01 MAY 2026

Agentic Coding Tools for *Biomechanics*

By Sinan Koparan

18 SLIDES

Installing *Claude Code*

Anthropic has many products such as Claude Desktop, Claude Code, Claude Design, Claude Cowork. These products are different but have some level of integration with each other.

You can install Claude Code by running a curl command. `curl` is a command-line tool used to transfer data to or from a server using URLs.

On Windows, you also need `Git for Windows` so that Claude Code can use Git Bash. Add the binary to the `PATH` environmental variable — it tells the OS which directories to search when you run a command.

MACOS · LINUX · WSL

```
$ curl -fsSL https://claude.ai/install.sh | bash
```

WINDOWS

```
> curl -fsSL https://claude.ai/install.cmd -o install.
cmd
  && install.cmd && del install.cmd
```

BASH

Bourne Again Shell

A program that lets you interact with your operating system by typing commands instead of using a graphical interface.

Because an agent has access to BASH, it can effectively do anything on your computer.

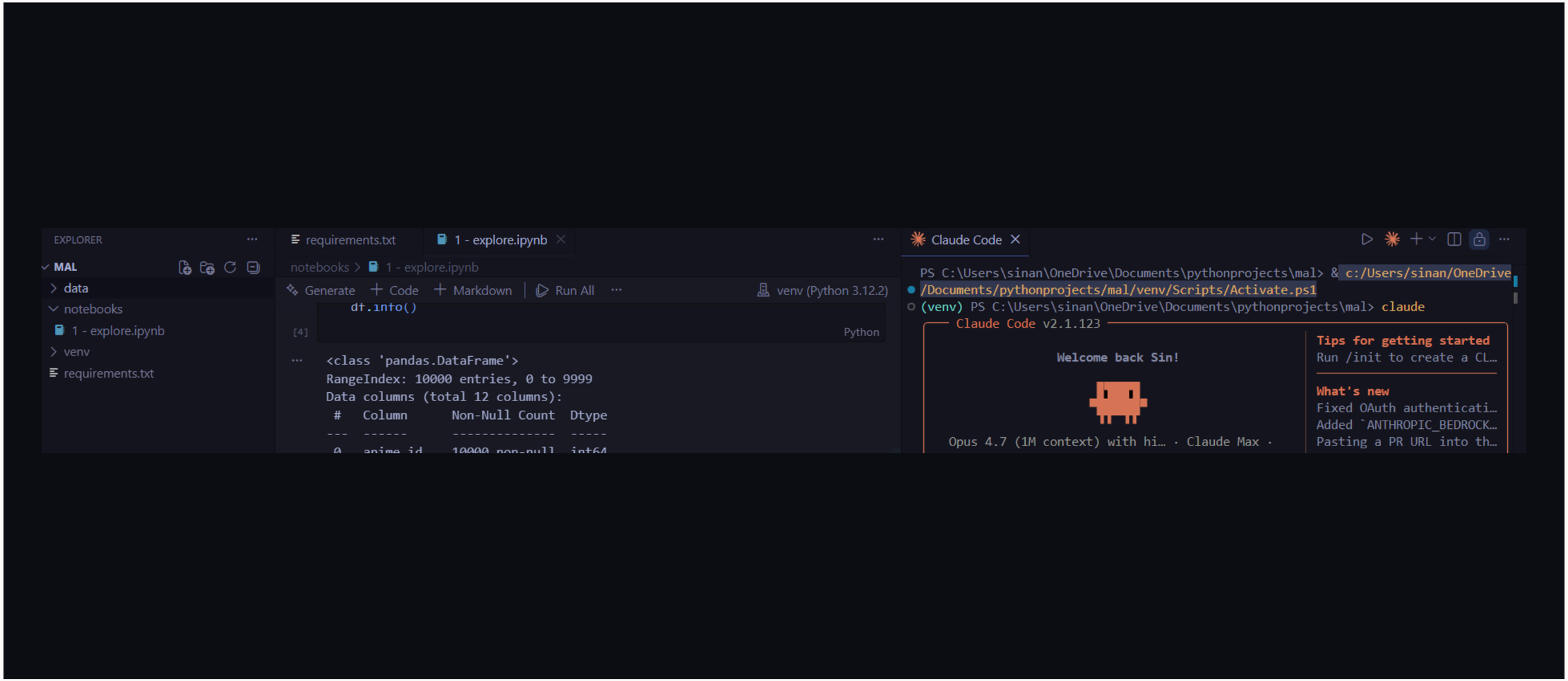
FOR EXAMPLE

```
# list all directories
$ ls

# change directory
$ cd ~/projects

# inspect a file
$ cat README.md
```

