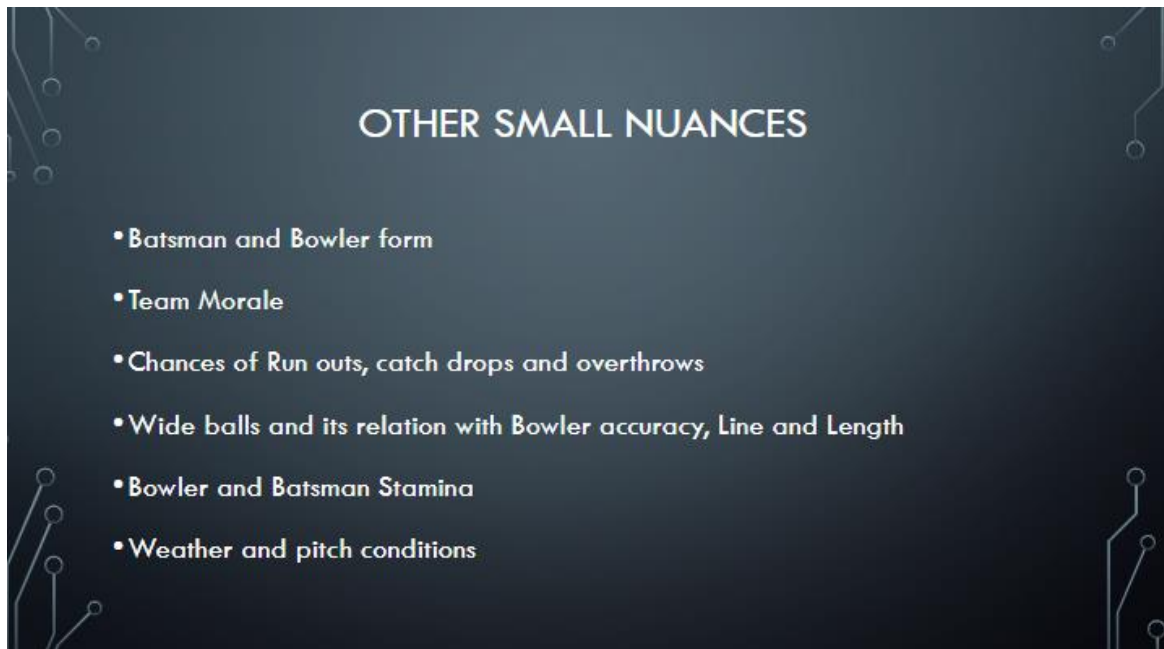


FIGURE 15 – Small Nuances

Conclusion

In this project we shall create a cricket manager game which implements AI and makes uses intricate backend code to make it intricate.

Reference Links –

<https://www.kaggle.com/nowke9/ipldata>

<https://www.footballmanager.com/>