**Application EC Dojo Use Cases**

**Summary**:

This document includes 5 use cases, each use case focusing on different development area:

1. **Use case 1: Unit Testing** - Create unit test cases for all Primary, alternative, positive & Negative flows.
2. **Use case 2: Code Refactoring** - Fix code quality issues reported by SonarQube tool
3. **Use case 3: Code Reviews** – Identify & fix potential coding standards, security issues during code reviews
4. **Use case 4: Performance optimization -** Analyze code for potential performance issues
5. **Use case 5: Design pattern implementation** – Refactor code to apply design patterns
6. **Use case 6: Documentation** – Understand application functionality & generate code documentation with technical specifications

Each use case is defined with specific context, tasks, tools and deliverables associated with each development area, offering a clear roadmap for implementation and highlighting the collaborative effort required to leverage data effectively.

**Objective of use-case**:

To demonstrate the possible scenarios where we can put the snowflake Co-Pilot into picture & address simple tasks. However, Co-Pilot is not limited to the below mentioned use cases, we can put it into use at multiple other scenarios.

**Prerequistes**:

AI Lab machine with

* Visual Studio Code/ Visual Studio
* GitHub Copilot Chat Extension
* GitHub Copilot Account sign in
* GitHub Repository access for sample code

**Setting up Sample Code**:

1. **Download Sample Source Code from GitHub Repository –** [**SampleCode**](https://github.com/divakarvm/copilot/blob/main/SampleCode/GitHubCopilot_DemoCode.zip)
2. **Extract zip file into local folder –**

