

# **JAVA PROJECT REPORT**

(Project Term January-May 2023)

**DLS CRICKET WIZARD: WINNING BY THE NUMBERS**

Submitted by

**Name: Aditya Mehra**

**Registration Number:12101319**

**COURSE CODE: CSE 310**

Under the Guidance of

Dr. Ranjith Kumar Sir

School of Computer Science and Engineering



**L** LOVELY  
**P** ROFESSIONAL  
**U** NIVERSITY

# DECLARATION

I hereby declare that the project work entitled (“DLS CRICKET WIZARD: WINNING BY THE NUMBERS.”) is an authentic record of our own work carried out as requirements of Capstone Project for the award of B. Tech degree in B. Tech CSE (Program Name) from Lovely Professional University, Phagwara, under the guidance of (DR. Ranjith Kumar A), during January to May 2023. All the information furnished in this capstone project report is based on my own intensive work and is genuine.

Name of Student :

Aditya Mehra

Registration Number:

12101319

Roll Number: K21WBB46

(Signature of Student )

Aditya Mehra

Date:

22/04/2023

# TABLE OF CONTENTS

---

Inner first page.....	(1)
Declaration.....	(2)
Table of Contents.....	(3)
1.INTRODUCTION	(4)
1.1. Background of the DLS Method	(4)
1.2. Objective of the Project	(5)
1.3. Problem Statement	(6)
1.4. System Specification	(6)
2.MODULE 1-SCREENSHOT OF THE CODE	
2.1 HOME CLASS	(7)
2.2. ODI CLASS FRAME	(7)
2.3. ODI CLASS PANEL	(8)
2.4. T20I LOGIC	(9)
3.MODULE2 (OUTPUT OF THE CODE)	
3.1 HOME WINDOW	(12)
3.2. DATA INPUT	(12)
3.3. MESSAGE DIALOGUE BOX	(13)
2.4. RESULT	(13-14)
4.CONCLUSION	(16)
5.FUTURE SCOPE	(17)
6.REFERENCE	(17)

# 1. INTRODUCTION

---

The Duckworth-Lewis-Stern (DLS) method is a complex mathematical formula used in cricket to adjust targets in weather-affected limited-overs matches. The method was first introduced in 1997 by Frank Duckworth and Tony Lewis and later revised in 2014 with the addition of Steven Stern's name. The DLS method has replaced the previous system of the 'rain rule' that was used in cricket before its introduction. The method takes into account the number of overs remaining, wickets in hand, and the resources of the team batting second, to set a revised target for them to achieve in the event of weather disruptions or other interruptions. The DLS method has proven to be a fairer and more accurate system than the previous rain rule, as it ensures that both teams have equal opportunities to win the game, regardless of the interruption. It is now widely used in all formats of limited-overs cricket worldwide, including One Day Internationals (ODIs) and T20 Internationals (T20Is), and has become an integral part of the sport.

## 1.1 Background of the Pong Game

---

The DLS method was introduced as a more fair and accurate alternative to the previous rain rule system used in cricket. The rain rule system, which was based on the number of overs bowled by the team batting first, often put the team batting second at a disadvantage. This was because they would have to score a higher run rate to win the match within the reduced overs, even if the target set by the team batting first was lower than what it should have been in a full match.

To address these issues, Frank Duckworth and Tony Lewis, two statisticians from the UK, were tasked with developing a new system in the mid-1990s. The result was the Duckworth-Lewis method, which was first implemented in international cricket in 1997 during the series between Zimbabwe and England.

Since its introduction, the DLS method has undergone several revisions and is now known as the Duckworth-Lewis-Stern (DLS) method, after the inclusion of Australian statistician Steven Stern. It has become the standard method for adjusting targets in limited-overs cricket matches that are

affected by rain, bad light or other interruptions. The method has been widely praised for providing a fairer and more accurate way of setting revised targets, ensuring that both teams have an equal opportunity to win the match.

## **1.2 OBJECTIVE OF THE PROJECT**

---

1. To develop a user-friendly and accurate DLS calculator tool that can be used by cricket fans, players, and coaches to predict and analyze match outcomes.
2. To incorporate the latest DLS method updates into the calculator and ensure that it provides reliable match analysis in all formats of limited-overs cricket.
3. To enable users to enter the current score and wicket status of both teams and adjust for any weather interruptions, to calculate revised targets for the team batting second.
4. To allow users to simulate different match scenarios and predict the likely outcome based on the revised targets, providing valuable insights for team planning and strategy.
5. To provide a platform for users to compare the performance of different teams and players based on DLS-adjusted match statistics.
6. To analyse the impact of different weather conditions on match outcomes and identify patterns and trends in DLS-adjusted scores.
7. To create a dynamic and responsive user interface that can be easily accessed on different devices and platforms.
8. To ensure the accuracy and reliability of the calculator by testing it with real match data and validating it against the official DLS method.
9. To provide a valuable educational resource for cricket fans and players, helping them to understand the complexities of the DLS method and how it affects match outcomes.
10. To promote the use of the DLS method in cricket and encourage greater awareness and understanding of this important aspect of the game.

