PROJECT REPORT FOR BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY

**TIC-TAC-TOE**



**RISHAV UPADHAYA [5-2-1113-64-2022]**

**SAKSHAM RIMAL [5-2-1113-67-2022]**

**SAPHAL POKHAREL [5-2-1113-72-2022]**

**SAUGAT KAFLE [5-2-1113-73-2022]**

**SAMRIDDHI COLLEGE**

**DEPARTMENT OF SCIENCE AND TECHNOLOGY**

**NOVEMBER 24, 2023**

PROJECT REPORT FOR BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY

**TIC-TAC-TOE**

**SUPERVISED BY Er. MOHAN BHANDARI**

**FACULTY, SAMRIDDHI COLLEGE**

A REPORT SUBMITTED

FOR

SECOND SEMESTER OOP’S PROGRAMMING PROJECT

**RISHAV UPADHAYA [5-2-1113-64-2022]**

**SAKSHAM RIMAL [5-2-1113-67-2022]**

**SAPHAL POKHAREL [5-2-1113-72-2022]**

**SAUGAT KAFLE [5-2-1113-73-2022]**

**SAMRIDDHI COLLEGE**

**DEPARTMENT OF SCIENCE AND TECHNOLOGY**

**NOVEMBER 24, 2023**

**DECLARATION**

I hereby declare that this project entitled **TIC-TAC-TOE** is based on my original research work. Related works on this project by other researchers have been duly acknowledged. I owe all the liabilities relating to the accuracy and authenticity of the data and any other information included hereunder.

Rishav Upadhaya

Saksham Rimal

Saphal Pokharel

Saugat Kafle

**Date: 2023-11-24**

**RECOMMENDATION**

This is to certify that this project entitled **TIC-TAC-TOE** prepared and submitted by **Rishav Upadhaya, Saksham Rimal, Saphal Pokharel, Saugat Kafle** for second semester C++ Programming Project of Bachelor of Computer Science and Information Technology awarded by Tribhuvan University, has been completed under my supervision.

**ER. MOHAN BHANDARI**

Faculty, Samriddhi College

Date: November 24, 2023

**ACKNOWLEDGEMENT**

At First, we would like to thank Samriddhi College, for providing this excellent opportunity to participate in this project and providing us with the environment and facility so that we could complete the project on time without any time constraints and any pressures. Also, we would also like to thank the faculty for giving us this opportunity to showcase our knowledge and understanding of C++ Programming and OOP’s concepts associated with it. We would also like to extend our thanks to Mr. Mohan Bhandari for guiding and sharing his invaluable knowledge on C++ and OOP’s concepts. Lastly, we would also like to thank our seniors and friends who have contributed their time to our project by helping us proofread, debug, and guide us along the way. This endeavor wouldn’t be successful without the involvement of these parties and although not named we would also like to thank anyone remotely associated with this project and we would like them to know that their helps and advice have been invaluable in this endeavor.

**ABSTRACT**

Tic-Tac-Toe is a simple multiplayer game that is played on a 3x3 grid board where the player takes turns and puts their symbols on the grid box. Players mostly use the symbol cross (X) and circle (O) to fill the grid. Thus, it is a very simple game that anybody with a PC can pick up and play.

Our project, Tic-Tac-Toe, begins with the input area on the console where the users put their username to play the game. Then the grid box of 3x3 is shown on the console and the player chooses their position to put their respective symbol. Users choose the number shown inside the grid box and press the number to print their respective symbol on that position.

The Tic-Tac-Toe algorithm we have developed gives the user the chance to compete with another user. Our Algorithm declares the winner as the original algorithm, accurately. The program includes features such as user friendly interface, options for looking at the name of the player1, player2, moves and the status of the games who won the game.

Overall, Tic-Tac-Toe provides an engaging and entertaining experience to the users, also enhances the decision-making abilities.