**A screenshot of a computer

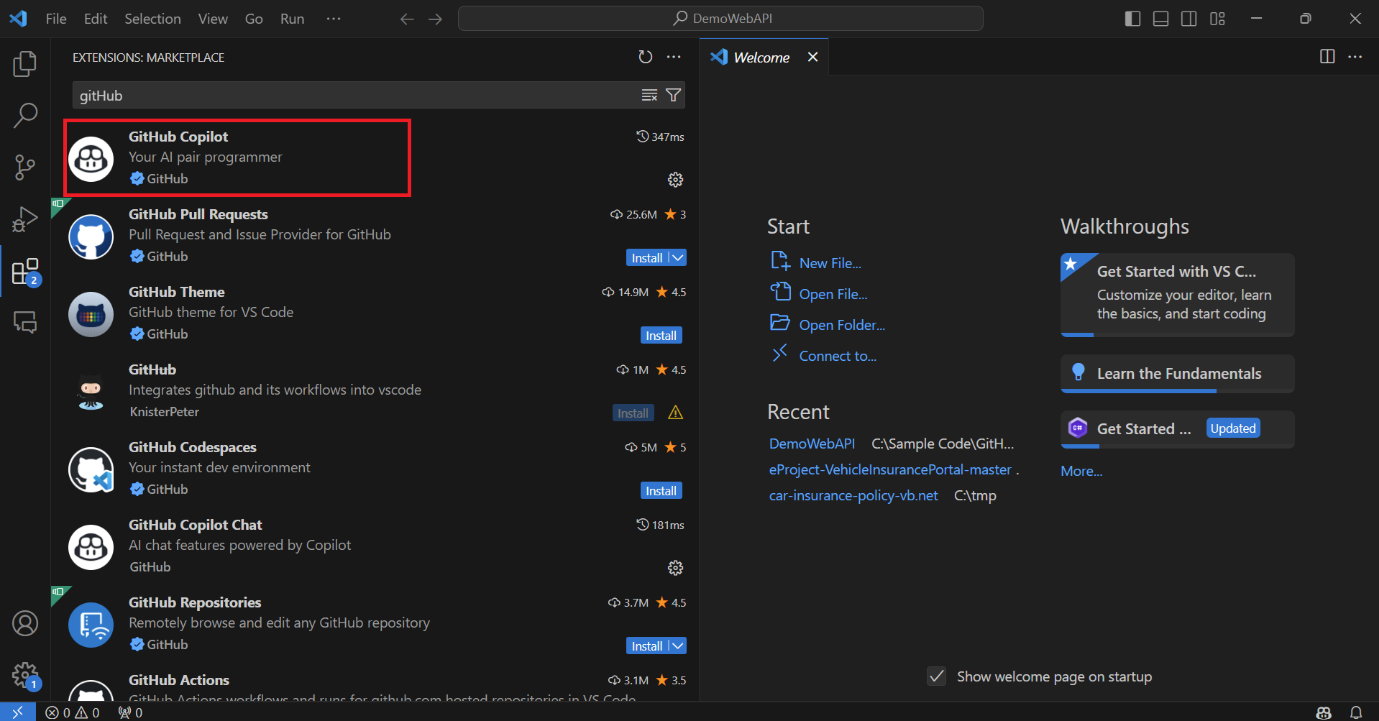
Description automatically generated**

1. **Open Solution using Visual Studio Code –**

**A screenshot of a computer

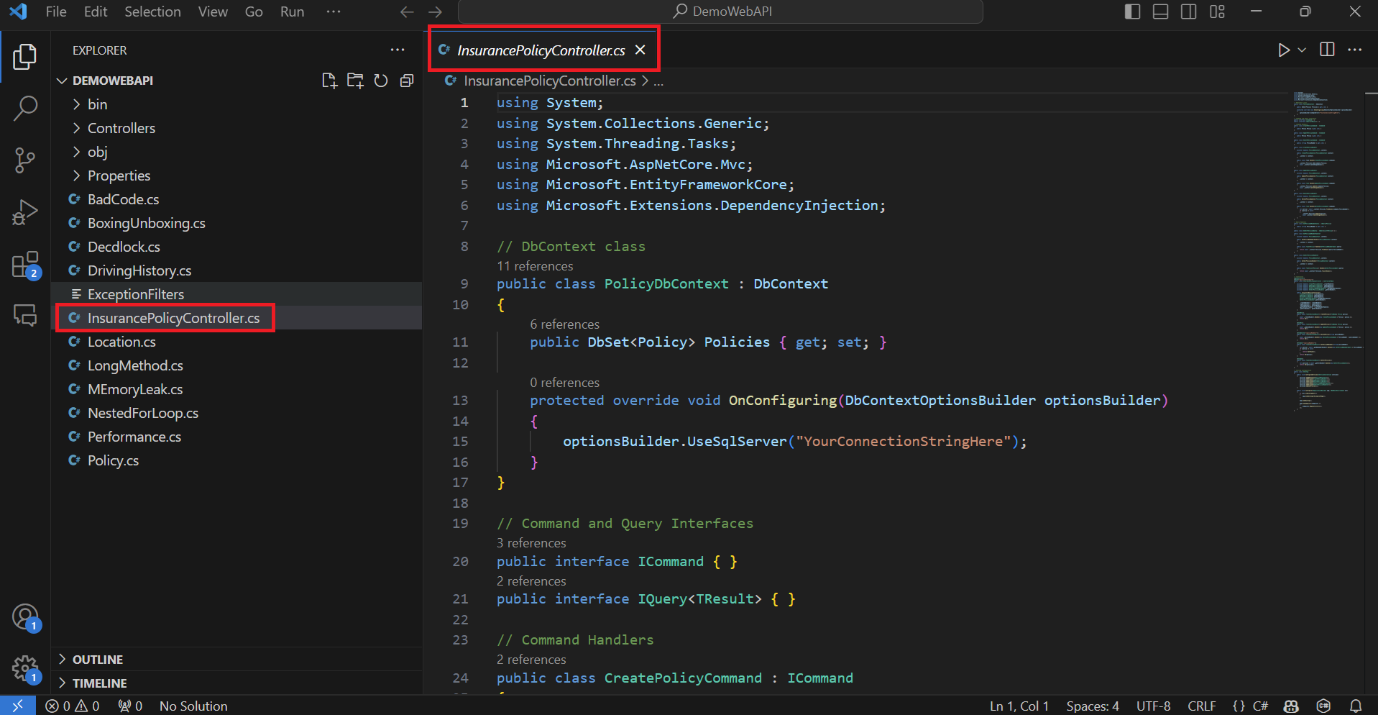
Description automatically generated**

1. **Verify GitHub Copilot Extension installation –**

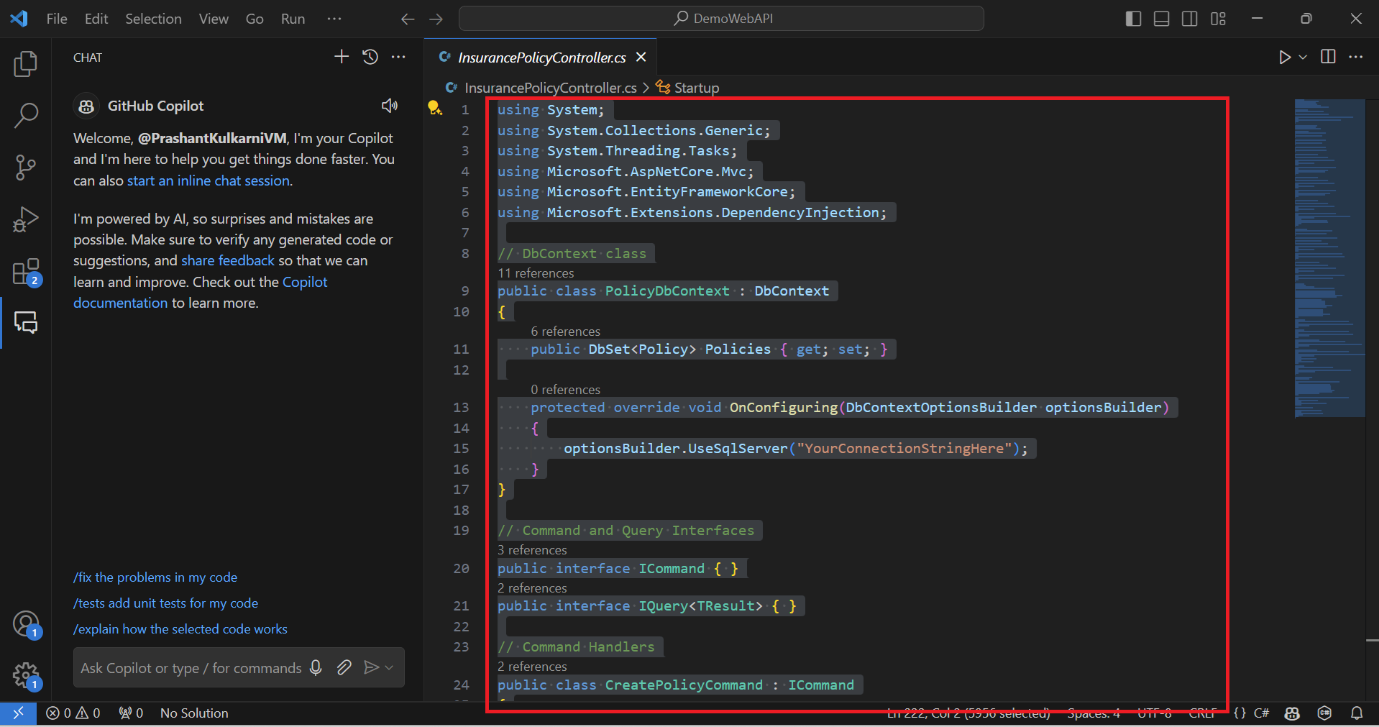
****

**1. Use case 1: Unit Testing**

1. **Use Case Description**: To create unit test cases for all Primary, alternative, positive & Negative flows.
2. **Tasks**:
   1. Open **InsurancePolicyController.cs** file in editor by clicking in the left navigation pane

****

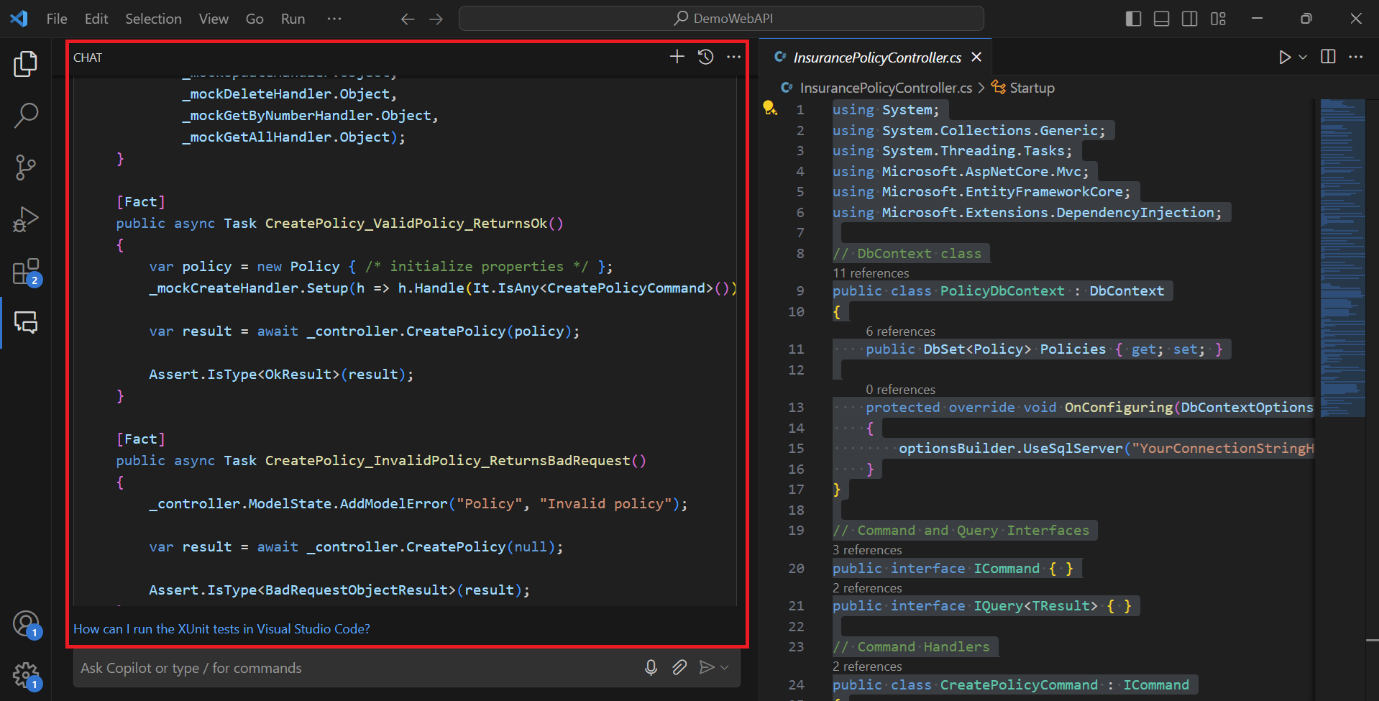
* 1. Select entire code for **InsurancePolicyController.cs** file in editor by pressing **Ctrl+ A** keys

