**EXECUTIVE SUMMARY**

**Tic-Tac-Toe Application**

* A simple application of a game of tic-tac-toe. This is made for two players alternating each turn. The score will add up on each player and once players decided to end the games, the score will be saved in the database using SQLite.

**GROUP INTRODUCTION**

**Group 7: Jeffrey Hermo (Only Member) – worked on everything**

**DETAILED SCREENSHOTS**

A picture containing application

Description automatically generated

1. Start Game – On click will be asked for players name and then game will start.
2. View Scores – On click will show the records of all the games played.

Graphical user interface, application, Teams

Description automatically generatedGraphical user interface, application, Teams

Description automatically generated

* Player name entry for both players.
* Need to enter first the player one and then upon clicking create, a dialog for player two will appear.
* If user click cancel, screen will go back to home screen.
* Validation present, if user inputted blank and clicked create, dialog box will remain until user entered a name or clicked cancel. Same with player two entry.

Diagram

Description automatically generated

1. Grid for the tic-tac-toe game. It’s a 3 x 3 grid for the game.
2. Players name with score.
3. Turn of the player.

Timeline

Description automatically generated

1. X icon – X icon is for player two.
2. Heart icon – heart icon is for player one.
3. Play Again button – game will restart but the score will be added.
4. Player who won will show.
5. Home Button – Goes back to home page and the game information will be saved.
6. Shows the winning combination with a red line.

A picture containing text

Description automatically generated

* Upon clicking the home button, it will go back to the home page and the game information will be saved.

Graphical user interface, application

Description automatically generated

1. Column titles
2. Data of the game
   1. Id – Auto-incrementing id for the record.
   2. Player 1 Name – Name of player one.
   3. Player 1 Score – Score of the player one on the game.
   4. Player 2 Name – Name of player two.
   5. Player 2 Score – Score of player two on the game.
   6. Date – Date of the game
3. Go Back Button – Goes back to the home page.

**MILESTONES AND TIMELINE**

A picture containing graphical user interface

Description automatically generated

Github commits - <https://github.com/jhermo1229/TicTacToe_FinalProject>

**APRIL 13**

A screenshot of a computer

Description automatically generated with medium confidence

Created home page

Graphical user interface, application

Description automatically generatedA close-up of a cell phone

Description automatically generated with low confidence

Created dialog box for player name creation

Table

Description automatically generated with low confidence

Created grid for the game

APRIL 15

A screenshot of a cell phone

Description automatically generated with medium confidence

Added icons for the game. Changed the background to black.

**APRIL 16**

A screenshot of a computer

Description automatically generated with medium confidenceDiagram

Description automatically generated with medium confidence

Added the red line for combination of win. Added also score counting.

**APRIL 17**

Graphical user interface, application, Teams

Description automatically generated

Added validation for both input name

A screenshot of a computer

Description automatically generated with medium confidence

Added Turn text

A screenshot of a phone

Description automatically generated with medium confidence

Added winner layout and button for playing again or going back to home page

A screenshot of a computer

Description automatically generated with low confidence

Added logic for playing again or going back to home

Graphical user interface, application

Description automatically generated

Added viewing of games played

A picture containing Teams

Description automatically generated

Added saving of game when home button is clicked

**APRIL 19**

Graphical user interface

Description automatically generated with medium confidence

Added back button

A screenshot of a phone

Description automatically generated with low confidence

Added Draw (No winner)

**COMMENTED CODES**

**MainActivity.java**

package com.example.finalproject;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.fragment.app.FragmentTransaction;  
  
import android.os.Bundle;  
import android.view.Menu;  
import android.view.MenuItem;  
import android.widget.FrameLayout;  
  
*/\*\*  
 \* Main activity of the final project  
 \*/*public class MainActivity extends AppCompatActivity {  
  
 private FrameLayout homeFrame;  
 public static int *playerOneScore* = 0, *playerTwoScore* = 0;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
  
 //Created a fragment which will be used as home page  
 homeFrame = findViewById(R.id.*homeFrame*);  
 FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();  
 transaction.replace(R.id.*homeFrame*, new HomeFragment());  
 transaction.commit();  
  
