

# Agentic Coding: Let Agents Build Agents for you!



ZACHARY HUANG  
MAR 22, 2025

45

12



SI



We all know AI agents are the future, but why build them by hand when AI agents could build them themselves for us? Imagine a world where you simply design the blueprint and AI agents construct the AI agents themselves—this is Agentic Coding, where humans focus on creative, high-level design while AI agents handle all the tedious implementation details.

## What is A

Looks like an article worth saving!

Option Q

Have you ever wished for a better way to live? That's the promise of Agentic Coding.

Hover over the brain icon or use hotkeys to save with Memex.

to

Remind me later

Hide Forever

- You do what humans are best at: coming up with creative ideas and designing your app should work
- AI does what it's best at: writing the detailed code to make your ideas actually work

Now, this is possible! For instance, in the image below, you can see the agentic coding process in action. On the left side, a human designs the system for an AI app, focusing on high-level architecture and data flow. On the right side, with just a simple prompt like “Start Agentic Coding! Implement based on this design doc and add enough logging for us to keep track,” the AI writes all the code with remarkable ease and precision.

Thanks for reading Pocket Flow! Subscribe for free to receive new posts and support my work.

The screenshot shows the Pocket Flow application interface. On the left, there is a "Flow High-level Design" section containing a flowchart titled "Edit File Agent". The flowchart starts with "User Request", leading to "Main Decision Agent", which then branches into "Edit File Agent" (containing "Read File Action", "Analyze and Plan Changes", and "Apply Changes Batch") and "Utility Functions" (containing "Read File Action", "Delete File Action", "Grep Search Action", "List Directory Action with Tree Viz", and "Format Response"). Arrows indicate the flow from "User Request" to "Main Decision Agent", and from "Main Decision Agent" to each of the two main agent sections. On the right, there is a "Agentic Coding Implementation and Logging" section. It shows a file named "design.md" with the content "@design.md Start Agentic Coding! Implement based on this design doc and add enough logging for us to keep track". Below this, a log window shows the AI's progress: "Thought for 5 seconds", "Searched codebase for 'utils file structure or implementation' (utils)", "Searched codebase for 'replace\_file grep\_search utility functions' (utils/)", "Searched codebase for 'call\_llm implementation' (utils/)", and "Now I'll implement the coding agent as requested.". It also lists files being updated: "flow.py +783 -18 ✓", "main.py +41 -9 ✓", and "Edited 2 files +824 -27". At the bottom, there is a "Notes for AI" section with "SSH: 34.145.12.78" and "main\*". A status bar at the bottom says "Dian search build anything".

**Looks like an article worth saving!**

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later      Hide Forever

*Think of it like building a house — you're the architect drawing the blueprint, and AI is construction crew doing the detailed work. You focus on the vision; AI handles the execu*

---

# Why Current AI Building Tools Fall Short

For the past year, I've been trying to build AI apps using existing frameworks like LangChain. The experience has been frustrating for both me and the AI assistants trying to help me:

- **Abstractions upon abstractions** — The fundamental problem with frameworks like LangChain is *abstraction overload*. As [Octomind's engineering team explain](#): “LangChain was helpful at first when our simple requirements aligned with its usage presumptions. But its high-level abstractions soon made our code more difficult to understand and frustrating to maintain.” These complex layers of code hide simple functionality behind unnecessary complexity.
- **Implementation nightmares** — Beyond unnecessary abstractions, these frameworks burden developers with dependency bloat, version conflicts, and constantly changing interfaces. As developers [note](#): “It’s unstable, the interface constantly changes, the documentation is regularly out of date.” Another developer [jokes](#): “In the time it took to read this sentence langchain deprecated 4 classes without updating documentation.”

Looks like an article worth saving!

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever

# why we no longer use LangChain for building our AI agents

When abstractions do more harm than good: Lessons learned using LangChain in production and what we should've done instead



*It's not the AI assistant's fault — many of these frameworks are so complex that even human developers have trouble using them!*

## Introducing Pocket Flow: Simple by Design

Looks like an article worth saving!

Option Q

After a year of struggling to find a way to build AI agents from scratch (code!) that captures everything we need, we're excited to introduce Pocket Flow's simple by design approach.

Hover over the brain icon or use hotkeys to save with Memex.

es (

ope

Remind me later

Hide Forever

- **Crystal clear concepts** — Simple building blocks that any AI agents can understand
- **Zero bloat** — No vendor lock-in or mysterious dependency issues
- **Perfect division of labor** — You design the high-level flow, AI implements the details

My personal journey with frameworks like LangChain taught me that the more complex the framework, the harder it is for AI to help. Pocket Flow was born from realization — designed specifically to create the perfect division of labor between human designers and AI implementers.

---

*When your AI agents aren't fighting to understand a complex framework, they can focus on implementing your vision correctly.*

---

This tutorial focuses specifically on setting up your environment for Agentic Coding with Pocket Flow. For details on implementation and how to use Pocket Flow, please refer to the [official documentation](#).

# Getting Started with Agentic Coding:

*The fastest Development Paradigm in 2025*

Whether you're building your first chatbot or a complex system, this guide will help you set up the perfect environment for Agentic Coding with Pocket Flow.

- **For quick questions:**

- Recommended:  
Setup Steps:  
coding.

**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever

- **For brainstorming or prototyping:**

- Recommended Setup: ChatGPT or Claude Project
- Setup Steps: Create a [ChatGPT Project](#) or a [Claude Project](#), and upload the [docs](#) for reference.
- For full application development:
  - Recommended Setup: Cursor AI, Windsurf, Cline
  - Setup Steps: Create a new project using the [project template](#). The [.cursorr](#) file teaches AI agents how to use Pocket Flow.

**Note:** This list covers the most common setup, but options are changing frequently as new AI coding agents emerge. The core principle remains the same: because Pocket Flow is simple and minimal, any AI agent can quickly understand it just by reading the documentation. Simply share the Pocket Flow docs with your preferred AI agent to get started!

## GPT Assistant: Instant Answers at Your Fingertips

Looking for the fastest way to learn about Pocket Flow? The [Pocket Flow GPT Assistant](#) is your instant knowledge companion! With zero setup required, you can immediately ask questions about [Pocket Flow](#) and receive thoughtful answers within seconds. It's like having a Pocket Flow expert on standby 24/7.

