APK FILE :
https://drive.google.com/file/d/1Sjy02VILP-kRb6a3kUm7oDV4y1zPbrKD/view?usp=sharing

Mobile Application Development Laboratory
106118083 - Rvs Satyanand

1.

Experiment Name: Basic Tic-Tac-Toe Android Application

Date: 15-02-2021

Aim: To design an android application using Android Studio for Tic-Tac-Toe game with the following specifications.

i) 9 buttons - toggle the player mode (display the player turn information using Toasts)
ii) Have two symbols for each player and change the button text to the symbol once pressed.
iii) Display the winner information and loser information using toasts.

Description of App: The app simulates a simple tic-tac-toe which is a multiple player game. The logic of the app is Once the Appln starts Player X will start and when player X has made his/her move then the player O will get a chance to perform his/her move. The app uses onClick listeners for the buttons from which the input is taken and based on the button it runs different instructions For example if the person clicks a button which was already filled up with “X” or “O” then it’ll show a Error in other cases it’ll check if there are any diagonal, horizontal, or vertical case getting fulfilled if any then the player Wins the Game. The result is shown in the textview and Toast.

Device Specifications: The app runs on min SDK version of 16 (so anything above API 16 - Android 4.1 - Jelly Bean would run this app which is 99.8% of devices). I have used my own phone for simulation (Samsung J8). The laptop has a 16GB ram.

Name: Pixel_3
Resolution: 1080 X 2220
API: 30
Target: Android 11.0
hw.lcd.height: 2220
hw.accelerometer: yes

hw.device.manufacturer: Google
hw.lcd.width: 1080
hw.lcd.density: 440
hw.cpu.ncore: 6
hw.sensors.proximity: yes
hw.sensors.orientation: yes
hw.gpu.enabled: yes

Technical Concepts Learnt:

1. Constraint Layout - Adding Constraints and GuideLines.
2. Activity Lifecycles
3. Instantiating views by findViewById
4. Click Listeners
5. Preventing crashes by catching exceptions
6. Listeners.
7. Bundle, View, Button, Grid, TableRow, RelativeLayout, TableLayout
8. **Separate logic to handle all the cases(vertical,diagonal,horizontal) for a player's win**
9. **Ui styling**

Source Code:

activity_main.xml

```
<?xml version="1.0"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingBottom="@dimen/activity_vertical_margin"
   tools:context=".MainActivity">


   <TextView
       android:id="@+id/turn"
```

```xml
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="sans-serif-medium"
        android:gravity="left"
        android:padding="10dp"
        android:text="          Turn:  "
        android:textColor="@color/black"
        android:textSize="24dp" />

    <TableLayout
        android:id="@+id/mainBoard"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="top|center"
        android:layout_marginTop="55dp"
        android:clickable="true"
        android:gravity="center"
        android:nestedScrollingEnabled="false"
        android:padding="10dp">

        <TableRow
            android:id="@+id/row0"
            style="@style/TableRow"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <Button
                style="@style/LeftCell"
                android:layout_column="0"
                android:width="50dp"></Button>

            <Button
                style="@style/MiddleCell"
                android:layout_column="1"
                android:width="50dp"></Button>

            <Button
                style="@style/RightCell"
                android:layout_column="2"
                android:width="50dp"></Button>

        </TableRow>

        <TableRow
            android:id="@+id/row1"
            style="@style/TableRow"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <Button
                style="@style/LeftCell"
                android:layout_column="0"
                android:width="50dp"></Button>

            <Button
                style="@style/MiddleCell"
                android:layout_column="1"
                android:width="50dp"></Button>

            <Button

                style="@style/RightCell"
                android:layout_column="2"
                android:width="50dp"></Button>

        </TableRow>

        <TableRow
            android:id="@+id/row2"
            style="@style/TableRow"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <Button
                style="@style/LeftCell"
                android:layout_column="0"
                android:width="50dp"></Button>

            <Button
```

```xml
                style="@style/MiddleCell"
                android:layout_column="1"
                android:width="50dp"></Button>

            <Button
                style="@style/RightCell"
                android:layout_column="2"
                android:width="50dp"></Button>

        </TableRow>

    </TableLayout>


    <TextView
        android:id="@+id/error"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:fontFamily="sans-serif-medium"
        android:gravity="left"
        android:padding="10dp"
        android:text=""
        android:textColor="@android:color/holo_red_light"
        android:textSize="24dp" />

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center">

        <Button
            android:id="@+id/reset"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
            android:layout_marginTop="10dp"
            android:text="Restart" />
    </RelativeLayout>

</LinearLayout>
```