## 1.3 PROBLEM STATEMENT

---

Limited-overs cricket matches are often affected by weather interruptions, leading to the need for revised targets using the DLS method. However, calculating and predicting these targets accurately can be complex and time-consuming, requiring a thorough understanding of the DLS formula and match conditions. This presents a challenge for cricket fans, players, and coaches who want to analyse and predict match outcomes, plan strategies, and compare performance based on DLS-adjusted statistics. There is a need for a user-friendly and reliable DLS calculator tool that can provide accurate and real-time match analysis, allowing users to adjust for weather interruptions and simulate different match scenarios. Such a tool would be valuable for cricket enthusiasts and professionals alike, helping them to better understand and utilize the DLS method in limited-overs cricket and enhancing their overall experience of the sport.

## 1.4 SYSTEM SPECIFICATIONS

---

### **Minimum system requirements:**

CPU:

Dual Core CPU

RAM: 2 GB

RAM GPU:

DX11 compliant graphics card / OpenGL 4-compliant onboard graphics

DX: Version 11 OS:

Microsoft Windows 7 with SP1 STO:

160 MB available space

### **Recommended system requirements:**

CPU:

Quad Core CPU

RAM: 8 GB

RAM GPU:

Dedicated Graphics Card

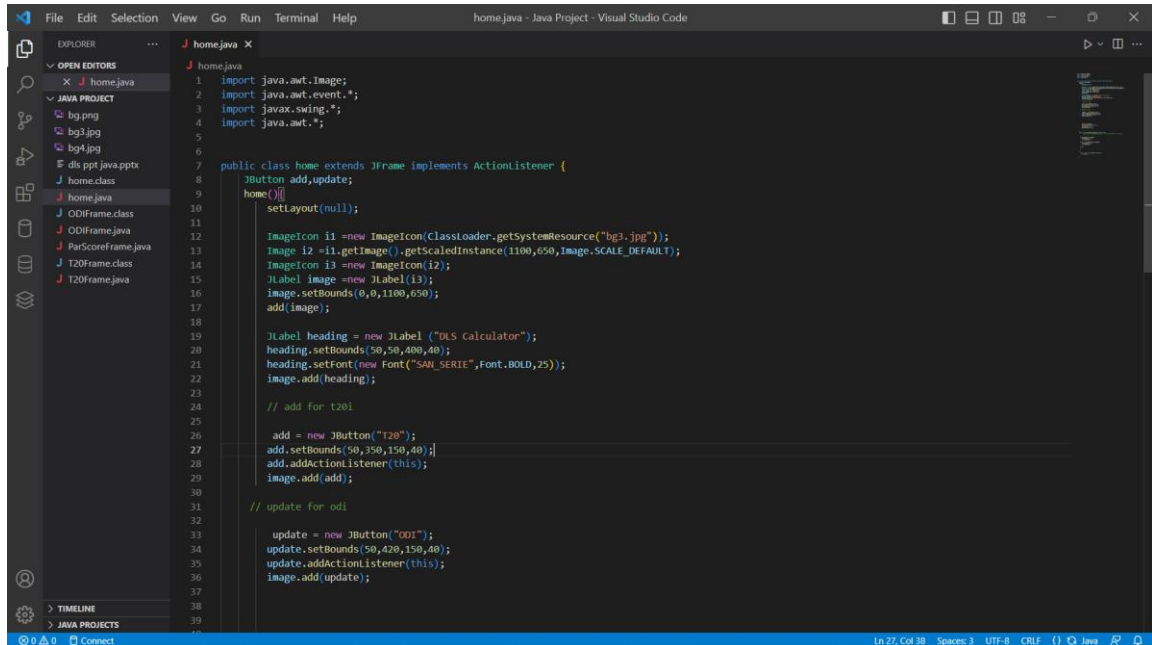
DX: Version 11 OS:

Microsoft Windows 10 STO:

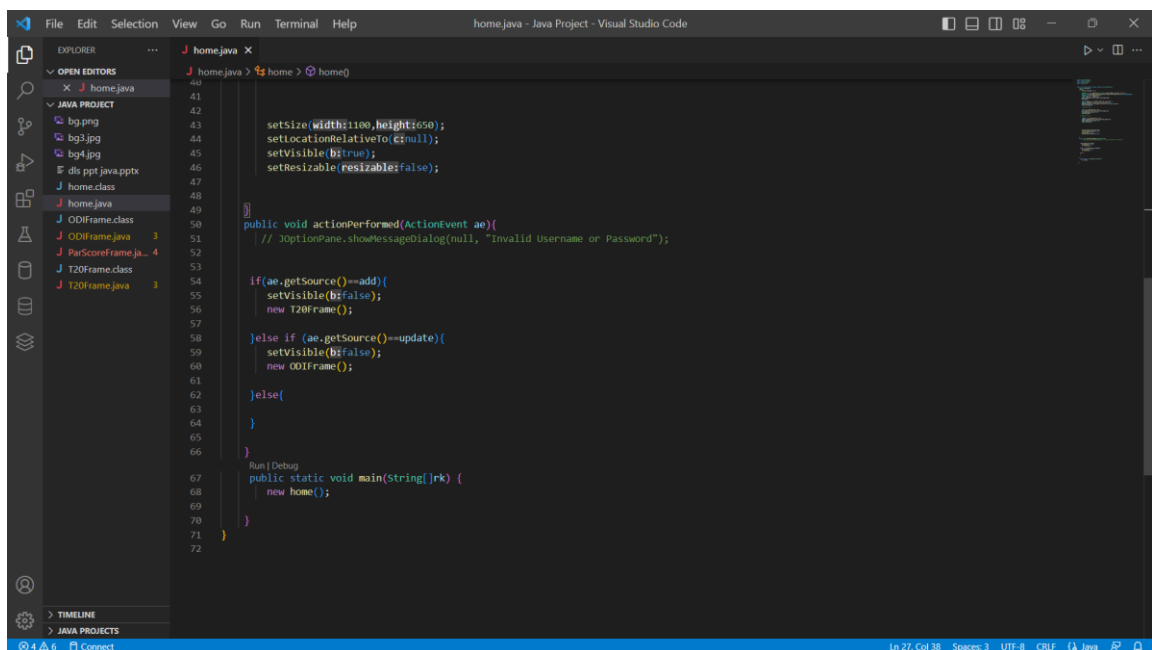
200 MB available space

# Module 1 (SCREENSHOTS OF THE CODE)

## HOME CLASS

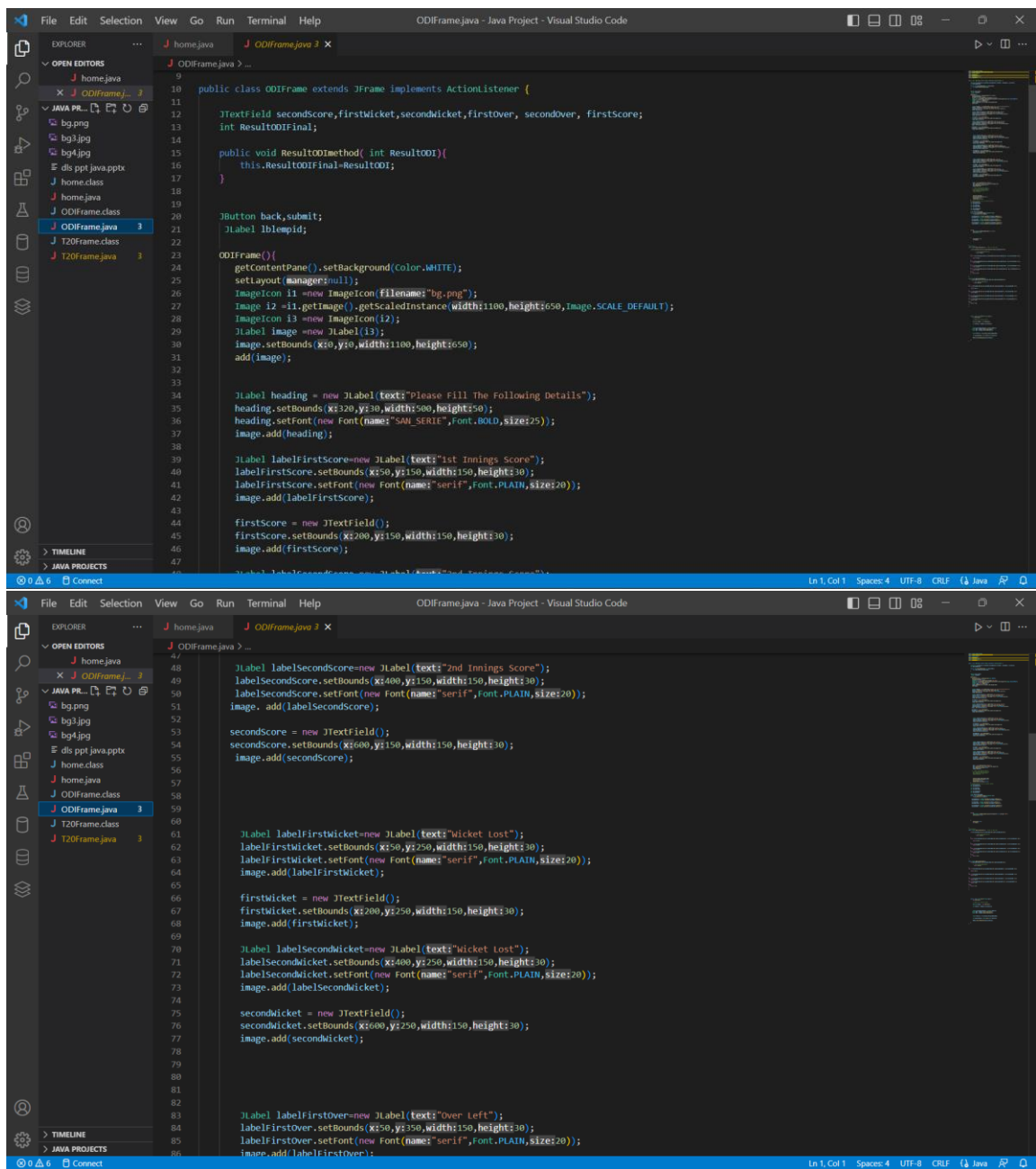


```
1 import java.awt.Image;
2 import java.awt.event.*;
3 import javax.swing.*;
4 import java.awt.*;
5
6
7 public class home extends JFrame implements ActionListener {
8     JButton add,update;
9     home(){
10         setLayout(null);
11
12         ImageIcon i1 =new ImageIcon(ClassLoader.getSystemResource("bg3.jpg"));
13         Image i2 =i1.getImage().getScaledInstance(1100,650,Image.SCALE_DEFAULT);
14         ImageIcon i3 =new ImageIcon(i2);
15         JLabel image =new JLabel(i3);
16         image.setBounds(0,0,1100,650);
