A Mini Project Report

On

# OUT!

Submitted in partial fulfillment of requirements for the Course
CSE18R272 - JAVA PROGRAMMING

**Bachelor's of Technology**

In

**Computer Science and Engineering**

Submitted By

**MUNAGA RAKESH**

**9918004073**

**SHAIK MAHABUB SHAARIIEF**

**9918004108**

Under the guidance of

**Dr. R. RAMALAKSHMI**

(Associate Professor)

**Department of Computer Science and Engineering**

**Kalasalingam Academy of Research and Education**

**Anand Nagar, Krishnankoil-626126**

**APRIL 2020**

# ABSTRACT

OUT! project is based on cricket scorer and it is a Android Application.It is Digital Scorebook.It provides the simplest way to do Cricket Scoring.It has the features of toss of the match,overs,players,teams which are required for test and one-day matches.It is easy to use interface to score the match.The aim of the project is to convert paper scorebook to digital scorebook.It is easy for everyone because using smartphones and need not carry paper.In the match there are a lot of riots about the score atleast in one run but the using of this application to reduce the disturbance the teams.

# DECLARATION

I hereby declare that the work presented in this report entitled "**OUT!**", in partial fulfilment of the requirements for the course CSE18R272- Java Programming and submitted in **Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education** (**Deemed to be University**) is an authentic record of our own work carried out during the period from **Jan 2020** under the guidance of Mr. **Dr. R. Ramalakshmi** (Associate Professor).

The work reported in this has not been submitted by me for the award of any other degree of this or any other institute.

**MUNAGA RAKESH**

**9918004073**

**SHAIK MAHABUB SHAARIIEF**

**9918004108**

# ACKNOWLEDGEMENT

First and foremost, I wish to thank the **Almighty God** for his grace and benediction to complete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.

I express deep sense of gratitude to “Kalvivallal” Thiru. **T. Kalasalingam** B.com., Founder Chairman, “Ilayavallal” **Dr.K.Sridharan** Ph.D., Chancellor, **Dr.S.ShasiAnand**, Ph.D., Vice President (Academic) , **Mr.S.Arjun Kalasalingam** M.S., Vice President (Administration) , **Dr.R.Nagaraj** Vice-Chancellor, **Dr.V.Vasudevan** Ph.D., Registrar , **Dr.P.Deepalakshmi** Ph.D., Dean (School of Computing) . And also a special thanks to **Dr. A. FRANCIS SAVIOUR DEVARAJ**. Head Department of CSE, Kalasalingam Academy of Research and Education forgranting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **Dr.R.Ramalakshmi** Ph.D, Associate Professor at Kalasalingam Academy of Research and Education [KARE] for her inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on.

**MUNAGA RAKESH**

**9918004073**

**SHAIK MAHABUB SHAARIIEF**

**9918004108**

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# Chapter 1

# INTRODUCTION

This application is helpful to conduct cricket tournaments in easy way without any disturbances of score mistake if we manually count, we designed this app to reduce man effort it is digital scorecard to calculate scorecard for both teams with their player score,strike rate and how many sixes and fours are hit by the player, and the teams stats and the Bowler stats.By creating this app we learned so many things like editing,coding,etc.In this we used LinearLayout, buttons, Imageview, RelativeLayout,Listview, DataBase to store the data of the players,teams etc. and we used sounds to coin when coin is tossed it produces. When the app is opened at first we used splash screen it displayed for some seconds and it goes to MainActivity then need some data to enter to next screen of ScoreUpdateActivity

### 1.0.1 Objectives

List the objectives of the project work...

1. First imagine how to do
2. Then designed the interface of each Activity how you need
3. Write the xml code
4. To develop the code of your project

5. Rectify the errors

6. At the designing need to implement some required in gradle file

7. Store the images in drawable in res folder

8. Required things need to kept in Manifest

9. Run the emulator to check the is app is working or not

10. If errors rectify it

# Chapter 2

# PROJECT DESCRIPTION

Every Android Applicatin designed in Android Studio is may consists of Activities,Fragments,Differnt types of Layouts such as RecyclerView, CardView, GridView etc. and each of it consists of different xml code it is used to design the interface of App. Here we used Android Studio to create our Application called **OUT!**

In Figure(2.1) The MainActivity is consists of LinearLayout,imageview, buttons.. When the MainActivity opened it need some data like the number of overs to play the game and add number of players to play and imageview it consists of coin to toss, below there are two buttons 1.Start 2.Continue I If we press start the app goes to new new match or If we press continue the app goes to resume match which was played last. Then it goes to ScoreUpdateActivity.

The ScoreUpdateActivity consists of many buttons,LinearLayout

In Figure(2.2), At the top left corner the Team A details consists of runs/wickets and overs and runrate.Then top right corner Team B stats and next to that the details of Team A players i.e the batsman played number of balls and how much is scored and how many boundaries like fours and sixes the players hits and the batsman strikerate.And Then now here the scores to be added manually by tournament organizers i.e if batsman scores one run and then press 1 then the batsman strike changes to another else if batsman scores two runs and then press 2 but here batsman strike doesn't change and so on.If over is complete then only batsman strike changes to another batsman.If the batsman out Now enters the new Batsman and he have strike.Below to the scores the current over is going on.There are 3 buttons like batsmen,bowlers,end. By pressing Batsman new we enters all the batsman names its stores the data in database and same By pressing

