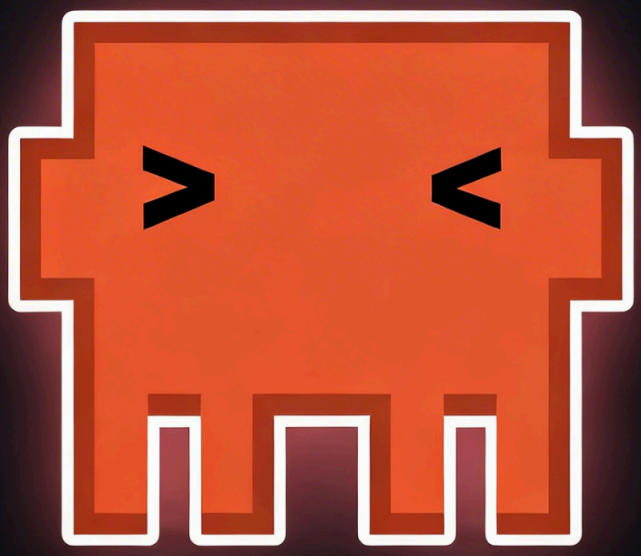
From Idea to App

Your Absolute Beginner's Guide to Claude Code



Fabian Markl

> This guide gives you copy-paste prompts for Claude Code (in the Claude Desktop app or at claude.ai) to go from zero to a working app — even if you've never written a line of code.

📚 What's in This Guide

This guide is organized in 5 parts. Work through them in order the first time, then jump to any section you need.

🔗 Get the Tools

PRO Voice Typing Chrome Extension — Instant Setup:
https://fabimarkl.com/library/#voice-to-content

Download Claude Desktop:
https://claude.ai/downloads

Build Automations and AI Agents — Let AI Build Them For You, Just like Claude Code builds apps, this kit lets AI build your automations and agents:
https://fabimarkl.com/ai-builder-kit

PART 1 — Getting Started

What is Claude Code?

Claude Code is an agentic coding tool made by Anthropic. Unlike the regular Claude chat (where Claude just talks to you), Claude Code can actually create files, write code, and run commands on your computer — or in the cloud.

> Think of it like having a developer sitting next to you who can actually type on your keyboard, run commands, and check if things work.

Where Can You Use Claude Code?

Claude Code works in several places — you don't have to use the terminal if that feels scary:

- 💻 **Claude Desktop App** — The easiest starting point. Install the app, open it, and Claude Code is ready inside it.
- 🌐 **claude.ai/code in your browser** — No installation needed. Open the website and assign coding tasks directly.
- 📱 **Claude iOS App** — Kick off or monitor tasks from your phone while Claude works in the background.
- 🖥️ **VS Code or JetBrains** — For developers who already use these editors — Claude Code integrates as a plugin.

What is a "Project Folder"?

When you use Claude Code on your computer, you give it access to a folder. This folder is your project's home — everything Claude creates goes in there. Claude can only touch files that are inside this folder.

> Think of it like giving Claude a desk drawer to work from. It can only touch what's in that drawer.

To set your project folder:

- Create a new empty folder somewhere on your computer (e.g., your Desktop, named something like my-app)
- When starting Claude Code in the Desktop app, point it to that folder
- Everything Claude creates will appear inside that folder

PART 2 — What Claude Code Can and Cannot Do

✅ What Claude Code CAN Do

🔨 Build Things

- Create and edit files: HTML, CSS, JavaScript, Python, JSON, and more
- Build Chrome Extensions that work in your browser
- Build web apps — websites that run locally or on the internet
- Build full-stack apps with databases (e.g., PostgreSQL, Supabase) and payments (Stripe)
- Build native mobile apps — iOS (SwiftUI), Android, React Native, Flutter
- Build desktop tools using frameworks like Electron (for Mac and Windows)
- Write and run tests to check its own work automatically

🌐 Search the Web and Fetch Live Content

Claude Code can search the internet and read web pages in real time. This means it can:

- Look up the latest documentation for any tool or library
- Research how to solve a specific error or bug it encounters
- Fetch and read content from any public website to help with your task
- Stay up to date — it's not limited to what it was trained on

🖥️ Control Your Browser (with Claude in Chrome)

When you install the free "Claude in Chrome" extension, Claude Code gains the ability to directly interact with websites in your browser:

- Navigate to pages, click buttons, fill out forms
- Read console errors, monitor network requests, and inspect page elements
- Test your web app live — build it in the terminal, verify it in the browser
- Access sites you're already logged into (Gmail, Notion, Google Docs, etc.)
- Record a workflow once and let Claude repeat it automatically

> ⚠️ Safety note: Claude will ask your approval before taking sensitive actions (submitting forms, making purchases, sending messages). Some websites may try to trick Claude with hidden instructions — always review what it's doing on financial or sensitive sites. Chrome and Edge are supported; Brave, Arc, and Firefox are not supported yet.