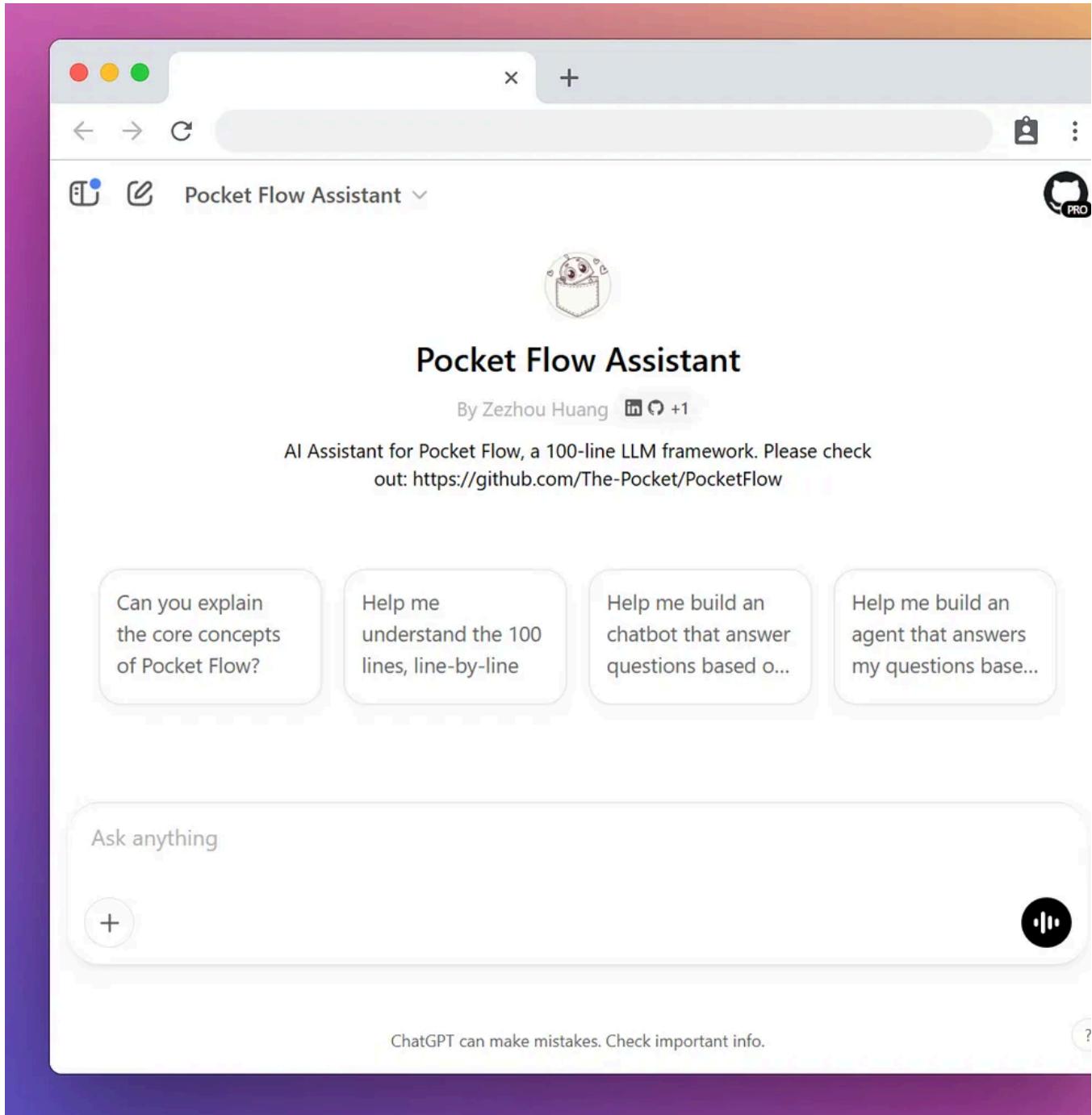
**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



Keep in mind that while perfect for learning and quick questions, the GPT Assistant isn't ideal for implementation work as it's using an older model (likely GPT-4o). For actual coding and building, you'll want to explore the more powerful development options below.

**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

# For One-time Prototypes

Remind me later

Hide Forever

Have a quick idea that you want to brainstorm or prototype, but aren't ready for full implementation? ChatGPT and Claude Projects are good options!

## With ChatGPT:

1. Open [ChatGPT](#) and create a [Project](#)
2. Feed it knowledge by uploading the md files in [Pocket Flow docs](#).
  - a. ChatGPT Project allows at most 20 files. Please exclude `_config.yml`, `utility_function/index.md`, `design_pattern/index.md`, `utility_function/index.md` and `utility_function/llm.md` under `utility_function/`, only upload `utility_function/llm.md`.

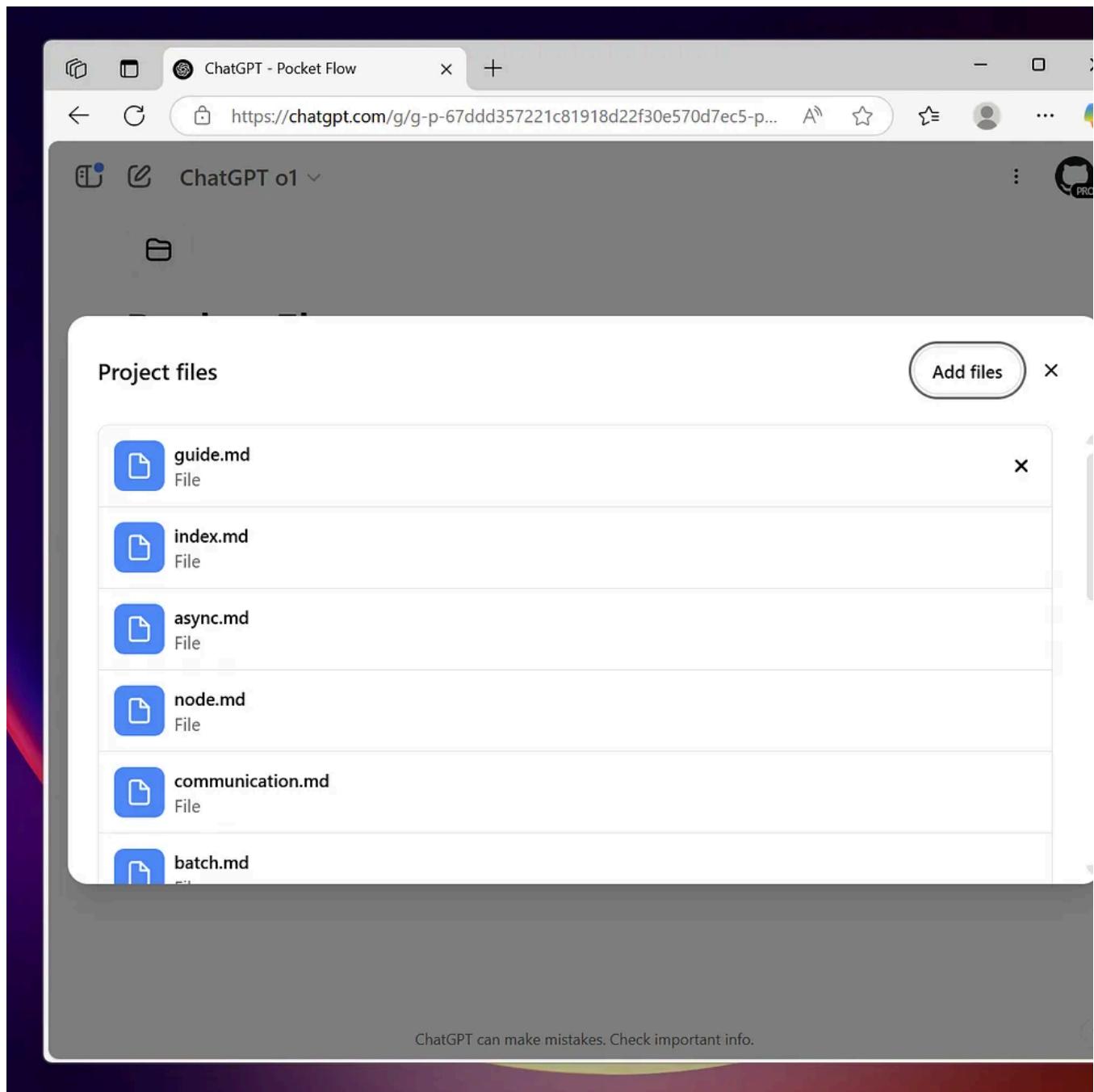
**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



### 3. Set the system prompt to be something like:

If a user asks you to build an AI app:

