Proposed Game:

Tic-tac-toe

Game Title:

4Square

**GitHub Link:**

<https://github.com/thepineappleandthepie/FourSquareFinale>

**Mentor:**

Kudzai Maziriri

By

Nyiko Ashanti Ngobeni

G22N3906

Christopher McCune

G22M5298

Alisha Schuler-Nitzsche

G22S7874

Mihlali Xhala

G22X7152

Mthobisi Latha

G22L5863

**Content page**

Overview 3

Goals and risks

Gantt Chart 4

Technologies

Technical architecture 5

Storyboard interface representation 6

Object diagram (UML) 8

Code Outline 9

Code Summary

Difficulties Faced

**Overview**

On opening the app, the user will see the home activity, and from there they can click either the play button or the rules button, both will change the user’s view to suit what button they clicked. The rules activity gives an explanation of the rules. Each activity will have two buttons the user can click, one will allow them to always return to the home activity and the others will allow them to continue the game.

To code the game to work we will make a 2D array that stores mutable values that will change as the game goes on and players choose the move they want to make. To simplify the coding of board we implemented buttons across the actual game board in the squares that the player clicks on to place a symbol. As part of our rules, each player’s first move is required to be on one of the 12 outer blocks. After each move is made the code will check if someone has won or if there is a draw. If this occurs the view for the users will change to the results activity where, the board will be reset to blank and the user can click on the home button or the play again button which are both displayed as options.

**Goals and Risks**

We want to create a working android app that is our own creation in that it is a unique version of the game tic tac toe. Our goals, hence are centred around the keywords; “working” and “unique”. For the sake of completing this project, we as a group have identified that a desirable takeaway is to understand how the creation of a functioning android app happens and we would rather stray away from overcomplicating our unique game. The uniqueness of our version of tic tac toe is that we apply a four-by-four grid with the requirement of the first move to be on one of the outer blocks. This variation is enough for our game to be original and only poses the challenge of figuring out how to encode our first move requirement. The risks we may be undertaking due to our inexperience would be categorized into what we can and can’t identify. Categorically we are aware that our variation poses that challenge and creating an android app that fits onto a phone screen, in the required time, without overcomplicating our task.