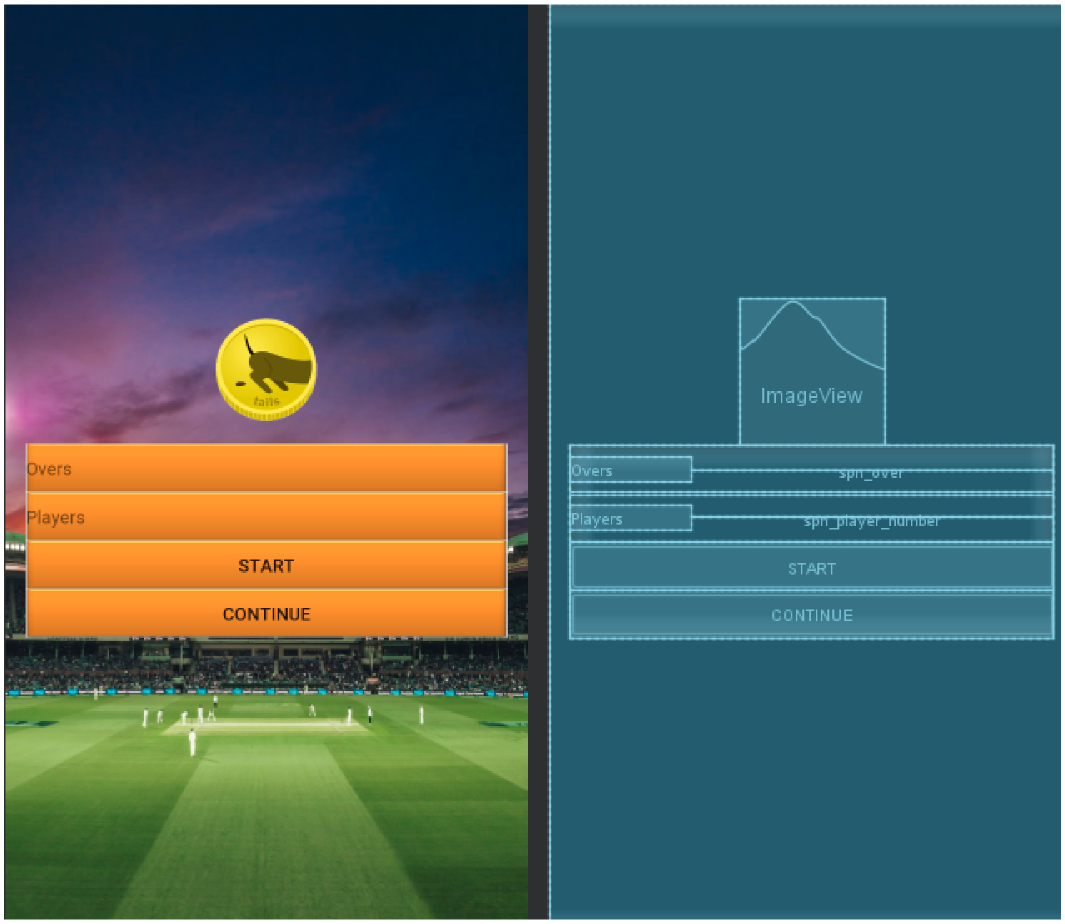


Figure 2.1: Figure MainActivity

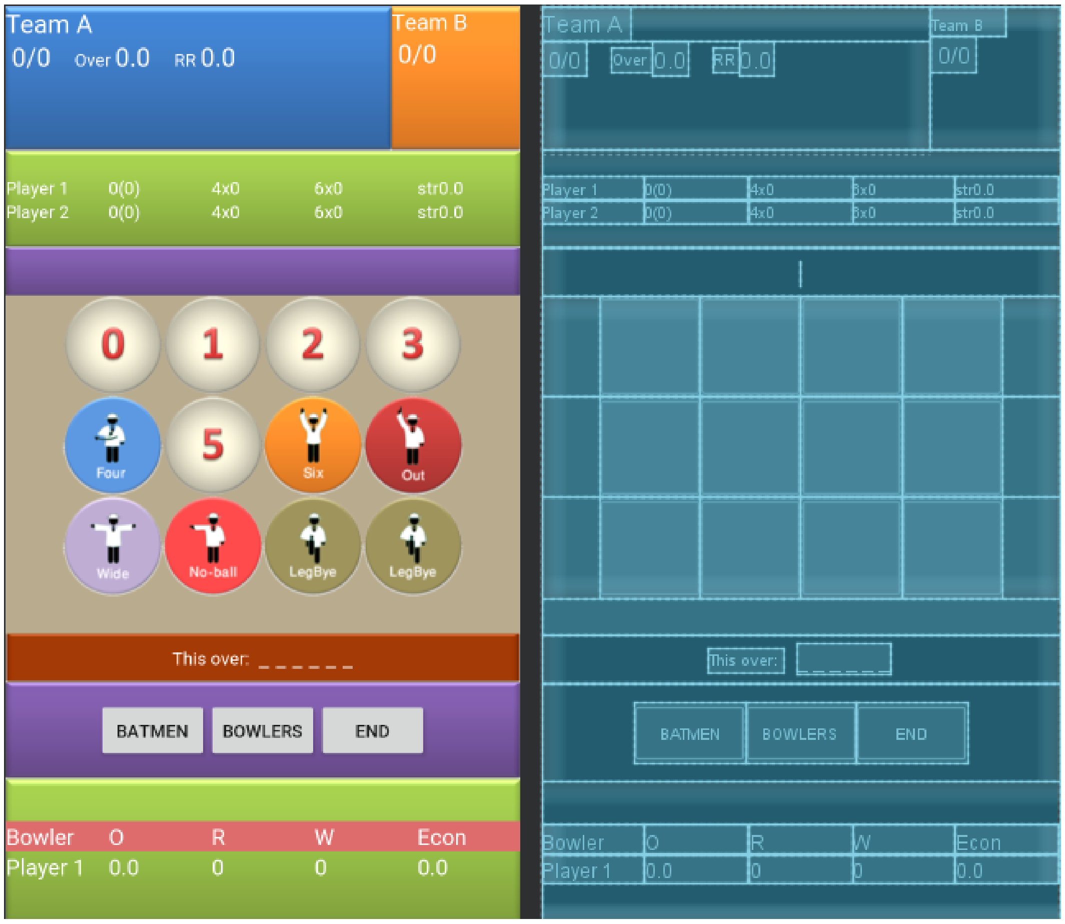


Figure 2.2: Figure ScoreUpdateActivity

bowlers we can enters all the bowlers names, By pressing the end it means that end of first inings and start the second inings. At the top we can change the teams names.At the bottom of interface Bowler details that means the number of overs does the bowler bowls and number of wickets he taken and calculate the economy of bowler.

Running the **OUT!** Application

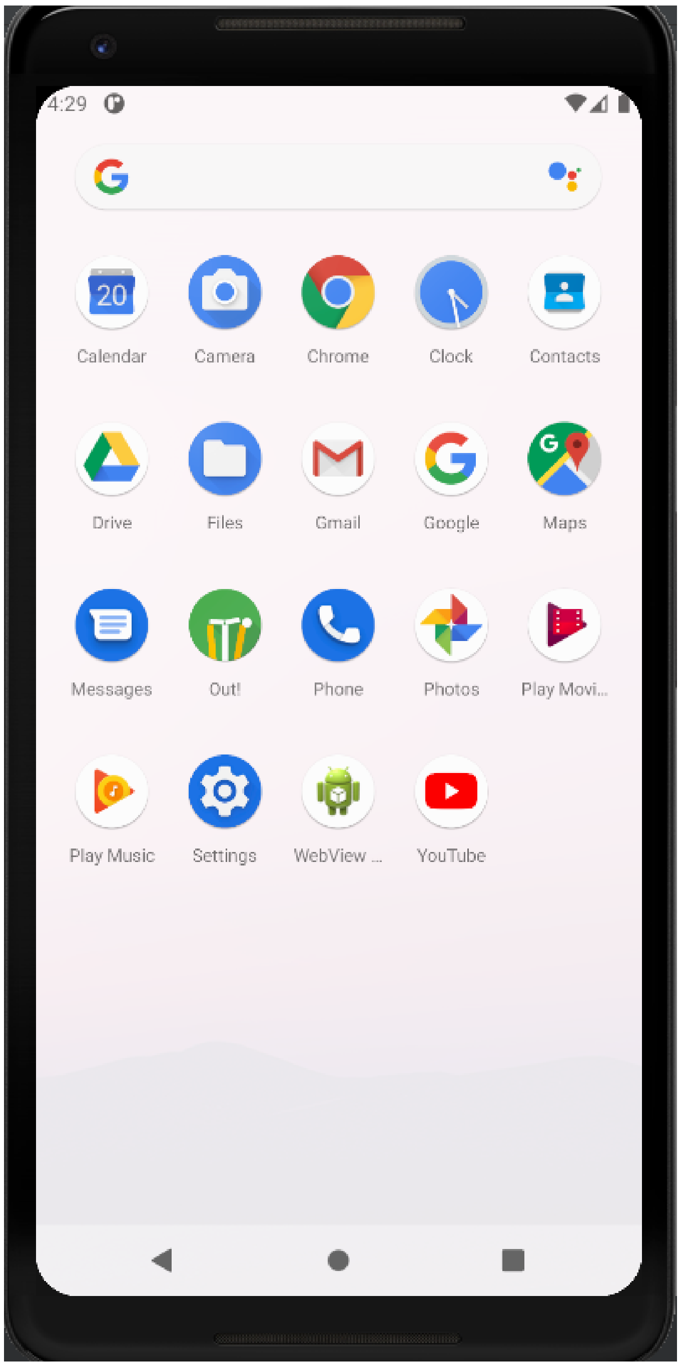


Figure 2.3: Figure When run is done the app installed in mobile

Figure 2.4: Figure It is SplashScreenActivity it displayed when app is open for few seconds

