Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

BACHELOR OF SCIENCE – YEAR IV
KCCGD_B_Y4

Fundamentals of Game Networking

Project

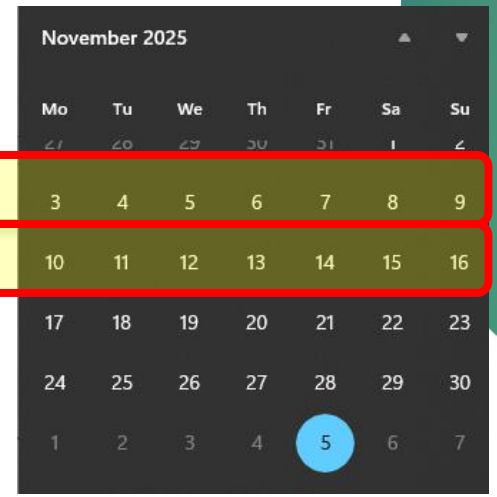
Martin Tolan
Sem I - 2025/2026

setu.ie
INSPIRING FUTURES

Project Outline

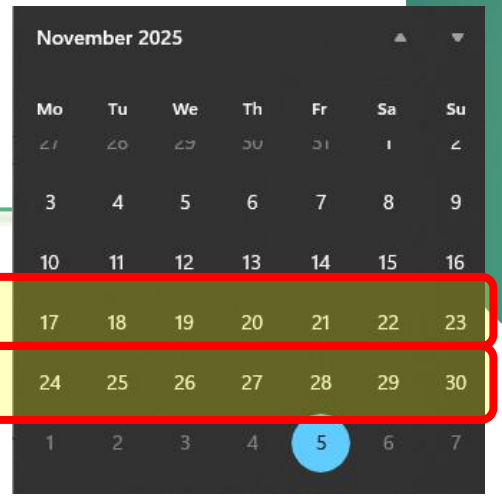
- Project is worth **30%** of final marks
- Choose a game that is a standalone console-based game and migrate to a network based multiplayer game.
- Choose a game that has been developed by yourself over the course of your studies or choose one of the stock games:
  - Battleship
  - Connect four
  - Pong
  - Tic tac toe
  - Snake
- Most focus is on the migration of the game into a multiplayer networked game. Graphics and game is secondary, but responsiveness and connectivity is important.

Project Timeline



- **Week 0 - 03 November 2025**:
  - Project Start
  - Initial Project description, requirements including Rubric shared.
  - Game choice: Everyone chooses either a project or their own or one from stock
  - Gitlab configured and available
- **Week 1 - 10 November 2025**:
  - Initial project baseline. Each student checks their game project into Git
  - Git project to include Markdown describing the project, the game chosen and a short description of how the baseline game works.
  - Include a workflow/flow diagram of how the game application operates.
  - Also, document must include instruction on how to build the game using visual studio from just a git clone/pull from the git repository.

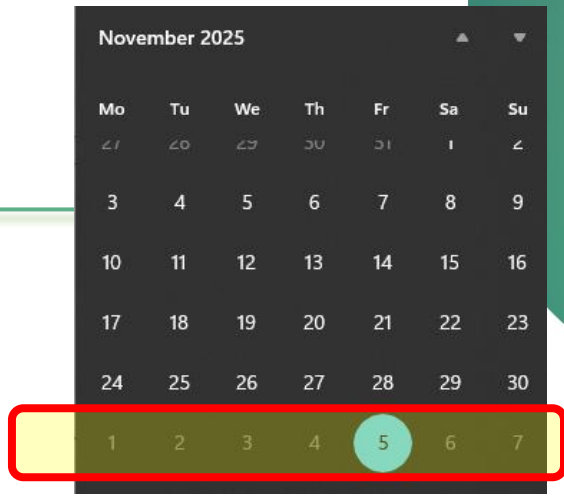
Project Timeline



- **Week 2 - 17 November 2025**:
  - Game demo < 5 minutes
  - Based on the baseline checked into git
  - Short demo of the game during the lab slot. The baseline document should support the demo.
  - Validates that the baseline is good and game is fit for migrating to a network-based model

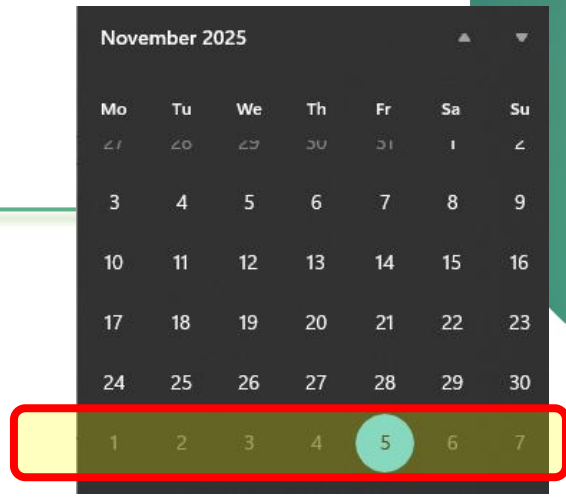
- **Week 3 - 24 November 2025**:
  - Continue project implementation.
  - No formal marking this week