Keyword: *Tic-Tac-Toe, tictactoe, Circle Cross Game*

TABLE OF CONTENT

[LIST OF FIGURES vi](#__RefHeading___Toc1603_3729750098)

[LIST OF TABLES vi](#__RefHeading___Toc1603_3729750098)i

[LIST OF ABBREVIATIONS vi](#__RefHeading___Toc1603_3729750098)ii

[1. INTRODUCTION 1](#__RefHeading___Toc3766_2489888596)

[1.1 INTRODUCTION 1](#__RefHeading___Toc3768_2489888596)

[1.2 PROBLEM STATEMENT 1](#__RefHeading___Toc3770_2489888596)

[1.3 OBJECTVES 2](#__RefHeading___Toc3772_2489888596)

[1.4 SCOPE 2](#__RefHeading___Toc3774_2489888596)

[1.5 OVERVIEW OF REPORT 2](#__RefHeading___Toc3776_2489888596)

[2. BACKGROUND STUDY AND LITERATURE REVIEW 3](#__RefHeading___Toc2644_530581198)

[2.1 BACKGROUND STUDY 3](#__RefHeading___Toc3780_2489888596)

[2.2 LITERATURE REVIEW 4](#__RefHeading___Toc3782_2489888596)

[3. SYSTEM ANALYSIS AND DESIGN 6](#__RefHeading___Toc3784_2489888596)

[3.1 SYSTEM ANALYSIS 6](#__RefHeading___Toc3786_2489888596)

[3.1.1 REQUIREMENT ANALYSIS 6](#__RefHeading___Toc3788_2489888596)

[3.1.2 FEASIBILITY ANALYSIS 8](#__RefHeading___Toc3790_2489888596)

[3.2 SYSTEM DESIGN 8](#__RefHeading___Toc3792_2489888596)

[3.2.1 ALGORITHM 8](#__RefHeading___Toc3794_2489888596)

[3.2.2 FLOWCHART 9](#__RefHeading___Toc3796_2489888596)

[4. RESULT AND DISCUSSION 10](#__RefHeading___Toc3798_2489888596)

[4.1 RESULT 10](#__RefHeading___Toc3800_2489888596)

[4.2 TEST CASES 12](#__RefHeading___Toc3802_2489888596)

[5. LIMITATION 16](#__RefHeading___Toc3804_2489888596)

[6. FUTURE WORK 17](#__RefHeading___Toc3806_2489888596)

[7. RECOMMENDATION AND CONCLUSION 18](#__RefHeading___Toc3808_2489888596)

[7.1 RECOMMENDATION 18](#__RefHeading___Toc1832_1815237228)

[7.2 CONCLUSION 18](#__RefHeading___Toc3810_2489888596)

[REFERENCES 19](#__RefHeading___Toc3814_2489888596)

LIST OF FIGURES

[FIGURE 3.1.1.1: USE CASE DIAGRAM FOR THE PROJECT 6](#Figure!0|sequence)

[FIGURE 3.2.2.1: FLOWCHART OF THE PROJECT 9](#Figure!1|sequence)

[FIGURE 4.1.1: SCREENSHOT OF THE MENU SCREEN 10](#Figure!2|sequence)

[FIGURE 4.1.2: SCREENSHOT OF THE GAME SCREEN 11](#Figure!3|sequence)

LIST OF TABLES

TABLE 4.2.1: TEST CASE 1 13

TABLE 4.2.2: TEST CASE 2 14

TABLE 4.2.3: TEST CASE 3 15

TABLE 4.2.4: TEST CASE 4 16

# 

# LIST OF ABBREVIATIONS

API : Application Programming Interface

EXEC : Executable

FPS : Frames Per Second

GL : Graphics Library

GUI : Graphical User Interface

OOPL : Object Oriented Programming Language

1. **INTRODUCTION**
   1. **INTRODUCTION**

Tic-tac-toe also known as, noughts (digit zero) and crosses. It was firstly invented in Egypt while. Roofing tiles reappeared in the 18th century in England to entertain children. It is still played widely in many countries around the globe [1].

The simplicity yet competitiveness of this game makes it more interesting. Usually, it is played between two players and each player must choose from noughts and crosses. They put X’s or O’s alternatively in grid. The players have to place their marks horizontally, vertical, diagonally or in continuous line. The first person to get their marks simultaneously in those lines wins the match and if both of the players fail to match their marks simultaneously then the game is a draw. “You can live life like a tic-tac-toe game either way there can only be one winner”, so its competitiveness and mind game also represent life [2].This game is short and can be played anywhere at any time so it can be played for time-pass or entertainment purposes. **Mathematicians were intrigued by the game**. Its simple rules and small grid can illustrate mathematical principles like probability. Combinatorics show that there are 362,800 distinct ways to place X’s and O’s on the grid, but only 255,168 of them are possible winning combinations [3].

