# **Standard Prompts in GitHub Copilot Chat for Testing Scenarios with Model Suggestions**

Table of Contents

[**Objective** 2](#_Toc184392872)

[**Target Audience** 2](#_Toc184392873)

[**Introduction** 2](#_Toc184392874)

[**Prerequisites** 2](#_Toc184392875)

[**Key Considerations** 2](#_Toc184392876)

[**Prompts for Different Test Scenarios** 2](#_Toc184392877)

[**GitHub Copilot Multimodal Comparison** 7](#_Toc184392878)

[**Final Score of all Models** 9](#_Toc184392879)

[**Preferred Model of each Scenario** 9](#_Toc184392880)

[**Model prioritization and Remarks :** 11](#_Toc184392881)

# **Objective**

The Objective of this documentation is to guide users on which prompts to use in GitHub Copilot Chat for various testing-related scenarios. It also provides model recommendations for each scenario to ensure optimal performance and accuracy.

# **Target Audience**

Quality Assurance teams who are using GitHub copilot.

Software Testers, Quality Assurance Engineers, Developers

# **Introduction**

GitHub Copilot is introducing new AI models: Anthropic’s Claude 3.5 Sonnet and OpenAI’s o1-preview and o1-mini. These models enhance coding tasks, offering developers more choice and control. The models will be available in Copilot Chat and other GitHub Copilot features. This update aims to improve efficiency and flexibility for developers

This document outlines the prompts a test engineer can use for their daily line testing scenarios and identifies which model delivers the best performance for each scenario.

# **Prerequisites**

* Git Hub Copilot access
* Any code editor (for vs code version should be above 1.94 to access multiple model featue)
* Multi model option should be enabled in github copilot chat

# **Key Considerations**

* User can select and ask any model for the response; In this document we provided prompts and models which are best suitable for the specific tasks in testing perspective
* These are not the final prompts, but are the best prompts with respect to each criteria, user can use other prompts also for there purpose
* Modify or change the prompts according to your use case
* while dealing with multiple file changes use GitHub copilot Edit feature

