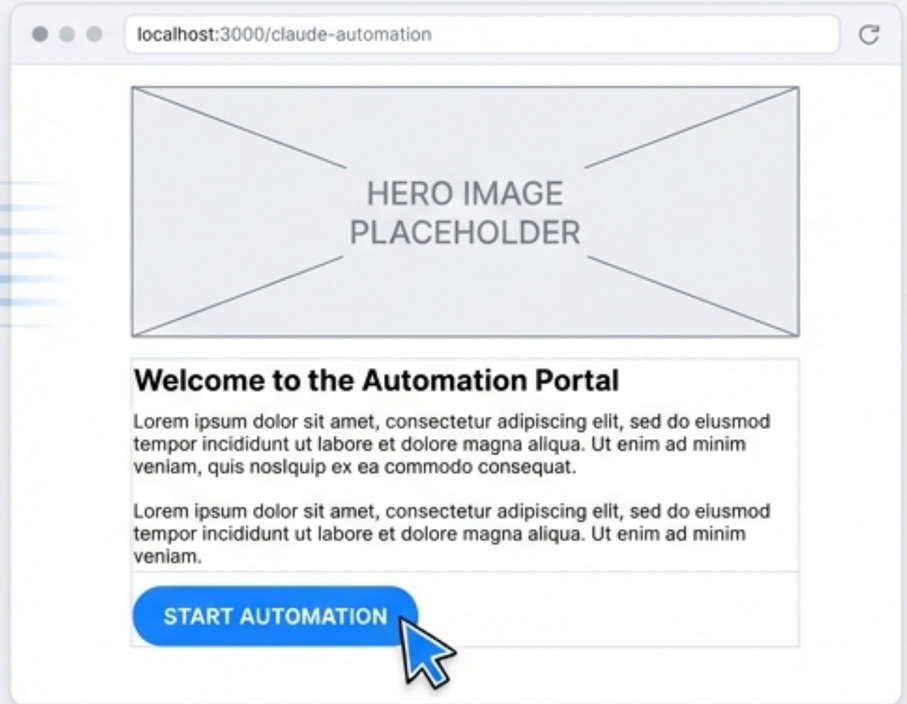
Claude in Chrome

The Browser Automation Playbook

```
> claude --chrome initialization...
Establishing connection to Chrome DevTools Protocol...
```

