Course Title: AI Assisted Coding

Course Code: 23CS002PC304

Faculty Name: Dr. R. Prashant Kumar

Name: Sherlin Varshitha

HT no: 2303A52266- Batch(36)

Question:

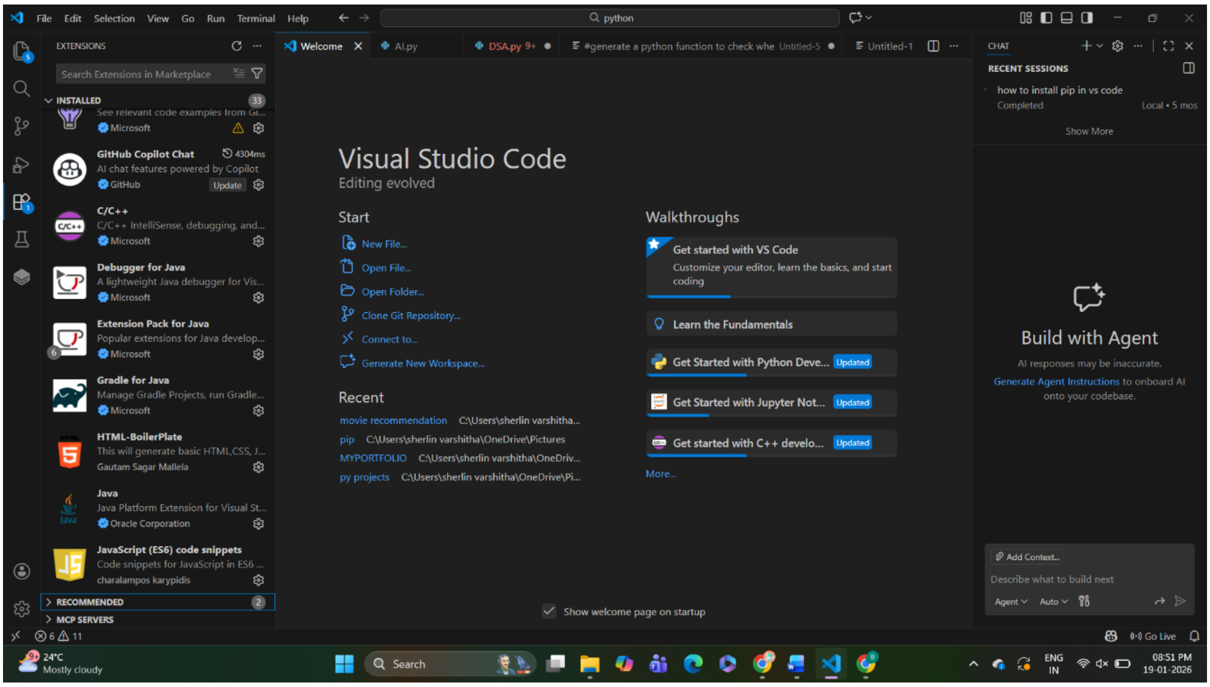
Lab 1: Environment Setup – GitHub Copilot and VS Code Integration + Understanding AI-assisted Coding Workflow

Task 0:

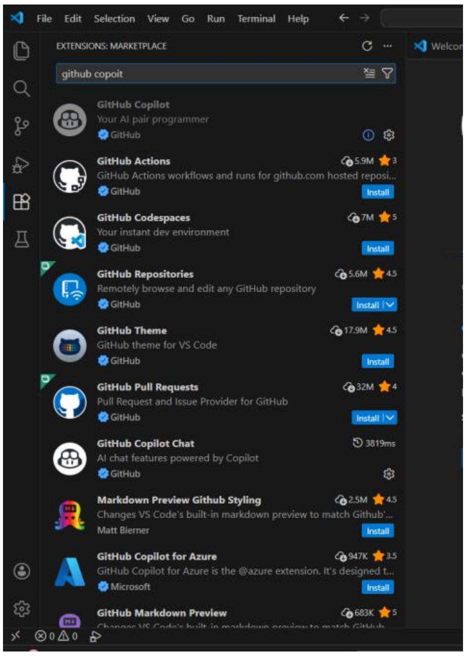
- Install and configure GitHub Copilot in VS Code. Take screenshots of each step.

