TORRENS UNIVERSITY AUSTRALIA

ASSESSMENT 3 BRIEF	
Subject Code and Title	NDS203 Networking and Database Systems
Assessment	Integrated Networking Project
Individual/Group	Individual
Length	Complete networking application.
Learning Outcomes	The Subject Learning Outcomes demonstrated by successful completion of the task below include: b) Implement server/client communication techniques. d) Apply database querying and manipulation techniques.
Submission	Due by 11:55pm AEST/AEDT Sunday end of Module 12.
Weighting	30%
Total Marks	100 marks

Assessment Task

This final assignment will help bring your database and networking knowledge together into a working integrated project. The overall goal is to build a client server tic-tac-toe game, with chat features and a database to keep track of client details.

Please refer to the Task Instructions for details on how to complete this task.

Context

This networked Tic Tac Toe game project will test your networking and database management knowledge. The server will maintain a database of users to keep track of their login details and scores in tic tac toe. The server will keep track of each clients state and sync the tic tac toe gameboard across all clients.

Up until this point we have worked directly with the database using a database management system software and running SQL commands, but this time we will need to work with code to communicate with the database locally to run these commands. Many programs, games and websites maintain a database on the server side. Managing a game across a network and keeping a database up to date can be difficult and will allow you to demonstrate a strong understanding of networking and database systems in a production environment.

TORRENS UNIVERSITY AUSTRALIA

Instructions

This assessment extends on the previous chat program. We will be adding new features and include a database on the server side. Finally, we will manage online games of Tic-Tac-Toe between chat participants.

For the database functionality we will use SQLite as it is lightweight and does not need database management software to run.

Step 1

To include SQLite into your project, we will need to use the Nuget Package Manager to install it into your project. Right click your project in the solution explorer and choose "Manage Nuget Packages". Next search for System.Data.SQLite.Core and install it. Add using System.Data.SQLite; to the top of any of your C# files to see if it worked correctly.

Step 2

If the user decides to host a server, open a database connection and create a Users table if it does not exist. It should have these columns.

ID	Username	Password	Wins	Losses	Draws

Step 3

You need to keep track of what state a client is in, as there are more states needed now. The server should keep track of each clients state. The client keeps track of its own state and changes it based on messages received from the server. The states a client can be:

Login
Chatting
Playing

When a client connects to the server they are in the login state. During that state, the client can either register a new username and password or login with existing credentials. If either work, then the client is progressed to chatting phase.

In the chat state, if the user types !join and there is a free spot in the Tic-Tac-Toe game, then the server will send a message back to the user letting them know they are player 1 or 2 e.g !player1 or !player2. This will progress the clients state to playing and they should store locally what player they are.

Step 4

The server will inform a player when it is their turn to make a move. When it is the clients turn, they will be allowed to use the Tic-Tac-Toe board. When they make a move, the move information needs to be sent to the server. The server updates its own board and broadcasts the boards state to all clients.

Step 5

When the game of Tic-Tac-Toe is over(win/lose/draw) then the server should update the players scores in the database, inform the players of the results and server lets players know to return to the chatting state.

Step 6

A new chat command !scores should be added that outputs all scores in the database to the asking client. Sort the scores from highest to lowest.

Referencing

It is essential that you use appropriate APA style for citing and referencing research. Please see more information on referencing here https://library.torrens.edu.au/academicskills/apa/tool

Submission Instructions

Zip all projects, related files and any instructional readme.txt files for submission. Name this zip file using this format: NDS203 [your name]_[studentID]_Assignment3.zip

Submit this task via the **Assessment** link in the main navigation menu in NDS203 Networking and Database Systems. The Learning Facilitator will provide feedback via the Grade Centre in the LMS portal. Feedback can be viewed in My Grades.

Before you submit your assessment, please ensure you have read and understand the conditions outlined in the Academic Integrity Code Handbook. If you are unsure about anything in the Handbook, please reach out to your Learning Facilitator.

Academic Integrity Declaration

I declare that except where I have referenced, the work I am submitting for this assessment task is my own work. I have read and am aware of Torrens University Australia Academic Integrity Policy and Procedure viewable online at http://www.torrens.edu.au/policies-and-forms

I am aware that I need to keep a copy of all submitted material and their drafts, and I will do so accordingly.