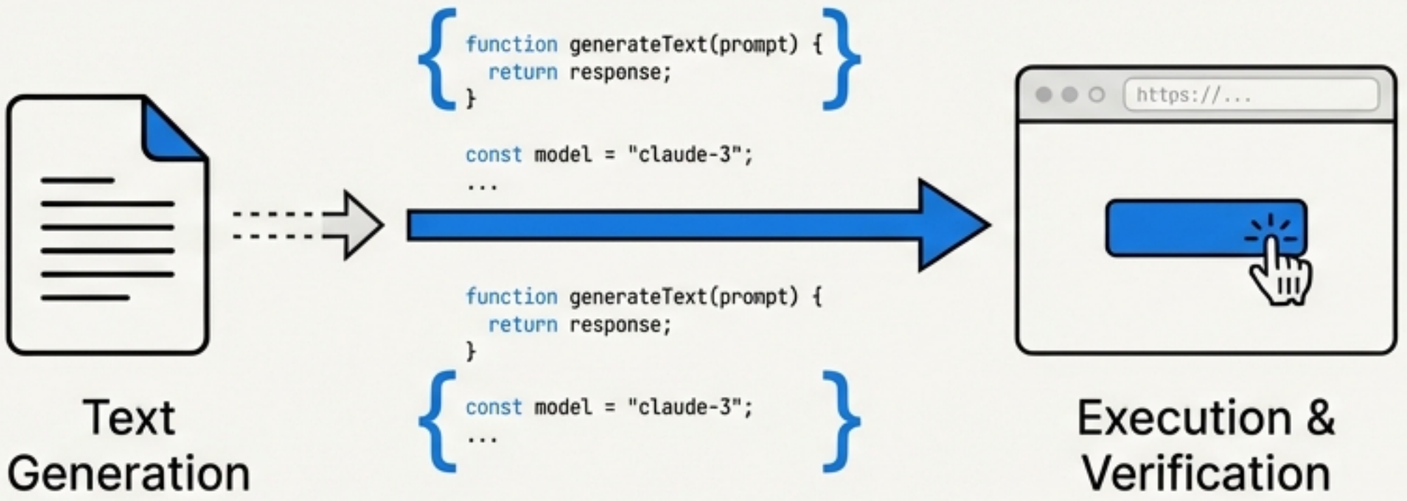


Bridging the gap between Terminal and Runtime.

Moving From Text Generation to Browser Operation

Why Claude is no longer just a chatbot, but a hands-on operator.

The Shift



Research Preview (Aug 2025)
→ Paid Plans (Dec 2025)

The Frictionless State

No APIs Needed

Interacts with the DOM directly. Works on any site you can visit.

No Bot Detection

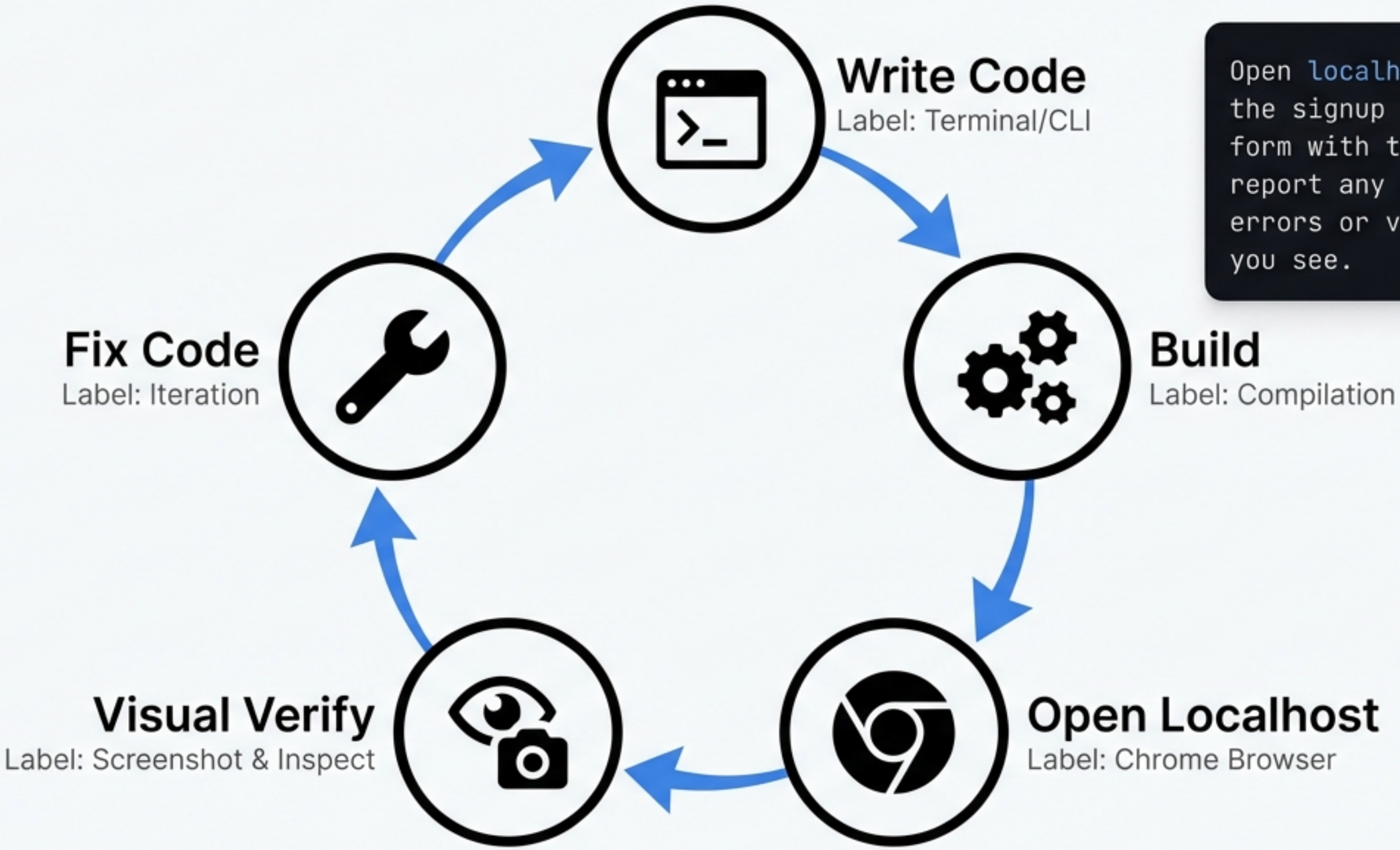
Uses existing browser sessions and cookies. You are the user.