ActivityMain.java

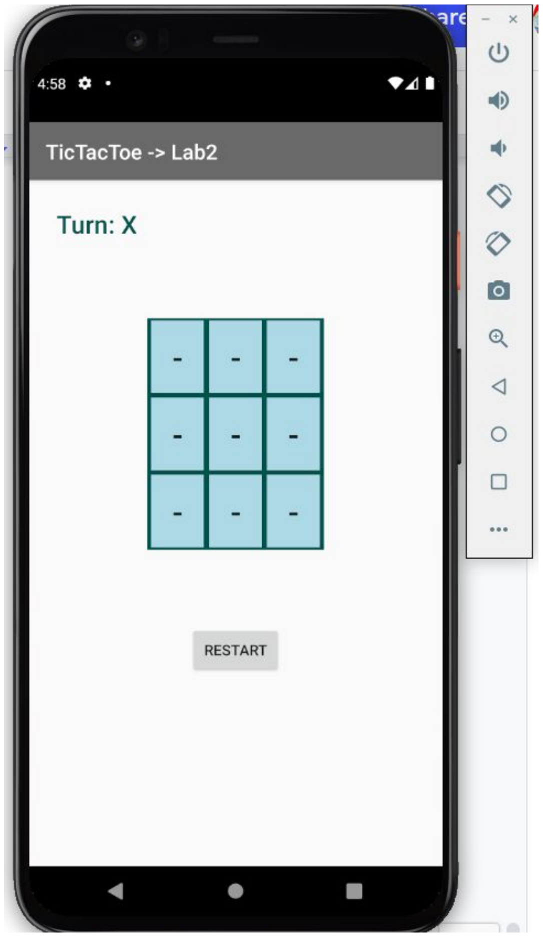
```
package com.dataflair.ticgame;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {


   private int grid_size;
   TableLayout gameBoard;
   TextView txt_turn, error_msg;
   char [][] my_board;
   char turn;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);

       grid_size = Integer.parseInt(getString(R.string.size_of_board));
       my_board = new char [grid_size][grid_size];
       gameBoard = (TableLayout) findViewById(R.id.mainBoard);
       txt_turn = (TextView) findViewById(R.id.turn);
       error_msg = (TextView) findViewById(R.id.error);
```

```java
        resetBoard();
        txt_turn.setText("Turn: "+turn);

        for(int i = 0; i< gameBoard.getChildCount(); i++){
            TableRow row = (TableRow) gameBoard.getChildAt(i);
            for(int j = 0; j<row.getChildCount(); j++){
                Button tv = (Button) row.getChildAt(j);
                tv.setText("-");
                tv.setOnClickListener(Move(i, j, tv));
            }
        }

        Button reset_btn = (Button) findViewById(R.id.reset);
        reset_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent current = getIntent();
                finish();
                startActivity(current);
            }
        });


        Toast.makeText(this, "Player X: Turn", Toast.LENGTH_SHORT).show();
    }

    protected void resetBoard(){
        turn = 'X';
        for(int i = 0; i< grid_size; i++){
            for(int j = 0; j< grid_size; j++){
                my_board[i][j] = ' ';
            }
        }
    }

    protected int gameStatus(){

        //0 Continue
        //1 X Wins
        //2 O Wins
    //-1 Draw

    int rowX = 0, colX = 0, rowO = 0, colO = 0;
    for(int i = 0; i< grid_size; i++){
        if(check_Row_Equality(i,'X'))
            return 1;
        if(check_Column_Equality(i, 'X'))
            return 1;
        if(check_Row_Equality(i,'O'))
            return 2;
        if(check_Column_Equality(i,'O'))
            return 2;
        if(check_Diagonal('X'))
            return 1;
        if(check_Diagonal('O'))
            return 2;
    }

    boolean boardFull = true;
    for(int i = 0; i< grid_size; i++){
        for(int j = 0; j< grid_size; j++){
            if(my_board[i][j]==' ')
                boardFull = false;
        }
    }
    if(boardFull)
        return -1;
    else return 0;
}

protected boolean check_Diagonal(char player){
    int count_Equal1 = 0,count_Equal2 = 0;
    for(int i = 0; i< grid_size; i++)
        if(my_board[i][i]==player)
            count_Equal1++;
    for(int i = 0; i< grid_size; i++)
        if(my_board[i][grid_size -1-i]==player)
            count_Equal2++;
    if(count_Equal1== grid_size || count_Equal2== grid_size)
        return true;
```