 }  
  
}

**HomeFragment.java**

package com.example.finalproject;  
  
import android.app.AlertDialog;  
import android.content.DialogInterface;  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentTransaction;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
  
*/\*\*  
 \* HomeFragment - This is the main page of the project  
 \*/*public class HomeFragment extends Fragment {  
  
  
 public static final String *PLAYER\_NAME* = "Player Name";  
 public static final String *CREATE* = "Create";  
 public static final String *ONE* = "one";  
 public static final String *TWO* = "two";  
 public static final String *CANCEL* = "Cancel";  
 public static final String *SET\_NAME\_FOR\_PLAYER* = "Set name for player ";  
 private Button startBtn, viewScoreBtn;  
 private View gameView;  
 public String playerOneName;  
 public String playerTwoName;  
 private PlayerListAdapter mAdapter;  
  
 //HomeFragment constructor  
 public HomeFragment() {  
 }  
  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 gameView = inflater.inflate(R.layout.*fragment\_home*, container, false);  
  
 startBtn = gameView.findViewById(R.id.*startBtn*);  
  
 startBtn.setOnClickListener((gradeEntryView) -> {  
 enterNameDialogBox(*ONE*);  
 });  
  
 viewScoreBtn = gameView.findViewById(R.id.*scoreBtn*);  
  
 viewScoreBtn.setOnClickListener((gradeEntryView) -> {  
  
 FragmentTransaction transaction = getActivity().getSupportFragmentManager().beginTransaction();  
 transaction.addToBackStack(ViewScoreFragment.*TAG*);  
 transaction.replace(R.id.*homeFrame*, new ViewScoreFragment());  
 transaction.commit();  
 });  
  
  
 return gameView;  
 }  
  
 //Alert Dialog box for getting the player name  
 private void enterNameDialogBox(final String player) {  
 AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());  
 final EditText text = new EditText(getActivity());  
 builder.setTitle(*PLAYER\_NAME*).setMessage(*SET\_NAME\_FOR\_PLAYER* + player).setView(text);  
 builder.setPositiveButton(*CREATE*, new DialogInterface.OnClickListener() {  
  
 public void onClick(DialogInterface di, int i) {  
  
 //If player clicks the create button, function will call a recursive method to itself to enter  
 //the same details with player two.  
 if (player.equals(*ONE*)) {  
 playerOneName = text.getText().toString();  
 if (!playerOneName.isEmpty() && !playerOneName.startsWith(" ")) {  
 enterNameDialogBox(*TWO*);  
 } else {  
 enterNameDialogBox(*ONE*);  
 }  
 } else {  
 playerTwoName = text.getText().toString();  
 if (!playerTwoName.isEmpty() && !playerTwoName.startsWith(" ")) {  
  
 //After player two, the game fragment will be called.  
 GameFragment gameFrag = new GameFragment();  
 Bundle args = new Bundle();  
 args.putString("playerOneName", playerOneName);  
 args.putString("playerTwoName", playerTwoName);  
 gameFrag.setArguments(args);  
 FragmentTransaction transaction = getActivity().getSupportFragmentManager().beginTransaction();  
 transaction.addToBackStack(GameFragment.*TAG*);  
 transaction.replace(R.id.*homeFrame*, gameFrag);  
  
 transaction.commit();  
 } else {  
 enterNameDialogBox(*TWO*);  
 }  
 }  
 }  
 });  
 builder.setNegativeButton(*CANCEL*, new DialogInterface.OnClickListener() {  
  
 public void onClick(DialogInterface di, int i) {  
 }  
 });  
 builder.create().show();  
 }  
}

**GameFragment.java**

package com.example.finalproject;  
  
import android.graphics.Bitmap;  
import android.os.Bundle;  
  
import androidx.annotation.NonNull;  
import androidx.fragment.app.Fragment;  
import androidx.recyclerview.widget.GridLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.ImageView;  
import android.widget.RelativeLayout;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
  