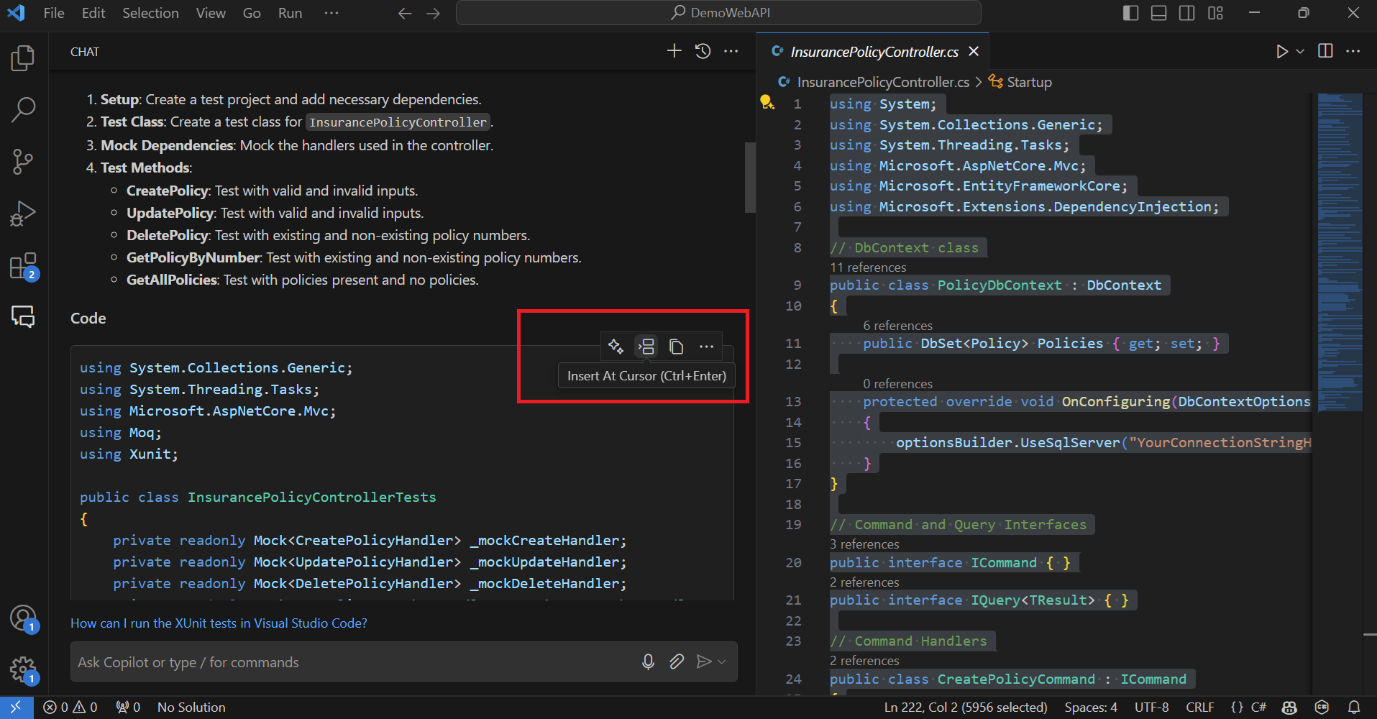


* 1. **Open GitHub Copilot Chat window & enter below prompt –**

#selection Generate unit test cases covering primary , alternative flows & positive, negative scenarios using Xunit

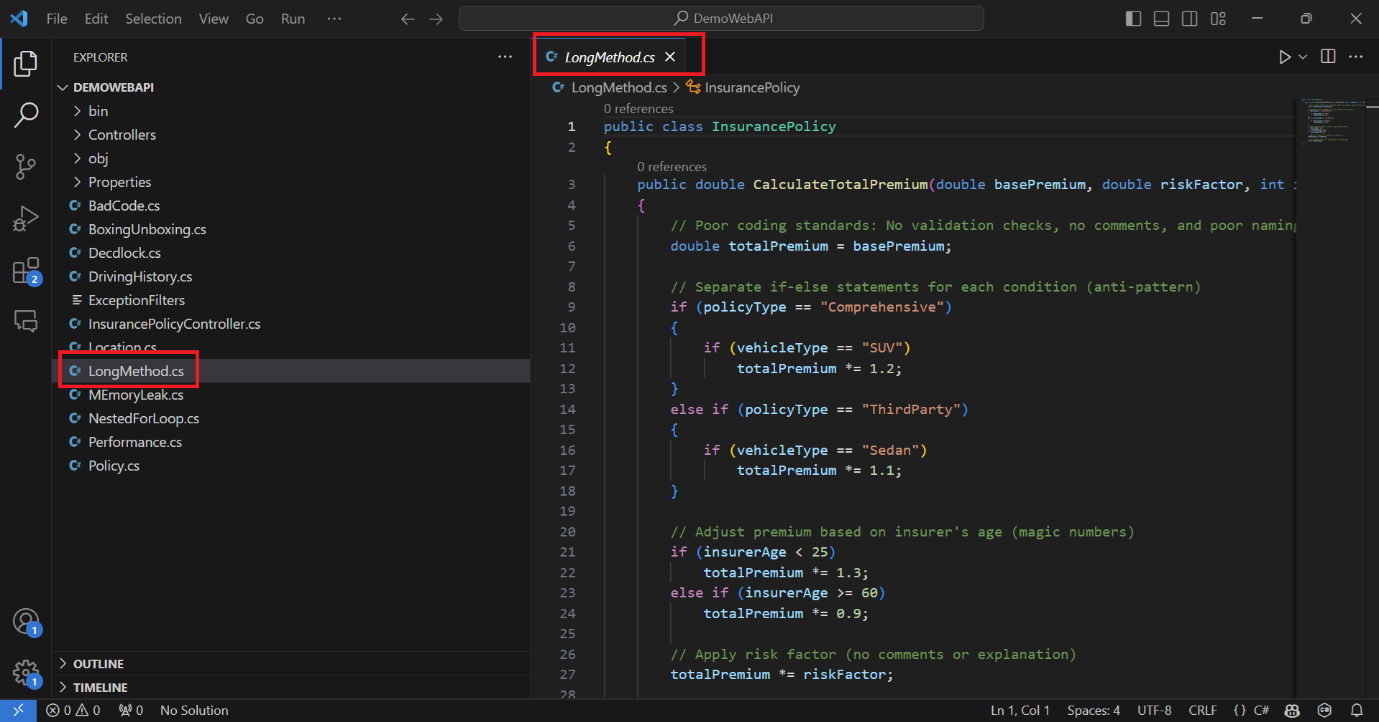
* 1. **Result**: GitHub Copilot will generate unit test cases code which can be copied into source code or inserted to new test case file



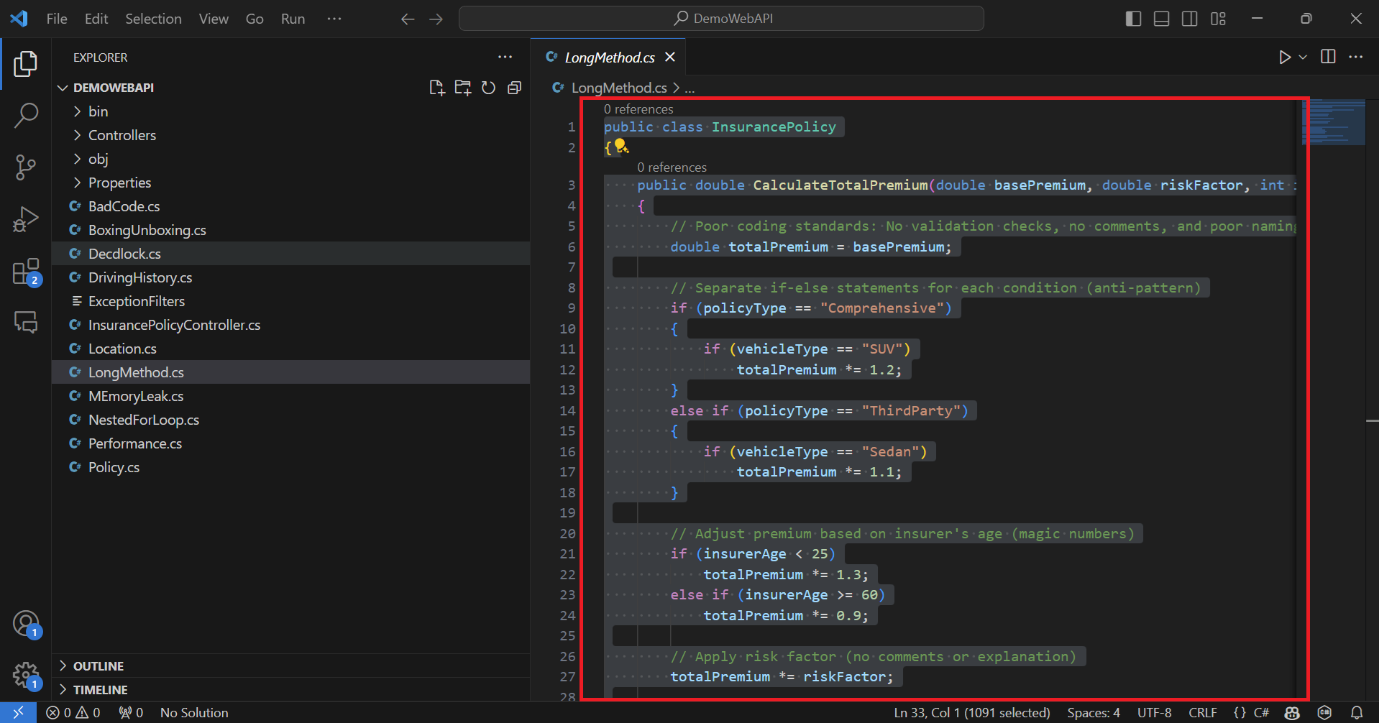


**Use case 2: Code Refactoring**

1. **Use Case Description**: To fix code quality issues reported by SonarQube tool
2. **Tasks**:
   1. Open **LongMethod.cs** file in editor by clicking in the left navigation pane