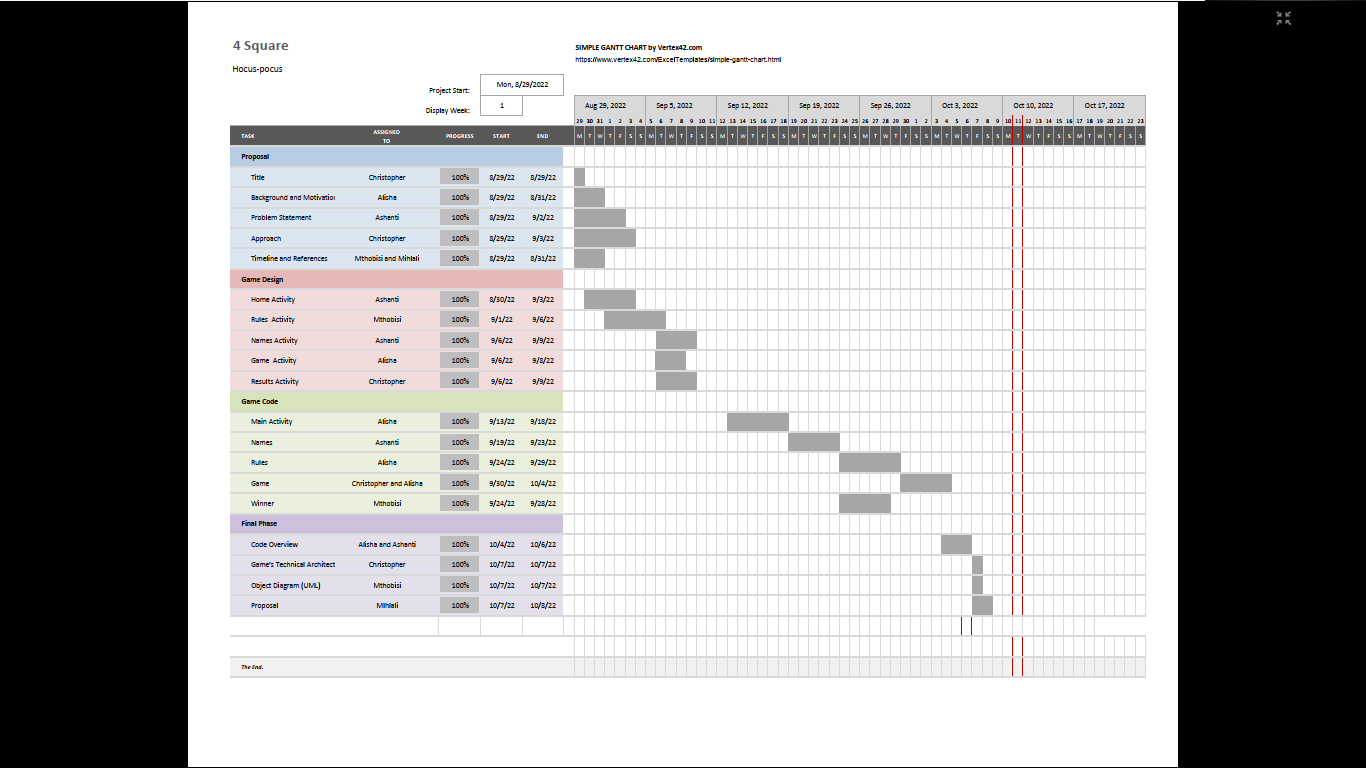


Figure 1: Gantt Chart for project.

**Technologies**

USER’S POINT OF VIEW

HOME

play

rules

**PLAY**

RULES

Player’s Names

play

home

Game button takes you HERE, where player’s names are added and symbols are assigned

**HOME**

**PLAY**

Rules

home

play

Rules are displayed HERE. You can return home or play game.

**HOME**

**PLAY**

RESULTS

home

play again

HERE the result is displayed. You can then play again or return home.

PLAY AGAIN

**HOME**

Figure 2: Games Technical Architecture.

Game

restart

home

HERE the game is played. You can quit and restart. Once someone won the game the player will immediately be taken to the results activity.

