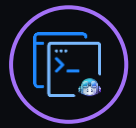
GitHub Expert Services

GitHub Copilot Tips, Tricks and Best practices

Agenda

Intro

Survey Outcome



Best Practices

Tips and Tricks

FAQ

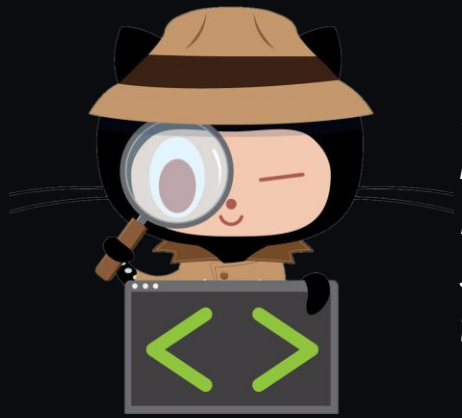
Q & A

Break

Hands-on Lab

GitHub Copilot

Beyond code completion



It's no longer just a code completion tool in your editor—it now includes a chat interface that you can use in your IDE, a command line tool via a GitHub CLI extension, a summary tool in your pull requests, a helper tool in your terminals, and much, much more.

- Copilot

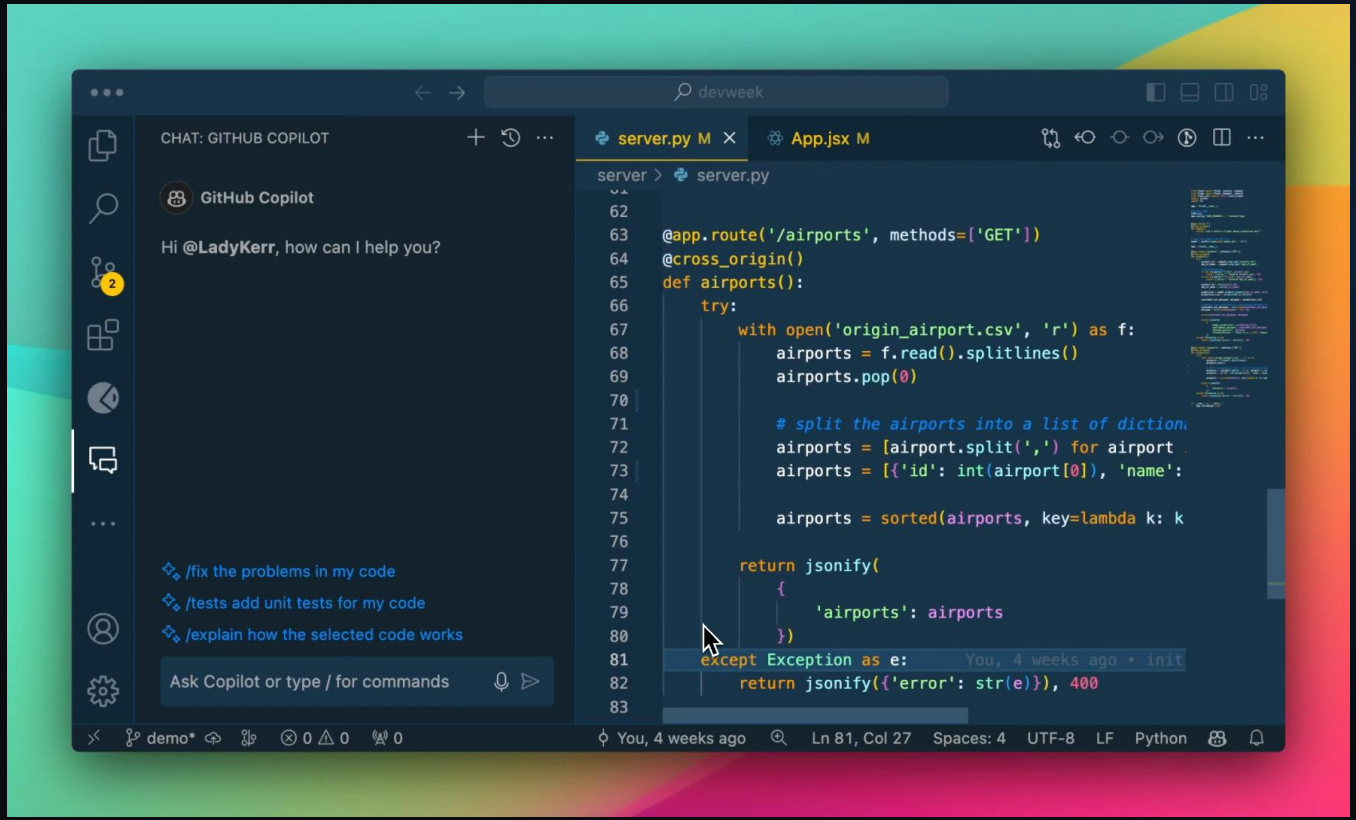
Context, Context, Context

- LLMs are designed to make prediction based on the context provided
- Provide ample context is the key
- Copilot is able to infer from the code in IDE editor
- How about adding more context?