*/\*\*  
 \* GameFragment - main fragment for the tic-tac-toe game  
 \*/*public class GameFragment extends Fragment {  
  
 public static final String *PLAYER\_ONE\_NAME* = "playerOneName";  
 public static final String *PLAYER\_TWO\_NAME* = "playerTwoName";  
 private View gameView;  
 public static String *TAG* = GameFragment.class.getName();  
 public static final String *RECORD\_ADDED\_SUCCESSFULLY* = "Record Added Successfully";  
 public static final String *UNSUCCESSFUL\_IN\_RECORD\_ADDING\_PLEASE\_CHECK\_LOGS* = "Unsuccessful in record adding. Please check logs";  
 private GameAdapter gameAdapter;  
 private RecyclerView gameRV;  
 public static boolean *playerOneTurn* = true;  
 public static ImageView *img\_stroke*, *img\_win*;  
 public static String *playerOneName*, *playerTwoName*;  
 public static RelativeLayout *win\_relative\_layout*;  
  
 public static TextView *playerOneTextView*, *playerTwoTextView*, *txt\_win*, *player\_turn*;  
 private Button btn\_reset, btn\_again, btn\_home;  
  
 private static final SimpleDateFormat *sdf1* = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss");  
  
 private DatabaseHelper dbh;  
 private Boolean insertStat;  
  
 public GameFragment() {  
 // Required empty public constructor  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 gameView = inflater.inflate(R.layout.*fragment\_game*, container, false);  
 //Initializing the active database  
 dbh = new DatabaseHelper(getActivity());  
 *player\_turn* = gameView.findViewById(R.id.*player\_turn*);  
 *playerOneTextView* = gameView.findViewById(R.id.*playerOne\_win*);  
 *playerTwoTextView* = gameView.findViewById(R.id.*playerTwo\_win*);  
 *playerOneName* = getArguments().getString(*PLAYER\_ONE\_NAME*);  
 *playerTwoName* = getArguments().getString(*PLAYER\_TWO\_NAME*);  
 *playerOneTextView*.setText(*playerOneName* + ": 0");  
 *playerTwoTextView*.setText(*playerTwoName* + ": 0");  
 *player\_turn*.setText("Turn of " + *playerOneName*);  
 *win\_relative\_layout* = gameView.findViewById(R.id.*win\_relative\_layout*);  
  
 *txt\_win* = gameView.findViewById(R.id.*txt\_win*);  
  
 //Recycler view for the game board  
 gameRV = gameView.findViewById(R.id.*tictactoeboard*);  
 *img\_stroke* = gameView.findViewById(R.id.*img\_stroke*);  
 btn\_again = gameView.findViewById(R.id.*btn\_again*);  
 btn\_home = gameView.findViewById(R.id.*btn\_home*);  
  
  
 //Number of created boxes for the board - total of 9 boxes  
 ArrayList<Bitmap> arrayBoxes = getBitmaps();  
  
 //Recycler adapter passing the grid and context as parameter  
 gameAdapter = new GameAdapter(getContext(), arrayBoxes);  
  
 //Number of grid columns for the Grid layout  
 RecyclerView.LayoutManager layoutManager = new GridLayoutManager(getContext(), 3);  
 gameRV.setLayoutManager(layoutManager);  
 gameRV.setAdapter(gameAdapter);  
  
 btn\_again.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 *win\_relative\_layout*.setVisibility(View.*INVISIBLE*);  
 reset();  
 }  
 });  
 btn\_home.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 reset();  
 Player player = new Player();  
 player.setPlayerOneName(*playerOneName*);  
 player.setPlayerTwoName(*playerTwoName*);  
 player.setPlayerOneWin(MainActivity.*playerOneScore*);  
 player.setPlayerTwoWin(MainActivity.*playerTwoScore*);  
 player.setCurrentTimeStamp(String.*valueOf*(*sdf1*.format(System.*currentTimeMillis*())));  
  
 //passing the object to the database handler  
 insertStat = dbh.InsertPlayer(player);  
  
 //if values was added successfully will toast a success message. If not, will toast a unsuccessful message  
 if (insertStat) {  
 Toast.*makeText*(getActivity(), *RECORD\_ADDED\_SUCCESSFULLY*, Toast.*LENGTH\_SHORT*).show();  
 } else {  
 Toast.*makeText*(getActivity(), *UNSUCCESSFUL\_IN\_RECORD\_ADDING\_PLEASE\_CHECK\_LOGS*, Toast.*LENGTH\_SHORT*).show();  
 }  
  
