

# GitHub Copilot

*Kick-Off*

*Andrew Scoppa*



+



# Useful Resources

- [Getting started with GitHub Copilot](#)
- [Insider newsletter digest: How to use GitHub Copilot](#)
- [GitHub Copilot Frequently Asked Questions](#)
- [GitHub Copilot Shortcuts](#)
- [Video - Get Started with the Future of Coding: GitHub Copilot](#)
- [Tutorial: GitHub Copilot and VS Code](#)

# GitHub Copilot Official Curriculum

**GitHub Copilot Fundamentals**

**GitHub Copilot Administration and Security**

**GitHub Copilot Developer**



# Agenda

- GitHub Copilot Acceleration Initiative + Upcoming Training
- GitHub Copilot Overview
- IDE Support
- Legal Requirements
- Setup Process
- Demo
- Questions

# GITHUB COPILOT ACCELERATION INITIATIVE



**19k+**

**total developers @AMD**

**≈800**

**Copilot Users**

## What?

GitHub Copilot offers suggestions by analyzing code in real-time, predicts the following lines, and provides context-aware suggestions to complete statements.

## Why?

The acceleration of Copilot adoption has become a high-priority initiative for AMD's Use of AI efforts, especially after the reported results of the initial users.

## How can you help?

Support in ramping up GitHub Copilot adoption across AMD. Namely –

- Providing us access to events to quickly explain the benefits of Copilot
- Encouraging Copilot sign-ups by pointing people to [dl.UseofAI.EngineeringRequests@amd.com](mailto:dl.UseofAI.EngineeringRequests@amd.com)



Here's findings from GitHub's research of developers using Copilot...

88%

...have faster completion

87%

...exert less effort on  
repetitive tasks

77%

...spend less time  
searching

73%

...stay more in the flow

60%

...are more fulfilled with  
[their] job

59%

...are less frustrated  
when coding

# Duolingo Success Story

25%

..increase in developer  
speed

1m

..set-up time for largest  
repo

67%

..decrease in median  
code review turnaround  
time

70%

..increase in pull requests

“

A tool like GitHub Copilot is so impactful at large companies because suddenly engineers can make impactful changes to other developers' code with little previous exposure.

“

Engineering time is the most valuable resource at Duolingo. Making the best use of that time with the help of GitHub Copilot and Codespaces enables us to reach our goals faster.

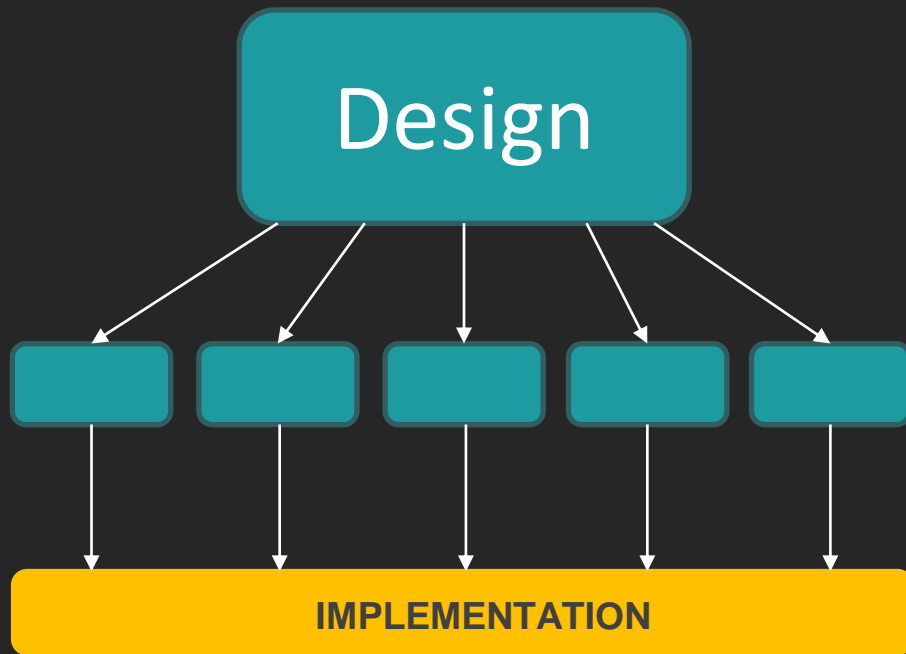
“

With GitHub Copilot, our developers stay in the flow state and keep momentum instead of clawing through code libraries or documentation.



