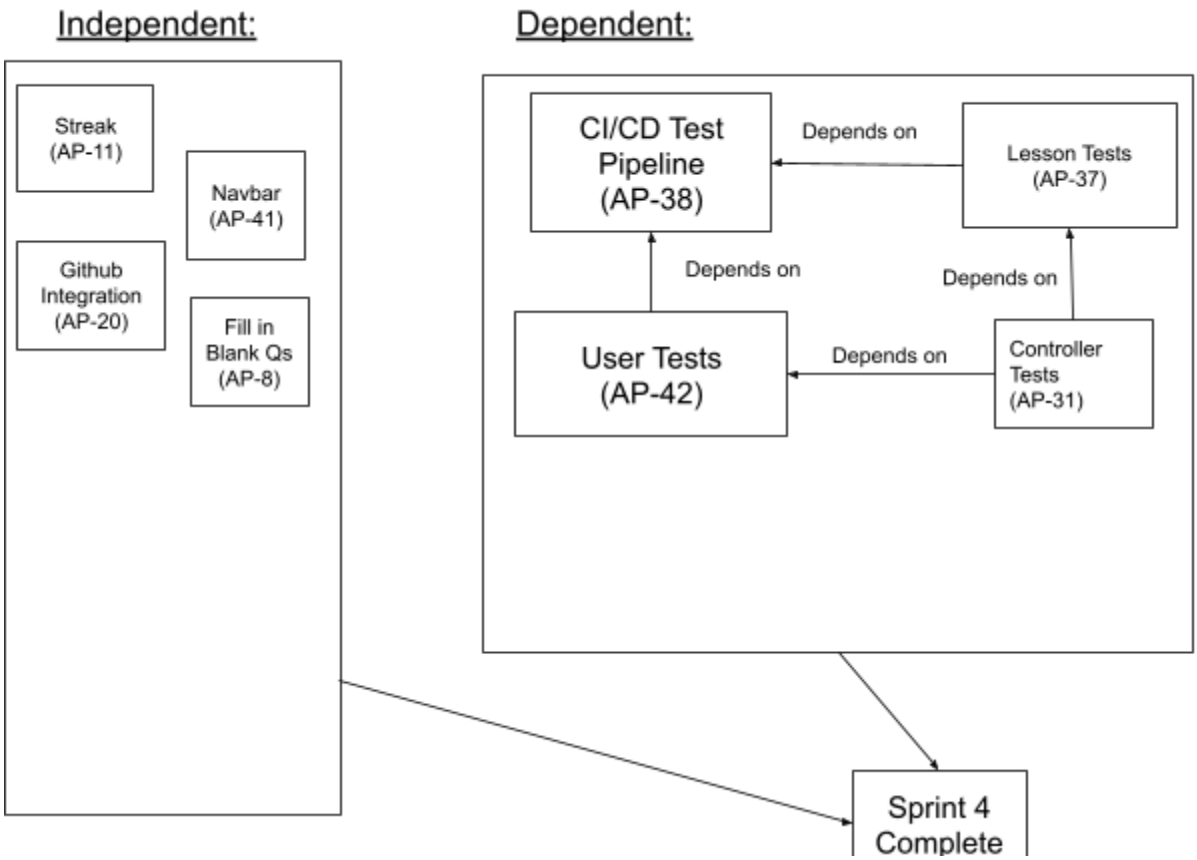

During Sprint 4, we continued to build on our earlier planning strategies by identifying key task dependencies and mapping them with a network diagram. This allowed us to determine the sprint's critical path and structure our workflow to avoid bottlenecks. Tasks that relied on backend schemas, API endpoints, or third-party integrations were prioritized early, ensuring smooth coordination across the team.

Strategies to Keep the Sprint on Track

To maintain momentum and prevent delays caused by dependencies, we adopted several strategies:

- **Daily Standups:** Regular check-ins allowed us to catch blockers early and quickly reassign tasks when needed, keeping the sprint velocity consistent.
- **Pair Programming for Complex Features**: For technically demanding tasks such as multiplayer logic and interactive fill-in-the-blank questions (AP-8), pair programming helped reduce bugs and improved code quality.
- **Early Backend Coordination**: Meetings held at the start of the sprint clarified schema designs and interface contracts, enabling both frontend and backend teams to move in sync.

Successful Completion of AP-8

In contrast to the challenges faced in Sprint 3, AP-8 was successfully completed this sprint. The team effectively scoped and delivered the fill-in-the-blank interactive question component.

Lessons Learned

- **Scoping Accuracy Has Improved:** Our experience from previous sprints helped us make more realistic story point estimates, particularly for interactive features.
- **Workflow Maturity:** The team has developed a strong cadence of async communication, documentation, and proactive problem-solving. This maturity was evident in how we managed technical handoffs and resolved blockers.

Sprint 4 stands out as our most polished iteration—highlighting not just task completion, but process improvement and team growth.