****

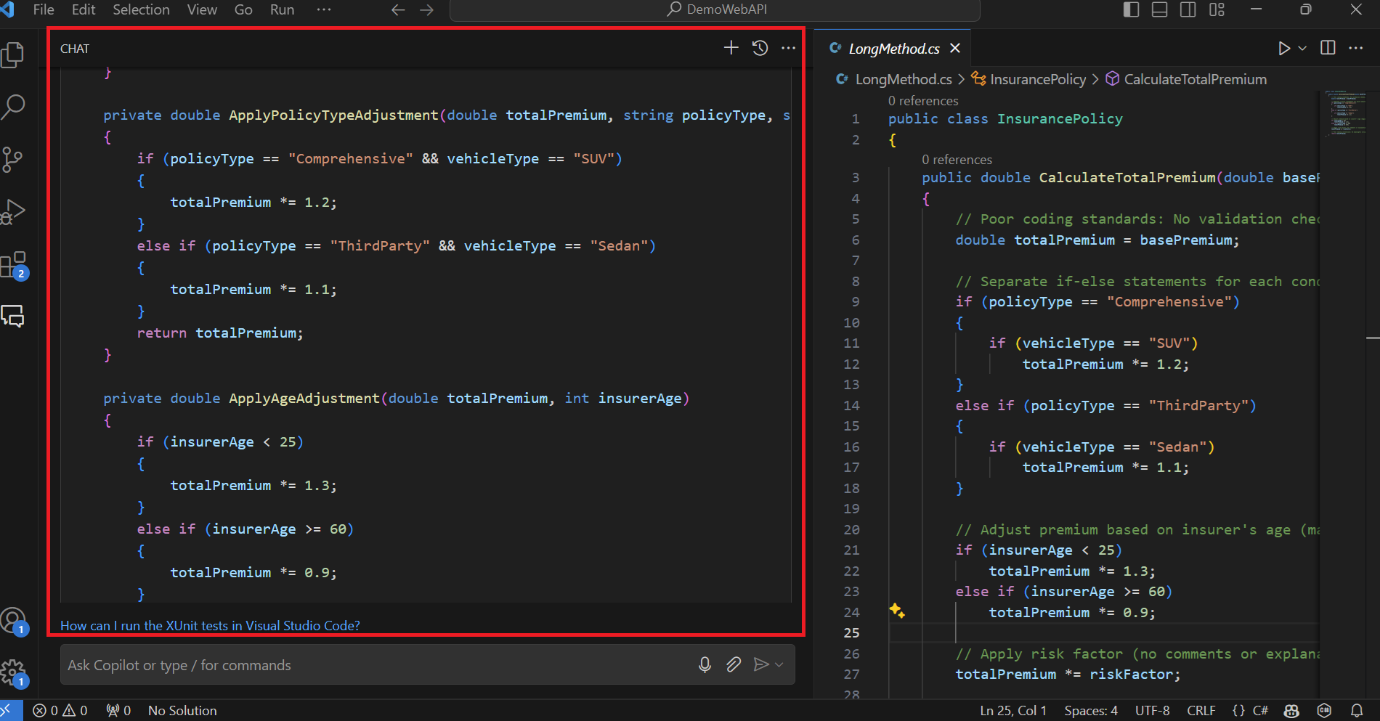
* 1. Select entire code for **LongMethod.cs** file in editor by pressing **Ctrl+ A** keys



* 1. **Open GitHub Copilot Chat window & enter below prompt to resolve cyclomatic complexity issue –**

#selection Refactor CalculateTotalPremium method to resolve cyclomatic complexity issue

* 1. **Result**: GitHub Copilot will generate refactored code by reducing complexity splitting code across small methods reducing number of comparisons