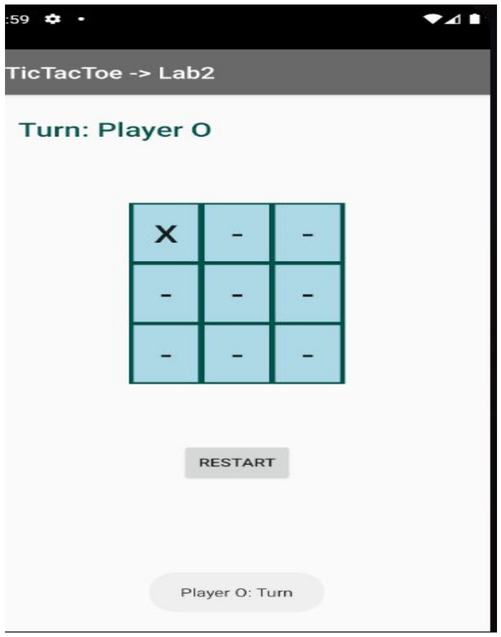
```java
        else return false;
    }

    protected boolean check_Row_Equality(int r, char player){
        int count_Equal=0;
        for(int i = 0; i< grid_size; i++){
            if(my_board[r][i]==player)
                count_Equal++;
        }

        if(count_Equal== grid_size)
            return true;
        else
            return false;
    }

    protected boolean check_Column_Equality(int c, char player){
        int count_Equal=0;
        for(int i = 0; i< grid_size; i++){
            if(my_board[i][c]==player)
                count_Equal++;
        }

        if(count_Equal== grid_size)
            return true;
        else
            return false;
    }

    protected boolean Cell_Set(int r, int c){
        return !(my_board[r][c]==' ');
    }

    protected void stopMatch(){
        for(int i = 0; i< gameBoard.getChildCount(); i++){
            TableRow row = (TableRow) gameBoard.getChildAt(i);
            for(int j = 0; j<row.getChildCount(); j++){
                TextView tv = (TextView) row.getChildAt(j);
                tv.setOnClickListener(null);
            }
        }
    }

    private void ToastFlash(String msg){
        Toast.makeText(this, msg, -1).show();
    }

    View.OnClickListener Move(final int r, final int c, final Button tv){

        return new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                error_msg.setText("");

                if(!Cell_Set(r,c)) {
                    my_board[r][c] = turn;
                    if (turn == 'X') {
                        tv.setText(R.string.X);
                        turn = 'O';
                        ToastFlash("Player O: Turn");
                    } else if (turn == 'O') {
                        tv.setText(R.string.O);
                        turn = 'X';
                        ToastFlash("Player X: Turn");
                    }
                    if (gameStatus() == 0) {
                        txt_turn.setText("Turn: Player " + turn);
                    }
                    else if(gameStatus() == -1){
                        txt_turn.setText("This is a Draw match");
                        ToastFlash("This is a Draw match");
                        stopMatch();
                    }
                    else{
                        txt_turn.setText("Player " + turn + " Loses! \n" +
"Player " + (turn == 'X' ? "O" : "X") + " Wins!");
                        ToastFlash("Player " + (turn == 'X' ? "O" : "X") +
" Wins!");
                        stopMatch();
                    }
                }
                else{
                    error_msg.setText(" Select an Empty Cell");
                    ToastFlash("Player " + (turn == 'X' ? "O" : "X") + "
Wins!");

                }
            }
        };
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_board, menu);
        return true;
    }

}
```

Screenshots:

Appln Start:



After First Move:

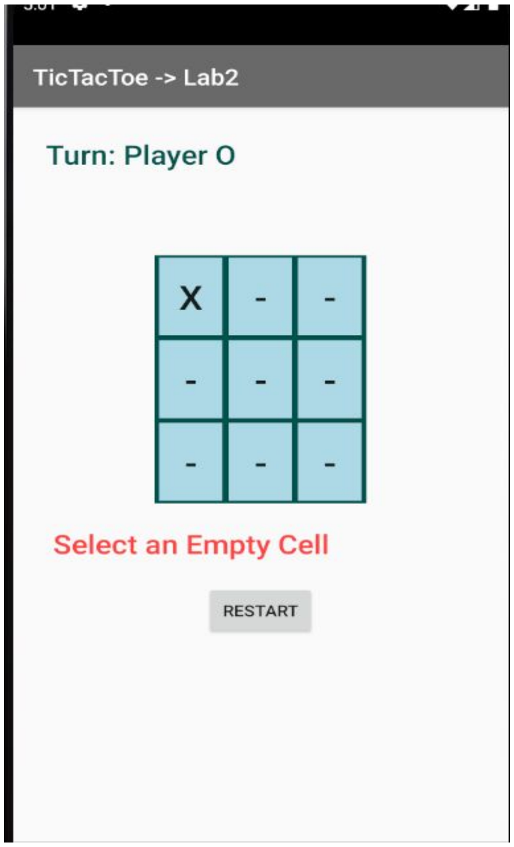

TicTacToe -> Lab2

Turn: Player O

X	-	-
-	-	-
-	-	-

Player O: Turn

Incase a player clicks the same cell:

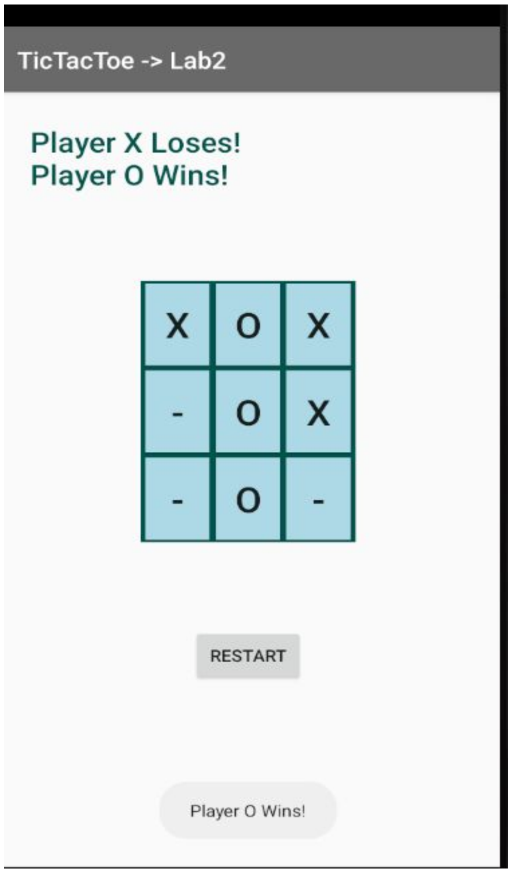

TicTacToe -> Lab2

Turn: Player O

X	-	-
-	-	-
-	-	-

Select an Empty Cell

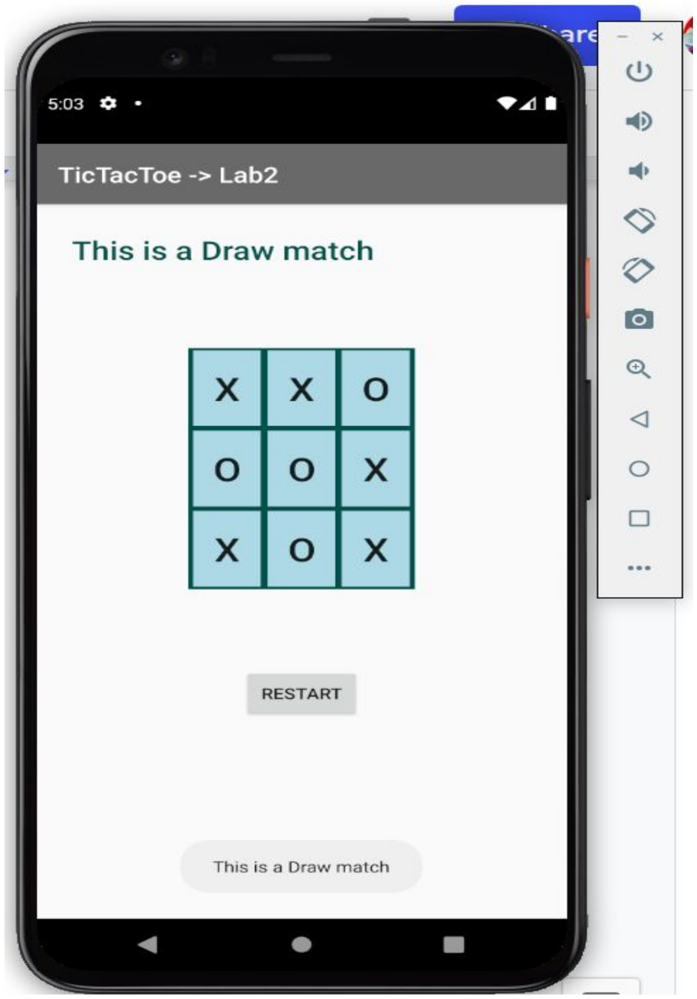
Case When Player X loses And Player O Wins!!!


TicTacToe -> Lab2

Player X Loses!
Player O Wins!

Player O Wins!

When there is a draw:



This is a Draw match

X	X	O
O	O	X
X	O	X

This is a Draw match

Outcomes:
The tasks given were accomplished without any bugs/crashes.