GitHub Copilot Best Practices

Contains recommendations when using and configuring GitHub Copilot

On this page

Copilot Best Practices

The UI

- There may be a time where you are wondering if Copilot is working or if it failed to get a response, know your UI and take a look at the Copilot Icon to see if it is spinning. If it is spinning that means it is thinking and to give it some time.
- Just because you are familiar with Copilot Chat and Copilot being different extensions, does not mean your customer does. Showcase the extensions panel in your IDE and how they are separate.
- Show how you have your UI set up.
  - Where you can enable and disable Copilot.
  - Where you have Copilot Chat set up on the right.
  - Where Copilot Labs is (If you want to)

Details & Instructions

- Providing more details to GitHub Copilot helps it better understand the context and requirements of the task at hand.
- This can lead to more accurate and relevant code suggestions, saving time and effort for the developer.

    - Specificity - The more specific your prompt, the more targeted the response can be from Copilot.
    - Contextual Understanding - The more information you provide, the more context Copilot has at hand to give accurate responses
    - Efficiency - The more details you provide, the less iterations you will need to complete to get the desired response
    - Quality of Response - The usefulness and accuracy of your prompts will increase with the more understanding Copilot has of the task at hand

Context Context Context

- When you start working on a file, add a comment that states the purpose of what you are trying to accomplish. All other prompts moving forward will take that into consideration.
- Add comments to your code regularly. If you don't want to or like to, copy and paste your code into Copilot Chat and ask it to provide Comments for everything that it sees. This makes it more readable for yourself and adds extra context moving forward.
- If you want Copilot to have the context of other files, ensure that they are open in your IDE so that Copilot can absorb the neighboring tabs.

Provide a Reference Text

- It's essential to emphasize the importance of providing reference texts. Just as a student may perform better in an exam with a cheat sheet, Copilot will be more accurate when given source material.
- This practice helps counteract the tendencies of LLM models to produce inaccurate or invented outputs, especially on niche topics or when asked for specific references.
- By supplying Copilot with relevant references, it gains a clearer context, much like how a developer thrives with detailed documentation.
- Consequently, Copilot can produce code snippets or solutions that are more in tune with the given material, enhancing its efficiency and reliability.
- A couple ways that you can guide the customer to overcome this obstacle would be:
    - Asking Copilot to answer your question using a reference text or also providing the documentation
    - If using Copilot Chat, have Copilot answer with references to the documentation along with a breakdown of the logic step by step

Split Big Tasks Into Smaller Tasks

- If Copilot is failing to provide the prompt you want, try breaking it into smaller prompts.
- You do not have to understand how to actually do the code either. Just try to start writing the different sections of comments out so that Copilot starts to get the context of what is going on.
- 3 tactics to splitting big tasks into smaller tasks that you can work on with the customer (if they ask):
    - Use intent classification to try to create smaller groups. For instance, if you have toys, group the ones that race, the ones that play dress up, or the ones that are puzzles.
    - For large files that exceed token limitations, summarize or filter previous work
    - Summarize your code, scripts, projects or files piecewise and construct a full summary recursively

Test Changes Systemically

- Always take note of the changes that you make in your comments or prompts to Copilot chat and how it affects the results that you are getting back.
- Often getting into the habit of analyzing your prompts and responses can yield for really interesting findings. Stop drastically deleting your prompts and starting from scratch!

Resources

- How to use GitHub Copilot - GitHub Blog
- GitHub Copilot Best Practices - LinkedIn Article
- GitHub Copilot in VS Code Best Practies - LinkedIn Article
- Prompting Best Practices by OpenAI
- ChatGPT Role Prompting Examples usable in Copilot chat