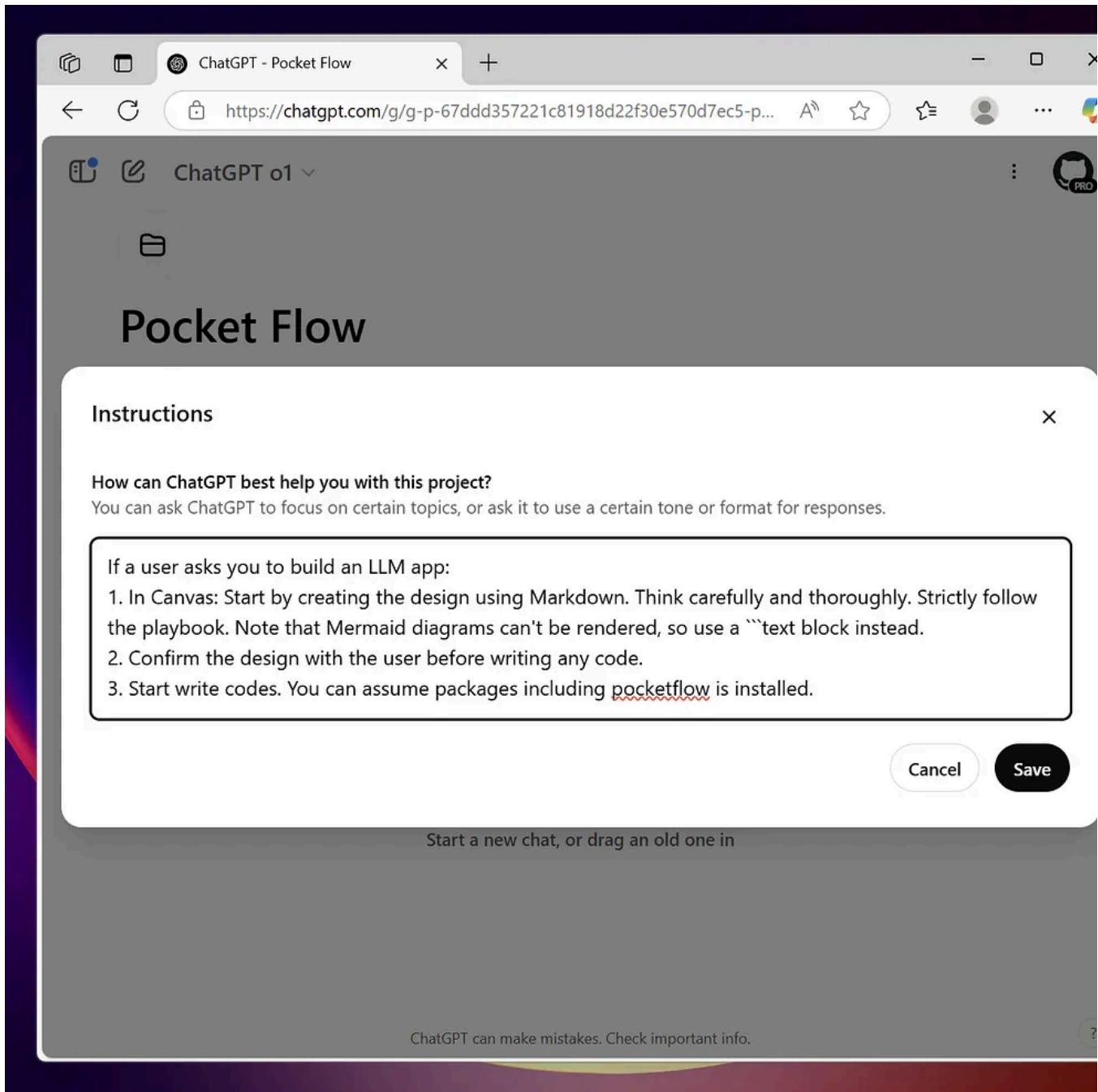
1. In Canvas: Start by creating the design using Markdown. Think carefully and the Mermaid diagrams
2. Confirm the design.
3. Start writing installed.

**Looks like an article worth saving!**

Option Q

Remind me later

Hide Forever



4. Start a conversation! For example: “Help me build a chatbot for a directory of P1

- For the best result, please choose the best model like O1.
- It is highly recommended to enable [Canvas](#)

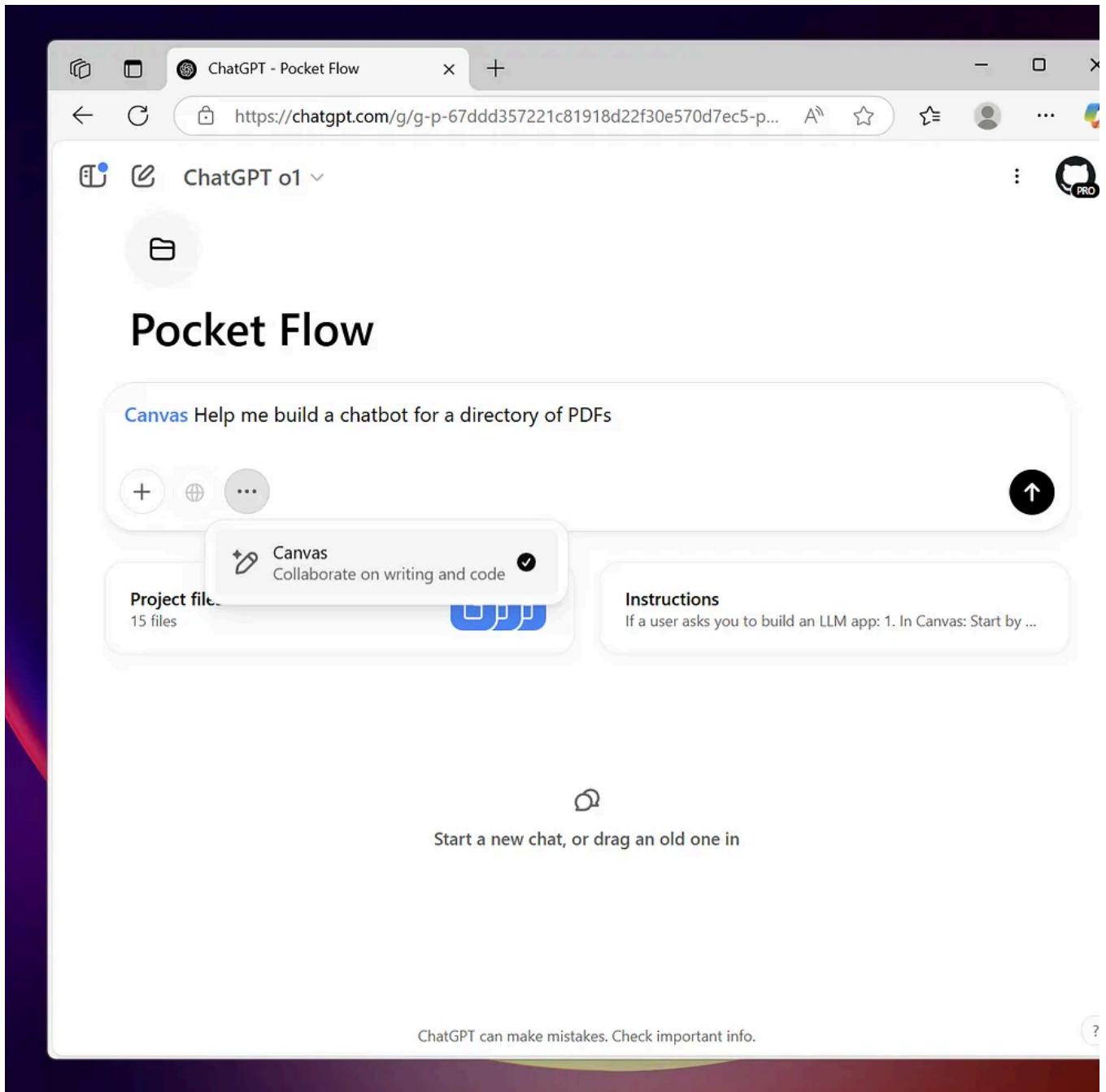
**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



