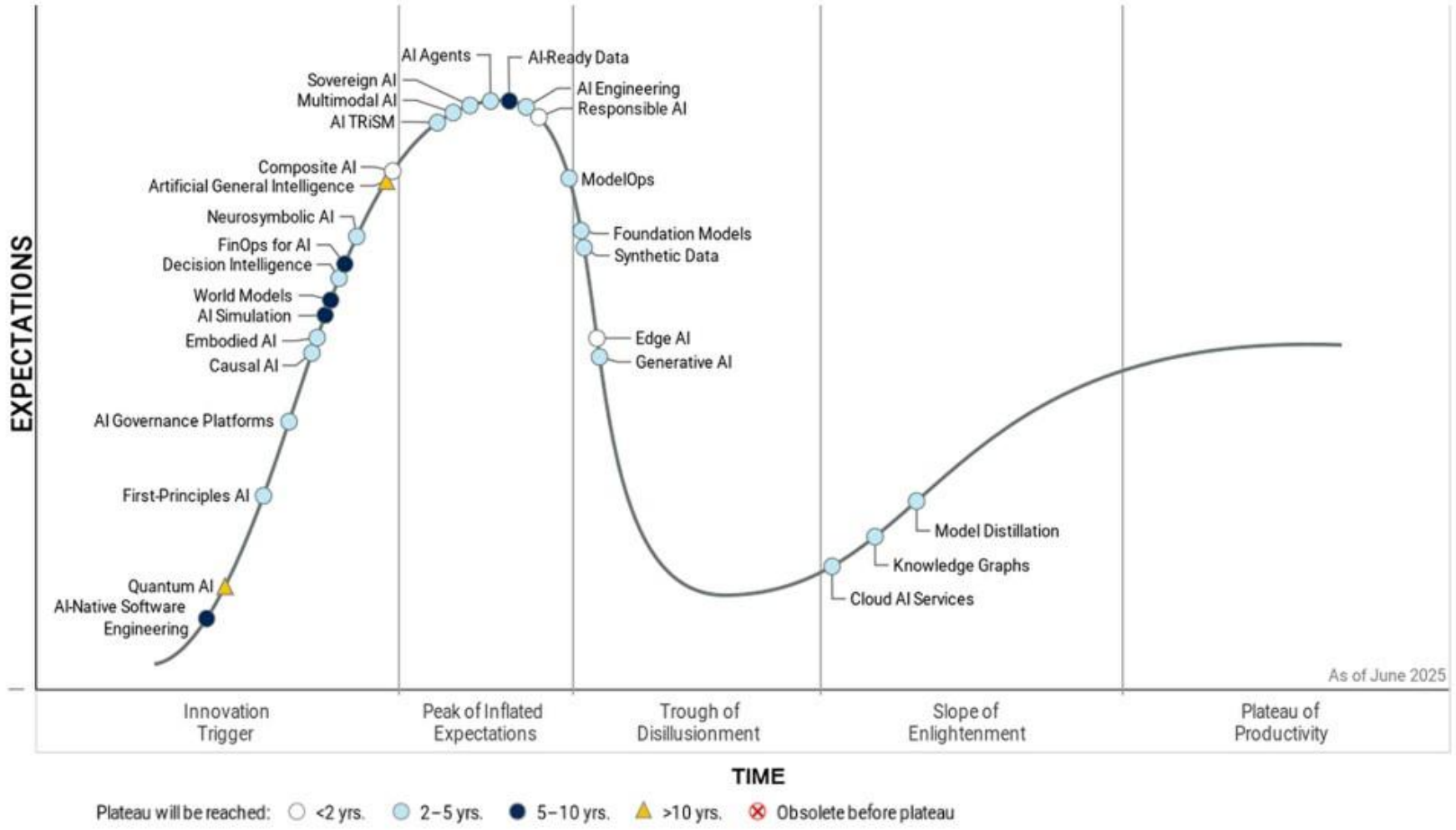
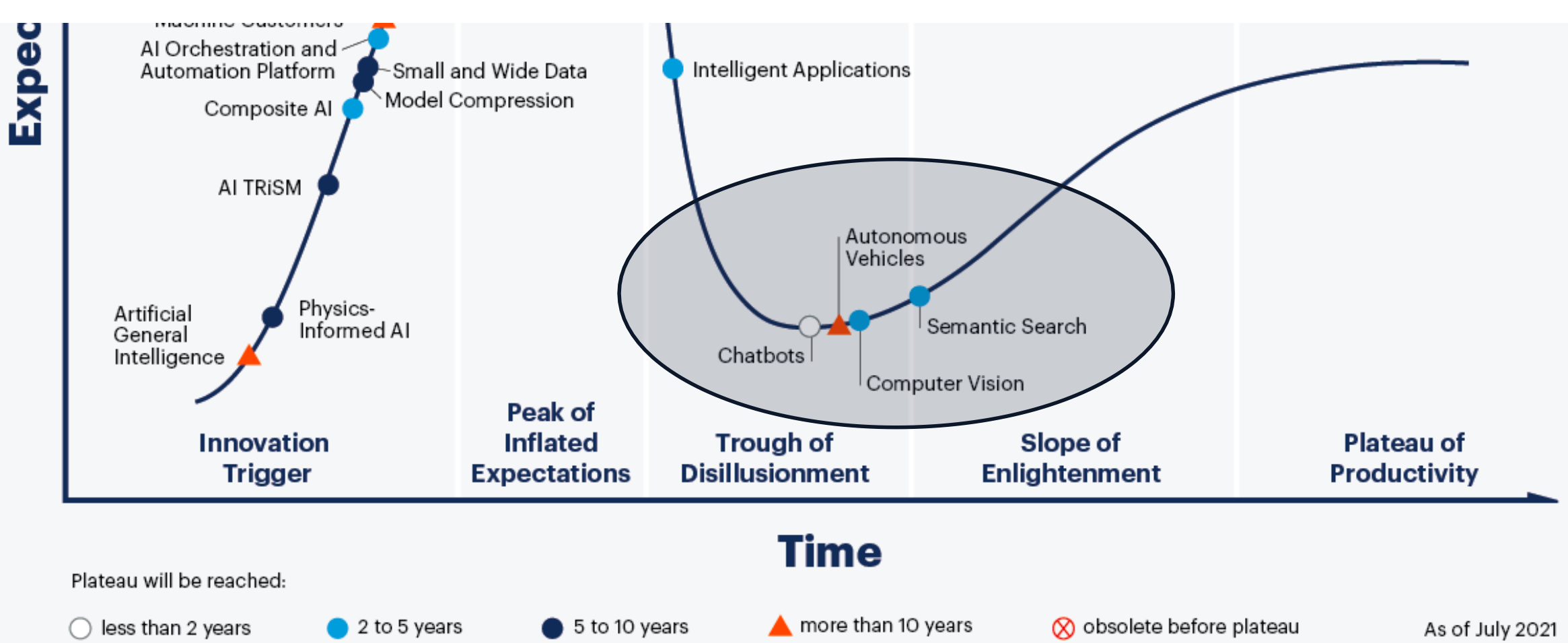
LLMs in Softwrae Testing

Gartner's Hype Cycle for AI, 2025 report



Gartner



gartner.com

Source: Gartner

Gartner®

https://www.gartner.com/en/articles/the-4-trends-that-prevail-on-the-gartner-hype-cycle-for-ai-2021

Limitations
Be Careful

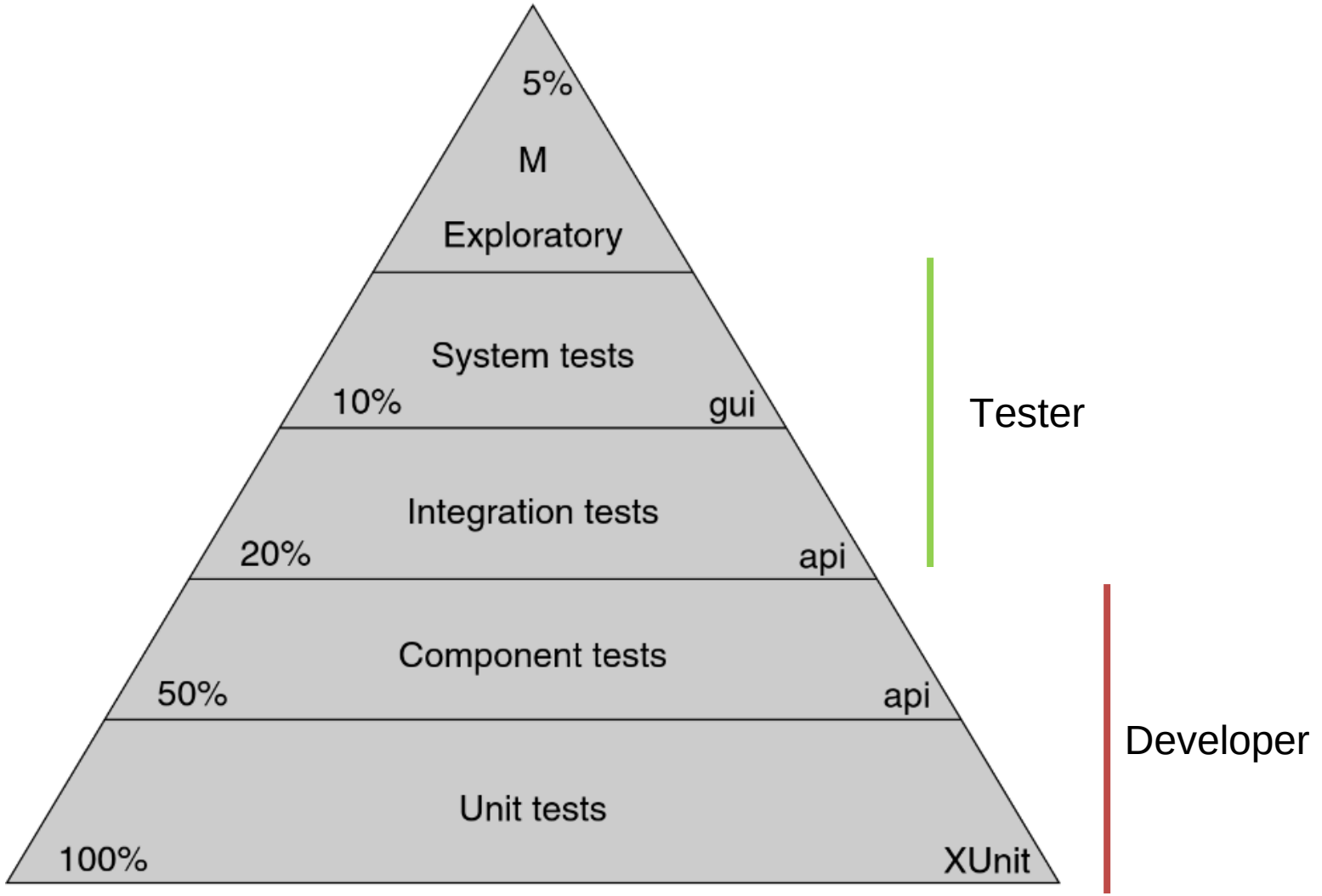
- **Never believe output from AI without verifying it!**

- AI is known to **hallucinate** (i.e. make up facts and references)
- Verify with knowledge you already know.
- Verify by using your textbook or other academic resources (e.g., the Library website).
- Google information if you are unsure.

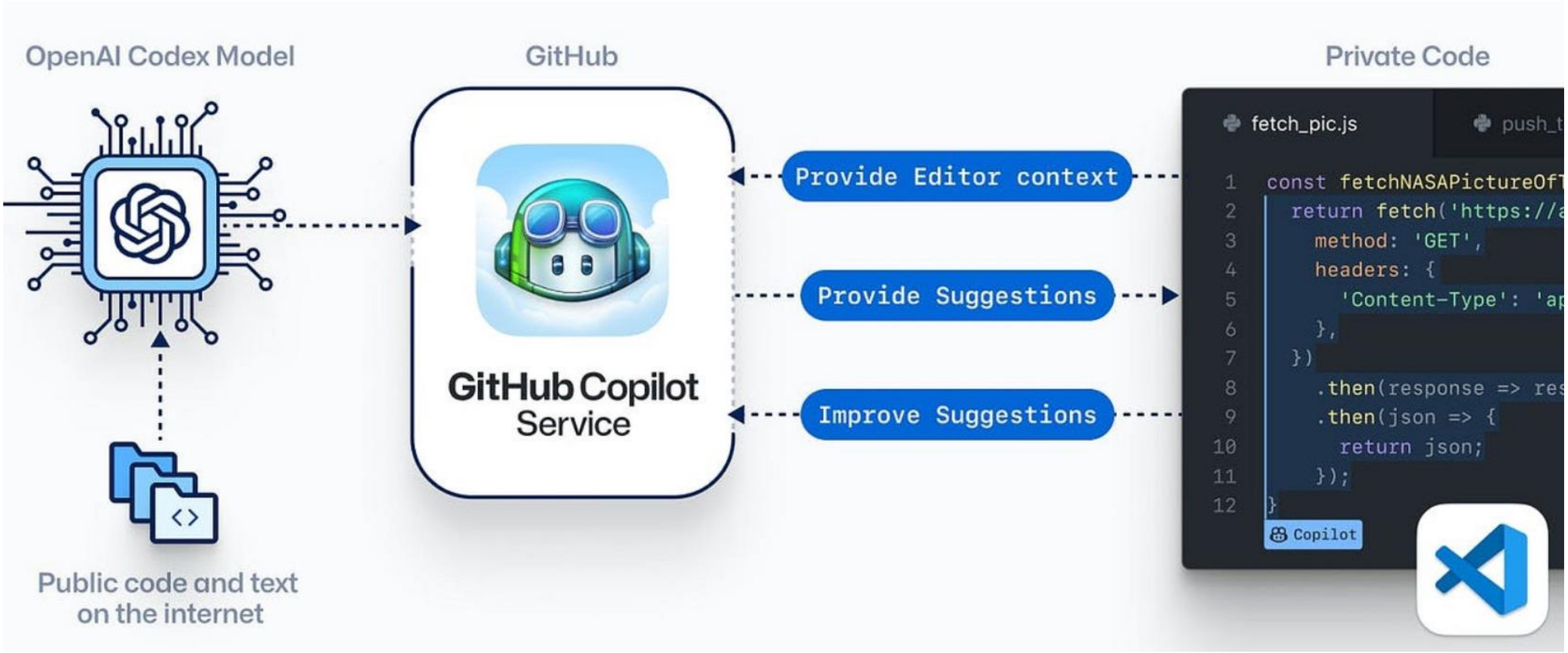
Large Language Models
Transformers

- Enable models to handle long-range dependencies in text.
- Rich context could be determined and considered.
- Not only useful for NLP
  - **Machine translation**: automatic speech translation, code translation, …
  - **DNA sequence analysis**: predict the effects of genetic mutations, …
  - **Protein structure analysis**: model long chains of amino acids that fold into complex protein structures