# **Prompts for Different Test Scenarios**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. No** | **Key Testing Use cases** | **Prompts** | | **Explanation** |
| 1 | Test Case generation | 1.1 | I am a manual tester trying to write manual test cases for the following requirement: | 1. specify who you are and what you want 2. explain use case in quotes 3. specify what are type testcases you are expecting 4. provide any example testcases/steps if you have 5. inform any note points we must check while generating response. It can we use case specific or format the testcases you want to generate like table format |
| ["From the URL: https://liveengineering-test.ad.infosys.com/liveengineering/, the user can select the Dev Manager, Portfolio Manager, or Product Owner persona from the profile icon located on the top header. After selecting a persona, the user should be able to see the Home page, which contains different sections with various cards. Check the Sprint Velocity card under the Agility section."](https://liveengineering-test.ad.infosys.com/liveengineering/) |
| Please help me generate all possible test cases, including boundary value analysis, security validations, performance validation (ensuring the pipeline tab loads within 2 seconds), and other non-functional requirements (NFR) test cases. |
| Below is an example of the test steps: |
| 1. Navigate to the Live Engineering Platform using the provided URL. |
| 2. Click on the profile icon and select the desired persona. |
| 3. Verify that the Home page is displayed for the selected persona. |
| 4. Verify the sections on the Home page. |
| 5. Verify the presence of the Lead Timecard under the Agility section. |
| 6. Verify the UI of the Lead Timecard when data is available and when no data is available. |
| 7. Verify the data displayed on the Lead Timecard for last 15, 30 and 60 days. |
| 8. Verify the last refreshed date and feedback section below the card. |
| 9. Verify the increment and decrement percentages for the selected period. |
| 10. Ensure the card shows "No data available" when the pipeline is failed or unstable. |
| Note: 1. Compatible browsers are Chrome and Edge |
| 2. Sprint Velocity Formula: Sprint Velocity = (Total story points delivered/Number of closed sprints within the time) |
| 1.2 | I am a manual tester trying to write manual test cases for the following requirement:  #file: "file name' Please help me generate all possible test cases, including boundary value analysis, security validations, performance validation (ensuring the pipeline tab loads within 2 seconds), and other non-functional requirements (NFR) test cases. | 1. if we are having use case description in any word doc, then use this prompt and #file command to select the file |
|  |
| 2 | Test Data Generation | 2.1 | Generate a comprehensive test dataset in Json format for a search field to ensure its effectiveness and security.  Include test data  for:  Valid Inputs: Exact matches, partial matches, and variations in capitalization, diacritics, and homonyms.  Invalid Inputs: Empty strings, whitespace, special characters, emojis, long strings, and potential security vulnerabilities (e.g., SQL injection, XSS).  Language-Specific Inputs: Test with queries in different languages (e.g., Chinese, Arabic, Russian).  Edge Cases: Consider scenarios like multiple-word queries, phrase search, and wildcard search.  Performance: Evaluate search performance with large datasets and complex queries.  Accessibility: Ensure search results are accessible to users with disabilities. | 1. In this prompt user can provide the data type they want to generate point wise 2. format of data generation also is users’ choice like Json, table |  |
| 3 | Test Script Generation | 3.1 | we can use similar prompt from step 1.1  just after the all the possible testcases provide python/java selenium code  e.g.: Please help me generate all possible test cases in python selenium including boundary value analysis, security validations, performance validation (ensuring the pipeline tab loads within 2 seconds), and other non-functional requirements (NFR) test cases. | 1. this is for the selenium/Appium script generation 2. same prompt as 1.1 just provide in which language you want to create automation script like java or python 3. if you do not want to create a test script in one go means provide details one by one and generate code line by line |  |
| 4 | Test Framework Creation | 4.1 | generate a java selenium code from scratch in vs code editor with all the necessary dependencies | 1. before start writing a code user needs to select language and framework where he wants to create code. this prompt we can use to generate framework from scratch 2. user can give code editor name in which they want to create code like vs code, eclipse |  |
| 5 | Code Migration | 5.1 | "@workspace" Migrate whole code in all files to java | 1. If we want to migrate whole code which is in multiple files to another language use this prompt 2. I am assuming my code is in other than java and workspace or file should be opened when we use this prompt |  |
|  |  | 5.2 | "@workspace" migrate whole code to java | 1. If we want to migrate only one file to another language use this prompt 2. I am assuming my code is in other than java and workspace or file should be opened when we use this prompt |  |
|  |  | 5.3 | migrate this code in this path "C:\Code\python\_code\_for\_api\developer\_details.py" to java | 1. If user is not opened the file which they want to migrate or they are in another file, in this case we can use the path of the fille which we want to migrate. 2. I am assuming my code is in other than java |  |
| 6 | Locator generation | 6.1 | Provide all locators for below Dom which I can use for functional automation, locators should be stable and will have less probability of changing after angular/Dom changes "Dom element" | 1. this will provide multiple locators based on requirement we can select the locator 2. if we want specific locator specify that like xpath locator, id locator 3. provide Dom element in quotes |  |
| 7 | Explain this | 7.1 | Explain this  (or) Explain this below code "code snippet" (or) Explain this code line by line | 1. highlight the code you want explanation, use first prompt for this  2. if you are having snippet then use Explain below code and provide that snippet of code or error, use second prompt 3. if you want explanation line use third prompt 4. if the code is large like more than 250 lines it is suggested to ask few lines for better explanation |  |
| 8 | Unit Testcases | 8.1 | Generate Unit Testcases for this code (or) Generate Unit testcases for below function "Function code snippet" | 1. For first prompt highlight or open the file for which you want to generate unit testcases 2. For second prompt provide function or code snippet of function in place of quotes |  |

# **GitHub Copilot Multimodal Comparison**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No** | **Key Use Cases** | **Weightage** | **GPT 4o** | **GPT 4o Score** | **Claude 3.5 Sonnet** | **Claude 3.5 Sonnet Score** | **o1-mini** | **o1-mini-Score** | **o1-preview** | **o1-preview Score** |
| 1 | Test Case Generation | 15% | Generate all possible scenarios for the test cases. However, some responses are stopped by the responsible AI service. | 6 | Covers all scenarios but provides response with fewer steps | 5 | Provides more detailed test cases, covering all scenarios comprehensively. | 7 | Provides more detailed test cases, covering all scenarios comprehensively. | 7 |
| 2 | Test Data Generation | 20% | Generates all the test data for the specified prompt and samples | 7 | Generates all the test data for the specified prompt and samples. However, without any prior context data, it produces fewer samples. | 6 | Generates all the test data in detail along with the explanation | 7 | Generates all the test data in detail along with the explanation | 7 |
| 3 | Test Script Generation, Framework creation and Code Migration (Python/Java + Selenium) | 35% | Script generation is good, but the generated/migrated code sometimes lacks exception handling. And need to ask more follow-up questions to get expected response. | 7 | Generating and migrating code. However, sometimes we encounter a public code match error issue , and the response is not returned. | 5 | The script generation and migration can happen in single prompt and provide detailed code with comprehensive exception handling. (can handle 50 prompts/day)  . | 9 | The script generation and migration can happen in single prompt and provide detailed code with comprehensive exception handling. (can handle 10 prompts/day) | 8 |
| 5 | Explain This Feature | 10% | Explains the entire code line by line. | 7 | Explains the code, but sometimes some lines are not explained. | 4 | Provides a detailed explanation without leaving any line of code | 8 | Provides a detailed explanation without leaving any line of code. | 8 |
| 7 | Speed of Output | 10% | 3-5 seconds | 6 | 5-10 sec | 8 | 8-12 sec | 6 | 15-25 sec | 4 |
| 8 | Unit Testcases | 10% | Achieved comparatively less coverage for the selected code. | 5 | Achieved better coverage compared to the GPT 4o model. | 6 | Achieved better coverage compared to the Sonnet model. | 7 | Achieved better coverage and provided a detailed explanation of the generated test case. | 8 |