This game can be played on any device regardless of low end or high end so it has compatibility, and the user can take the experience of this game for free. The code was written in C language, so it is reliable and versatile.

* 1. **PROBLEM STATEMENT**

In the 21st century people mostly depend upon machines. The game provides entertainment as well as problem solving skills. It can also be beneficial to the children as they can learn to identify the digits, grid, shape, counting numbers of steps and they can use their brain at their early age to improve their problem-solving skill. This can be beneficial to disable people as well for example the blind people can feel the grid and by counting the number of steps and with proper assistance, they can speak to fill the grid.

Thus, this game can entertain people and can increase their cognitive skills, counting and spatial skills, shape identification. It can help improving their strategy, tactics, and observation and is available for all including disable persons.

* 1. **OBJECTIVES**

The objectives of the project are given below:

1. To make the simulation of real time tic tac toe game using C++ and OOP’s concepts.
   1. **SCOPE**

The game is free and available for all, and it is applicable in any place so people can use it anytime and anywhere regardless of the specification of the computer. The game is easy to learn and consumes less time so people can be more interested in this game. We used conditional statements and other basic object-oriented programming concepts. Simple C++ languages like if else, switch and class-based object-oriented concepts are used to create the project, so it is easy to understand and implement.

* 1. **OVERVIEW OF REPORT**

The report contains 7 chapters including introduction, background study, system analysis and design, result and discussion, limitation, future work and the recommendation and conclusion. The introductory part of this report contains the brief of the game, problem statements, objectives of creating the game, scope that are found in the society, and the overview of the report.

Next, the background study includes the reason of creating the project and the literature review where we surf the information in internet. Next, system analysis contains the requirements that were needed to complete the project and the feasibility of the project that is easily available. The system design contains the algorithm and flowchart of the code that are to be executed. The result and the discussion of the project in written in the next chapter and the test cases are included explaining what kind of errors were found. Limitation and the future work of the program is written. At last recommendation and conclusion of the project is written to end the report.

Thus, these are the overview of the report to explain the chapters and processes of creating the game “tic-tac-toe”.

**2. BACKGROUND STUDY AND LITERATURE REVIEW**

**2.1 BACKGROUND STUDY**

The project tic-tac-toe was done because it is used widely around the globe and decided to do our project making tic-tac-toe because it required less memory, low specification, no graphics card thus, it is available for all, and all the people can play this game regardless of their computers. This game is so popular and almost everyone knows how to play it or has heard of this game at least once in their life. So, the group decided to do this project because of availability, simple ness, versatility, and reliability.

C++ language was used to make this project because C++ is a more powerful language that could support object-oriented programming features. While C++ is a programming language widely used in programming games and fit for creating simple games. Thus, C++ can be easily used in any PC because it can be easily compiled, and it is easy to understand and easy to use.

Branching statements are used in the program including if else statements because there are two players playing at the same time thus when the player 1 gives data then the control of the program goes to take data from player 2 and determine the winner of the game using if else statements. Within the Vs Computer mode, a single player competes against a computer-controlled opponent. When it's the computer's turn, it will make a random move and make sure the game plays out correctly, with each player's turn involving the right action. As the branching statements in C++ programming are easy to use and easy to understand and is widely used in C++ programming to make simple project consuming less time, branching statement are used for the control of the program.

Thus, these are the reasons and the background of making the project in C and making tic-tac-toe for our project.

**2.2 LITERATURE REVIEW**

Reviews of the tic-tac-toe research articles, and the findings are:

The game was first developed in the 18th Century, the game was firstly invented to entertain the children’s and some portions of tic-tac-toe or the sample of tic-tac-toe was seen playing by children’s at that time. It is believed that the game made its way to, and became popular in, America and the rest of the world [3].

Traditionally, the Tic-tac-toe game was a pencil and paper game played between two players and each player must choose from noughts and crosses. They put X’s or O’s alternatively in grid. The players must place their marks horizontally, vertical, diagonally or in a continuous line.

The first video game that used tic-tac-toe as the game was in Cambridge University. The player could play the perfect game against a computer at that time. Then the tic-tac-toe was again experimented on MIT and the game could be perfectly played and finally the official tic-tac-toe game was invented [4].