**RESTART**

**HOME**

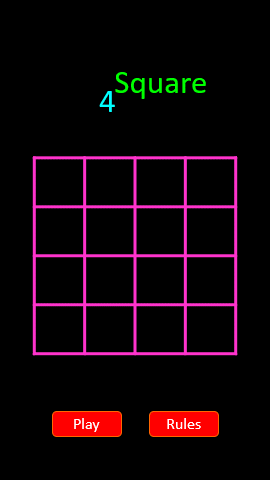
****

Figure 3: How players could potentially use the app*.*

**Story Interface Representation**

Shape

Description automatically generated with medium confidenceShape

Description automatically generated with medium confidenceShape

Description automatically generated with medium confidenceShape

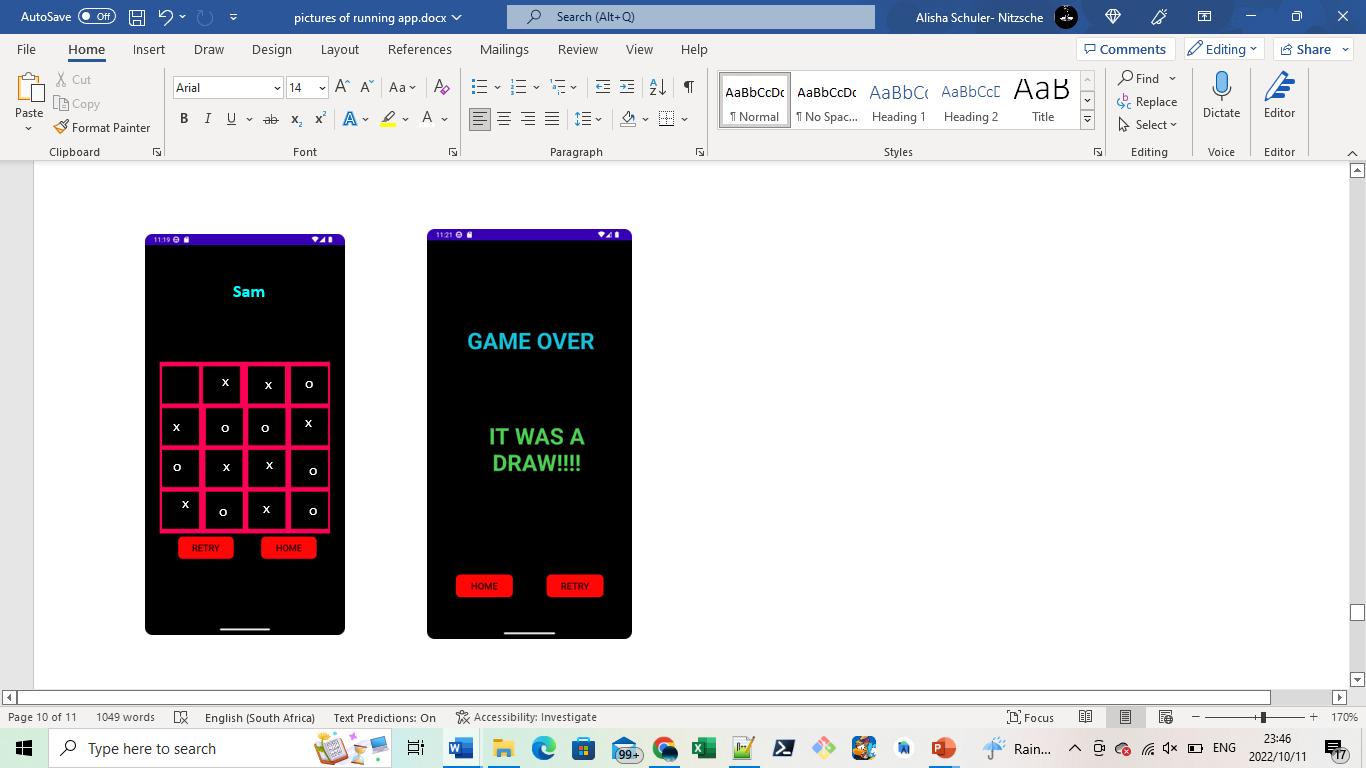
Description automatically generated with medium confidenceShape