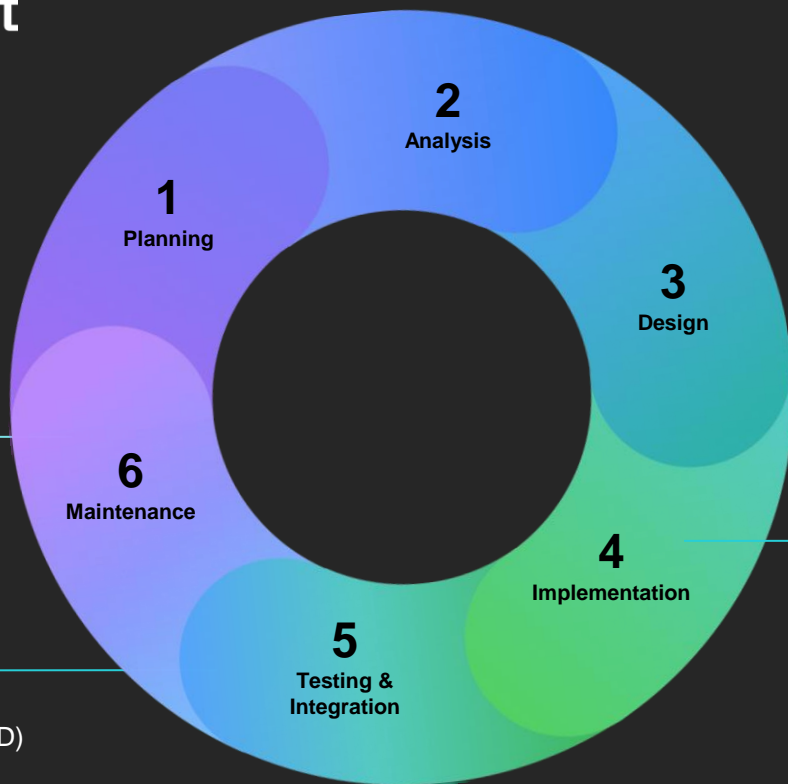
# GitHub Copilot Overview







Helps developers stay  
in the flow throughout  
the entire SDLC



Refactoring code (code translate)  
Reviewing code (code explain)  
Documentation

Unit testing (TDD and BDD)  
Finding code errors  
Debugging  
Code review  
AI Pull Requests

Convert comments to code  
Autofill for repetitive code  
Show alternatives


# GitHub Copilot

- An intelligent pair programmer
- Draws context from comments & code to suggest individual lines and whole functions
- Powered by OpenAI Codex
  - Copilot uses a transformative model
  - Trained on large datasets to ensure accuracy
- Available as extensions to popular IDEs
- Programming Languages and Technology available in Public code base all are supported

February 2023

```
TS sentiment.ts  -GO write_sql.go  parse_expense

1  #!/usr/bin/env ts-node
2
3  import { fetch } from "fetch-h2";
4
5  // Determine whether the sentiment of
6  // Use a web service
7  async function isPositive(text: string) {
8      const response = await fetch(`http://
9          method: "POST",
10         body: `text=${text}`,
11         headers: {
12             "Content-Type": "application/x-www
13         },
14     });
15     const json = await response.json();
16     return json.label === "pos";
17 }
```

