GitHub

GitHub Copilot

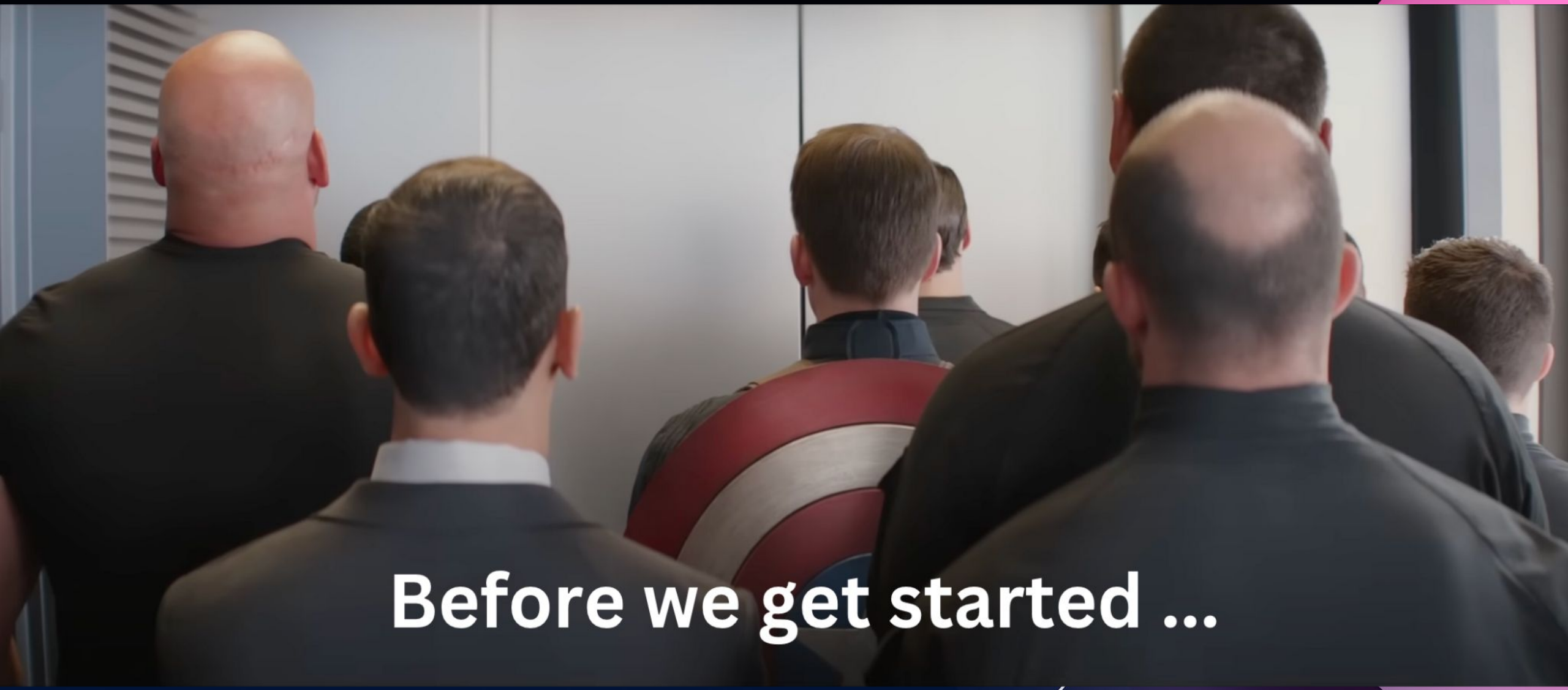
Sukumar P
Github CSA team
https://www.linkedin.com/in/sukumar-p/

Level 100

Let’s boost your productivity with GitHub’s AI-powered developer platform

Agenda

- Copilot flow
- Code completion
- Next Edit Suggestion
- Inline Chat vs Normal Chat
- Slash commands, Chat participants, Context variables
- Custom instructions
- Reusable prompts
- Code-adjacent tasks
- Public-code blocking
- Ask vs Edit
- Copilot voice chat
- Saving chat conversations
- Best Practices
- Model selection
- Debugging with Copilot
- Q&A


Before we get started ...


I would like to know your experience with Copilot
imgflip.com

GitHub Copilot

What is GitHub Copilot?

Increase developer productivity

And satisfaction by focusing on real problems

Accelerate innovation

Prototype & innovate more rapidly

Bridge skill gaps

Learn new languages and techniques

How Copilot Fulfills a Request

Apply several **pre-model filters**

- Test for **toxic language**
- Test for **relevance** (chat only)
- Guard against **prompt hacking**