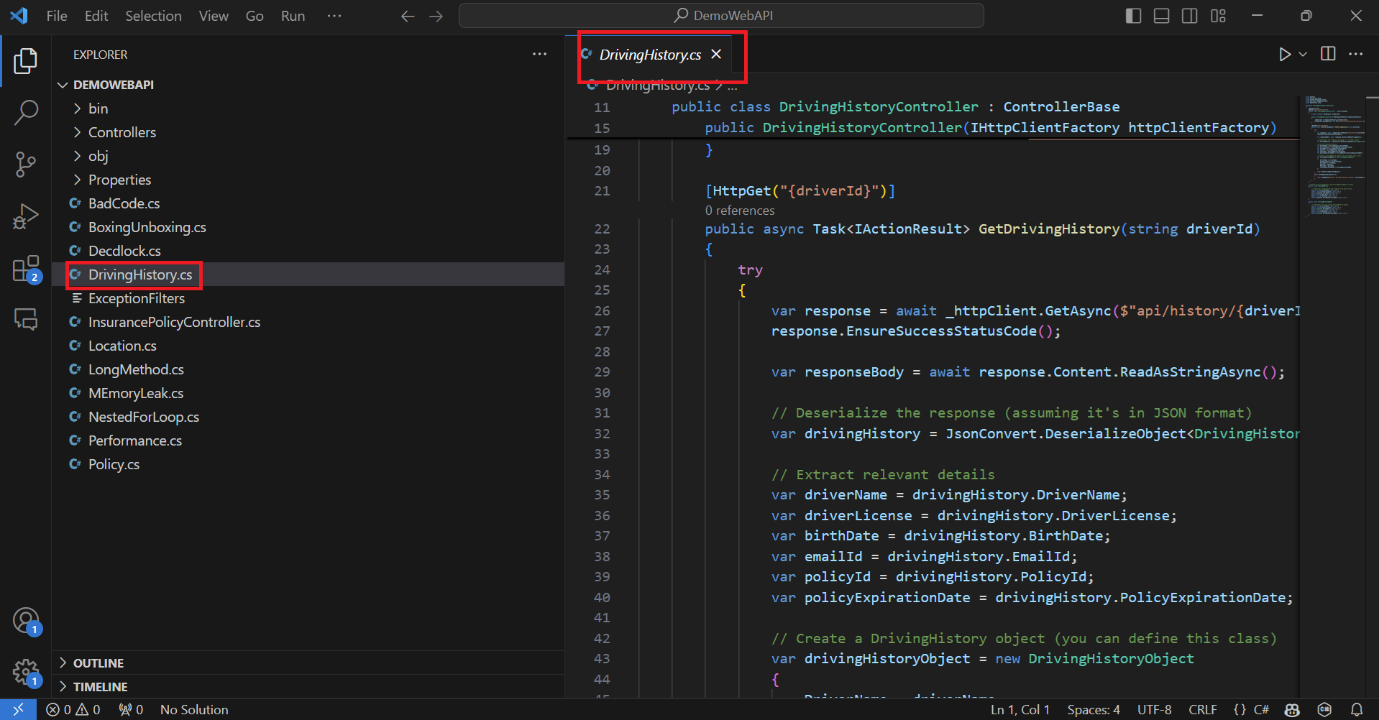


A screenshot of a computer program

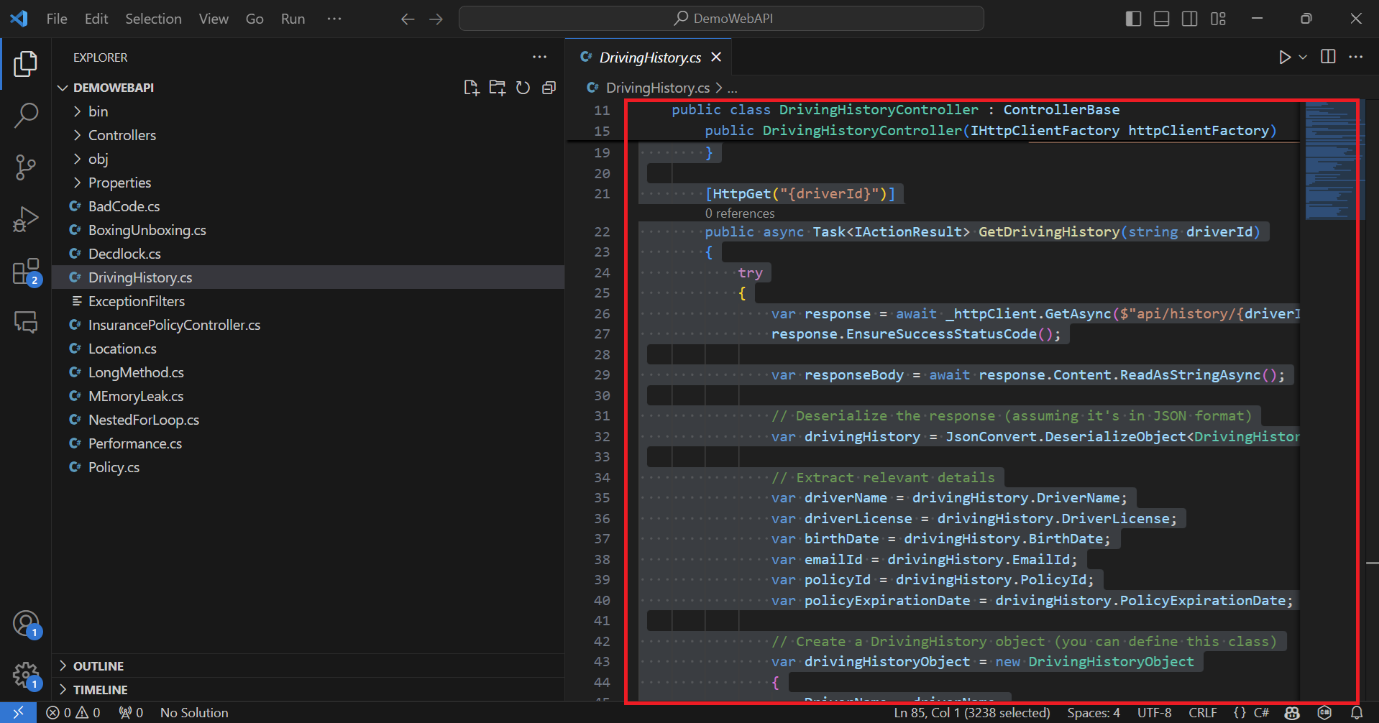
Description automatically generated

**Use case 3: Code Reviews**

1. **Use Case Description**: To identify & fix potential coding standards and security issues during code reviews
2. **Tasks**:
   1. Open **DrivingHistory.cs** file in editor by clicking in the left navigation pane

****

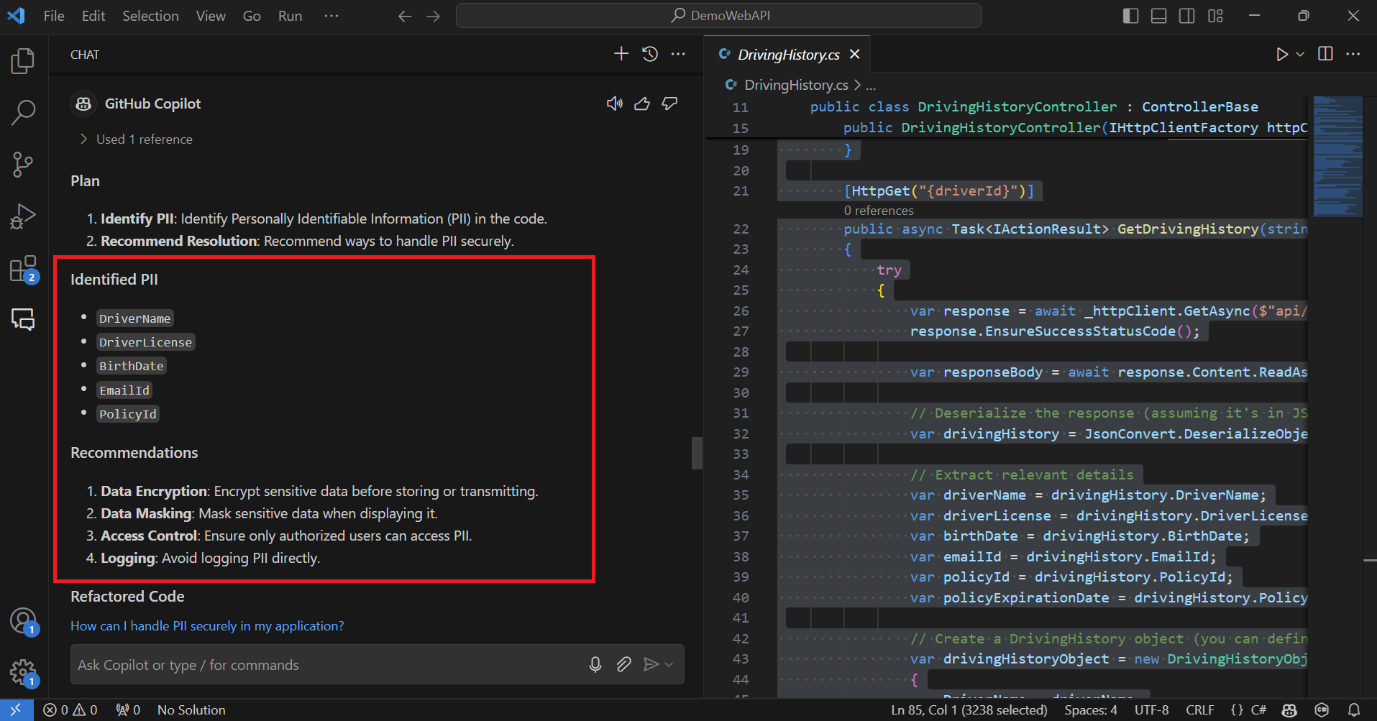
* 1. Select entire code for **DrivingHistory.cs** file in editor by pressing **Ctrl+ A** keys



* 1. **Open GitHub Copilot Chat window & enter below prompt to identify PII issues –**

#selection Identify PII issues & recommend resolution for all such PII issue instances

* 1. **Result**: GitHub Copilot will report with identified PII issues along with refactored code with resolution

