How to Code Using Claude AI

ZolaXTech · a practical workflow, not just prompt examples

1. Pick the Right Surface

Claude is available in a few different places, and which one you use changes your workflow more than the prompt does.

Surface	Best for
Claude.ai (chat)	Explaining concepts, reviewing snippets, planning before you write code
Claude Code (CLI/IDE)	Working directly in a real repo — multi-file edits, running tests, git operations
API / SDK	Building AI features into your own product

2. Give Context Before You Ask for Code

The single biggest quality difference in AI-assisted coding is how much real context the model has. Paste the actual error message, the relevant function, and what you've already tried — not just "this doesn't work."

```
Weak: "Why is my React component not updating?" Strong: "This component re-renders on prop change but not on this internal state update: [paste code]. I expected the list to re-sort after setSortOrder runs. I've confirmed setSortOrder is being called with console.log."
```

3. Ask for a Plan Before a Full Implementation

For anything beyond a small function, ask Claude to outline its approach first — files it will touch, the order of changes, edge cases it's considering — before generating code. This catches misunderstandings while they're still cheap to fix, instead of after 200 lines are written the wrong way.

```
"Before writing code, outline your approach: which files change, in what order, and any edge cases I should confirm first."
```

4. Review Before You Run — Every Time

- Read the diff, not just the summary. AI-generated code can be subtly wrong in ways that look plausible.
- Check anything touching auth, payments, or data deletion especially carefully — these are the highest-cost places to be wrong.
- Ask "what could break with this change?" as a follow-up prompt. It's a cheap way to surface edge cases the model didn't mention unprompted.
- Run your test suite, not just the one case you were debugging.

5. Use It to Read Code, Not Just Write It

Some of the highest-value uses of Claude in a coding workflow aren't generation at all: pasting an unfamiliar file and asking for a walkthrough, asking it to trace a specific execution path through a codebase, or having it explain why a particular pattern was likely chosen. Understanding existing code well is often the actual bottleneck, not writing new

code.

6. Common Mistakes

- Accepting a large generated change without reading it line by line.
- Asking for a fix without sharing the actual error text — the model ends up guessing at the same rate you would.
- Treating one confident-sounding answer as verified. Confidence in tone isn't evidence of correctness — check against docs or by running it.
- Not telling the model about constraints (framework version, team conventions, performance requirements) that a human teammate would already know.

This guide reflects general good practice for AI-assisted development as of 2026 — specific tool interfaces change, but context, review, and verification stay the fundamentals.