 MainActivity.*playerOneScore* = 0;  
 MainActivity.*playerTwoScore* = 0;  
 getParentFragmentManager().popBackStack();  
 }  
  
  
 });  
  
 return gameView;  
 }  
  
  
 private void reset() {  
 ArrayList<Bitmap> arrayBox = getBitmaps();  
 *img\_stroke*.setImageBitmap(null);  
 gameAdapter.setArrayBoxes(arrayBox);  
 gameAdapter.notifyDataSetChanged();  
 *playerOneTurn* = true;  
 *player\_turn*.setText("Turn of " + *playerOneName*);  
 }  
  
  
 //Resetting the boxes  
 @NonNull  
 private ArrayList<Bitmap> getBitmaps() {  
 ArrayList<Bitmap> arrayBoxes = new ArrayList<>();  
 for (int i = 0; i < 9; i++) {  
 arrayBoxes.add(null);  
 }  
 return arrayBoxes;  
 }  
}

**GameAdapter.java**

package com.example.finalproject;  
  
import android.annotation.SuppressLint;  
import android.content.Context;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.ArrayList;  
  
*/\*\*  
 \* GameAdapter - Adapter for the recycler view of the game  
 \*/*public class GameAdapter extends RecyclerView.Adapter<GameAdapter.ViewHolder> {  
  
 public static final String *WINS* = "wins!!!";  
 public static final String *BLANK* = " ";  
 public static final String *DRAW* = "DRAW!!!!";  
 public static final String *TURN\_OF* = "Turn of";  
 private ArrayList<Bitmap> arrayBoxes, arrayWin;  
 private Context context;  
 private Bitmap heartBitmap, xBitmap;  
 private String winningPlayer, playerOneName, playerTwoName;  
  
 //Constructor with parameter for context and the number of boxes in the grid  
 public GameAdapter(Context context, ArrayList<Bitmap> arrayBoxes) {  
 this.context = context;  
 this.arrayBoxes = arrayBoxes;  
 heartBitmap = BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*blue\_heart*);  
 xBitmap = BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*red\_x*);  
 arrayWin = new ArrayList<>();  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win1*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win2*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win3*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win4*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win5*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win6*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win7*));  
 arrayWin.add(BitmapFactory.*decodeResource*(context.getResources(), R.drawable.*win8*));  
 playerOneName = GameFragment.*playerOneName*;  
 playerTwoName = GameFragment.*playerTwoName*;  
  
 }  
  
  
 //Created view holder for the game adapter  
 class ViewHolder extends RecyclerView.ViewHolder {  
  
 private ImageView box\_table;  
  
 public ViewHolder(@NonNull View v) {  
 super(v);  
 box\_table = itemView.findViewById(R.id.*box\_table*);  
 }  
 }  
  
 //OnCreate, inflate the box table layout  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 return new ViewHolder(LayoutInflater.*from*(context).inflate(R.layout.*box\_table*, parent, false));  
 }  
  
 //Bind each boxes with the array passed  
 @Override  
 public void onBindViewHolder(@NonNull final ViewHolder holder, @SuppressLint("RecyclerView") int position) {  
 holder.box\_table.setImageBitmap(arrayBoxes.get(position));  
  
 holder.box\_table.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View view) {  
 //Check first if box has a value already  
 if (arrayBoxes.get(position) == null) {  
  
 //player one is set to true thus will always start first  
 //If user click a box, it will be set and then added a bitmap depending on who is the player  
 if (GameFragment.*playerOneTurn*) {  
 arrayBoxes.set(position, heartBitmap);  
 GameFragment.*playerOneTurn* = false;  
 GameFragment.*player\_turn*.setText(*TURN\_OF* + *BLANK* + playerTwoName);  
 } else {  
 arrayBoxes.set(position, xBitmap);  
 GameFragment.*playerOneTurn* = true;  
 GameFragment.*player\_turn*.setText(*TURN\_OF* + *BLANK* + playerOneName);  
 }  
  