5. Verify and improve the high-level system design, then ask it to implement the c

**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever

The screenshot shows a web-based development environment. On the left, a sidebar titled "Canvas" contains a message: "Help me build a chatbot for a directory of pdf". Below it, a message says "Reasoned about chatbot for directory management for a second" and a button labeled "Design For Chatbot Directory". A message from the bot follows: "Here is the high-level system design for the chatbot. Let me know if this meets your needs or if you'd like me to refine any parts before writing code." Another message from the user: "Great! Help me implement it!". The right side of the screen is a code editor with Python code for a "Chatbot Implementation". The code defines several functions and classes, including `post`, `OfflineFlow`, `OnlineFlow`, and `main`. The code is well-structured with comments and imports.

```

199
200     def post(self, shared, prep_res, answer):
201         shared["answer"] = answer
202         return "default"
203
204 ##### # 4. Building the Flows #####
205
206 # OfflineFlow:
207 loader = LoadPDFs()
208 chunker = ChunkDocs()
209 embedder = EmbedDocs()
210 indexer = StoreIndex()
211
212 loader >> chunker >> embedder >> indexer
213 OfflineFlow = Flow(start=loader)
214
215 # OnlineFlow:
216 query_embedder = EmbedQuery()
217 retriever = RetrieveDocs()
218 answerer = GenerateAnswer()
219
220 query_embedder >> retriever >> answerer
221 OnlineFlow = Flow(start=query_embedder)
222
223 ##### # 5. Main Script Example #####
224
225 # 1) gather pdf files (for demo, we'll just use place
226 pdf_files = ["doc1.pdf", "doc2.pdf"]
227
228
229 def main():
230     import sys
231
232     # 1) gather pdf files (for demo, we'll just use place
233     pdf_files = ["doc1.pdf", "doc2.pdf"]

```

## With Claude:

1. Head to [Claude](#) and start a new [Project](#)
2. Feed it knowledge by uploading the md files under [Pocket Flow docs](#). Please exclude `_config.yml`, `utility_function/index.md`, `design_pattern/index.md`, and `utility_function/index.md`.

**Looks like an article worth saving!**

Option



Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever

The screenshot shows the Claude AI project interface. At the top, there's a header with icons for file operations, a title 'Claude', and a URL 'https://claude.ai/project/b2a8bae7-6626-49a...'. Below the header, a banner says 'START A CHAT TO KEEP CONVERSATIONS ORGANIZED AND RE-USE PROJECT KNOWLEDGE.' On the left, a sidebar has a 'All projects' section. The main area is titled 'Project knowledge' and contains a 'Set project instructions' field labeled 'Optional'. Below this, it says '11% of knowledge capacity used'. There are five rows of cards, each representing a file: 'index.md' (70 lines), 'guide.md' (227 lines), 'communication.md' (127 lines), 'batch.md' (107 lines), 'async.md' (55 lines); 'parallel.md' (56 lines), 'node.md' (105 lines), 'flow.md' (178 lines), 'agent.md' (151 lines), 'workflow.md' (53 lines); 'structure.md' (114 lines), 'rag.md' (161 lines), 'multi\_agent.md' (186 lines), 'mapreduce.md' (73 lines), 'ilm.md' (146 lines). Each card has an 'MD' button at the bottom.

### 3. Set the system prompt to be something like:

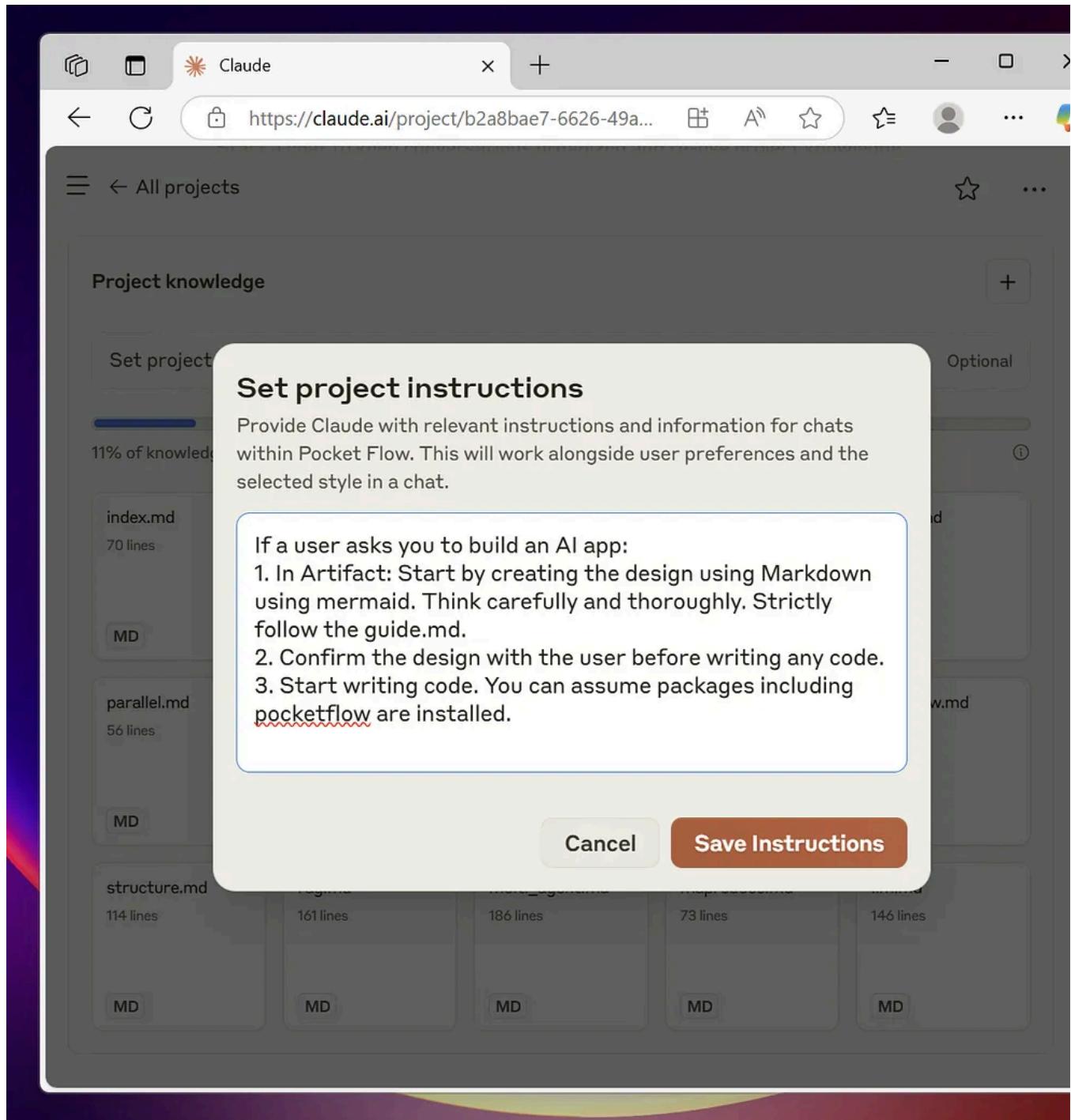
If a user asks yo  
1. In Artifact: 'Looks like an article worth saving!'  
mermaid. Think ca  
2. Confirm the de  
3. Start writing  
installed.

**Looks like an article worth saving!**

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



4. Start a conversation! For example: “Help me build a chatbot for a directory of P1  
For the best result, please choose the best model like Sonnet-3.7 with extended  
thinking.

**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever

The screenshot shows the Claude AI interface. At the top, there's a toolbar with icons for file operations, a search bar containing "Claude", and a tab for "Claude". Below the toolbar is a navigation bar with links like "All projects" and "Pocket Flow". A "Private" button is also present. The main content area has a heading "Help me build a chatbot for a directory of PDFs". On the left, there's a sidebar titled "Claude 3.7 Sonnet" with a dropdown menu. The menu includes options like "Claude 3.7 Sonnet" (selected), "Claude 3.5 Haiku" (Fastest model for daily tasks), "Thinking mode", "Normal" (Best for most use cases), "Extended" (Best for math and coding challenges), and "More models". A note about "Extended" says "Claude will think for longer before responding. This may use your rate limits faster". Below the sidebar, it says "11% of knowledge capacity used". At the bottom, there are five file tabs: "index.md", "guide.md", "communication.md", "batch.md", and "async.md".