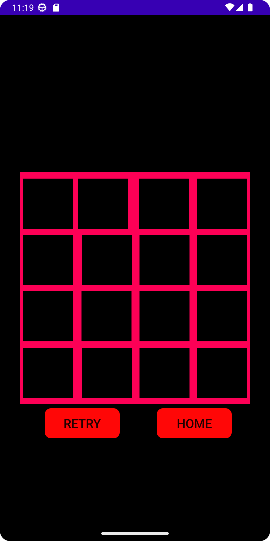
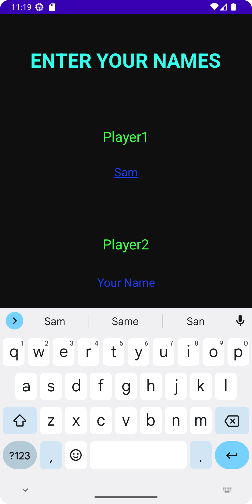
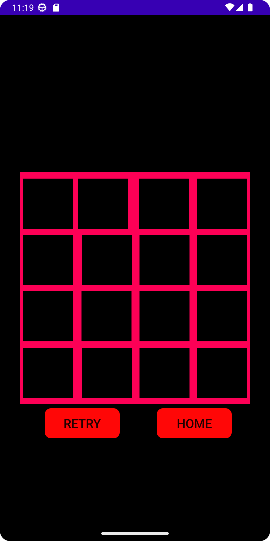
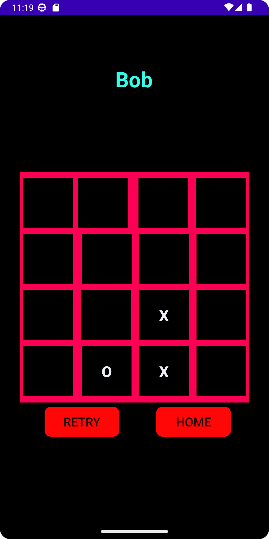
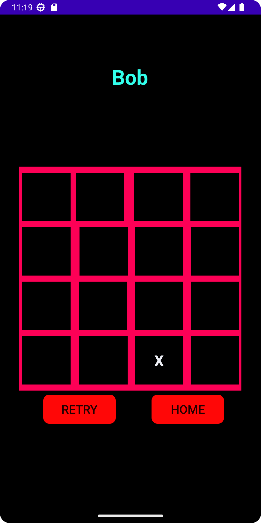
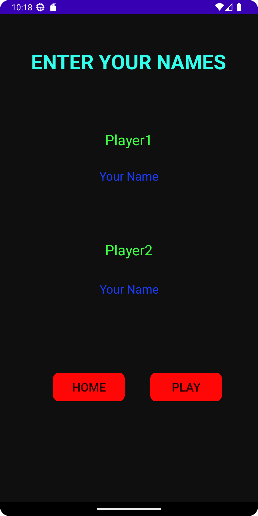
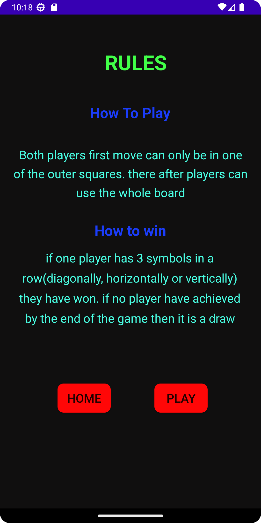
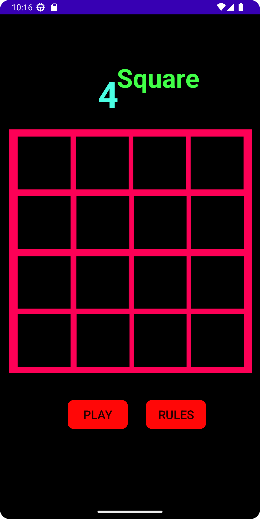
Description automatically generated with medium confidenceShape