The game has various name in various countries, and they are:

1. Tick-tack-toe, tic-tac-toe, tick-tat-toe, or tit-tat-toe in United States, Canada, India.
2. Noughts and crosses in United Kingdom, New Zealand, Australia, India, etc. [4]

Thus, these were the finding our group got know from different social media sites and surfing all over the internet.

# SYSTEM ANALYSIS AND DESIGN

## SYSTEM ANALYSIS

### 3.1.1 REQUIREMENT ANALYSIS

1. **FUNCTIONAL REQUIREMENT**

Some of the major entities of our program with their functions are:

1. Menu Page: It includes the menu and options to start game and others.
2. Start Game: It includes the input screen for the usernames and the game box.
3. Grid Box: It shows the position where the user can put their keys on it.
4. Circle and Cross: It is the symbol user use to play the game.
5. Game End: When the symbols are aligned together, either straight or diagonally then the game ends.
6. Records: During the execution of the game, data like usernames, status of who won the game and the number of moves is shown on the console.
7. **NON-FUNCTIONAL REQUIREMENT**
8. **ACCESSIBILITY**

Any User can get easily access to our game. It is easily available over the internet; users can share the game through pen drive or can download it from internet. Users don’t have to pay anything to get the game. It is the best game to get engaged and enhances one’s decision-making ability.

1. **PERFORMANCE**

Tic Tac Toe game can be used in any computer device. Since our game runs on console there is no problem with getting it on your computer. The game doesn’t have any glitches and malfunctions.

1. **APPERANCE**

The program (code) has the appearance of the original tic-tac-toe game. But, due to certain changes on the code makes it different to the original game. Program uses the grid like box structure to run the game in it. Positions of every box are showed where user can put their symbols to.

### FEASIBILITY ANALYSIS

1. **TECHNICAL FEASIBILITY**

Program is developed using C++ programming language, So, one who have the knowledge of C++ programming Language can fix the error of the code.

1. **OPERATIONAL FEASIBILITY**

The operation cost of the program is very low nearly Zero. One needs to download the program to use the file. And all the necessary files will be downloaded to the user’s computer automatically.

1. **ECONOMIC FEASIBILITY**

Our game is free to download online via the internet. One can share the game very easily through the pen drive.

1. **SCHEDULE FEASIBILITY**

Our program only requires a weeklong time to code and the minor error can be easily fixed within a day.

## SYSTEM DESIGN

### ALGORITHM

1. Start
2. Show the menu page.
3. Ask for Input
4. When user press 1, press 2 or press 3, go to STEP 5, STEP 9 and STEP 10 respectively.
5. Ask for the name of User1 and User2.
6. Show the box.
7. Ask the user for the input of the symbol.
8. Go to STEP 6.  
   When the game ends then go to STEP 2
9. Ask for the name of User1.
10. Show the box.
11. Ask the user for the input of the symbol.
12. Go to STEP 10.  
    When the game ends then go to STEP 2
13. Show the history.
14. END

### FLOWCHART

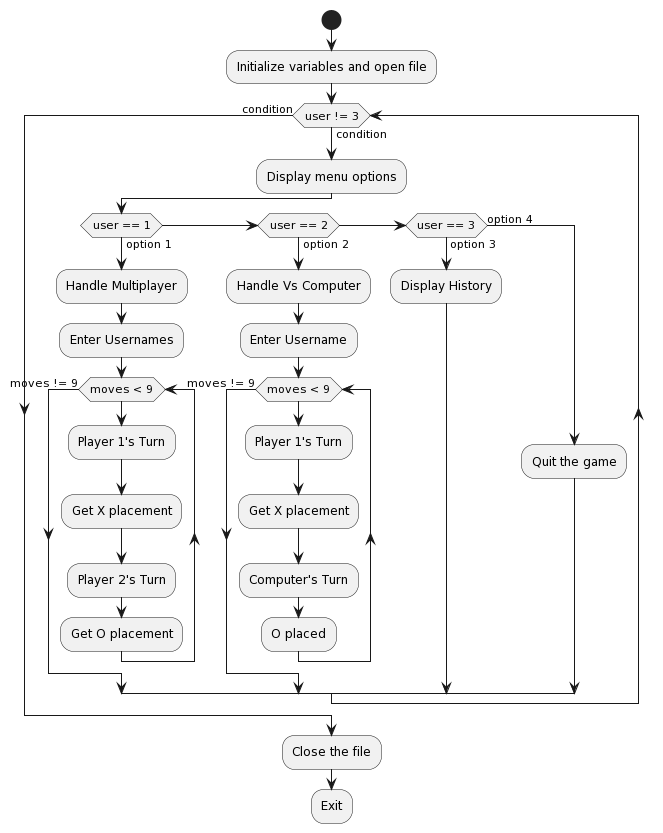


FIGURE 3.2.2: SCREENSHOT OF THE FLOWHCART SCREEN

# RESULT AND DISCUSSION

## RESULT

Following are the outputs of our program:

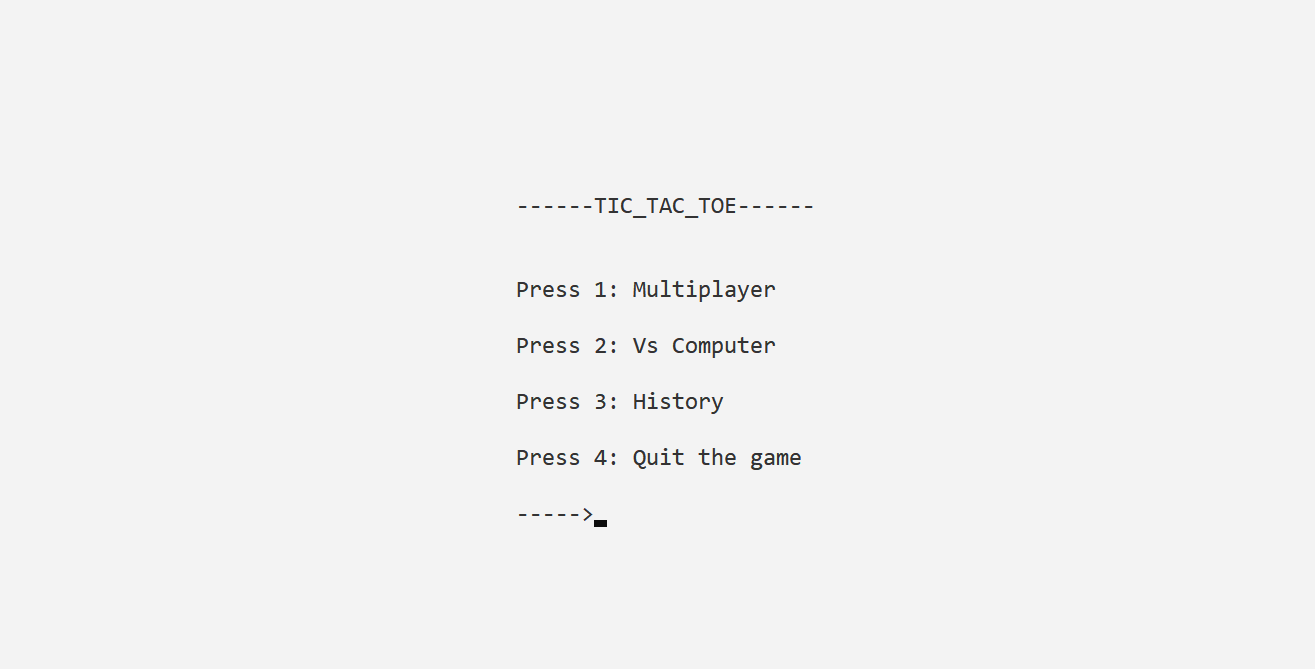


FIGURE 4.1.1: SCREENSHOT OF THE MENU SCREEN

The above screen is what the users will see first. The above screen is the menu screen of our program. Here, User is supposed to give the number as they would like to use the program. If user press 1 and enter, then user will be asked to enter the username for both the player and User will see the following screen:

A screenshot of a computer

Description automatically generated

FIGURE 4.1.2: SCREENSHOT OF THE GAME SCREEN

Here, the user will be asked to give the position based on where user would like to put their symbol.

Also, Users cannot choose their symbols, but by default, the player1 will be assigned the symbol of ‘X’ and the player2 will be assigned the symbol of ‘O’. As the user chooses the position to put their symbol then it will be shown on the screen.