Software Testing Pyramid



Github Copilot



Github Copilot

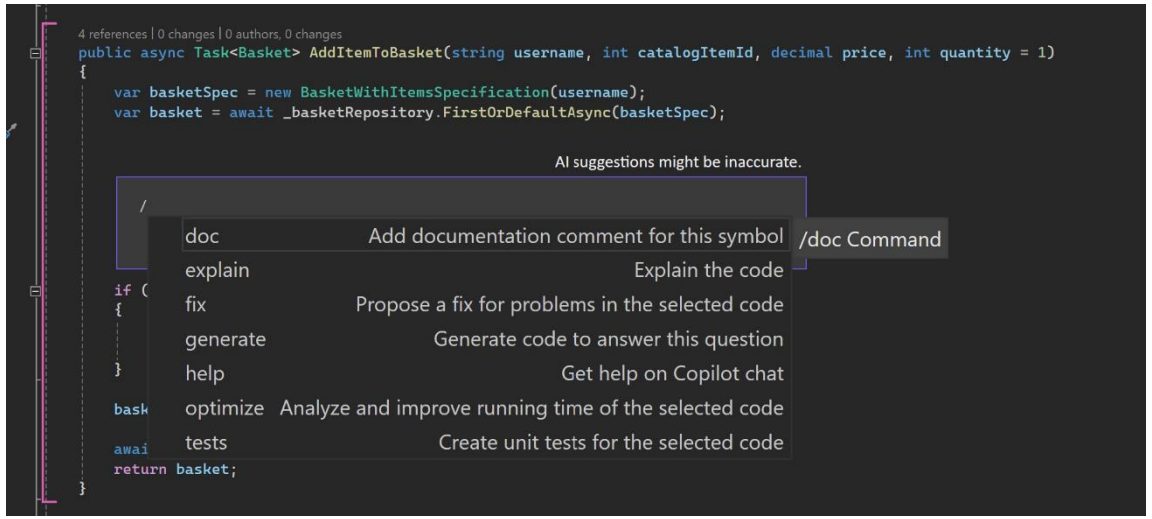
Models and Comparison:

Model name	Provider	Release status	Agent mode	Ask mode	Edit mode
GPT-4.1	OpenAI	GA	✓	✓	✓
GPT-5	OpenAI	GA	✓	✓	✓
GPT-5 mini	OpenAI	GA	✓	✓	✓
GPT-5-Codex	OpenAI	Public preview	✓	✓	✓
GPT-5.1	OpenAI	Public preview	✓	✓	✓
GPT-5.1-Codex	OpenAI	Public preview	✓	✓	✓
GPT-5.1-Codex-Mini	OpenAI	Public preview	✓	✓	✓
GPT-5.1-Codex-Max	OpenAI	Public preview	✓	✓	✓
GPT-5.2	OpenAI	Public preview	✓	✓	✓
Claude Haiku 4.5	Anthropic	GA	✓	✓	✓
Claude Opus 4.1	Anthropic	GA	✗	✓	✓
Claude Opus 4.5	Anthropic	Public preview	✓	✓	✓

See https://docs.github.com/en/copilot/reference/ai-models/model-comparison for comparison

Github Copilot

Commands:



Full list of commands in *labs/Resources*

Github Copilot

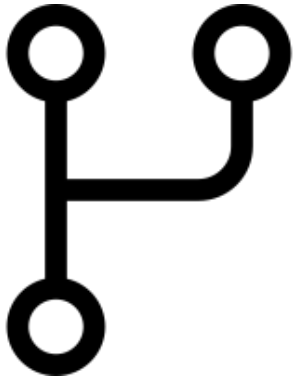
Context:

Current File

The entirety of the file, not just where the cursor is

Tabs Open in Editor

Tabs closest to the active tab given priority



Entire Codebase

Without any open file

Github Copilot

Best supported programming language:

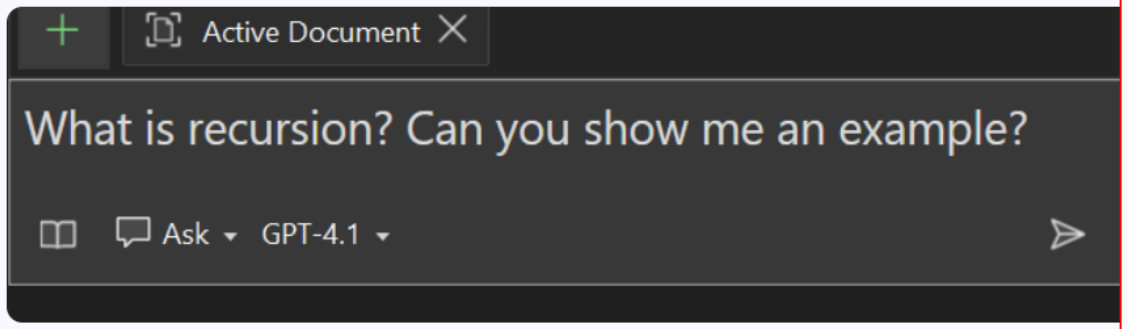
Language	GitHub Copilot
C	✓
C++	✓
C#	✓
Go	✓
Java	✓
JavaScript	✓
Kotlin	✓
PHP	✓
Python	✓
Ruby	✓
Rust	✓
Scala	✓
Swift	✓
TypeScript	✓

- Which one is the best supported programming language?
- *As more structed, as better supported!*

Github Copilot Agent vs. Ask Mode

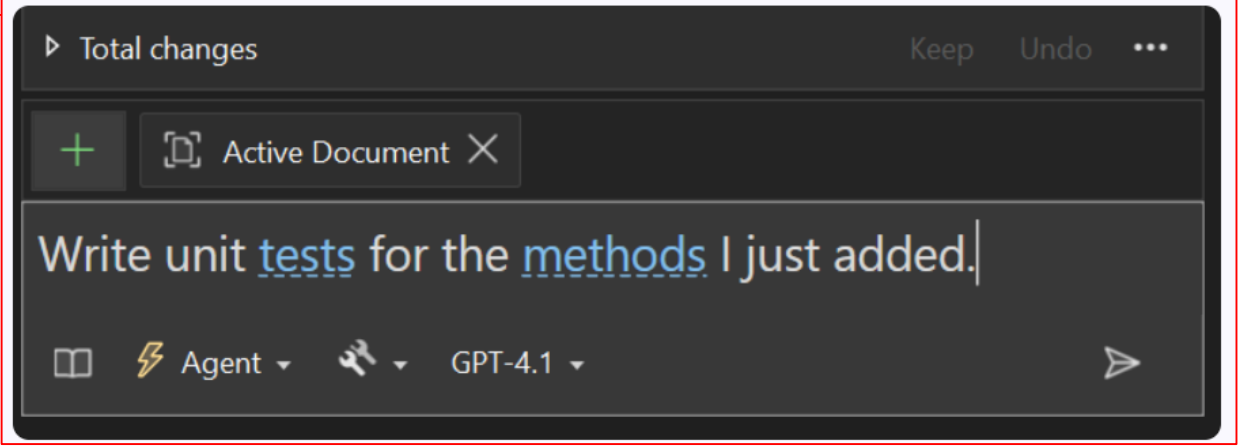
Understanding Ask Mode

Ask Mode is your go-to setting when you need quick, conversational support—think of it as asking an experienced developer for advice, troubleshooting, or code samples. In ask mode, Copilot Chat doesn't directly interact with your workspace files; instead, it provides responses based on the context you provide.



Understanding Agent Mode

Agent Mode takes things a step further by allowing Copilot Chat to act as an intelligent agent within your codebase. Here, Copilot can reason about your actual project files, execute commands, make edits, and even help refactor or generate new code directly in your solution.



Github Copilot Agent vs. Ask Mode

Feature	Ask Mode	Agent Mode
Workspace Scope	Current file & selection	Entire workspace
Primary Use	Learning & guidance	Code analysis & modification
Response Speed	Fast	May take longer (analyzes workspace)
Code Changes	Provides suggestions	Can make direct edits
Context Awareness	Active file & selection	Multi-file project context
Best For	Conceptual questions	Refactoring & automation

Github Copilot Agent vs. Ask Mode

Feature	Ask Mode	Agent Mode
Workspace Scope	Current file & selection	Entire workspace
Primary Use	Learning & guidance	Code analysis & modification
Response Speed	Fast	May take longer (analyzes workspace)
Code Changes	Provides suggestions	Can make direct edits
Context Awareness	Active file & selection	Multi-file project context
Best For	Conceptual questions	Refactoring & automation

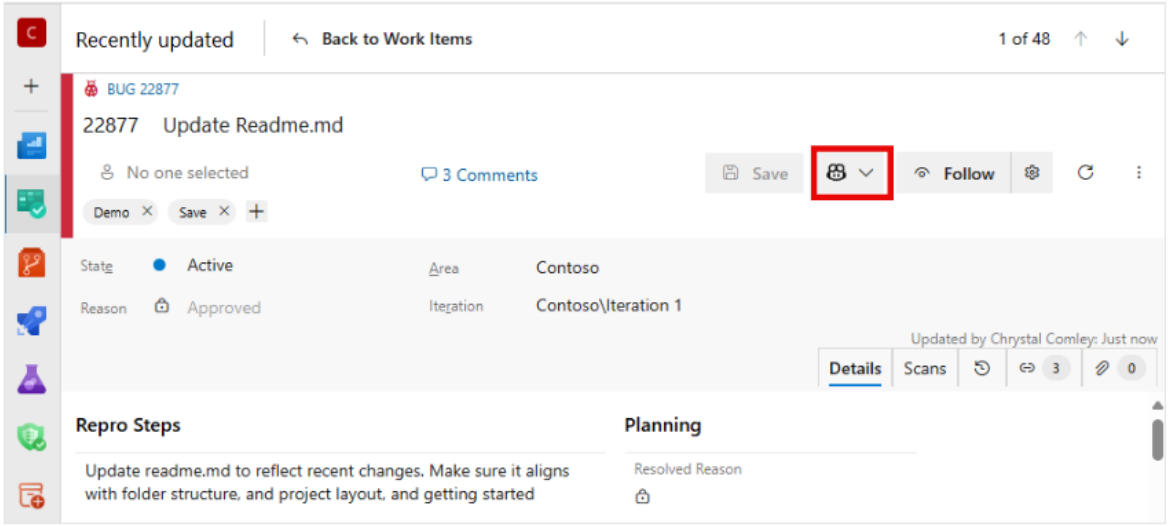
Github Copilot in Azure Boards

Use GitHub Copilot with Azure Boards

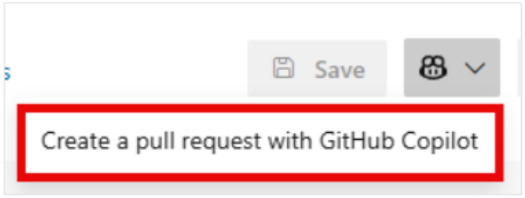
Azure DevOps Services

> **Note**
>
> This feature is in Private Preview. Access is limited and functionality might change before general availability.

If multiple options are available, the button becomes a dropdown menu with different choices.



3. Select **Create a pull request with GitHub Copilot**.



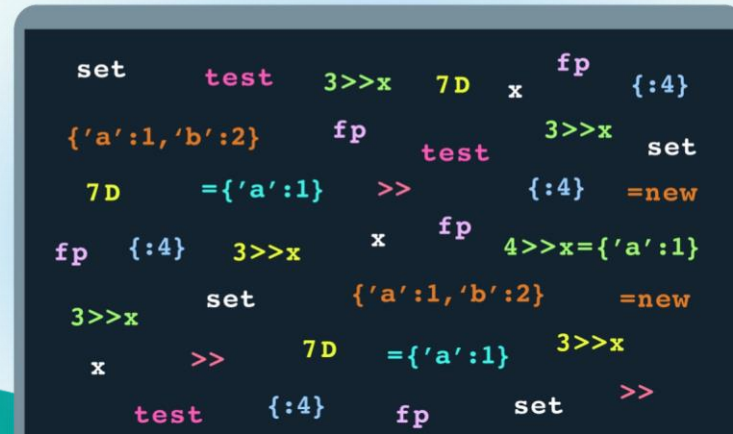
Prompt Engineering

We need to know how to use an LLM first!