 //Win checker on click. If win will display a layout who is the winner  
 //Else it will be a draw  
 if (checkWin()) {  
 win();  
 } else {  
 int count = 0;  
 for (int i = 0; i < arrayBoxes.size(); i++) {  
 if (arrayBoxes.get(i) != null) {  
 count++;  
 }  
 }  
 if (count == 9) {  
 GameFragment.*win\_relative\_layout*.setVisibility(View.*VISIBLE*);  
 GameFragment.*txt\_win*.setText(*DRAW*);  
 }  
 }  
 notifyItemChanged(position);  
 }  
 }  
 });  
 }  
  
 private boolean checkWin() {  
  
 //Check if horizontal rows of table has a winning player. Need to check a box if null to prevent winning from the start of the game (all boxes are null)  
 if ((arrayBoxes.get(0) == arrayBoxes.get(1)) && (arrayBoxes.get(1) == arrayBoxes.get(2)) && arrayBoxes.get(0) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(5));  
 checkWinningPlayer(0);  
 return true;  
 } else if ((arrayBoxes.get(3) == arrayBoxes.get(4)) && (arrayBoxes.get(4) == arrayBoxes.get(5)) && arrayBoxes.get(3) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(6));  
 checkWinningPlayer(5);  
 return true;  
 } else if ((arrayBoxes.get(6) == arrayBoxes.get(7)) && (arrayBoxes.get(7) == arrayBoxes.get(8)) && arrayBoxes.get(6) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(7));  
 checkWinningPlayer(6);  
 return true;  
  
 //Check if vertical columns of table has a winning player  
 } else if (arrayBoxes.get(0) == arrayBoxes.get(3) && arrayBoxes.get(3) == arrayBoxes.get(6) && arrayBoxes.get(0) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(2));  
 checkWinningPlayer(0);  
 return true;  
 } else if ((arrayBoxes.get(1) == arrayBoxes.get(4)) && (arrayBoxes.get(4) == arrayBoxes.get(7)) && arrayBoxes.get(1) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(3));  
 checkWinningPlayer(1);  
 return true;  
 } else if ((arrayBoxes.get(2) == arrayBoxes.get(5)) && (arrayBoxes.get(5) == arrayBoxes.get(8)) && arrayBoxes.get(2) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(4));  
 checkWinningPlayer(2);  
 return true;  
  
 //Check if slant boxes has a winning player  
 } else if (arrayBoxes.get(0) == arrayBoxes.get(4) && arrayBoxes.get(4) == arrayBoxes.get(8) && arrayBoxes.get(0) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(1));  
 checkWinningPlayer(0);  
 return true;  
 } else if (arrayBoxes.get(2) == arrayBoxes.get(4) && arrayBoxes.get(4) == arrayBoxes.get(6) && arrayBoxes.get(2) != null) {  
 GameFragment.*img\_stroke*.setImageBitmap(arrayWin.get(0));  
 checkWinningPlayer(2);  
 return true;  
 }  
 return false;  
 }  
  
 //Gets the array box as a parameter (any of the 3 combination box can be sent) and then checks  
 //what bitmap is inserted in the array  
 private void checkWinningPlayer(int i) {  
 if (arrayBoxes.get(i) == heartBitmap) {  
 winningPlayer = GameFragment.*playerOneName*;  
 } else {  
 winningPlayer = GameFragment.*playerTwoName*;  
 }  
 }  
  
 //Main activity contains the main score and it loops depending on the winner.  
 //Also contains the setting of the textview in the fragment for the respective score  
 private void win() {  
 GameFragment.*win\_relative\_layout*.setVisibility(View.*VISIBLE*);  
 if (winningPlayer.equals(playerOneName)) {  
 MainActivity.*playerOneScore*++;  
 GameFragment.*playerOneTextView*.setText(playerOneName + ": " + MainActivity.*playerOneScore*);  
 GameFragment.*txt\_win*.setText(playerOneName + *BLANK* + *WINS*);  
  
 } else {  
 MainActivity.*playerTwoScore*++;  
 GameFragment.*playerTwoTextView*.setText(playerTwoName + ": " + MainActivity.*playerTwoScore*);  
 GameFragment.*txt\_win*.setText(playerTwoName + *BLANK* + *WINS*);  
  