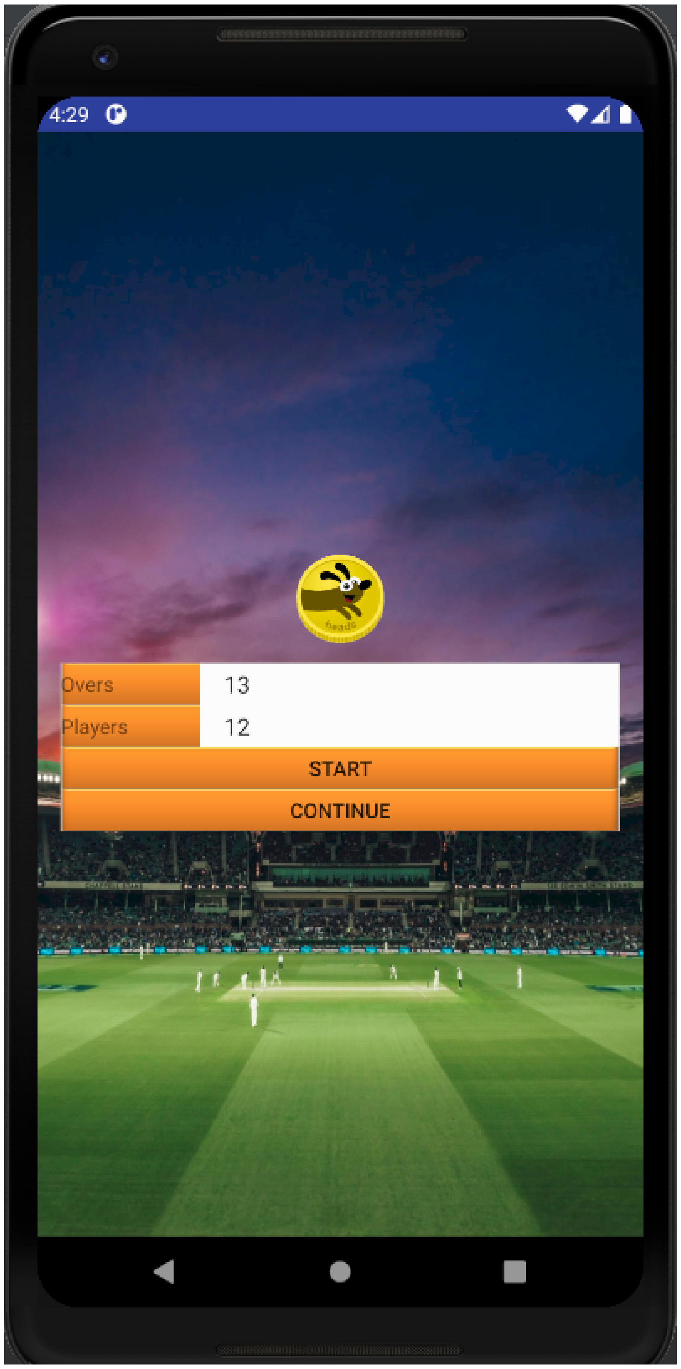
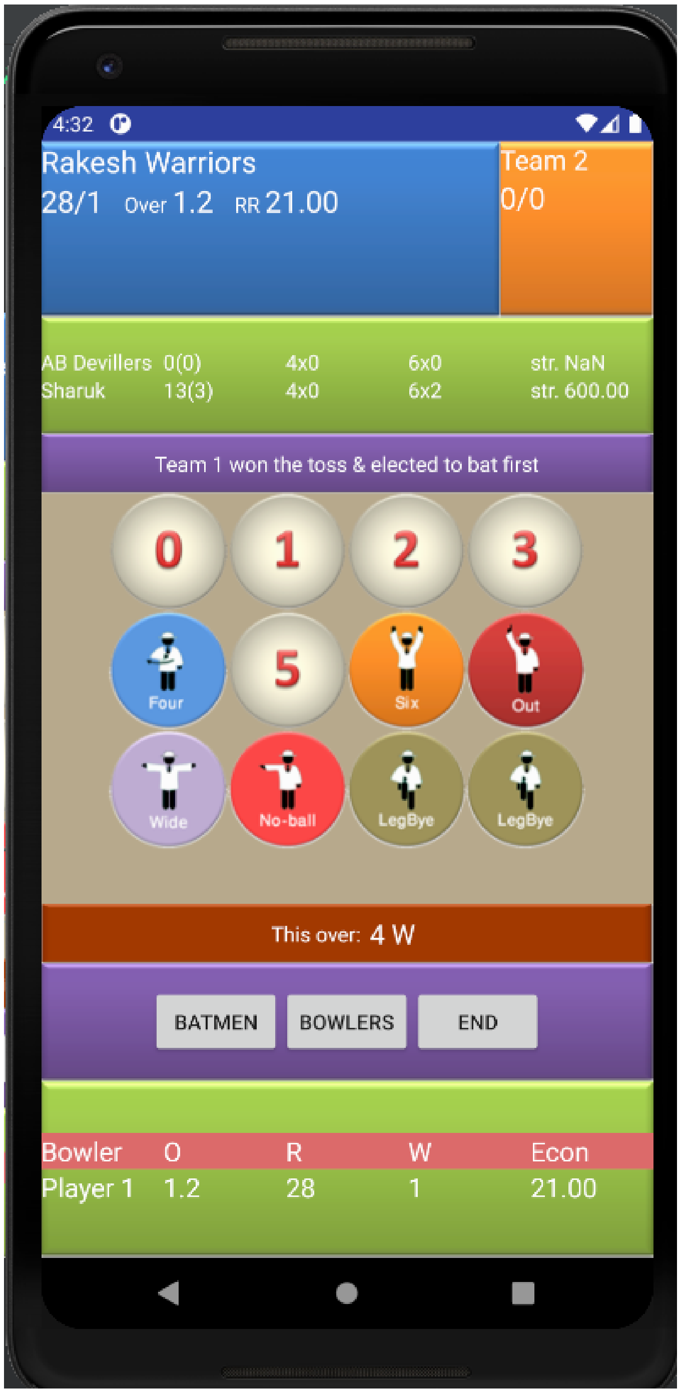


Figure 2.5: Figure After SplashScreenActivity it goes MainActivity



Rakesh Warriors
28/1 Over 1.2 RR 21.00

Team 2
0/0

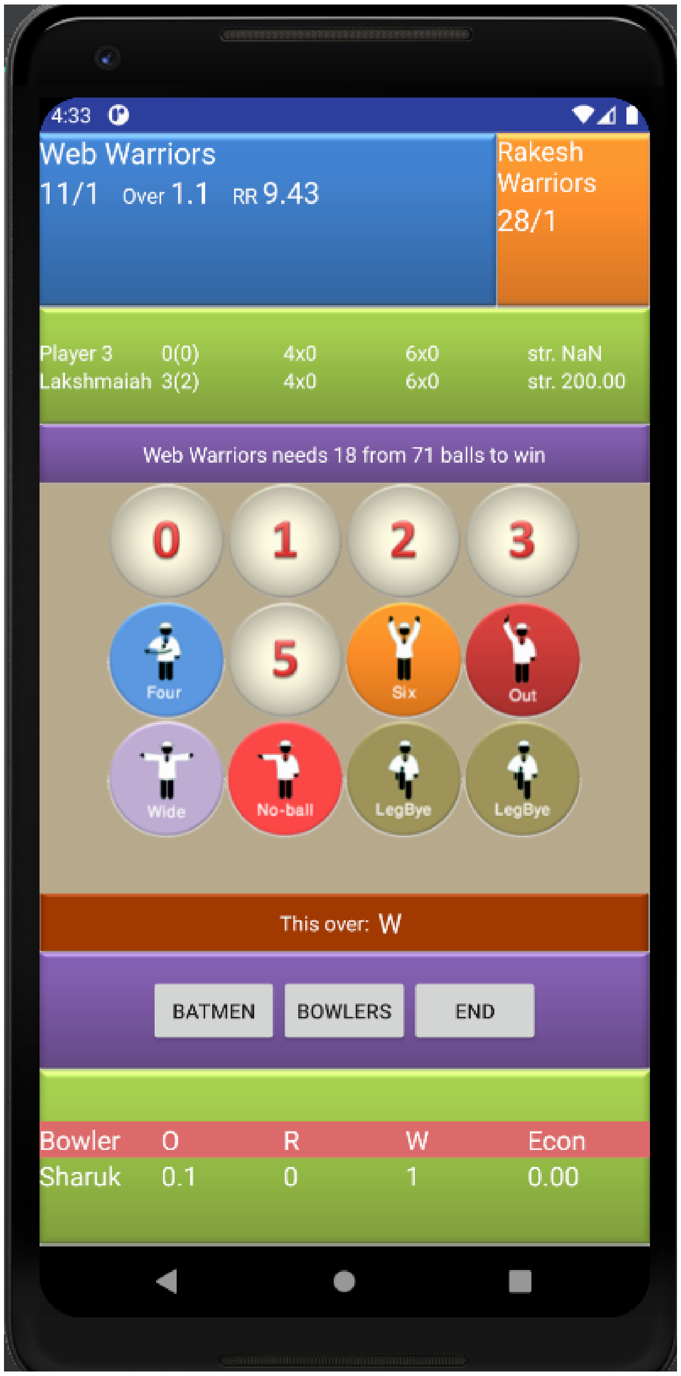
| | | | | |
|---|---|---|---|---|
| AB Devillers | 0(0) | 4x0 | 6x0 | str. NaN |
| Sharuk | 13(3) | 4x0 | 6x2 | str. 600.00 |

Team 1 won the toss & elected to bat first

This over: 4 W

| Bowler | O | R | W | Econ |
|---|---|---|---|---|
| Player 1 | 1.2 | 28 | 1 | 21.00 |

Figure 2.6: Figure In MainActivity press the start and goes to ScoreUpdateActivity and here we enter the scorecard for Team A



4:33

Web Warriors
11/1 Over 1.1 RR 9.43

Rakesh Warriors
28/1

|  |  |  |  |  |
|---|---|---|---|---|
| Player 3 | 0(0) | 4x0 | 6x0 | str. NaN |
| Lakshmaiah | 3(2) | 4x0 | 6x0 | str. 200.00 |

Web Warriors needs 18 from 71 balls to win

This over: W

| Bowler | O | R | W | Econ |
|---|---|---|---|---|
| Sharuk | 0.1 | 0 | 1 | 0.00 |

Figure 2.7: Figure After press the end button it goes to 2nd Second inings

# Chapter 3

# CONCLUSION

This project augments both the normal brain and the defective brain. It displays the outer view and the internal view of both the brain structures. The application is also accompanied with background voice which explains about the detailed description of the internal brain parts. Thus some added features like scaling, rotation and translation are applied to the AR models so that it would help the user to view the object in the desirable angle.