5. Verify and improve the high-level system design, then ask it to implement the code.

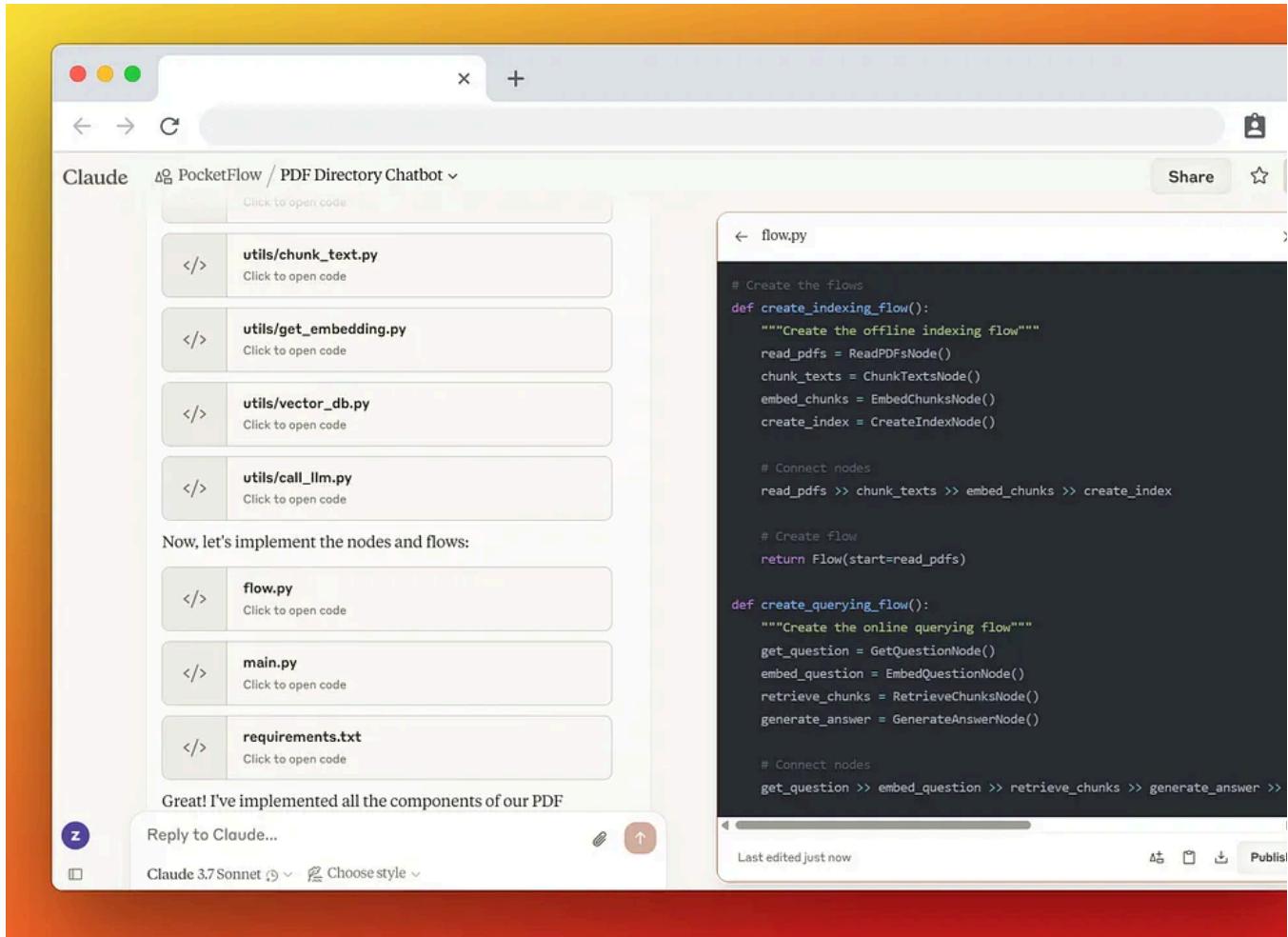
**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



# For Serious Development: The Full Agentic Coding Experience

Ready to build a production-ready application? This is where the real magic of Ag Coding happens:

## Cursor AI:

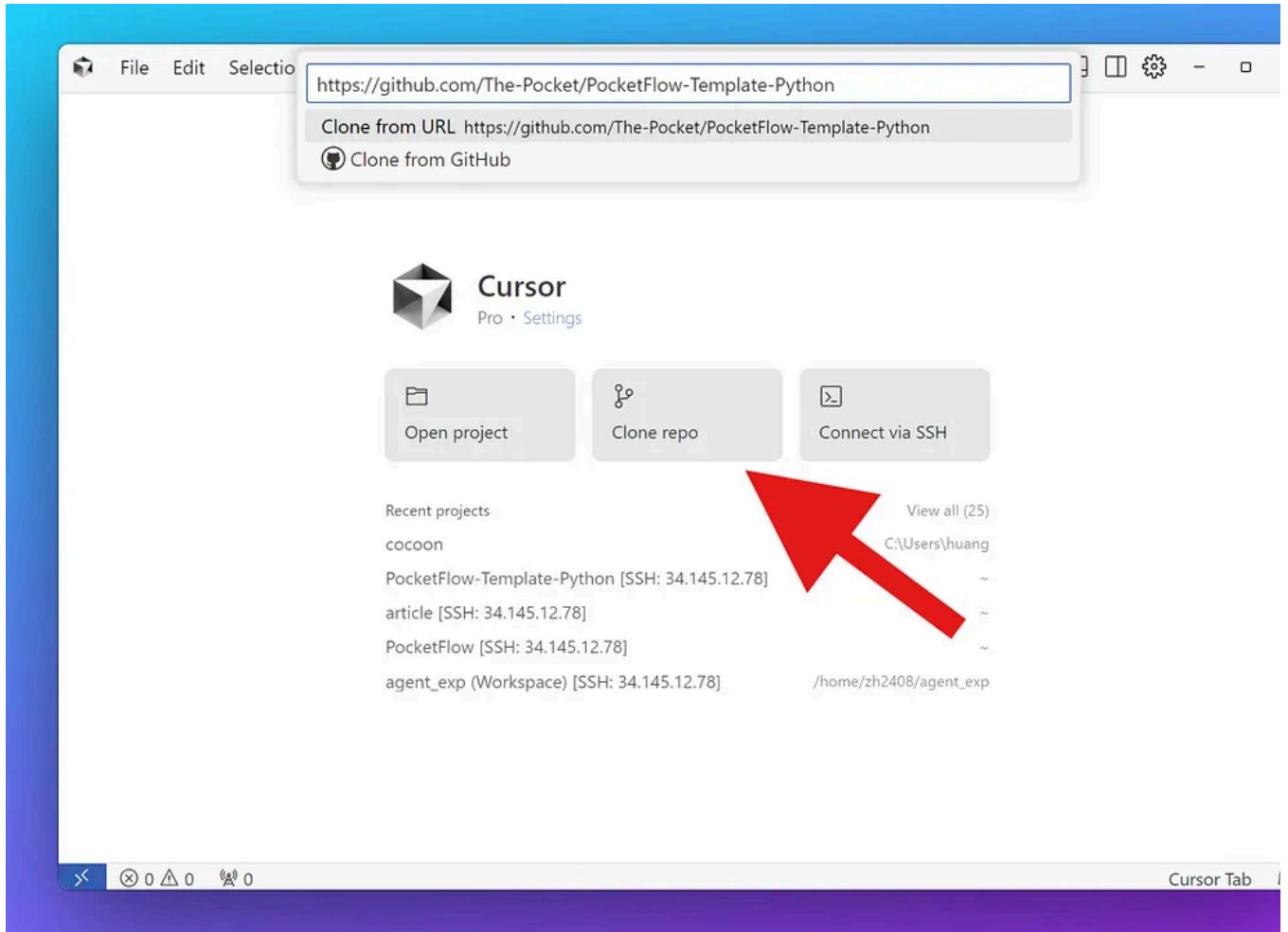
1. Create a New Window, and click on the Clone Repo.
2. Enter the URL for the GitHub repository.