TORRENS UNIVERSITY AUSTRALIA

Assessment Rubric

Assessment Attributes	Fail (Yet to achieve minimum standard) 0-49%	Pass (Functional) 50-64%	Credit (Proficient) 65-74%	Distinction (Advanced) 75-84%	High Distinction (Exceptional) 85-100%
User Database *Demonstrates understanding of creating databases and tables, and can insert, update and select data as required by the task.* **Percentage for this criterion = 30%**	Implements little to none of the database requirements or only demonstrates partially developed understanding of creating and maintaining a database	Database and tables created based on assignment structure Users can register their details and data inserted into the database Login feature not implemented User scores not maintained in the database No user command implemented to view scores of all players	Database and tables created based on assignment structure Users can register their details and data inserted into the database Users can login with a username and password which is compared against values in the database User scores not maintained in the database No user command implemented to view scores of all players	Database and tables created based on assignment structure Users can register their details and data inserted into the database Users can login with a username and password which is compared against values in the database User's scores are updated in the database correctly. No user command implemented to view scores of all players	Database and tables created based on assignment structure Users can register their details and data inserted into the database Users can login with a username and password which is compared against values in the database User's scores are updated in the database correctly. Clients can run command a command to see the scores of all the clients in highest to lowest order.

Manage Client States Manage the clients' states on the server as clients move through different phases, keeping the clients and server informed with relevant packets **Percentage for this criterion = 30%**	Implements little to none of the client state management requirements or only demonstrates partially developed understanding of managing the changing states of clients on a server	Server keeps track what state each client is in as they attempt to join, chat, and play tic tac toe. Server does not inform clients know which state they are in Server does not keep track of players for the game and not inform clients which players they are New games cannot be initiated successfully	Server keeps track what state each client is in as they attempt to join, chat, and play tic tac toe. Server informs client which state they are in so the client can understand its own state and its options available Server does not keep track of players for the game and does not inform clients which players they are New games cannot be initiated cleanly	Server keeps track what state each client is in as they attempt to join, chat, and play tic tac toe. Server informs client which state they are in so the client can understand its own state and its options available Client is informed of what player they are when joining a game of tic tac toe and server keeps track of which client is which player New games cannot be initiated cleanly	Server keeps track what state each client is in as they attempt to join, chat, and play tic tac toe. Server informs client which state they are in so the client can understand its own state and its options available Client is informed of what player they are when joining a game of tic tac toe and server keeps track of which client is which player Games can finish and different clients can initiate another game of tic tac toe without issue.

Networked Gameplay Manage client and game states across the network **Percentage for this criterion = 30%**	Implements little to none of the game state management requirements or only demonstrates partially developed understanding of managing client states and syncing game states across a network	Users who are not the current players, cannot touch the game board. Players can interact with the board on their turn with the correct tile piece. Players unable to send their game turn information over to the server Server does not update the gameboard across all clients Server does not inform clients of game outcomes	Users who are not the current players, cannot touch the game board. Players can interact with the board on their turn with the correct tile piece. Players send their turn data to the server and the server plays the move on its end. Server does not update the gameboard across all clients Server does not inform clients of game outcomes	Users who are not the current players, cannot touch the game board. Players can interact with the board on their turn with the correct tile piece. Players send their turn data to the server and the server plays the move on its end. The server syncs the gameboard with all clients when players take turns. Server does not inform clients of game outcomes	Users who are not the current players, cannot touch the game board. Players can interact with the board on their turn with the correct tile piece. Players send their turn data to the server and the server plays the move on its end. The server syncs the gameboard with all clients when players take turns. Server informs all clients of game outcome (who is the winner or if it's a draw) and resets the board

Code Quality *Understands the underlying project structure and code, extends it and maintains a clean project* **Percentage for this criterion = 10%**	Project is maintained poorly, has bugs or issues or is unchanged from assessment 2.	Project exhibits 2 of the below qualities: Project runs without issues, bugs or crashes Project broken into functions and classes cleanly Good naming conventions are used for files, classes, functions and variables. Project's indentations and whitespaces are consistent and practical. Code is well commented to explain each code segments intention.	Project exhibits 3 of the below qualities: Project runs without issues, bugs or crashes Project broken into functions and classes cleanly Good naming conventions are used for files, classes, functions and variables. Project's indentations and whitespaces are consistent and practical. Code is well commented to explain each code segments intention.	Project exhibits 4 of the below qualities: Project runs without issues, bugs or crashes Project broken into functions and classes cleanly Good naming conventions are used for files, classes, functions and variables. Project's indentations and whitespaces are consistent and practical. Code is well commented to explain each code segments intention.	Project exhibits all of the below qualities: Project runs without issues, bugs or crashes Project broken into functions and classes cleanly Good naming conventions are used for files, classes, functions and variables. Project's indentations and whitespaces are consistent and practical. Code is well commented to explain each code segments intention.

The following Subject Learning Outcomes are addressed in this assessment	
SLO b)	**Implement server/client communication techniques.**
SLO d)	**Apply database querying and manipulation techniques.**