17         add(image);
18
19         JLabel heading = new JLabel ("DLS Calculator");
20         heading.setBounds(50,50,400,40);
21         heading.setFont(new Font("SAN_SERIF",Font.BOLD,25));
22         image.add(heading);
23
24         // add for t201
25
26         add = new JButton("T201");
27         add.setBounds(50,350,150,40);
28         add.addActionListener(this);
29         image.add(add);
30
31         // update for odi
32
33         update = new JButton("001");
34         update.setBounds(50,430,150,40);
35         update.addActionListener(this);
36         image.add(update);
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

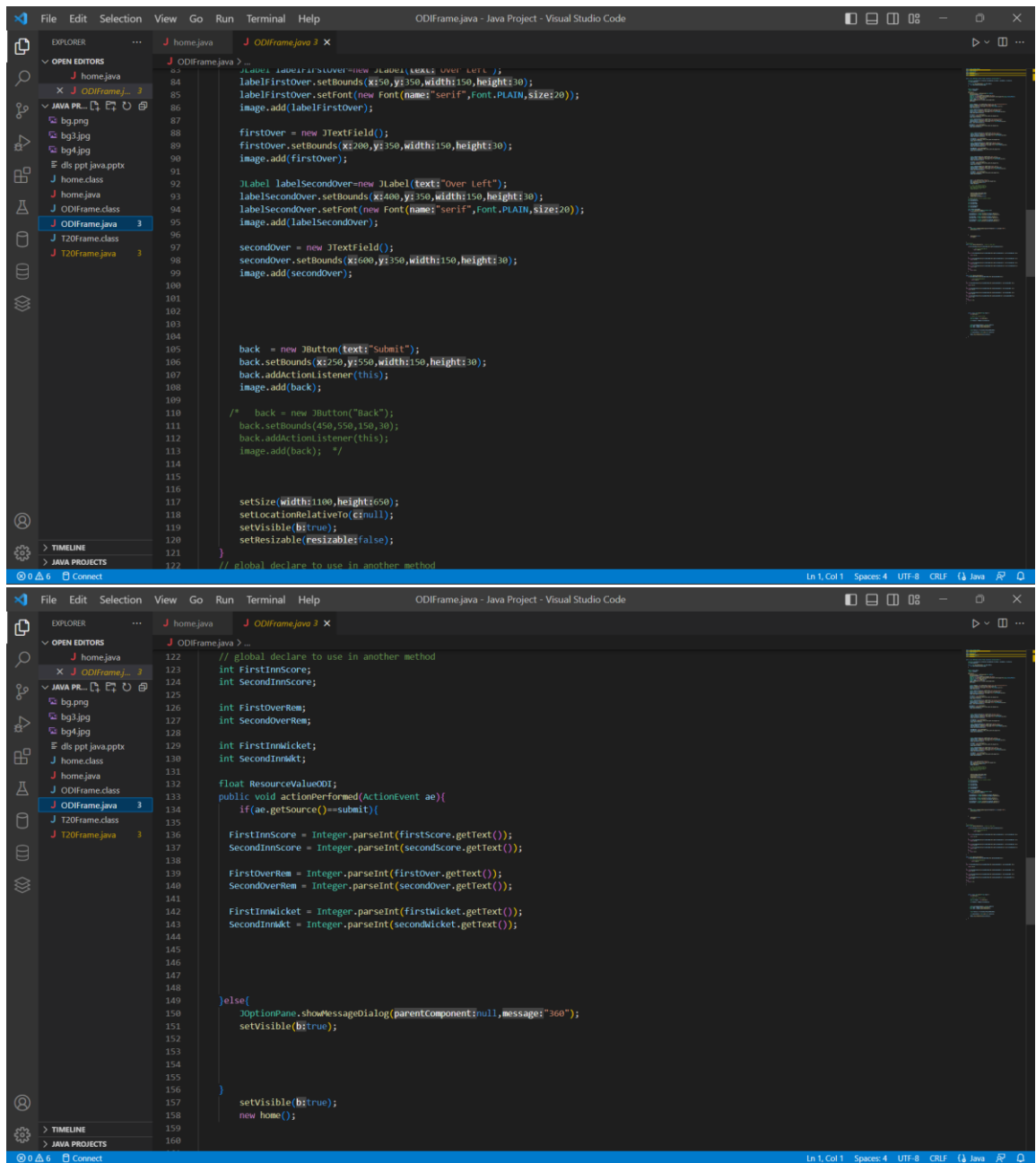


