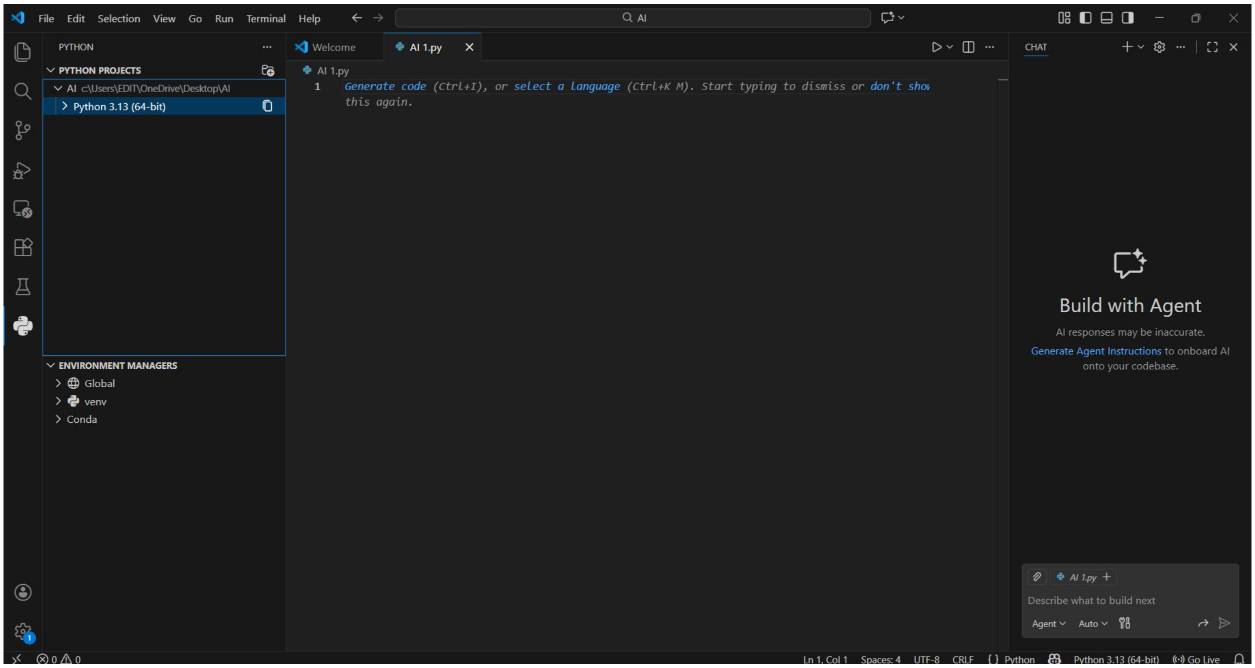
ASSIGNMENT 1.5

2303A52T02

BT.NO-50

TASK 0

Install and configure GitHub Copilot in VS Code. Take screenshots of each step.



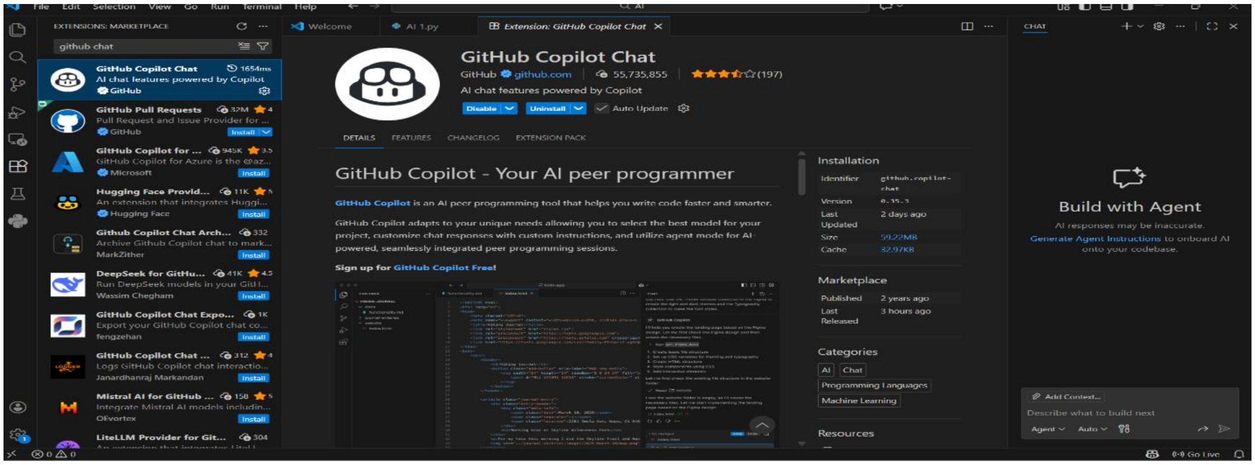
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to dismiss or don't show this again.