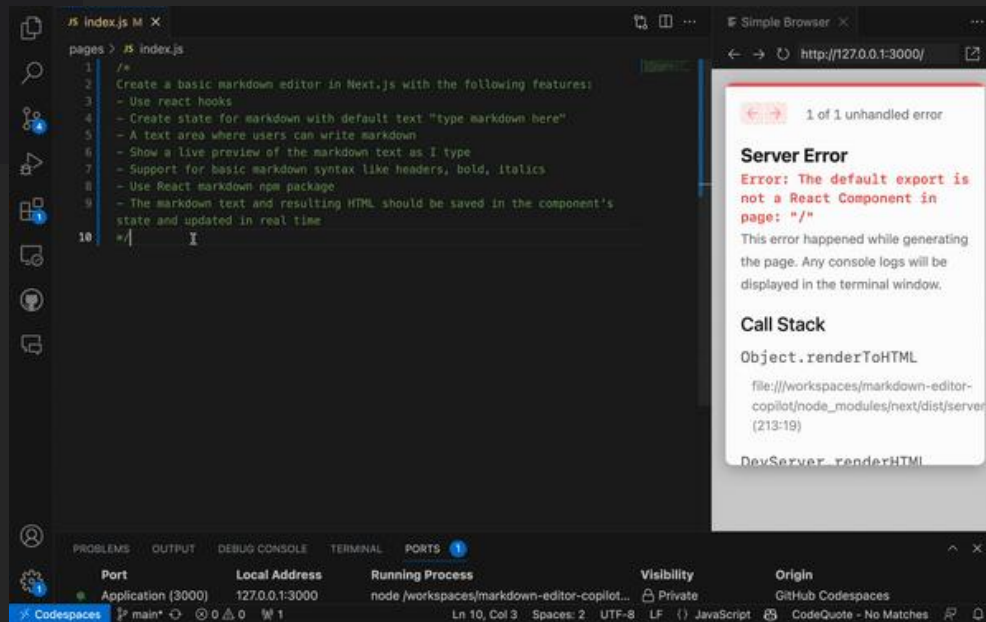
 Copilot

# Copilot offers autocomplete-style suggestions

```
def connect_to_endpoint(url):  
    response = requests.get("GET", url, auth=bearer_oauth, stream=True)  
    print(response.status_code)  
    for response_line in response.iter_lines():  
        if response_line:  
            print(response_line)  
  
    .....
```

## Inline

## Multiline Prompt



# ● GitHub Copilot @ AMD

- ✓ Copilot inline suggestions
- ✓ Copilot prompts
- ⌚ Copilot Chat... Not yet





How does Copilot work?

# GitHub Copilot

Once enabled...



