Lorenzo Naranjo November 11, 2026

How These Notes Are Built

Every chapter and notebook on this site was written with an AI coding agent. I decide what a chapter should argue, the agent drafts text and code, and then I check what it produced: I render the page, read it, run the code, and compare every claim about a paper with the paper itself. This chapter describes that workflow, because it is also the workflow you will use for your project, and because the most important part of it is not the part the AI does.

Why the workflow matters

A year ago, most of a course like this one went into teaching students to write code. That is no longer the bottleneck. An agent such as Claude Code writes a working Python script, a regression, or a figure faster than any of us can type the request. What it does not do reliably is know whether the result is right. It can cite a paper that says something different from what the sentence claims, use a data source that stopped being free last month, or describe output it never ran. It does all of this in the same confident tone it uses when it is right.

So the skill that matters has shifted from producing code and text to directing the work and checking it. A finance professional who can do both is far more productive than one who can do neither, and far safer than one who only does the first.

The pieces

The workflow has four pieces, all of them free or provided by WashU.

- **A repository.** A folder tracked by `git` and kept on GitHub. It keeps every version of every file, with who changed what and when.
- **Quarto.** A publishing system for documents that mix prose, math and code. It turns one `.qmd` source file into a web page, a PDF and a notebook.

- **Python.** It downloads the data, runs the analysis, and draws the figures when the page renders.
- **An AI agent.** Claude Code, which comes with Claude Edu. At your request, it reads and edits the files in the repository and runs commands.

The source of every page is plain text. A chapter is a `.qmd` file: prose in Markdown, math in LaTeX, and Python code in chunks that run when the page is rendered. Because the figures are produced by code that runs at every render, a figure on this site is always the output of the code next to it, not a picture someone pasted in.

Instructions the agent follows

The agent reads a file called `CLAUDE.md` at the top of the repository before it starts working. Mine is a list of rules accumulated over months: how to write, which notation to use, and, above all, how to behave when it is unsure. Three of them do most of the work.

> When the cause of a problem is unknown, say so plainly. Report what was measured, label any hypothesis as a guess, and do not present it as the explanation or act on it without asking.
>
> Never call something fixed, ready or verified without showing the command and output that prove it, run on the real files and data, never on synthetic ones.
>
> Check claims about a cited source against its PDF there.

The first rule matters because an agent, like a person, would rather give an answer than admit it has none. The second matters because “done” means nothing until you have seen it run. The third matters because a citation is a promise to the reader that the source says what the sentence says, and that promise is broken more often than you would think.

Your project repository comes with its own `CLAUDE.md`, a shorter version of mine. It tells Claude never to invent a source, to run code before reporting its output, to use real data only, and never to rewrite the repository’s history. Read it before you start: it is a short description of how these notes are produced.

The loop

Most work on this site follows the same loop.

1. **Ask.** Describe what you want in plain words: a section, a figure, a fix. Say what you want, not how to do it.
2. **Let the agent work.** It edits files and runs commands. Read what it says it did.
3. **Render and look.** Render the page and read it as a reader would. Most problems are visible at this step: a figure that is empty, a number that makes no sense, a sentence that says more than the evidence does.
4. **Check.** For every claim that matters, check it yourself: open the paper, rerun the code, look at the data.
5. **Commit.** Save the version with a message that says what changed and what you checked.

The loop is short, and you go around it many times for a single page. The commit history then records how the page was built. That history is part of what you hand in for your project.

What checking found while preparing this course

Preparing the project topic list for this course is a good example of why step 4 cannot be skipped. I asked the agent for eighteen project topics, each with a question, one or two papers to start from, and free data for a figure. The first draft was fluent and plausible. Then every data source was called, every citation was looked up, and every paper was read. Among the problems that turned up:

- **A citation that had moved.** Budish's paper on the economic limits of Bitcoin was cited as a 2018 working paper. It has since been published in the *Quarterly Journal of Economics* under a different title (Budish 2025).
- **A paper that did not say what the topic needed.** One topic asked how Treasury yields react to the tone of Federal Reserve statements, and cited a paper that measures the tone of FOMC transcripts to estimate the Fed's objectives. It does not study how markets react to statements. Two papers that do were found and checked instead.

- **A measure the literature rejects.** A topic on liquidity providers asked students to plot "impermanent loss," a popular measure that the paper cited in the same brief argues against, because its value depends on the arbitrary start date you choose.
- **Data that were no longer free.** Several sources that were once free now require a paid plan, or a crypto wallet just to sign in.
- **A tool that no longer existed.** The first plan for collecting your work relied on GitHub Classroom, which GitHub retired in August 2026. The agent's knowledge was simply out of date.

None of these problems was visible in the draft. Each was found by doing the boring part: opening the source and reading it. That is the habit this course, and your project's grading, rewards most.

What this means for your project

Your project is a page of notes like the ones on this site, written by your group with Claude, from a repository that starts as a copy of the project template. Forty percent of the grade is verification: every claim has a source, the source exists, and it says what your page says it says. The **What we checked** section at the end of your page is where you show that work, claim by claim.

Use the agent for everything it is good at: drafting, coding, formatting, explaining an idea you do not understand yet. Then do what it cannot do for you. Read the sources, run the code, and look at the page before you commit it.

Practice Problems

These problems give you a chance to practice the concepts introduced in this chapter. Try to solve each one on your own before expanding the solution.

Solutions to all problems can be found at https://lorenzonaranjo.com/class-materials/fintech/.

Problem 1 (An Agent Reports a Fix). You ask Claude Code to fix a figure that showed no data. It replies: "Fixed. The figure now shows daily Bitcoin miner revenue since 2016." You have not rendered the page since. Which statement is most accurate?

a. The figure is fixed, because the agent has access to the code and the data.
b. Nothing has been verified yet; render the page, look at the figure, and check one or two values against the source before committing.
c. The figure must still be broken, because AI agents cannot fix figures.
d. The fix is verified once the change is committed, because git records it.

Problem 2 (Checking a Citation). A draft of your group's page contains the sentence: "Stablecoin issuers are, in effect, unregulated banks whose money is vulnerable to runs (Gorton and Zhang 2023)." The entry in `references.bib` gives the title, journal, volume and pages. Select all statements that apply.

a. The citation is correct if the entry in `references.bib` has a title, a journal and page numbers.
b. Someone in the group must open the paper and find where it makes this argument.
c. The **What we checked** section should record where in the paper the claim appears and who checked it.
d. If Claude wrote the sentence, the group is not responsible for whether the paper supports it.

Problem 3 (Reading a Commit History). The history of a group's project repository, a week before the deadline, shows 31 commits by one member, 2 by another and none by the third. Twenty of the 31 commits carry the line `Co-Authored-By: Claude`. Which statement is most accurate?

a. The page was written by Claude, so the group should lose the points for the page.
b. The history shows that one member has done most of the committing and one has done none, which is worth raising within the group now; the Claude co-author lines show that AI was used, which is allowed.
c. The history proves that the third member did no work at all.
d. Commits with a Claude co-author line should be removed from the history before the deadline.

References

Budish, Eric. 2025. “Trust at Scale: The Economic Limits of Cryptocurrencies and Blockchains.” *Quarterly Journal of Economics* 140 (1): 1–62. https://doi.org/10.1093/qje/qjae033.