Copilot assembles context from tabs open in the code editor

OpenAI model

Context / Prompt

N Suggestions

Proxy

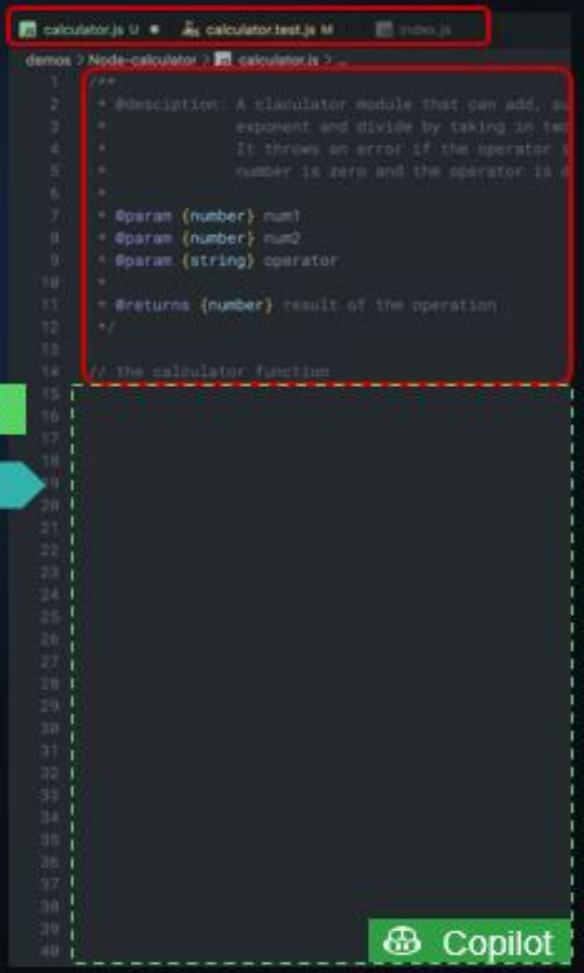
Context / Prompt

N Suggestions

Copilot Agent

Context / Prompt

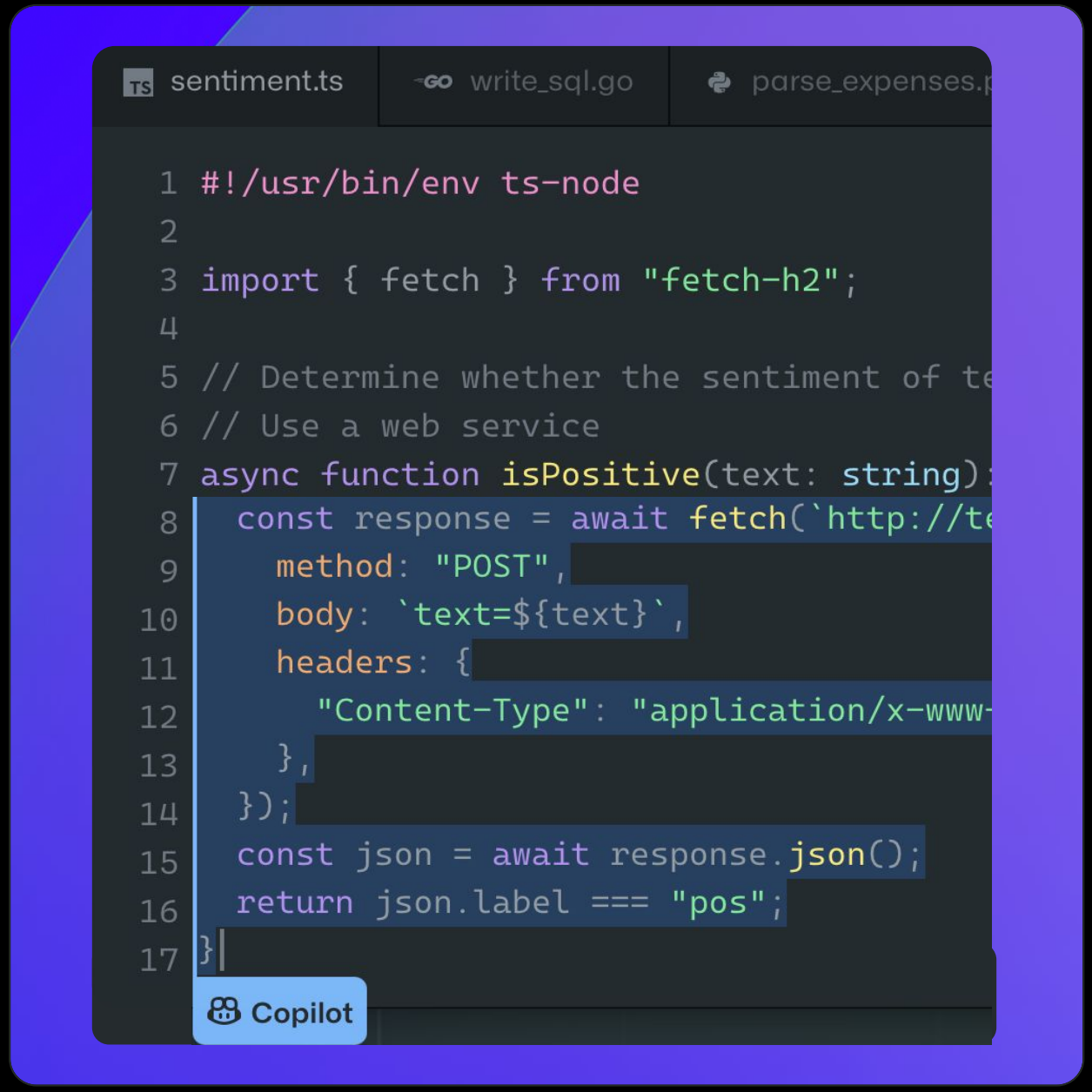
N Suggestions



Apply several **post-model filters**

- **Code quality**
- **Unique Identifiers** (eg.: emails, IPs etc.)
- Suggestion **matching public code**

Code Completion



Code completion

Your AI pair programmer

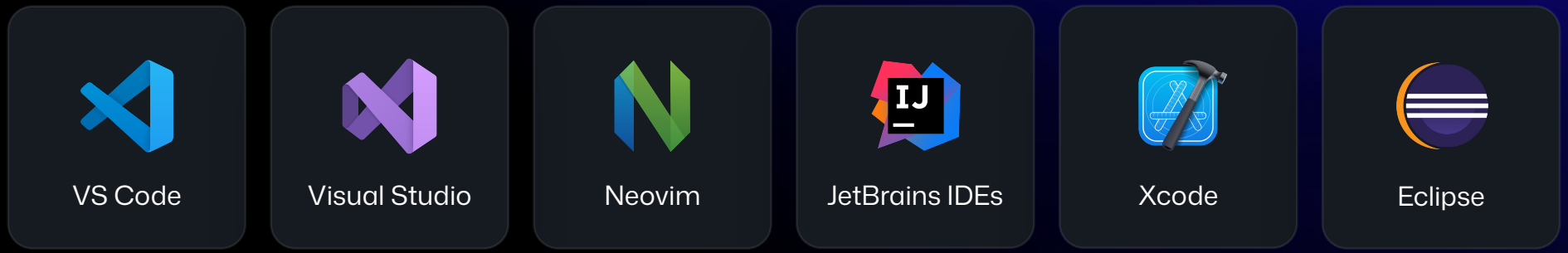
- Real time code suggestions
- Comments to code

GitHub Copilot / Code Generation

	Completions
Scope Of Changes	Next Few Lines
Frequency of Interaction	Hundreds of Milliseconds
Inner/Outer Loop	Inner Loop
Developer Canvas	VS Code (Editor)