Continuous State

No re-authentication required. Inherits your logged-in context.

The Killer App: The Build-Test-Verify Loop

Turning a solo developer into an engineering team with built-in QA.



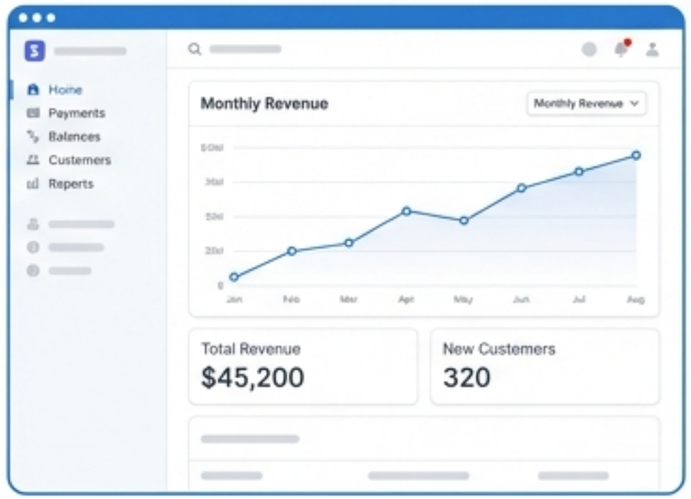

```
Open localhost:3000, test
the signup flow, fill the
form with test data, and
report any validation
errors or visual issues
you see.
```

Solves the "Coding Blind" problem by closing the feedback loop.

Advanced Capabilities Beyond Localhost

Authenticated Interaction

Dashboard Automation

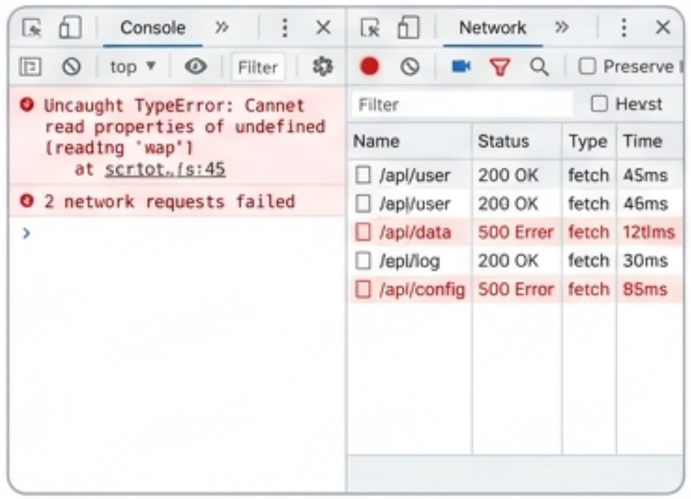


Access data behind logins without APIs or OAuth.

```
JetBrains Mono
"Go to my Stripe dashboard, pull
this month's revenue numbers..."
```

