```cpp
#include<iostream>
#include "GameBoard.h"
using namespace std;
void GameBoard::updateGameStatus()
{
	for (int i = 0; i < 3; i++)
	{
		for (int j = 0; j < 3; j++)		//checking 3D WIN status
		{
			if ((data[0][i][j] < '1') || (data[0][i][j] > '9'))
			{
				if (data[0][i][j] == data[1][i][j] && data[1][i][j] == data[2][i][j])
				{
					gameStatus = WIN;
					return;
				}
			}
		}
	}

	int board = 0;
	while (board < 3)			//checking every board WIN status
	{
		if (checkBoardWinStatus(board))
		{
			gameStatus = WIN;
			return;
		}
		board++;
	}
	if (validMovesCount == 27)
		gameStatus = DRAW;
}

GameBoard::GameBoard()
{
	char ch = '1';
	for (int i = 0; i <= 2; i++)
	{
		for (int j = 0; j <= 2; j++)
		{
			for (int k = 0; k <= 2; k++, ch++)
			{
				data[i][j][k] = ch;
			}
		}
		ch = '1';
	}
}

void GameBoard::displayBoard()
{
	cout << endl;
		for (int j = 0; j <= 2; j++)
		{
			for (int k = 0; k <= 2; k++)
			{
				cout << " ";
				for (int i = 0; i <= 2; i++)
				{
					cout << data[k][j][i];
					cout << "   ";
				}
				cout << "    ";
			}
			cout << endl;
		}
		cout << "-----------      -----------      -----------\n     1\t              2\t
3";
```

```
                cout << "\n\n";
}

void GameBoard::markBoard(int pos, char playerSymbol)
{
        int i = (pos / 10) - 1, j = (pos % 10) - 1;
        data[i][j / 3] [j % 3] = playerSymbol;
        validMovesCount++;
        updateGameStatus();                //updating gameStatus
}

bool GameBoard::isValidPosition(int pos)
{
        if ((pos / 10) < 1 || (pos / 10) > 3)
                return false;
        if ((pos % 10) >= 1 && (pos % 10) <= 9)
        {
                return true;
        }
        return false;
}

bool GameBoard::checkBoardWinStatus(int board)     //to check every board WIN status
{
        bool status = false;
        if ((data[board][0][0] == data[board][1][1] && data[board][1][1] == data[board][2][2]) ||
(data[board][0][2] == data[board][1][1] && data[board][1][1] == data[board][2][0]) ||
(data[board][1][0] == data[board][1][1] && data[board][1][1] == data[board][1][2]) ||
(data[board][0][1] == data[board][1][1] && data[board][1][1] == data[board][2][2]))
        {
                status = true;
        }
        else if ((data[board][0][0] == data[board][1][0] && data[board][1][0] == data[board][2][0])
|| (data[board][0][0] == data[board][0][1] && data[board][0][1] == data[board][0][2]))
        {
                status = true;
        }
        else if ((data[board][2][2] == data[board][1][2] && data[board][1][2] == data[board][0][2])
|| (data[board][2][2] == data[board][2][1] && data[board][2][1] == data[board][2][0]))
        {
                status = true;
        }
        return status;
}

bool GameBoard::isAlreadyMarked(int pos)
{
        int i = (pos / 10) - 1, j = (pos % 10) - 1;
        if (data[i][j / 3][j % 3] >= '1' && data[i][j / 3][j % 3] <= '9')
        {
                return false;
        }
        else
        {
                return true;
        }
}

int GameBoard::getValidMovesCount()
{
        return validMovesCount;
}

GameStatus GameBoard::getGameStatus()
{
        return gameStatus;
}
```

```cpp
#include<iostream>
#include "TicTacToe.h"
using namespace std;
void TicTacToe :: playGame()
{
	char player1Symbol;
	char player2Symbol;
	GameBoard board;
	int pos;
	bool validSymbol;
	board.displayBoard();
	do
	{
		cout << "\nEnter Player 1 Symbol : ";
		cin >> player1Symbol;
		validSymbol = isValidPlayerSymbol(player1Symbol);
		if (!validSymbol)
			cout << "Not a Valid Symbol!\n";
	}
	while (!validSymbol);

	do
	{
		cout << "\nEnter Player 2 Symbol : ";
		cin >> player2Symbol;
		validSymbol = isValidPlayerSymbol(player2Symbol);
		if (!validSymbol || player2Symbol == player1Symbol)
			cout << "Not a Valid Symbol!\n";
	}
	while (!validSymbol || player2Symbol == player1Symbol);

	PlayerTurn currentPlayer = FIRST_PLAYER;
	char currentSymbol = player1Symbol;
	while (board.getGameStatus() == IN_PROGRESS)
	{
		bool status = false;
		board.displayBoard();
		do
		{
			cout << "\nPlayer " << currentPlayer << " turn: Enter Cell #: ";
			cin >> pos;
			if (board.isValidPosition(pos) && !board.isAlreadyMarked(pos))
			{
				board.markBoard(pos, currentSymbol);
				status = true;
			}
		}
		while (!status);

		if (board.getGameStatus() == WIN)
		{
			cout << "\nPlayer " << currentPlayer << " WON !";
		}
		else if (board.getGameStatus() == DRAW)
		{
			cout << "\nGame has DRAWN";
		}
		else
		{
			currentSymbol = (currentPlayer == FIRST_PLAYER ? player2Symbol :
player1Symbol);
			currentPlayer = (currentPlayer == FIRST_PLAYER ? SECOND_PLAYER :
FIRST_PLAYER);
		}
	}
	board.displayBoard();
}
bool TicTacToe::isValidPlayerSymbol(char symbol)
{
```

```
        return !(symbol >= '1' && symbol <= '9');
}
```