A screenshot of a computer

Description automatically generated

FIGURE 4.1.3: SCREENSHOT OF THE GAME SCREEN AFTER GAME BEGINS

After the user gives the position then the alignment of the symbols is checked. Either they are in a straight order or diagonally order. If any of the users succeed in aligning the symbols then they are declared as a winner as shown in the fig below.

A screenshot of a game

Description automatically generated

FIGURE 4.1.4: SCREENSHOT OF THE WINNER SCREEN

The game is all about winning and losing, but the program also has the feature whether both of the user have no any chance of aligning those symbols and consider them as a draw.

A screenshot of a game

Description automatically generated

FIGURE 4.1.5: SCREENSHOT OF THE DRAW SCREEN

Also, the user only has 9 moves to win the game and there are certain logics to win the game. So, the users must use their brain at the fullest and made the decision which can turn them as a winner.  
As the users plays, the records like name of user 1, user2 and the status (who won the game) and the moves will be saved inside the computer. If the user wants to look who won the game , who played and what is the shortest move anyone have ever played can be shown as user press 2 in the menu page screen.

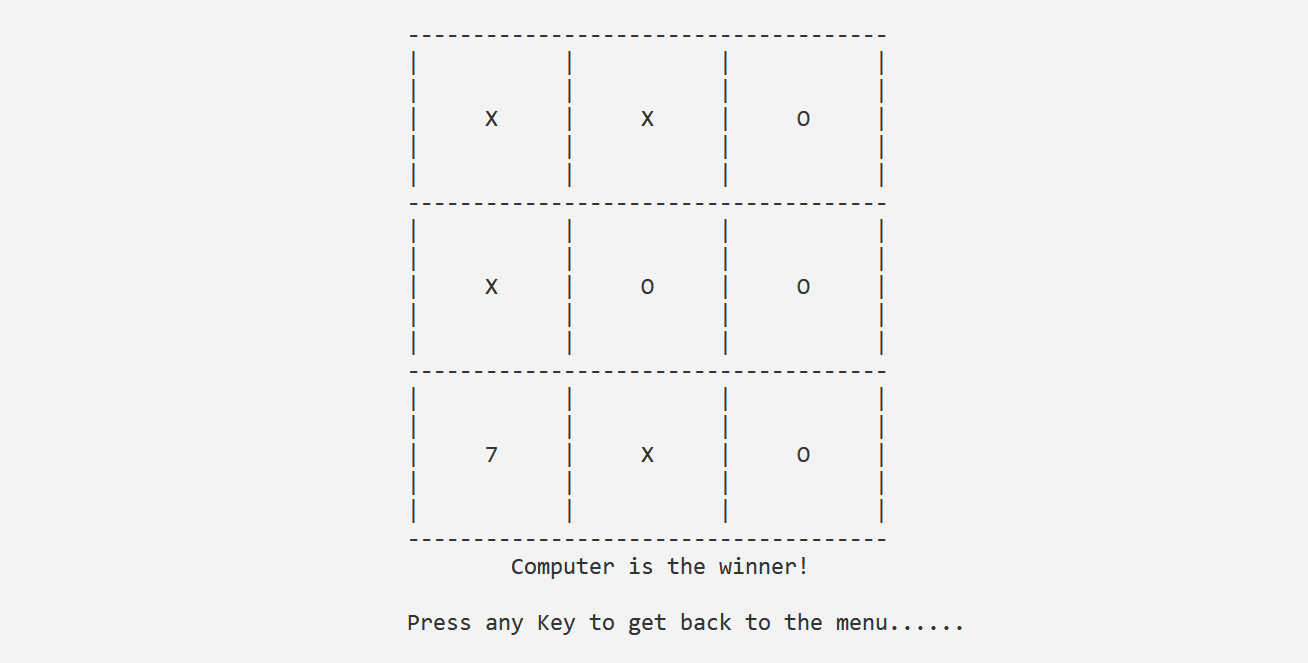


FIGURE 4.1.6: SCREENSHOT OF THE COMPUTER WINNER SCREEN

Here, Users are now able to play with computer. When user is at main menu then user needs to press2 and then the algorithm will ask for the user name and as user provide their name, then the game will start.

A screenshot of a computer

Description automatically generated

FIGURE 4.1.7: SCREENSHOT OF THE HISTORY SCREEN

The above figure is the screenshot of the history of the game played in the project. As the game is played the algorithm stores the user data and showcase it to the user as a history. The history on the project includes player1 name, player2 name, Winner name and the moves they use to finish the game.

