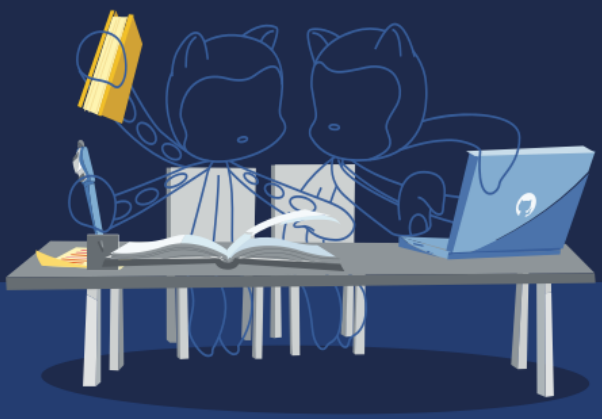
GitHub

Study Guide
GitHub Copilot



Get exam-ready for the GitHub Copilot Certification with our comprehensive study guide. We've curated the essential resources and learning activities to better prepare you for the exam and boost your chances of success.

Audience Profile

This exam is designed for individuals in the field of software development who are proficient in using GitHub, including software developers, administrators, and project managers. This certification is intended for individuals who have a foundational understanding of GitHub Copilot as a product and its available features, along with hands-on experience in optimizing software development workflows using GitHub Copilot.

Objective Domains

An objective domain for a certification exam, often referred to as a "domain" or "exam domain," is a structured outline or framework that defines the specific knowledge, skills, and topics that the certification exam will cover. It provides a clear roadmap for what candidates should expect to encounter on the exam and what they need to study and prepare for.

The domains provided in this study guide are intended to provide insight into the topic categories covered in the GitHub Copilot exam, along with the learning objectives within each domain.

Domain Breakdown	Exam Percentages
Domain 1: Responsible AI	7%
Domain 2: GitHub Copilot plans and features	31%
Domain 3: How GitHub Copilot works and handles data	15%
Domain 4: Prompt crafting and Prompt engineering	9%
Domain 5: Developer use cases for AI	14%
Domain 6: Testing with GitHub Copilot	9%
Domain 7: Privacy fundamentals and context exclusions	15%

Recommendations and Best Practices for Success

To increase your chances of success in passing the GitHub Copilot exam, candidates should have a fundamental understanding of GitHub, as well as hands-on experience in using GitHub Copilot features. The recommended learning paths for this exam provide you with an in-depth study of the learning content, followed by hands-on exercises and preparation assessment questions that were created to enable you to fine-tune your knowledge and readiness for the certification exam.

Content Resources

The following resources have been created in collaboration with GitHub as recommended content that covers the learning objectives in each domain for the GitHub Copilot exam. Both Microsoft Learn and LinkedIn Learning provide a complete learning path for the exam, but offer a different learning experience.

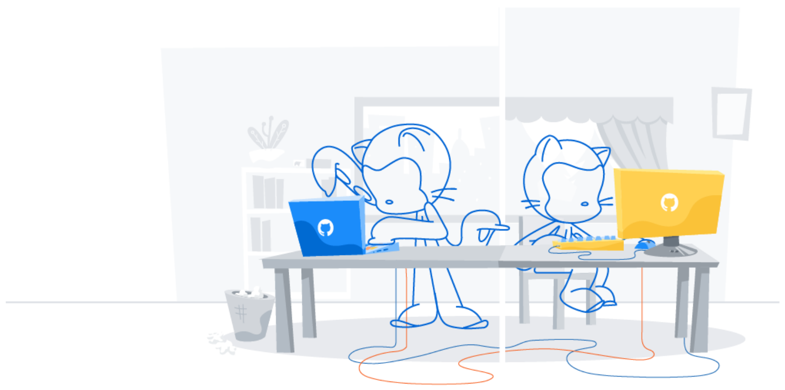
Microsoft Learn

In collaboration with MS Learn, we've created two learning paths that offer a comprehensive collection of modules designed to prepare you for using GitHub Copilot effectively. Learn how to enhance your coding experience with AI-powered assistance at every stage of your development lifecycle. GitHub Copilot is an AI tool integrated into GitHub that helps you write code faster and with fewer errors by suggesting code snippets, completing lines, and even generating entire functions. The following modules will guide you through GitHub Copilot's capabilities, equipping you with the skills needed to recognize, apply, and evaluate these features within your own GitHub environment.