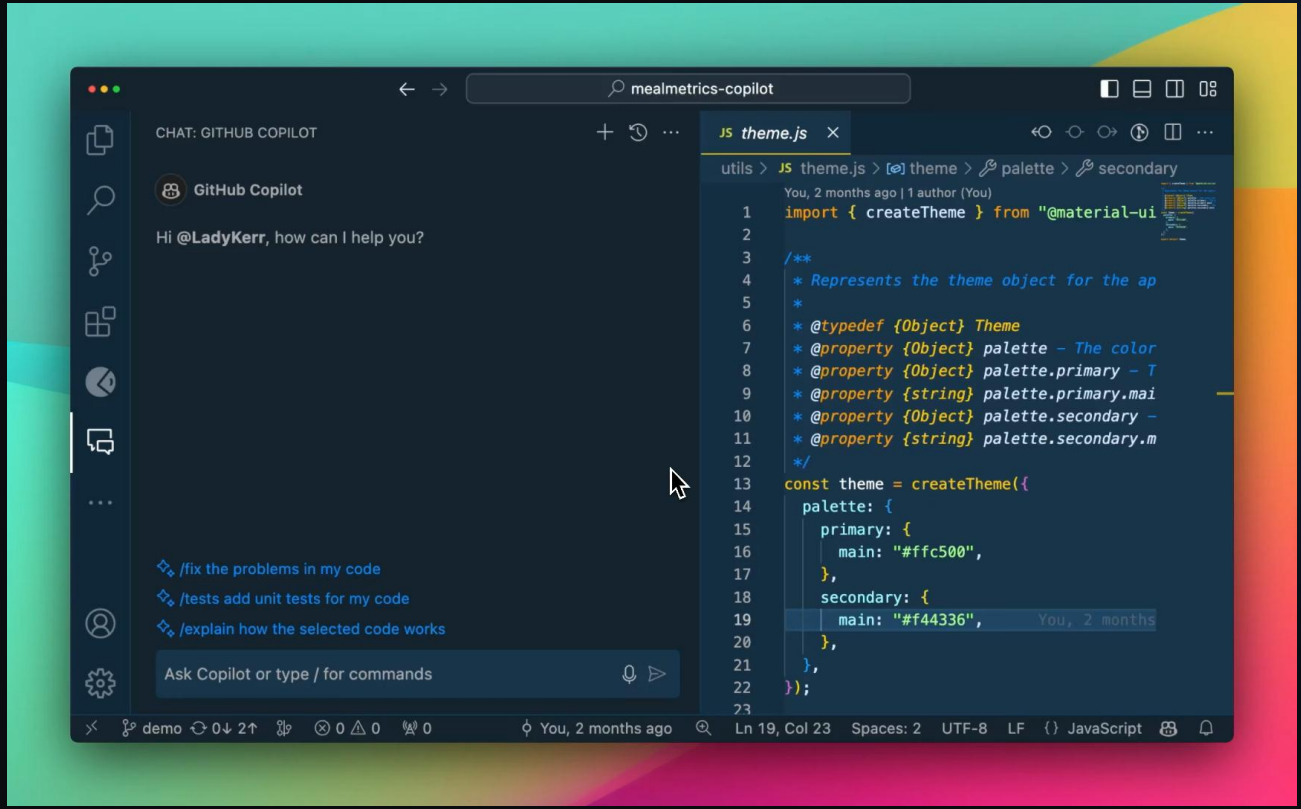
Context, Context, Context

- Open Relevant files
- Provide Top level comment
- Set Includes and references

Context, Context, Context



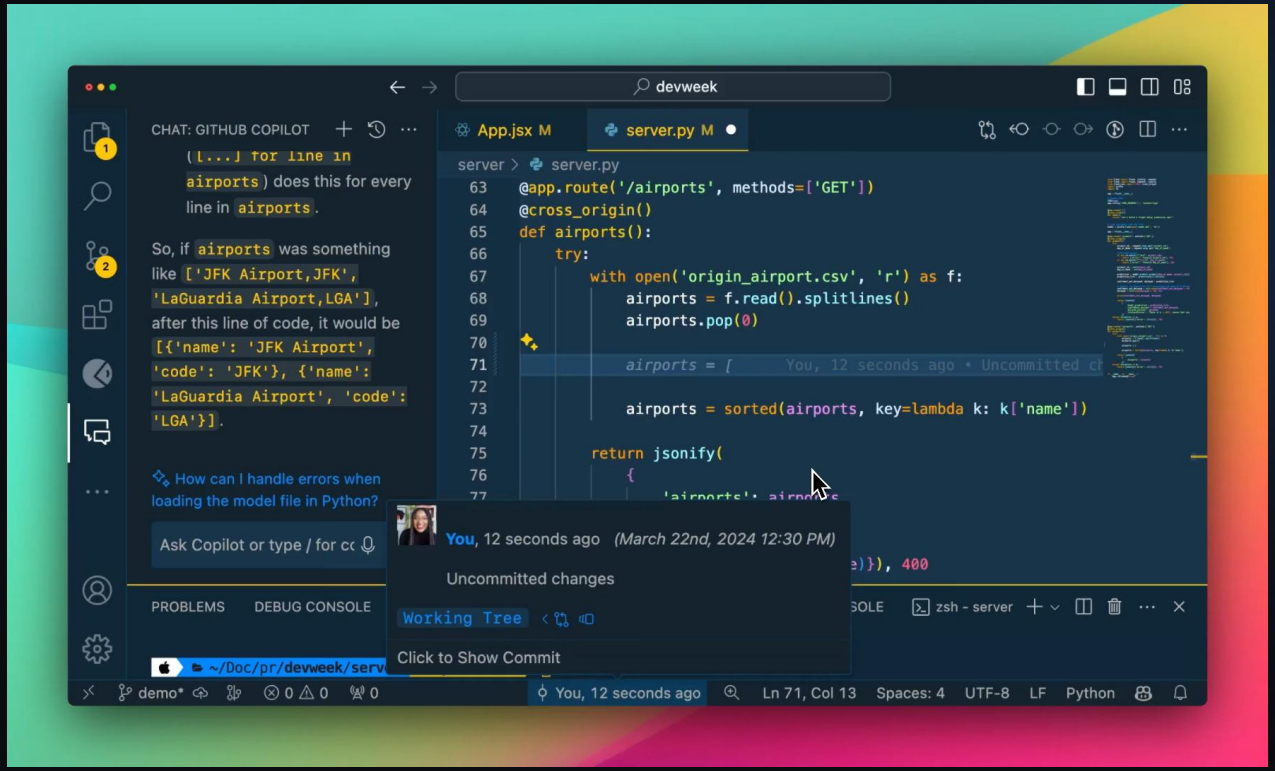
Context, Context, Context



Context, Context, Context

- Meaningful names