Description automatically generated with medium confidenceShape

Description automatically generated with medium confidenceShape

Description automatically generated with low confidenceShape

Description automatically generated with low confidence****



**Sam**

x

o

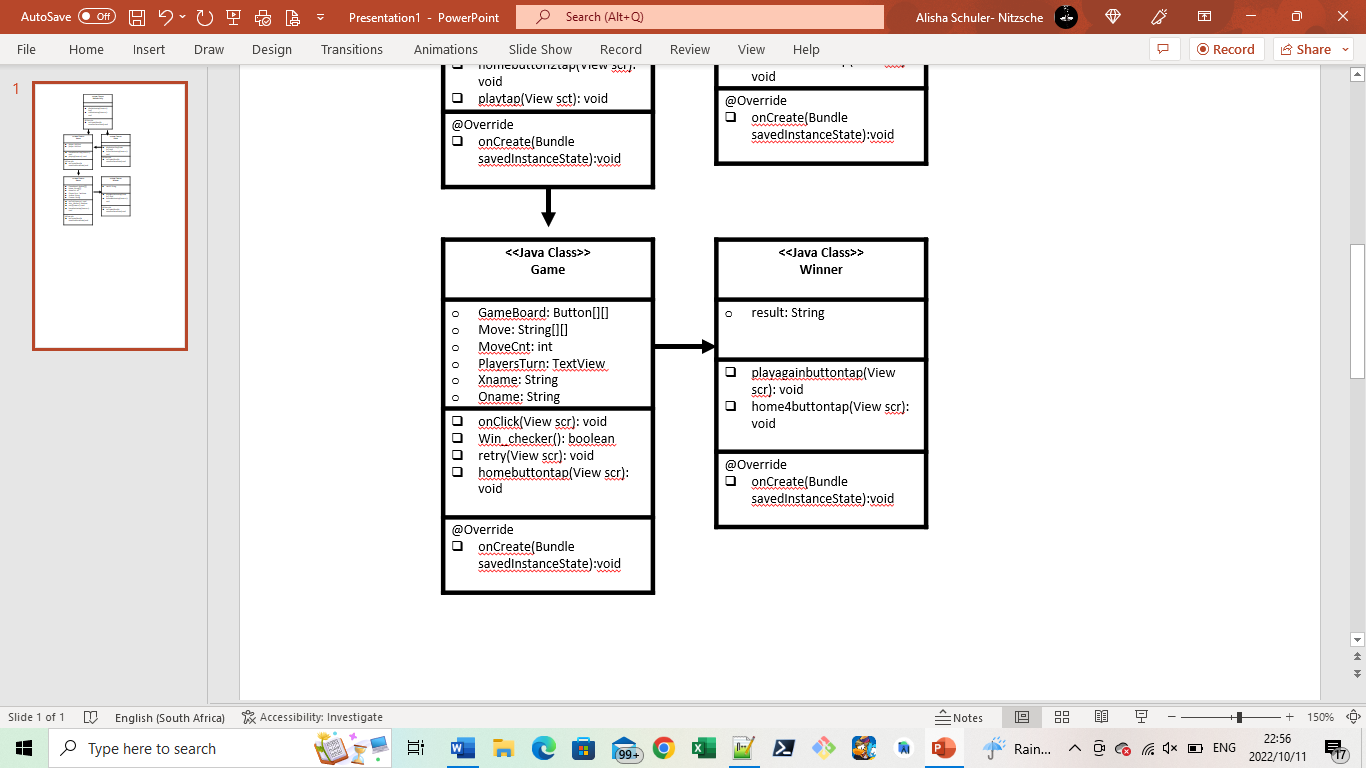
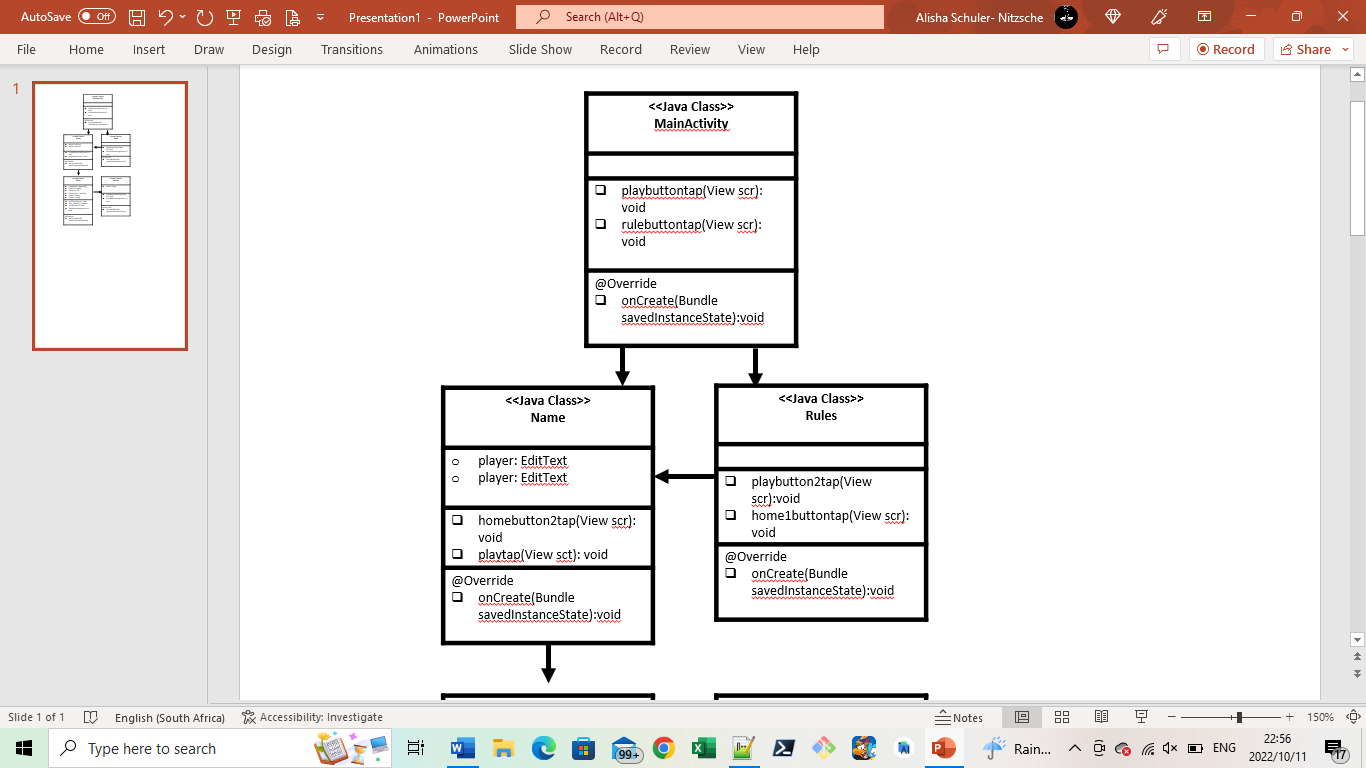