```
41
42
43         setSize(width:1100,height:650);
44         setLocationRelativeTo(null);
45         setVisible(true);
46         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
47
48     }
49
50     public void actionPerformed(ActionEvent ae){
51         // JOptionPane.showMessageDialog(null, "Invalid Username or Password");
52
53         if(ae.getSource()==add){
54             setVisible(false);
55             new T20Frame();
56         }
57         else if (ae.getSource()==update){
58             setVisible(false);
59             new ODIFrame();
60         }
61         else{
62
63         }
64     }
65
66 }
67
68 Run | Debug
69 public static void main(String[]rk) {
70     new home();
71 }
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

# ODI CLASS FRAME







```
File Edit Selection View Go Run Terminal Help
ODIFrame.java - Java Project - Visual Studio Code

EXPLORER
  home.java
  ODIFrame.java
  bg3.jpg
  bg4.jpg
  dls ppt.java.pptx
  home.class
  ODIFrame.class
  ODIFrame.java
  T20Frame.class
  T20Frame.java

JAVAC
  home.java
  ODIFrame.java
  T20Frame.java

TIMELINE
  ODIFrame.java
  T20Frame.java

JAVAC PROJECTS
  Connect

162
163
164 //calculation
165 public float ResourceValueist() { //float kr dena hai
166
167     if((FirstOverRem==40 && FirstOverRem<=50) && (FirstInnMkct==0)){
168
169         // ResourceValueODI=100.0f;
170         return 100.0f;
171     }
172
173     else if((FirstOverRem==40 && FirstOverRem<=50) && (FirstInnMkct<=2 && FirstInnMkct >0)){
174
175         return 83.8f;
176     }
177
178     else if((FirstOverRem==40 && FirstOverRem<=50) && (FirstInnMkct<=5 && FirstInnMkct >2)){
179
180         return 49.5f;
181     }
182
183     else if((FirstOverRem==40 && FirstOverRem<=50) && (FirstInnMkct<=7 && FirstInnMkct >5)){
184
185         return 26.5f;
186     }
187
188     else if((FirstOverRem==40 && FirstOverRem<=50) && (FirstInnMkct<=10 && FirstInnMkct >7)){
189
190         return 7.6f;
191     }
192
193     else{
194         return 0.0f;
195     }
196 }
197
198 public float ResourceValue2nd(){
199     if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct==0)){
200
201         // ResourceValueODI=100.0f;
202         return 100.0f;
203     }
204
205     else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=2 && SecondInnMkct >0)){
206
207         return 83.8f;
208     }
209
210     else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=5 && SecondInnMkct >2)){
211
212         return 49.5f;
213     }
214
215     else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=7 && SecondInnMkct >5)){
216
217         return 26.5f;
218     }
219
220     else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=10 && SecondInnMkct >7)){
221
222         return 7.6f;
223     }
224
225     else{
226         return 0.0f;
227     }
228 }
229
230 public static void main(String args[]) {
231     new ODIFrame();
232
233     // object for resource value
234     ODIFrame MyObj1 = new ODIFrame();
235 }
```

```
File Edit Selection View Go Run Terminal Help
ODIFrame.java - Java Project - Visual Studio Code

EXPLORER
  home.java
  ODIFrame.java
  bg3.jpg
  bg4.jpg
  dls ppt.java.pptx
  home.class
  ODIFrame.class
  ODIFrame.java
  T20Frame.class
  T20Frame.java

JAVAC
  home.java
  ODIFrame.java
  T20Frame.java

TIMELINE
  ODIFrame.java
  T20Frame.java

JAVAC PROJECTS
  Connect

197
198 // ResourceValueODI=100.0f;
199 return 100.0f;
200
201 else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=2 && SecondInnMkct >0)){
202
203     return 83.8f;
204 }
205
206 else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=5 && SecondInnMkct >2)){
207
208     return 49.5f;
209 }
210
211 else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=7 && SecondInnMkct >5)){
212
213     return 26.5f;
214 }
215
216 else if((SecondOverRem==40 && SecondOverRem<=50) && (SecondInnMkct<=10 && SecondInnMkct >7)){
217
218     return 7.6f;
219 }
220
221 else{
222     return 0.0f;
223 }
224
225 }
226
227
228
229 Run|Debug
230 public static void main(String args[]) {
231     new ODIFrame();
232
233     // object for resource value
234     ODIFrame MyObj1 = new ODIFrame();
235 }
```

```

197 // ResourceValueODI=100.0f;
198 return 100.0f;
199
200 }
201 else if((SecondOverRem>=40 && SecondOverRem<50) &&(SecondInnMkt<=2 && SecondInnMkt >0)){
202     return 83.8f;
203 }
204
205 }
206 else if((SecondOverRem>=40 && SecondOverRem<50) &&(SecondInnMkt<=5 && SecondInnMkt >2)){
207     return 49.5f;
208 }
209
210 }
211 else if((SecondOverRem>=40 && SecondOverRem<50) &&(SecondInnMkt<=7 && SecondInnMkt >5)){
212     return 26.5f;
213 }
214
215 }
216 else if((SecondOverRem>=40 && SecondOverRem<50) &&(SecondInnMkt<=10 && SecondInnMkt >7)){
217     return 7.6f;
218 }
219
220 }
221 else{
222     return 0.0f;
223 }
224
225 }
226
227 }
228
229
230 Run|Debug
231 public static void main(String args[]) {
232     new ODIFrame();
233 }
234
235 // object for resource value
236 ODIFrame MyObj1 = new ODIFrame();