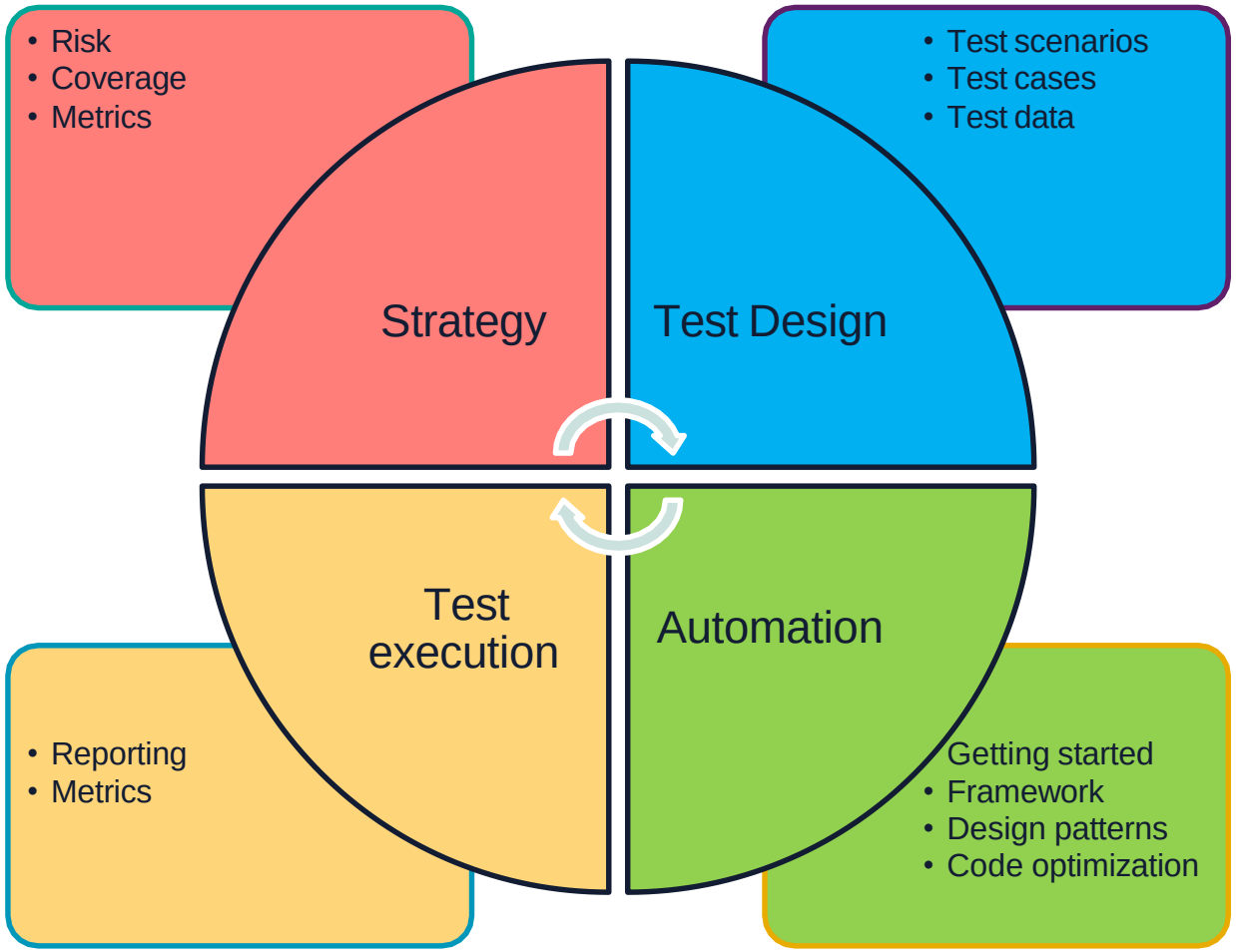
See „01 Prompt engineering best practises.pdf“

LLMs in Testing

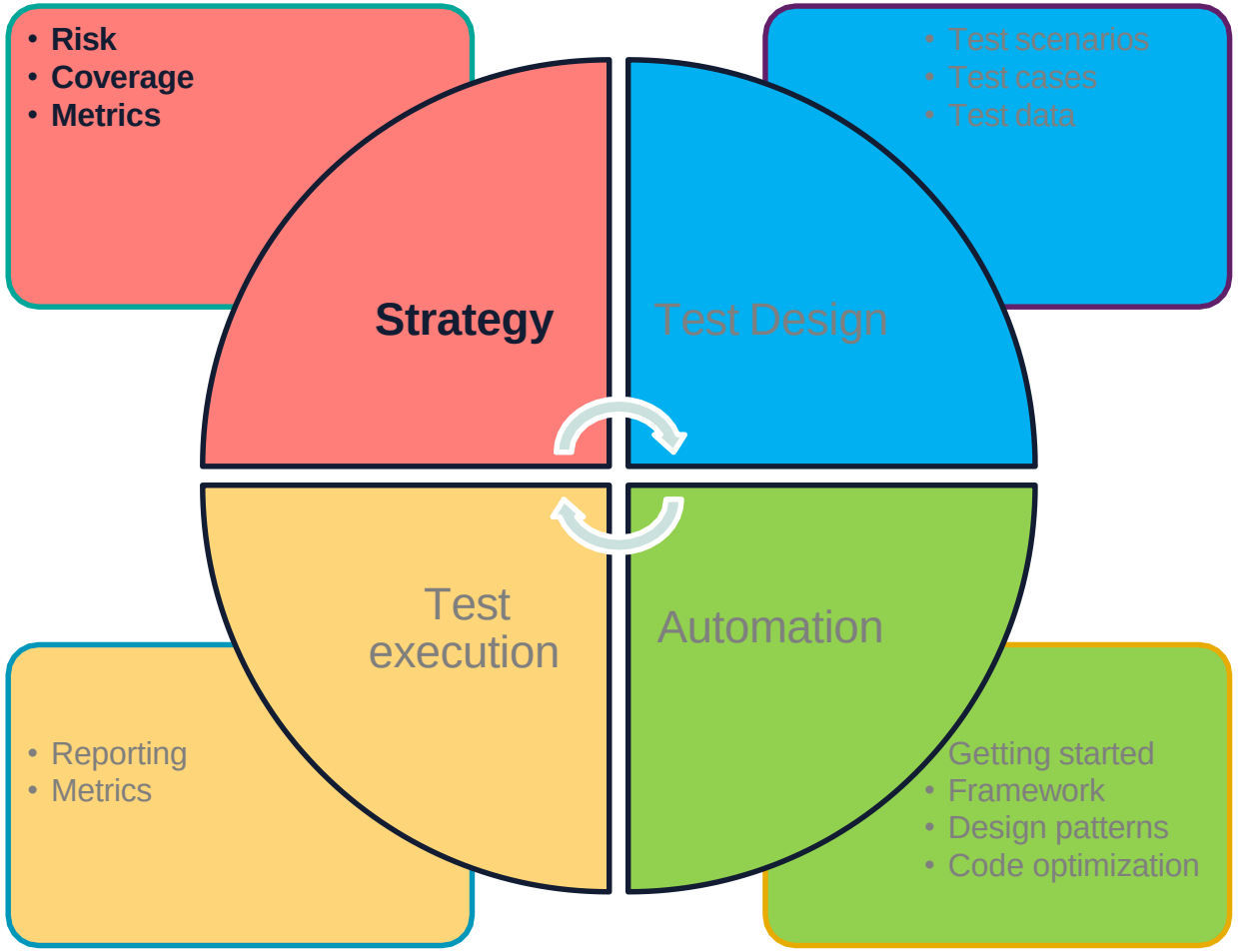
Example: Testing a Website



Testing Approach



Testing - Strategy



Used Application

Spring Petclinic:

- Frontend UI: http://ec2-54-188-50-153.us-west-2.compute.amazonaws.com:8000/
- Backend API: http://ec2-54-188-50-153.us-west-2.compute.amazonaws.com:9966/petclinic/v3/api-docs

Testing - Strategy

Question asked:

Create a test strategy for

testing and automating

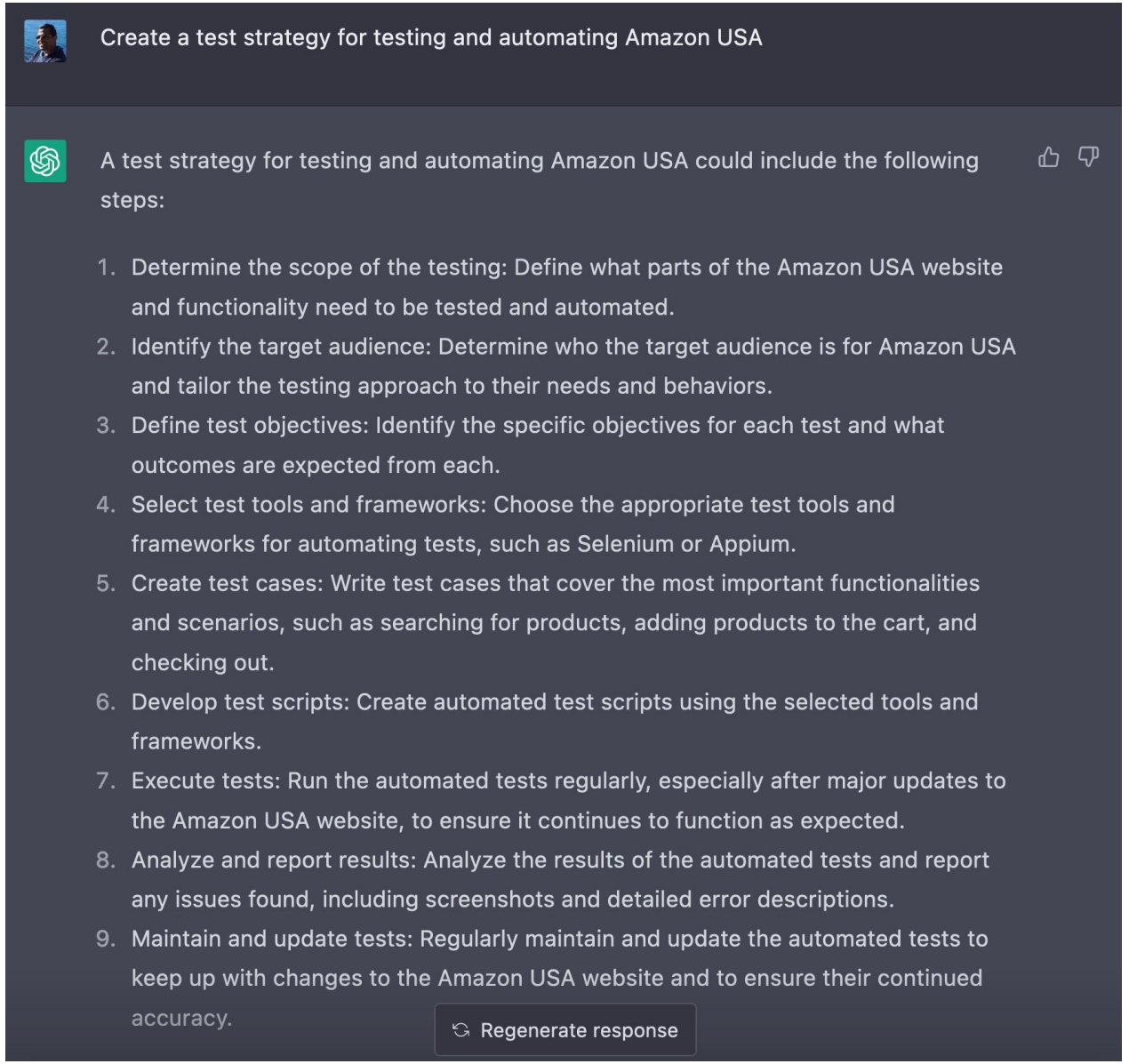
the website under

<http://..>

Testing - Strategy

Question asked:

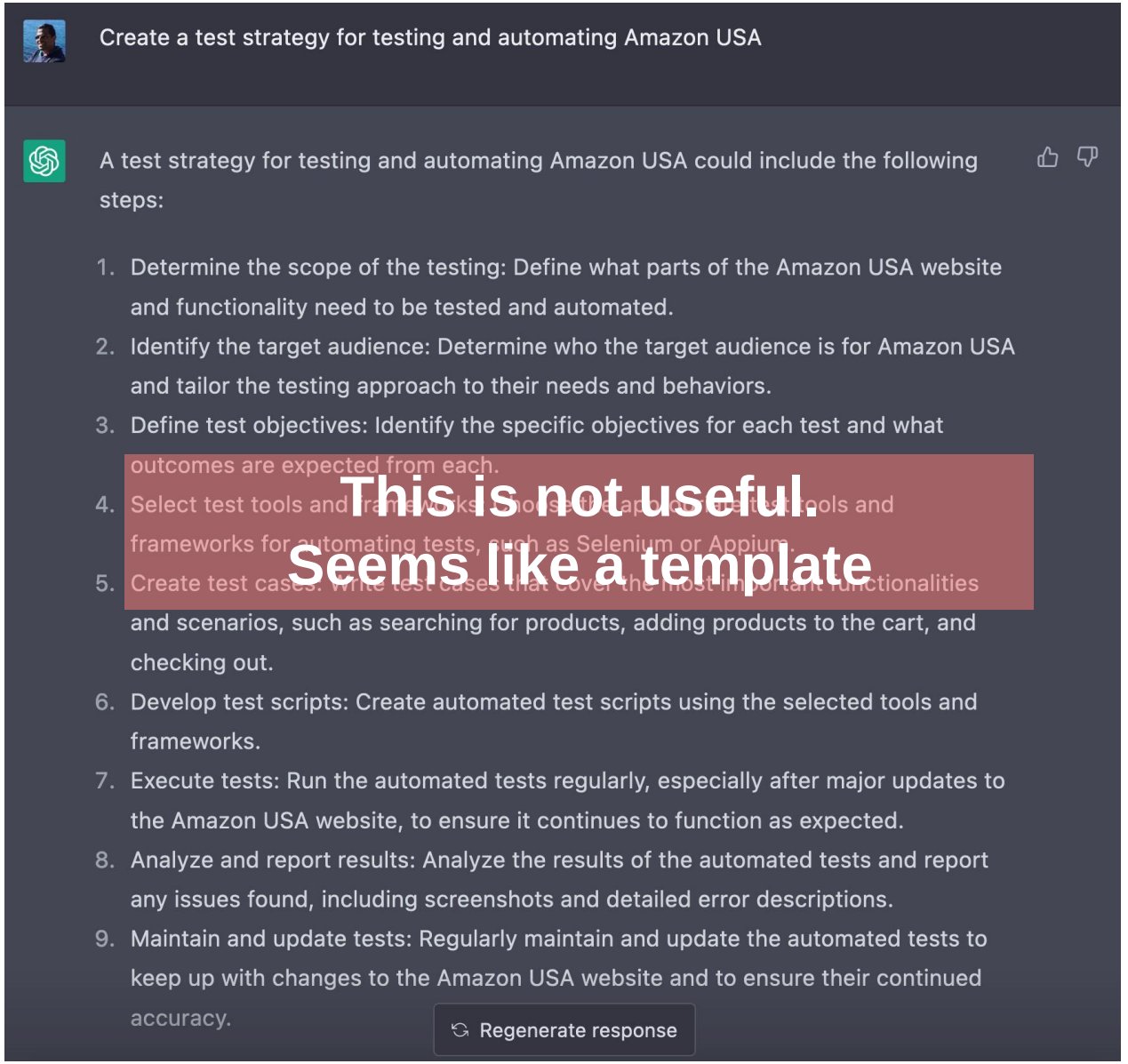
Create a test strategy for testing and automating Amazon USA



Testing - Strategy

Question asked:

Create a test strategy for testing and automating Amazon USA



This is not useful.