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next



github chat

GitHub Copilot Chat 1654ms
AI chat features powered by Copilot
GitHub

GitHub Pull Requests 32M 4
Pull Request and Issue Provider for ...
GitHub

GitHub Copilot for ... 945K 3.5
GitHub Copilot for Azure is the @az...
Microsoft

Hugging Face Provid... 11K 5
An extension that integrates Huggi...
Hugging Face

Github Copilot Chat Arch... 332
Archive Github Copilot chat to mark...
MarkZither

DeepSeek for GitHu... 41K 4.5
Run DeepSeek models in your GitH...
Wassim Chegham

GitHub Copilot Chat Expo... 1K
Export your GitHub Copilot chat co...
fengzehan

GitHub Copilot Chat ... 312 4
Logs GitHub Copilot chat interactio...
Janardhanraj Markandan

Mistral AI for GitHub ... 158 5
Integrate Mistral AI models includin...
OEvortex

LiteLLM Provider for Git... 304

GitHub Copilot Chat
GitHub github.com | 55,735,855 | (197)
AI chat features powered by Copilot

GitHub Copilot - Your AI peer programmer

GitHub Copilot is an AI peer programming tool that helps you write code faster and smarter.

GitHub Copilot adapts to your unique needs allowing you to select the best model for your project, customize chat responses with custom instructions, and utilize agent mode for AI-powered, seamlessly integrated peer programming sessions.

Sign up for GitHub Copilot Free!

Installation

Identifier	github.copilot-chat
Version	0.35.1
Last Updated	2 days ago
Size	59.22MB
Cache	32.97KB

Marketplace

Published	2 years ago
Last Released	3 hours ago

Categories

AI
Chat
Programming Languages
Machine Learning

Resources

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

TASK1:

Use GitHub Copilot to generate a Python program that:

- ➢ Reverses a given string
- ➢ Accepts user input
- ➢ Implements the logic directly in the main code
- ➢ Does not use any user-defined functions

PROMPT:

Write a python code forReverse a given string that accepts a user input and implements the logic directly in the main code and it doesnot use any user defined functions.

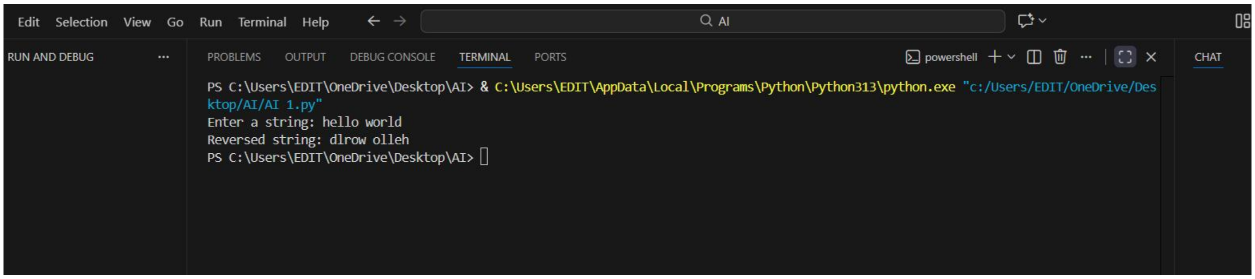
CODE

```
user_input = input("Enter a string: ")

reversed_string = user_input[::-1]

print("Reversed string:", reversed_string)
```

OUTPUT



```
PS C:\Users\EDIT\OneDrive\Desktop\AI> & C:\Users\EDIT\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/EDIT/OneDrive/Des
ktop/AI/AI 1.py"
Enter a string: hello world
Reversed string: dlrow olleh
PS C:\Users\EDIT\OneDrive\Desktop\AI> 
```

EXPLANATION

This code accepts user input i.e,a string.Then for reversing a string we use ::-1,Then print reversed string.

