Build with your brain on

Free sampler: chapter one, plus the tutor mode prompt pack

Thank you for picking this up. This free sampler is the real opening of the book, not a teaser. It is the entire first chapter, which answers the question most people are quietly asking right now, plus the one page prompt pack that teaches you to use AI as a tutor instead of a crutch.

If it speaks to you, the full book takes you all the way from here to building and shipping a real working app with AI, and understanding every piece of it. You can get it here: **[your Gumroad link]**.

Now, the question the whole internet has been answering wrong.

Chapter 1: should you even learn this in 2026?

LEARNING OBJECTIVES

By the end of this chapter you will be able to:

- Explain why AI replaced certain coding tasks but not the people who exercise judgment.
- Tell the difference between memorizing syntax and building durable judgment.
- Decide what to actually aim for as a learner in an AI-shaped job market.
- Name the deeper, non-career reason that being able to build is worth your time.

There is a question sitting in the back of your mind.

The internet has been answering it for you, loudly, and the answer it keeps giving is the discouraging one. So let us drag the question into the light and look at it properly.

If the machine can write the code, why learn any of this at all?

Hold on to that question. By the end of this chapter I am going to give you an answer that almost nobody online will, and it is the answer that puts you on the right side of what is coming. But first, the uncomfortable part, because I promised you honesty and I am not going to open this book with a pep talk you will see straight through.

The discouraging headlines are not entirely wrong.

The job market for brand new programmers genuinely got harder over the last couple of years. Companies that used to hire armies of junior developers to do the simple, repetitive work discovered that the machine could do a lot of that simple, repetitive work. Entry level openings thinned out. I have friends who run engineering teams, and the task they used to hand a new graduate, “go write this small straightforward piece,” now often goes to a machine instead. That is real. I will not pretend it away, because you would catch me the first time you went looking for a job.

But look closer at what actually happened, because the headline writers missed it.

The machine replaced a task. It did not replace the person.

The work that vanished was the work that needed the least judgment. The simple stuff. The stuff that was, honestly, never that valuable in the first place. It was just the only way in.

Now look at what did not vanish, because this is the part they get wrong.

Someone still has to decide what to build. Someone has to know whether the thing the machine produced is correct, or safe, or about to cause a disaster. Someone has to take a fuzzy half formed idea from a real human being with a real problem and sharpen it until a machine can act on it. Someone has to stare at a screen full of confident code and say, no, this is wrong, and here is why.

Every one of those is judgment. And judgment is the one thing the machine cannot hand you. It can imitate it. It cannot replace it. Tuck that sentence away, because the entire book is built on it.

I have watched this exact movie before, more than once, which is part of why I am not panicking and neither should you. Early in my career there was a wave of fear that visual tools would mean nobody needed to understand the systems underneath anymore. Drag a button, drop it, done, who needs to know how it works. The people who only learned to drag and drop hit a ceiling almost immediately, the first time something broke in a way the tool had no button for.

The people who understood what was happening underneath picked up the same tools, went faster, and never hit the ceiling at all. The tool was real and useful. It just rewarded understanding instead of replacing it.

AI is that movie, with the volume turned all the way up. It is the most powerful tool any builder has ever held. And it rewards understanding more ruthlessly than any tool before it, because it will so happily, and so convincingly, do the wrong thing when you cannot tell the difference.

So here is the answer I promised you. Should you learn this in 2026?

Yes. But learn it for the right reason, and aim at the right target.

Do not aim to become a person who memorizes syntax. The machine has that covered, and you will lose that race standing still. Aim to become a person who can build, who can judge, who can take responsibility for what comes out the other end. That person is not threatened by AI. That person is holding the leash.

There is a quieter reason too, and for a lot of people it turns out to be the real one. Being able to build the thing you imagine is a kind of freedom. Most people go their whole lives with the software equivalent of a blueprint they can never construct, dependent on someone else to make their thoughts real, or, more often, just never making them real at all. The ability to sit down and build the thing yourself, fix it when it breaks, change it when you change your mind, is a quiet sort of power. It does not show up on a resume. It changes how you move through the world anyway.