Seems like a template

Testing - Strategy

Follow-up question:

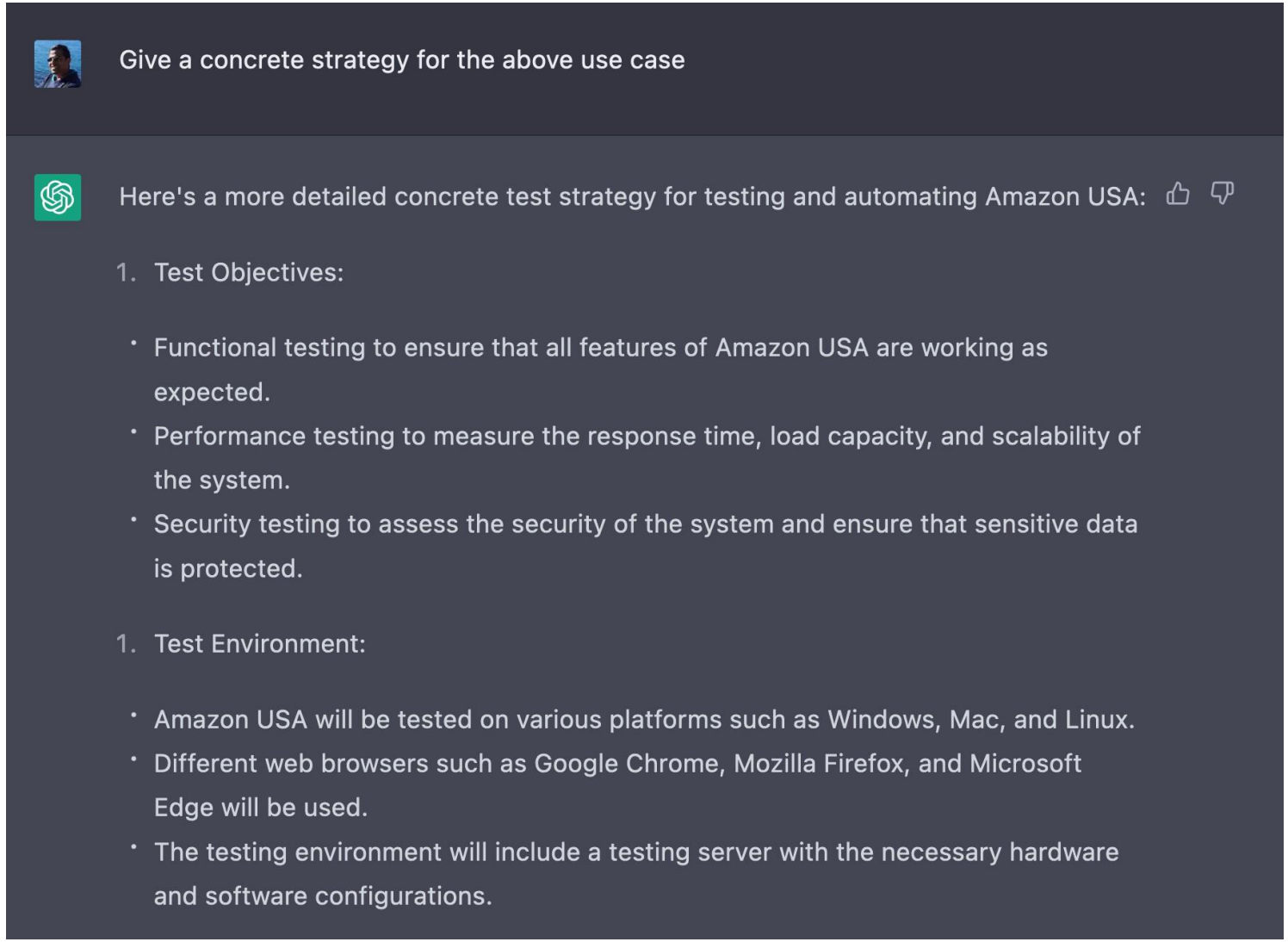
Access the page http:// …

and give a concrete strategy

for the above use case.

Testing - Strategy

Follow-up question: *Access the page http:// ... and give a concrete strategy for the above use case.*



Testing - Strategy

Follow-up question: *Access the page http:// ... and give a concrete strategy for the above use case.*

1. Test Cases:

- A comprehensive set of test cases will be created to cover all functionality of Amazon USA, including but not limited to:
  - User registration and login
  - Product search and filtering
  - Adding products to the cart
  - Checkout and payment
  - Order tracking
  - Customer service
- Negative scenarios, such as incorrect inputs and invalid operations, will also be tested.

1. Test Data:

- Test data will be created or sourced from existing data to support functional testing.
- Test data will also be used for performance testing to simulate real-world conditions.

1. Test Automation:

- Selenium WebDriver will be used for automating the functional tests.
- Apache JMeter will be used for performance testing.

Testing - Strategy

Follow-up question: *Access the page http:// ... and give a concrete strategy for the above use case.*

1. Test Execution:

- The test cases will be executed in a phased approach, starting with functional testing, followed by performance testing, and finally security testing.
- The test execution will be done in a controlled environment and the test results will be recorded and analyzed.

1. Test Metrics:

- Test coverage, test success rate, and test execution time will be measured and used to evaluate the effectiveness of the testing process.
- Performance metrics such as response time, throughput, and scalability will be recorded and analyzed.

1. Test Reporting:

- Test results will be reported using JIRA, including both pass and fail results.
- Test results will be analyzed and any bugs or issues will be reported and tracked.

1. Test Maintenance:

- Test cases, test data, and test results will be regularly reviewed and updated.
- Outdated or irrelevant test cases will be archived.

This strategy provides a systematic approach for testing and automating Amazon USA, ensuring that all features are thoroughly tested and any issues are identified and resolved in a timely manner.

This is promising.
A great start!

Lab

- Ask copilot to describe the content of the website first.
- Create your own test strategy
- We use **lab-s02** (fresh copy of s01)

Testing – Strategy with Risk Mitigation

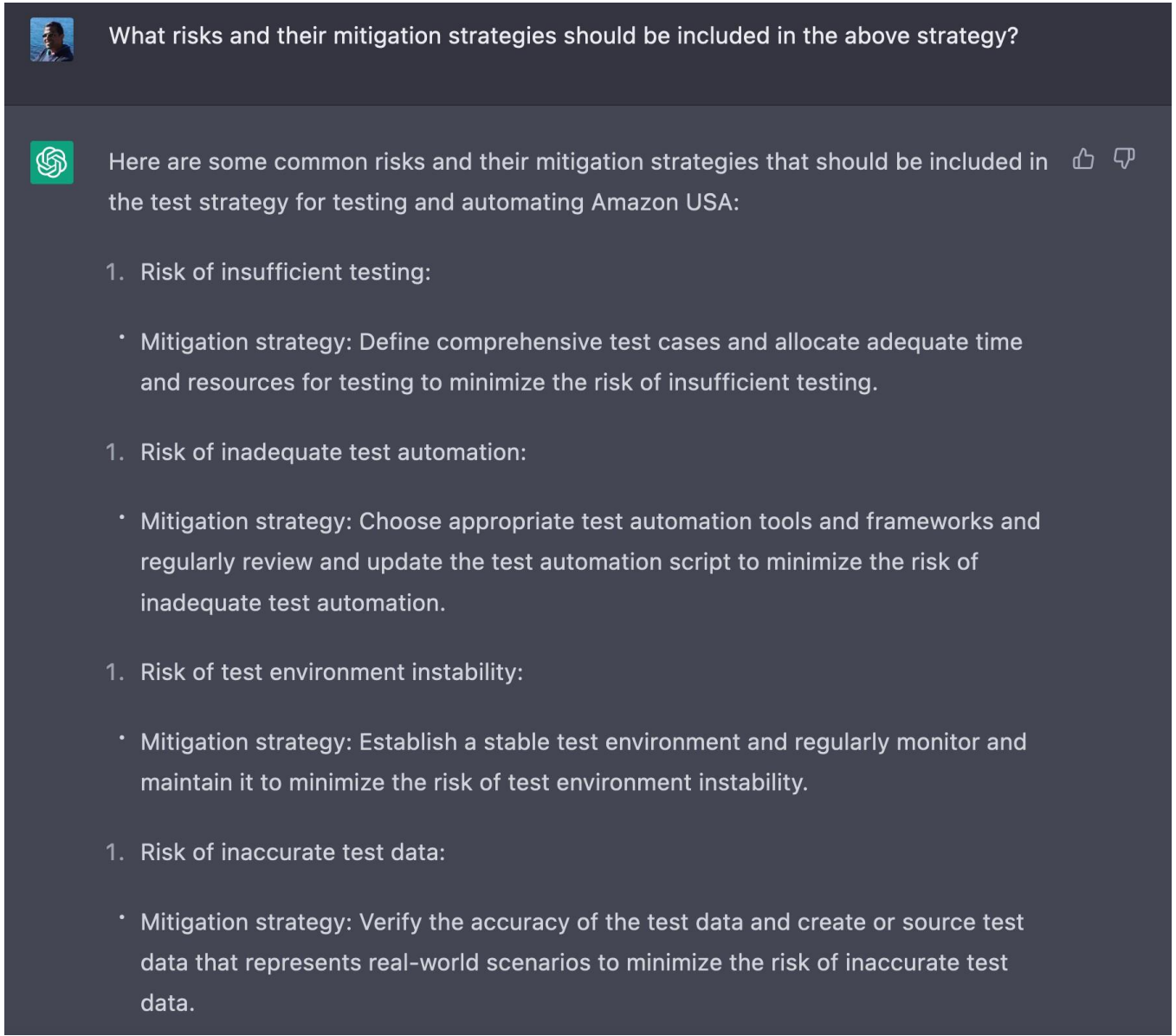
Next question:

What risks and their mitigation strategies should be included in the above strategy?

Testing – Strategy with Risk Mitigation

Next question:

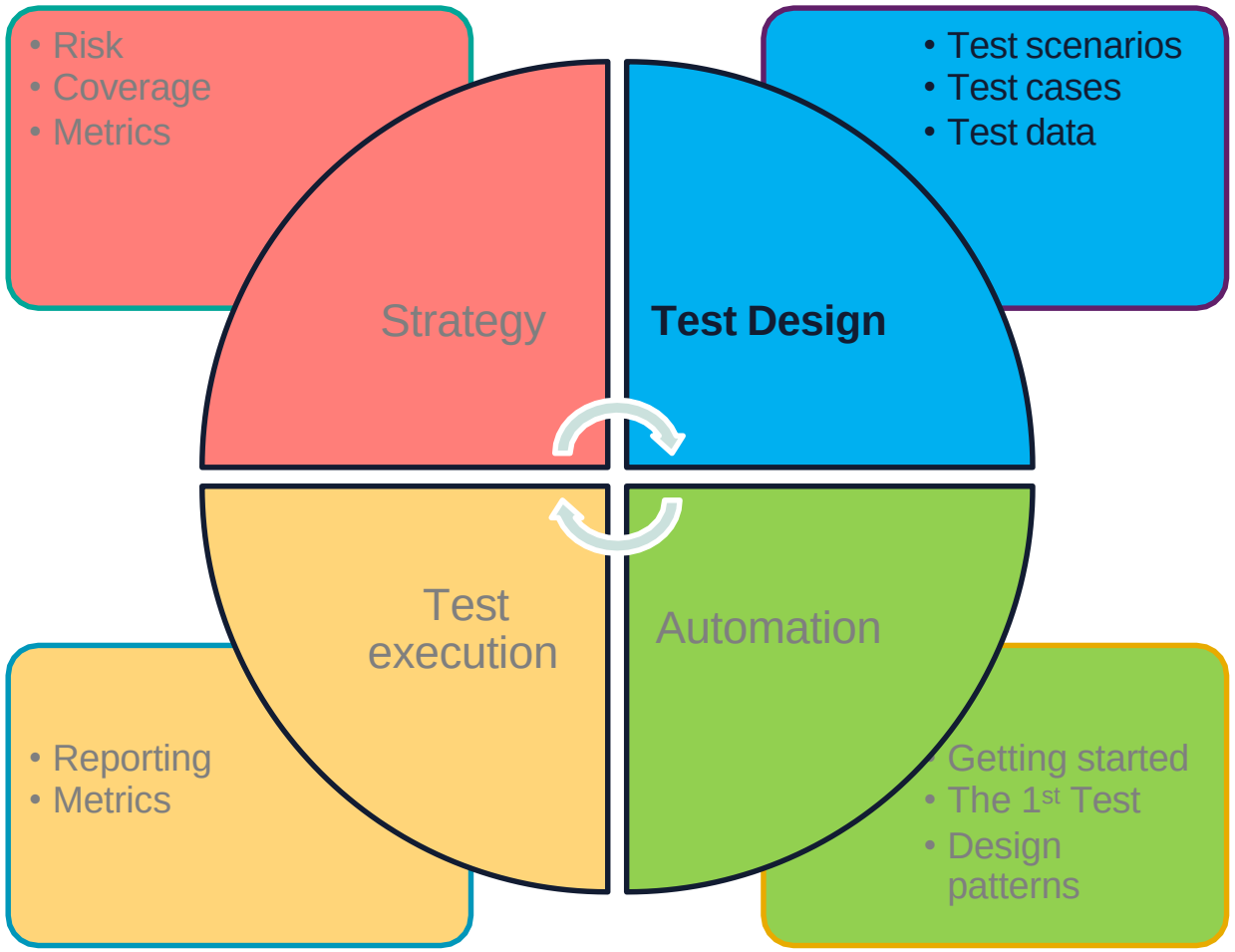
What risks and their mitigation strategies should be included in the above strategy?