TASK 2

Examine the Copilot-generated code from Task 1 and improve it by:

➢ Removing unnecessary variables

➢ Simplifying loop or indexing logic

➢ Improving readability

➢ Use Copilot prompts like:

▪ “Simplify this string reversal code”
▪ “Improve readability and efficiency”
Hint:
Prompt Copilot with phrases like
“optimize this code”, “simplify logic”, or “make it more readable”

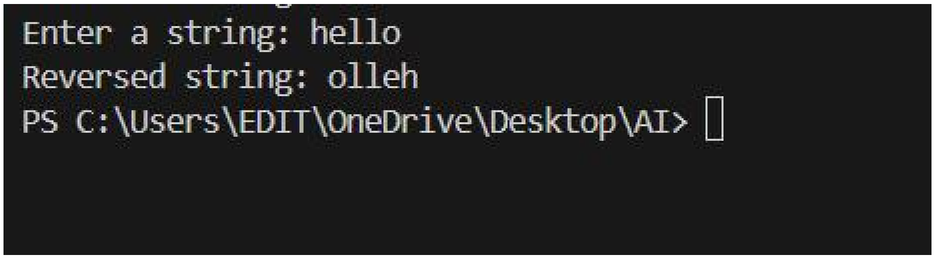
PROMPT

optimize this code, simplify logic, or make it more readable

CODE

print("Reversed string:", input("Enter a string: ")[::-1])

OUTPUT



```
Enter a string: hello
Reversed string: olleh
PS C:\Users\EDIT\OneDrive\Desktop\AI> 
```

EXPLANATION

Here,Task1 code is compressed and used efficiently,in only one line we can complete the code,like:in print statement both user input and reversed string logic must be done.It reduces time complexity.

TASK 3

Use GitHub Copilot to generate a function-based Python program that:

➢ Uses a user-defined function to reverse a string

➢ Returns the reversed string

➢ Includes meaningful comments (AI-assisted)

PROMPT

# Uses a user-defined function to reverse a string,Returns the reversed string,Includes meaningful comments (AI-assisted)

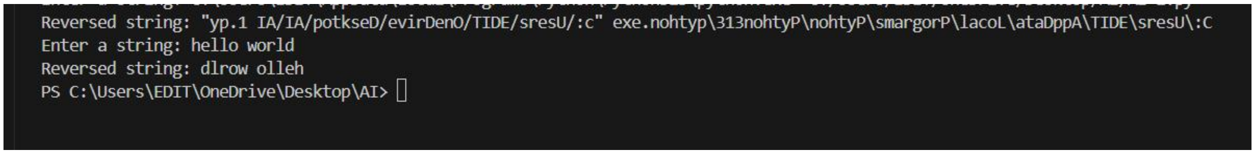
CODE

```
def reverse_string(s):
    """
    This function takes a string as input and returns the reversed string.

    Parameters:
    s (str): The string to be reversed.

    Returns:
    str: The reversed string.
    """
    return s[::-1]
# Get user input
user_input = input("Enter a string: ")
# Call the function and display the result
reversed_string = reverse_string(user_input)
print("Reversed string:", reversed_string)
```

OUTPUT



```
Reversed string: "yp.1 IA/IA/potkseD/evirDenO/TIDE/sresU/:c" exe.nohtyp\313nohtyP\nohtyP\smargorP\lacoL\ataDppA\TIDE\sresU\:C
Enter a string: hello world
Reversed string: dlrow olleh
PS C:\Users\EDIT\OneDrive\Desktop\AI> ▯
```

EXPLANATION

Using a user defined function i.e,def reversed we should reverse a string,Return the reversed string and call the function and print the output.