Code Completion

Keep flying in your favorite editor



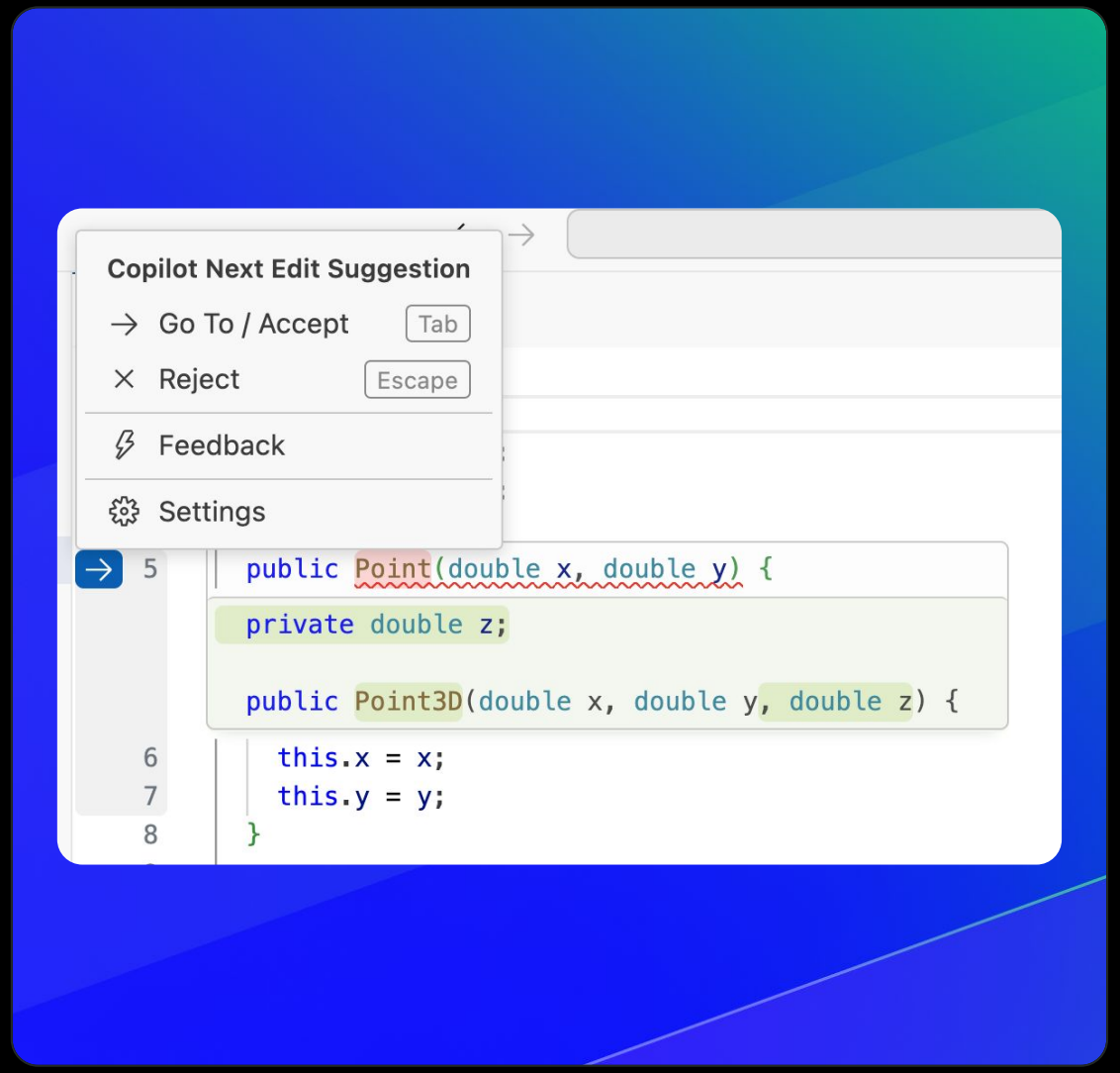
GA

Next Edit Suggestions

Predicts where the next edit should be

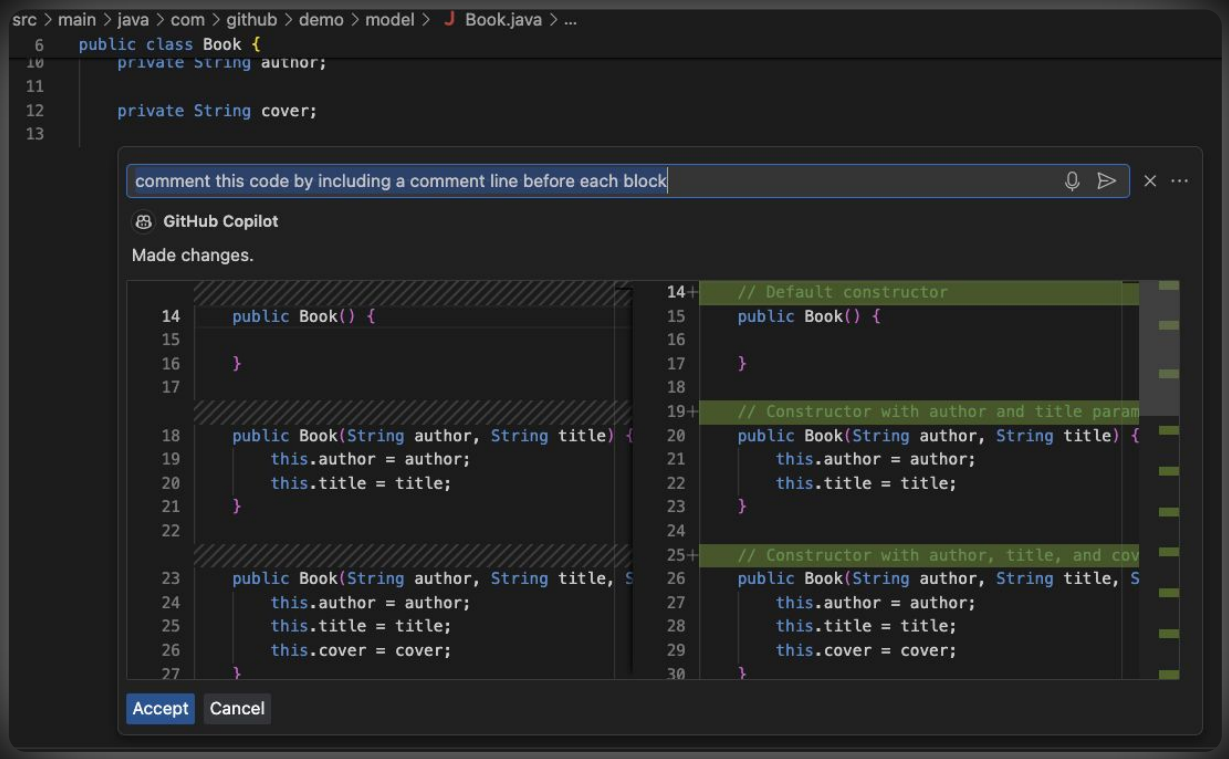
Based on the edits you're making, Copilot NES both predicts the location of the next edit you'll want to make and what that edit should be.

VS Code



GitHub Copilot Inline Chat

Quickly access Chat inline to ask questions and generate code suggestions



Chat Experience

You can use shortcuts to code faster and better (context awareness) with:

/ Slash commands

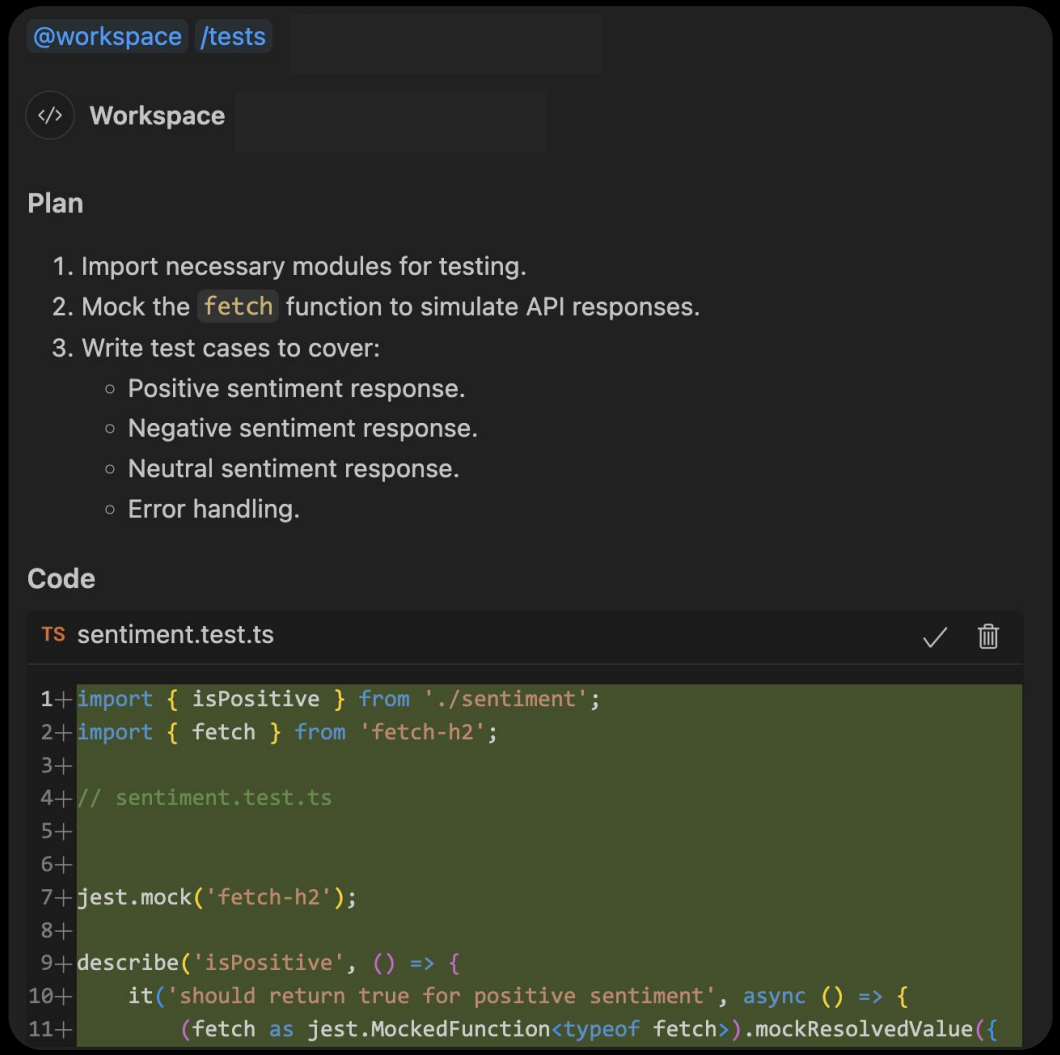
```
/help
/explain
/fix
/tests
/new
/newNotebook
/search
/startDebugging
```

@ Chat participants

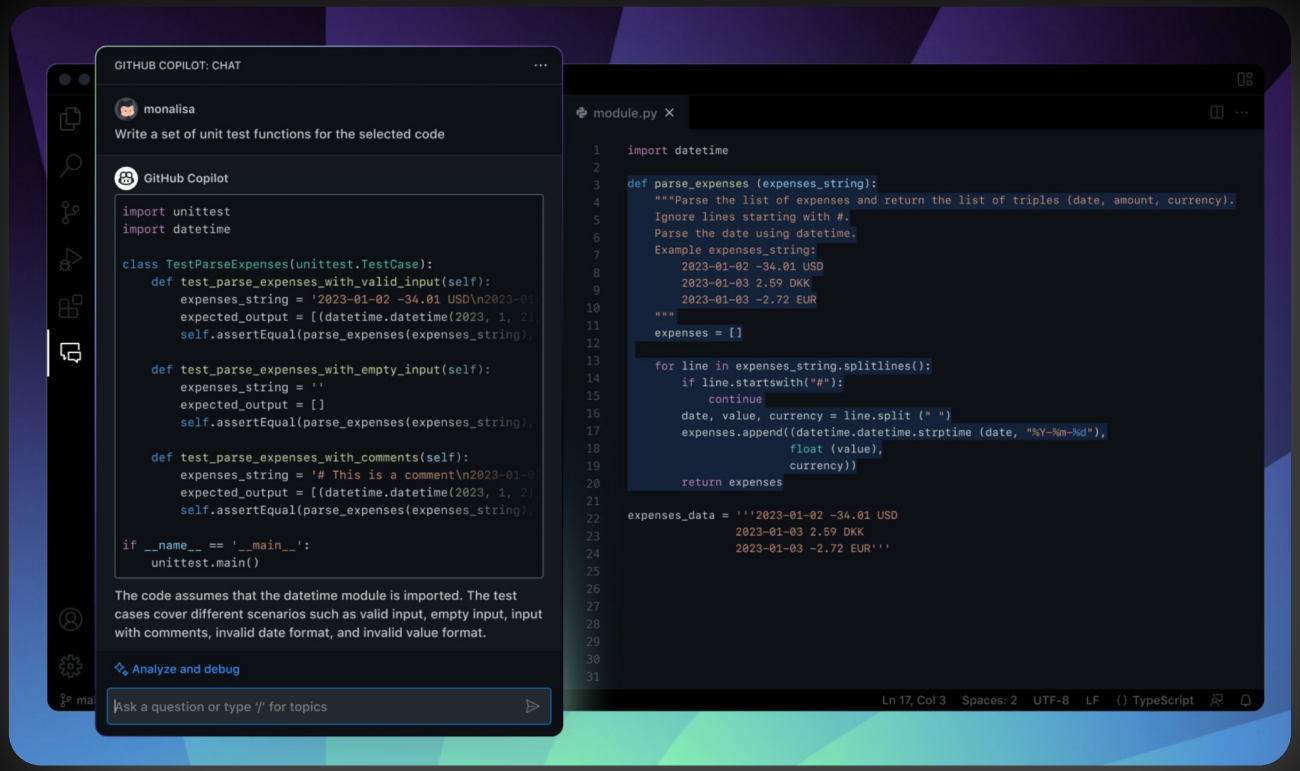
```
@workspace
@terminal
@vscode
@azure
```