- Specific function comments
- Provide sample code
- Let’s give them a try

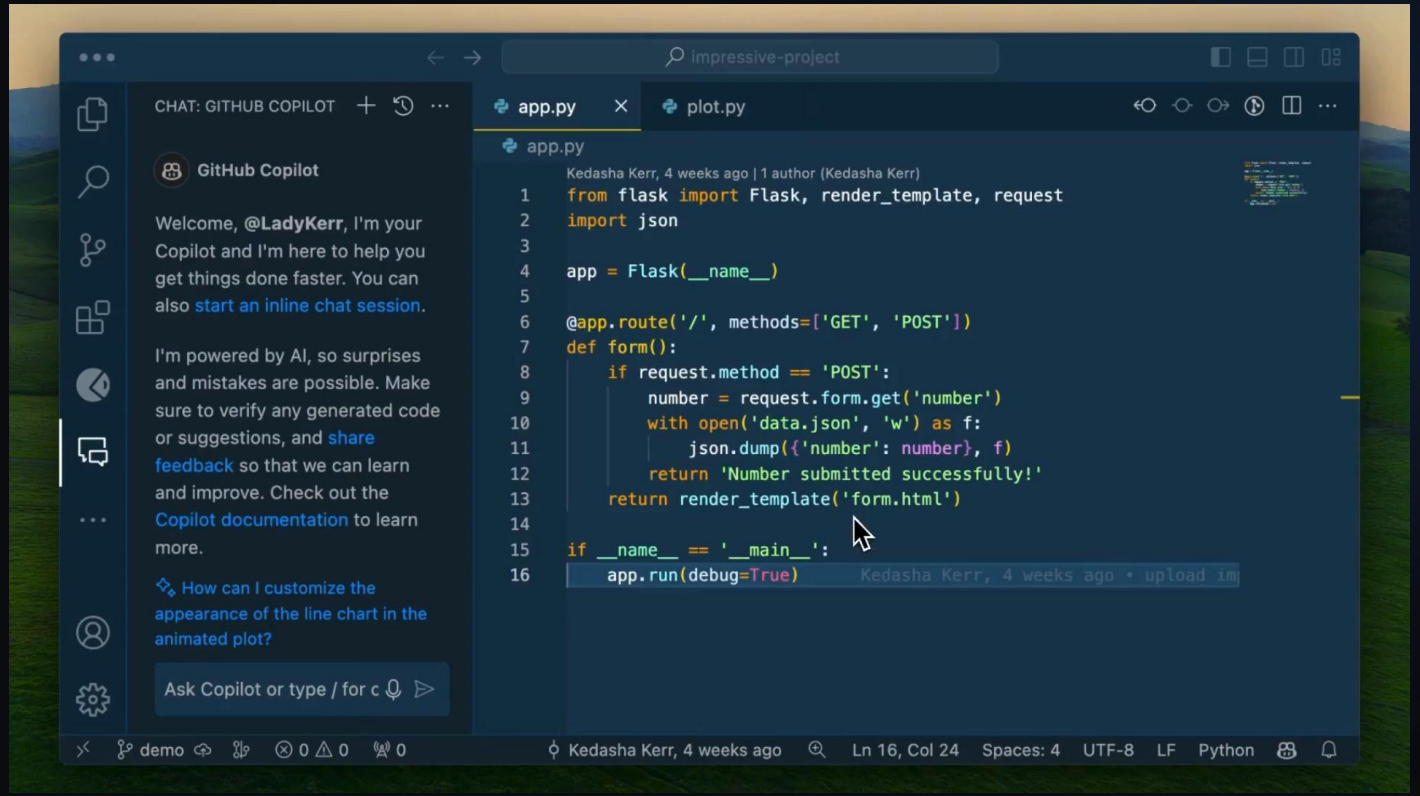
Context, Context, Context



Inline Chat with GitHub Copilot

- CTRL + I on Windows
- Ask specific questions
- Inline code diffs
- Slash command support
- Let’s give it a go