🔁 Work Across Multiple Sessions

Claude Code is designed to persist project context across sessions. While each session starts with a fresh context window, Claude Code can:

- Read your CLAUDE.md project notes file to pick up where it left off
- Remember your project's structure by reading the files in your folder
- Run tasks in the cloud while you're away, then report back when done
- Be controlled remotely from your phone or browser while running on your computer

⚙️ Run Commands and Manage Your Project

- Install software packages it needs (npm, pip, etc.)
- Run your code and read error messages to fix them automatically
- Start local development servers so you can see your app in the browser
- Create entire project structures with all files in the right places
- Work on multiple tasks in parallel (on cloud plan)

❌ Limitations to Know About

✗ **Not perfect on first try** — Expect some back and forth, especially for complex apps. Plan to test and iterate.

✗ **Large codebases get harder** — Beyond ~10,000 lines of code, Claude may struggle with the big picture and break things. Best for smaller focused projects.

✗ **Manual external setup** — App Store Connect, Google Play, and some third-party accounts must be set up by you manually.

✗ **"Early victory" problem** — Claude sometimes declares a task done before it's fully tested. Always verify with the test prompts in Part 4.

✗ **No memory between sessions by default** — Start each session by pointing Claude to your project notes file (see Prompt 11).

✗ **Chrome/Edge browser only** — The browser control feature doesn't work on Brave, Arc, Firefox, or Safari yet.

PART 3 — Planning Prompts

> How to use these prompts: Copy any prompt below and paste it into Claude Code. Fill in the [brackets] with your own details. Use them in order the first time.

PROMPT 1 — Describe Your Idea (Start Here)

```
I want to build something but I'm not a developer. Please help me define my idea
clearly.

Here's what I want in plain language:
[Write your idea here in 2-5 sentences. Don't worry about being technical.]

Example: "I want a tool that sits in my Chrome browser and whenever I'm on
LinkedIn,
it highlights people who went to the same university as me."

Please ask me clarifying questions — one at a time — to help me understand:
1. Who will use this?
2. What problem does it solve?
3. What happens step by step when someone uses it?
4. What information goes IN, and what comes OUT?

After the questions, summarize my idea back to me in simple terms so I can confirm
you understood it correctly.
```

PROMPT 2 — Map the Full Process

```
Now that you understand my idea, please map out the full process as a workflow.

For each step, tell me:
- What triggers this step? (e.g., user clicks a button, page loads, timer fires)
- What INPUT does this step need? (data, files, user actions)
- What happens in this step?
- What OUTPUT does this step produce?
- What TOOL or TECHNOLOGY handles this step?

Also tell me:
- Where does data get saved? (in the browser, in a file, in a database?)
- Does this need the internet? If yes, what for?
- Are there steps that happen automatically vs. steps the user triggers?

Format it as a numbered list so I can see the full flow from start to finish.
```

PROMPT 3 — Explore Your Solution Options

```
I want to automate or improve this process I currently do manually:

WHAT I DO NOW:
[Describe what you do step by step. Be specific.]
Example: "Every morning I open 5 tabs — Gmail, Notion, my calendar, Slack, and a
Google Sheet
— and manually copy numbers from the sheet into a Notion page."

TOOLS I USE:
[List the tools involved, e.g.: Gmail, Notion, Google Sheets, Slack]

WHAT I WISH HAPPENED:
[Describe your dream outcome]
Example: "I wish this all happened automatically and I just saw a summary."

Please suggest 3 different solutions I could build:
1. A Chrome Extension
2. A Web App
3. A Desktop Tool or Script

For each one, explain:
- What it would look like and how I'd use it
- The main advantage and the main limitation
- How hard it is to build (easy / medium / complex)
- What you'd need from me to build it (API keys, accounts, etc.)

Then recommend which one is best for my situation and why.
```

PROMPT 4 — Create the Full Build Plan

```
We've decided to build: [Name of your solution]

Before writing any code, please create a complete build plan that includes:

1. PROJECT OVERVIEW — What it does in 2 sentences and who uses it

2. ALL INPUTS — Every piece of data, file, or user action the app needs
   and where each one comes from

3. ALL OUTPUTS — Everything the app produces or displays
   and where each output goes

4. STEP-BY-STEP PROCESS — Every step from start to finish
   and what tool/technology handles each step
```

```
5. FILES WE WILL CREATE — Every file with a one-line description

6. TECHNOLOGIES WE'LL USE — Each technology and why we need it

7. THINGS THAT COULD GO WRONG — Top 3 risks and how we'll handle each

8. BUILD ORDER — What to build first, second, third, and why

Please write this as a clear document I can refer back to.
Do NOT start writing code yet. I want to review and approve this plan first.
```

PROMPT 5 — Plan Review & Refinement

```
Here is the plan you created. Let's review it together before building.

Please re-read your plan and then answer:

1. Is there anything I likely WANT that isn't in this plan yet?
2. Are there edge cases — unusual situations that might break this?
3. Are there steps where you'll need information from me that you don't have yet?
   (API keys, passwords, account details, preferences)
4. Is the scope realistic? Could we simplify anything without losing core value?
5. Is there anything that would make this much better with only a little extra
effort?

After your answers, give me a REVISED final plan with any improvements included.
This revised plan will be what we actually build.
```

PART 4 — Build Prompts

> Use these prompts during the actual build process — to start coding, test each piece, fix problems, and wrap up sessions.