```

## T20 CLASS FRAME

```

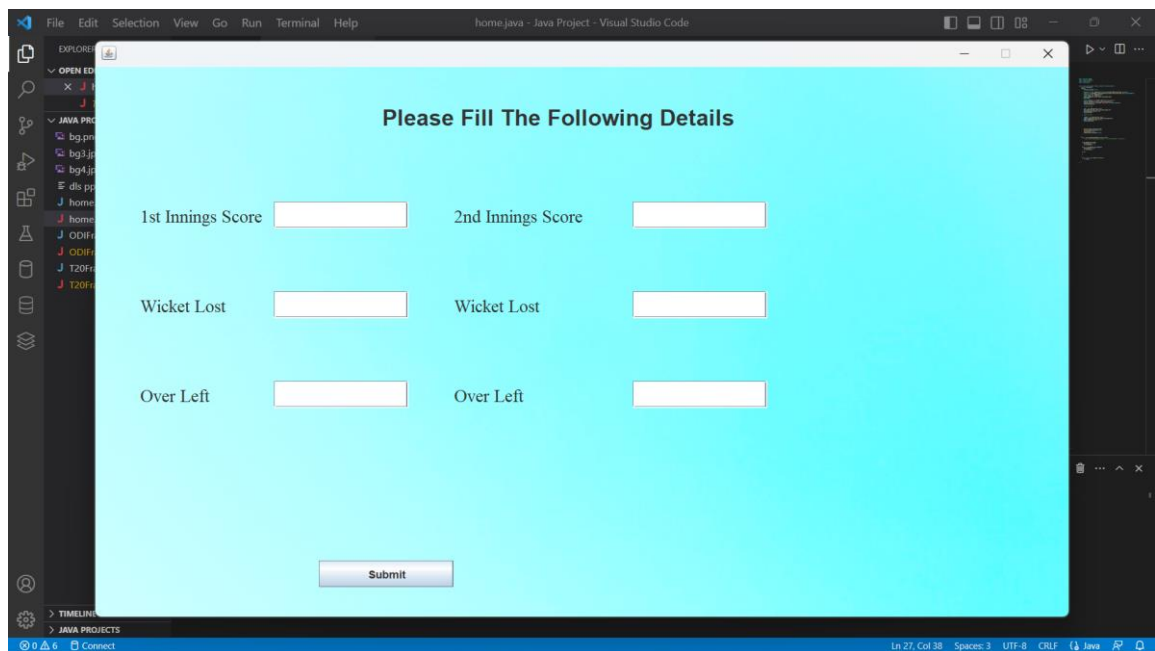
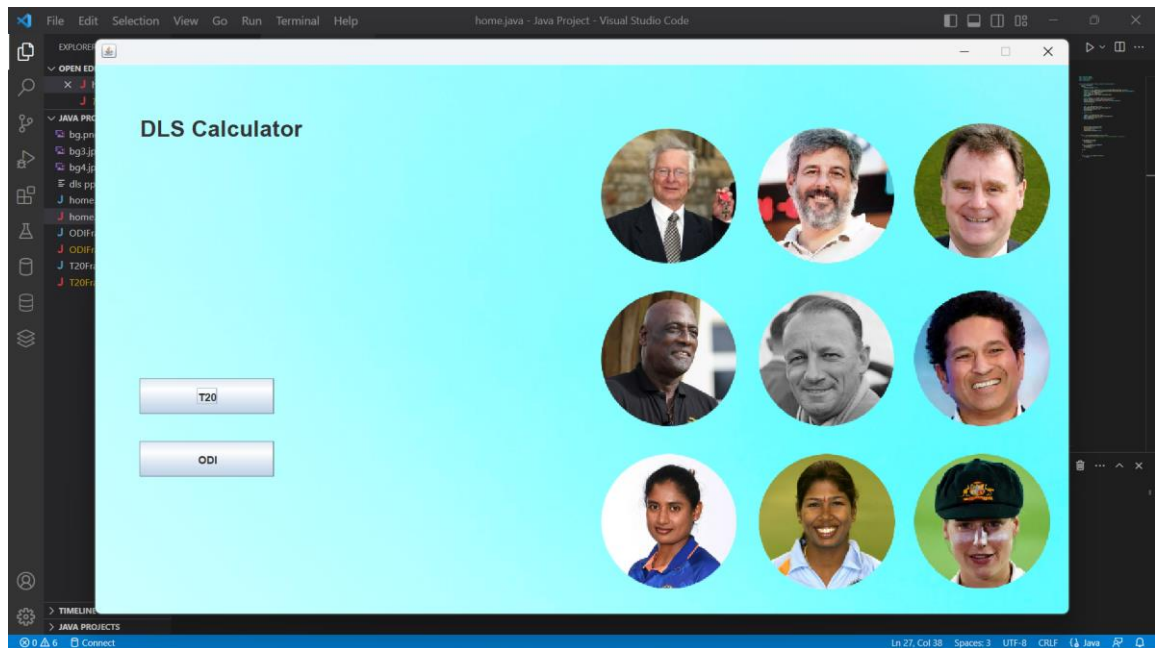
1 import javax.swing.JFrame;
2 import javax.swing.JOptionPane;
3 import javax.swing.*;
4 import java.awt.*;
5 import java.util.*;
6 import java.awt.event.*;
7 import java.io.*;
8
9
10 public class T20Frame extends JFrame implements ActionListener {
11
12     JtextField secondScore,firstwicket,secondwicket,firstOver, secondOver, firstScore;
13     int ResultODIFinal;
14
15     public void ResultODIMethod( int ResultODI){
16         this.ResultODIFinal=ResultODI;
17     }
18
19
20     JButton back,submit;
21     JLabel lblEmpid;
22
23     T20Frame(){
24         getContentPane().setBackground(Color.WHITE);
25         setLayout(null);
26         ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("bg.png"));
27         Image i2 = i1.getImage().getScaledInstance(1100,650,Image.SCALE_DEFAULT);
28         ImageIcon i3 = new ImageIcon(i2);
29         JLabel image = new JLabel(i3);
30         image.setBounds(10,10,1100,650);
31         add(image);
32
33
34         JLabel heading = new JLabel("Please fill The following Details");
35         heading.setBounds(120,30,400,50);
36         heading.setFont(new Font("SAN_SERIF",Font.BOLD,25));
37         image.add(heading);
38
39         JLabel labelFirstScore=new JLabel("1st Innings Score");

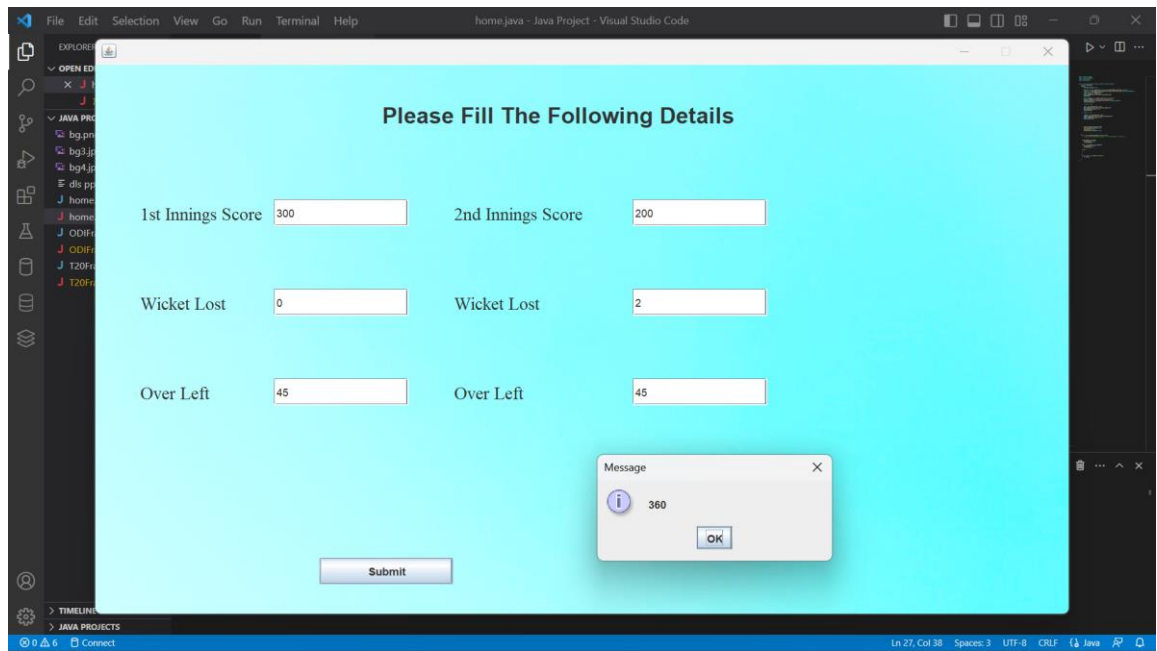
```

```
163 }
164 //calculation
165 public float ResourceValue1st(){ //float kr dena hai
166
167     if((FirstOverRem>=40 && FirstOverRem<=50) && (FirstInnMicket==0)){
168         // ResourceValueODI=100.0f;
169         return 100.0f;