 }  
  
  
 }  
  
 @Override  
 public int getItemCount() {  
 return arrayBoxes.size();  
 }  
  
  
 public ArrayList<Bitmap> getArrayBoxes() {  
 return arrayBoxes;  
 }  
  
 public void setArrayBoxes(ArrayList<Bitmap> arrayBoxes) {  
 this.arrayBoxes = arrayBoxes;  
 }  
}

**DatabaseHelper.java**

package com.example.finalproject;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
import androidx.annotation.Nullable;  
  
public class DatabaseHelper extends SQLiteOpenHelper {  
  
 public static final String *dbName* = "Tictactoe.db";  
 public static final int *version* = 2;  
 public static final String *TABLE\_NAME* = "Games";  
 public static final String *ID* = "id";  
 public static final String *COL\_PLAYER\_ONE* = "player\_one";  
 public static final String *COL\_PLAYER\_TWO* = "player\_two";  
 public static final String *COL\_PLAYER\_ONE\_SCORE* = "player\_one\_score";  
 public static final String *COL\_PLAYER\_TWO\_SCORE*= "player\_two\_score";  
 public static final String *COL\_TIMESTAMP*= "game\_timestamp";  
  
 //Create table constant. ID is primary key and auto-incremented  
 public static final String *CREATE\_TABLE* = "create table " + *TABLE\_NAME* + "(" + *ID* + " INTEGER PRIMARY KEY AUTOINCREMENT, "  
 + *COL\_PLAYER\_ONE* + " TEXT NOT NULL, " + *COL\_PLAYER\_TWO* + " TEXT, " + *COL\_PLAYER\_ONE\_SCORE* + " INTEGER, " + *COL\_PLAYER\_TWO\_SCORE* + " INTEGER, "  
 + *COL\_TIMESTAMP* + " TEXT NOT NULL);";  
 public static final String *DROP\_TABLE* = "DROP TABLE IF EXISTS " + *TABLE\_NAME*;  
  
  
 //Constructor for initializing database handler  
 public DatabaseHelper(@Nullable Context context) {  
 super(context, *dbName*, null, *version*);  
 }  
  
  
 @Override  
 public void onCreate(SQLiteDatabase sqLiteDatabase) {  
 sqLiteDatabase.execSQL(*CREATE\_TABLE*);  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
 sqLiteDatabase.execSQL(*DROP\_TABLE*); //Drop if table exists  
 onCreate(sqLiteDatabase);  
 }  
  
 public boolean InsertPlayer(Player player) {  
 SQLiteDatabase db = this.getWritableDatabase(); // instance of SQL Lite db  
 ContentValues contentValues = new ContentValues();  
  
 //COL1 is ID and it is auto-incremented  
 contentValues.put(*COL\_PLAYER\_ONE*, player.getPlayerOneName());  
 contentValues.put(*COL\_PLAYER\_TWO*, player.getPlayerTwoName());  
 contentValues.put(*COL\_PLAYER\_ONE\_SCORE*, player.getPlayerOneWin());  
 contentValues.put(*COL\_PLAYER\_TWO\_SCORE*, player.getPlayerTwoWin());  
 contentValues.put(*COL\_TIMESTAMP*, player.getCurrentTimeStamp());  
  
 long result = db.insert(*TABLE\_NAME*, null, contentValues);  
  
 if (result == -1) {  
 return false;  
 }  
 return true;  
  
 }  
  
 //Database call for selecting data in table  
 public Cursor viewData() {  
 SQLiteDatabase db = this.getWritableDatabase();  
  
 Cursor cursor = null;  
  
 cursor = db.rawQuery("Select \* from " + *TABLE\_NAME*, null);  
  
 //if data gathered is not null, then will move to first record.  
 if (cursor != null) {  
 cursor.moveToFirst();  
 }  
  
 return cursor;  
 }  
}

**Player.java**

package com.example.finalproject;  
  
