ASSIGNMENT 1

Online Tic-Tac-Toe

Due date : Sunday, 18 March 2018 @11:59pm

Type : Individual

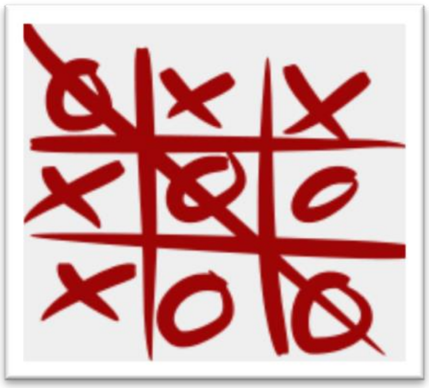
Worth : 5%

IN A NUTSHELL

Develop an app that acts as a tic-tac-toe game such that two players (clients) can play from their browsers (i.e. the client is a webpage) and interact in realtime.

INTRODUCTION

Tic-tac-toe (also known as **noughts and crosses** or **Xs and Os**) is a paper-and-pencil game for two players, *X* and *O*, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.[1]



Game Rules

The object of Tic Tac Toe is to get three in a row. You play on a three by three game board. The first player is known as X and the second is O. Players alternate placing Xs and Os on the game board until either oppent has three in a row or all nine squares are filled. X always goes first, and in the event that no one has three in a row, the stalemate is called a cat game.[2]

To practice this game, you can visit:

- https://playtictactoe.org/

[1] https://en.wikipedia.org/wiki/Tic-tac-toe

[2] http://www.cyberoculus.com/tic-tac-toe.asp?Action=Rules

- Or just Google “tic tac toe”

WHAT TO SUBMIT?

In one ZIP file, you have to submit:

- Game source code
- A two-page report

SYSTEM ARCHITECTURE AND SPECIFICATIONS

The app consists of a server that orchestrates the players‘(clients) requests and updates the status of each player in real time.

Basically, the server has to do the following (but not limited to):

- Provides the players webpage. This means, when a player types the server’s URL (http://localhost:8080 for instance) in its browser, the server returns back the player’s page.
- Manages the game matrix (3x3).
- Accepts Players’ moves.
- Updates the game matrix when a new move arrives.
- Switches players. After each move, the server informs the other player: “it is your turn”.
- Updates players board. After each move received by the server, the server has to broadcast it to all players. In other words, the game boards in both players have to be synced on real time.
- Determines the winner. After each move, the server has to check if there is a winner and it has to inform both players.
- Notify the player if a mistake happens, such as “it is not your turn”, “select an empty square”, etc.
- Any player might reset the game at any moment by sending a request to the server. The server has to reset the game of both players.
- The server has to reset the game if a player leaves (closes its tab).

While the client:

- Accepts messages from the server and print them out

- Accepts commands from the server and execute them, such as: update the game board, rest game board
- Send the current selection
- Send a rest game command to the server

Note: the players page should be simple and clear, which basically has to contain:

  - The gameboard
  - A reset button
  - An output area to show the server's messages

NOTE: For simplicity and without loss of generality, you can assume that the server is able to manage only one session at a time. This means, there is no need to handle more than two players at a time unless you want to get some bounces (see the next section).

Note: Feel free to use any UI library or framework such as Angular, MDL or bootstrap.

HINT: to allow the server to communicate with the clients (players) in realtime and bi-directional, we have to use socket.io library. Click HERE to get more details.

Bonuses

You will get bonus marks if you implement any of these.

  - The support of multiple sessions. This means more than one game can run simultaneously. For example, two sets of players {X1,O1} and {X2,O2} play concurrently their games {game1} and {game2} respectively.
  - The server has a waiting list for players. For example, if a third and forth players connect to the server, the server would put them in the waiting list and activate them once the current game finishes.

RESOURCES AND HELP

- Node.js Application:
  - Lab Week 2
- Socket.io for Node.js projects
  - Lab Week 2
  - https://socket.io/get-started/chat/

If you do prefer to use python:

- Python for Web application
  - https://www.djangoproject.com

    - http://dfpp.readthedocs.io/en/latest/chapter_01.html
- Socket.io for Python projects
    - https://pypi.python.org/pypi/python-socketio
    - https://github.com/stephenmcd/django-socketio

IDEs

You might use the IDE you are happy with, but I would recommend:

- Visual Studio Code (General purpose IDE)
- WebStorm (Web development)
- PyCharm (Python)

REPORT

You have to submit a report (not more than four pages) that discusses the following:

- background on the tools and technique you have used.
- the design of your app. In other words, discuss the main components of your app. You might draw a chart to depict the relationship between the components.
- special requirements and the steps that are required to run your app.
- features you have added.
- prototype in such a way that it would be easy to extend.

INCREMENTAL DEVELOPMENT

You should develop your program incrementally, getting intermediate parts working and tested before going on to the next.

EFFICIENCY

Correctness and a clean design are more important than fast performance for this assignment; however, you should choose your data structures and algorithms to avoid obvious bottlenecks (for instance, searching through a list one-by-one repeatedly rather than using a dictionary).

SUBMISSION

Moodle

Final version of your project should be in a ZIP format and submitted to Moodle by the due date.

Interview

Your Lab tutor will do a short interview in lab week 4 to measure your knowledge and understanding.

PLAGIARISM AND COPYING

This assignment is an individual assignment. Students should develop their solutions independently. Fill out the electronic plagiarism statement when you submit. No other third party code should be used, either through inclusion as a library or by cut-and-paste.

MARKING CRITERIA

Your program will be marked on the following criteria:

- Functionality, including correctness of output on client side webpage.
- Use of appropriate algorithms and data structures
- Readability, including comments
- Compliance with submission instructions

SPECIAL CONSIDERATION

If a student faces exceptional circumstances (serious illness or injury, family emergency etc) that prevent them completing the assignment, they may apply for special consideration according to the policy and procedure on the Faculty website:

http://www.infotech.monash.edu.au/resources/student/equity/special-consideration.html

LATE SUBMISSIONS

Students must contact the lecturer if they do not submit the assignment by the deadline. A 10% penalty per day late will apply.

ASSIGNMENT FORUM

There will be an assignment forum on the unit Moodle site. This is the preferred venue for assignment clarification-type questions. You should check the assignment forum regularly, as unit teaching staff responses to questions are "official" and can constitute amendments or additions to the assignment spec. Before asking for a clarification, please look in the assignment forum. You may of course also email the lecturer or arrange to see them in person if you prefer.