The fear that you have missed the boat is almost always wrong, and right now it is spectacularly wrong. The boat just changed shape. The old boat was genuinely hard to board. You had to grind through years of brutal syntax before you could make anything interesting, and a lot of capable people gave up for reasons that had nothing to do with their potential. The new boat lets you build something real far sooner. It only asks one thing in return.

It asks you to keep your brain switched on the entire time, instead of handing your thinking to the machine the second it offers to take it.

That is the whole game. That is the title of this book. Build with your brain on.

The rest of these pages teach you how. But before we get to the how, I have to warn you about something, because there is one mistake that quietly ends more building careers than any tool, any layoff, any market ever has. I made it myself, early, and it cost me. That is where we go next.

Key takeaways

The market got harder for people who only learned the easy, replaceable parts, and easier for people who learn judgment. The machine replaced a task, not a person. The tool is powerful, and it rewards understanding rather than replacing it. You are learning this to become the person who can build, judge, and take responsibility, the one holding the leash, not the one memorizing things a machine already memorized. And underneath the career math sits a simpler reward, the freedom of being able to make your own ideas real.

Exercises

No computer needed. Get a piece of paper. Write down, in one or two plain sentences, the thing you wish existed. Not a career goal, an actual thing. An app, a tool, a small program that would make your life or someone else’s life a little better. Do not worry about whether you can build it yet. You cannot. That is what the book is for. Just name it, and put the paper somewhere you will see it. We are going to come back to it, and by the last chapter you are going to build something, and it might just be that.

Appendix A: the tutor mode prompt patterns, on one page

These are the ways of talking to the machine that keep you in the thinking seat. Keep this page near you. The goal of every one of them is the same. Use the machine fully while staying the brain.

Front load context, always. Begin with who you are, what you are using, what you are trying to do, and what you already understand. Example, I am a beginner working in Python, I understand variables, loops, and functions but not files, I am trying to read a list of names from a file, explain the simplest way and define any new term.

Ask it to teach, not just answer. - Explain this as if to a beginner, and define every new term. - Explain each line and why it is needed, and what would break if it were not there. - Before you answer, ask me what I already know so you can pitch it right.

Ask it to withhold the answer and coach you. - Do not give me the solution yet. Give me one hint and let me try first. - I am stuck. Do not solve it. Ask me a question that helps me find it myself.

The explain back, the closest thing to a cheat code. - Let me say it back in my own words to check I understand. (Your version.) Is that right, and what did I get wrong?

Use it to challenge you, not only help you. - Here is my code. What is wrong with it that I have not noticed? - What input would break this? - What would an experienced reviewer criticize here? - Quiz me on what I just learned, one question at a time, and correct me.

Use it to read and judge code. - Read this line by line as if to a beginner, and pause after each line for me to confirm. - What edge cases does this fail to handle? What happens with empty, wrong, or hostile input? - Is this safe? Walk me through exactly how it handles this sensitive data, and whether that is correct.

Use it to plan, not to think for you. - Here is what I want to build and my proposed minimum version. What is too big, what is missing, what is the simplest version still worth having, and what concrete pieces would I need?

The one rule under all of them. After anything the machine gives you, ask yourself, could I have produced this, and do I understand it now? If you do not understand it, do not accept it yet. Turn it into a lesson first.

Where this goes next

That was chapter one and the prompt pack. The full book, *Build with your brain on*, carries you the rest of the way:

- The real fundamentals, taught with AI as a tutor, so you understand instead of copy.
- The tutor mode method in full: how to use AI to get sharper instead of dependent.
- Reading and judging AI generated code, debugging by reasoning, and keeping what you build safe.
- A complete build: a real web app, from a blank screen to a live page, with every line explained.
- Graded exercises with worked solutions, deeper challenges, and ten clear diagrams.

It comes as a 153 page PDF and a reflowable EPUB.

Get the full book: **[your Gumroad link]**

Build with your brain on. Understand everything, automate the rest, own the outcome.

Mike Salari, salari.dev