170     }
171
172     else if((FirstOverRem>=40 && FirstOverRem<=50) && (FirstInnMicket<=2 && FirstInnMicket >0)){
173         return 83.8f;
174     }
175
176     else if((FirstOverRem>=40 && FirstOverRem<=50) && (FirstInnMicket<=5 && FirstInnMicket >2)){
177         return 49.5f;
178     }
179
180     else if((FirstOverRem>=40 && FirstOverRem<=50) && (FirstInnMicket<=7 && FirstInnMicket >5)){
181         return 26.5f;
182     }
183
184     else if((FirstOverRem>=40 && FirstOverRem<=50) && (FirstInnMicket<=10 && FirstInnMicket >7)){
185         return 7.6f;
186     }
187
188     else{
189         return 0.0f;
190     }
191 }
192
193 public float ResourceValue2nd(){
194     if((SecondOverRem>=40 && SecondOverRem<=50) && (SecondInnMicket==0)){
195         // ResourceValueODI=100.0f;
196         return 100.0f;
197     }
198
199     else if((SecondOverRem>=40 && SecondOverRem<=50) && (SecondInnMicket<=2 && SecondInnMicket >0)){
200
201
202     }
203
204     else if((SecondOverRem>=40 && SecondOverRem<=50) && (SecondInnMicket<=5 && SecondInnMicket >2)){
205         return 49.5f;
206     }
207
208     else if((SecondOverRem>=40 && SecondOverRem<=50) && (SecondInnMicket<=7 && SecondInnMicket >5)){
209         return 26.5f;
210     }
211
212     else if((SecondOverRem>=40 && SecondOverRem<=50) && (SecondInnMicket<=10 && SecondInnMicket >7)){
213         return 7.6f;
214     }
215
216     else{
217         return 0.0f;
218     }
219 }
220
221
222
223
224
225
226
227
228
229 public static void main(String args[]) {
```

```
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229 public static void main(String args[]) {
```

