AI Partnership Report

A Commander's Perspective on Leveraging AI in Development

Commander Eyüb YILDIRIM

July 11, 2025

Executive Summary: This report documents my approach to leveraging an AI Battle Companion for the rapid development of the Tower Defense Data Processing System. The project's success was rooted in a clear division of labor: I provided strategic direction, architectural oversight, and critical analysis, while the AI served as a high-speed implementation and knowledge-on-demand tool. This synergistic model proved exceptionally effective.

Contents

1 My Collaborative Philosophy and Model **2**
- 1.1 My Role as Strategic Commander . . . 2
- 1.2 My Utilization of the AI Battle Companion . . . 2

2 Key Milestones in Our Collaboration **3**
- 2.1 Initial Design and Refinement . . . 3
- 2.2 Evolving the Core Gameplay . . . 3
- 2.3 Ensuring System Robustness . . . 3

3 Conclusion: AI as a Force Multiplier **3**

1 My Collaborative Philosophy and Model

From the outset, my strategy was to treat the AI not as a partner to be debated with, but as a highly advanced tool to be wielded. I established a clear **Strategist-Agent** interaction model where my role was to provide the "what" and "why," and the AI's role was to execute the "how." This allowed me to remain focused on the high-level system architecture and user experience, delegating the often time-consuming task of code generation.

Also, it should be stated that almost everything in this task was created using AI, as was requested, including this brief and the summary report. Aside from this and the next paragraph, everything is generated by Gemini 2.5 Pro model as it is available on Google's AI Studio. I've only done some adjustments and corrections where necessary.

Normally, I would modify the language the AI uses heavily, but I wanted to keep it in this instance, since I think it reflected the gamified nature of the original challenge, which is what I loved the most about doing this challenge. I not only utilised AI and my coding skills but also my love of games and filled in the blanks where I thought the objective wasn't clear, at least in my mind, to the best of my abilities. Especially the group deployments was a point I wasn't really sure at the beginning but in the end, I've decided to go with the "Squad" design you saw in the final product.

1.1 My Role as Strategic Commander

- **Architectural Authority:** I made the final decisions on the technology stack (Go, Vue, WebSockets) and the core backend architecture proposed by the AI.
- **Feature Direction:** I drove the project's evolution beyond the initial specification by conceiving and ordering the implementation of advanced mechanics like the "Squad Blueprint" system and the "Hitpoint/Processing Power" combat model.
- **Quality Assurance and Logic Vetting:** I served as the primary and final code reviewer. My analysis was crucial in identifying subtle but critical flaws in AI-generated code, including logical race conditions, off-by-one tick errors in the game loop, and compile-time errors.
- **System Integrator:** I ensured that the independently generated backend and frontend components were compatible and functioned as a cohesive whole.

1.2 My Utilization of the AI Battle Companion

I utilized the AI as a specialist tool to accelerate development in several key areas:

- **Rapid Prototyping:** I tasked the AI with generating complete, functional code for entire services and components, such as the initial WebSocket hub, the Go game engine, and the Vue frontend components. This compressed weeks of boilerplate and setup into minutes.
- **Implementation of Complex Logic:** After designing a new feature, I would provide a high-level description to the AI and task it with the full implementation. For example, I described the "partial deployment" logic for squads, and the AI generated the Go code to achieve it.
- **On-Demand Debugging and Refactoring:** When I identified a bug, I would describe the undesirable behavior (e.g., "the animations are not firing") and task the AI with diagnosing and fixing the underlying cause in the generated code.
- **Documentation and Finalization:** I delegated all final documentation and deployment scripting tasks to the AI, including the generation of 'Dockerfile's, 'docker-compose.yml', a 'Makefile', and this very report in LaTeX.

2 Key Milestones in Our Collaboration

Our partnership was defined by a cycle of my direction, AI implementation, and my critical review.

2.1 Initial Design and Refinement

The AI proposed a solid initial architecture. I immediately identified and corrected a flaw related to Go's unexported fields, establishing our working model of AI generation followed by my expert review and correction. This pattern proved vital throughout the project.

2.2 Evolving the Core Gameplay

My strategic vision shaped the game's depth. The initial concept was a simple "click-to-destroy" model.

1. I first directed the implementation of the **Squad Blueprint** system, which added a layer of strategic planning. The AI successfully implemented this complex state management on both the backend and frontend.
2. I then determined that the combat model lacked depth. I conceived the **Hitpoint and Processing Power** mechanic and tasked the AI with a major refactor of the entire game engine to support it.
3. During this phase, I identified several logical flaws in the AI's implementation of the weapon firing cycle, providing corrections to ensure the "lock-on then fire" mechanic worked as intended without sacrificing throughput.

2.3 Ensuring System Robustness

The AI's speed occasionally led to subtle errors. My role was to catch them.

- I was instrumental in identifying the race condition that prevented animations from firing, providing the necessary insight for the AI to correct its game loop logic.
- I consistently caught compile-time errors in the Go code, acting as the final quality gate before deployment.

3 Conclusion: AI as a Force Multiplier

This project has conclusively demonstrated the power of using AI as a force multiplier under skilled human command. The AI's ability to handle the "heavy lifting" of code generation allowed me to focus my energy on architecture, strategy, and critical analysis—areas where human insight remains irreplaceable.

I did not simply "ask the AI to build a game." I directed it, corrected it, and steered it. I was the architect and the quality control engineer; the AI was the high-speed construction crew. This partnership model is, in my assessment, the future of efficient and sophisticated software development. The final system is a direct result of this successful synergy.

— END OF REPORT —