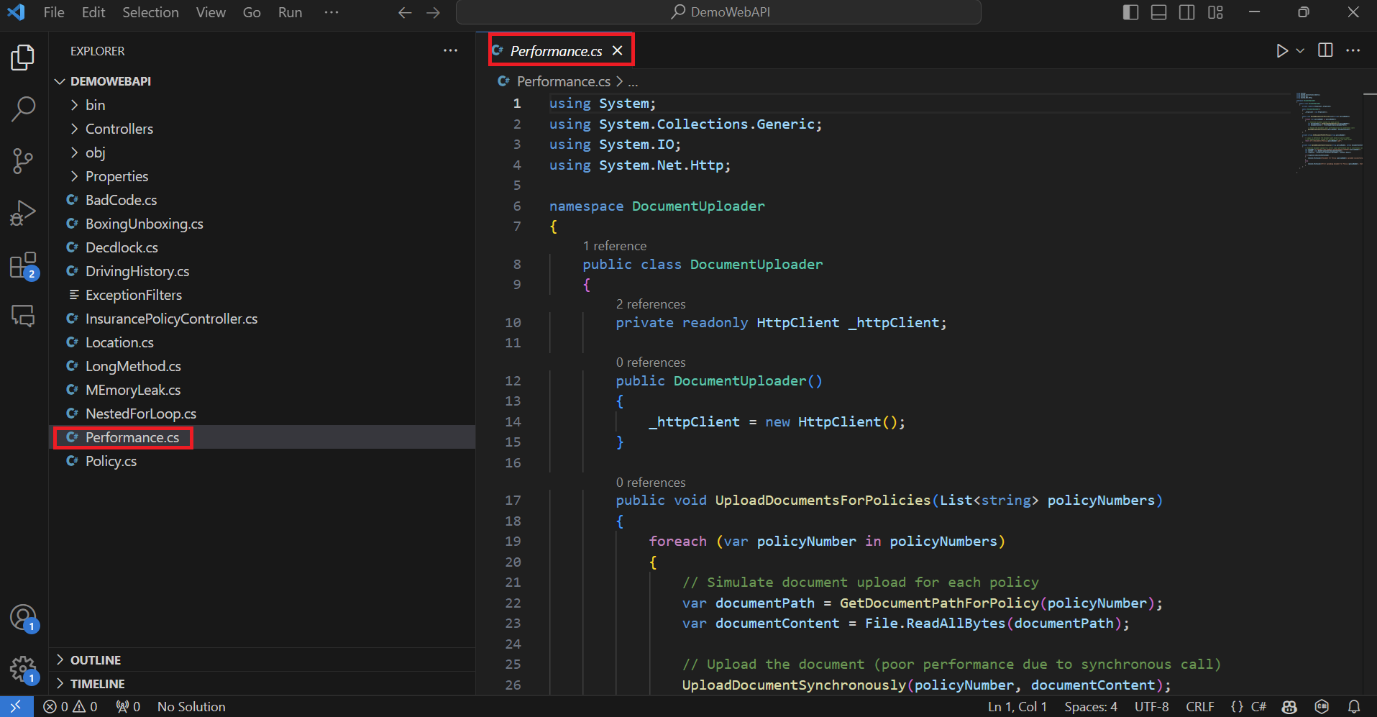


A screenshot of a computer screen

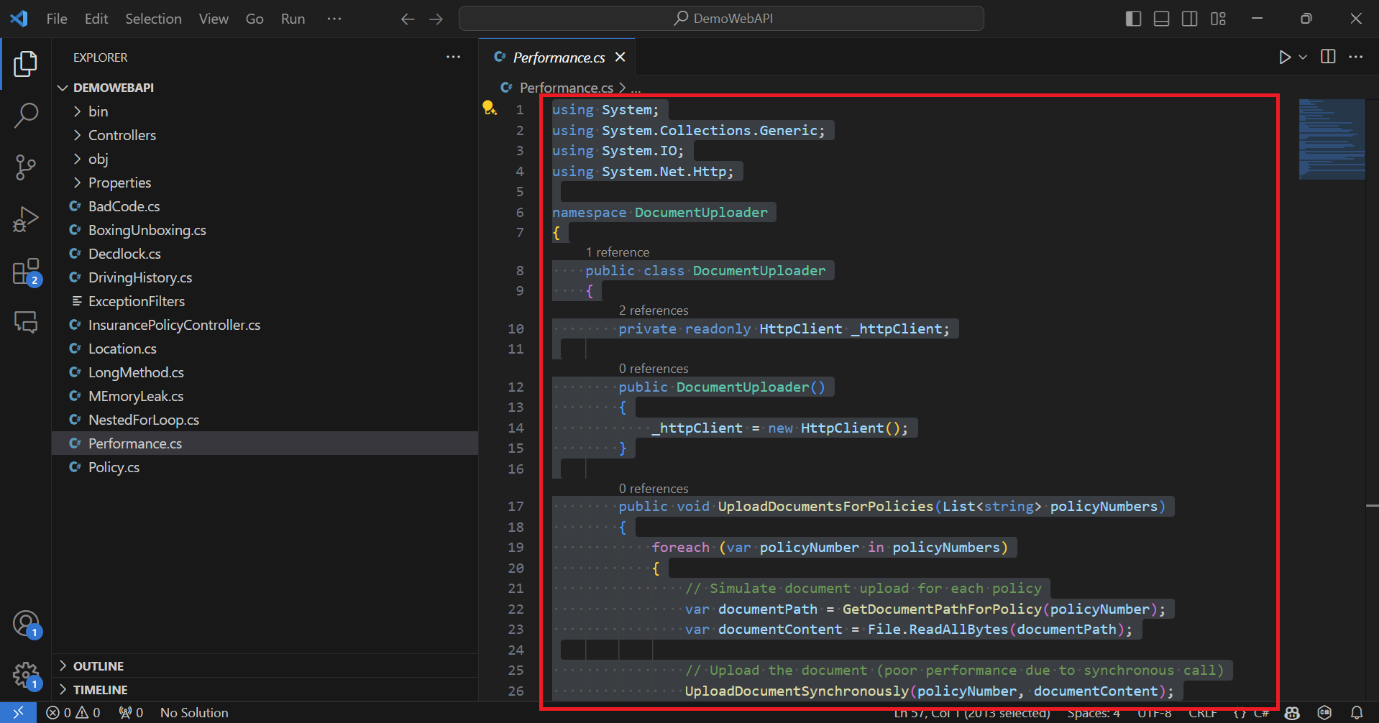
Description automatically generated

**Use case 4: Performance optimization**

1. **Use Case Description**: To analyze code for potential performance issues
2. **Tasks**:
   1. Open **Performance.cs** file in editor by clicking in the left navigation pane

****

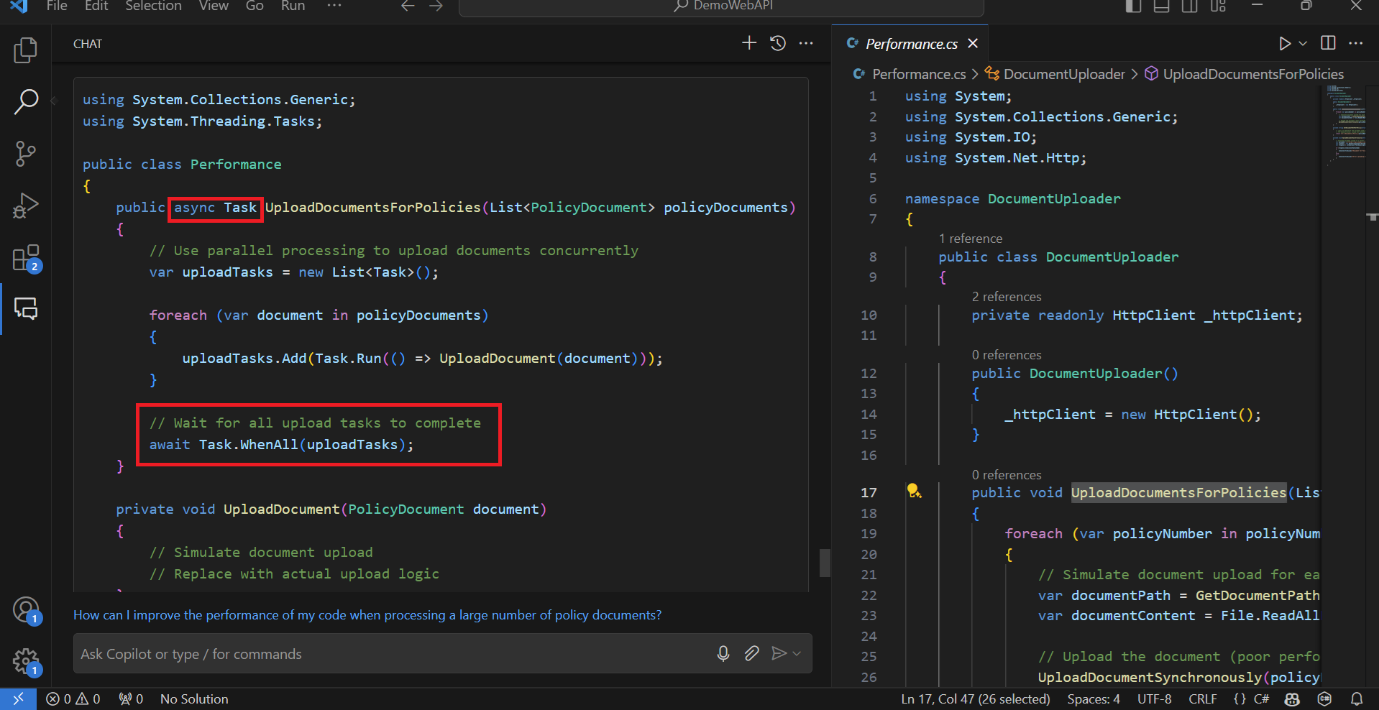
* 1. Select entire code for **Performance.cs** file in editor by pressing **Ctrl+ A** keys