 # Context variables

```
#editor
#selection
#[file_name]
#changes
#terminalSelection
#terminalLastCommand
#problems
#kb
```



Chat & Edit



Copilot Chat

Context-aware conversations

- Answer coding questions
- Explain code
- Write unit tests
- Translate languages

GA

Chat / Multi-file edits

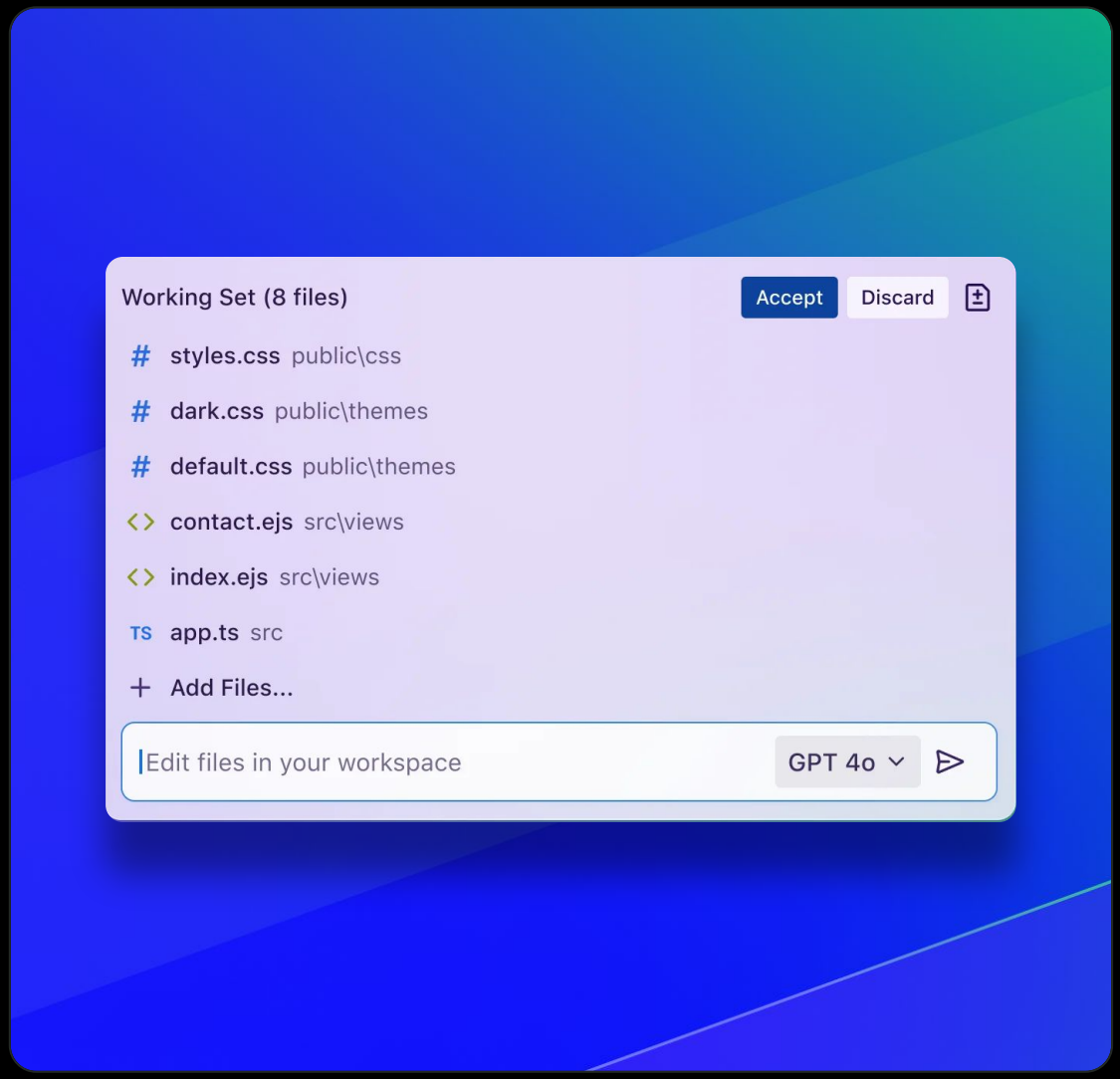
Code across your codebase in a single view

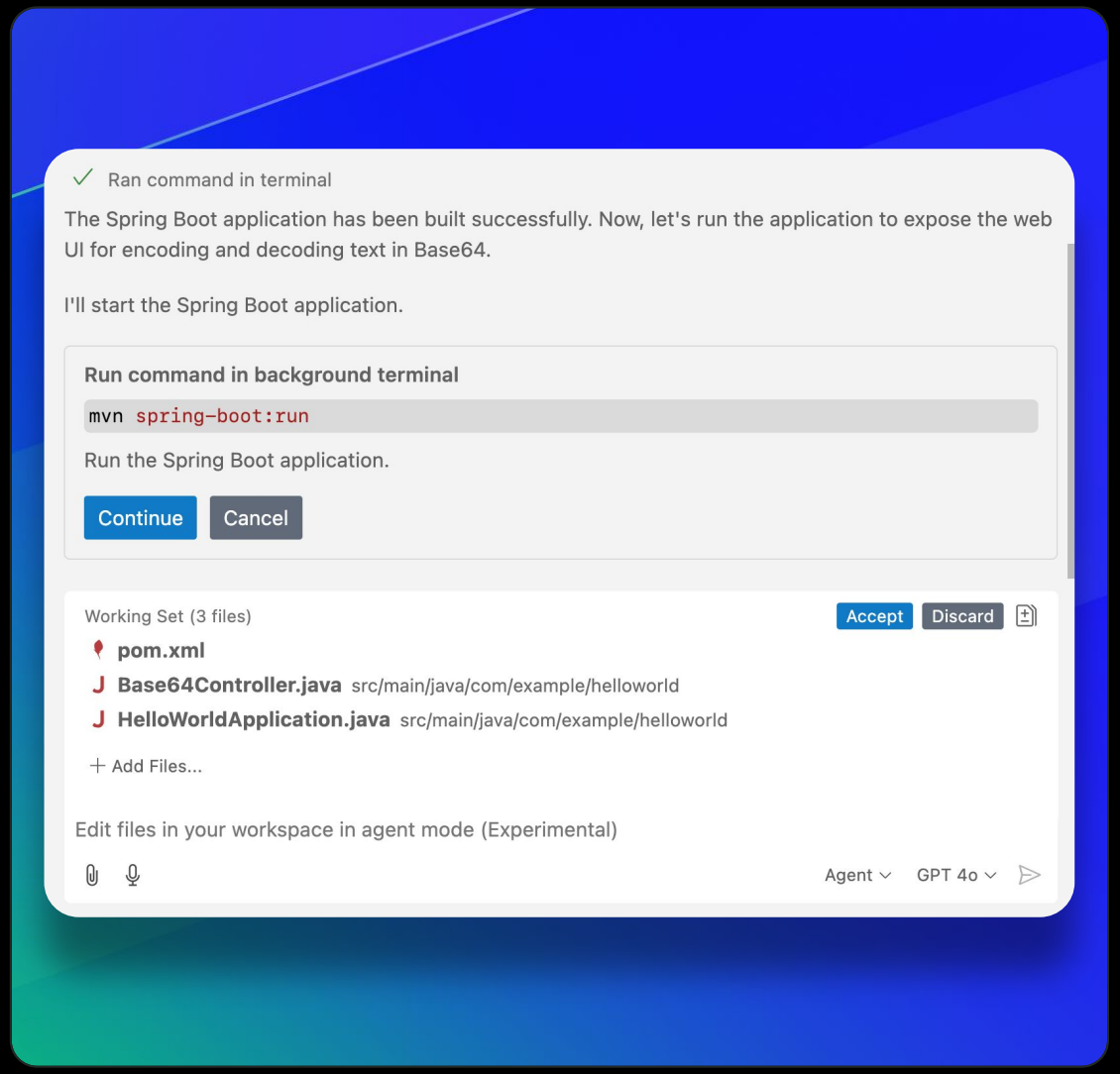
Tell GitHub Copilot what to do, and watch
It updates multiple files right in your editor.