*/\*\*  
 \* Player model class - also to be used for database  
 \*/*public class Player {  
  
 private Integer id;  
 private String playerOneName;  
 private String playerTwoName;  
 private int playerOneWin;  
 private int playerTwoWin;  
 private String currentTimeStamp;  
  
 public Player(Integer id, String playerOneName, String playerTwoName, int playerOneWin, int playerTwoWin,  
 String currentTimeStamp) {  
 this.setId(id);  
 this.setPlayerOneName(playerOneName);  
 this.setPlayerTwoName(playerTwoName);  
 this.setPlayerOneWin(playerOneWin);  
 this.setPlayerTwoWin(playerTwoWin);  
 }  
  
 public Player() {  
  
 }  
  
  
 public String getPlayerOneName() {  
 return playerOneName;  
 }  
  
 public void setPlayerOneName(String playerOneName) {  
 this.playerOneName = playerOneName;  
 }  
  
 public String getPlayerTwoName() {  
 return playerTwoName;  
 }  
  
 public void setPlayerTwoName(String playerTwoName) {  
 this.playerTwoName = playerTwoName;  
 }  
  
 public int getPlayerOneWin() {  
 return playerOneWin;  
 }  
  
 public void setPlayerOneWin(int playerOneWin) {  
 this.playerOneWin = playerOneWin;  
 }  
  
 public int getPlayerTwoWin() {  
 return playerTwoWin;  
 }  
  
 public void setPlayerTwoWin(int playerTwoWin) {  
 this.playerTwoWin = playerTwoWin;  
 }  
  
  
 public String getCurrentTimeStamp() {  
 return currentTimeStamp;  
 }  
  
 public void setCurrentTimeStamp(String currentDate) {  
 this.currentTimeStamp = currentDate;  
 }  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
}

**PlayerListAdapter.java**

package com.example.finalproject;  
  
import android.content.Context;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.List;  
  
*/\*\*  
 \* PlayerListAdapter - Adapter for the recycler view  
 \*/*public class PlayerListAdapter extends RecyclerView.Adapter<RecyclerView.ViewHolder> {  
  
 private List<Player> playerList;  
  
 public PlayerListAdapter(List<Player> list, Context context) {  
 super();  
 playerList = list;  
 }  
  
 class ViewHolder extends RecyclerView.ViewHolder {  
 public TextView mTextId;  
 public TextView mTextPlayerOneName;  
 public TextView mTextPlayerOneScore;  
 public TextView mTextPlayerTwoName;  
 public TextView mTextPlayerTwoScore;  
 public TextView mDate;  
  
 public ViewHolder(View v) {  
 super(v);  
 mTextId = v.findViewById(R.id.*txtId*);  
 mTextPlayerOneName = v.findViewById(R.id.*player\_one\_name*);  
 mTextPlayerOneScore = v.findViewById(R.id.*player\_one\_score*);  
 mTextPlayerTwoName = v.findViewById(R.id.*player\_two\_name*);  
 mTextPlayerTwoScore = v.findViewById(R.id.*player\_two\_score*);  
 mDate = v.findViewById(R.id.*date*);  
  
 }  
 }  
  
 @NonNull  
 @Override  
 public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View v = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*record\_layout*, parent, false);  
 ViewHolder viewHolder = new ViewHolder(v);  
 return viewHolder;  
 }  
  
 //Setting the data from database to the recyclerview  
 @Override  
 public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {  
 Player playerAdapter = playerList.get(position);  
 ((ViewHolder) holder).mTextId.setText(Integer.*toString*(playerAdapter.getId()));  
 ((ViewHolder) holder).mTextPlayerOneName.setText(playerAdapter.getPlayerOneName());  
 ((ViewHolder) holder).mTextPlayerOneScore.setText(Integer.*toString*(playerAdapter.getPlayerOneWin()));  
 ((ViewHolder) holder).mTextPlayerTwoName.setText(playerAdapter.getPlayerTwoName());  
 ((ViewHolder) holder).mTextPlayerTwoScore.setText(Integer.*toString*(playerAdapter.getPlayerTwoWin()));  
 ((ViewHolder) holder).mDate.setText(playerAdapter.getCurrentTimeStamp());  
 }  
  
 @Override  
 public int getItemCount() {  
 return playerList.size();  
 }  
}

**ViewScoreFragment.java**

package com.example.finalproject;  
  
import android.database.Cursor;  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.Toast;  
  
import java.util.ArrayList;  
import java.util.List;  
  