# **Final Score of all Models**

|  |  |
| --- | --- |
| **GPT 4o** | **6.45** |
| **Claude 3.5 Sonnet** | **5.5** |
| **o1-mini** | **7.7** |
| **o1-preview** | **7.25** |

# **Preferred Model of each Scenario**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S. No | Key Testing Use Cases | Complexity | Preferred Model | Explanation/ Reason |
| 1 | Test Case generation | Complex/large prompt | 1. o1-mini 2. o1- preview | Provides more detailed test cases, covering all scenarios comprehensively. Initially, we can use o1-mini, which has a 50-prompt limit, compared to o1-preview’s 10-prompt limit. |
| Simple/Small prompt | GPT 4o | Generates all the possible test cases covering all scenarios for simple scenarios |
| 2 | Test Data Generation | Complex/large samples of data | 1. o1- preview 2. o1- mini | Generates all the test data in detail along with explanations. While o1-preview has a slight edge over o1-mini, the difference is minimal. We can consider the prompt limits: o1-mini has a 50-prompt limit, whereas o1-preview has a 10-prompt limit and use. |
| Medium samples of data | GPT 4o | Compared to the o1 models, it generates less data, but a considerable number of samples will still be produced. |
| Simple/Small samples of data | Claude 3.5 Sonnet | Comparatively less data will be generated, we can use for less samples |
| 3 | Test Script Generation | Complex scenarios | 1. o1-mini 2. o1- preview | Test Script generation is very good with these models And provide more detailed code with comprehensive exception handling. Initially, we can use o1-mini, which has a 50-prompt limit, compared to o1-preview’s 10-prompt limit. |
| Medium scenarios | GPT 4o | Test script generation is good here, but sometimes exception handling is missing. For large code generation, we must ask multiple follow up questions |
| simple scenarios | Claude 3.5 Sonnet | We can use this for simple scenarios and shorter lines of code. However, we may encounter errors such as public code matches or no response returned in some scenarios |
| 4 | Test Framework Creation | Framework from scratch | GPT 4o | We can use this model for framework creation as it provides good responses. However, if we encounter a public code match error, we can switch to the o1 models. |
| 5 | Code Migration | More loc/multiple file migration | 1. GPT 4o 2. o1-mini/o1-preview | GPT-4o and mini models can migrate code with multiple files and more lines of code (LOC). However, it is recommended to use GPT-4o first, as the o1 models have prompt limit issues. |
| less loc/single file | Claude 3.5 Sonnet | We can use sonnet model for files which are having a smaller number of codes |
| 6 | Locator generation | all the possible locator generation for the given Dom | GPT 4o | All the models provide better locators. We can use GPT-4o as it does not have prompt limit issues. |
| 7 | Explain this | complex scenarios/more loc code | 1. o1-mini 2. o1- preview | Provides a detailed explanation without leaving any line of code, however, for large sections, we need to ask separately. Initially, we can use o1-mini, which has a 50-prompt limit, compared to o1-preview’s 10-prompt limit. |
| simple scenarios/less loc code | GPT 4o | Explains the entire code line by line. However again for large section we must ask separately. |
| 8 | Unit Testcases | More code coverage | 1. o1- preview 2. o1- mini | All models provide unit test cases that increase coverage. Among them, o1-preview generates the most coverage, followed by o1-mini. |
| less code coverage | 3. GPT 4o 4. Claude 3.5 Sonnet | All models provide unit test cases that increase coverage. However, compared to the o1 models, GPT-4o generates less coverage, followed by Claude 3.5. |

# **Model prioritization and Remarks :**

|  |  |
| --- | --- |
| Model name | Remarks |
| o1- preview | There are only 10 questions we can ask, after 10 questions user must wait for 8 hrs. Comparatively the responses are more efficient with respect to o1 mini. speed of output: 20-30 sec  o1-preview is a better choice for test automation involving highly complex scenarios that benefit from in-depth reasoning. |
| o1-mini | There are only 50 questions we can ask, after 50 questions user must wait for 8 hrs.  speed of output: 15-25 sec o1-mini is generally the better choice for most test automation needs due to its speed and cost-efficiency. |
| GPT 4o | There is no limit for the questions  speed of output: 2-5 sec |
| Claude 3.5 Sonnet | There is no limit for questions but, Multiple issues are encountered with this model concerning testing scenarios.   speed of output: 3-7 sec |

\*\*\*\*\*End of Document\*\*\*\*\*