The future enhancement may include the addition of dissection property to the augmented object. When the user has the ability to dissect the objects it would the understandability of the structure. This can also be applied for other deformities in the human anatomy.

We take many References like brother's help, github, udemy, youtube, google..

# Appendices

**SOURCE CODE**

```
package com.example.out;

import android.app.Activity;
import android.content.Intent;
import android.graphics.drawable.AnimationDrawable;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;
import java.util.Random;

public class MainActivity extends Activity implements
    ↪ View.OnClickListener {

    private Button btnContinue;

    private ImageView tossView;

    private Button btnStart;

    private Spinner spnPlayerNum;
    private Spinner spnOver;

    private AnimationDrawable tossAnimation;


    @Override
    protected void onCreate(Bundle savedInstanceState)
        ↪ {
         super.onCreate(savedInstanceState);
```

```
setContentView(R.layout.activity_main);


tossView = findViewById(R.id.btn_toss);
btnContinue = findViewById(R.id.btn_continue);
btnStart = findViewById(R.id.btn_start);

spnOver = findViewById(R.id.spn_over);
spnPlayerNum = findViewById(R.id.
    ↪ spn_player_number);

btnContinue.setOnClickListener(this);
btnStart.setOnClickListener(this);

tossView.setBackgroundResource(R.drawable.
    ↪ toss_animation_tail);

tossView.setOnClickListener(new View.
    ↪ OnClickListener() {
     public void onClick(final View v) {
          MediaPlayer player = MediaPlayer.create
              ↪ (MainActivity.this, R.raw.
              ↪ coinflip);
          player.start();

          final CustomAnimationDrawableNew cad =
              ↪ new CustomAnimationDrawableNew(
                   (AnimationDrawable) tossView.
                       ↪ getBackground()) {
               @Override
               void onAnimationFinish() {

                    Random random = new Random();
                    int r = random.nextInt();
                    if (r % 2 == 0) {
                         v.setBackgroundResource(R.
                             ↪ drawable.
                             ↪ toss_animation_tail);
                    } else {
```

```
                              v.setBackgroundResource(R.
                                  ↪ drawable.
                                  ↪ toss_animation_head);
                         }
                     }
                 };


                 v.setBackgroundDrawable(cad);


                 cad.start();
             }
         });

         List<String> overs = new ArrayList<>();
         for (int i = 1; i <= 50; i++) {
             overs.add(String.valueOf(i));
         }

         ArrayAdapter<String> adapter = new ArrayAdapter
             ↪ <>(this, android.R.layout.
             ↪ simple_list_item_1, overs);
         spnOver.setAdapter(adapter);

         List<String> players = new ArrayList<>();
         for (int i = 1; i <= 20; i++) {
             players.add(String.valueOf(i));
         }

         adapter = new ArrayAdapter<>(this, android.R.
             ↪ layout.simple_list_item_1, players);
         spnPlayerNum.setAdapter(adapter);
         spnPlayerNum.setSelection(11);
         spnOver.setSelection(12);

     }

     @Override
     protected void onPause() {
```

```
    super.onPause();
}

@Override
public void onClick(View v) {
    if (v == btnContinue) {
        Intent intent = new Intent(this,
            ↪ ScoreUpdateActivity.class);
        intent.putExtra("CONTINUE", true);
        intent.putExtra("START", false);
        Datasource db = new Datasource(this);
        db.open();
        try {
            db.getTeamScore(1);
            startActivity(intent);
        } catch (Exception e) {
            Toast.makeText(this, "No␣match␣is␣
                ↪ running.␣Start␣match␣first",
                ↪ Toast.LENGTH_SHORT).show();
        }
    } else if (v == btnStart) {
        Intent intent = new Intent(this,
            ↪ ScoreUpdateActivity.class);
        intent.putExtra("OVERS", Integer.parseInt(
            ↪ spnOver.getSelectedItem().toString())
            ↪ );
        intent.putExtra("PLAYERS", Integer.parseInt
            ↪ (spnPlayerNum.getSelectedItem().
            ↪ toString()));
        intent.putExtra("START", true);
        startActivity(intent);
    }
}

public abstract static class
   ↪ CustomAnimationDrawableNew extends
   ↪ AnimationDrawable {


    Handler mAnimationHandler;
```

```
public CustomAnimationDrawableNew(
    ↪ AnimationDrawable aniDrawable) {

    for (int i = 0; i < aniDrawable.
        ↪ getNumberOfFrames(); i++) {
        this.addFrame(aniDrawable.getFrame(i),
            ↪ aniDrawable.getDuration(i));
    }
}

@Override
public void start() {
    super.start();

    mAnimationHandler = new Handler();
    mAnimationHandler.postDelayed(new Runnable
        ↪ () {

        public void run() {
            onAnimationFinish();
        }
    }, getTotalDuration());

}


public int getTotalDuration() {

    int iDuration = 0;

    for (int i = 0; i < this.getNumberOfFrames
        ↪ (); i++) {
        iDuration += this.getDuration(i);
    }

    return iDuration;
}
```

```
        abstract void onAnimationFinish();
    }
}
```

```
package com.example.out;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class ScoreUpdateActivity extends Activity
   ↪ implements View.OnClickListener {

    private Datasource datasource;

    private Button btnZero;
    private Button btnOne;
    private Button btnTwo;
    private Button btnThree;
    private Button btnFour;
    private Button btnFive;
    private Button btnSix;

    private TextView tvTeamOneName;
    private TextView tvTeamTwoName;

    private TextView tvTeamOneScore;
    private TextView tvTeamTwoScore;
    private TextView tvTeamOneOver;
    private TextView tvTeamOneRunRate;

    private Button btnOut;
    private Button btnEnd;
private Button btnWd;
private Button btnNb;
private Button btnBye;
private Button btnLegBye;

private Button btnBowler;
private Button btnBatsman;

private TextView tvThisOver;

private TextView tvStriker;
private TextView tvStrikerScore;
private TextView tvStrikerFour;
private TextView tvStrikerSix;
private TextView tvStrikerStrikeRate;

private TextView tvNonStriker;
private TextView tvNonStrikerScore;
private TextView tvNonStrikerFour;
private TextView tvNonStrikerSix;
private TextView tvNonStrikerStrikeRate;

private TextView tvBowlerName;
private TextView tvBowlerOver;
private TextView tvBowlerRun;
private TextView tvBowlerWicket;
private TextView tvBowlerEconomyRate;

private TextView tvCommentary;

private int strikerNo;
private int nonStrikerNo;

private boolean isTeamTwoInBatting;

private StringBuilder thisOver;

private int bowlerNo;

private Team teamOne;
private Team teamTwo;

private int totalBall = 300;


@SuppressLint("SetTextI18n")
@Override
protected void onCreate(Bundle savedInstanceState)
   ↪ {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
       ↪ activity_score_update_activity);

    datasource = new Datasource(this);

    btnZero = findViewById(R.id.zero_run_button);
    btnOne = findViewById(R.id.one_run_button);
    btnTwo = findViewById(R.id.two_run_button);
    btnThree = findViewById(R.id.three_run_button);
    btnFour = findViewById(R.id.four_run_button);
    btnFive = findViewById(R.id.five_run_button);
    btnSix = findViewById(R.id.six_run_button);

    btnBowler = findViewById(R.id.btn_bowler);
    btnBatsman = findViewById(R.id.btn_batsman);
    btnOut = findViewById(R.id.btn_out);
    btnEnd = findViewById(R.id.btnEnd);

    btnNb = findViewById(R.id.btn_nb);
    btnWd = findViewById(R.id.btn_wide);
    btnBye = findViewById(R.id.btn_bye);
    btnLegBye = findViewById(R.id.btn_leg_bye);

    btnNb.setOnClickListener(this);
    btnWd.setOnClickListener(this);
    btnBye.setOnClickListener(this);
    btnLegBye.setOnClickListener(this);


    tvTeamOneName = findViewById(R.id.team_one_name
        ↪ );
    tvTeamTwoName = findViewById(R.id.team_two_name
