Project Requirement and Specification

on

TIC TAC TOE

(CSE III Semester Mini project)

2020-2021

Submitted to : Submitted by:

Mr.Pankaj Kumar Rohit Singla

University Roll No. : 2014815

Guided by:

Dr.Ashook Sahoo CSE-D-3rd-Sem

(Resource Person) Session:2020-2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GRAPHIC ERA UNVERSITY, DEHRADUN

----------CONTENTS----------

1.1 ABOUT PROJECT----------

1.2 SYSTEM REQUIREMENT---------- 1.3

MODULE----------

1.4 DATA FLOW DIAGRAM---------- 1.5

REFERENCE----------

1.1 About Project

Tic-Tac-Toe is a two player game where each player uses a signature. The player who successfully places three respective signatures in a vertical, horizontal or diagonal row is the winner..

However, it is not necessary the case that every game has a winner when there is not a single grid and when neither player has managed to mark a winning row then it is called a tie.

Step 1: Use 3 X 3 matrix to get a new 9 cell tic tac toe board to play.

Step 2: Signature 'X' is used for player 1 and 'Y' for player 2. And the players can play alternatively.

Step 3: Get index as input from the players. And the index range is 1-9. i.e) from cell 1 to cell 9.

Step 4: After each move, check whether that player has placed three respective signatures in vertical, horizontal or diagonal row. If yes, declare that player as winner.

Step 5: If no, give chance to other player.

Step 6: If all the cells in the Tic-Tac-Toe board got filled, then the game is drawn.

1.2 Requirement of Project

1.2.1 Hardware Requirement:

- **PROCESSOR: INTEL® PENTIUM® CPU @ 2.0 GHZ**
- **RAM: 2 GB(MINIMUM)**
- **STORAGE: 5 MB(MAXIMUM)**
- **OS: WINDOWS 8**

1.2.2 Software Requirement:

- **CODEBLOCKS**
- **DEV C++**
- **TURBO C++**
- **ANY OTHER SOFTWARE SUPPORTING C and C++**

1.3 Modules of Project:

1.3.1 BOARD[3][3] :

BOARD[3][3] is our playing board and it is also a global 2D Array which will be the games or playing area.

So the :

1st Row consists {1,2,3}.

2nd Row consists {4,5,6}.

3rd Row consists {7,8,9}.

All these numbers are the positions or indexes of each box or of each index where Player will mark X or O.

1.3.2 ROW AND COLUMN :

ROW and COLUMN are also the global variable in the project.

1.3.3 INSTRUCTIONS() :

INSTRUCTION() function is used to tell users about some rules of the game.

Necessary rules of the game will be display to the users or players by this function.

1.3.4 DISPLAY_BOARD() :

DISPLAY_BOARD() function is created to display the playing board marking the positions of each box and is updated regularly in PLAY_GAME() function according to the game. This function is called each time the players gets chance and fills something in the game board.

1.3.5 PLAY_GAME() :

PLAY_GAME() function takes the variable TURN as its argument.

If TURN = 0 , then it is the turn of PLAYER [X] HARRY , otherwise
If TURN = 1 , then it is the turn of PLAYER [O] MOHIT.

Then it ask the player to enter the position where he wants to enter the X or O.

Then it call the SET_POS() function to mark the character X or O at the position specified by the respective player.

If SET_POS() returns Boolean value (FALSE or 0), then TURN remains 0 i.e current player re-plays his/her turn, otherwise,

If SET_POS() returns Boolean value (TRUE or 1), then TURN changes to 1 i.e the chance shifts to other player.

Then its calling the DISPLAY_BOARD() function each tim to display the updated board and after that call the CHECK_WIN() function whether the current player win or not.

If it returns TRUE then the control goes back to MAIN() function and the match is finished.

Else it re-calling the PLAY_GAME() function recursively until the game is finshed.

1.3.6 SET_POS() :

It takes two arguments POS and TURN .

If POS is not between 1 and 9 , then it print a message "Invalid Position..!!!\n Play

Again" and returns a Boolean value (FALSE).
Else,

Switch() statements are invoked taking POS as argument and according to POS we set the value of global variable ROW and COLUMN.

Then it check whether the position is already filled or not and place X or O according to the turn of player and returns a Boolean value (TRUE).