## TEST CASES

|  |  |
| --- | --- |
| Test Case Identifier | TCOCH |
| Test Case Name | Start Game and Play |
| Test Case Description | Verify that the game starts, players can make valid moves ,and the game ends correctly with a winner or a draw. |
| Test Items | Tic -Tac -Toe Game |
| Input Specifications | 1.Press1 to start the game. |
|  | 2.Enter player1 name (without spaces) as "Alice ". |
|  | 3.Enter player2 name (without spaces) as "Bob ". |
| Output Specifications | 1.The Tic -Tac -Toe board is displayed. |
|  | 2.Players take turns to make moves. |
|  | 3.The game displays the winner's name or declares a draw after |
|  | the game ends. |
|  | Continued…. |
|  |  |
| Environmental Needs | Operating System: Windows or Linux. |
| Special Procedures /Prerequisites | None |
| Test Procedure | 1.Run the Tic -Tac -Toe game program. |
|  | 2.Select option to start the game. |
|  | 3.Enter player name an "Alice "and player2 name a "Bob ". |
|  | 4.Verity that the Tic -Tac -Toe board is displayed. |
|  | 5.Play the game with the following moves |
|  | -Player1 (Alice) selects position 1. |
|  | -Player2(Bob) selects position 5. |
|  | -Player1 (Alice) selects position 2. |
|  | -Player2(Bob) selects position 9. |
|  | -Player1 (Alice) selects position 3. |
| Expected Results | 1.The Tic -Tac -Toe board is displayed correctly after each move. |
|  | 2.Playeri (Alice) wins the game, and the game displays "Player1.  (Alice) is the winner!". |

TABLE 4.2.1: TEST CASE 1

|  |  |
| --- | --- |
| Test Case Identifier | TC002 |
| Test Case Name | Display Game History |
| Test Case Description | Verify that the game history is displayed correctly. |
| Test Items | Tic -Tac -Toe Came |
| Input Specifications | Press 3 to view the game history. |
| Output Specifications | Previous game records are displayed, including player names, |
|  | winners, and number of moves. |
| Environmental Needs | Operating System: Windows or Linux. |
| Special Procedures /Prerequisites | Play multiple games to create game history |
| Test Procedure | 1.Run the Tic -Tac -Toe game program |
|  | 2.Select option 2 to view the game history. |
| Expected Results | The previous game records are displayed correctly, showing |
|  | player names, winners, and moves used to win the game. |

TABLE 4.2.2: TEST CASE 2

|  |  |
| --- | --- |
| Test Case Identifier | TC003 |
| Test Case Name | Invalid Player Name |
| Test Case Description | Verify that the game handles Invalid player names without crashing |
|  | or displaying errors. |
| Test Items | Tic -Tac -Toe Game |
| Input Specifications | 1.Press 1to start the game. |
|  | 2.Enter player name with spaces, e.g.: “Alice Smith ". |
| Output Specifications | 1.The game should display an error message stating that player |
|  | names cannot contain spaces. |
| Environmental Needs | Operating System: Windows or Linux. |
| Special Procedures /Prerequisites | None |
| Test Procedure | 1.Run the Tic -Tac -Toe game program. |
|  | 2.Select option 1to start the game. |
|  | 3.Enter player 1 name as "Alice Smith "(with spaces ). |
| Expected Results | 1.The game should detect the Invalid player’s name and display an error message such as "Invalid player name. Player names cannot contain spaces." |