- GitHub Copilot Fundamentals - Understand the AI Pair Programmer
- Accelerate App Development by Using GitHub Copilot

LinkedIn Learning

Master the art of coding efficiency and quality with AI-powered assistance throughout your software development process by exploring the Prepare for the GitHub Copilot Certification learning path on LinkedIn Learning. With GitHub Copilot, you have access to a powerful AI-driven tool that helps you write code faster, suggests improvements, and generates entire functions. This video-based learning path features a series of engaging courses that will guide you through the capabilities of GitHub Copilot. By the end of this learning path, you'll be well-equipped with the knowledge and expertise to seamlessly apply, assess, and maximize GitHub Copilot's features within your coding environment. ***(Learning path coming soon)***



Domain 1: Responsible AI

Explain responsible usage of AI
Describe the risks associated with using AI
Explain the limitations of using generative AI tools (dept of the source data for the model, bias in the data, etc.)
Explain the need to validate the output of AI tools
Identify how to operate a responsible AI
Identify the potential harms of generative AI (bias, secure code, fairness, privacy, transparency)
Explain how to mitigate the occurrence of potential harms
Explain ethical AI

Domain 2: GitHub Copilot plans and features

Identify the different GitHub Copilot plans
Understand the differences between Copilot Individual, Copilot Business, Copilot Enterprise, and Copilot Business for non-GHE
Understand Copilot for non-GitHub customers
Define GitHub Copilot in the IDE
Define GitHub Copilot Chat in the IDE
Describe the different ways to trigger GitHub Copilot (chat, inline chat, suggestions, multiple suggestions, exception handling, CLI)

Identify the main features with GitHub Copilot Individual
Explain the difference between GitHub Copilot Individual and GitHub Copilot Business (data exclusions, IP indemnity, billing, etc.)
Understand the available features in the IDE for GitHub Copilot Individual

Identify the main features of GitHub Copilot Business
Demonstrate how to exclude specific files from GitHub Copilot
Demonstrate how to establish organization-wide policy management
Describe the purpose of organization audit logs for GitHub Copilot Business
Explain how to search audit log events for GitHub Copilot Business
Explain how to manage GitHub Copilot Business subscriptions via the REST API

Identify the main features with GitHub Copilot Enterprise
Explain the benefits of using GitHub Copilot Chat on GitHub.com
Explain GitHub Copilot pull request summaries
Explain how to configure and use Knowledge Bases within GitHub Copilot Enterprise
Describe the different types of knowledge that can be stores in a Knowledge Base (e.g. code snippets, best practices, design patterns)
Explain the benefits of using Knowledge Bases for code completion and review (e.g. improve code quality, consistency, and efficiency)
Describe instructions for creating, managing, and searching Knowledge Bases within GitHub Copilot Enterprise, including details on indexing and other relevant configuration steps
Explain the benefits of using custom models

Identify the main features with GitHub Copilot Chat
Identify the use cases where GitHub Copilot Chat is most effective
Explain how to improve performance for GitHub Copilot Chat
Identify the limitations of using GitHub Copilot Chat
Identify the available options for using code suggestions from GitHub Copilot Chat
Explain how to share feedback about GitHub Copilot Chat
Identify the common best practices for using GitHub Copilot Chat
Identify the available slash commands when using GitHub Copilot Chat

Using GitHub Copilot in the CLI
Discuss the steps for installing GitHub Copilot in the CLI
Identify the common commands when using GitHub Copilot in the CLI
Identify the multiple settings you can configure within GitHub Copilot in the CLI