Lab

Add risk migitation to your strategy

Testing – Test Design



Test Design – Test Scenarios

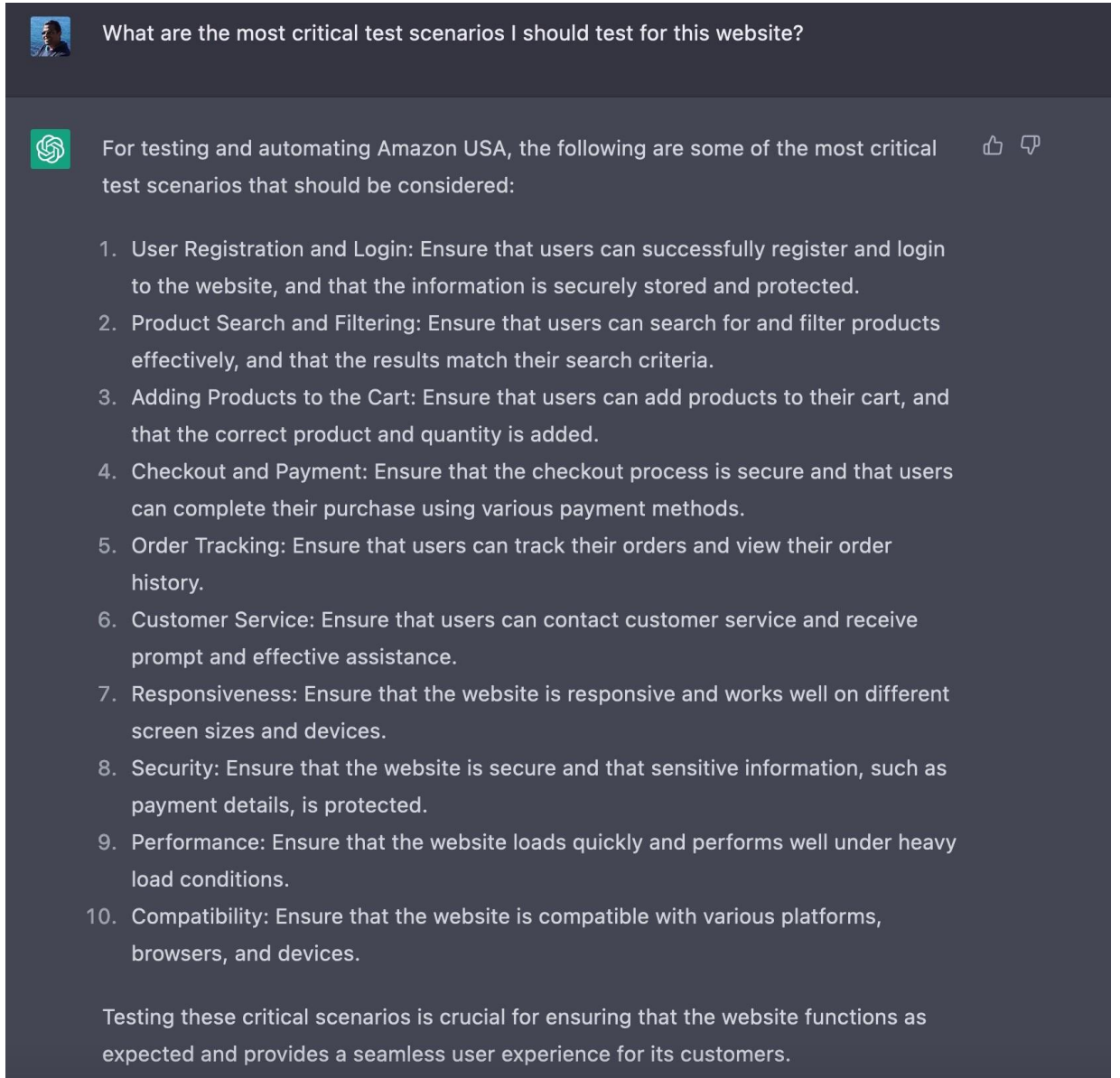
Next question:

What are the most critical test scenarios I should test for this website?

Test Design – Test Scenarios

Next question:

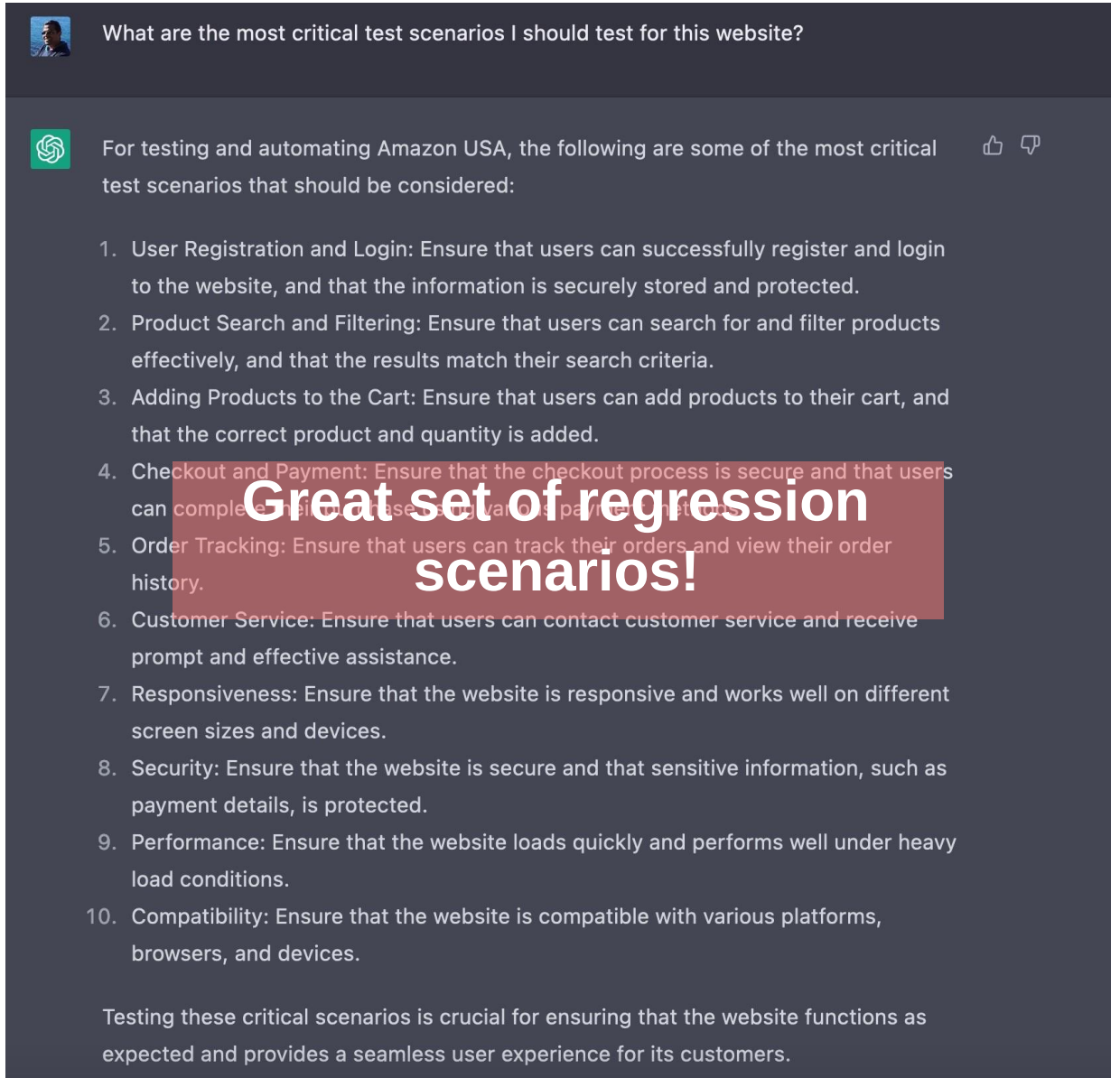
What are the most critical test scenarios I should test for this website?



Test Design – Test Scenarios

Next question:

What are the most critical test scenarios I should test for this website?



Lab

Define critical test scenarios and cases for your tests

Test Design – Test Cases

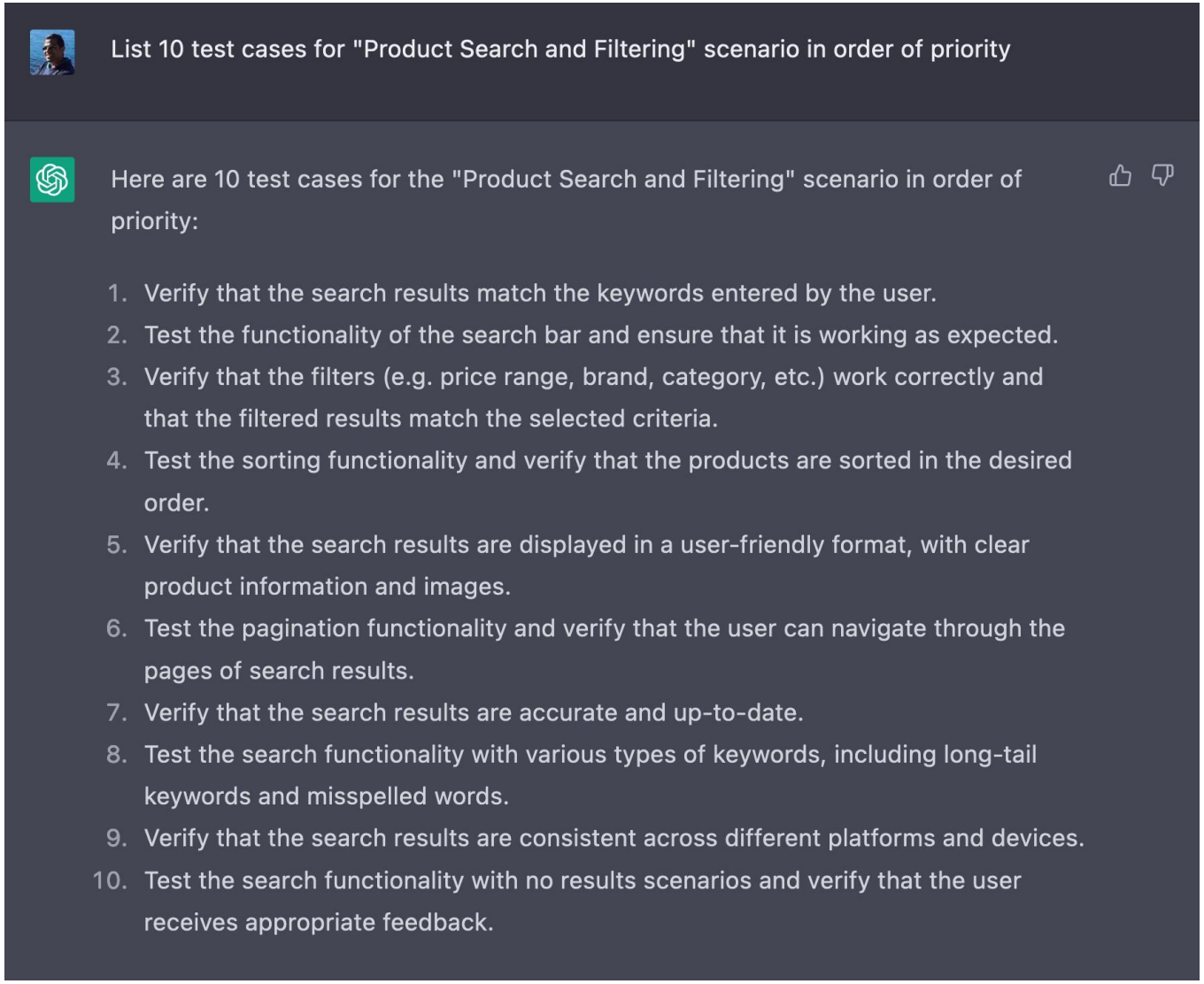
Next question:

List 5 tests for the most critical test cases in order of priority.

Test Design – Test Cases

Next question:

List 5 tests for the most critical test cases in order of priority.



Lab

- Use your LLM to create test descriptions.

- Try to guide the LLM to define test boundaries/edge cases for the most critical tests.

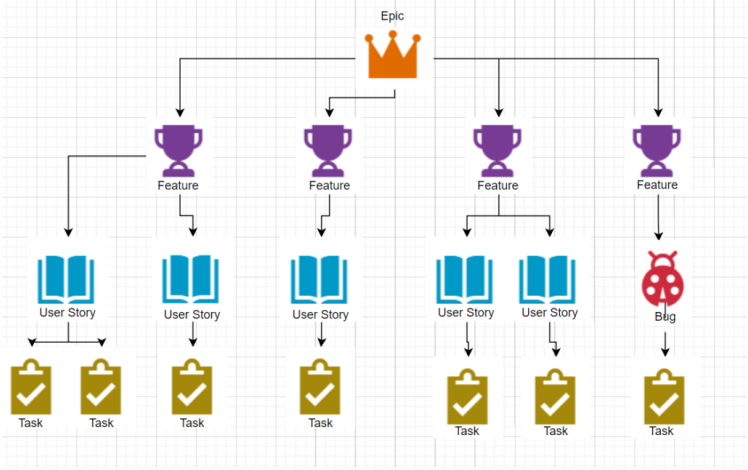
Test Design – Test Cases

What can we do with the tests from within the Chat?

Implement them (already done before)!

Test Design – New Feature

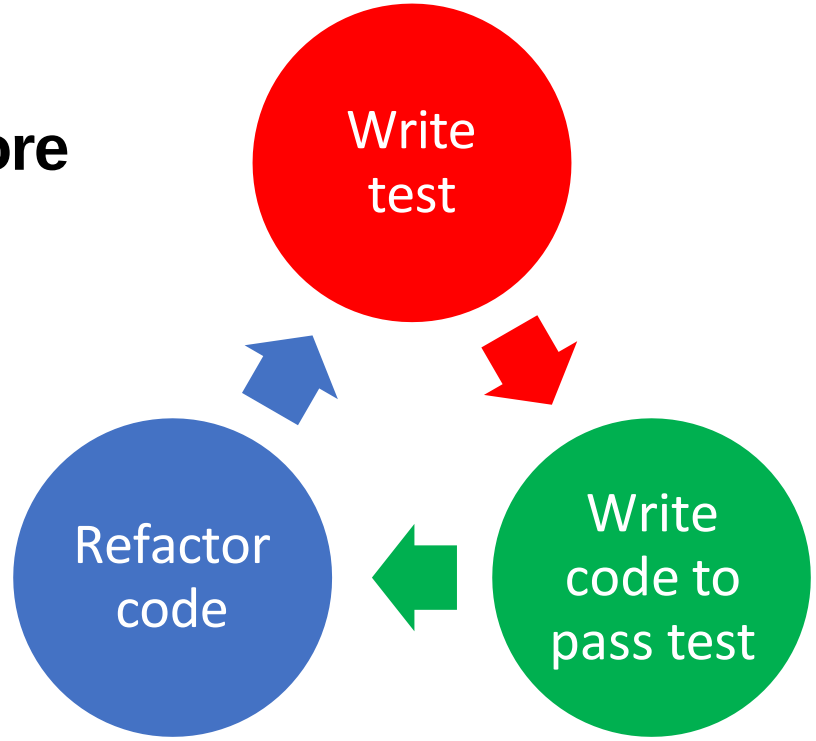
- Existing code until now!
- Tests for features before the implementation has been started.
- Test driven development (TDD) is possible with Copilot as well.