Step 1: Open Visual Studio Code

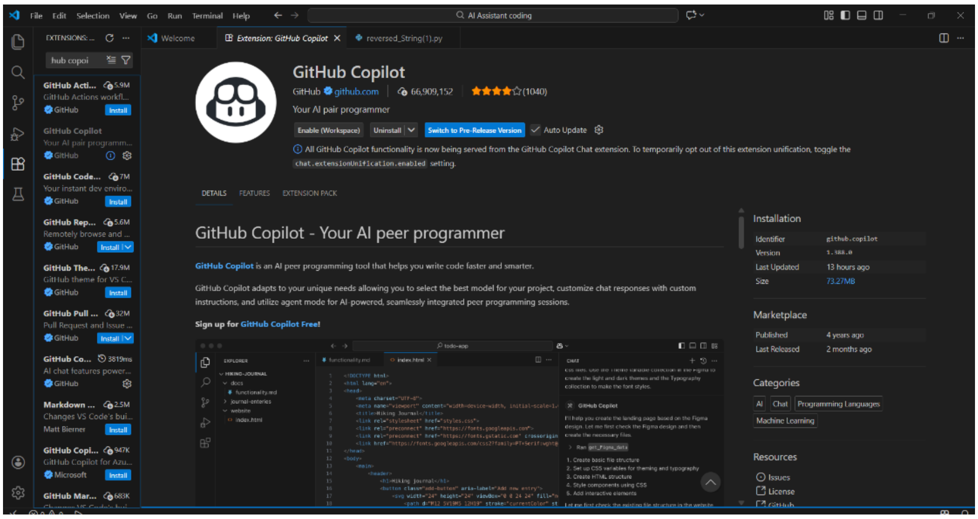
Step 2: Open Extensions Panel


File Edit Selection View Go Run Terminal Help
python
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 33
See relevant code examples from Gi...
Microsoft
GitHub Copilot Chat 4304ms
AI chat features powered by Copilot
GitHub Update
C/C++
C/C++ IntelliSense, debugging, and...
Microsoft
Debugger for Java
A lightweight Java debugger for Vis...
Microsoft
Extension Pack for Java
Popular extensions for Java develop...
Microsoft
Gradle for Java
Manage Gradle Projects, run Gradle...
Microsoft
HTML-BoilerPlate
This will generate basic HTML,CSS, J...
Gautam Sagar Mallela
Java
Java Platform Extension for Visual St...
Oracle Corporation
JavaScript (ES6) code snippets
Code snippets for JavaScript in ES6 ...
charalampos karypidis
RECOMMENDED 2
MCP SERVERS
Welcome
AI.py
DSA.py 9+
#generate a python function to check whe Untitled-5
Untitled-1
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Generate New Workspace...
Recent
movie recommendation C:\Users\sherlin varshitha...
pip C:\Users\sherlin varshitha\OneDrive\Pictures
MYPORTFOLIO C:\Users\sherlin varshitha\OneDriv...
py projects C:\Users\sherlin varshitha\OneDrive\Pi...
Walkthroughs
Get started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Get Started with Python Deve... Updated
Get Started with Jupyter Not... Updated
Get Started with C++ develo... Updated
More...
Show welcome page on startup
CHAT
RECENT SESSIONS
how to install pip in vs code
Completed Local • 5 mos
Show More
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Add Context...
Describe what to build next
Agent Auto
6 11
Go Live
24°C
Mostly cloudy
Search
ENG IN
08:51 PM
19-01-2026