PROMPT 6 — Start the Build

```
The plan is approved. Let's start building.

Rules for this build:
- Build one piece at a time, in the order we agreed
- After each piece, tell me: what you just built, what to do next,
  and if I need to test anything
- If you hit a problem, explain it in plain English before trying to fix it
- If you need to make a decision, ask me instead of deciding alone
- Add comments in the code explaining what each section does in simple language
- If you're unsure about a library or API, search the web for the latest docs
  rather than guessing from memory
- At the end, give me a "progress summary" so we can pick up easily next time

Please start with: [Step 1 from your approved plan]
```

PROMPT 7 — Test What Was Just Built

```
You just finished building [describe what was just built].

Before we continue, let's test it properly. Please:

1. Tell me exactly what I should do to test this (step by step,
   like I've never done this before)
2. Tell me what I should SEE if it's working correctly
3. Tell me what might look wrong even if it IS working
   (so I don't panic unnecessarily)
4. Give me 3-5 specific things to try that would catch the most common problems
5. Tell me what information to send you if something goes wrong
   (error messages, screenshots, what happened)

I'll test it and report back.
```

PROMPT 8 — Something Isn't Working

```
Something isn't working. Here's what I can tell you:
```

```
WHAT I DID:
[Describe exactly what you did, step by step]

WHAT I EXPECTED:
[What did you think would happen?]

WHAT ACTUALLY HAPPENED:
[What happened instead? Be as specific as possible]

ERROR MESSAGE (if any):
[Paste the full error message here, exactly as it appears]

WHAT I ALREADY TRIED:
[Did you try anything to fix it? What happened?]

Please:
1. Search for this error online if you don't immediately know the cause
2. Explain in plain English what you think went wrong
3. Tell me if this is a simple fix or a bigger problem
4. Fix it, or tell me what you need from me to fix it
5. Tell me how to confirm the fix worked
```

PROMPT 9 — Add a New Feature Mid-Build

```
I want to add a new feature. But first, help me decide whether to do it now.

NEW FEATURE IDEA:
[Describe what you want to add]

Please tell me:
1. Does this fit naturally with what we've already built,
   or would it require major changes?
2. How much additional work is this — small, medium, or large?
3. Should we do it now or save it for later?
4. Does adding this create any new risks for the parts already built?
5. If we do it now, what's the updated build plan?

If you recommend we add it now, proceed after I confirm.
If not, save the idea to a file called "future-features.md"
in my project folder so we don't forget it.
```

PROMPT 10 — Wrap Up a Session

```
We need to wrap up for now. Before we stop, please create a file in my project
```

```
folder called "session-notes.md" with the following:

1. WHAT WE BUILT TODAY
   List everything completed in this session

2. CURRENT STATE
   What is working right now?
   What is partially built but not finished?

3. WHAT'S NEXT
   The exact next step we should take
   Any decisions I need to make before the next session

4. OPEN QUESTIONS
   Anything you're unsure about or that I need to decide

5. HOW TO PICK UP
   The exact prompt I should use at the start of the next session
   (write the prompt for me, ready to copy-paste)

Save this file now so I have it for next time.
```

PROMPT 11 — Resume a Previous Session

```
We're continuing work on my project. I'm sharing my project folder with you.

Please:
1. Read the file called "session-notes.md" if it exists
2. Read any other key files to understand the current project state
3. Summarize for me: what exists, what works, and what was next on the list
4. Confirm the next step you think we should take

Wait for my confirmation before starting any work.
```

PART 5 — Quick Reference

Before You Start Any Project — Checklist

- ☐ I can describe what the app does in 2 sentences
- ☐ I know who will use it (just me? others?)
- ☐ I know what goes IN and what comes OUT
- ☐ I've created an empty project folder on my computer
- ☐ I've used Prompts 1–5 to create and approve a plan

Types of Things You Can Build with Claude Code

Type	What It Is	Best For
Chrome Extension	A mini-app built into your browser (import under chrome://extensions/)	Changing or reading websites, browser automation
Web App	A website that runs locally on your computer	Tools with forms, dashboards, data tracking
Script	A file that runs a task automatically	Processing files, automation, scheduled jobs
Desktop Tool	An app you open like a normal application	Anything you want to run outside the browser
Full-Stack App	A web app with a database and backend	SaaS tools, user accounts, persistent data
iOS / Android App	A native mobile app	SwiftUI, React Native, Flutter projects

Golden Rules for Working with Claude Code

1. Plan before you build. Rushing into code without a plan wastes time and tokens.
2. Test after every piece. Don't let problems pile up.
3. One problem at a time. Don't dump 5 issues into one message.

4. Save session notes. Claude doesn't remember previous conversations by default.
5. It's okay to say "I don't understand." Ask Claude to explain in simpler terms.
6. You can always undo. Ask Claude to reverse what it just did if something broke.
7. Tell Claude to search for docs. If it seems to be guessing, ask it to look up the latest documentation online first.