TASK 4

Compare the Copilot-generated programs:

- Without functions (Task 1)
- With functions (Task 3)

Analyze them based on:

- Code clarity
- Reusability
- Debugging ease
- Suitability for large-scale applications

PROMPT

Compare the two codes without function and with function.and make a comparison table based on code clarity,reusability,debugging ease,suitability for large scale applications.

EXPLANATION AND OUTPUT

```
comparison_table = {
    "Aspect": ["Code Clarity", "Reusability", "Debugging Ease", "Suitability for Large-Scale Applications"],
    "Without Function": [
        "Straightforward for simple tasks but less clear for complex logic.",
        "Limited to a single use case; not reusable.",
        "Harder to debug as logic is not isolated.",
        "Less adaptable to larger projects."
    ],
    "With Function": [
        "Clearer by encapsulating logic in a named function.",
        "Reusable; can be called multiple times with different inputs.",
        "Easier to debug due to isolated logic.",
        "More suitable for large-scale applications due to modularity."
    ]
}
```

TASK 5

Prompt GitHub Copilot to generate:

➢ A loop-based string reversal approach

➢ A built-in / slicing-based string reversal approach

PROMPT

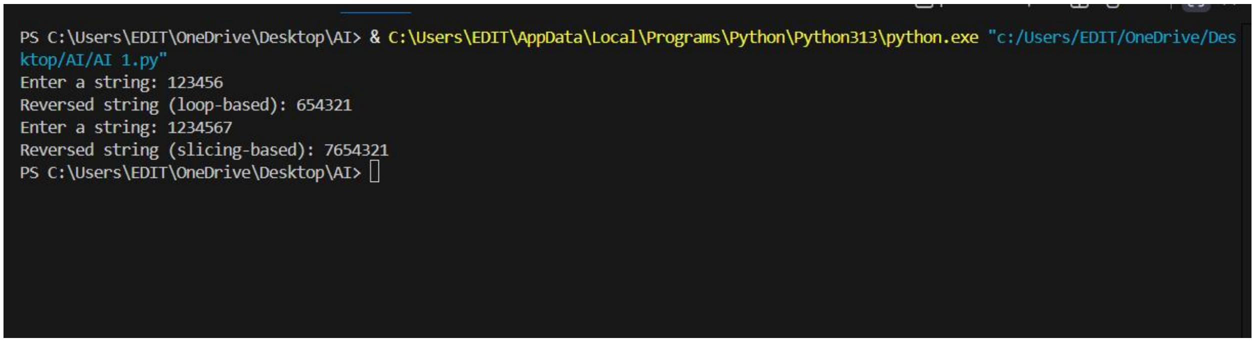
A loop-based string reversal approach

A built-in / slicing-based string reversal approach

CODE

```
# Loop-based string reversal
user_input = input("Enter a string: ")
reversed_string_loop = " "
for char in user_input:
    reversed_string_loop = char + reversed_string_loop
print("Reversed string (loop-based):", reversed_string_loop)
# Built-in / slicing-based string reversal
user_input = input("Enter a string: ")
reversed_string_slicing = user_input[::-1]
print("Reversed string (slicing-based):", reversed_string_slicing)
```

OUTPUT



```
PS C:\Users\EDIT\OneDrive\Desktop\AI> & C:\Users\EDIT\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/EDIT/OneDrive/Des
ktop/AI/AI 1.py"
Enter a string: 123456
Reversed string (loop-based): 654321
Enter a string: 1234567
Reversed string (slicing-based): 7654321
PS C:\Users\EDIT\OneDrive\Desktop\AI> 
```

EXPLANATION

Using a loop like: for write a code for reversing a string,then built-in / slicing-based string reversal approach and print the output.then compare the two codes The loop-based approach iterates through each character of the string, building the reversed string step by step. The slicing-based approach directly accesses the string in reverse order using Python's slicing syntax.Time Complexity:Both approaches have a time complexity of O(n), where n is the length of the string. However, the slicing approach may have a slight overhead due to the way Python handles slicing internally. When Each Approach is Appropriate:The loop-based approach is more appropriate when you need to perform additional operations on each character during the reversal process. The slicing-based approach is ideal for quick and straightforward string reversals without any additional processing.