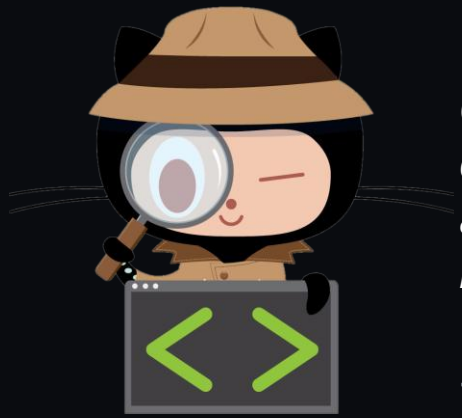
Inline Chat with GitHub Copilot



GitHub Copilot CLI

- **You can use Copilot with the GitHub CLI to get suggestions and explanations for the command line.**
- **To ask Copilot in the CLI to explain a command, run** gh copilot explain **followed by the command that you want explained.**
- **To ask Copilot in the CLI to suggest a command, run** gh copilot suggest **followed by the command that you want**

Copilot Chat



GitHub Copilot Chat provides an experience in your editor where you can have a conversation with the AI assistant. You can improve this experience by using built-in features to make the most out of it.

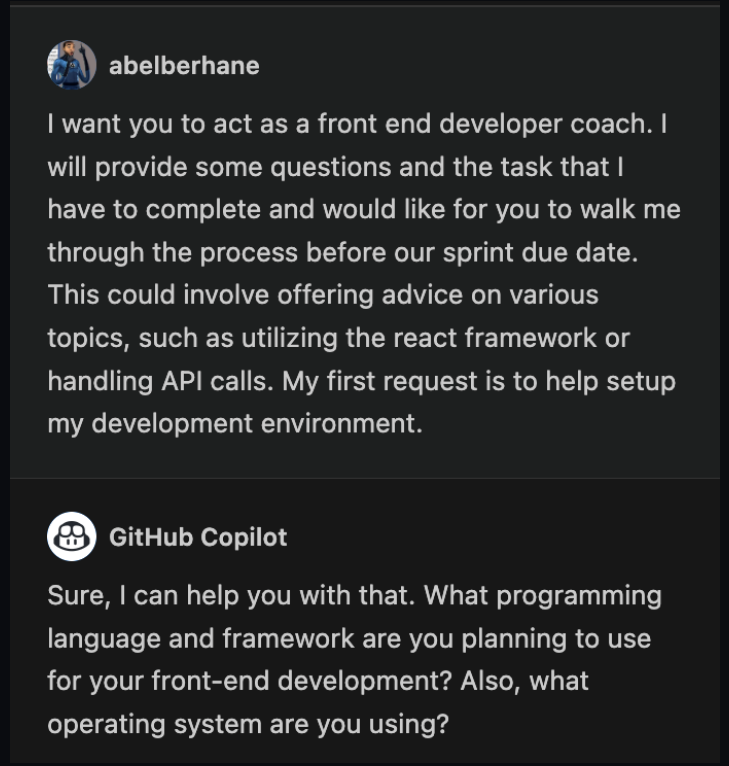
- Copilot

Copilot Chat

- Remove irrelevant requests
- Navigate through your conversation
- Use the @workspace agent
- Highlight relevant code
- Organize your conversations with threads

Role Prompting

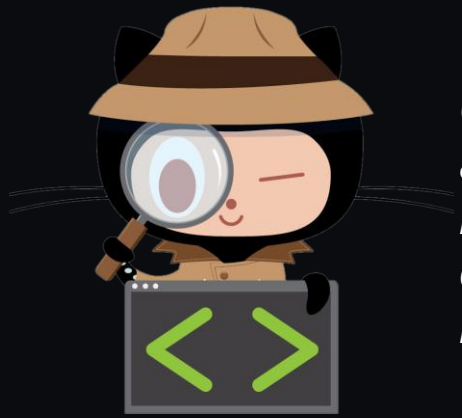
- Use copilot to assume a role during a task
- Focus to improves results
  Improves results by narrowing the AI's focus
- Enhanced Results



AI Hallucinations

- Nature of the Model
- Fabricated Outputs
- Highly Creative but not always right
- Reliance on Developers
- Feedback Loop

Copilot Edits



Copilot Edits start an AI-powered code editing session and iterate quickly on code changes across multiple files by using natural language. Copilot Edits applies the edits directly in the editor, where you can review them in-place, with the full context of the surrounding code.