<https://github.com> Looks like an article worth saving!

This template in work with Pocke Hover over the brain icon or use hotkeys to save with Memex. w t

Remind me later

Hide Forever



3. You can verify the setup by asking Cursor AI: “What is Pocket Flow?”

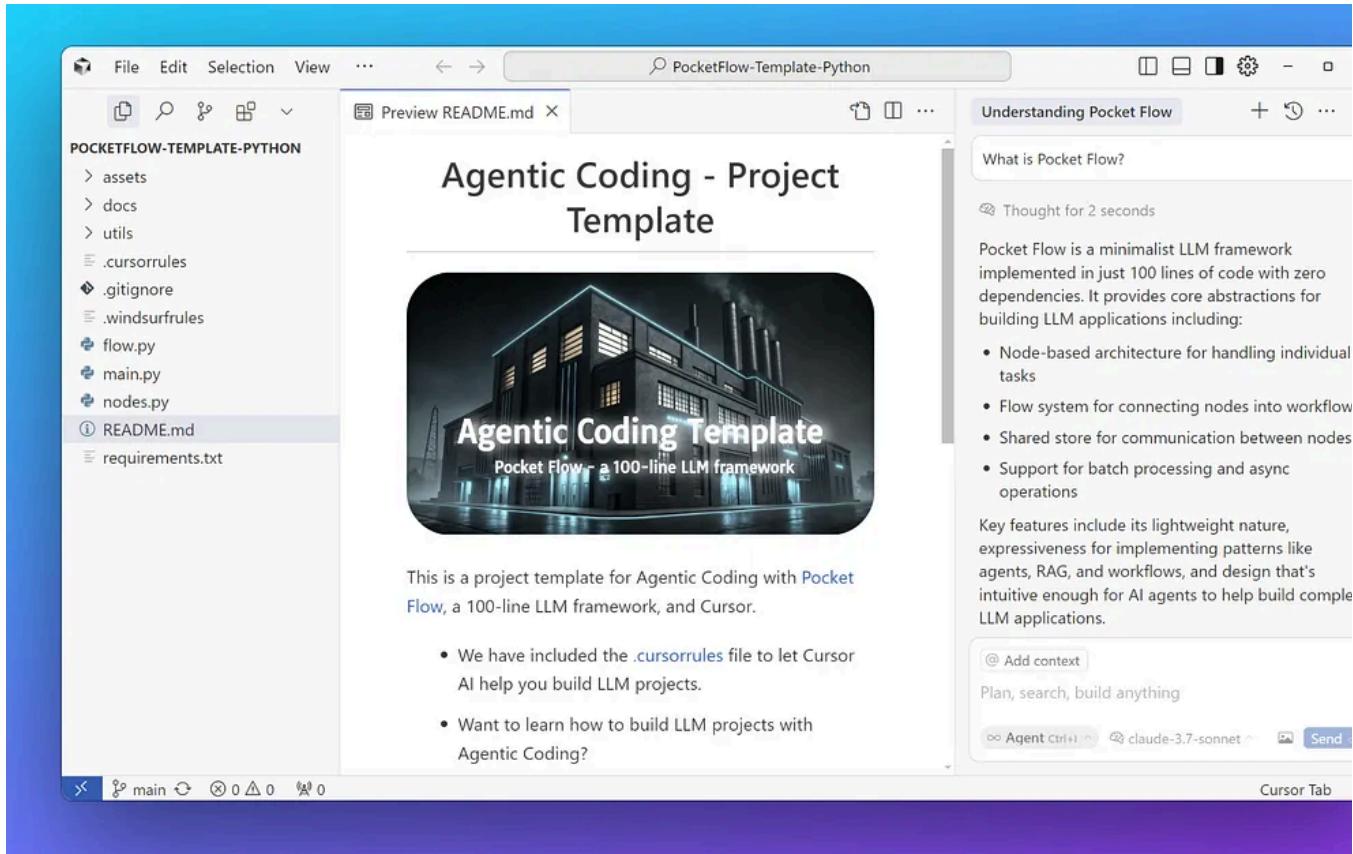
**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



## 4. Start building LLM systems!

**Pro Tip:** Use **Chat** mode during the design phase to avoid the AI getting ahead of itself, then switch to **Agent** mode during implementation.

## Windsurf:

1. Create a **New Window**, click on the **Explorer** side bar icon, then **Clone Repository**.
2. Enter the URL for the ready-to-use Pocket Flow Template:

<https://github.com/The-Pocket/PocketFlow-Template-Python>.

This template includes a special [.windsurfrules](#) file that teaches Windsurf how to work with Pocket Flow.