Project Timeline



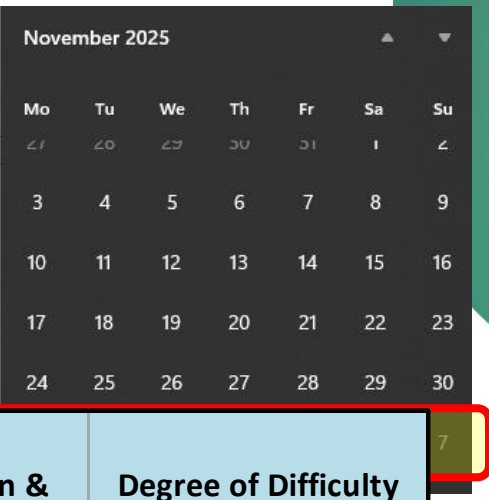
- **Week 4 - 01 December 2025:**
  - Project complete!
  - All code pushed to git repo.
  - Code freeze on **Monday December 1st @ 11:55pm** with all artifacts checked into the git repository. Artifacts to include all diagrams, documents, code elements including the solution, project, sources, packages, etc.
  - Documents include instructions on how to build the application and the migration of the game from standalone to networked.
  - In person live demos during the lab session on Tuesday 2nd December & Friday 5th December (if required).
  - Must be able to build project based on a git clone/pull from the git repository!

Project Demo



- **Week 4 - 01 December 2025:**
  - Demo consists of:
    - Git clone/pull of entire repo.
    - Build project from sources.
    - Project demo, running on multiple computers.
    - Walk through of the migration of the game application (basically how you brought this game from a standalone game to a multiplayer networked game). Use your documents to support the demo: flow charts, workflows, rubric, etc.
    - Marks assigned based on git repo + your knowledge of the migration of the game. You must be able to explain how the game works and what you did to migrate the game into a multiplayer + networked game.

Project Marking Scheme



- **Week 4 - 01 December 2025:**
  - Scoring

Criterion	Socket Integration & Networking Layer	Game Logic Integration (Networked Play)	Protocol Design & Message Handling	Code Structure, Modularity & Error Handling	Documentation & Explanation	Degree of Difficulty (Game Type Factor)
Description	Client-server communication implemented using sockets (TCP/UDP).	Adapting game logic to operate correctly over the network.	Message structure and handling between client and server.	Separation between logic, networking, and UI layers. Handling of invalid inputs, disconnects, or network issues	Code comments, design report, or README.	Adjustment for complexity of the chosen game.
Weight (%)	**25**	**25**	**10**	**10**	**20**	**10**
Poor (<50%)	Non-functional or missing.	Fails to synchronize states.	No defined message format.	Poorly structured, tightly coupled. No error handling.	No documentation.	Trivial or incomplete.
Satisfactory (50–69%)	Basic communication only.	Partially working; some inconsistencies.	Hard-coded or unstructured messages.	Limited structure but readable. Minimal validation;	Minimal comments or notes.	Low (Turn-based): Battleship, Connect Four (0%)
Good (70–89%)	Working but minor sync issues.	Mostly synchronized; minor desyncs.	Simple, functional protocol.	Mostly modular; minor mixing. Some handling;	Some documentation of design.	Medium (+5%)
Excellent (90–100%)	Fully functional, stable connection with error handling.	Seamless play; fully synchronized game states.	Clear, structured protocol; handles errors gracefully.	Clean modular code; reusable components. Graceful recovery from errors	Clear explanation of architecture and protocol.	High (Real-time): Pong, Snake (+10%)

Project Logistics

- Gitlab is up: https://gitlab.netapps-5gmediahub.eu/
- Accounts setup:
  - Username: Martin Tolan => mtolan
  - Default password: **TempPass123**
  - Change your password **ASAP**
- All accounts have a default Group assigned. Create project(s) under this groups.
- All users siloed – no cross contamination
- Sample games up on shared drive: https://nextcloud.waltoninstitute.ie/s/pc5Fy3ALSQ6kAYd

Project Marking Scheme

- **Week 4 - 01 December 2025:**
  - Complete the Rubric for your project:

Criterion	Socket Integration & Networking Layer	Game Logic Integration (Networked Play)	Protocol Design & Message Handling	Code Structure, Modularity & Error Handling	Documentation & Explanation	Degree of Difficulty (Game Type Factor)
Description	Client-server communication implemented using sockets (TCP/UDP).	Adapting game logic to operate correctly over the network.	Message structure and handling between client and server.	Separation between logic, networking, and UI layers. Handling of invalid inputs, disconnects, or network issues	Code comments, design report, or README.	Adjustment for complexity of the chosen game.
Weight (%)	**25**	**25**	**10**	**10**	**20**	**10**
Poor (<50%)	Non-functional or missing.	Fails to synchronize states.	No defined message format.	Poorly structured, tightly coupled. No error handling.	No documentation.	Trivial or incomplete.
Satisfactory (50–69%)	Basic communication only.	Partially working; some inconsistencies.	Hard-coded or unstructured messages.	Limited structure but readable. Minimal validation;	Minimal comments or notes.	Low (Turn-based): Battleship, Connect Four (0%)
Good (70–89%)	Working but minor sync issues.	Mostly synchronized; minor desyncs.	Simple, functional protocol.	Mostly modular; minor mixing. Some handling;	Some documentation of design.	Medium (+5%)
Excellent (90–100%)	Fully functional, stable connection with error handling.	Seamless play; fully synchronized game states.	Clear, structured protocol; handles errors gracefully.	Clean modular code; reusable components. Graceful recovery from errors	Clear explanation of architecture and protocol.	High (Real-time): Pong, Snake (+10%)

Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

BACHELOR OF SCIENCE – YEAR IV
KCCGD_B_Y4

Fundamentals of Game Networking

Project

Martin Tolan
Sem I - 2025/2026

setu.ie
INSPIRING FUTURES