**REPORT**

# **C Programming on Multiple Platforms - Project Based Learning Approach**

NAME: Naveen Rathod

PS NO: 99007684

**Project Title: Tic-Tac-Toe game**

[**C Programming on Multiple Platforms - Project Based Learning Approach** 1](#_Toc96984762)

[**1. Requirements** 2](#_Toc96984763)

[**1.1 Description** 3](#_Toc96984764)

[**1.4 High level requirements** 3](#_Toc96984765)

[**Table 1.4** 4](#_Toc96984766)

[**1.5 Low level requirements** 4](#_Toc96984767)

[**1.6 SWOT Analysis** 4](#_Toc96984768)

[**1.7 4W's-1H** 5](#_Toc96984769)

[ **WHO -** 5](#_Toc96984770)

[ **WHAT -** 6](#_Toc96984771)

[ **WHEN -** 6](#_Toc96984772)

[ **WHERE -** 6](#_Toc96984773)

[ **HOW -** 6](#_Toc96984774)

[**4. Test plan and Output** 11](#_Toc96984775)

[**4.1 High level Test plan** 11](#_Toc96984776)

[**Table 4.1** 12](#_Toc96984777)

[**4.2 Low level Test plan** 12](#_Toc96984778)

# 

# **1. Requirements**

# **1.1 Description**

* Tic-tac-toe is a game in which two players seek in alternate turns to complete a row, a column, or a diagonal with either three O's or three X's drawn in the spaces of a grid.Tic-tac-toe, noughts and crosses, or Xs and Os is a game for two players who take turns marking the spaces in a three-by-three grid with X or O. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner. If both players are not able to mark in all the possible ways then the game is draw.

**1.2 Identifying feature**

1. Showing the grid every time after a mark is done by the player.
2. Display the player who has won with the grid.
3. Display the grid with draw message when the game is drawn.
4. If player enters wrong input display as wrong input and give another to enter correct input.

**1.3 State of Art/Research**

Tic-tac-toe is played on a three-by-three grid by two players, who alternately place the marks X and O in one of the nine spaces in the grid. This can be implemented in C using the functions.

## **1.4 High level requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **ld** | **Description** | **Category** | **Status** |
| HLR1 | Displaying which player goes first | Technical | Implemented |
| HLR2 | Enter player choice | Technical | Implemented |
| HLR3 | Display if a player wins | Technical | Implemented |
| HLR4 | When game is tie restart the game freshly| | Technical | Implemented |

## **Table 1.4**

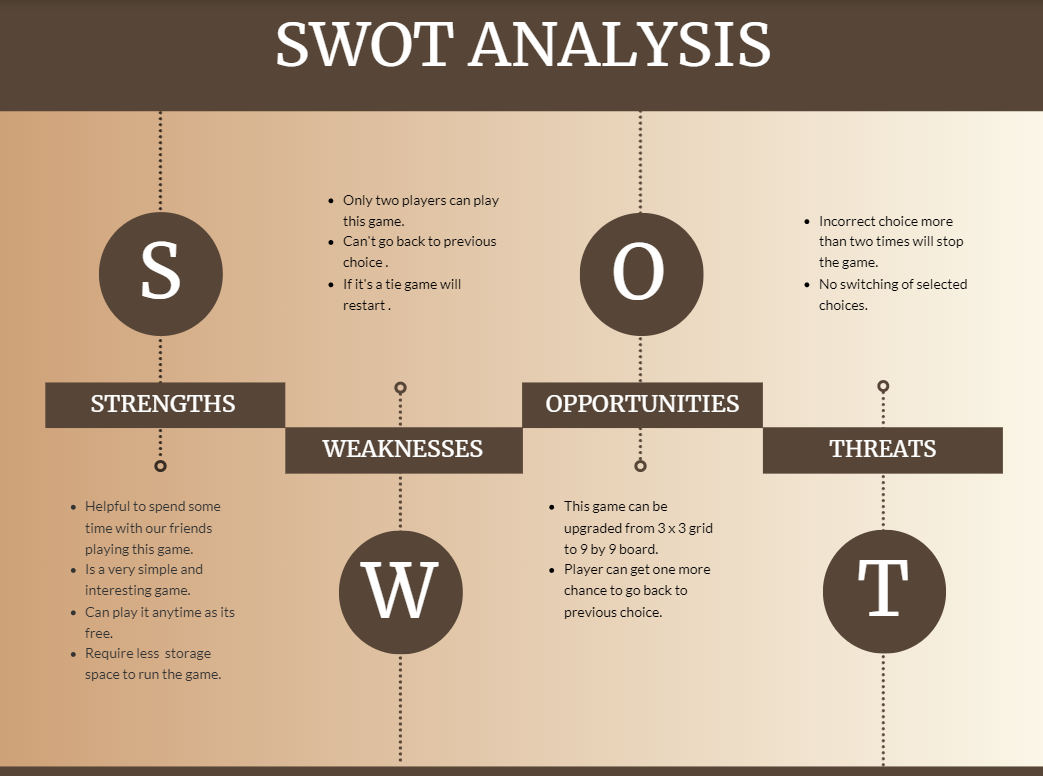
## **1.5 Low level requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **ld** | **Description** | **Category** | **Status** |
| LLR1 | Selecting 'X' or 'O | Technical | Implemented |
| LLR2 | If player enters wrong choice again a chance is given | Technical | Implemented |
| LLR3 | Block with both player choices | Technical | Implemented |