- Copilot

Copilot Edits

- Select which files to edit
- Provide the relevant context and prompt
- Review the suggested edits
- Accept or discard the suggested edits
- Iterate on the code changes

How is Copilot Edits different from Copilot Chat?

- Copilot Edits puts you in the context of code editing. It can generate and apply code changes directly across multiple files
- The Chat view gives you a more general-purpose chat interface for asking questions about your code or technology topics in general.

Copilot Edits (Agent Mode)

In agent mode, Copilot Edits operates in a more autonomous and dynamic manner to achieve the desired outcome. Copilot agent mode determines the relevant context, offers both code changes and terminal commands, and iterates to remediate issues.

- Copilot

Copilot Edits (Agent Mode)

- Copilot Edits Agent Mode uses a set of tools to accomplish the individual tasks to complete a request. E.g., listing the files in a directory, editing a file in your workspace, running a terminal command, getting the output from the terminal, and more.
- Still in Preview

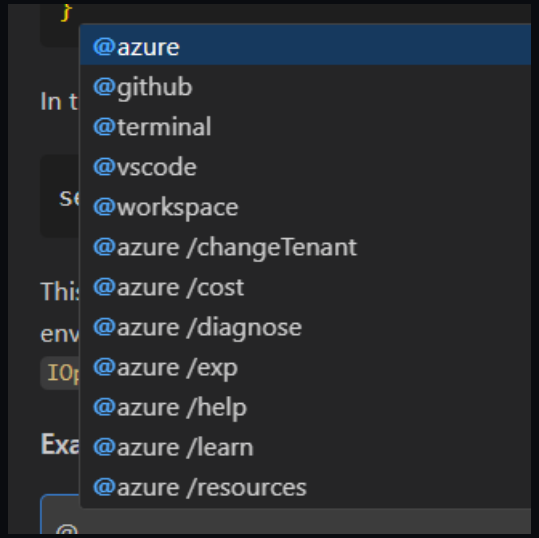
Slash Commands

Command	Description	Usage
/explain	Get code explanations	Open file with code or highlight code you want explained and type: /explain what is the fetchPrediction method?
/fix	Receive a proposed fix for the problems in the selected code	Highlight problematic code and type: /fix propose a fix for the problems in fetchAirports route
/tests	Generate unit tests for selected code	Open file with code or highlight code you want tests for and type: /tests
/help	Get help on using Copilot Chat	Type: /help what can you do?
/clear	Clear current conversation	Type: /clear
/doc	Add a documentation comment	Highlight code and type: /doc You can also press CMD+I in your editor and type /doc/ inline

Slash Commands

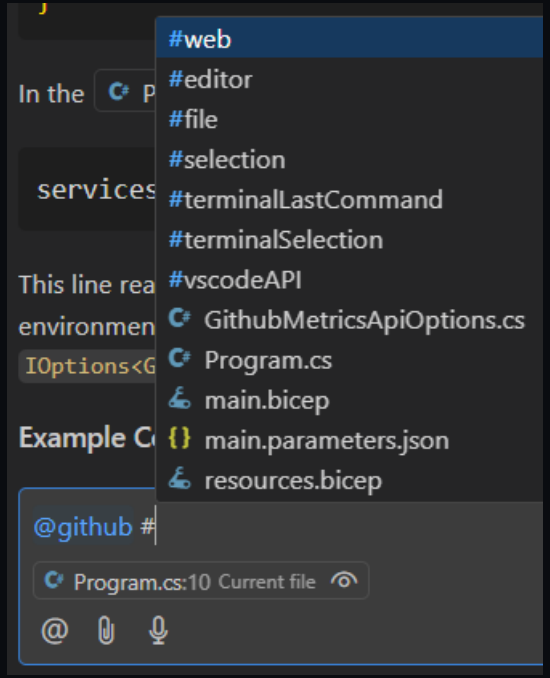
/generate	Generate code to answer your question	Type: /generate code that validates a phone number
/optimize	Analyze and improve running time of the selected code	Highlight code and type: /optimize fetchPrediction method
/clear	Clear current chat	Type: /clear
/new	Scaffold code for a new workspace	Type: /new create a new django app
/simplify	Simplify the selected code	Highlight code and type: /simplify
/feedback	Provide feedback to the team	Type: /feedback

Chat Participants



- Doesn’t respond straight away
- Analyzes the current workspace files and directories
- Creates a context from everything it was able to collect.