Test Driven Development

Test driven development (TDD):

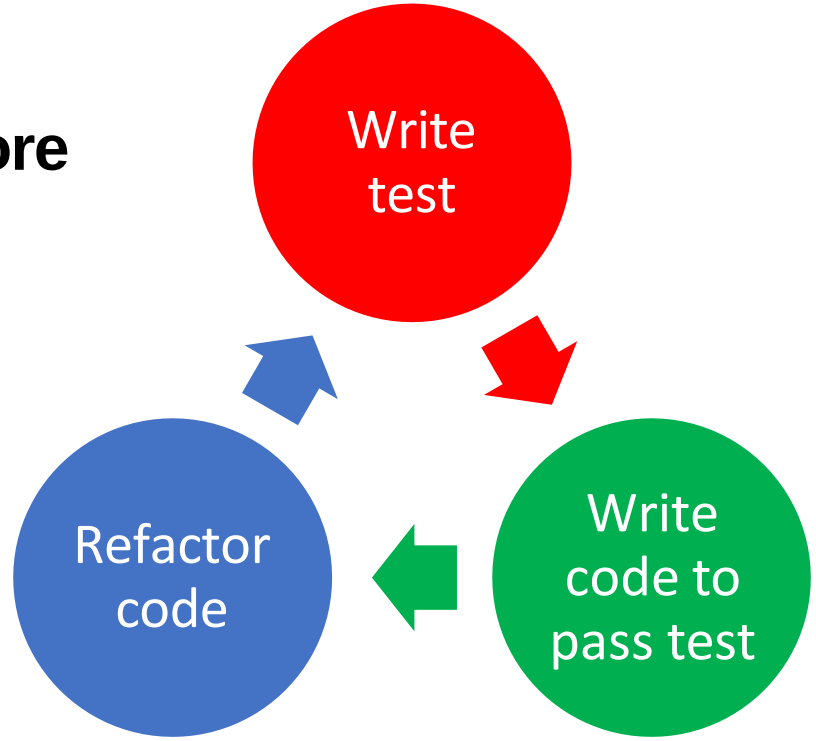
- What:
  - Test is based on the spec and is developed **before** the code is written
  - Will fail initially
  - Write just enough code to make it pass!
- Why?



Test Driven Development

Test driven development (TDD):

- What:
  - Test is based on the spec and is developed **before** the code is written
  - Will fail initially
  - Write just enough code to make it pass!
- Why?
  - In practice, significantly less defect rate
  - Improved understanding of requirements and ability to influence design
  - Not influenced by implementation choices



Lab

- Use **lab-s4** which is a **fresh copy of lab-s1.**

- Open and read *requirements_user_registration.md* first.
- Ask Copilot to create a new module in Gherkin language for the tests and add also the required steps to the solution.

- Evaluate the results.

Test Data Generation

- LLMs have strong capabilities to generate coherent text.
- Can be useful for generating data which mimic realistic test cases.

- Ask what data is required first.
- We can use what we have learned before:
- Sentiment analyse,
- Few-Shot Prompting by providing sample data,
- Case or edge case identification
- …

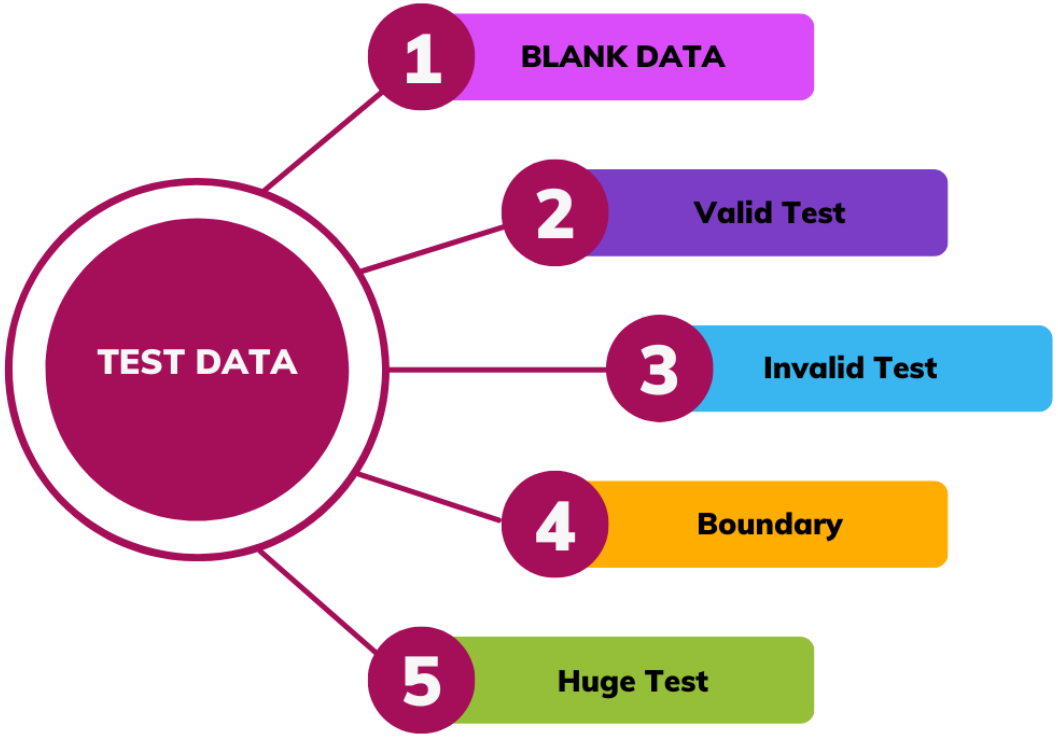
Test Design – Test Data

Test Data Sets

Exceptional Data (Out of Range Data)

Exceptional data is used to test that the program can properly handle unexpected inputs.

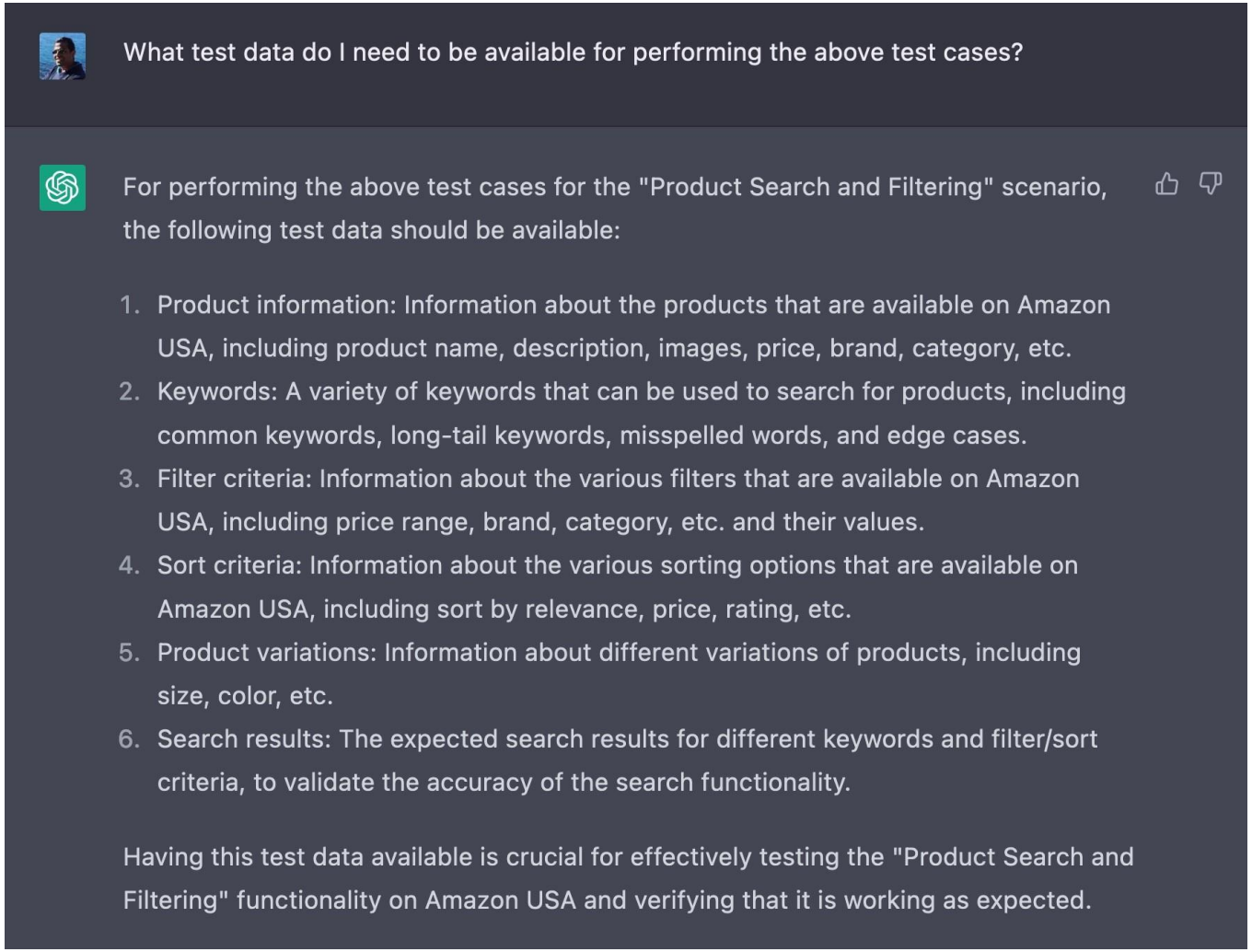
Employee Name	Employee Age	Expected Output	Actual Output	Pass/Fail
Kris Blair	1000	Invalid Employee	Invalid Employee	✓
Julie Harris	-1	Invalid Employee	Invalid Employee	✓
Rick Thomas	Nineteen	Invalid Employee	Invalid Employee	✓
Annette Reece	abc	Invalid Employee	Invalid Employee	✓



Test Design – Test Data

Next question:

What test data do I need to be available for performing the above test cases?

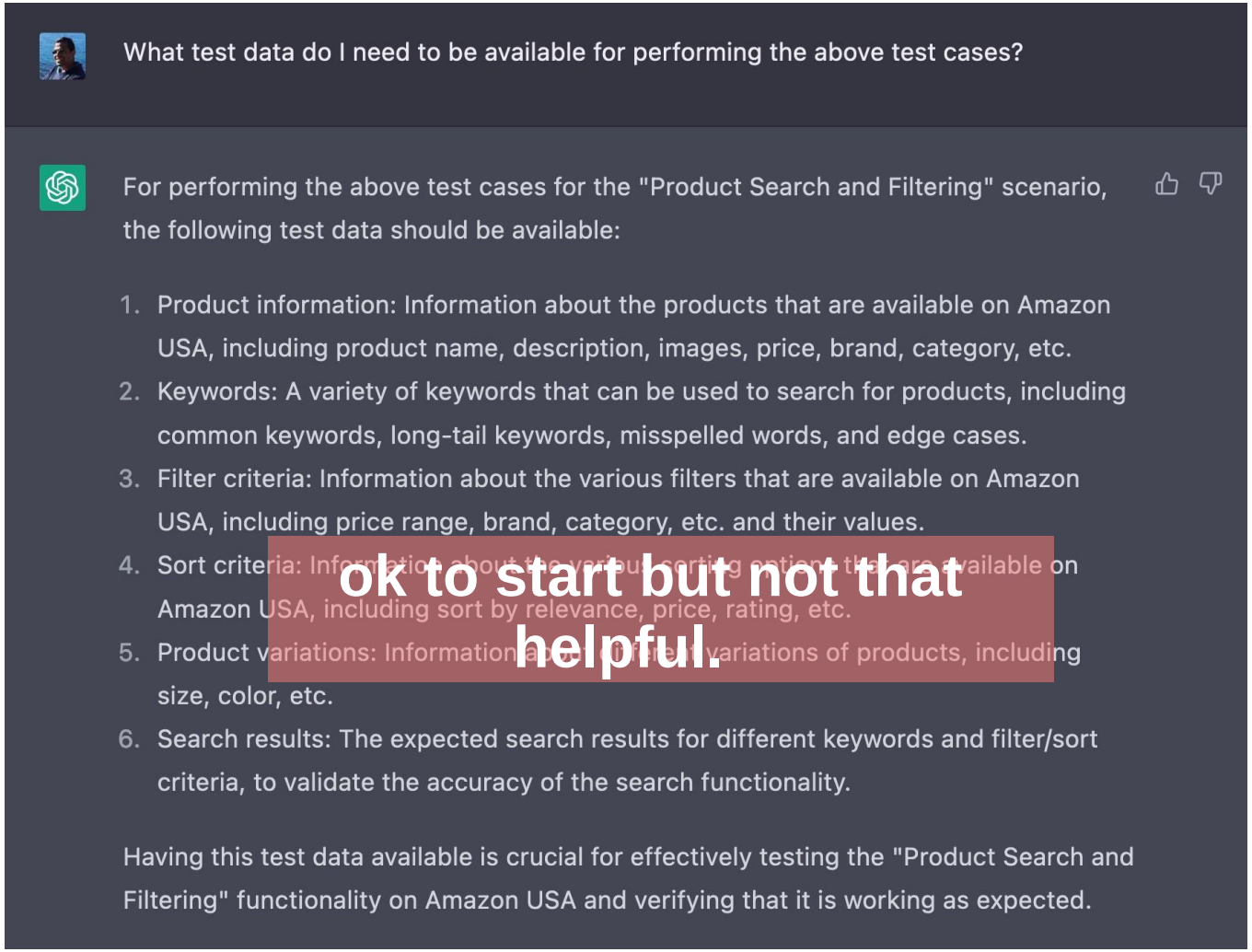


Test Design – Test Data

Next question:

What test data do I need to be available for performing the above test cases?