**Table 1.5**

# 

# **1.6 SWOT Analysis**



**Figure 1.6**

# 

# **1.7 4W's-1H**

## **WHO -**

This system is just for entertainment.

## **WHAT -**

This is a simple game with 3 x 3 grid and can be played using ‘X’ and ‘O’.

## **WHEN -**

This is simple game and can be played by any two players whenever they are free.

## **WHERE -**

This is implemented in mobile applications.

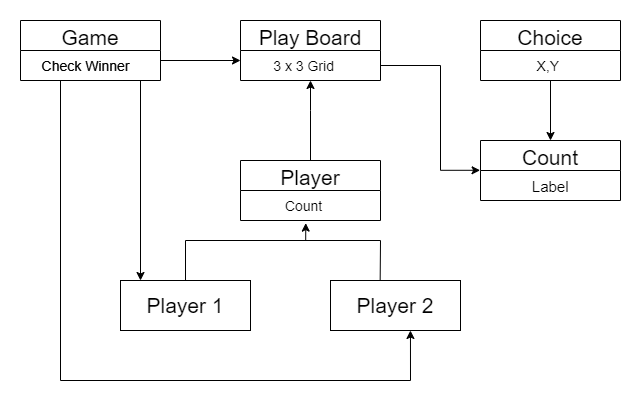
## **HOW -**

It is free and can be used by everyone as its simple.

**2. Design**

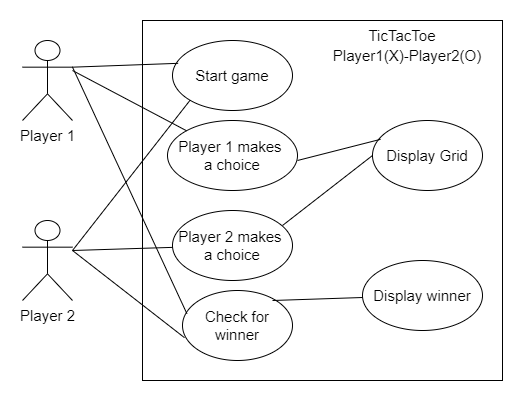
**2.1 Structural diagram**

**2.1.1 Class diagram**

 **Figure 2.1.1**

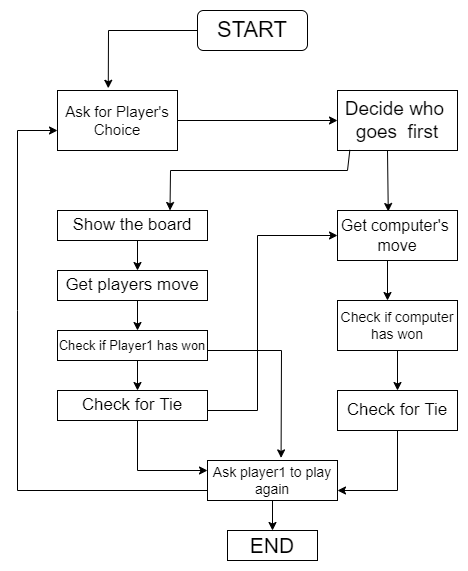
**2.2 Behavioral diagram**

**2.2.2 Use case diagram**



**Figure 2.2.2**

**2.3 Flowchart**



**Figure 2.3**

**3. IMPLEMENTATION**

This design is implemented in visual studio code using WSL Linux commands.

1. Displaying the 3 x 3 grid with play.

2. Ask the player to enter the choice.

3. Displaying the player input in grid and ask for another player’s input.

4. Finally if the player is succeeded in placing three of his marks in a horizontal, vertical, or diagonal row he is the winner.