VS Code

VS

JetBrains





GA

Chat / Agent Mode

Agent mode for Copilot Edit

File & codesearch tools for discovery
Edits files & iterates on errors/tests
Terminal execution and follow-up fixes

VS Code

VS

JetBrains

GA

PREVIEW - JUNE 2025

AGENTIC FUTURE

What is Agent mode ?

Autonomous Pair Programmer

🔄 Runs Copilot in **“plan → act → observe → iterate” loop** on your workspace.

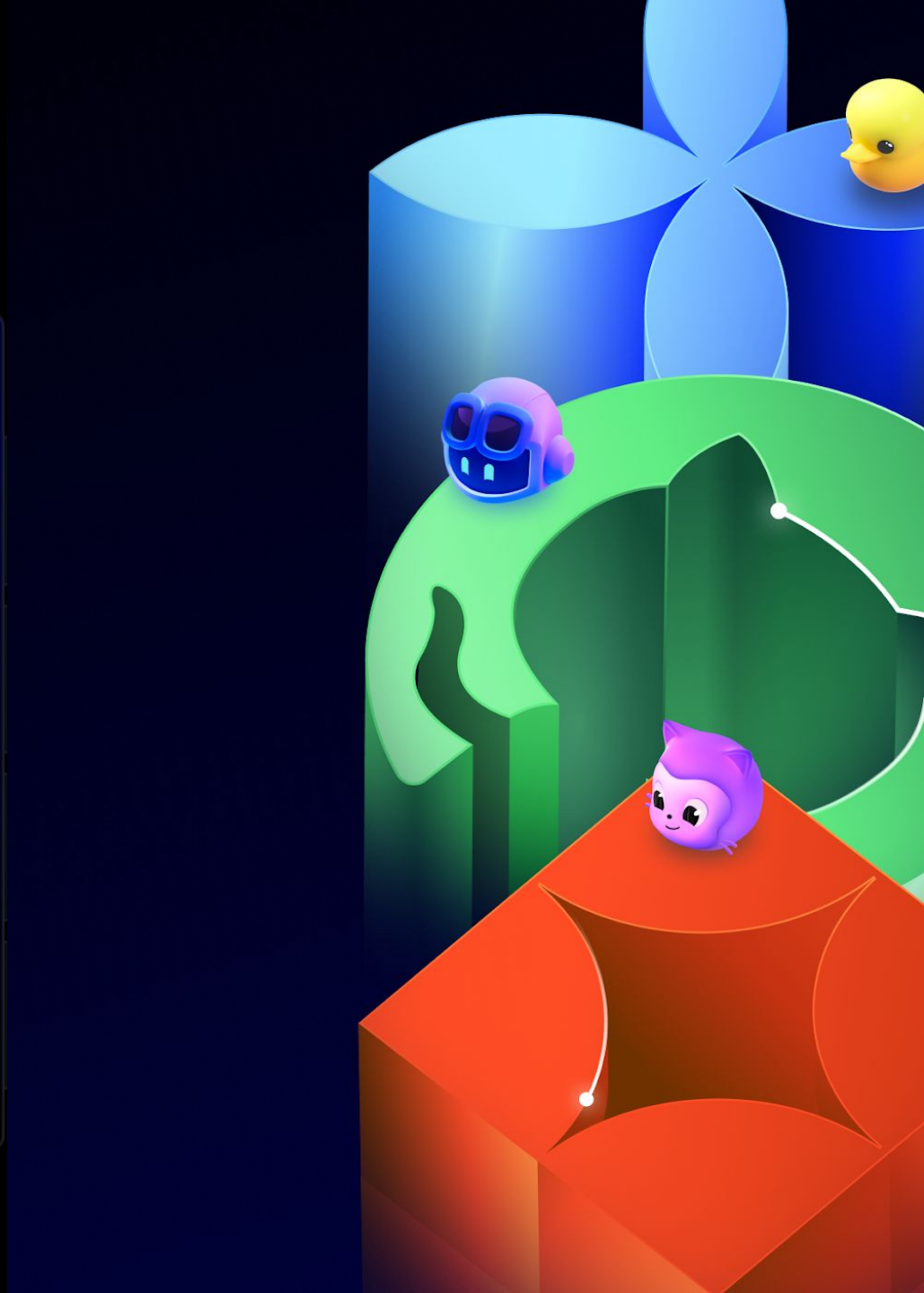
✏️ Can **read, edit, run, and test code** across multiple files without manual steps.

🧠 Acts like an **eager junior dev** with an infinite knowledge

🔗 Fully **integrated** inside supported IDEs; no extra CLI or context-switching.

GitHub Copilot / Code Generation

	Completions	Chat/Edits
Scope Of Changes	Next Few Lines	Multi-File Edits
Frequency of Interaction	Hundreds of Milliseconds	Seconds
Inner/Outer Loop	Inner Loop	Inner Loop
Developer Canvas	VS Code (Editor)	VS Code (Chat)