-> be more precise



Lab

- Use the "*Accommodation Booking"* from the reference application here and generate 20 test data examples.
- Use **lab-s5** which is a **fresh copy of lab-s1.**

1. Let Copilot summarize the purpose of the page first.
2. After that, tell Copilot to generate test data in a new file.

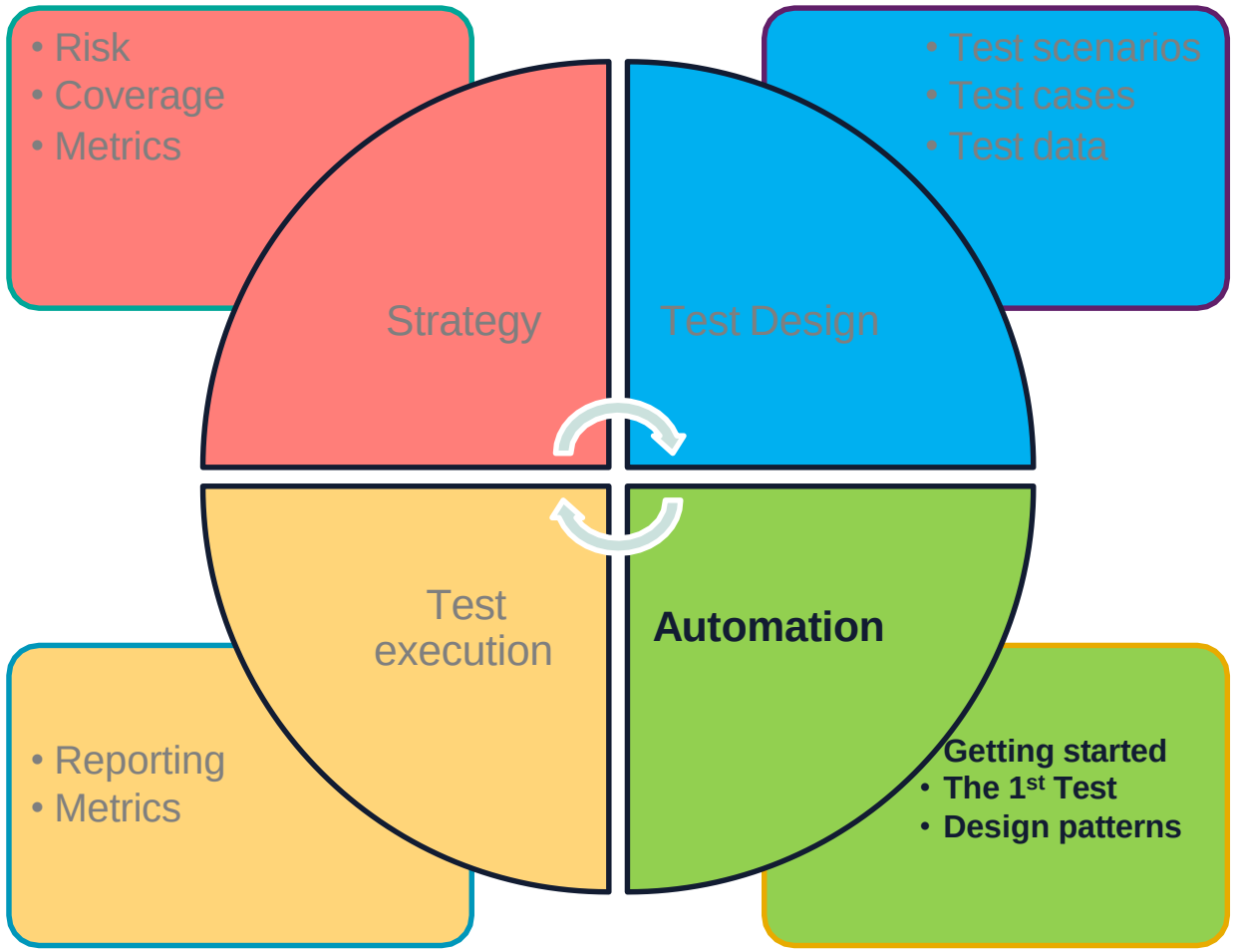
a) Take at least 10 edge cases into account.

a) Create data for positive and negative cases as well.

- If the data is too simple, re-ask and use the techniques learned before.

- ***Hint:*** Use Copilot to generate BDD-Tests in Gherkin language and the required steps that cover the generated test data (ensure completeness).

Testing - Automation



Automation – Strategy

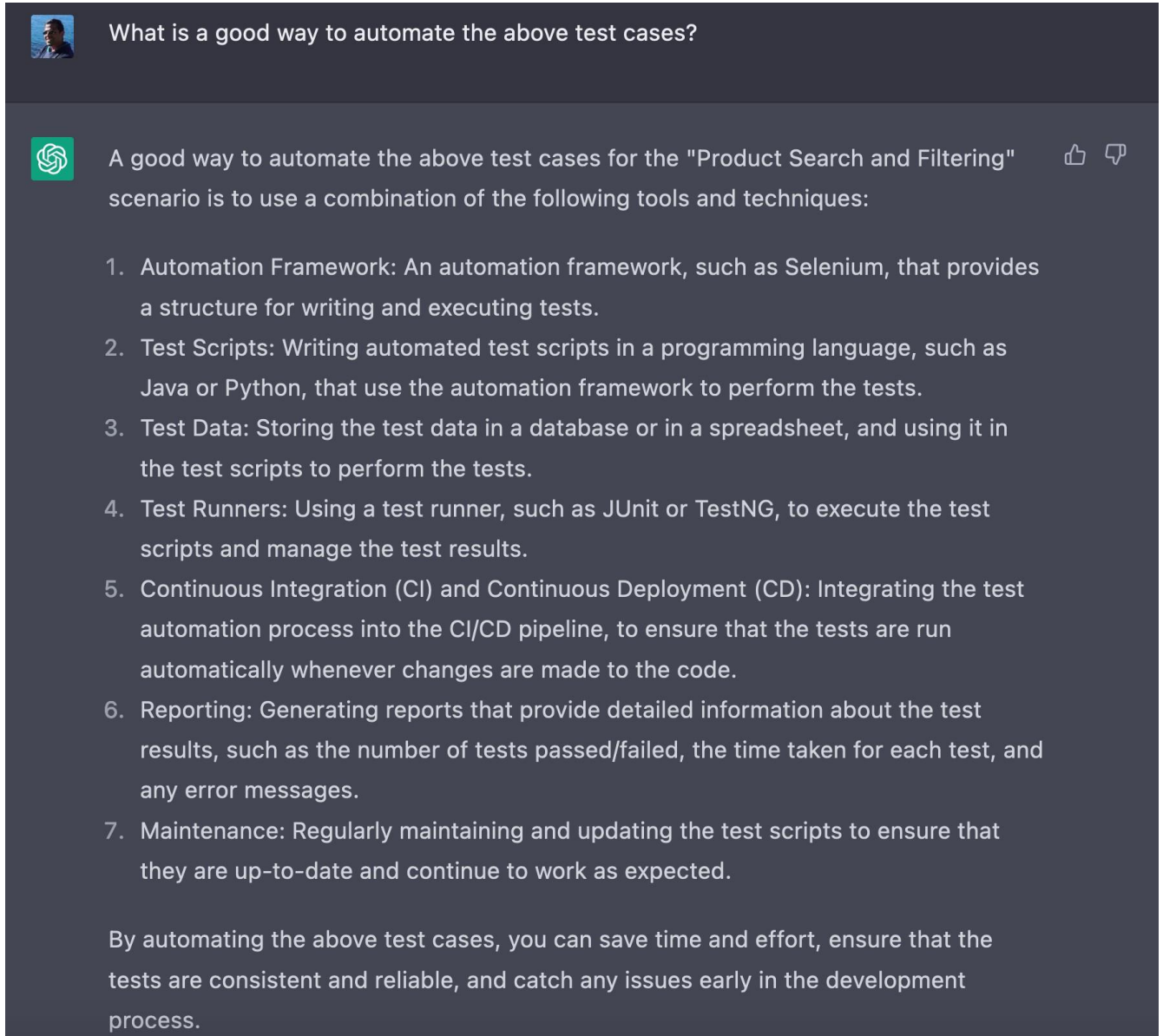
Next question:

What is a good way to automate the above test cases?

Automation – Strategy

Next question:

What is a good way to automate the above test cases?



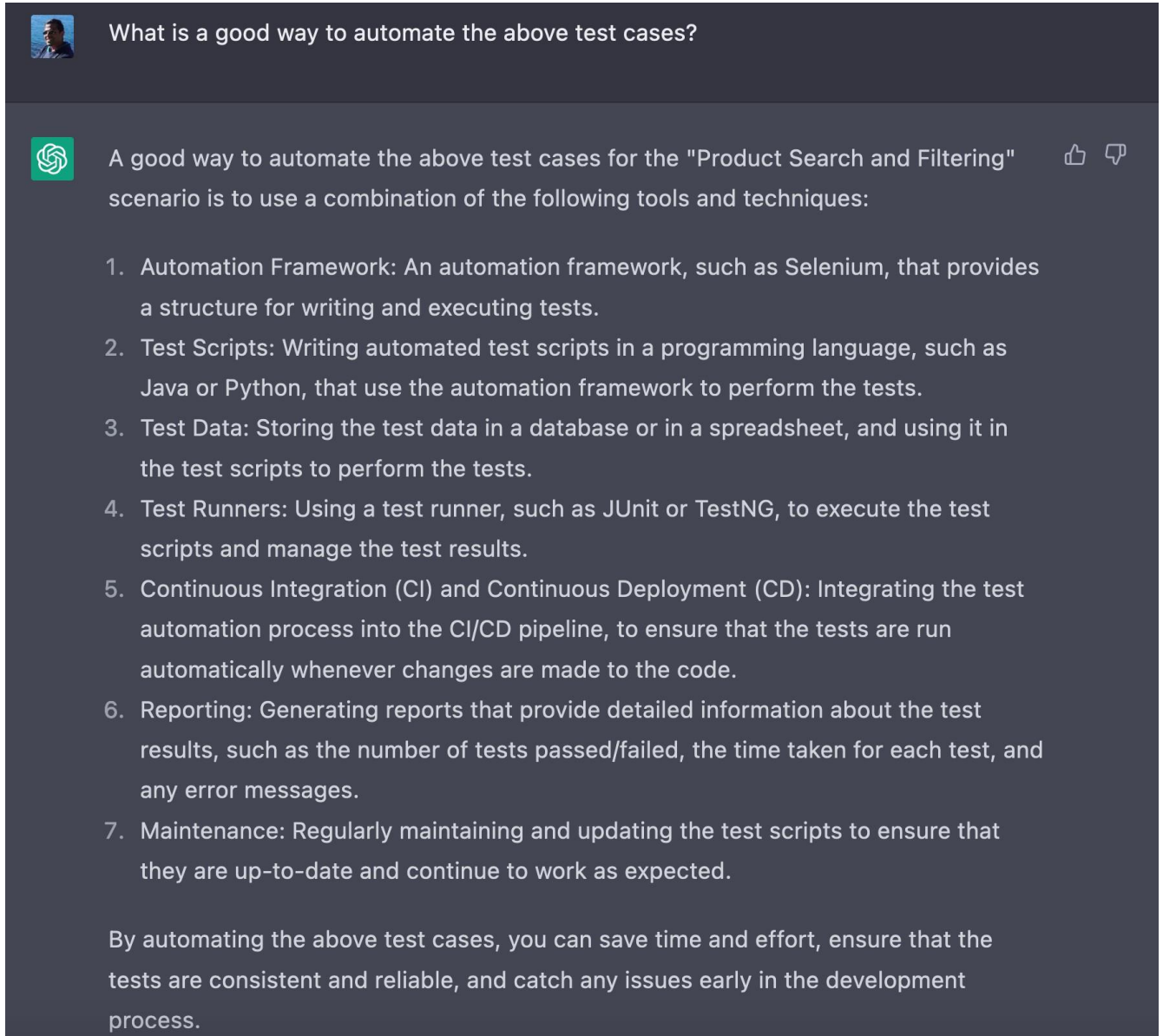
Behavior Driven Development (BDD)

See “001 bddwithcucumber.pptx”

Automation – Strategy

Next question:

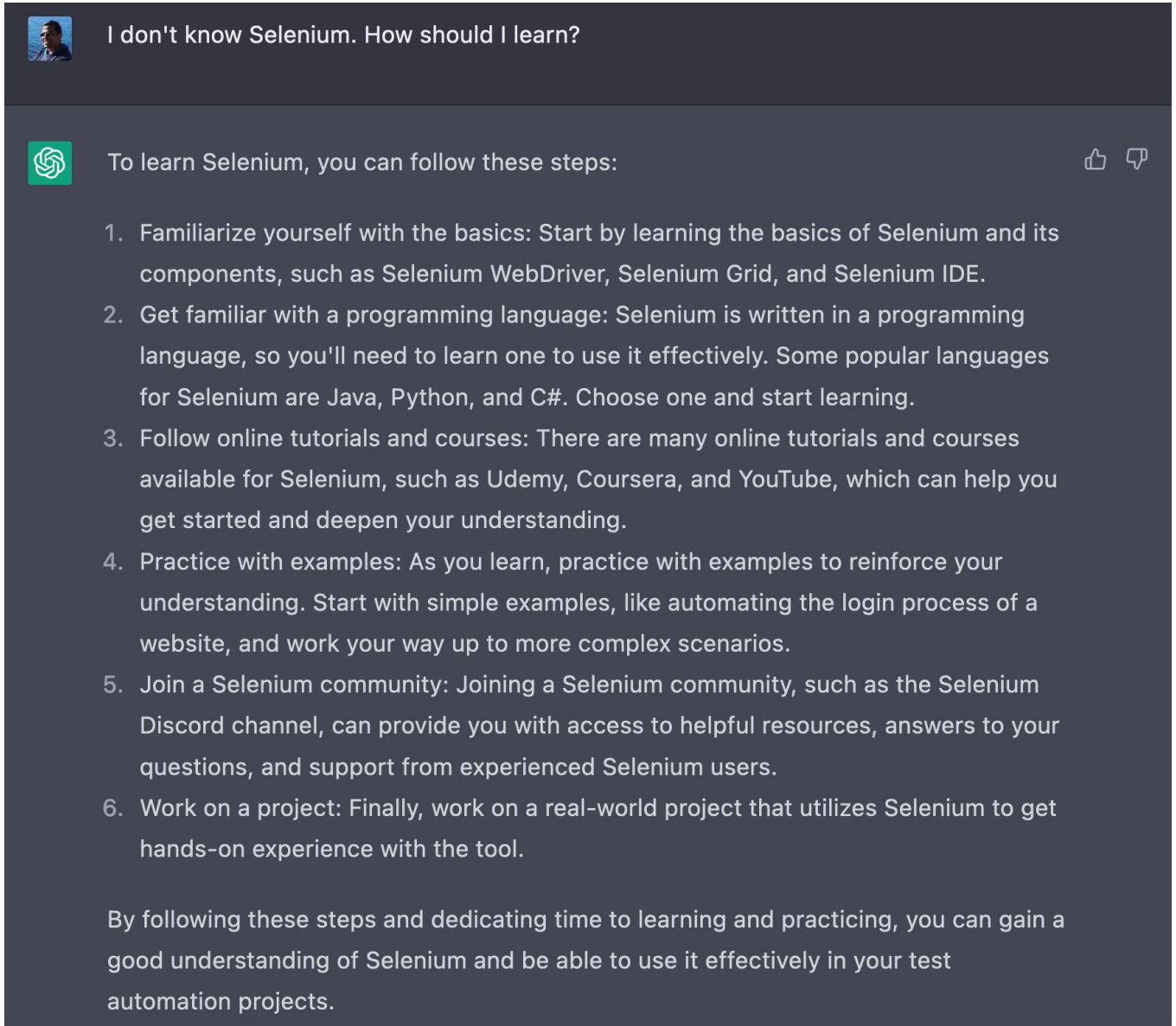
What is a good way to automate the above test cases?