Full-Stack Debugging

Runtime Inspection

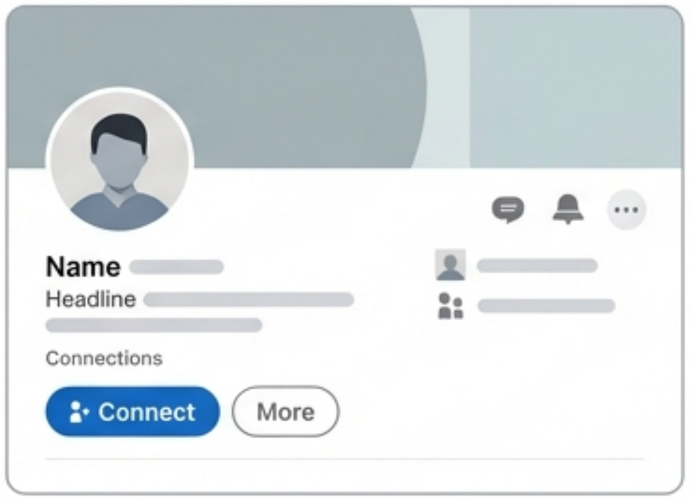


Read console errors and failed network requests directly.

```
JetBrains Mono
"Check network requests for failed
API calls and fix them."
```

Safe Outreach

Human-Like Outreach



Automated interaction using real sessions to avoid bans.

```
JetBrains Mono
"Visit target profiles and send
connection requests."
```

Architecture & Native Messaging

How Claude bridges the CLI and the Browser via Native Host.

Native Host Locations

OS	Location
macOS	~/Library/Application Support/Google/Chrome/NativeMessagingHosts/
Windows	HKEY_CURRENT_USER\Software\Google\Chrome\NativeMessagingHosts\
Linux	~/.config/google-chrome/NativeMessagingHosts/

The MCP Toolset: 17 Capabilities

Navigation

- navigate (URL/History)
- tabs_create_mcp
- tabs_context_mcp

Interaction

- computer (Click, Type, Scroll, Drag)
- form_input
- find (Natural Language Search)

Reading & Analysis

- read_page (Accessibility Tree)
- read_console_messages (Logs/Errors)
- read_network_requests (XHR/Fetch)
- javascript_tool

Media

- screenshot
- gif_creator
- upload_image
- resize_window

System Requirements & Compatibility

Checklist

REQUIRED ✅ ✅

- ✓ **Browser:** Google Chrome or Microsoft Edge (Chromium)
- ✓ **Extension Version:** `1.0.36+`
- ✓ **Claude Code Version:** `2.0.73+`
- ✓ **Plan:** Pro, Max, Team, or Enterprise

NOT SUPPORTED ❌ ❌

- ❌ **Browsers:** Brave, Arc, Firefox, Safari
- ❌ **Environment:** WSL (Windows Subsystem for Linux)
- ❌ **Plan:** Free Tier

Model Availability: Pro (Haiku only) | Max/Team (Opus, Sonnet, Haiku)

Installation: The Critical Path

1. **Install Extension**
   (Chrome Web Store)
2. **Pin to Toolbar**
3. **Sign In**
   (Anthropic Account)
4. **Allow Host**
   (Native Messaging Dialog)
5. **RESTART CHROME**
   Must close completely to register Native Host.
6. **Verify** ✅

Warning: Skipping Step 5 is the #1 cause of “Extension Not Detected” errors.

Terminal Configuration & Activation

The Flag

```
$ claude --chrome
```

Starts session with browser tools loaded.

In-Session

```
> /chrome
```

Toggles connection or reconnects mid-session.

Verification

```
> /mcp
```

Checks loaded tools. Look for ‘claude-in-chrome’.

Permanent Activation

Select 'Enabled by default' in the /chrome menu.

Note: Increases context usage per session.

IDE Support

Works in VS Code & Cursor via ‘@browser’.

Workflow Optimization: Best Practices

Tab Hygiene

Use Tab Groups.