        ↪ );


    tvTeamOneScore = findViewById(R.id.
       ↪ team_one_score);
    tvTeamTwoScore = findViewById(R.id.
       ↪ team_two_score);
    tvTeamOneOver = findViewById(R.id.tv_over);
    tvTeamOneRunRate = findViewById(R.id.tv_runrate
       ↪ );

    tvStriker = findViewById(R.id.strikerName);
    tvStrikerScore = findViewById(R.id.strikerScore
       ↪ );
    tvStrikerFour = findViewById(R.id.strikerFour);
    tvStrikerSix = findViewById(R.id.strikerSix);
    tvStrikerStrikeRate = findViewById(R.id.
       ↪ strikerStrikeRate);

    tvNonStriker = findViewById(R.id.nonStrikerName
       ↪ );
    tvNonStrikerScore = findViewById(R.id.
       ↪ nonStrikerScore);
    tvNonStrikerFour = findViewById(R.id.
       ↪ nonStrikerFour);
    tvNonStrikerSix = findViewById(R.id.
       ↪ nonStrikerSix);
    tvNonStrikerStrikeRate = findViewById(R.id.
       ↪ nonStrikerStrikeRate);

    tvCommentary = findViewById(R.id.commentry);

    tvBowlerName = findViewById(R.id.bowlerName);
    tvBowlerOver = findViewById(R.id.bowlerOver);
    tvBowlerRun = findViewById(R.id.bowlerRun);
    tvBowlerWicket = findViewById(R.id.bowlerWicket
       ↪ );
    tvBowlerEconomyRate = findViewById(R.id.
       ↪ bowlerEconomyRate);

    tvThisOver = findViewById(R.id.thisOver);

    btnZero.setOnClickListener(this);
    btnOne.setOnClickListener(this);
    btnTwo.setOnClickListener(this);
    btnThree.setOnClickListener(this);
    btnFour.setOnClickListener(this);
    btnFive.setOnClickListener(this);
    btnSix.setOnClickListener(this);

    btnBatsman.setOnClickListener(this);
    btnBowler.setOnClickListener(this);
    btnEnd.setOnClickListener(this);
    btnOut.setOnClickListener(this);

    tvTeamOneName.setOnClickListener(this);
    tvStriker.setOnClickListener(this);
    tvNonStriker.setOnClickListener(this);
    tvBowlerName.setOnClickListener(this);

    boolean continueFlag = getIntent().
       ↪ getBooleanExtra("CONTINUE", false);
    boolean startFlag = getIntent().getBooleanExtra
       ↪ ("START", false);

    if (startFlag) {
       teamOne = new Team(1);
       teamTwo = new Team(2);
       int playerNum = getIntent().getIntExtra("
           ↪ PLAYERS", 11);
       for (int i = 0; i < playerNum; i++) {
           teamOne.players.add(new Player(i + 1));
           teamTwo.players.add(new Player(i + 1));
       }
       datasource.open();
       datasource.addTeamScore(teamOne);
       datasource.addTeamScore(teamTwo);
       datasource.insertPlayerOne(teamOne.players)
```

```
            ↪ ;
        datasource.insertPlayerTwo(teamTwo.players)
            ↪ ;
        datasource.close();
        totalBall = getIntent().getIntExtra("OVERS"
            ↪ , 50) * 6;
    }

    if (continueFlag) {
        datasource.open();
        teamOne = datasource.getTeamScore(1);
        teamTwo = datasource.getTeamScore(2);
        teamOne.players = datasource.
            ↪ getPlayerOneList();
        teamTwo.players = datasource.
            ↪ getPlayerTwoList();
        datasource.close();

        totalBall = teamOne.ball;
    }

    strikerNo = 0;
    nonStrikerNo = 1;
    bowlerNo = 0;

    tvTeamOneName.setText(teamOne.name);
    tvTeamTwoName.setText(teamTwo.name);

    thisOver = new StringBuilder();
    tvCommentary.setText(teamOne.name + " won the 
        ↪ toss & elected to bat first");

}

@Override
protected void onPause() {
    super.onPause();
    datasource.open();

    teamOne.id = 1;
```

```
        teamTwo.id = 2;

        datasource.updateTeamScore(teamOne);
        datasource.updateTeamScore(teamTwo);
        datasource.updatePlayerOne(teamOne.players);
        datasource.updatePlayerTwo(teamTwo.players);

        datasource.close();
    }

    @Override
    public void onClick(View v) {
        if (v == btnEnd) {
            swapTeam();
            strikerNo = 0;
            nonStrikerNo = 1;
            bowlerNo = 0;
            isTeamTwoInBatting = true;
            tvTeamOneName.setText(teamOne.name);
            tvTeamTwoName.setText(teamTwo.name);
        } else if (v == btnZero) {
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            thisOver.append("0␣");
            if (teamOne.overBall() == 0) {
                Toast.makeText(this, "Over␣end.␣Change␣
                    ↪ the␣Bowler", Toast.LENGTH_LONG).
                    ↪ show();
            }
        } else if (v == btnOne) {
            teamOne.run++;
            teamOne.ball++;
            teamOne.players.get(strikerNo).run++;
            teamOne.players.get(strikerNo).ball++;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun++;
            thisOver.append("1␣");
            swapBatsman();
        } else if (v == btnTwo) {
            teamOne.run += 2;
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamOne.players.get(strikerNo).run += 2;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun +=
                ↪ 2;
            thisOver.append("2␣");
        } else if (v == btnThree) {
            teamOne.run += 3;
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamOne.players.get(strikerNo).run += 3;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun +=
                ↪ 3;
            thisOver.append("3␣");
            swapBatsman();
        } else if (v == btnFour) {
            teamOne.run += 4;
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamOne.players.get(strikerNo).run += 4;
            teamOne.players.get(strikerNo).four++;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun +=
                ↪ 4;
            thisOver.append("4␣");
        } else if (v == btnFive) {
            teamOne.run += 5;
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamOne.players.get(strikerNo).run += 5;

            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun +=
                ↪ 5;
            thisOver.append("5␣");
        } else if (v == btnSix) {
            teamOne.run += 6;
            teamOne.ball++;
            teamOne.players.get(strikerNo).ball++;
            teamOne.players.get(strikerNo).run += 6;
            teamOne.players.get(strikerNo).six++;

            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamTwo.players.get(bowlerNo).bowlerRun +=
                ↪ 6;
            thisOver.append("6␣");
        } else if (v == btnOut) {
            teamOne.wicket++;
            teamTwo.players.get(bowlerNo).wickets++;
            teamOne.ball++;
            teamTwo.players.get(bowlerNo).bowlerBall++;
            teamOne.players.add(new Player());
            thisOver.append("W␣");
            strikerNo = teamOne.wicket + 1;
        } else if (v == btnNb) {
            teamOne.run++;
            teamTwo.players.get(bowlerNo).bowlerRun++;
            thisOver.append("Nb␣");
        } else if (v == btnWd) {
            teamOne.run++;
            teamTwo.players.get(bowlerNo).bowlerRun++;
            thisOver.append("Wd␣");
        } else if (v == btnBye) {
            teamOne.run++;
            teamTwo.players.get(bowlerNo).bowlerRun++;
            thisOver.append("1b␣");
        } else if (v == btnLegBye) {
            teamOne.run++;
            teamTwo.players.get(bowlerNo).bowlerRun++;
            thisOver.append("1b␣");
        } else if (v == tvStriker) {
            setNameDialog(tvStriker, teamOne.players.
                ↪ get(strikerNo).name);
        } else if (v == tvNonStriker) {
            setNameDialog(tvNonStriker, teamOne.players
                ↪ .get(nonStrikerNo).name);
        } else if (v == tvTeamOneName) {
```

```
        setNameDialog(tvTeamOneName, teamOne.name);
    } else if (v == tvBowlerName) {
        setNameDialog(tvBowlerName, teamTwo.players
            ↪ .get(bowlerNo).name);
    } else if (v == btnBowler) {
        playerListDialog(teamTwo);
    } else if (v == btnBatsman) {
        playerListDialog(teamOne);
    }

    if (teamTwo.players.get(bowlerNo).bowlerBall %
        ↪ 6 == 1) {
        char c = thisOver.charAt(thisOver.length()
            ↪ - 2);
        thisOver = new StringBuilder();
        thisOver.append(c + "␣");
    }

    if (isTeamTwoInBatting) {