Step 3: Search for GitHub Copilot


EXTENSIONS: MARKETPLACE
github copoit
GitHub Copilot
Your AI pair programmer
GitHub Actions
GitHub Actions workflows and runs for github.com hosted reposi...
GitHub Codespaces
Your instant dev environment
GitHub Repositories
Remotely browse and edit any GitHub repository
GitHub Theme
GitHub theme for VS Code
GitHub Pull Requests
Pull Request and Issue Provider for GitHub
GitHub Copilot Chat
AI chat features powered by Copilot
Markdown Preview Github Styling
Changes VS Code's built-in markdown preview to match Github'...
GitHub Copilot for Azure
GitHub Copilot for Azure is the @azure extension. It's designed t...
GitHub Markdown Preview
Install

Step 4: Install GitHub Copilot



GitHub Copilot

GitHub github.com | 66,909,152 | ★★★★☆(1040)

Your AI pair programmer

All GitHub Copilot functionality is now being served from the GitHub Copilot Chat extension. To temporarily opt out of this extension unification, toggle the `chat.extensionUnification.enabled` setting.

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
Version	1.388.0
Last Updated	13 hours ago
Size	73.27MB

Marketplace

Published	4 years ago
Last Released	2 months ago

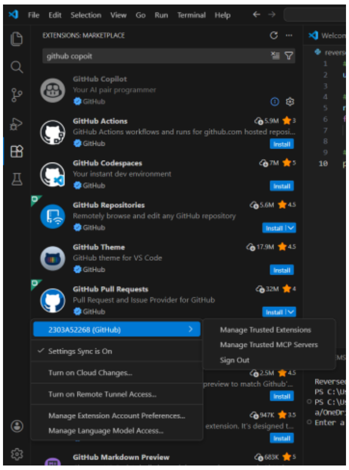
Categories

AI | Chat | Programming Languages | Machine Learning

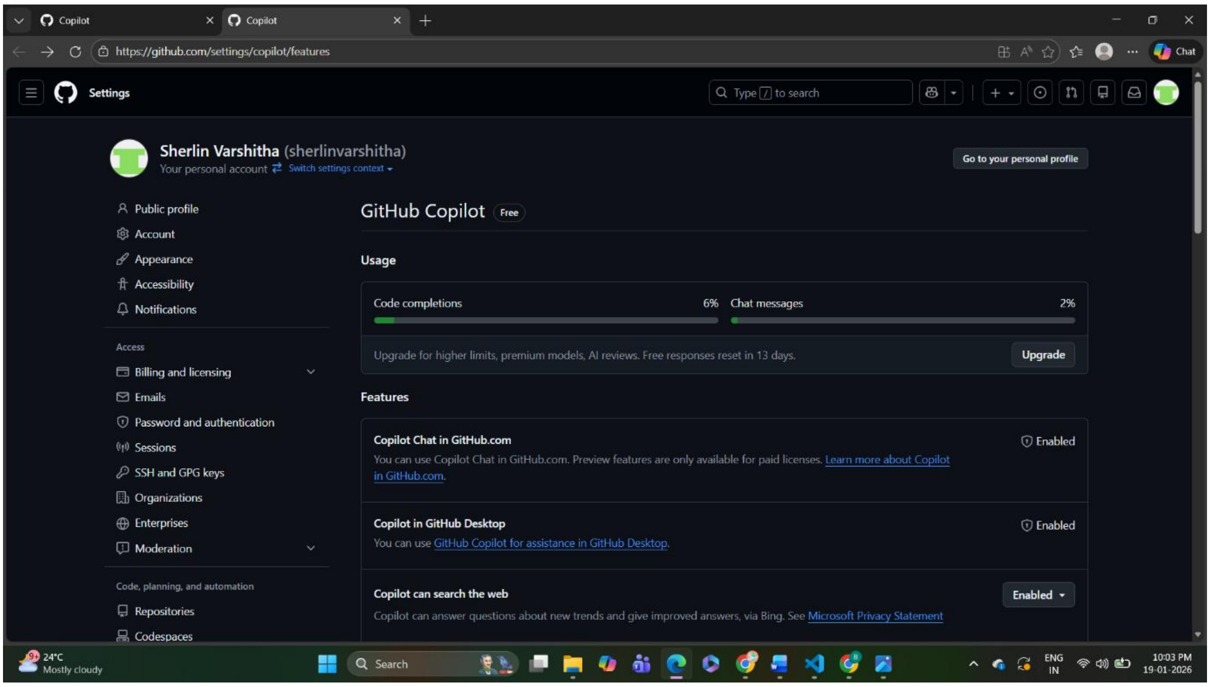
Resources

Issues
License

Step 5: Sign in to GitHub Account



Step 6: Authorize GitHub Copilot



Sherlin Varshitha (sherlinvarshitha)
Your personal account · Switch settings context

