Assignment 4 — Board Game (AI + Networking)

1) Overview & Goals

Build a turn-based or real-time board-style game with clear, non-trivial rules that reward strategy. Your game must include (1) a computer-controlled opponent (AI) with multiple difficulty levels and a correct implementation of either Minimax, Alpha–Beta pruning, or Monte Carlo Tree Search (MCTS), (2) online multiplayer (player-vs-player over the network), and (3) a functional menu with a working Continue/Resume capability. Teams may choose any game engine; a 3D-capable engine is recommended to prepare for the next assignment.

2) Learning Outcomes

- Model a board game's state, rules, and legal actions in a way suitable for search-based AI.
- Implement and evaluate adversarial search (Minimax / Alpha–Beta / MCTS) and expose difficulty levels.
- Design simple, reliable networking for online PvP, including session join/leave and message handling.
- Create a robust game loop and UI with save/resume semantics (Continue).
- Package, document, and test a small game project to a reproducible standard.

3) Scope (Functional Requirements)

3.1 Game Concept & Rules

Choose a board game (not trivial) with clear rules and meaningful strategy (examples: Connect 4+, Othello/Reversi, Checkers-like, Hex, Gomoku, Tablut, small TBS variants). Document the rules in README.md: objective, win/lose/draw conditions, turn order, legal moves, scoring (if any), and edge cases.

3.2 Artificial Intelligence (5 pts)

1 Playable vs NonPlayer (≥ random baseline) — 1 pt.

2 Correct implementation of Minimax or Alpha–Beta or MCTS — 3 pts.

3 Expose ≥ 2 difficulty levels that materially change AI strength — 1 pt.

AI Acceptance Criteria

- Random baseline: AI selects legal moves uniformly at random, never crashes, and respects time/turn limits.
- Minimax / Alpha–Beta: Correct terminal evaluation and utility propagation; no illegal nodes are expanded; alpha–beta actually prunes where expected.
- MCTS: UCT/PUCT variant is documented; rollouts or evaluations are legal and terminate; exploration constant is justified.

- Difficulty: Provide at least two modes (e.g., search depth/time budget/rollouts) with a visible effect on play strength.

3.3 Networking (4 pts)

1. Support online PvP (human vs human) over the network — 4 pts total.
2. Include a session model (host/join or matchmaking) and a minimal lobby or connect screen.
3. Synchronize game state with deterministic, authoritative rules; reject illegal moves.
4. Handle disconnects and reconnection or graceful termination.

Networking Acceptance Criteria

- Two players can connect from different machines; each sees the same board and turn order.
- Latency ≤ ~200 ms round-trip yields a playable experience (turn-based games are tolerant).
- Security basics: ignore malformed messages; validate moves server-side; no obvious code-injection surfaces.
- Clear errors for timeouts/disconnects; no permanent softlocks.

3.4 Menu & Game States (1 pt)

Provide a functional menu with **Continue** that resumes an in-progress game session (local or networked if feasible). Recommended states: Splash (optional) → Main Menu → Gameplay → Pause → End (win/lose/draw) → Back to Menu. Continue must restore the last saved session or paused state reliably.

4) Bonus (Optional)

- Use Machine Learning (e.g., supervised policy/value from self-play or dataset; integrate with search).
- Multiple NonPlayer opponents (e.g., 3-player variants) with working turn management and fair rules.

5) Deliverables

- Source code and assets.
- README.md describing rules, controls, how to run (local + network), AI method (with parameters), and known issues.
- Instructions to host/join an online match (ports, NAT, or relay service).

6) Grading Rubric (10 points + bonus)

Category	Details	Points
AI (NonPlayer + Search + Difficulty)	Random baseline (1), Minimax/Alpha–Beta/MCTS correct (3), ≥2 difficulty levels (1	5
Networking (Online PvP)	Connect & play with another human over network; lobby/connect screen; validation	4
Menu + Continue	Menu includes working Continue/Resume	1

Bonus: Machine Learning integration; multiple NonPlayer opponents (3-player variants).