        int targetRun = teamTwo.run - teamOne.run +
            ↪  1;
        int ballsRemain = totalBall - teamOne.ball;
        tvCommentary.setText(teamOne.name + "␣needs
            ↪ ␣" + targetRun + "␣from␣" +
            ↪ ballsRemain + "␣balls␣to␣win");
    }

    teamOne.players.get(strikerNo).strikeRate();

    tvThisOver.setText(thisOver);

    tvStrikerScore.setText(teamOne.players.get(
        ↪ strikerNo).run + "(" + teamOne.players.
        ↪ get(strikerNo).ball + ")");
    tvStrikerFour.setText("4x" + teamOne.players.
        ↪ get(strikerNo).four);
    tvStrikerSix.setText("6x" + teamOne.players.get
        ↪ (strikerNo).six);
    tvStrikerStrikeRate.setText("str.␣" + String.
        ↪ format("%.2f", teamOne.players.get(
        ↪ strikerNo).strikeRate));

    tvNonStrikerScore.setText(teamOne.players.get(
        ↪ nonStrikerNo).run + "(" + teamOne.players
        ↪ .get(nonStrikerNo).ball + ")");
    tvNonStrikerFour.setText("4x" + teamOne.players
        ↪ .get(nonStrikerNo).four);
    tvNonStrikerSix.setText("6x" + teamOne.players.
        ↪ get(nonStrikerNo).six);
    tvNonStrikerStrikeRate.setText("str.␣" + String
        ↪ .format("%.2f", teamOne.players.get(
        ↪ nonStrikerNo).strikeRate));

    tvTeamOneScore.setText(teamOne.run + "/" +
        ↪ teamOne.wicket);
    tvTeamOneOver.setText(teamOne.over() + "." +
        ↪ teamOne.overBall());
    tvTeamOneRunRate.setText(String.format("%.2f",
        ↪ teamOne.runRate()));
    tvTeamTwoScore.setText(teamTwo.run + "/" +
        ↪ teamTwo.wicket);

    tvBowlerName.setText(teamTwo.players.get(
        ↪ bowlerNo).name);
    tvBowlerOver.setText(teamTwo.players.get(
        ↪ bowlerNo).over() + "." + teamTwo.players.
        ↪ get(bowlerNo).overBall());
    tvBowlerRun.setText(String.valueOf(teamTwo.
        ↪ players.get(bowlerNo).bowlerRun));
    tvBowlerEconomyRate.setText(String.format("%.2f
        ↪ ", teamTwo.players.get(bowlerNo).
        ↪ economyRate()));
    tvBowlerWicket.setText(String.valueOf(teamTwo.
        ↪ players.get(bowlerNo).wickets));

    tvStriker.setText(teamOne.players.get(strikerNo
        ↪ ).name);
    tvNonStriker.setText(teamOne.players.get(
        ↪ nonStrikerNo).name);
}
```

```
private void swapBatsman() {
    int temp = strikerNo;
    strikerNo = nonStrikerNo;
    nonStrikerNo = temp;
}

private void swapTeam() {
    Team temp = teamOne;
    teamOne = teamTwo;
    teamTwo = temp;
}

private void setNameDialog(final TextView view,
  ↪ String name) {
    AlertDialog.Builder builder = new AlertDialog.
       ↪ Builder(this);
    final EditText changeName = new EditText(this);
    changeName.setText(name);
    builder.setView(changeName);
    builder.setTitle("Set␣Name");
    builder.setPositiveButton("OK",
            new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface
                   ↪ dialog, int which) {
                    if (view == tvTeamOneName) {
                        teamOne.name = changeName.
                           ↪ getText().toString();
                    } else if (view == tvStriker) {
                        teamOne.players.get(
                           ↪ strikerNo).name =
                           ↪ changeName.getText().
                           ↪ toString();
                    } else if (view == tvNonStriker
                       ↪ ) {
                        teamOne.players.get(
                           ↪ nonStrikerNo).name =
                           ↪ changeName.getText().
                           ↪ toString();
                    } else if (view == tvBowlerName
                        ↪ ) {
                         teamTwo.players.get(
                            ↪ bowlerNo).name =
                            ↪ changeName.getText().
                            ↪ toString();
                    }
                    view.setText(changeName.getText
                        ↪ ().toString());
                }
            });
    builder.setNeutralButton("Cancel",
            new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface
                    ↪  dialog, int which) {
                    dialog.dismiss();
                }
            });
    AlertDialog simpleDialog = builder.create();
    simpleDialog.show();
}

private void playerListDialog(Team team) {

    String[] playerList = new String[team.players.
        ↪ size()];
    for (int i = 0; i < team.players.size(); i++) {
        playerList[i] = team.players.get(i).name;
    }

    AlertDialog.Builder builder = new AlertDialog.
        ↪ Builder(this);
    builder.setTitle("Player␣List");
    builder.setCancelable(true);
    builder.setItems(playerList, new
        ↪ DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog,
            ↪  int which) {
            bowlerNo = which;
            tvBowlerName.setText(teamTwo.players.
                  ↪ get(bowlerNo).name);
                tvBowlerOver.setText(teamTwo.players.
                  ↪ get(bowlerNo).over() + "." +
                  ↪ teamTwo.players.get(bowlerNo).
                  ↪ overBall());
                tvBowlerRun.setText(String.valueOf(
                  ↪ teamTwo.players.get(bowlerNo).
                  ↪ bowlerRun));
                tvBowlerEconomyRate.setText(String.
                  ↪ format("%.2f", teamTwo.players.
                  ↪ get(bowlerNo).economyRate()));
                tvBowlerWicket.setText(String.valueOf(
                  ↪ teamTwo.players.get(bowlerNo).
                  ↪ wickets));
            }
        });
        builder.setNegativeButton("Cancel",
                new DialogInterface.OnClickListener() {

                  @Override
                  public void onClick(DialogInterface
                    ↪ dialog, int which) {
                    dialog.dismiss();
                  }
                });
        AlertDialog mapTypeDialog = builder.create();
        mapTypeDialog.show();
    }

}
```

```
package com.example.out;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBOpenHelper extends SQLiteOpenHelper {
    private static final String DATABASE_NAME = "cric.
      ↪ db";
```

```
private static final int DATABASE_VERSION = 1;
public static final String TABLE_TEAM = "Team";

public static final String TABLE_TEAM_ONE_PLAYER =
    ↪ "Player1";
public static final String TABLE_TEAM_TWO_PLAYER =
    ↪ "Player2";


public static final String COLUMN_PLAYER_ID= "id";
public static final String COLUMN_PLAYER_NAME= "
    ↪ Name";
public static final String COLUMN_PLAYER_RUN= "Run"
    ↪ ;
public static final String COLUMN_PLAYER_BALL= "
    ↪ Ball";
public static final String COLUMN_PLAYER_FOUR= "
    ↪ Four";
public static final String COLUMN_PLAYER_SIX= "Six"
    ↪ ;
public static final String
    ↪ COLUMN_PLAYER_BOWLER_BALL= "BowlerBall";
public static final String COLUMN_PLAYER_BOWLER_RUN
    ↪ = "BowlerRun";
public static final String
    ↪ COLUMN_PLAYER_BOWLER_WICKET = "Wickets";

public static final String COLUMN_TEAM_ID = "id";
public static final String COLUMN_TEAM_NAME = "name
    ↪ ";
public static final String COLUMN_TEAM_RUN = "run";
public static final String COLUMN_TEAM_BALL = "ball
    ↪ ";
public static final String COLUMN_TEAM_WICKET = "
    ↪ wicket";

public DBOpenHelper(Context context) {
    super(context, DATABASE_NAME, null,
        ↪ DATABASE_VERSION);
```

```
}

@Override
public void onCreate(SQLiteDatabase db) {
    String CREATE_TEAM_TABLE = "CREATE␣TABLE␣" +
       ↪ TABLE_TEAM + "("
            + COLUMN_TEAM_ID + "␣INTEGER␣PRIMARY␣
               ↪ KEY," + COLUMN_TEAM_NAME + "␣TEXT
               ↪ ,"
            + COLUMN_TEAM_RUN + "␣INTEGER,␣" +
               ↪ COLUMN_TEAM_BALL + "␣INTEGER,␣" +
               ↪  COLUMN_TEAM_WICKET + "␣INTEGER"
               ↪ + ")";
    String CREATE_PLAYER1_TABLE = "CREATE␣TABLE␣" +
       ↪  TABLE_TEAM_ONE_PLAYER + "("
            + COLUMN_PLAYER_ID + "␣INTEGER␣PRIMARY␣
               ↪ KEY," + COLUMN_PLAYER_NAME + "␣
               ↪ TEXT,"
            + COLUMN_PLAYER_RUN + "␣INTEGER,␣" +
               ↪ COLUMN_PLAYER_BALL + "␣INTEGER,␣"