# 

# **4. Test plan and Output**

## **4.1 High level Test plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Expected i/p** | **Expected o/p** | **Actual o/p** | **Type of test** |
| H\_01 | Display player choice | Program execution | Formatted display with blocks | Formatted display with blocks | Requirement |
| H\_02 | Taking input from player | User choice | Display user choice with block | Display user choice with block | Requirement |
| H\_03 | Check if player has won | Program execution | Display if player wins with the block | Display if player wins with the block | Requirement |
| H\_04 | Check for tie condition | Program execution | Display game is tie and restart the game | Display game is tie and restart the game | Requirement |

## **Table 4.1**

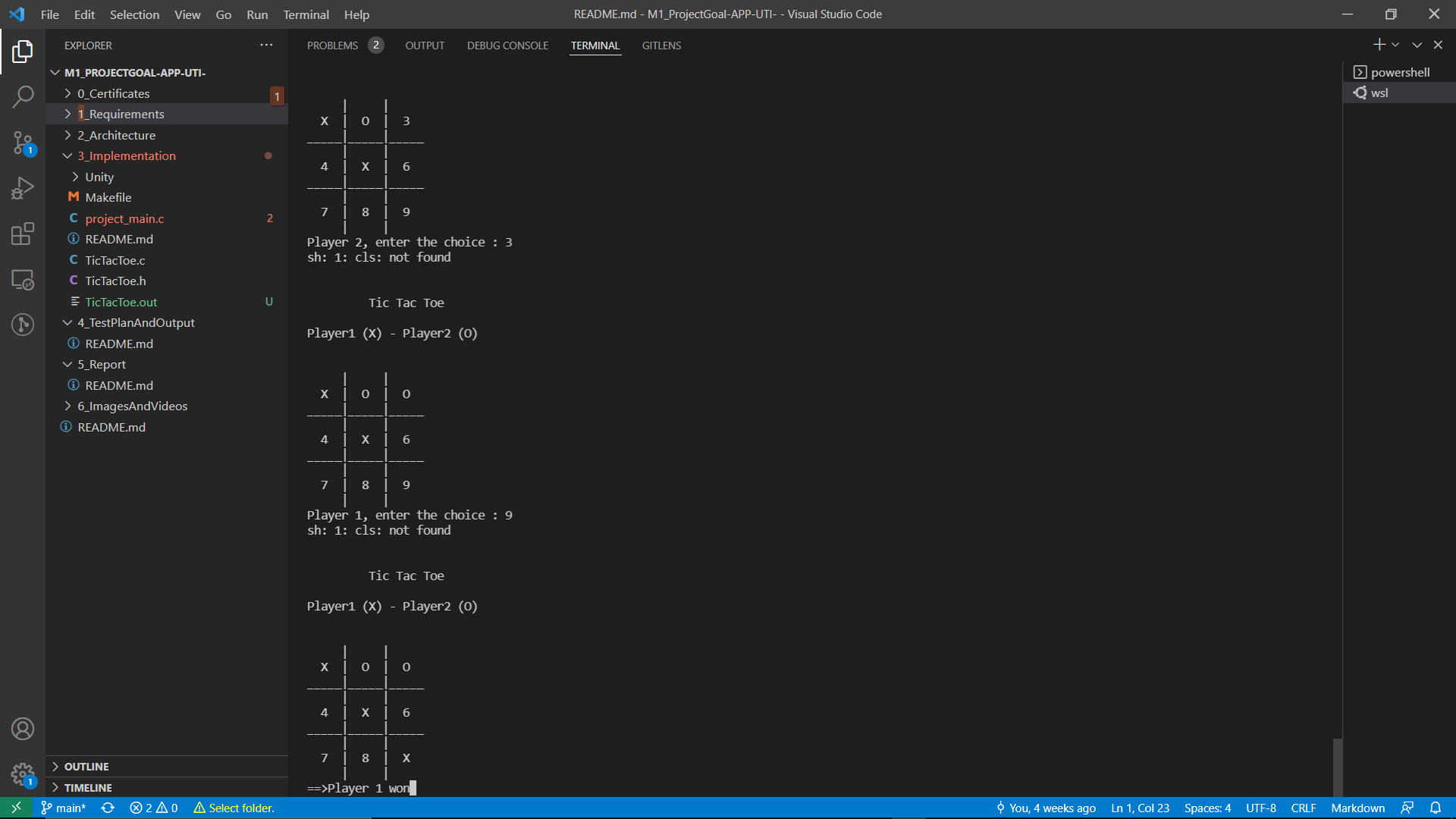
## 

## **4.2 Low level Test plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Expected i/p** | **Expected o/p** | **Actual o/p** | **Type of test** |
| L\_01 | Displaying the options for the player to select | User choice | Based on the choice functions are called | Based on the choice functions are called | Technical |
| L\_02 | When wrong input is entered | User choice | Another chance is given for user to enter correct option | Another chance is given for user to enter correct option | Technical |
| L\_03 | Displaying both player choices with the block | Details | Block is displayed with all the choices of both players | Block is displayed with all the choices of both players | Technical |
| L\_04 | After every move by player the block is displayed with next to enter | Details | Block is displayed with the current status of game | Block is displayed with the current status of game | Technical |