# OUTCOME





## CONCLUSION

---

1. The DLS calculator is an innovative tool that can greatly enhance the accuracy and reliability of match analysis in limited-overs cricket.
2. By providing a user-friendly interface that incorporates the latest DLS method updates, the calculator can help cricket fans, players, and coaches to make more informed decisions and predictions.
3. The calculator can adjust targets based on various match scenarios and weather conditions, enabling users to simulate different outcomes and test their strategies.
4. The DLS calculator can also provide a platform for comparing the performance of different teams and players, highlighting strengths and weaknesses and identifying areas for improvement.
5. The calculator is based on a complex mathematical formula that has been validated against real match data and has become the standard method for adjusting targets in limited-overs cricket.
6. The calculator can be used in all formats of limited-overs cricket, including One Day Internationals (ODIs) and T20 Internationals (T20Is), and is therefore highly versatile and adaptable.
7. The DLS calculator can be accessed on different devices and platforms, making it accessible to a wide range of users.
8. The calculator is designed to be intuitive and user-friendly, reducing the need for extensive technical knowledge or expertise.
9. The DLS calculator is an important educational resource for cricket enthusiasts, helping them to understand and utilize the complexities of the DLS method and enhancing their overall appreciation of the sport.
10. Overall, the DLS calculator is an innovative and valuable tool that has the potential to revolutionize the way cricket fans, players, and coaches analyze and predict match outcomes in limited-overs cricket.

## **FUTURE SCOPE:**

---

1. Integration with live streaming platforms: The DLS calculator can be integrated with live streaming platforms, allowing users to follow the match in real-time and receive real-time analysis and predictions.
2. Expansion to other sports: The DLS calculator can be adapted to other sports that utilize similar methods of target adjustment, such as soccer and field hockey.
3. Enhanced data analysis: The DLS calculator can be integrated with advanced data analysis tools, allowing for deeper insights into match performance and trends.
4. Integration with fantasy sports platforms: The DLS calculator can be integrated with fantasy sports platforms, allowing users to make more informed decisions and predictions in their fantasy cricket leagues.
5. Partnership with cricket boards: The DLS calculator can partner with cricket boards and leagues to provide official match analysis and predictions, enhancing the credibility and reliability of the tool.
6. Integration with artificial intelligence: The DLS calculator can be integrated with artificial intelligence algorithms to provide even more accurate and precise analysis and predictions.
7. Development of a mobile app: The DLS calculator can be developed into a mobile app, allowing users to access the tool on their smartphones and tablets.
8. Incorporation of player performance data: The DLS calculator can be integrated with player performance data, allowing users to analyze the impact of individual players on match outcomes and identify key performers.
9. Expansion to other DLS methods: The DLS calculator can be expanded to include other DLS methods, such as the VJD and J Duckworth-Lewis methods, providing a more comprehensive suite of tools for match analysis and prediction.



10.Integration with social media platforms: The DLS calculator can be integrated with social media platforms, allowing users to share their analysis and predictions with their friends and followers, promoting greater engagement and interest in limited-overs cricket.

## **REFERENCE**

---

1.WIKIPEDIA

2.YOUTUBE

3.JAVA ORACLE FOR JFRAMES