OpenAI  
Codex



Context

Suggestions



Visual Studio



Neovim



VS Code



JetBrains IDEs

runtime.go course.rb time.js IsPrimeTest.java

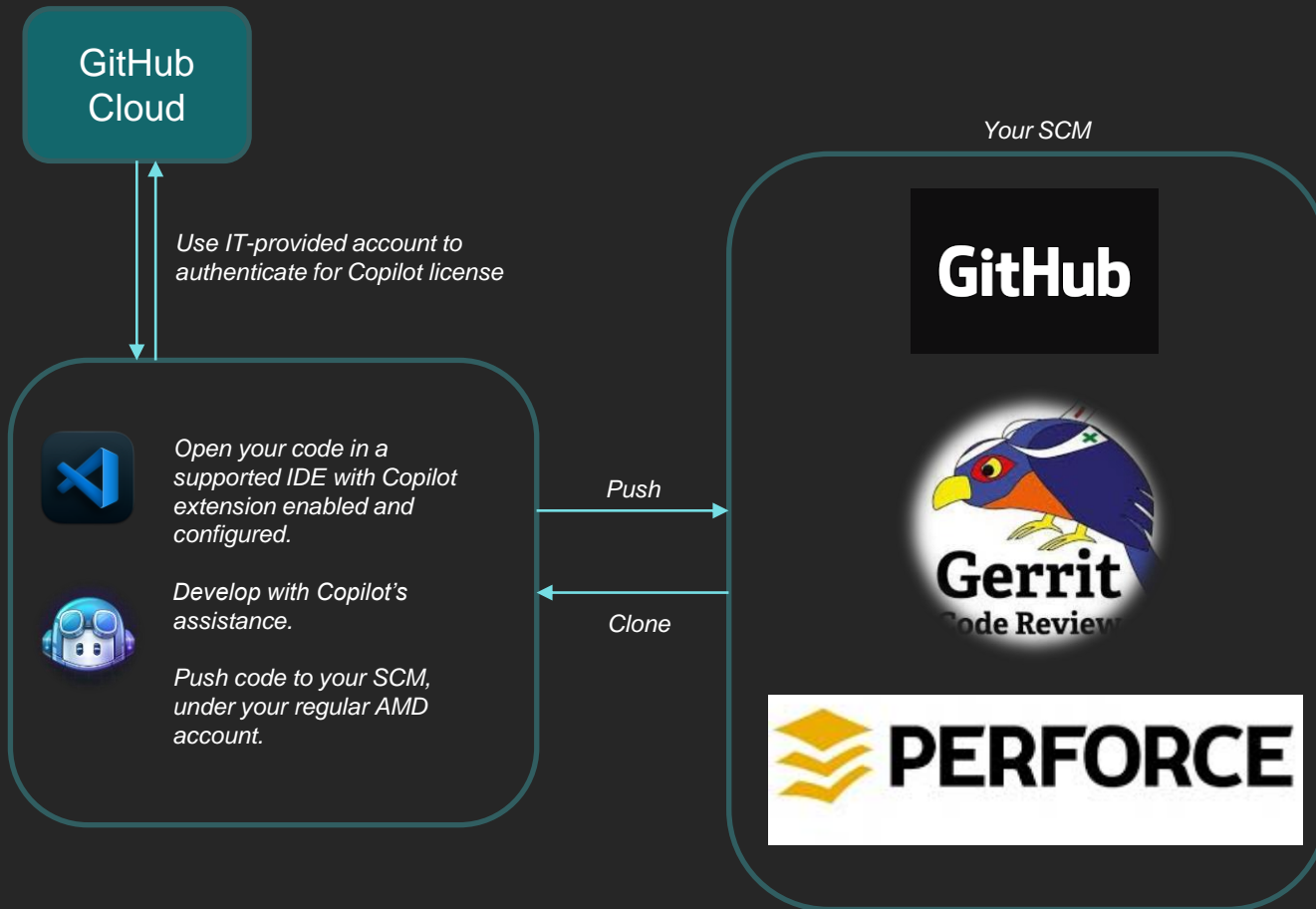
```
1 package main
2
3 type Run struct {
4     Time int // in milliseconds
5     Results string
6     Failed bool
7 }
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

# GitHub Copilot works something like this

- 1) Code starts in the user's editor. The Copilot extension gathers context from open files. That context is called the "prompt."
- 2) The prompt is sent with SSL encryption to a proxy server living in a GitHub-owned Azure subscription.
- 3) At the proxy server, we apply filters to scrub PII, tokens, vulnerabilities etc.
- 4) The proxy server sends the scrubbed prompt to the model.
- 5) The model produces 'n' suggestions based on the prompt, then deletes the prompt from memory.
- 6) The suggestions produced by the model are retained for a period of up to 30 days
- 7) The responses are sent back to the proxy server for another scrubbing.
- 8) The proxy server sends the suggestions back to the editor via HTTPS where they are displayed.



# Developer Workflow



# Common Questions

1. *Is GitHub Copilot “attached” to a repository?*  
Copilot provides context from local files
2. *Does Copilot take our code and put it back into the data model for future suggestions?*  
Copilot shares just long enough for GitHub Copilot to provide a suggestion.
3. *Does Copilot always write novel code?*  
Copilot will occasionally regurgitate what it thinks is a pattern if it's seen many times.
4. *Is there a filter which detects code suggestions matching public code on GitHub?*  
The Copilot team created an IP filter. You can choose to enable or disable the filter. When the filter is enabled, GitHub Copilot checks code
5. *What is a large language model?*  
Language models are a type of natural language processing (NLP) model that has been trained on large datasets of text to generate correct responses. The models do not contain a database of code, and they do not ‘look up’ snippets.



# Private GitHub Copilot Model

## ● How is private model different?

- A private and secure instance of Copilot model to be used only within AMD.
- Fine-tuned on our code for improved performance.
- Tailored better for hardware and low-level software languages (C/C++, Verilog, System Verilog, UVM)
- **Not all code is cleared for use with Copilot.**





## IDE Support

# ● Copilot Support in IDEs

- Azure Data Studio
- JetBrains IDEs (IntelliJ IDEA, PyCharm, and others)
- Vim/Neovim
- Visual Studio
- Visual Studio Code



# Getting started with GitHub Copilot

You can start using GitHub Copilot by installing the extension in your preferred environment.