Else, it prints the message "Position already filled..!!\n Please enter a valid position." And returns a Boolean value (FALSE). And this continues till the user enters a valid position.

1.3.7 CHECK_WIN() :

It takes TURN as an argument and is created to check if any of the player has won the game and if the game should stop.

Then it checks whether the player mark the winning streak vertically , horizontally or diagonally. It means that this function checks if the player is having his respective character X or O in s streak of 3 either vertically or horizontally or diagonally and if he have then he has won the game.

If yes , then it checks whose turn it is and print the appropriate message for the winner and returns a Boolean value (TRUE).

If no, then it checks whether there is any space left and if any space is left then it returns a Boolean value (FALSE).

If above two conditions are falsed then it print the message "Match is Tie " and returns a Boolean value (TRUE).

1.3.8 In MAIN() :

In MAIN() I have taken two strings or char arrays PLAY1 and PLAY2 and then we have called INTSRUCTIONS() to display instructions..

Then we prompt user for the name of players.

If the name entered by the users does not matches the pre-registered players for game like "HARRY" for PLAYER [X] and "MOHIT" for PLAYER [O] , then it will show and error and the user exit the game.

Else, it will call the DISPLAY_BOARD() function with number already mentioned for each position.

Then ask the user which player want to play first and input it in TURN variable and then call the PLAY_GAME() function with passing TURN as an argument.

ANY OTHER RELEVANT INFORMATION SUCH AS (E-R DIAGRAM, DFD ETC)

1.4 ----------Data Flow Diagram ----------

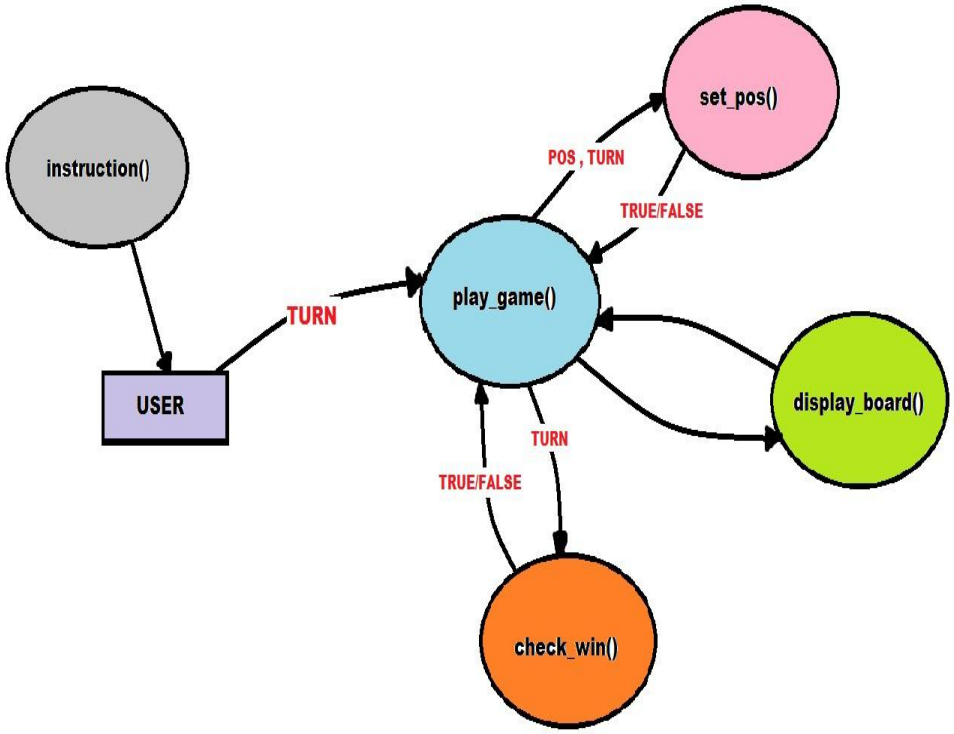

FIG. Data flow diagram of tic-tac-toe

REFERENCE

1. GEEKS FOR GEEKS.
2. W3 SCHOOL.
3. STACK OVERFLOW.
4. BOOKS :

- ANSI C 2019 EDITION

- C IN DEPTH