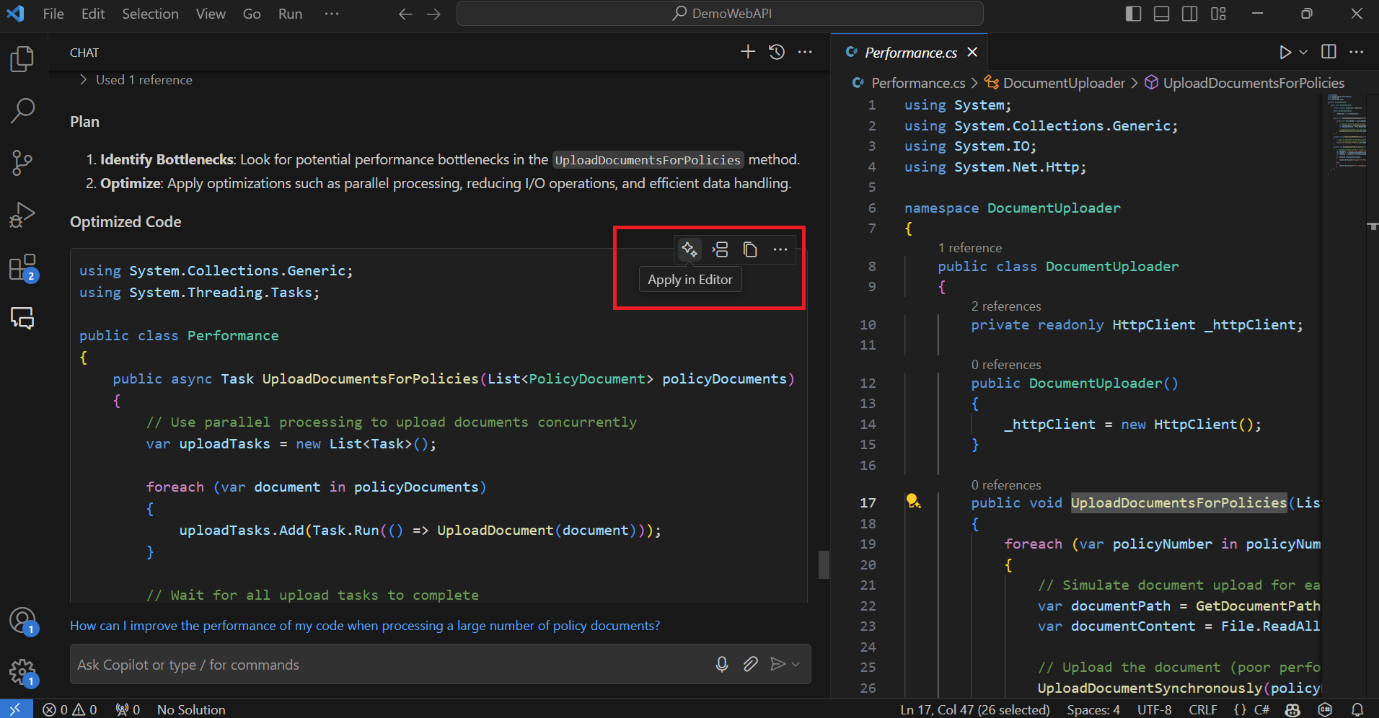


* 1. **Open GitHub Copilot Chat window & enter below prompt to identify performance issue in document upload functionality –**

#selection optimize code for performance issues

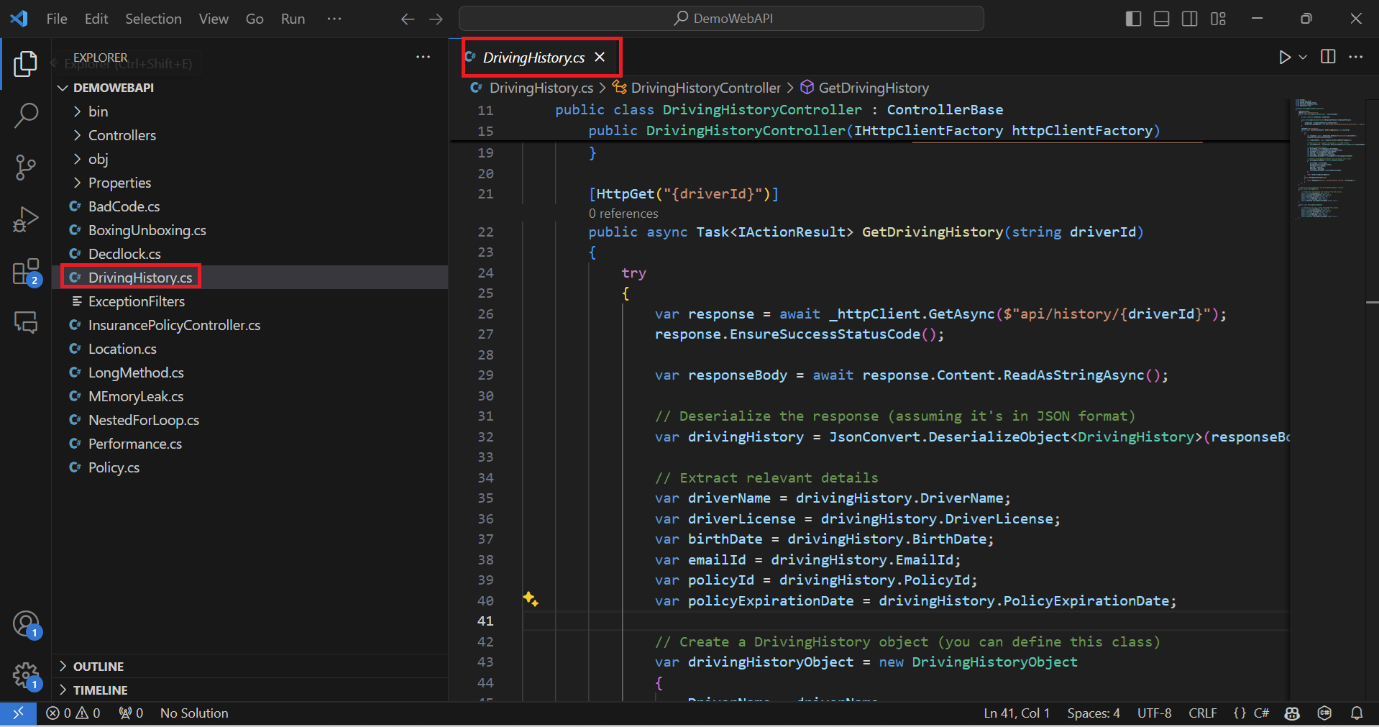
* 1. **Result**: GitHub Copilot will generate refactored code by applying parallism & asyn implementation to optimize code performance.



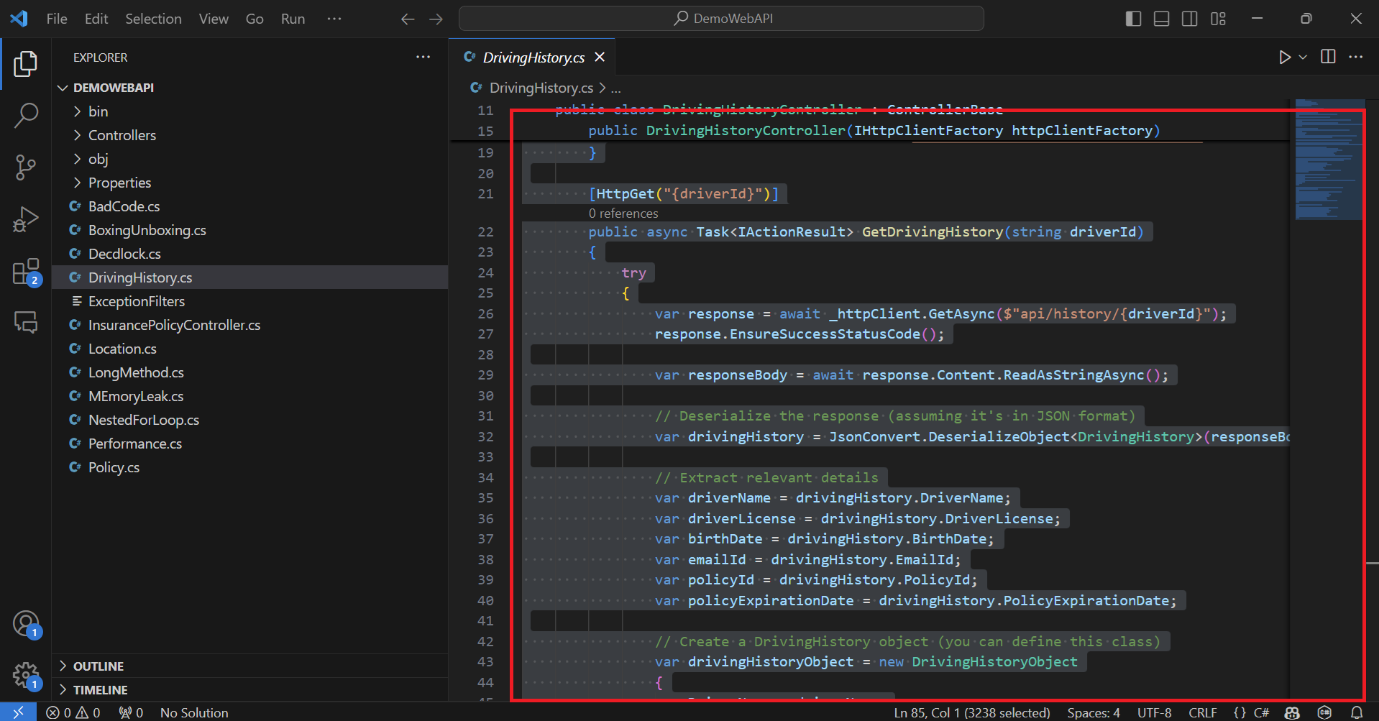


**Use case 5: Design pattern implementation**

1. **Use Case Description**: To Refactor code to apply design patterns
2. **Tasks**:
   1. Open **DrivingHistory.cs** file in editor by clicking in the left navigation pane