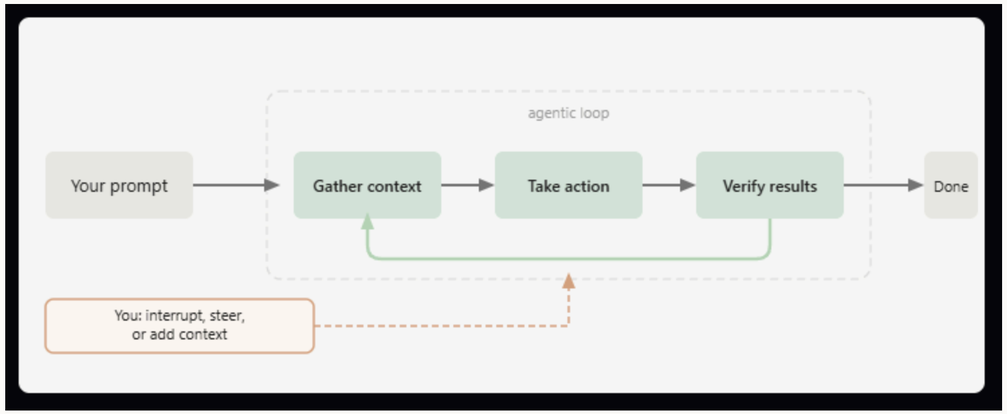
...and more.

Running Claude Code

VSCode Extension



The Agentic *Loop*



Tools

Tools are what make Claude Code *agentic*.

Without tools, Claude can only respond with text.

With tools, Claude can *act*: read your code, edit files, run commands, search the web, and interact with external services.

Each tool use returns information that feeds back into the loop, informing Claude's next decision.

Built-in Tools

CATEGORY	WHAT CLAUDE CAN DO
File operations	Read files, edit code, create new files, rename and reorganise.
Search	Find files by pattern, search content with regex, explore codebases.
Execution	Run shell commands, start servers, run tests, use git.
Web	Search the web, fetch documentation, look up error messages.
Code intelligence	See type errors and warnings after edits, jump to definitions, find references.

Claude also has tools for spawning subagents, asking you questions, and other orchestration tasks.

What Claude can access

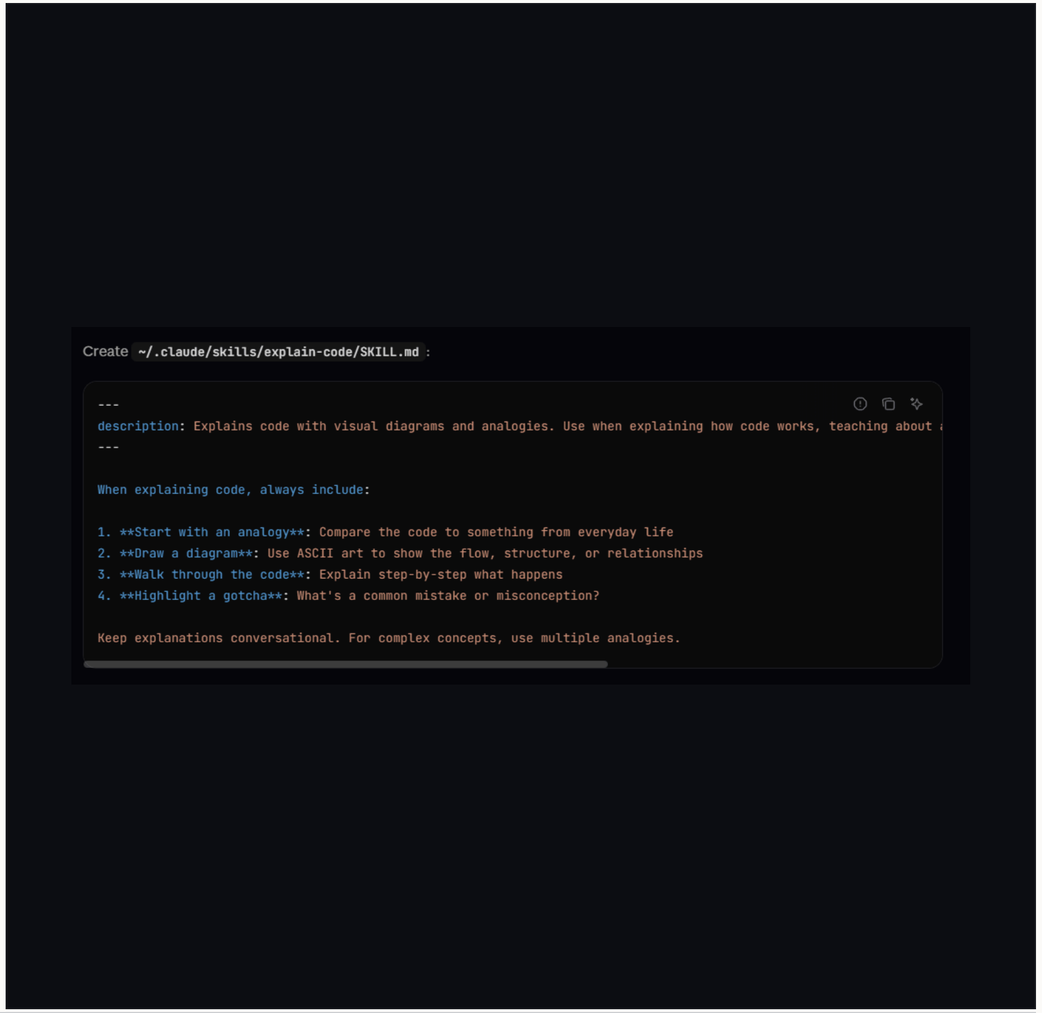
ACCESS	DESCRIPTION
Your project	Files in your directory and subdirectories, plus files elsewhere with your permission.
Your terminal	Any command you could run: build tools, git, package managers, system utilities, scripts. If you can do it from the command line, Claude can too.
Your git state	Current branch, uncommitted changes, and recent commit history.
Your CLAUDE.md	A markdown file where you store project-specific instructions, conventions, and context that Claude should know every session.
Auto memory	Learnings Claude saves automatically as you work, like project patterns and your preferences. The first 200 lines or 25KB of MEMORY.md, whichever comes first, load at the start of each session.

