# Week 1 – GitHub Copilot Overview Lab use cases

**Note** – Complete all this exercise & save your code in AI Lab machine folder. Share that path to Training coordinator for assessment. It is mandatory to complete Week1 workshop before beginning next week sessions.

1. Revise GitHub Copilot Extension installation & configuration steps in Visual Studio

Reference - [Configuring GitHub Copilot in your environment - GitHub Docs](https://docs.github.com/en/copilot/managing-copilot/configure-personal-settings/configuring-github-copilot-in-your-environment)

1. Revise GitHub Copilot context setting options with different prompts at code block, method, class, editor window & project level to understand code functionality
2. Modify existing code to apply logging & exception handling. Verify changes using Preview option before merging changes to original code
3. Revise prompts to generate functions for “Document upload & Solid principals adherence “ through GitHub Copilot various chat options - Inline Chat, Chat view, Quick Chat
4. Build & execute GitHub Copilot prompts for below actions
5. Generate new Policy class with specifications –

* Email should be valid expression
* Account Number is 8 digit number & cannot be empty
* Policy Number is 8 digit number
* First Name is 50 character non-empty string
* Last Name is 50 character non-empty string
* Postal Code is 7 digit number
* Phone is valid phone number format
* Producer Code is 5 character not empty string
* Group Code is 5 character not empty string
* Master Code is 5 character not empty string
* City is 20 character non-empty string
* State is 20 character non-empty string
* Effective Date is valid date format less than or equal to current date
* Expiration Date is valid date format greater than or equal to current date
* Annual Premium decimal type with 2 decimal places non zero value

1. Refactor Policy class to move policy validation code for all fields into new PolicyValidator class
2. Create PolicyController class to implement CRUD operations using Entity Framework for above created Policy class
3. Refactor PolicyController to implement Token-based authorization, exception handling & logging
4. Refactor Policy & PolicyController classes to extract interfaces
5. Analyse PolicyController class to identify potential issues in CRUD operations implementation
6. Fix PolicyController class to fix potential issues in CRUD operations implementation
7. Ask a question what this code in the editor does functionally?
8. Ask a question to explain what this code in the editor technically?
9. Generate documentation for Policy & PolicyController classes