*/\*\*  
 \* ViewScoreFragment - Fragment for showing the scores of players who played  
 \*/*public class ViewScoreFragment extends Fragment {  
  
 public static final String *NO\_RECORDS\_FOUND* = "No Records Found";  
 public static String *TAG* = ViewScoreFragment.class.getName();  
 public static final String *ID* = "id";  
 public static final String *PLAYER\_ONE\_NAME* = "player\_one";  
 public static final String *PLAYER\_TWO\_NAME* = "player\_two";  
 public static final String *PLAYER\_ONE\_SCORE* = "player\_one\_score";  
 public static final String *PLAYER\_TWO\_SCORE* = "player\_two\_score";  
 public static final String *DATE* = "game\_timestamp";  
 private RecyclerView mRecyclerview;  
 private List<Player> playerList = new ArrayList<>();  
 private PlayerListAdapter mAdapter;  
 DatabaseHelper dbh;  
 private Button backBtn;  
  
  
 public ViewScoreFragment() {  
 // Required empty public constructor  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 View viewFragment = inflater.inflate(R.layout.*fragment\_view\_score*, container, false);  
  
 //getting the recycler view  
 mRecyclerview = viewFragment.findViewById(R.id.*recyclerView*);  
  
 //initializing database  
 dbh = new DatabaseHelper(getActivity());  
  
 //Getting all the available data using getAll parameter  
 Cursor cursor = dbh.viewData();  
 System.*out*.println("CURSOR: " + cursor);  
 //if no record was found, will toast a no records message.  
 //Otherwise will set the values to object  
 if (cursor == null || cursor.getCount() == 0) {  
 Toast.*makeText*(getContext(), *NO\_RECORDS\_FOUND*, Toast.*LENGTH\_SHORT*).show();  
 return viewFragment;  
 } else {  
 //Setting the values in object then added to list  
 if (cursor.moveToFirst()) {  
 do {  
 Player player = new Player();  
 player.setId(cursor.getInt(cursor.getColumnIndexOrThrow(*ID*)));  
 player.setPlayerOneName(cursor.getString(cursor.getColumnIndexOrThrow(*PLAYER\_ONE\_NAME*)));  
 player.setPlayerTwoName(cursor.getString(cursor.getColumnIndexOrThrow(*PLAYER\_TWO\_NAME*)));  
 player.setPlayerOneWin(cursor.getInt(cursor.getColumnIndexOrThrow(*PLAYER\_ONE\_SCORE*)));  
 player.setPlayerTwoWin(cursor.getInt(cursor.getColumnIndexOrThrow(*PLAYER\_TWO\_SCORE*)));  
 player.setCurrentTimeStamp(cursor.getString(cursor.getColumnIndexOrThrow(*DATE*)));  
  
 playerList.add(player);  
 } while (cursor.moveToNext());  
 }  
 }  
  
 //Closing the cursor and database then binding to the adapter needed  
 cursor.close();  
 dbh.close();  
 bindAdapter();  
  
 //Back button to home  
 backBtn = viewFragment.findViewById(R.id.*backBtn*);  
  
 backBtn.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 getParentFragmentManager().popBackStack();  
 }  
 });  
  
 return viewFragment;  
 }  
  
 //Binding the list to recyclerview by using the object adapter  
 private void bindAdapter() {  
 RecyclerView.LayoutManager layoutManager = new LinearLayoutManager(getContext());  
 mRecyclerview.setLayoutManager(layoutManager);  
 mAdapter = new PlayerListAdapter(playerList, getContext());  
 mRecyclerview.setAdapter(mAdapter);  
 mAdapter.notifyDataSetChanged();  
 }  
  
  
}

**CHALLENGES**

* **First challenge is design and how to approach the tic-tac-toe board.**
* **Next is the calling a dialogbox on click of another dialogbox.**
* **Next is the logic for getting the winning combination.**
* **Next is the back button for the fragment.**
* **Mostly also is how to transfer data from each fragment and how to have the score global.**

**REFERENCES**

* **Lecture from Professor.**
* **Youtube videos.**
* **Google.**