Slash *Commands*

`/rewind`	Allows you to revert to a previous state.
`/model`	Changes a model.
`/init`	Creates a CLAUDE.md file.
`/btw`	Allows you to ask questions while it's generating code.
`/remote-control`	Allows you to continue your session on the Claude mobile phone app.

Skills

A markdown file that describes certain behaviour you want the agent to have.



Three different ways to interact with Claude Code

MODE 01 · RECOMMENDED

STABLE

Planning *Mode*

Claude drafts a plan before touching your code. You review, edit, and approve — then it executes. No surprises, no runaway edits.

- Toggle with `shift` + `tab`
- Covers ~90% of everyday work
- Best place to start as a new user

MODES 02 — 03 · FOR POWER USERS

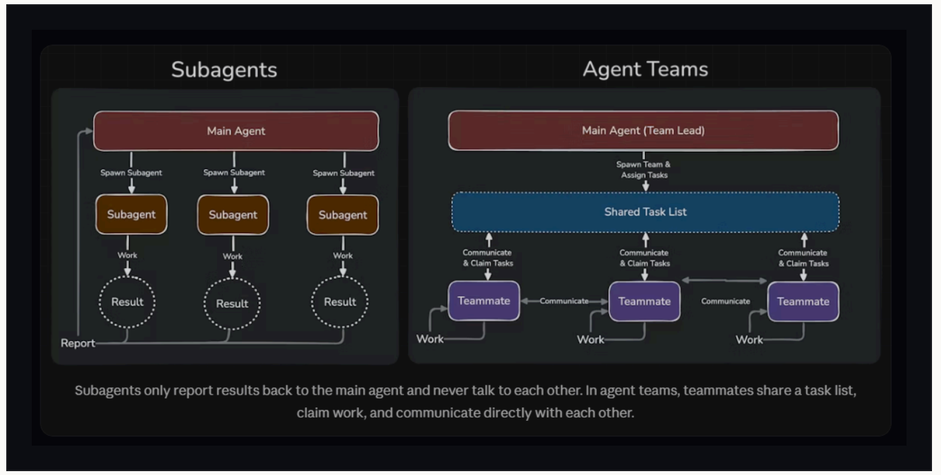
EXPERIMENTAL

Subagents

A main agent spawns helpers that work in isolation and report results back. No cross-talk between them.