Figure 5: Object diagram (UML)

**Code Outline**

**Code Summary**

A really important part of our task was designing the activities that the user will interact with, we did this at the beginning of the creation the code. We started designing the activities which involved coding the button format so that they will be able to call the necessary methods, coding in the various String resources, adding the TextViews and hardcoding the String resources, adding the buttons in their places, constraining the TextViews and the buttons in their places, changing the text colours and formats, and choosing and applying a background to all of the activities.

The game board that the game will be played in needed to be coded in a particular way. We coded it as a board of buttons that is placed in a 2D array. We then implemented a checker to determine if the button clicked is empty or not and a move counter to determine whose turn it is, when it was even its X’s turn and odd was O’s turn. Next the necessary code to print the correct symbol into the button that gets clicked was configured and put into place.

We coded a win checker by a making a String array where each symbol that gets added to a button gets added to a two-dimensional String array and then by use of a for-loop, we check for wins; three of the same symbol placed in a row, either horizontally, vertically, and diagonally, which are all checked for. The diagonal wins are checked for manually. The win checker method is called after each move is made.

The next action that happens when someone wins or there is a draw was coded so that the user’s will be taken to the result activity and the result is displayed to the user. During each player’s turn, the name they entered is displayed above the active gameboard, we did this by bringing across a bundle- that contained the players name- from the Names activity and set those names to the TextView.

The code for each of the working buttons and their overall combination was finally combined in one new Android studios project. To check that each activity worked as it should we added each part using the waterfall model, fixing any errors that occurred.

**Difficulties Faced**

Originally we tried hard coding a drawable resource board but as we lack experience in this field we failed, thus we made a playing board out of buttons.

Another struggle we had was bringing the user input from one activity to the next but we did manage to fin a way using bundles.

A major issue for us while coding this game is that we do not have a private computer and the lab computers do not allow us to install HAXM, which is needed for the emulator on android studios to work.

Android studios also made it difficult to upload code to a previously made repository and hence we ended up making multiple different GitHub repositories. This is frustrating you as the markers do not see in what order we coded or who uploaded things.

Another major issue is that we could not get the win\_check method to work even though we had test run it on Notepad++. Converting it to android studios was a challenge and one we were sadly not able to complete, but the code for it is in our GitHub repository.