Our final system aligns closely with the goals and requirements outlined in the original SAR. The intended functionality providing real-time multiplayer interaction, handling matchmaking, presence tracking, and enabling a unified chat system, the only out of original scope was adding multiple games vs just “connect 4” but other than that we went through a lot of architecture changes that separated concerns between backend game logic, frontend UI, and shared communication layers. While some features evolved from the initial vision, especially around presence handling and chat integration, the core expectations of responsiveness, modularity, and extensibility were achieved.

Across the project, teamwork and communication were very important to keeping development on track. We learned how important it is to establish shared ideas and intentions early, especially for file structure, event naming, and connection handling. Miscommunications often led to duplicated work or mismatched assumptions between frontend and backend, but resolving these issues helped the team improve coordination and become more intentional about discussing changes before implementing them. Our project management practices also matured over time, shifting from informal planning to more structured sprint goals and clearer division of tasks.

Architecture, documentation, and testing practices all evolved and came to be as the system grew more complex. Early sprints focused mostly on functionality and less on things like security,testing, or documentation, but later stages forced us to document workflows, add diagrams, define module responsibilities, and create repeatable test cases. This made the system easier to debug and ensured that new features didn’t break existing ones.

AI played a surprisingly large role in shaping decisions. It helped generate architectural ideas, debug tricky synchronization issues, produce diagrams, and refine code readability. Most importantly, it accelerated experimentation by providing quick alternatives when we were stuck.

Individually, I contributed heavily to integrating chat, refining game logic, restructuring backend modules, and improving documentation. Through the project, I developed stronger debugging skills, a deeper understanding of real-time systems, and more confidence in coordinating multi-module architectures.