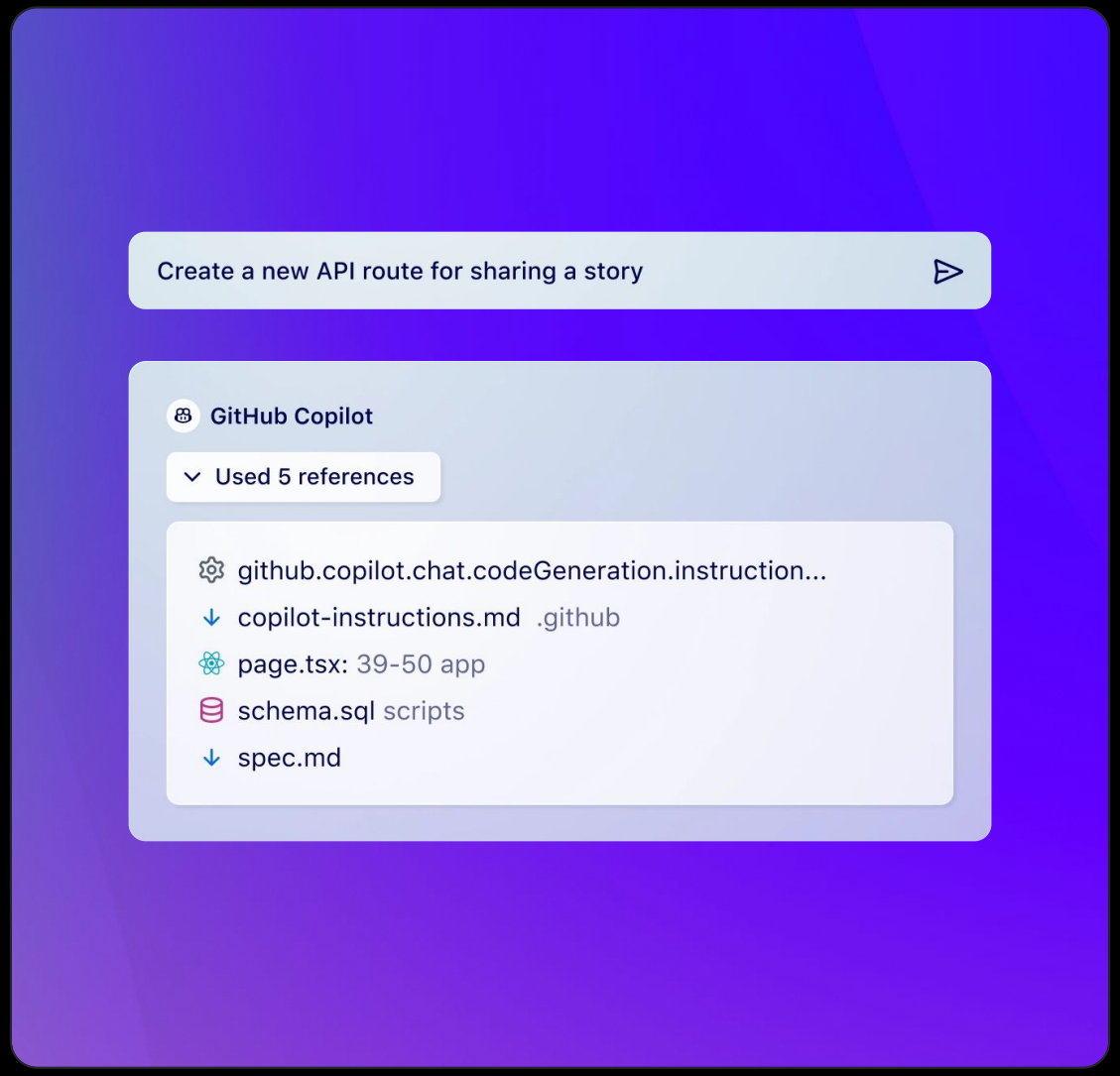


GA

Chat / Custom instructions

Tailor-made answers, defined by you

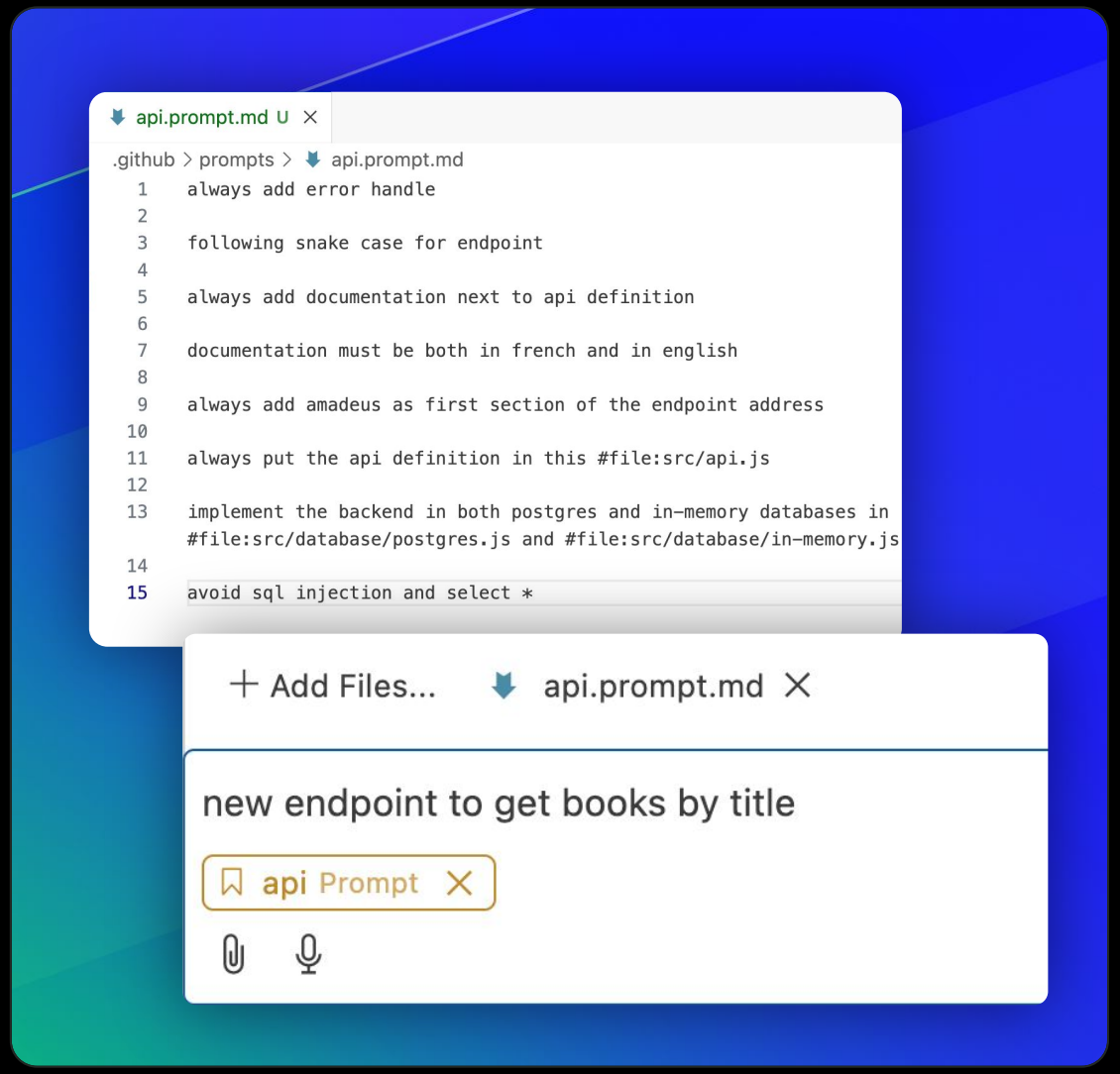
Specify custom instructions to personalize chat responses in VS Code and Visual Studio based on your preferred tools, organizational knowledge, and coding best practices.



copilot-instructions.md vs ***.instructions.md**

- Single file
- Automatically attached in the chat conversation
- Stored in **.github** directory

- Multiple files specific to file type
- To be attached to the chat explicitly
- Stored in **.github/instructions** directory



GA

Chat / Reusable prompts

Build and share reusable prompt instructions with additional context.

