Agentic Coding Playbook

Umay ŞAMLI

February 23, 2026

Contents

1 Introduction

This playbook is designed for people who want to learn the best practices of agentic coding. We will be defining **different types of agentic** coding styles and how to choose which style to adopt. We will use **Cursor and Claude Code** for agentic coding examples. Hopefully, this resource will be helpful for those who want to use AI in their coding journey.

2 What is agentic coding?

In the current landscape, agentic coding refers to a person who completely vibe-codes but with well-defined hooks, tests, and paradigms. For this resource, agentic coding will be defined as the process of using AI agents to assist in coding tasks. The level at which it assists can vary. However, if an environment has a deployed agentic coding system, it means it is engaging in agentic coding in some capacity. This approach gives us the ability to manage different styles of using AI in coding.

3 Types of agentic coding

We will be defining 3 different types of agentic coding styles. These are:

- **Agentic Vibe Coder**: This is also known as the person who writes, "hey opus, write me GTA 6 and make no mistakes." This approach lacks proper hooks or tests; the developer just vibe-codes with the agent. This is the most basic form of agentic coding and is not recommended for anything other than testing the capabilities of the models.
- **Classic Agentic Coder**: This is a person who makes the necessary configurations but still heavily relies on the code generated by the agent. Some big tech companies are currently trying to practice this. It is not the best practice but could be extremely useful for small tool or script development.
- **Agentic Developer**: This is a person who uses AI agents to assist in coding tasks but also writes their own code and tests. They also review the code created by the agent. This is the most recommended approach for most developers.

We do not recommend the **Agentic Vibe Coder** style for any production code. This approach is highly susceptible to serious bugs and incidents. Never forget that AI agents are not perfect and can make mistakes. They can hallucinate and may also be biased.

The **Classic Agentic Coder** approach could be useful, especially for security researchers. However, it is not recommended for any software developer writing code for a commercial product.

The **Agentic Developer** approach is the best practice for developers. It can boost productivity and help write better code, but it must be applied properly.

4 How to configure my agent?

When you decide to adopt an agentic approach, you need to configure your agent properly. Proper configuration will significantly increase the code quality and capabilities of your agent. We will take a look at how to configure an agent.

4.1 Skills

Skills are mainly helpers for the agents. They give context to the agent to perform tasks better. There are a lot of skills that you can use. You can find skills at `https://skills.sh/`. There are also other sites that host skills for agents.