

Thomas Wenner  
Gauntlet AI Cohort 4  
AI Development Log 1 Page Report

I am on a low budget laptop running windows [IdeaPad Slim 3 \(15" AMD\) | Versatile everyday-use AMD laptop | 82XMX002US | Lenovo USOutlet](#).

I used Claude Code in the browser almost exclusively (99% of code). I may have made a few local edits in Visual Studio Code. I have used Claude Code itself for about 6 months and about 2 months in the browser, so I know some of its behavior.

Workflow:

1. Prompt claude (if something is more than what I would consider a 2-3 point story for a jira sprint, I ask it to plan its work first)
2. Inspect its planning output myself before telling it to start working
3. Tell it to explain what it did and to write tests
4. Commit to github branch, pull down locally, and test it

Effective Prompts:

I do not use any general seeding prompts. I used to use a general prompts until Anthropic removed when claude tells you it is cleaning its context window. So now since I never know exactly when that has happened, I just ask it very often if it understands what it's doing, and if it says something wildly different than what I thought it was going to say, I just update it on what I think it should know about its current workflow state and continue.

Still, example of a prompt:

"You are implementing changes on the dashboard page for a real-time canvas collaboration application. We are about to analyze project state and discuss choices and their tradeoffs for our next feature. (Describe a feature for it to implement) What are your plans for this?"

Code Analysis:

99% AI-generated, but slowly crafted with heavy validation at every commit.

Strengths:

Less mental friction from idea to implementation, although probably the same amount of time to code something as an engineer specialized in the language/frameworks that claude is using on your behalf.

Weaknesses:

Sometimes claude will half lie and say some troublesome tests are now passing but that's only because it removed the code that was causing them to fail. Even Opus needs hand holding for specific things, like "stop using inline styling" or "this page layout is wrong and here is nearly exactly how you fix it"

Insights/key learnings:

-Most times you tell claude code to do something non-trivial, it has a ramp up phase where you sit around and do nothing while it collects stuff and tries to understand what's going on. I think there is room for improvement from my side for giving better instructions and also in the future for claude code and other agents for constructing better understandings of the codebase it is working within.

-I need to learn how to use git to better orchestrate claude code agents

- Initial architecture decisions set the guardrails and entropy for your future changes, so be careful
- ENFORCE DOCUMENTATION FORMAT, GUIDELINES, AND PRACTICES DURING ALL PHASES OF DEVELOPMENT, NEVER LET UP
- unit tests are mandatory, and mocking resources for unit tests is still better than nothing at all
- I need to set up CI/CD pipelines as a hard requirement for any future project
- claude code is an opinionated take on coding agents, and has built in strengths and weaknesses that you cannot modify as far as I know. Custom coding agents for every codebase might be the future, who knows.
- Sometimes claude code will tell you that the color red is not red, and then will ruin your github repository