               ↪   + COLUMN_PLAYER_FOUR
            + "␣INTEGER,␣" + COLUMN_PLAYER_SIX + "␣
               ↪ INTEGER,␣" +
               ↪ COLUMN_PLAYER_BOWLER_RUN  + "␣
               ↪ INTEGER,␣"
            + COLUMN_PLAYER_BOWLER_BALL + "␣INTEGER
               ↪  ,␣" + COLUMN_PLAYER_BOWLER_WICKET
               ↪  +  "␣INTEGER" + ")";

    String CREATE_PLAYER2_TABLE = "CREATE␣TABLE␣" +
       ↪  TABLE_TEAM_TWO_PLAYER + "("
            + COLUMN_PLAYER_ID + "␣INTEGER␣PRIMARY␣
               ↪ KEY," + COLUMN_PLAYER_NAME + "␣
               ↪ TEXT,"
            + COLUMN_PLAYER_RUN + "␣INTEGER,␣" +
               ↪ COLUMN_PLAYER_BALL + "␣INTEGER,␣"
               ↪   + COLUMN_PLAYER_FOUR
            + "␣INTEGER,␣" + COLUMN_PLAYER_SIX + "␣
               ↪ INTEGER,␣" +
               ↪ COLUMN_PLAYER_BOWLER_RUN  + "␣
```

```
                    ↪ INTEGER,␣"
                 + COLUMN_PLAYER_BOWLER_BALL + "␣INTEGER
                    ↪ ,␣" + COLUMN_PLAYER_BOWLER_WICKET
                    ↪ + "␣INTEGER" + ")";

        db.execSQL(CREATE_TEAM_TABLE);
        db.execSQL(CREATE_PLAYER1_TABLE);
        db.execSQL(CREATE_PLAYER2_TABLE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int
       ↪ oldVersion, int newVersion) {
        // Drop older table if existed
        db.execSQL("DROP␣TABLE␣IF␣EXISTS␣" + TABLE_TEAM
           ↪ );
        onCreate(db);
    }
}
```

```
package com.example.out;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import java.util.ArrayList;
import java.util.List;

public class Datasource {
    private SQLiteDatabase database;
    private SQLiteOpenHelper dbhelper;

    public Datasource(Context context) {
        dbhelper = new DBOpenHelper(context);
    }

    public void addTeamScore(Team team) {
```

```
    ContentValues values = new ContentValues();
    values.put(DBOpenHelper.COLUMN_TEAM_ID, team.id
       ↪ );
    values.put(DBOpenHelper.COLUMN_TEAM_NAME, team.
       ↪ name);
    values.put(DBOpenHelper.COLUMN_TEAM_RUN, team.
       ↪ run);
    values.put(DBOpenHelper.COLUMN_TEAM_BALL, team.
       ↪ ball);
    values.put(DBOpenHelper.COLUMN_TEAM_WICKET,
       ↪ team.wicket);

    database.insert(DBOpenHelper.TABLE_TEAM, null,
       ↪ values);
}

public void insertPlayerOne(List<Player> players) {
    for (Player player : players) {
        ContentValues values = new ContentValues();
        values.put(DBOpenHelper.COLUMN_PLAYER_ID,
           ↪ player.id);
        values.put(DBOpenHelper.COLUMN_PLAYER_NAME,
           ↪  player.name);
        values.put(DBOpenHelper.COLUMN_PLAYER_RUN,
           ↪ player.run);
        values.put(DBOpenHelper.COLUMN_PLAYER_BALL,
           ↪  player.ball);
        values.put(DBOpenHelper.COLUMN_PLAYER_FOUR,
           ↪  player.four);
        values.put(DBOpenHelper.COLUMN_PLAYER_SIX,
           ↪ player.six);
        values.put(DBOpenHelper.
           ↪ COLUMN_PLAYER_BOWLER_RUN, player.
           ↪ bowlerRun);
        values.put(DBOpenHelper.
           ↪ COLUMN_PLAYER_BOWLER_BALL, player.
           ↪ bowlerBall);
        values.put(DBOpenHelper.
           ↪ COLUMN_PLAYER_BOWLER_WICKET, player.
           ↪ wickets);
```

```
            database.insert(DBOpenHelper.
               ↪ TABLE_TEAM_ONE_PLAYER, null, values);
        }
    }

    public void insertPlayerTwo(List<Player> players) {
        for (Player player : players) {
            ContentValues values = new ContentValues();
            values.put(DBOpenHelper.COLUMN_PLAYER_ID,
               ↪ player.id);
            values.put(DBOpenHelper.COLUMN_PLAYER_NAME,
               ↪  player.name);
            values.put(DBOpenHelper.COLUMN_PLAYER_RUN,
               ↪ player.run);
            values.put(DBOpenHelper.COLUMN_PLAYER_BALL,
               ↪  player.ball);
            values.put(DBOpenHelper.COLUMN_PLAYER_FOUR,
               ↪  player.four);
            values.put(DBOpenHelper.COLUMN_PLAYER_SIX,
               ↪ player.six);
            values.put(DBOpenHelper.
               ↪ COLUMN_PLAYER_BOWLER_RUN, player.
               ↪ bowlerRun);
            values.put(DBOpenHelper.
               ↪ COLUMN_PLAYER_BOWLER_BALL, player.
               ↪ bowlerBall);
            values.put(DBOpenHelper.
               ↪ COLUMN_PLAYER_BOWLER_WICKET, player.
               ↪ wickets);
            database.insert(DBOpenHelper.
               ↪ TABLE_TEAM_TWO_PLAYER, null, values);
        }
    }

    public void updatePlayerTwo(List<Player> players) {
        for (Player player : players) {
            ContentValues values = new ContentValues();
            values.put(DBOpenHelper.COLUMN_PLAYER_ID,
               ↪ player.id);
            values.put(DBOpenHelper.COLUMN_PLAYER_NAME,
            ↪  player.name);
        values.put(DBOpenHelper.COLUMN_PLAYER_RUN,
            ↪ player.run);
        values.put(DBOpenHelper.COLUMN_PLAYER_BALL,
            ↪  player.ball);
        values.put(DBOpenHelper.COLUMN_PLAYER_FOUR,
            ↪  player.four);
        values.put(DBOpenHelper.COLUMN_PLAYER_SIX,
            ↪ player.six);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_RUN, player.
            ↪ bowlerRun);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_BALL, player.
            ↪ bowlerBall);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_WICKET, player.
            ↪ wickets);
        database.update(DBOpenHelper.
            ↪ TABLE_TEAM_TWO_PLAYER, values, "id="
            ↪ + player.id, null);
    }
}

public void updatePlayerOne(List<Player> players) {
    for (Player player : players) {
        ContentValues values = new ContentValues();
        values.put(DBOpenHelper.COLUMN_PLAYER_ID,
            ↪ player.id);
        values.put(DBOpenHelper.COLUMN_PLAYER_NAME,
            ↪  player.name);
        values.put(DBOpenHelper.COLUMN_PLAYER_RUN,
            ↪ player.run);
        values.put(DBOpenHelper.COLUMN_PLAYER_BALL,
            ↪  player.ball);
        values.put(DBOpenHelper.COLUMN_PLAYER_FOUR,
            ↪  player.four);
        values.put(DBOpenHelper.COLUMN_PLAYER_SIX,
            ↪ player.six);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_RUN, player.
            ↪ bowlerRun);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_BALL, player.
            ↪ bowlerBall);
        values.put(DBOpenHelper.
            ↪ COLUMN_PLAYER_BOWLER_WICKET, player.
            ↪ wickets);
        database.update(DBOpenHelper.
            ↪ TABLE_TEAM_ONE_PLAYER, values, "id="
            ↪ + player.id, null);
    }
}

public void updateTeamScore(Team team) {
    ContentValues values = new ContentValues();
    values.put(DBOpenHelper.COLUMN_TEAM_ID, team.id
        ↪ );
    values.put(DBOpenHelper.COLUMN_TEAM_NAME, team.
        ↪ name);
    values.put(DBOpenHelper.COLUMN_TEAM_RUN, team.
        ↪ run);
    values.put(DBOpenHelper.COLUMN_TEAM_BALL, team.
        ↪ ball);
    values.put(DBOpenHelper.COLUMN_TEAM_WICKET,
        ↪ team.wicket);
    database.update(DBOpenHelper.TABLE_TEAM, values
        ↪ , "id=" + team.id, null);
}

public List<Player> getPlayerOneList() {
    List<Player> players = new ArrayList<>();

    Cursor cursor = database.query(DBOpenHelper.
        ↪ TABLE_TEAM_ONE_PLAYER,
            new String[]{DBOpenHelper.
                ↪ COLUMN_PLAYER_ID,
                    DBOpenHelper.COLUMN_PLAYER_NAME
                        ↪ ,
                    DBOpenHelper.COLUMN_PLAYER_RUN,
                    DBOpenHelper.COLUMN_PLAYER_BALL
                      ↪ ,
                    DBOpenHelper.COLUMN_PLAYER_FOUR
                      ↪ ,
                    DBOpenHelper.COLUMN_PLAYER_SIX,
                    DBOpenHelper.
                      ↪ COLUMN_PLAYER_BOWLER_RUN,
                    DBOpenHelper.
                      ↪ COLUMN_PLAYER_BOWLER_BALL
                      ↪ ,
                    DBOpenHelper.
                      ↪ COLUMN_PLAYER_BOWLER_WICKET
                      ↪ }, null, null,
            null, null, null);

    if (cursor.getCount() > 0) {
        while (cursor.moveToNext()) {
            Player player = new Player();