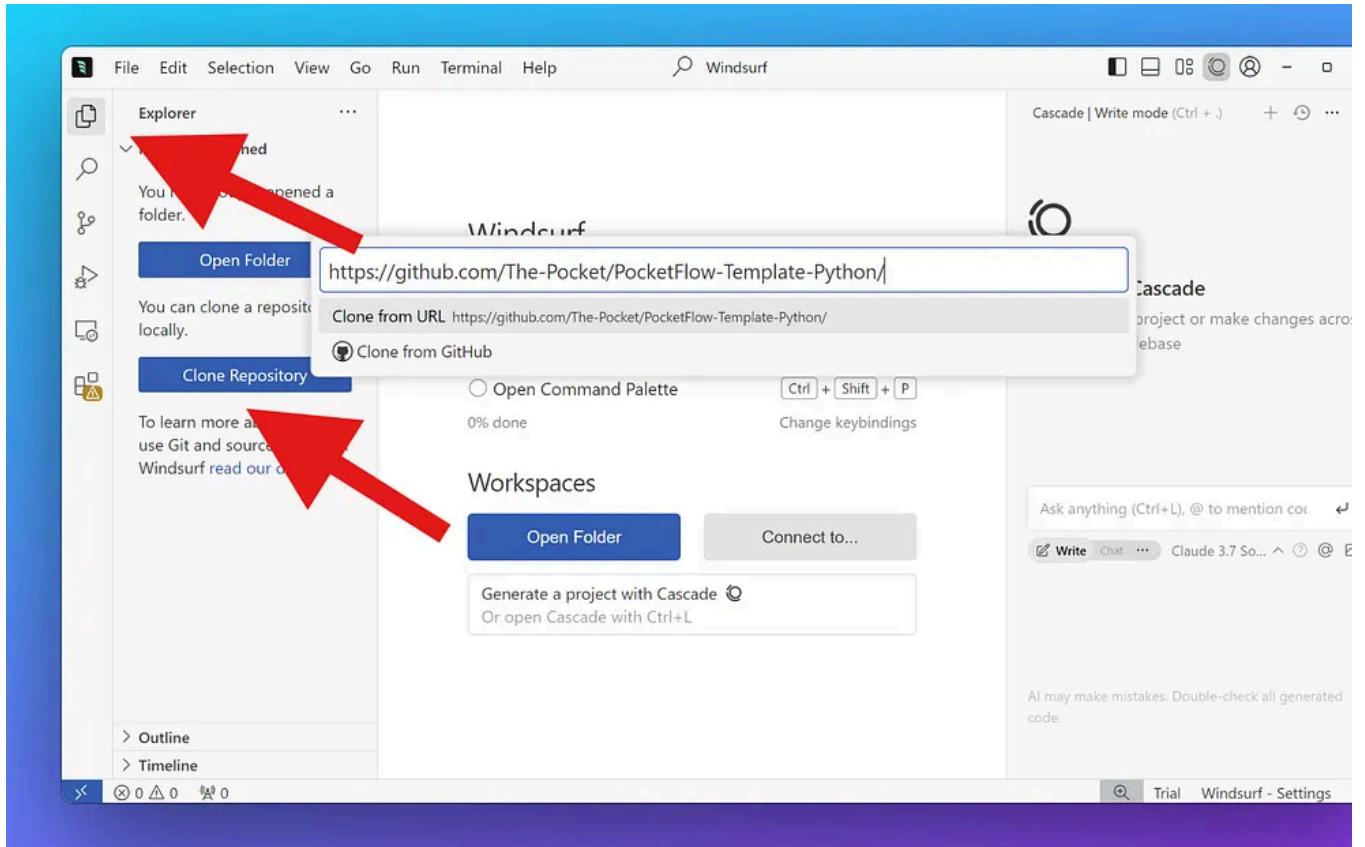
**Looks like an article worth saving!**

Option Q

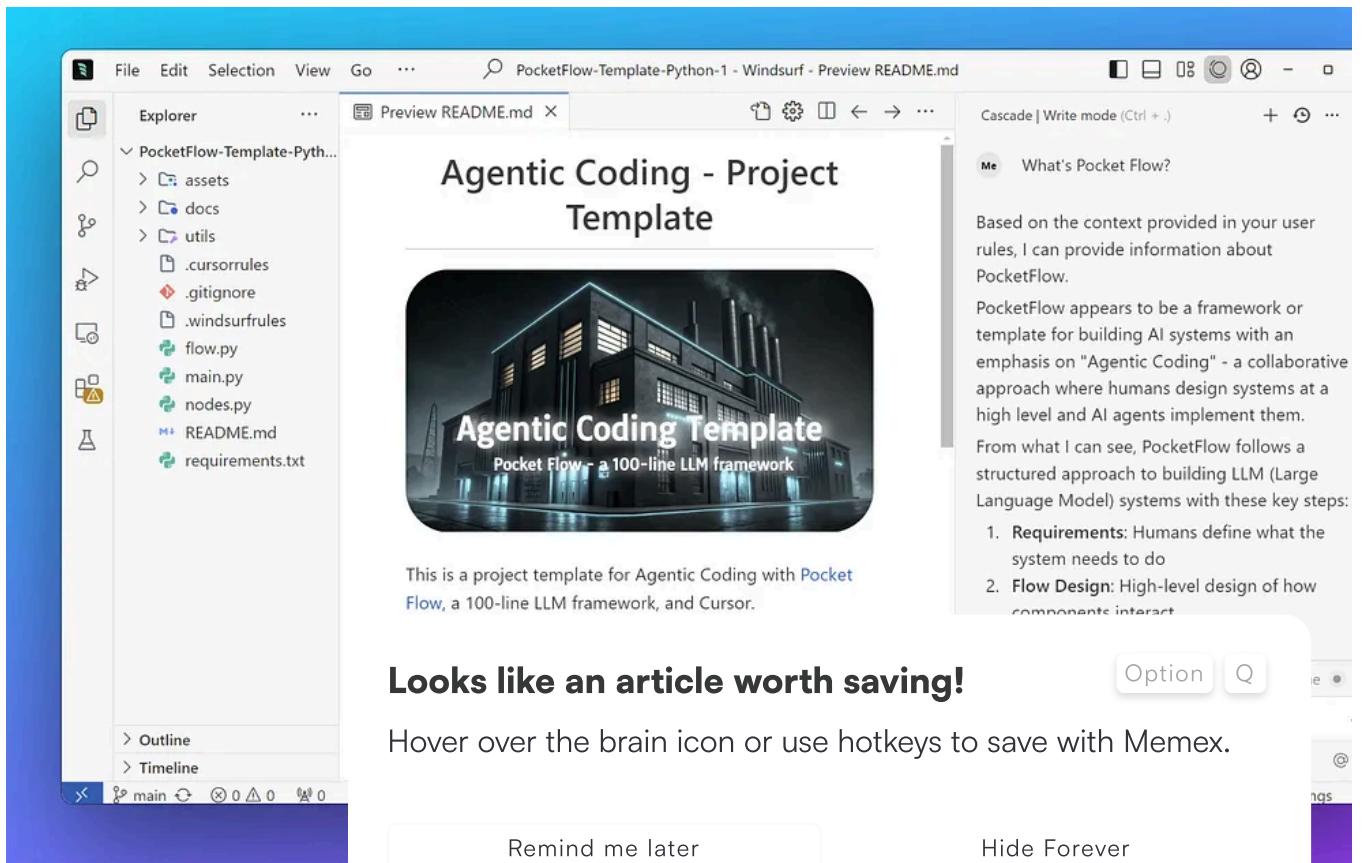
Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever



### 3. You can verify the setup by asking Windsurf AI: “What is Pocket Flow?”



## 4. Start building LLM systems!

Pro Tip: Use **Chat** mode during the design phase to avoid the AI getting ahead of itself, then switch to **Write** mode during implementation.

### Cline:

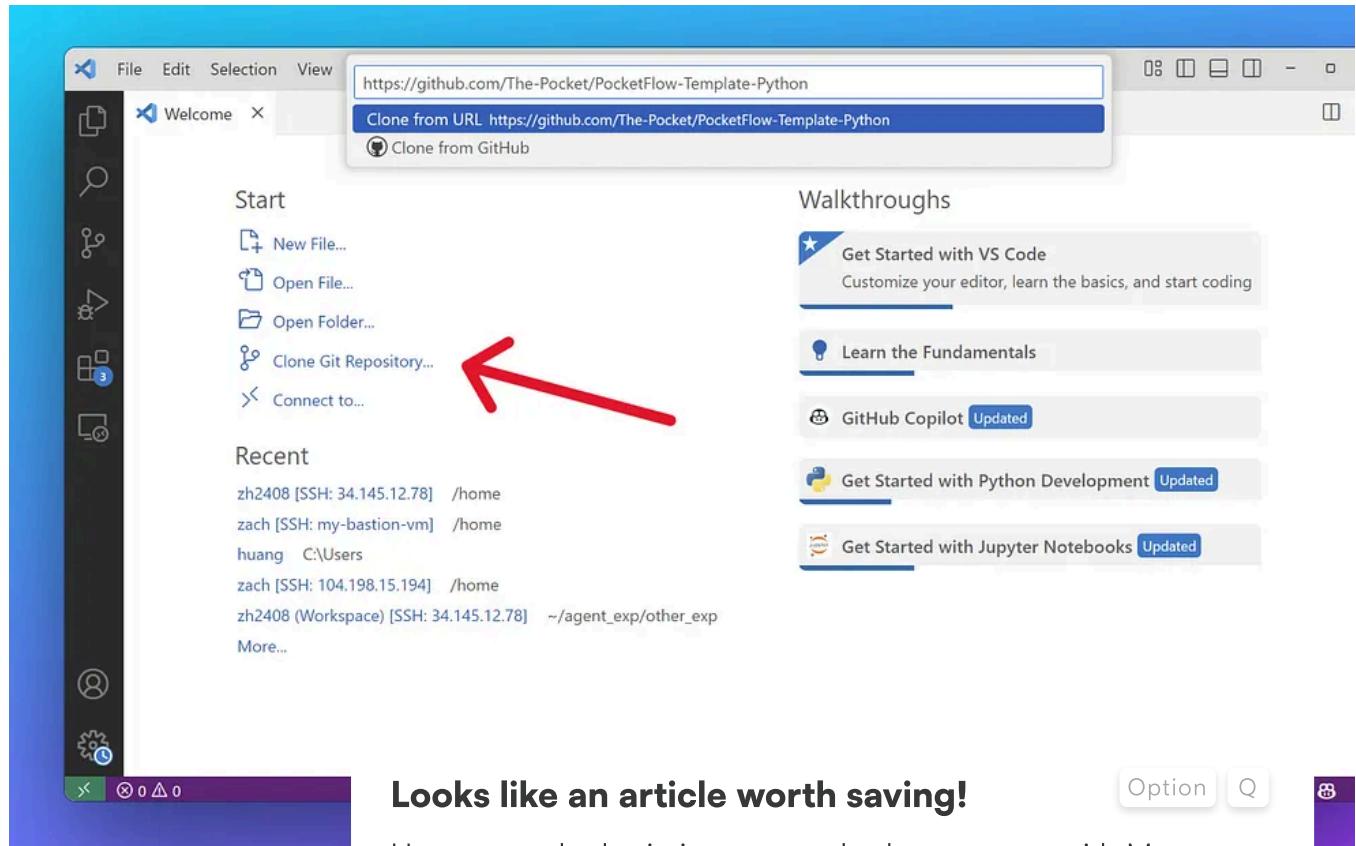
**Note:** As of March 2025, Cline's Agent doesn't work that well with rule files. This section will be updated as compatibility improves.