Claude only sees tabs inside its assigned group. Drag relevant tabs in.

The Golden Rule

Start with `tabs_context_mcp`.

Always call this first to prevent stale tab ID errors.

Prompt Engineering

Be Specific.

Bad Prompt: "Check the site."
Good Prompt: "Open `localhost:3000/signup` and screenshot the form."

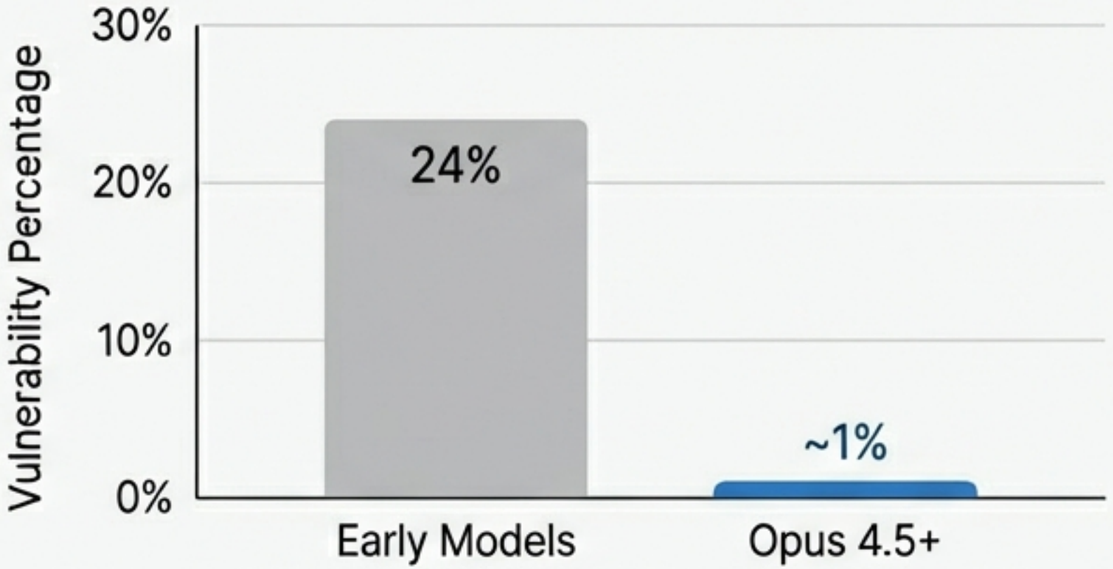
Safety First

Ask Before Acting.

Enable confirmation mode for sensitive sites (email, banking, admin).

Security, Safety, & Hard Limits

Prompt Injection Defense



Layered defenses against malicious web page instructions.

The Hard Limits

Claude **Cannot**:

- ✗ Enter passwords
- ✗ Handle credit card/bank details
- ✗ Create new accounts (Identity Verification)
- ✗ Bypass CAPTCHAs

JavaScript Access: `javascript_tool` has full DOM/Cookie access. Restrict via site-level settings.

Troubleshooting Common Friction Points

Conflict/Issue	The Cause		The Fix
Claude Desktop Conflict	Desktop App and Code CLI compete for the same socket.	⇛	**Close Claude Desktop App** (GitHub #20887).
Stale Tabs / Tab Not Found	Old Tab IDs in context.	⇛	Run `tabs_context_mcp` to refresh.
Action Blocked	Browser Modals (alert/confirm).	⇛	Manually dismiss dialogs in the browser.

Quick Reference Guide

Essential Commands

`claude --chrome`
Start

`/mcp`
Verify Tools

`/chrome`
Toggle/Reconnect

`tabs_context_mcp`
First Step

Key Resources

Docs
code.claude.com/docs/en/chrome

Extension ID
fcoeoabgfenejglbffodgkkbkcdhcgfn

Repo
github.com/charlesdove977/linkedin-automator

Version Gates

Claude Code	2.0.73+
Extension	1.0.36+

Scan for Docs
Chrome Grey Inter