University of Westminster
School of Computer Science & Engineering
6CCGD008W: Games Networking and Security (2025/26)

Module leader	Jack Ingram
Unit	Coursework 1
Weighting:	60%
Qualifying mark	30%
Description	Practical Work 1 (Network Game Phase 1)
Learning Outcomes Covered in this Assignment:	This assignment contributes towards the following Learning Outcomes (LOs): LO1. Gain practical experience of developing a multiplayer networked game through the full development lifecycle. Understand the problem domain using effective analysis, the elicitation of requirements, applying the formal design, and implementing and testing. LO2. Demonstrate an applied understanding of computer networking knowledge specific to games development. LO4 Develop practical computer networking skills to create a game that runs across multiple clients.
Handed Out:	October 2025
Due Date	**Tuesday 18th November 2025 before 1pm (Week 9)**
Expected deliverables	**Submit on Blackboard:** A zip file containing: 1. **Unity project containing source code and assets** 2. **Built executable version of the game** Not zipped up: 1. **Link to the video demo hosted on YouTube (criterion 11)** 2. **The PDF or DOCX short implementation summary (criterion 9)** It is **YOUR RESPONSIBILITY** to check prior to the submission that the demos run as well on the University machines and that the submitted files contain all relevant files and can be run. **Use Unity version 6000.0.58f2.** If you **do not** provide a video demo **and** cannot demonstrate your work at the viva you will receive **zero marks** for the Implementation.
Method of Submission:	Electronic submission on BB via a provided link close to the submission time. The file you upload should have the following naming format: <6CCGD008W_CW1_StudentNumber_FullName.zip> E.g. 6CCGD008W_CW1_w1234567_HungryJason.zip **ZIP, RAR AND 7Z ARE THE ONLY ACCEPTABLE ARCHIVE FORMATS**

Type of Feedback and Due Date:	● Verbal feedback will be provided in the tutorials as the assessment project progresses. ● Verbal feedback for the submitted coursework will be provided directly during the viva/demo of the project if one is required. ● Written feedback and marks 15 working days (3 weeks) after the submission deadline. **All marks will remain provisional until formally agreed by an Assessment Board.**
BCS Criteria	2.1.1 Knowledge and understanding of facts, concepts, principles & theories 2.1.3 Problem solving strategies 2.1.4 Analyse if/how a system meets current and future requirements 2.1.6 Recognise legal, social, ethical & professional issues 2.1.9 Knowledge of information security issues 2.2.3 Recognise risk/safety for safe operation of computing and information systems 3.1.2 Methods, techniques and tools for information modelling, management and security 3.1.3 Knowledge of systems architecture 4.2.1 Specify, deploy, verify and maintain computer-based systems

Assessment regulations

Refer to section 4 of the “How you study” guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website: **http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

Academic Integrity Reminder

Any use of generative AI tools (e.g. ChatGPT, Copilot, etc.) must comply with the University’s Academic Integrity Policy. Submissions must reflect your own work and understanding, and you may be asked to explain any part of your code during the viva.

Coursework Description

The objective of this coursework is to build a simple network ready battle arena game for up-to-four players based on the series of tutorial tasks delivered in this module. You will build the game in Unity and implement the networking aspects of the game via the backend solution Photon. The implementation of the game will follow a prescribed path, but there will be an opportunity to improve certain features for extra marks.

The focus of this coursework is making the game network-ready. At least a two-player networked game must be demonstrable to pass the coursework. Your build may be reviewed in a viva (live coursework demonstration) after submission where you will be asked to explain your understanding of your code.

Warning: You are expected to follow the tutorials supplied with this module. If you follow other tutorials to build a Photon networked game, you may not be able to demonstrate all functionality required by this coursework and will be *limited to up-to 50% of the coursework marks*. Further to this you will struggle to implement the changes required in Coursework 2, which build upon the features of Coursework 1.

Assessment Criteria:

Asterisks denote that these requirements are covered in the tutorial tasks.