Copilot for code-adjacent tasks

- Writing a CI/CD file
- Writing an IaC file
- Creating Mermaid diagrams

Public code blocking

- Enable/Disable it, IP indemnity still applies

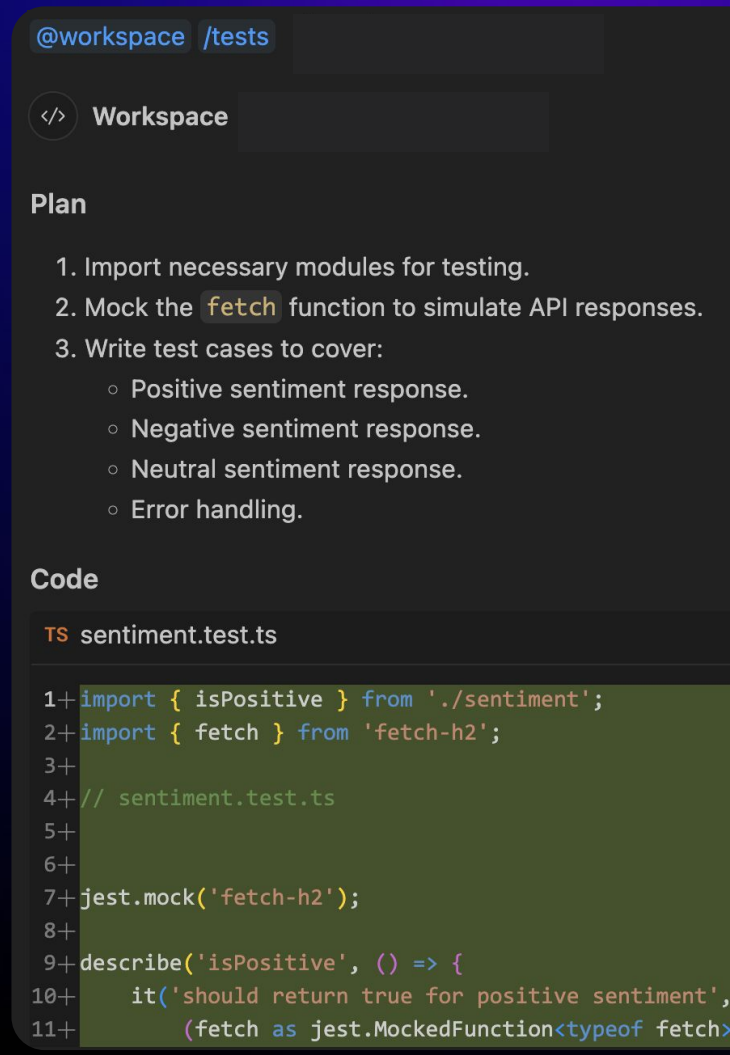
Chat vs Edit Mode

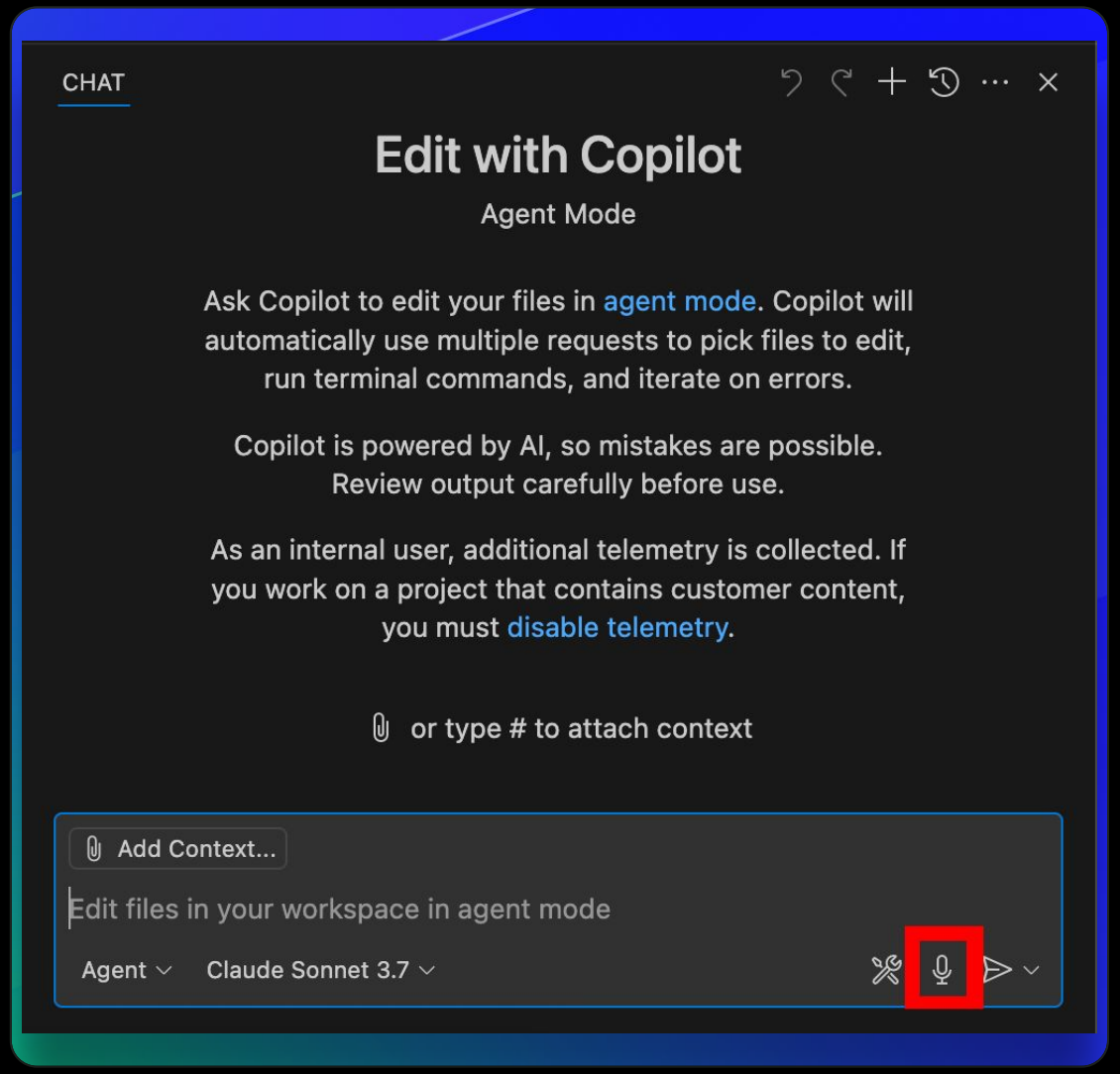
Edit mode: Multi-file edits.

Ask mode: Safe. Just for responses.
Doesn’t change any code directly.

Utilize /slash commands

/explain	/tests	/fix	@workspace	/help
Explains how the selected code works	Generates unit tests for the selected code	Proposes a fix for the problems in the selected code	Preface your prompts/commands to ask a question about the files in your workspace	Learn how to use Copilot Chat





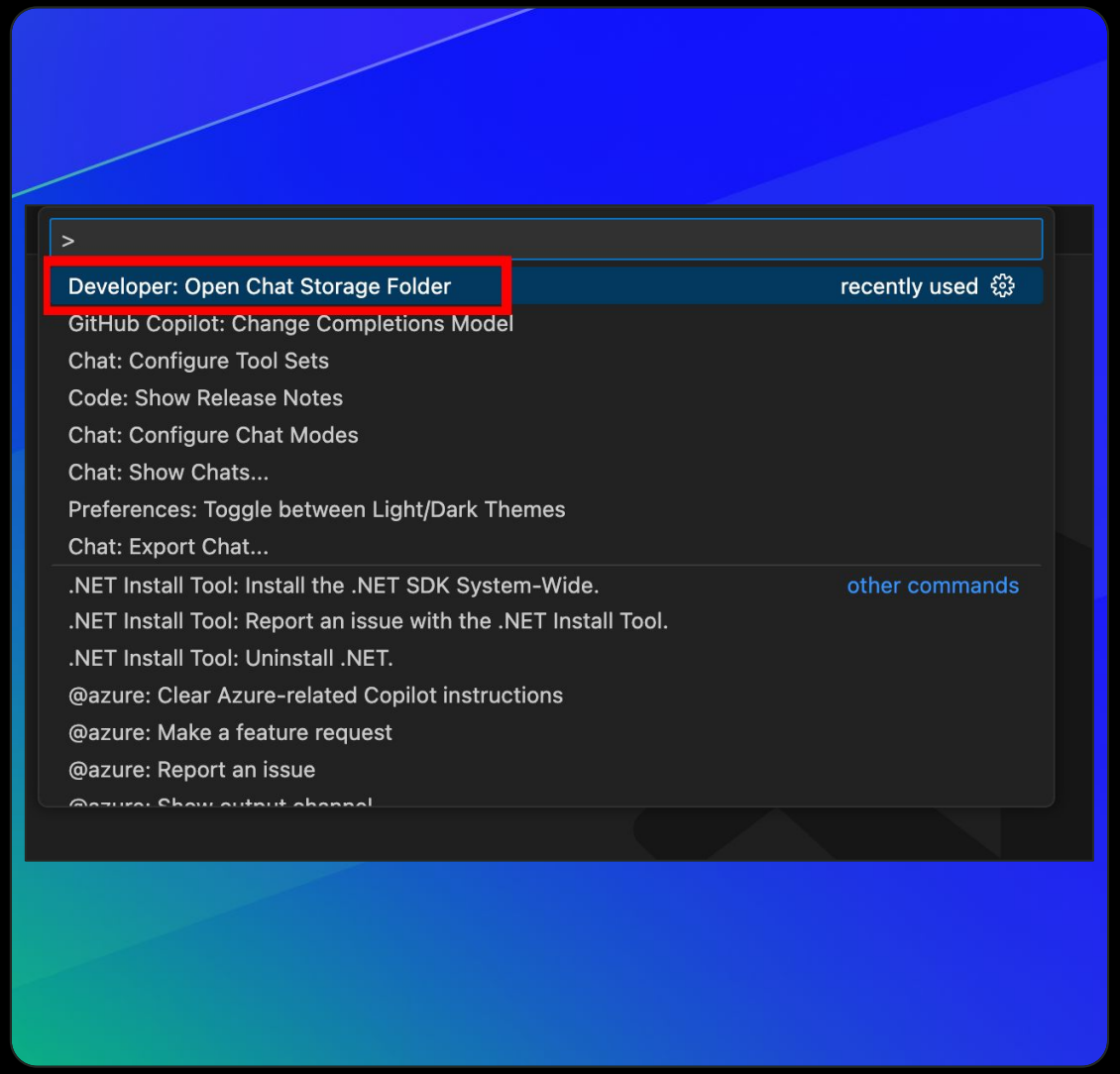
Chat / Voice chat

Talk to Copilot

With the help of VS Code Speech extension, directly talk to copilot chat and convert speech to text

Utilize #hash context variables

#terminalSelection	#new	#problems	#changes	#filename	#codebase
Last command from the active terminal	Scaffold new workspace in VS Code	Check errors for particular files	Get diffs of changed file	File as context to the chat	Codebase as context