****

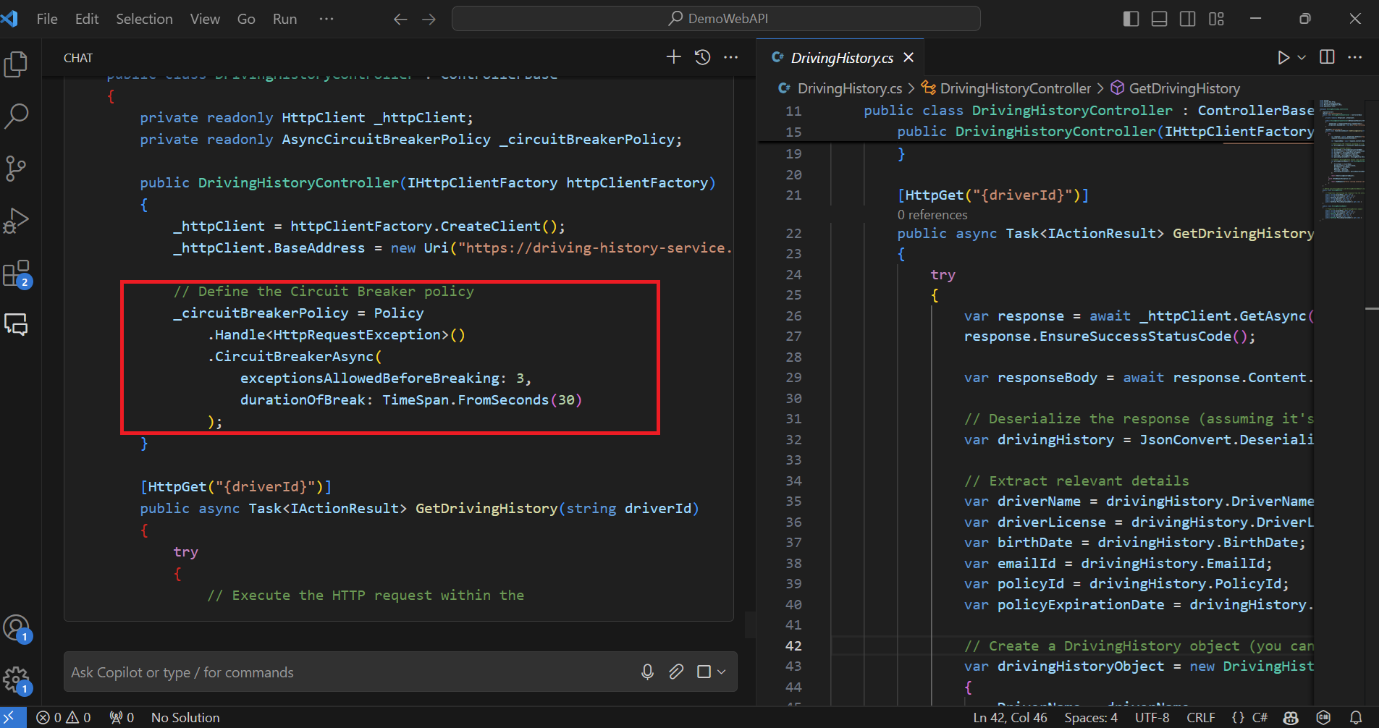
* 1. Select entire code for **DrivingHistory.cs** file in editor by pressing **Ctrl+ A** keys

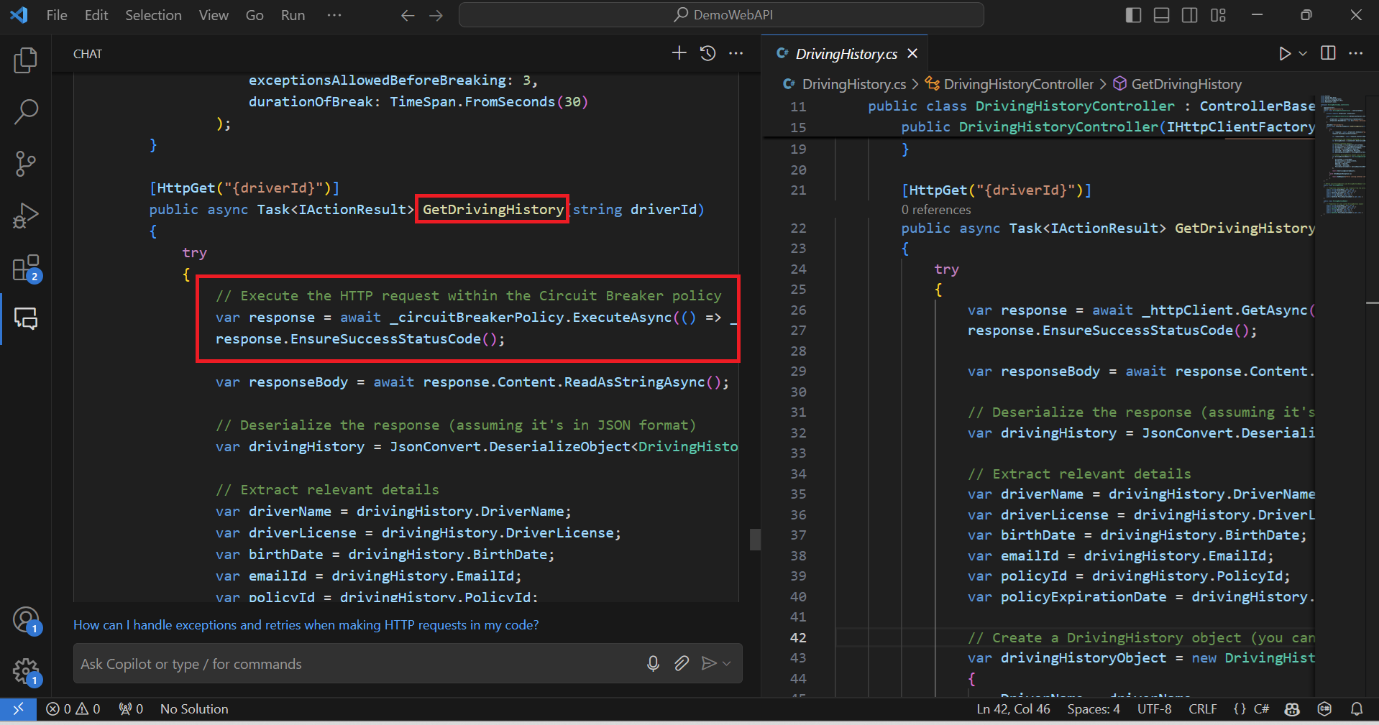


* 1. **Open GitHub Copilot Chat window & enter below prompt to apply CircuitBreaker design pattern to DrivingHistory API calls –**

#selection Reafactor DrivingHistoryController class to implement Circuit Breaker pattern

* 1. **Result**: GitHub Copilot will generate refactored code by applying CircuitBreaker policy invocation during DrivingHistory API calls





A screenshot of a computer

Description automatically generated

**Use case 6: Documentation**

1. **Use Case Description**: To Understand application functionality & generate code documentation with technical specifications
2. **Tasks**:
   1. Open **DrivingHistory.cs** file in editor by clicking in the left navigation pane

**A screenshot of a computer program

Description automatically generated**

* 1. Select entire code for **DrivingHistory.cs** file in editor by pressing **Ctrl+ A** keys

A screenshot of a computer program

Description automatically generated

* 1. **Open GitHub Copilot Chat window & enter below prompt to generate API contract with endpoints –**

#selection Generate API Contract documentation for DrivingHistoryController class

* 1. **Result**: GitHub Copilot will generate API contract documentation for DrivingHistoryConttoller class with endpoint details