1. All mandatory basic features are implemented.*

The game must:

- Take place in an enclosed arena
- Count how many kills each player gets during a round
- Have appropriate UI layout without bugs, scaled to FHD (1920x1080)
- Not have any severe bugs that affect gameplay or connections

The player must:

- Move in 8 directions based on user input
- Fire a projectile forward based on the direction it is travelling based on user input
- Take damage when hit by an opponent projectile
- Die and respawn when health is depleted to 0 and the game is still in play **[15 Marks]**

2. A lobby can be created and joined by up to 4 players (but not less than 2). All other lobby buttons and functions should work correctly.* **[15 Marks]**

3. A game can be successfully played and completed by all players.* **[10 Marks]**

4. The winner is detected, and their name is announced to all clients.* **[5 Marks]**

5. A custom message is displayed to each player depending on whether they won or lost. **[7 Marks]**

6. Winner is calculated based on both kill tally *and* time. **[7 Marks]**

7. Game will give the players the option to play another game on finish. At least 2 players must still be connected. You are required to implement this logic yourself. **[10 Marks]**

8. Customise the game to make it more engaging. It is up to you how you do this. You could improve the menu, the multiplayer game world, the assets, the weapons, the UI etc. You could even add a new feature that you have created yourself. Marks will be awarded for appropriateness (does it fit with the game?), quality, creativity and effort. **[10 Marks]**

9. A short, 1-page (or more if required) implementation summary documenting your customisations and how you implemented these. Include screenshots of your game or code excerpts to illustrate. **[6 Marks]**

10. Submission instructions have been correctly followed, and the Unity project reduced in size by removing unnecessary files. The submission instructions will be sent in an announcement on Blackboard and an email when the submission link is made available. **[5 Marks]**

11. A demonstration Video (YouTube Link). A 5–7-minute video demonstrating the networked features of your game and explaining how your code works (including key methods or functions). A spoken voice-over is required. The video should show both gameplay and relevant parts of your code editor. **[10 Marks]**

Coursework Marking Scheme

The Coursework will be marked based on the following marking criteria:

Criteria	Mark per component
Unity-based Implementation	**79 marks (subtotal)**
1) All mandatory basic features are implemented. (See above for list).	**15**
2) A lobby can be created and joined by up to 4 players (but not less than 2). All other lobby buttons and functions should work correctly.	**15**
3) A game can be successfully played and completed by all players.	**10**
4) The winner is detected, and their name is announced to all clients.	**5**
5) A custom message is displayed to each player depending on whether they won or lost.	**7**
6) Winner is calculated based on both kill tally and time.	**7**
7) Game will give the players the option to play another game on finish. At least 2 players must still be connected.	**10**
8) Customise the game to make it more engaging. Marks will be awarded for appropriateness (does it fit with the game?), quality, creativity and effort.	**10**
Additional Requirements	**21 marks (subtotal)**
9) A short, 1-page (or more if required) implementation summary documenting your customisations and how you implemented these. Include screenshots of your game or code excerpts to illustrate.	**6**
10) Submission instructions have been correctly followed, and the Unity project reduced in size by removing unnecessary files.	**5**
11) A demonstration Video (YouTube Link). A 5–7-minute video demonstrating the networked features of your game and explaining how your code works (including key methods or functions). A spoken voice-over is required.	**10**
Total	**100**

Grade Descriptors:

For every section the following grade descriptor will be followed.
For example, on a maximum number of 5 marks:
Excellent = 5, Good = 4, Average = 3, Basic = 2, Insufficient = 0 or 1

Excellent: *demonstrates an outstanding understanding of multiplayer game development and networking concepts. The implementation is robust, innovative, and polished, showing clear creativity and technical skill. The game runs smoothly with well-structured, readable code and a high level of visual and gameplay quality. The documentation or video explanation is professional, clear, and demonstrates deep understanding of the code and design decisions.*

Good: *demonstrates a strong understanding of networking and gameplay implementation with most features correctly working. Code is logically structured and mostly efficient. The game is engaging and visually cohesive with only minor issues. Explanations and documentation show good insight and clear understanding of design choices, though some areas could be developed further.*

Average: *demonstrates a reasonable understanding of the task and implements most required features. Some bugs or inconsistencies may remain, but core networking and gameplay functions work as intended. Customisation or presentation may be limited. Explanations show a general understanding but lack depth, critical reflection, or detail about technical decisions.*

Basic: *shows a limited understanding of game networking and implementation. Only partial functionality is achieved, with several missing or unstable features. Code may be disorganised or lack clarity. Explanations and documentation are minimal, unclear, or show little evidence of understanding the underlying systems.*

Insufficient: *fails to demonstrate a functional or coherent implementation. Most key features are missing or non-functional. There is little or no evidence of understanding the networking concepts or the programming process. Documentation or explanation is incomplete, unclear, or absent.*

End of Document