Chat Variables

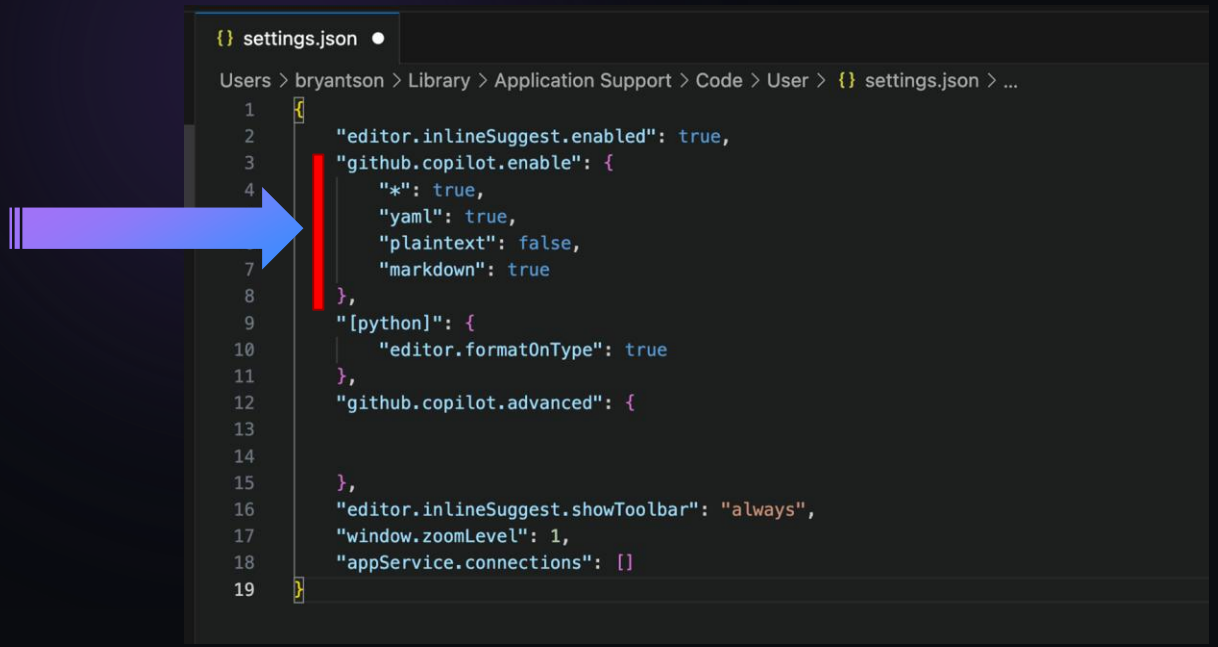


- Use chat variables to include specific context in your prompt.
- Use # in the chat prompt box, followed by a chat variable.
- #webSearch is in GA that uses Bing web search engine

Enable non-programming Meta Files

In settings.json, add

```
“github.copilot.enable”:
{
  “*”: true,
  “yaml”: true,
  “plaintext”: false,
  “markdown”: true
}
```