Agent Teams

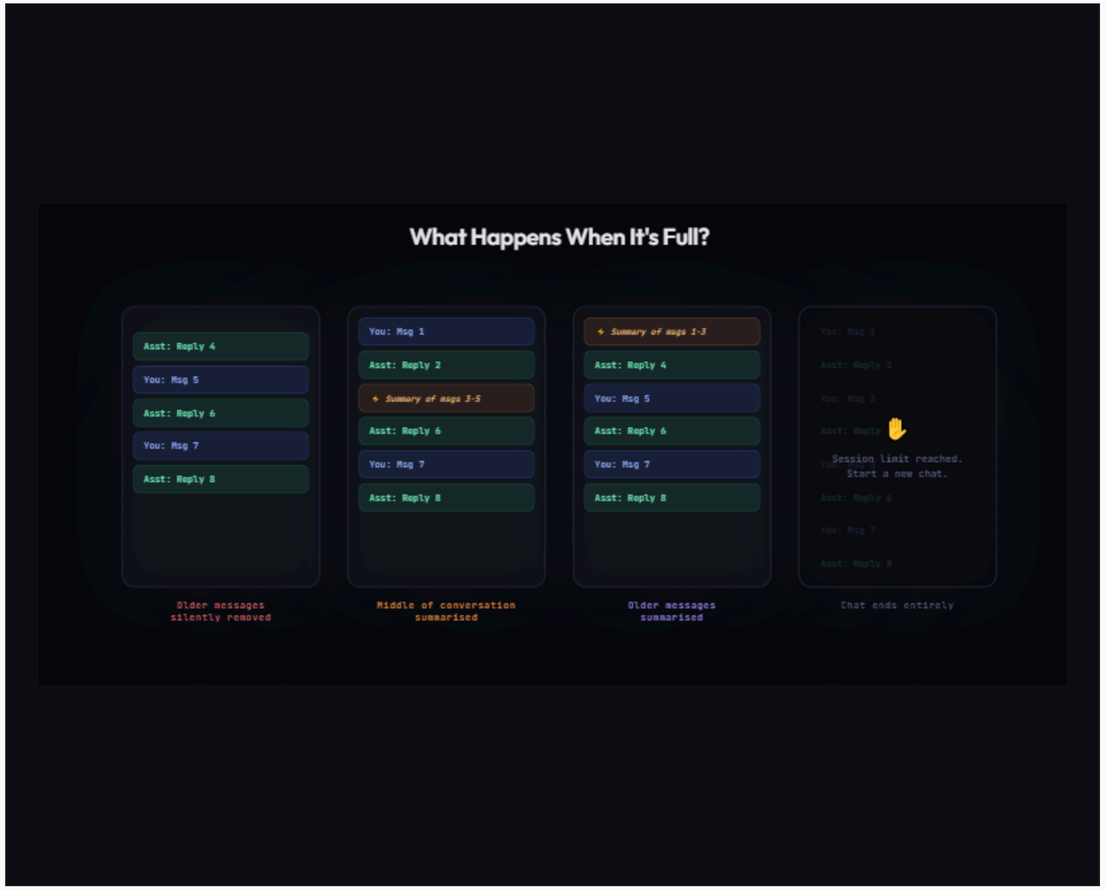
Teammates share a task list, claim work, and communicate directly. Coordinated, but harder to steer.



Context *Window*

A MULTI-TURN CONVERSATION

WHAT HAPPENS WHEN IT'S FULL



Mentality

You can outsource your thinking, but you cannot outsource your understanding.

— YACINE, VIA ANDREJ KARPATHY

Risks — *Prompt Injection & Hallucination Exploitation*

SMART WAYS HACKERS HAVE ABUSED AGENTIC CODING TOOLS

01

Prompt injection via web fetch

Agents use web search to find relevant pages, then may fetch and extract text from those pages before reasoning over it. Because that content is untrusted, a page can include malicious instructions (prompt injection) that try to manipulate the agent's behavior if it isn't properly constrained.

02

Hallucinated package exploitation

Another abuse tactic is that agents may hallucinate Python (not limited to Python, could be in R too) packages. Hackers may register these hallucinated packages on the package manager index (`PyPI`) and input malicious code.

Configuration *scopes*

Claude Code uses a scope system to determine where configurations apply and who they're shared with — for personal use, team collaboration, or enterprise deployment.

SCOPE	LOCATION	WHO IT AFFECTS	SHARED WITH TEAM?
Managed	Server-managed settings, plist / registry, or system-level `managed-settings.json`	All users on the machine	Yes (deployed by IT)
User	`~/.claude/` directory	You, across all projects	No
Project	`.claude/` in repository	All collaborators on this repository	Yes (committed to git)
Local	`.claude/settings.local.json`	You, in this repository only	No (gitignored)

Excluding *sensitive files*

To prevent Claude Code from accessing files containing sensitive information like API keys, secrets, and environment files, use the `permissions.deny` setting in your `.claude/settings.json` file.

```
{
  "permissions": {
    "deny": [
      "Read(./.env)",
      "Read(./.env.*)",
      "Read(./secrets/**)",
      "Read(./config/credentials.json)",
      "Read(./build)"
    ]
  }
}
```

FIN.

Thanks for *listening*.

Sinan Koparan

QUESTIONS WELCOME