Chat

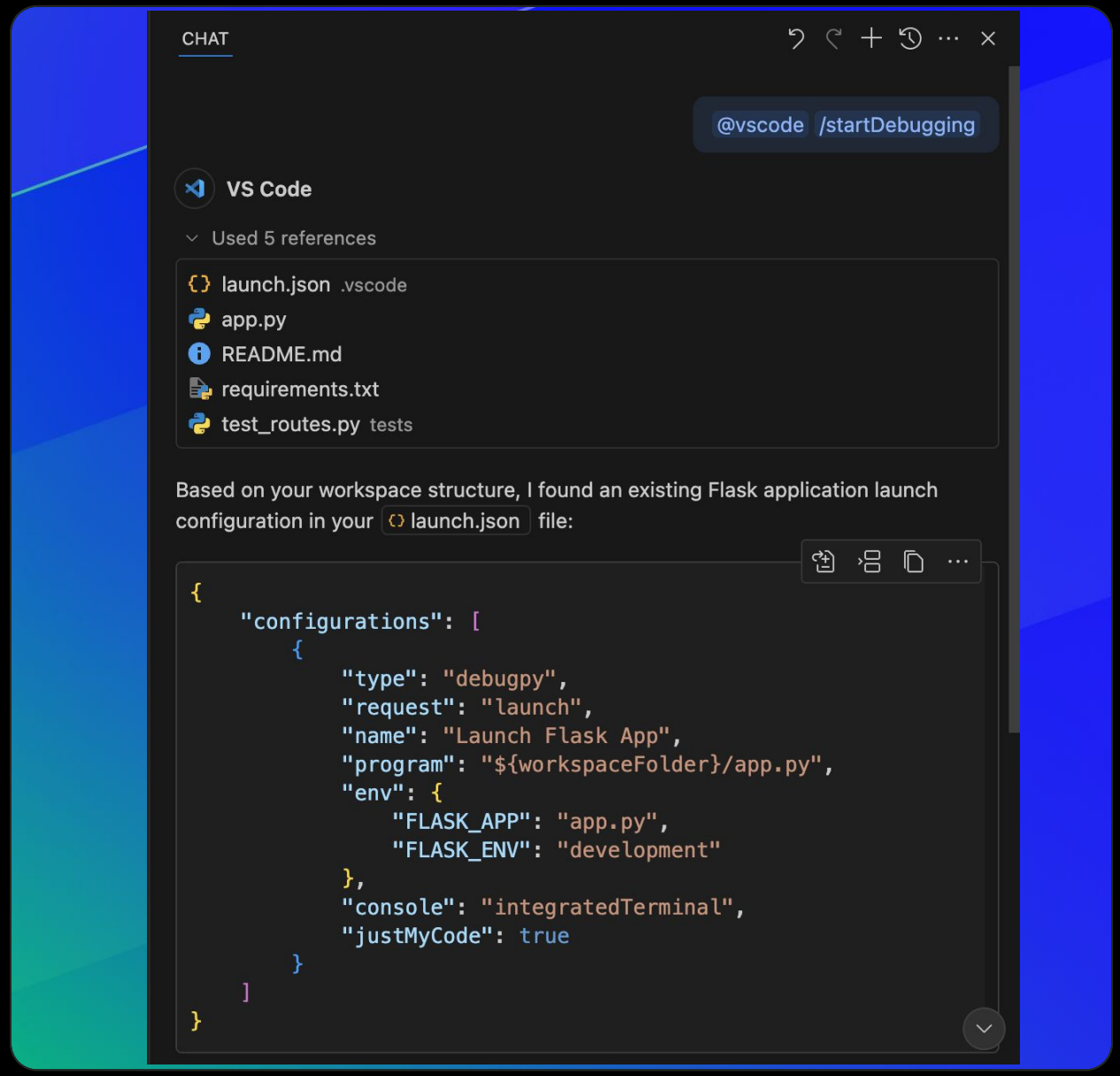
/save - to save the conversation

Chat conversations written to disk as .json.
Can save the conversation in Markdown format

Copilot best practices

- Use the appropriate model for the use case (premium vs base)
- Custom-instructions for all tasks
- Re-usable prompts when needed
- Always review the code generated by Copilot

https://docs.github.com/en/copilot/using-github-copilot/ai-models/choosing-the-right-ai-model-for-your-task



Chat / Debugging

Generate debug configuration

Generate the debug configuration - **launch.json** file with **/startDebugging**.

Run the app in debug mode using the config

Prompt Crafting

Prompt crafting is the process of giving clear natural language instructions to computer programs, such as the large language models (LLMs) that drive GitHub Copilot functionalities.

- Zero-shot prompting
- Few-shot prompting
- Chain of Thought prompting

https://www.promptingguide.ai/techniques

Context is key

- Copilot relies on both the underlying model - trained on billions of lines of code - *and* the context in your editor to provide high quality suggestions
- Code completion prioritizes the immediate context, as well as context found in open files
- Chat has the ability to bring in even more context via the **@workspace** command to include additional project files that may not be open in the IDE

LLM is probabilistic, not deterministic

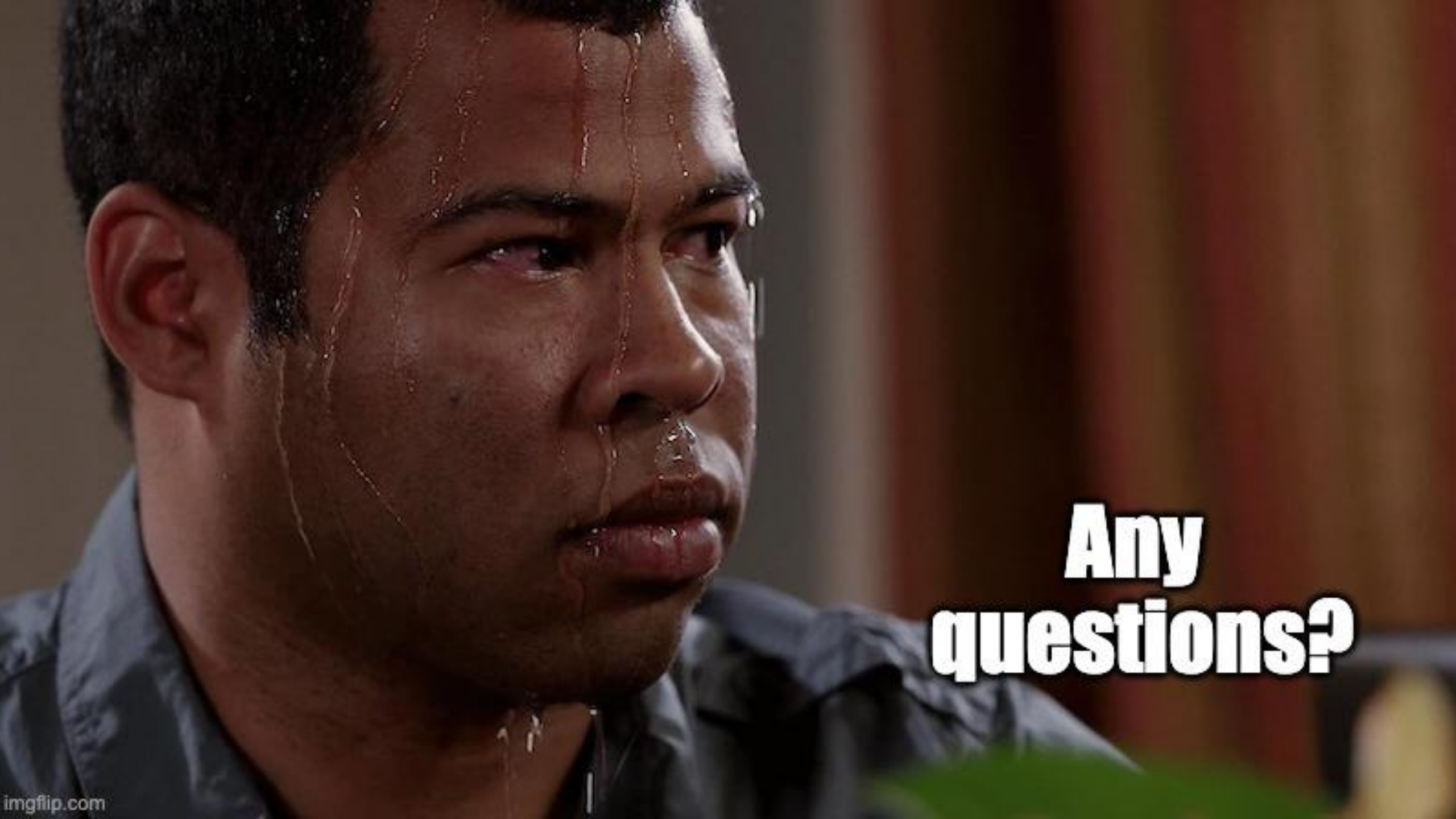
Again, LLMs are sophisticated probability engines, so you might not get the same results for the exact same prompts every time

If you don't succeed,

iterate

```
1   //Write a JavaScript function that finds
2   //the maximum value in an array.
3   function max(array) {
4       return Math.max.apply(null, array);
5   }
```


```
1   //Create a JavaScript function, `findMax`, that takes an array of strings as input
2   //and returns the string with the maximum length.
3   //Ensure that the function works correctly for arrays that are empty,
4   //as well as arrays that contain both strings and numbers.
5   function findMax(array) {
6       var max = array[0].length;
7       array.map(v => max = Math.max(max, v.length));
8       result = array.filter(v => v.length == max);
9       return result;
10  }
```


Any
questions?
imgflip.com

Thank you