Let’s think step by step

Breaking Down Tasks

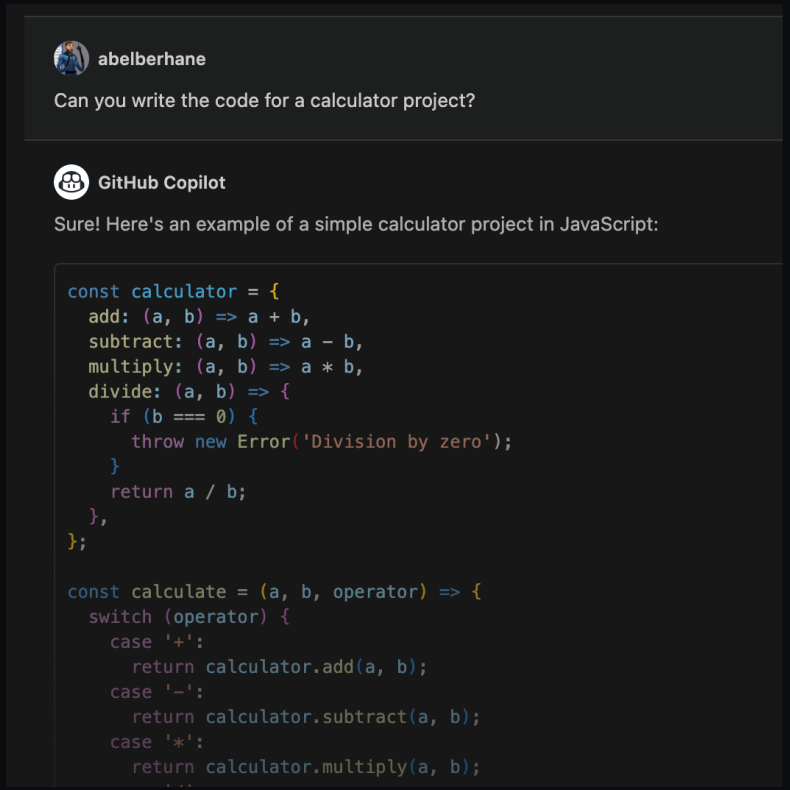
Itemises each step for you

Ensuring Clarity

Takes time to spell everything out

Avoiding Overwhelm

Organised approach



Let’s think step by step

Breaking Down Tasks

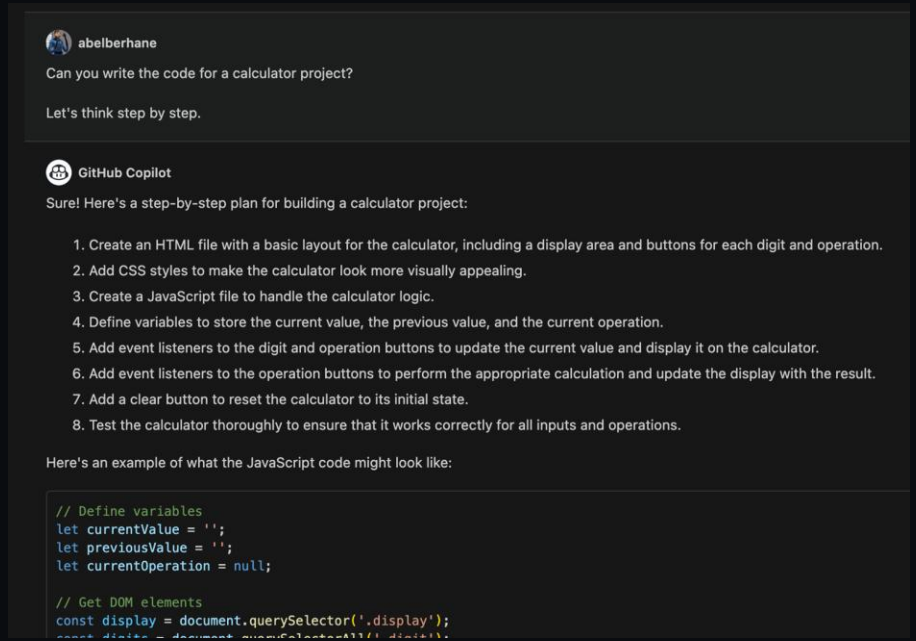
Itemises each step for you

Ensuring Clarity

Takes time to spell everything out

Avoiding Overwhelm

Organised approach



abelberhane

Can you write the code for a calculator project?

Let's think step by step.

GitHub Copilot

Sure! Here's a step-by-step plan for building a calculator project:

1. Create an HTML file with a basic layout for the calculator, including a display area and buttons for each digit and operation.
2. Add CSS styles to make the calculator look more visually appealing.
3. Create a JavaScript file to handle the calculator logic.
4. Define variables to store the current value, the previous value, and the current operation.
5. Add event listeners to the digit and operation buttons to update the current value and display it on the calculator.
6. Add event listeners to the operation buttons to perform the appropriate calculation and update the display with the result.
7. Add a clear button to reset the calculator to its initial state.
8. Test the calculator thoroughly to ensure that it works correctly for all inputs and operations.

Here's an example of what the JavaScript code might look like:

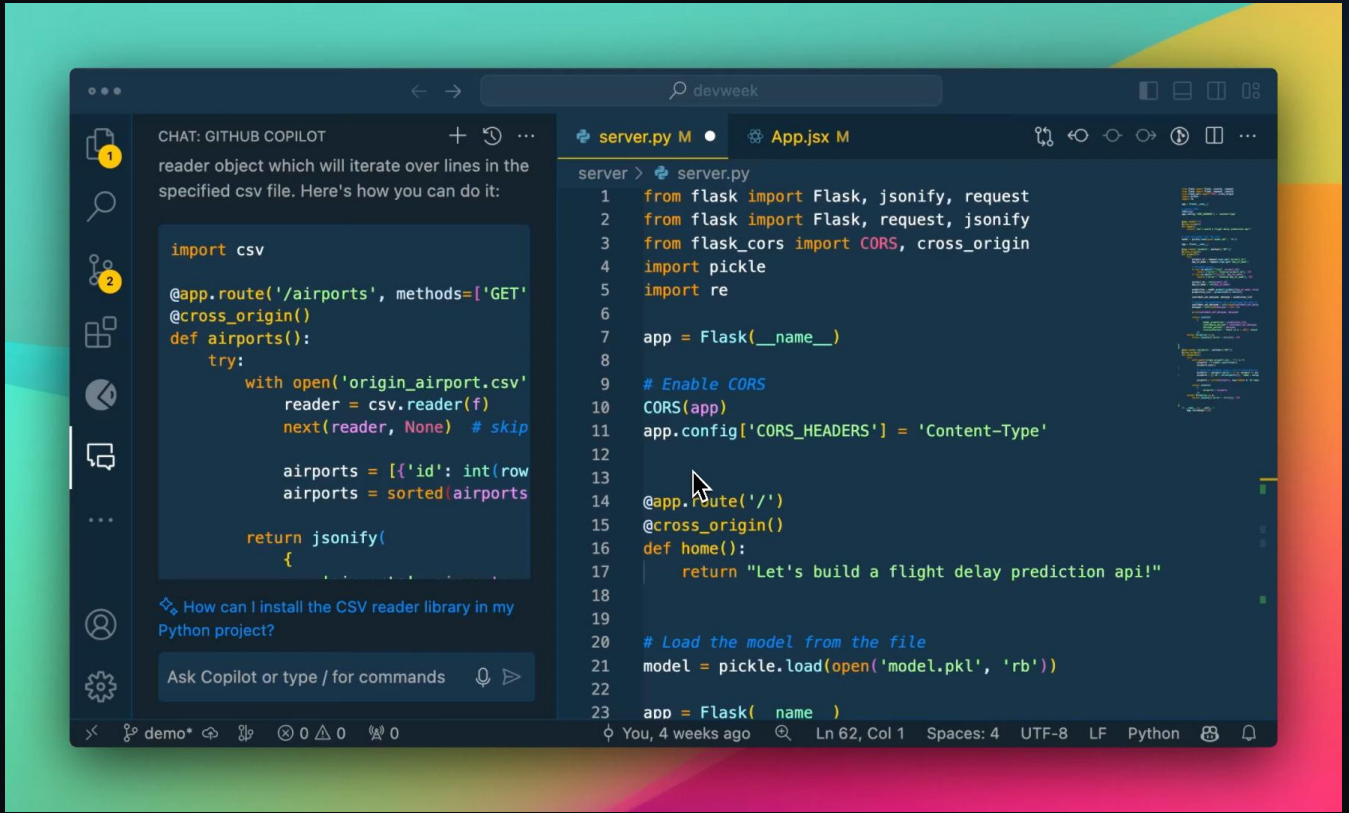
```
// Define variables
let currentValue = '';
let previousValue = '';
let currentOperation = null;

// Get DOM elements
const display = document.querySelector('.display');
```

