FREE SAMPLE

Build Your First AI-Powered SaaS in a Weekend

The step-by-step blueprint for entrepreneurs who want to build — no coding experience required.

Sample includes: Introduction · Chapter 1 · Prompt Structures (Ch. 4)

Get the full guide → shipitguide.com

Get Full Guide — $19.99 →

INTRODUCTION

Before You Start Reading This

> *"The best time to build was 10 years ago. The second best time is right now — and for the first time in history, you don't need to know how to code."*

This guide will teach you to build a real, live web application using artificial intelligence as your developer.

We'll build a specific example together: a Client Proposal Generator with a Mini CRM — a tool that creates professional proposals using AI, tracks your deals, and sends emails to clients. It's a real product people pay for.

But here's what matters most: The example is just the vehicle. The skills, process, and exact steps you learn apply to any web app you want to build.

You are not locked into our example. You are learning a process.

Define Your App (Fill This In Now)

Before we start, take 3 minutes and answer these questions about the app you want to build.

My app idea: _____

The problem it solves: _____

Who uses it: _____

The 3 main things it needs to do:

1. _____
2. _____
3. _____

What You'll Need

Before we install anything, let's be honest about costs:

Tool	Cost	What for
Claude Pro	$20/month	Your AI developer
Supabase	Free to start	Database + user auth
Vercel	Free to start	Hosting your app
Resend	Free to start	Sending emails
Stripe	Free + 2.9%	Taking payments
Upstash	Free to start	Preventing abuse

→ *Total while building: $20/month. Every other tool is free until you have paying customers.*

What kind of computer do you need: Any Mac or Windows made in the last 7 years. Nothing special required.

PART ONE: SETUP

One afternoon. Never again.

Chapter 1

Opening Your Terminal

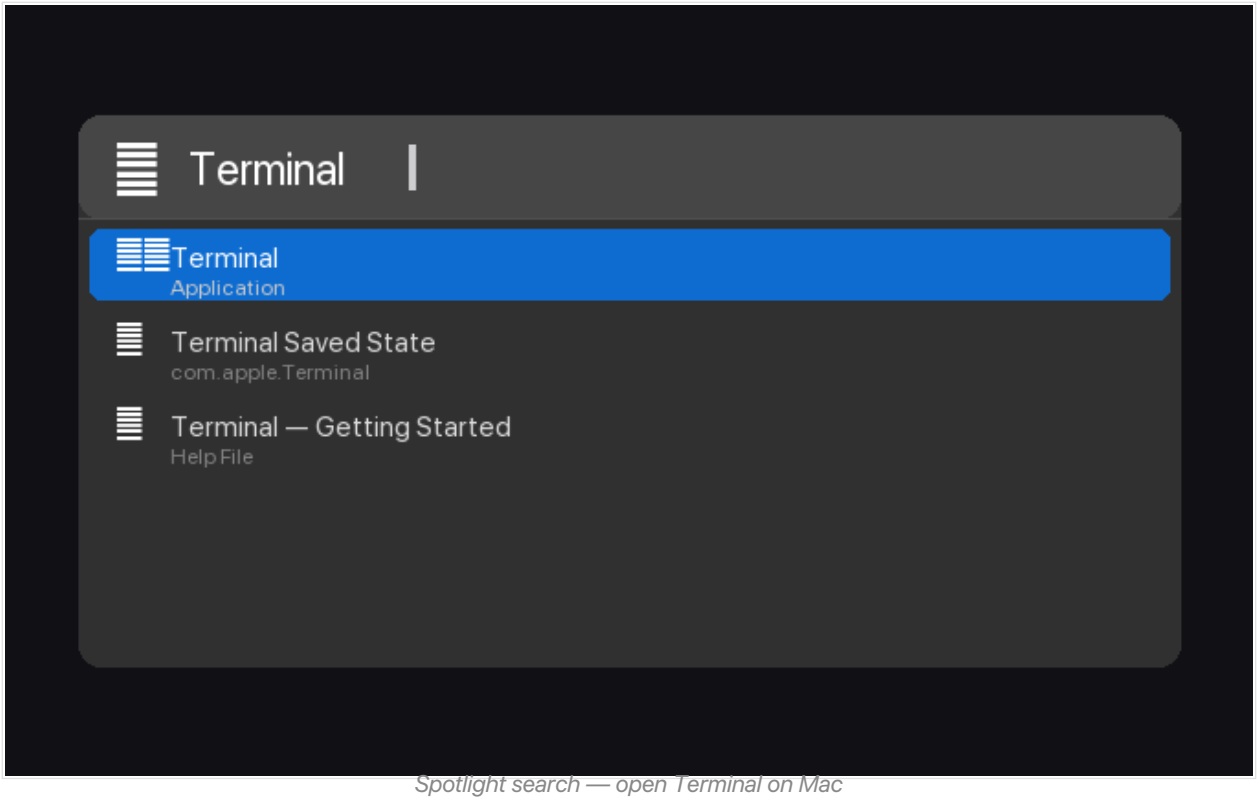
Everything in this guide happens in the Terminal — a text-based interface where you type commands and your computer executes them. It sounds intimidating. It's actually simpler than it sounds.

Think of it like texting your computer. You type what you want. Your computer does it.

On Mac

1. Press ⌘ + Space to open Spotlight
2. Type "Terminal"
3. Press Enter

You'll see a window with a blinking cursor. That's it. That's the Terminal.