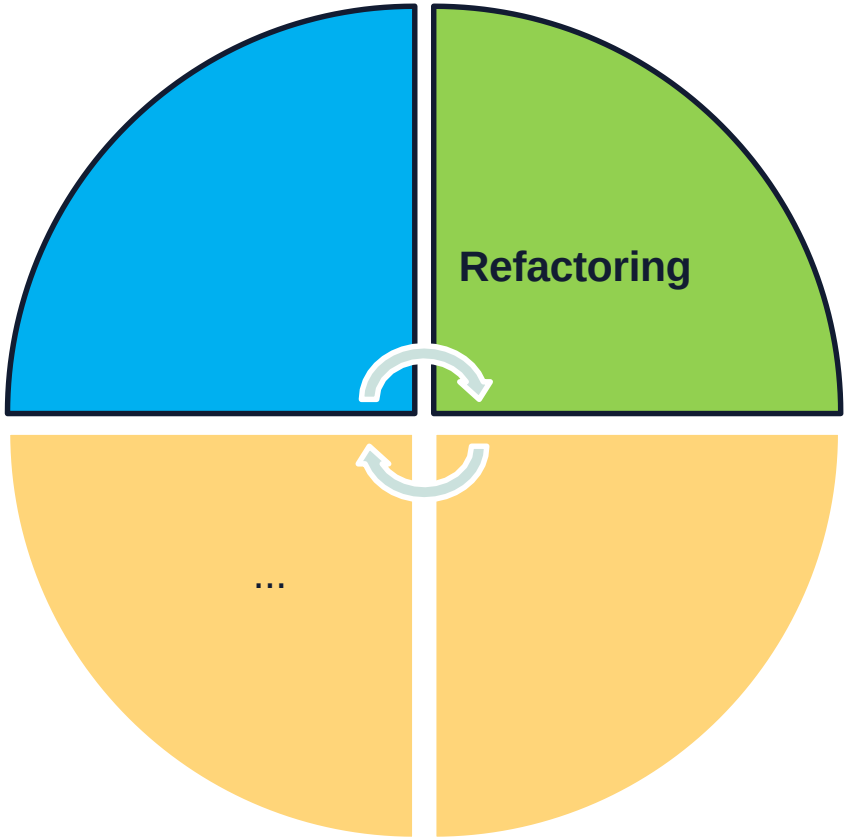
Automation – Getting Started

Next question:

I don't know Selenium. How should I learn?



Automation – Refactoring to Follow Design Patterns



Automation – Refactoring to Follow Design Patterns

Next question:

Implement the

above test using

Page- Object-Model

design pattern

Automation – Refactoring to Follow Design Patterns

Next question:

Implement the above test using Page- Object-Model design pattern

```
class HomePage {
    private WebDriver driver;

    public HomePage(WebDriver driver) {
        this.driver = driver;
    }

    public SearchResultsPage search(String keyword) {
        // Locate the search box and enter the keyword
        driver.findElement(By.id("twotabsearchtextbox")).sendKeys(keyword);

        // Locate the search button and click it
        driver.findElement(By.xpath("//input[@value='Go']")).click();

        return new SearchResultsPage(driver);
    }
}

class SearchResultsPage {
    private WebDriver driver;

    public SearchResultsPage(WebDriver driver) {
        this.driver = driver;
    }

    public String getResultsHeader() {
        return driver.findElement(By.xpath("//span[contains(text(), 'results for')]")).getText();
    }
}
```

Take with caution and have regression tests in place.

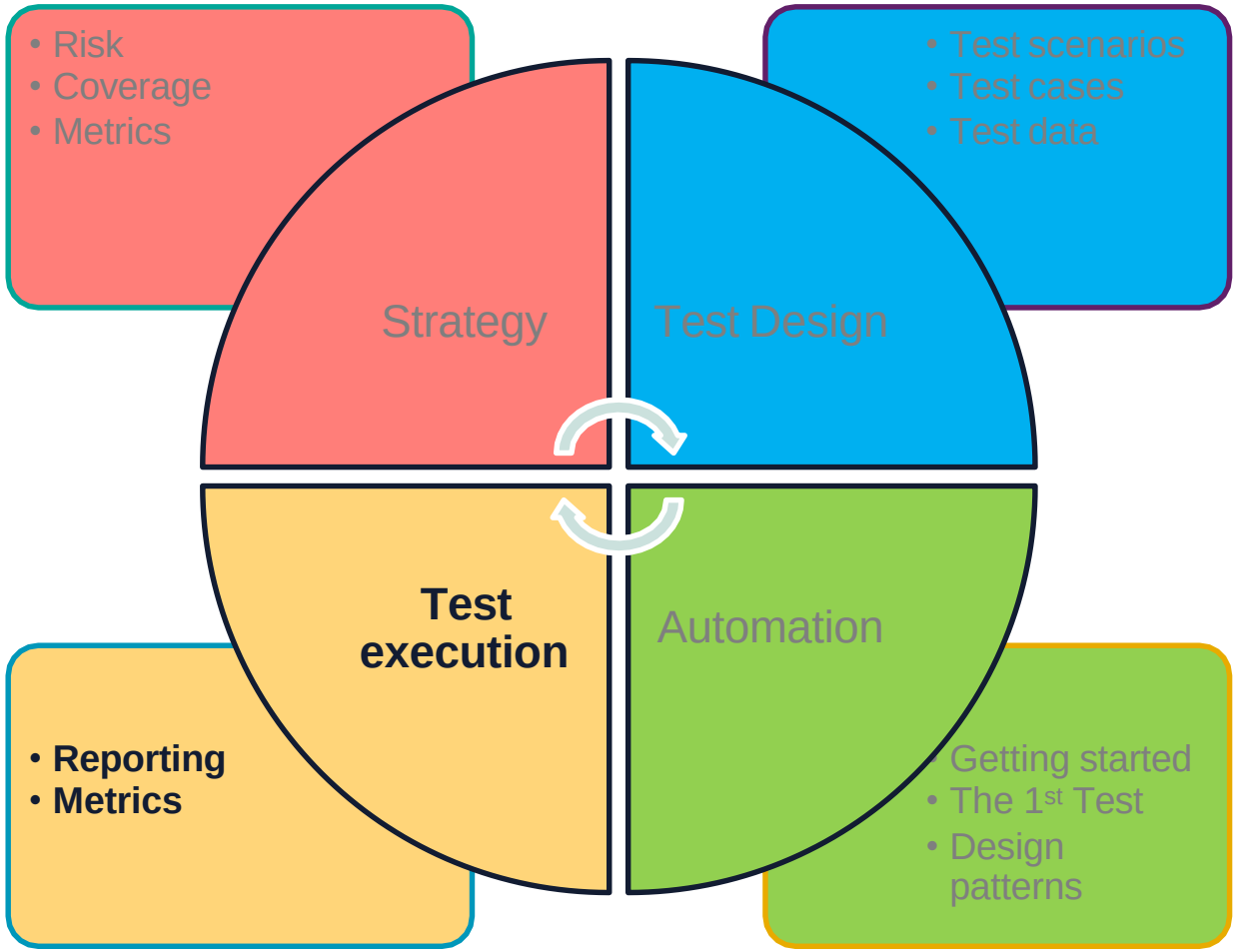
Lab

- Use **lab-s6** which is a **fresh copy of lab-s1.**

- Refactor the test suites to follow Page Object Model.
- Validate if all test are still passing (*regression*)!

- What is your opinion on the quality?

Testing – Test Execution



Test Execution – Reports

Next question:

How to generate

reports and metrics

for automated tests?

Test Execution – Reports

Next question:

How to generate reports and metrics for automated tests?

In order to determine the best reporting solution for your tests, consider the following factors:

- Integration: How easy is it to integrate the reporting solution into your existing testing infrastructure?
- Customization: Can you customize the appearance and content of the reports to meet your needs?
- Interactivity: How interactive are the reports? Can you drill down into the results to see more detail?
- Integration with Metrics: Can you track metrics such as test execution time, test success rate, and test coverage?
- Accessibility: How easily can stakeholders access the reports? Can they be accessed over the web, or do they need to be downloaded and viewed locally?

This is not what I was looking for!

Test Execution – Reports

Follow-up question: *Show an example to generate Allure reports for the automated tests*

Test Documentation

How can we document test cases with an LLM?

Open Questions:

- What do we want do document?
- Documentation structure?
- Level of detail?
- References to code necessary?
- Incorporate the results into the documentation?

→ How do you do test documentation?

https://www.geeksforgeeks.org/software-testing/testing-documentation-software-engineering/

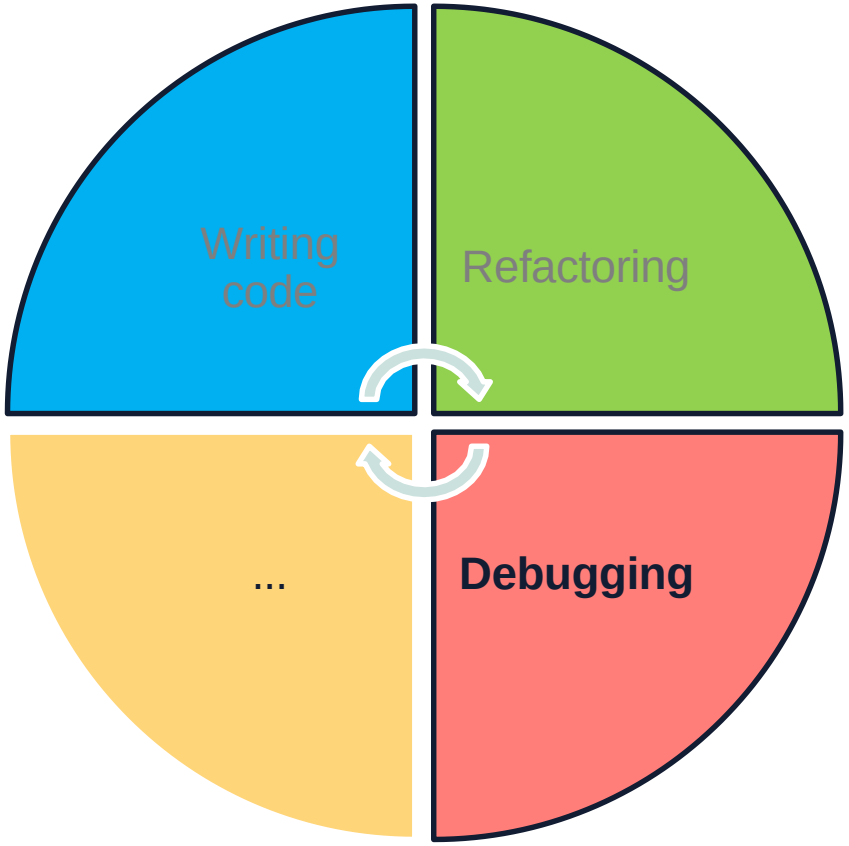
Lab

- Use **lab-s4.**

- Ask Copilot to create a test documentation and include the requirements based on the questions from before.
- Tune and edit until suitable.

- Evaluate the results.

Debugging



Lab

- Find file **ConvertCelsius.feature** from **lab-s02**.

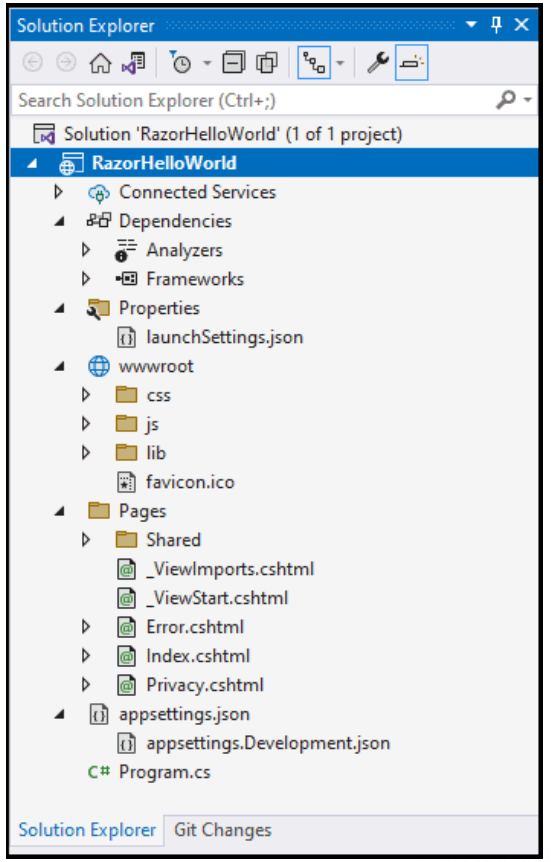
- Include the test that is excluded (commented).
- Rerun and validate that the test is failing.
- Ask Copilot to fix it.
- just ask to fix the failed test nothing more.
- Let Copilot find the problem on its own.

Follow existing Solution Structure

How can we direct the LLM to follow the existing solution structure?

Open Questions:

- Follow solution structure?
- Put resouces into the correct locations?
- Create documentation and put it into the correct location?
- Path-specific instrcutions?



Follow existing Solution Structure

How can we direct the LLM to follow the existing solution structure?

Open Questions:

- Follow solution structure?
- Put resouces into the correct locations?
- Create documentation and put it into the correct location?
- Path-specific instrcutions?

In general it tries to do it automatically.

https://docs.github.com/en/copilot/tutorials/coding-agent/get-the-best-results#repository-wide-instructions