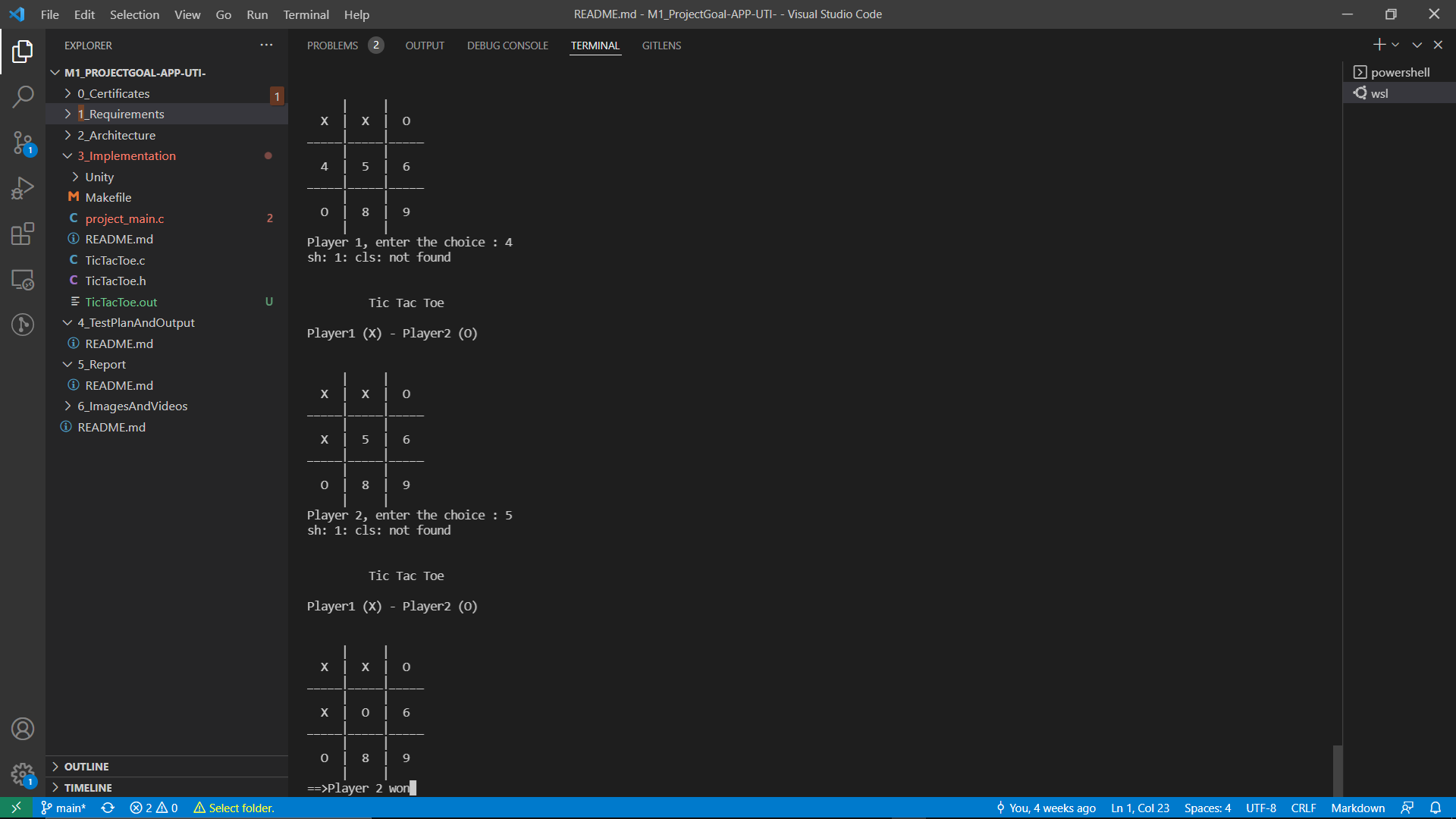
**Table 4.2**

**4.3 Output 1:**



**Figure 4.3**

**4.3.1 Output 2:**



**Figure 4.3.1**