GitHub Copilot Free

Usage

Code completions 6%

Chat messages 2%

Upgrade for higher limits, premium models, AI reviews. Free responses reset in 13 days. Upgrade

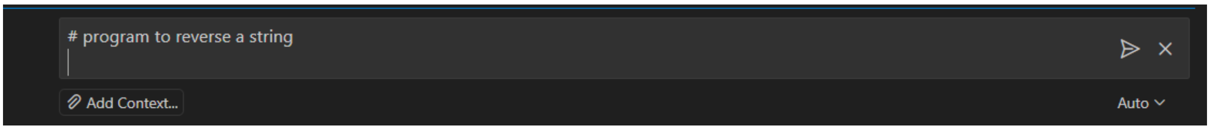
Features

Copilot Chat in GitHub.com — Enabled
You can use Copilot Chat in GitHub.com. Preview features are only available for paid licenses. Learn more about Copilot in GitHub.com.

Copilot in GitHub Desktop — Enabled
You can use GitHub Copilot for assistance in GitHub Desktop.

Copilot can search the web — Enabled
Copilot can answer questions about new trends and give improved answers, via Bing. See Microsoft Privacy Statement

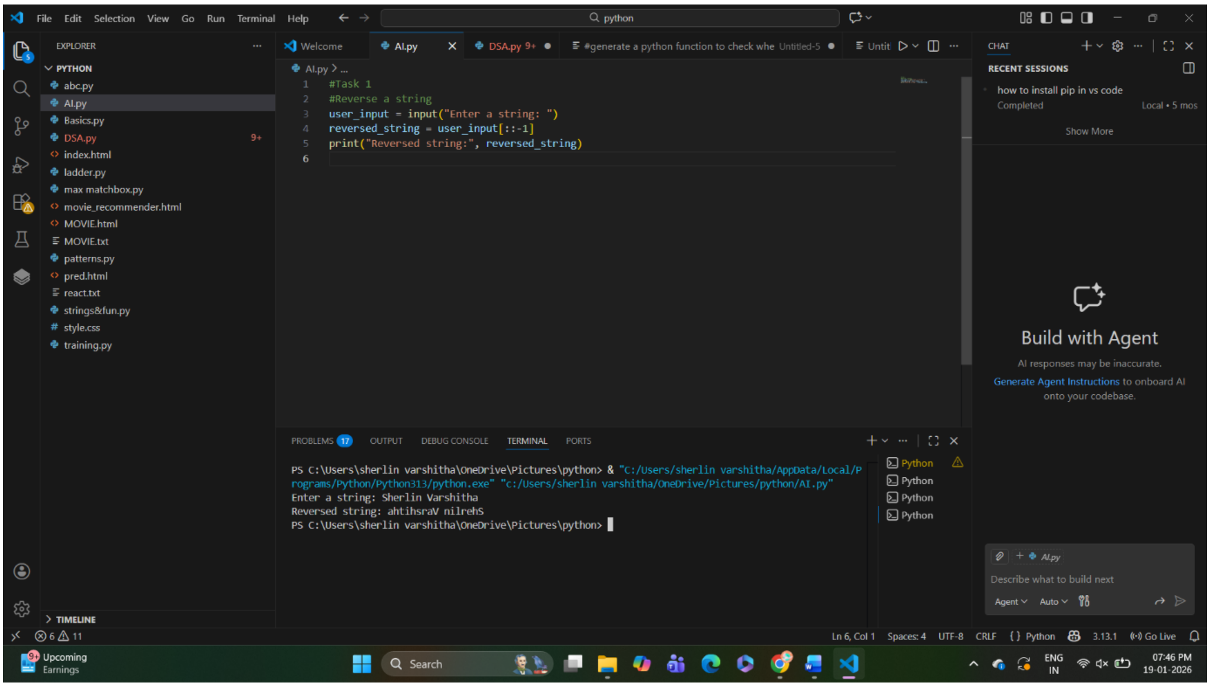
Step 7: Verify Copilot is Enabled



```
# program to reverse a string
```