More tips

- Attach relevant files for reference
- Start with GitHub Copilot Chat for faster debugging
- Be on the lookout for sparkles!

More tips


devweek
CHAT: GITHUB COPILOT
reader object which will iterate over lines in the specified csv file. Here's how you can do it:
import csv
@app.route('/airports', methods=['GET'
@cross_origin()
def airports():
try:
with open('origin_airport.csv'
reader = csv.reader(f)
next(reader, None) # skip
airports = [{'id': int(row
airports = sorted(airports
return jsonify(
{
How can I install the CSV reader library in my Python project?
Ask Copilot or type / for commands
server.py M
App.jsx M
server > server.py
from flask import Flask, jsonify, request
from flask import Flask, request, jsonify
from flask_cors import CORS, cross_origin
import pickle
import re
app = Flask(__name__)
# Enable CORS
CORS(app)
app.config['CORS_HEADERS'] = 'Content-Type'
@app.route('/')
@cross_origin()
def home():
return "Let's build a flight delay prediction api!"
# Load the model from the file
model = pickle.load(open('model.pkl', 'rb'))
app = Flask(name)
demo*
You, 4 weeks ago
Ln 62, Col 1
Spaces: 4
UTF-8
LF
Python

Conclusion

- Context
- Prompt Crafting
- Stay in control

Frequently Asked Questions

“Is the information shared with GitHub Copilot secure?

“For example, let’s say I enter some sensitive information through GitHub Copilot, is it going to store the data like my user information?”

“Can I train GitHub Copilot on my private codebase?”

“Is the Copilot suggestion from an IDE based on my computer, a project opened in my IDE, files opened through tabs or just a single file?”

“What telemetry data can I get from GitHub Copilot ?”

“How can GitHub Copilot integrate with existing pipelines/automations/scripts that we have?

“For example, I have this GitHub Actions/Azure DevOps pipeline, can GitHub Copilot integrate with those products so our code will always be secure?”

“Will Copilot replace my job”

“How much of my work can it take over?”

Q & A

Generate code explanations using GitHub Copilot Chat

- Use GitHub Copilot to explain unfamiliar and complex code
- https://learn.microsoft.com/en-gb/training/modules/generate-documentation-using-github-copilot-tools/3-exercise-generate-explanations-using-github-copilot-chat

Generate project documentation by using GitHub Copilot Chat

- Use GitHub Copilot to generate project documentation
- https://learn.microsoft.com/en-gb/training/modules/generate-documentation-using-github-copilot-tools/4-exercise-generate-project-documentation-using-github-copilot-chat

Generate inline code documentation by using GitHub Copilot Chat

- Use GitHub Copilot to generate code comments
- https://learn.microsoft.com/en-gb/training/modules/generate-documentation-using-github-copilot-tools/5-exercise-generate-code-documentation-using-github-copilot

Create code by using GitHub Copilot Inline Chat

- Use inline chat feature to update existing code and generate new code features.
- https://learn.microsoft.com/en-gb/training/modules/develop-code-features-using-github-copilot-tools/5-exercise-create-code-copilot-chat

Convert code from one programming language to another

- Use **chat interface** to convert code between programming languages
- https://learn.microsoft.com/en-gb/training/modules/develop-code-features-using-github-copilot-tools/10-exercise-convert-code-between-programming-languages

Create unit tests by using GitHub Copilot Chat

- Use **chat interface** to create unit tests for a code project.
- https://learn.microsoft.com/en-gb/training/modules/develop-unit-tests-using-github-copilot-tools/3-exercise-create-method-unit-tests-github-copilot-inline-chat

Upcoming Sessions

1. Intro to Prompt Engineering
2. GitHub Copilot for Data/ML Engineers
3. GitHub Copilot for DevOps Engineers
4. GitHub Copilot for Developers

Thank you!