1. Create a New Window, and click on the Clone Git Repository....

2. Enter the URL for the ready-to-use Pocket Flow Template:

<https://github.com/The-Pocket/PocketFlow-Template-Python>.

This template includes a special `.clinerules` file that teaches Cline how to work with Pocket Flow

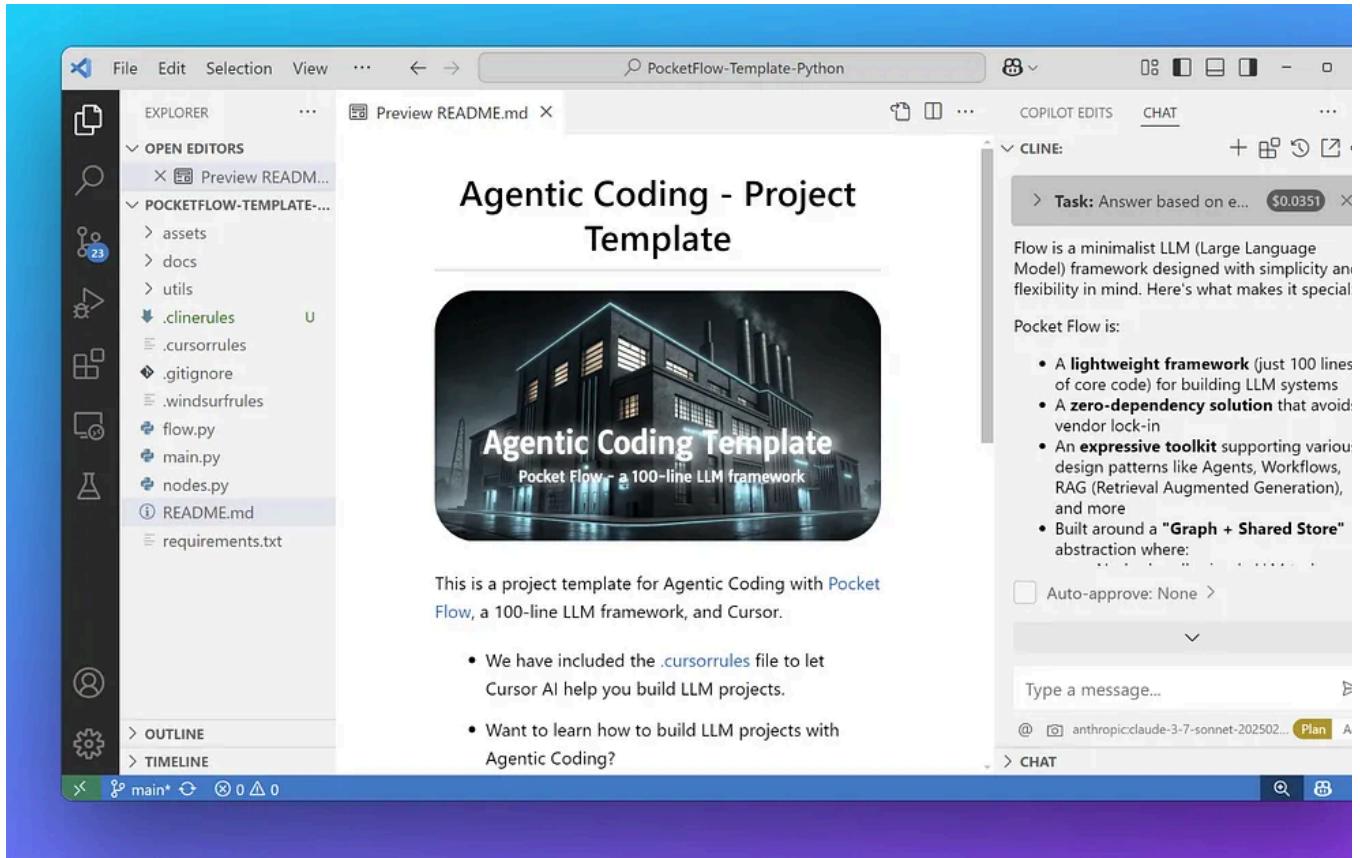


3. You can verify the What's Pocket Flow?

[Remind me later](#)

[Hide Forever](#)

let



#### 4. Start building LLM systems!

**Pro Tip:** Use **Chat** mode during the design phase to avoid the AI getting ahead of itself, then switch to **Write** mode during implementation.

## What's Next?

Once you've set up your environment, the next step is understanding the Agentic Coding workflow.

More tutorials coming soon! Stay tuned! In the meanwhile, check out these resources:

- [Agentic Coding Guidance](#) — Official documentation
- [The Pocket Gif](#) **Looks like an article worth saving!** Hover over the brain icon or use hotkeys to save with Memex.
- [Pocket Flow You](#)

[Remind me later](#)

[Hide Forever](#)

For more examples and detailed documentation, visit the [Pocket Flow GitHub](#).

Thanks for reading Pocket Flow! Subscribe for free to receive new posts and support my work.



45 Likes

 [Previous](#)[Next](#) 

## Discussion about this post

[Comments](#)   [Restacks](#)


Write a comment...



Solaris Mar 22 Edited

heart icon Liked by Zachary Huang

Pocket Flow examples design documents are really detailed. As a software engineer, writing an important best practice that helps understand and consider the implementation thorough this approach is especially needed for critical thinking / productive reviews of what the LLM developing, so its great to see them as part of Pocket's workflow. Thank you for the excellen


heart icon LIKE (2)   comment icon REPLY

1 reply by Zachary Huang

### Looks like an article worth saving!

[Option](#)   [Q](#)


G K May 11

heart icon Liked by Zachary

Hover over the brain icon or use hotkeys to save with Memex.

Hey, great post ☺

Remind me later

Hide Forever

I just started to work with pocketflow and it seems to be really easy to maintain consistency.

I do have a quick question, though:

How does Pocketflow differ from LangGraph, aside from LangGraph having extra features like LangGraph Platform?

Other than being easy to maintain, what are some other advantages of using Pocketflow over

 LIKE (1)  REPLY

2 replies by Zachary Huang and others

10 more comments...

---

© 2025 Zachary Huang · [Privacy](#) · [Terms](#) · [Collection notice](#)  
[Substack](#) is the home for great culture

**Looks like an article worth saving!**

Option Q

Hover over the brain icon or use hotkeys to save with Memex.

Remind me later

Hide Forever