TABLE 4.2.3: TEST CASE 3

|  |  |
| --- | --- |
| Test Case Identifier | TC004 |
| Test Case Name | Incorrect. Winning Move |
| Test Case Description | Verify that the game correctly identifies the winning move and does |
|  | not declare a player as the winner when the winning condition is not |
|  | met. |
| Test Items | Tic -Tac -Toe Came |
| Input Specifications | 1.Press 1to start the game. |
|  | 2.Enter player name as "Alice ". |
|  | 3.Enter player2 name as ""Bob ". |
| Output Specifications | 1.The game should display the correct winner when a player meets |
|  | the winning condition. |
| Environmental Needs | Operating System: Windows or Linux |
|  | Continued… |
|  |  |
| Special Procedures /Prerequisites | None |
| Text Procedure | 1.Run the Tic -Tac -Toe game program. |
|  | 2.Select option to start the game. |
|  | 3.Enter player name an "Alice "and player2 name an "Bob ". |
|  | 4.Play the game with the following moves |
|  | -Player1 (Alice) selects position 1. |
|  | -Player2(Bob) selects position 5. |
|  | -Player1(Alice) selects position 2. |
|  | -Player2(Bob) selects position B. |
|  | -Player1 (Alice) selects position 3. |
| Expected Results | 1.The game should not declare a winner when the winning condition |
|  | is not met, and it should display the connect winner when the winning |
|  | condition is met .For example ,Player 1 (Alice ) wins before reaching |
|  | position 3, the game should not declare Player2 (Bob) as the winner. |

TABLE 4.2.4: TEST CASE 4

# LIMITATION

The Tic Tac Toe game have built has a lot of limitations. There is a lot of work that need to be done. Although the game was supposed to give a head-to-head fight to the original one, due to some limitations, it cannot. It lacks many things like Graphics, User Experience etc.

1. **Graphics:** Our game lacks GUI and different graphical effects.
2. **User Experience:** The game also lacks user experience. As the user experience is mostly dependent on the graphics. As, in the original game, the user can touch the screen and play the game which makes it a bit less tedious than ours.
3. **Skipping of the user chances after more than 2 mistake on the same position:** The users chance is skipped if the user put their symbol on the same position again and again (i.e more than 1 times).
4. **Users are limited to provide name without using spaces:** As the user runs the program, user needs to input their name, where they are unable to provide spaces on their names..

# FUTURE WORK

As the program has come this far and there was a lot to learn about c programming and the different logics behind it. As the team have come this far, team will also move ahead and work with different graphical libraries, more concepts to make the game even better.

1. **More User-Friendly:** I can also be said that, all the limitations are the future work for the program. There are a lot of things that need to be done. The program will first start by limiting the limitations on the game. The team will definitely work on Graphics and will use more graphics which makes the user feel more friendly to the game.
2. **Gathering User throughout the globe:** The user around the world will be added to the game, so that one user can look through the least moves that another player ever had played and won the game.
3. **No more limitation to provide name without using spaces:** The user will not have to worry about the pressing spaces during their names input.
4. **No skipping of the user turn even after same mistake for more than 1 times:** Users can play their chance again after the similar mistake, which will be a fair game.

# RECOMMENDATION AND CONCLUSION

## RECOMMENDATION

To those people who might take this report as reference, here are some suggestions and recommendations:  
First Recommendation: Only the common concept of C++ programming have been used, if one know the advanced concept of C++, then the program can be better. The graphics libraries can be used to make it look amazing. After all these changes, test cases must be checked, like whether it works on low end devices or not. Is there any malfunction? and so on.

Second Recommendation: Make sure to keep the expectations at the limits, the team also had a lot of expectations during the start of the program. But the journey was fun, we got to learn so many things.

## CONCLUSION

Those who are interested in building a game in C++ programming language, make sure to investigate different graphics libraries, advanced concept of C++ and you can also learn OOP’s and Computer Graphics to make a better version of it. You can also look through the addition of API’s. One must have a good grasp of the C++ programming language and some other graphical libraries to make it better. This project was a great experience and had a lot to learn.

This was the very first project, and the team learned a lot about the different concepts of C++ programming language. Some times whenever the error is seen through, we gets frustrated. We couldn’t find the error in the code. During this project, we also had the thought that there was a lot of work left and lots of knowledge are needed more to bring some more to the table. This was indeed the best experience.

# REFERENCES

[1] <https://ownquotes.com/quote/1751> -Christopher Reaves

[2] Interesting facts about tic-tac-toe, 27 November 2021. [Online], Available: <https://justfunfacts.com/interesting-facts-about-tic-tac-toe/> , [Accessed 21 July 2023].

# [3] Gamesver Team, Tic-Tac-Toe (Noughts & Crosses): Origins, History, Evolution, …, 27 November 2021. [Online], Available: <https://www.gamesver.com/tic-tac-toe-noughts-crosses-origins-history-evolution/> , [Accessed 21 July 2023]

[4] Gamescrafter, [Online] <http://gamescrafters.berkeley.edu/games.php?game=tictactoe>, [Accessed 24 July 2023]