CHAPTER ONE

The Vibe-Coding Revolution

How people who never wrote code started shipping real software — and why building the app is the easy part.

FROM CLAUDE CODE TO THE APP STORE

Something strange happened in 2025.

People who had never written a line of code started shipping real software. Not toy projects. Not demo apps. Real products that real people paid real money for — on the App Store, on the web, everywhere.

The tools changed everything. Cursor, Claude, Bolt, Replit Agent, v0 — suddenly you could describe what you wanted in plain English and watch an AI write the code. Designers were building apps. Marketers were shipping MVPs. Founders who'd spent years outsourcing development were doing it themselves in a weekend.

They're calling it "vibe coding." You don't need to understand every line. You describe the vibe, the AI writes the code, you iterate until it works. It sounds like magic. And in a lot of ways, it is.

But here's what nobody tells you.

> **Building the app is the easy part.**

Getting it on the App Store? That's where most people get stuck. Apple's review process is a labyrinth of guidelines, metadata requirements, screenshot specifications, and rejection reasons that nobody explains clearly. I know because I lived it — my app got rejected, and the reason wasn't a bug or a crash. It was the *font size on my subscription pricing screen*.

And even if you get through the review — how do you avoid the rejections that waste days? How do you set up subscriptions correctly on the first try? How do you structure a project so AI actually produces a shippable app instead of a beautiful mess?

That's what this guide is for.

This isn't a story about how I made a million dollars. This is a story about how I actually got an app from "idea" to "live on the App Store."

Who This Is For

You've built something. Maybe it's an iOS app, maybe it's a web app, maybe it's half-built and sitting in a folder called "my-app-v3-FINAL-actually-final." You used AI to write most of the code. You're not a traditional developer, and you don't want to become one — you want to build products.

You need someone to tell you: here's how you actually take this thing from a folder on your laptop to live on the App Store.

That's me. I built PrimeState — a cognitive optimization tracker — using Claude Code and a lot of late nights. I'm not a developer by training. I navigated the entire process: ideation, building, App Store submission, rejection, resubmission, and everything in between.

I'm going to be honest with you: PrimeState is early. This isn't a story about how I made a million dollars. This is a story about how I actually got an app from "idea" to "live on the App Store" — and everything I learned along the way. The parts that were surprisingly easy. The parts that were unnecessarily hard. And the parts nobody talks about.

What You'll Learn

By the end of this guide, you'll know how to:

THINK IT	Turn a vague idea into a build-ready concept using multiple AI models as your board of advisors. Pressure-test before you write a line of code.
BUILD IT	Use a structured system — the App Incubator — that takes your concept docs and produces a running app. Not hacking and hoping. A repeatable pipeline.
SHIP IT	Take whatever you've built and get it live on the App Store. Not "theoretically possible" — actually submitted, reviewed, and approved.

Every chapter includes real examples, real numbers, and real mistakes. No guru energy. No "just build something people want" platitudes. The actual steps, in order, with the gotchas flagged.

The Gap Nobody's Filling

There are a thousand tutorials on how to use Cursor. There are a thousand videos on prompt engineering. There are entire YouTube channels dedicated to building apps with AI in 30 minutes.

But when you search "how to actually get an AI-built app on the App Store," you get... nothing useful. Blog posts from 2019. Apple's developer documentation (good luck). Reddit threads full of conflicting advice. Stack Overflow answers for a version of Xcode that doesn't exist anymore.

The vibe-coding tools solved the building problem. Nobody solved the *everything else* problem.

Until now. Let's go.