[Azure Data Studio](#)[JetBrains IDEs](#)[Vim/Neovim](#)[Visual Studio](#)[Visual Studio Code](#)

## Important

GitHub Copilot can be managed through personal accounts with GitHub Copilot Individual or through organization accounts with GitHub Copilot Business.

GitHub Copilot is free to use for verified students, teachers, and maintainers of popular open source projects. If you are not a student, teacher, or maintainer of a popular open source project, you can try GitHub Copilot for free with a one-time 30-day trial. After the free trial, you will need a paid subscription for continued use. For more information, see "[About billing for GitHub Copilot](#)."

[Start a free trial](#)

## About GitHub Copilot and Vim/Neovim



# Environment Setup



## ● Step 1: Request a License

- Email [dl.useofai.engineeringrequests@amd.com](mailto:dl.useofai.engineeringrequests@amd.com) stating your ask and the purpose for the license.
- Read and follow instructions in the response email 😊
- Details on the [Copilot Access & Licensing Process](#) page.



## ● Step 2: Legal Guidelines

● Familiarize yourself with the official usage guidelines for engineers:

- *Access GitHub Copilot using the provided Copilot account.*
- *Code must only be pushed in approved branches/folders.*
- *Static code analysis (CodeQL/Coverity) is enabled for the approved branches/folders.*
- *Source code files must include the required copyright notice.*
- *All AI-generated code must be reviewed by a human.*



## ● Step 3: Environment Setup

- Configure proxy using the provided steps
- Install Copilot extension
- Use IT-provided account to sign-in in the IDE to activate Copilot

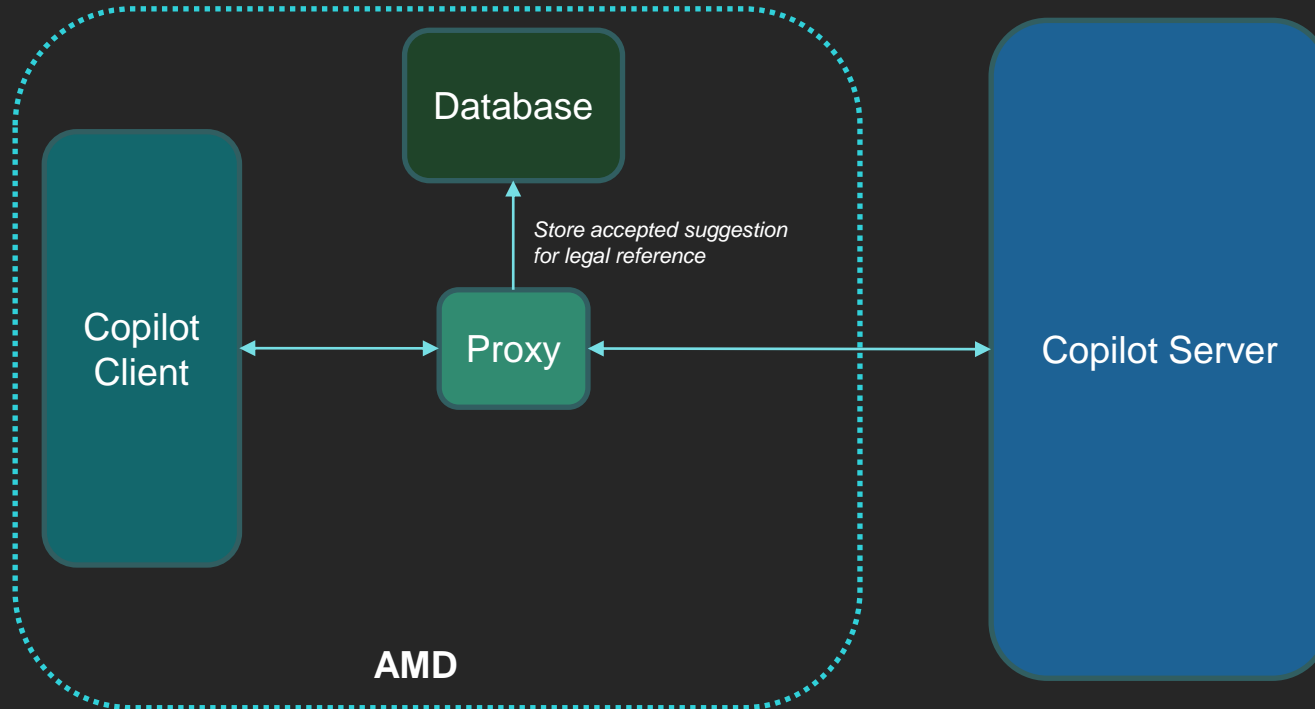
**NO MORE MANUAL CODE ANNOTATIONS!!**

AMD's new *Copilot Code Tracing solution* captures accepted Copilot code suggestions and store them for future reference, removing the need for devs to manually mark up their code.

*Notes: Must be on AMD Network (VPN) and have AMD Certificated installed.*



# Copilot Proxy



## ● MORE LEGAL Guidelines

● Please familiarize yourself thoroughly with the following resources:

- [AMD Requirements for AI Pilot Program Evaluation](#) legal document, which outlines the specific requirements for using pilot AI tools that have been approved by management for use within AMD.
- [Copilot for RTL Generation - Pilot Program](#) guidelines, which outlines the use cases in which you can legally use Copilot as well as some of the rules that must be followed for AI-generated files to be approved for use.
- [GitHub Copilot - Usage Guideline for Graphics Engineering](#) course, which outlines the AMD general guidelines for using Copilot and AI-generated code.







# Questions?

Post your questions on the [GitHub Copilot](#) MS Teams channel!