            player.id = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_ID));
            player.name = cursor.getString(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_NAME));
            player.run = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_RUN));
            player.ball = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_BALL));
            player.four = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_FOUR));
            player.six = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_SIX));
            player.bowlerRun = cursor.getInt(cursor
                    .getColumnIndex(DBOpenHelper.
                      ↪ COLUMN_PLAYER_BOWLER_RUN)
                      ↪ );
```

```
                player.bowlerBall = cursor.getInt(
                    ↪ cursor
                            .getColumnIndex(DBOpenHelper.
                                ↪ COLUMN_PLAYER_BOWLER_BALL
                                ↪ ));
                player.wickets = cursor.getInt(cursor
                            .getColumnIndex(DBOpenHelper.
                                ↪ COLUMN_PLAYER_BOWLER_WICKET
                                ↪ ));

                players.add(player);
            }
        }

        return players;
    }

    public List<Player> getPlayerTwoList() {
        List<Player> players = new ArrayList<>();

        Cursor cursor = database.query(DBOpenHelper.
            ↪ TABLE_TEAM_TWO_PLAYER,
                new String[]{DBOpenHelper.
                    ↪ COLUMN_PLAYER_ID,
                        DBOpenHelper.COLUMN_PLAYER_NAME
                            ↪ ,
                        DBOpenHelper.COLUMN_PLAYER_RUN,
                        DBOpenHelper.COLUMN_PLAYER_BALL
                            ↪ ,
                        DBOpenHelper.COLUMN_PLAYER_FOUR
                            ↪ ,
                        DBOpenHelper.COLUMN_PLAYER_SIX,
                        DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_RUN,
                        DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_BALL
                            ↪ ,
                        DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_WICKET
                            ↪ }, null, null,
                null, null, null);

        if (cursor.getCount() > 0) {
            while (cursor.moveToNext()) {
                Player player = new Player();
                player.id = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_ID));
                player.name = cursor.getString(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_NAME));
                player.run = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_RUN));
                player.ball = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_BALL));
                player.four = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_FOUR));
                player.six = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_SIX));
                player.bowlerRun = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_RUN)
                            ↪ );
                player.bowlerBall = cursor.getInt(
                    ↪ cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_BALL
                            ↪ ));
                player.wickets = cursor.getInt(cursor
                        .getColumnIndex(DBOpenHelper.
                            ↪ COLUMN_PLAYER_BOWLER_WICKET
                            ↪ ));

                players.add(player);
            }
        }
        return players;
    }

    public Team getTeamScore(int id) {
        Cursor cursor = database.query(DBOpenHelper.
            ↪ TABLE_TEAM, new String[]{DBOpenHelper.
            ↪ COLUMN_TEAM_ID,
                            DBOpenHelper.COLUMN_TEAM_NAME,
                                ↪ DBOpenHelper.
                                ↪ COLUMN_TEAM_RUN,
                                ↪ DBOpenHelper.
                                ↪ COLUMN_TEAM_BALL,
                                ↪ DBOpenHelper.
                                ↪ COLUMN_TEAM_WICKET},
                                ↪ DBOpenHelper.
                                ↪ COLUMN_TEAM_ID + "=?",
                new String[]{String.valueOf(id)}, null,
                    ↪  null, null, null);
        if (cursor != null) {
            cursor.moveToFirst();
        }

        Team team = new Team();
        team.id = cursor.getInt(0);
        team.name = cursor.getString(1);
        team.run = Integer.parseInt(cursor.getString(2)
            ↪ );
        team.ball = Integer.parseInt(cursor.getString
            ↪ (3));
        team.wicket = Integer.parseInt(cursor.getString
            ↪ (4));

        return team;
    }

    public void open() {
        database = dbhelper.getWritableDatabase();
    }
    public void close() {
        dbhelper.close();
    }


}
```

```
package com.example.out;

public class Player {
    public int id;
    public String name;
    public int run;
    public int ball;
    public int four;
    public int six;
    public int bowlerRun;
    public int bowlerBall;
    public int wickets;
    public double economyRate;
    public double strikeRate;

    public Player() {
        init("Player");
    }

    public Player(int playerNum) {
        init("Player␣" + playerNum);
        id = playerNum;
    }

    private void init(String playerName) {
        name = playerName;
        run = 0;
        ball = 0;
        four = 0;
        six = 0;
        bowlerBall = 0;
        bowlerRun = 0;
        wickets = 0;
        strikeRate = 0.0;
        economyRate = 0.0;
    }

    public void strikeRate() {
        strikeRate = run * 1.0 / ball * 100.0;
    }

    public double economyRate() {
        economyRate = bowlerRun * 1.0 / bowlerBall * 6;
        return economyRate;
    }

    public int over() {
        return bowlerBall / 6;
    }

    public int overBall() {
        return bowlerBall % 6;
    }

    @Override
    public String toString() {
        return name;
    }
}
```

```
package com.example.out;

import java.util.ArrayList;
import java.util.List;

public class Team {
    public long id;
    public String name;
    public List<Player> players;
    public int run;
    public boolean isFirstInnings;
    public boolean isWon;
    public int wicket;
    public int ball;

    public Team() {
        init("Untitled");
    }

    public Team(int teamNo) {
        init("Team " + teamNo);
        id = teamNo;
    }

    private void init(String teamName) {
        name = teamName;
        players = new ArrayList<>();
        run = 0;
        isFirstInnings = false;
        isWon = false;
        wicket = 0;
        ball = 0;
    }

    public double runRate() {
        return run * 1.0 / ball * 6;
    }

    public int over() {
        return ball / 6;
    }

    public int overBall() {
        return ball % 6;
    }

    @Override
    public String toString() {
        return "Team{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", run=" + run +
                ", wicket=" + wicket +
                ", ball=" + ball +
                '}';
    }
}
```

```
package com.example.out;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

public class SplashScreenActivity extends Activity {



    // Splash screen timer
    private static int SPLASH_TIME_OUT = 2000;

    @Override
    protected void onCreate(Bundle savedInstanceState)
        ↪ {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash_screen)
            ↪ ;

          new Handler().postDelayed(new Runnable() {

            @Override
            public void run() {

                Intent i = new Intent(
                    ↪ SplashScreenActivity.this,
                    ↪ MainActivity.class);
                startActivity(i);


                finish();
            }
        }, SPLASH_TIME_OUT);
    }
}
```

```
package com.example.out;

import android.app.Activity;
import android.os.Bundle;

public class TossActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState)
       ↪ {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_toss);
    }
}
```