Spotlight search — open Terminal on Mac

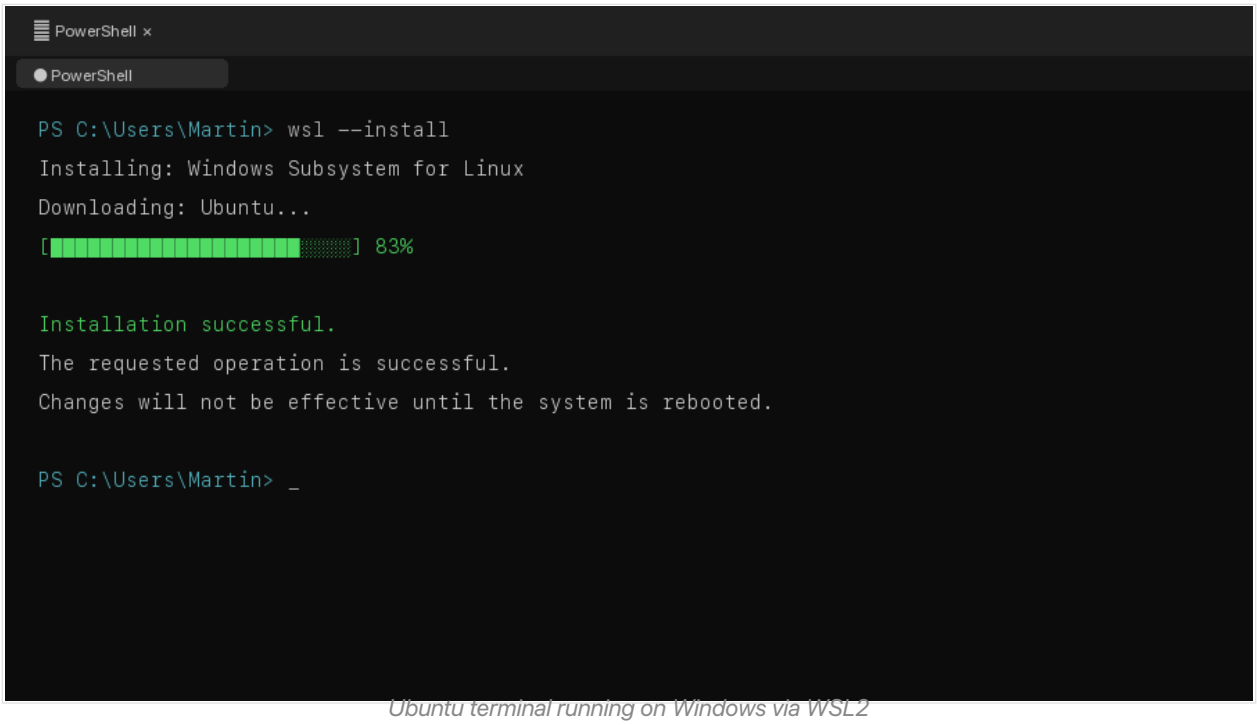
On Windows

Windows requires one extra step: installing WSL2 (Windows Subsystem for Linux). This gives your Windows computer a Linux environment — the same one professional developers use. This is a one-time setup.

1. Click the Start button
2. Search for "PowerShell"
3. Right-click PowerShell → "Run as administrator"
4. Type this command and press Enter:

```
wsl --install
```

5. Wait for installation to complete (about 5–10 minutes)
6. Restart your computer when prompted
7. After restart, open "Ubuntu" from your Start menu
8. Create a username and password when prompted



Ubuntu terminal running on Windows via WSL2

→ *From this point forward, all instructions are the same for Mac and Windows. When we say "open Terminal," Mac users open Terminal, Windows users open Ubuntu.*

PART TWO: HOW TO TALK TO YOUR AI DEVELOPER

Chapter 4

The Exact Prompt Structures That Work

Most people fail with Claude Code because they ask vague questions and get vague results. The difference between "it just works" and "I have no idea what happened" comes down entirely to how you structure your prompts.

Claude Code is not a search engine. It's a developer waiting for a well-scoped ticket. The more precise you are about what you want, what you have, and what success looks like — the better the output.

Pattern 1: The Feature Build Prompt

Use this when you want to add something new to your app. Be specific about what it does, what data it touches, and what it returns.

```
Build a [feature name] that:
- Does [specific thing 1]
- Does [specific thing 2]
- Stores data in [table name] in Supabase
- Returns [what you expect back]

Use TypeScript. Follow the existing code patterns in this project.
```

→ *Always end with "follow the existing code patterns." This keeps Claude consistent with your codebase.*

Pattern 2: The Fix Prompt

Use this when something is broken. Paste the exact error — never paraphrase it.

```
I'm getting this error:
[paste the exact error message]

It happens when I [describe what you did].
The relevant file is [filename].
Fix it without changing the behavior of other features.
```

Pattern 3: The Explain Prompt

Use this when you want to understand what was just built. Don't skip this step.

```
Explain what [filename or feature] does in plain English.
Focus on: what data flows in, what happens to it, what comes out.
I'm not a developer — skip the technical jargon.
```

Pattern 4: The Refactor Prompt

Use this when something works but feels messy, or when you're about to build on top of quickly-written code.

```
Refactor [filename] to be cleaner and more maintainable.
Requirements:
- Keep all existing functionality exactly the same
- Add TypeScript types where missing
- Add comments explaining non-obvious logic
- Don't change the API interface
```

The more specific your prompt, the better the output. Vague in = vague out. Specificity is the skill. You get better at it by doing it.

These four patterns cover 90% of everything you'll do while building. The remaining 10% is covered in the Bonus Prompt Library at the end of the full guide — 30+ ready-to-use prompts for every situation.

— End of Sample —

Ready to build your first app?

The full guide covers all 19 chapters — setup, database, auth, AI features, email, CRM, Stripe payments, deployment, getting users, and more.

Get Full Guide at shipitguide.com →

One-time $19.99 · Lifetime updates · Instant download