Practical Assignment - Copilot

Task: Explore GitHub Copilot in your preferred IDE for creating a NUMBER GUESSING GAME application:

Create a console-based Number Guessing Game using Python (or your preferred programming language). The game will randomly select a number within a range, and the player has to guess it. Feedback should be provided for each guess (higher/lower), and the game should track the number of attempts. Throughout this assignment, you will leverage various features of GitHub Copilot to assist in your development

Requirements

1. **Random Number Generation**: The game should generate a random number between 1 and 100.
2. **User Input**: Prompt the user to guess the number.
3. **Feedback**: Provide feedback for each guess, indicating if the guess is too high, too low, or correct.
4. **Attempt Tracking**: Keep track of the number of attempts the player makes.
5. **Play Again Option**: After guessing the correct number, ask the player if they want to play again.
6. **Input Validation**: Ensure that the user enters valid numeric input.
7. **Enhanced Features** (Optional but encouraged):
    - Allow the player to set the range for the random number.
    - Implement a scoring system.
    - Provide hints after a certain number of attempts.
    - Store high scores in a file and display them.

The application should be created using the following features of GitHub Copilot

List the prompts and commands used to create the application and showcase the following features individually

1. **Code Completion and Context:**
    - Open relevant files in your project.
    - Write a code snippet where you need assistance (e.g., a function, class, or method).
    - Observe how GitHub Copilot provides code suggestions based on the context.
    - Take a screenshot of the code suggestion and include it in your assignment document.
2. **Chat Interface:**
    - Use the chat interface within GitHub Copilot to ask for help with a specific coding problem.
    - Describe the problem briefly and capture a screenshot of the chat interaction.
    - Explain how the chat interface assisted you in solving the problem.
3. **Ghost Text and Code Generation:**
    - Write a comment or placeholder in your code (ghost text).
    - Observe how GitHub Copilot generates code based on the ghost text.
    - Provide an example of ghost text usage and include the generated code.
    - Explain how this feature can improve your coding efficiency.
4. **Inline Chat and Collaboration:**
    - Collaborate with a colleague or friend using GitHub Copilot's inline chat.

    - Share code snippets, discuss solutions, and make real-time edits.
    - Describe your experience with inline chat and its benefits.
5. **Best Practices and Dos and Don’ts:**
    - Research best practices for using GitHub Copilot.
    - List at least three dos and three don’ts when working with the tool.
    - Explain why following these practices is essential.
6. **Security and Privacy Considerations: Unauthorized Code Access**
    - GitHub Copilot may inadvertently generate code that exposes sensitive information, such as authentication tokens or API keys.
    - Discuss the risks associated with exposing such information in code comments.
    - Explain the potential impact if an unauthorized person gains access to these tokens.
    - Propose preventive measures to avoid accidental exposure.

Screenshots and Documentation:

1. Compile all your findings, screenshots, and explanations into a document
2. Include step-by-step instructions for each task and segregate the development under each feature use (use as many features as you can)
3. Submit the document as part of your assignment along with the application in a ZIP file
4. The application should be running without errors
5. Remember to be thorough and provide clear explanations for each task