Domain 3: How GitHub Copilot works and handles data

Describe how GitHub Copilot handles data
Describe how the data in GitHub Copilot individual is used and shared
Explain the data flow for GitHub Copilot code completion
Explain the data flow for GitHub Copilot Chat
Describe the different types of input processing for GitHub Copilot Chat (types of prompts it was designed for)

Describe the data pipeline lifecycle of GitHub Copilot code suggestions in the IDE
Visualize the lifecycle of a GitHub Copilot code suggestion
Explain how GitHub Copilot gathers context
Explain how GitHub Copilot builds a prompt
Describe th proxy service and the filters each prompt goes through
Describe how the large language model produces its response
Explain the post-processing of GitHub Copilot's responses through the proxy server
Identify how GitHub Copilot identifies matching code

Describe the limitations of GitHub Copilot (and LLMs in general)
Describe the effect of most seen examples on the source data
Describe the age of code suggestions (how old and relevant the data is)
Describe the nature of GitHub Copilot providing reasoning and context from a prompt vs calculations
Describe limited context windows

Domain 4: Prompt Crafting and Prompt Engineering

Describe the fundamentals of prompt crafting
Describe how the context for the prompt is determined
Describe the language options for promoting GitHub Copilot
Describe the different parts of a prompt
Describe the role of prompting
Describe the difference between zero-shot and few-shot prompting
Describe the way chat history is used with GitHub Copilot
Identify prompt crafting best practices when using GitHub Copilot

Describe the fundamentals of prompt engineering
Explain prompt engineering principles, training methods, and best practices
Describe the prompt process flow

Domain 5: Developer use cases for AI

Improve developer productivity
Describe how AI can improve common use cases for developer productivity - Learning new programming languages and frameworks - Language translation - Context switching - Writing documentation - Personalized context-aware responses - Generating sample data - Modernizing legacy applications - Debugging code - Data science - Code refactoring
Discuss how GitHub Copilot can help with SDLC (Software Development Lifecycle) management
Describe the limitations of using GitHub Copilot
Describe how to use the productivity API to see how GitHub Copilot impacts coding

Domain 6: Testing with GitHub Copilot

Describe the options for generating testing for your code
Describe how GitHub Copilot can be used to add unit tests, integration tests, and other test types to your code
Explain how GitHub Copilot can assist in identifying edge cases and suggesting tests to address them

Enhance code quality through testing
Describe how to improve the effectiveness of existing tests with GitHub Copilot's suggestions
Describe how to generate boilerplate code for various tests types using GitHub Copilot
Explain how GitHub Copilot can help write assertions for different testing scenarios

Leverage GitHub Copilot for security and performance
Describe how GitHub Copilot can learn from existing tests to suggest improvements and identify potential issues in the code
Explain how to use GitHub Copilot Enterprise for collaborative code reviews, leveraging security best practices, and performance considerations
Explain how GitHub Copilot can identify potential security vulnerabilities in your code
Describe how GitHub Copilot can suggest code optimizations for improved performance

Domain 7: Privacy fundamentals and context exclusions

Describe the different SKUs for GitHub Copilot
Describe the different SKUs and the privacy considerations for GitHub Copilot
Describe the different code suggestion configuration options on the organization level
Describe the GitHub Copilot Editor config file

Identify content exclusions
Describe how to configure content exclusions in a repository and organization
Explain the effects of content exlcusions
Explain the limitations of content exclusions
Describe the ownership of GitHub Copilot outputs

Safeguards
Describe the duplication detector filter
Explain contractual protection
Explain how to confiture GitHub Copilot settings on GitHub.com - Enabling / disabling duplication detection - Enabling / disabling prompt and suggestion collection
Describe security checks and warnings

Troubleshooting
Explain how to solve the issue if code suggestions are not showing in your editor for some files
Explain why context exclusions may not be applied
Explain how to trigger GitHub Copilot when suggestions are either absent or not ideal
Explain steps for context exclusions in code editors