Task 1: AI-Generated Logic Without Modularization (String Reversal Without Functions)

program to reverse a string



AI.py

```python
#Task 1
#Reverse a string
user_input = input("Enter a string: ")
reversed_string = user_input[::-1]
print("Reversed string:", reversed_string)

```

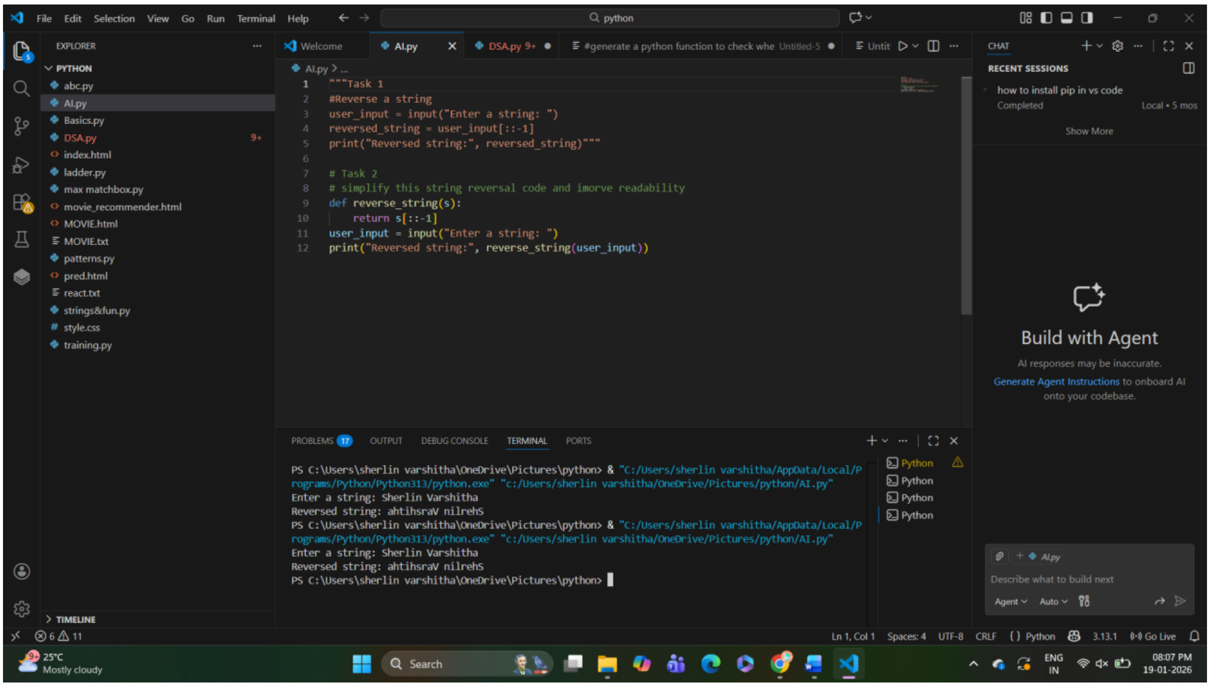
```
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python>
```

Explanation

- The input() function takes a string from the user.
- An empty string rev is created to store the reversed result.
- The for loop iterates through the string from the last character to the first.
- Each character is appended to rev.
- The final reversed string is printed.
- The logic is written directly in the main code without using functions

Task 2: Efficiency & Logic Optimization (Readability Improvement)

# Simplified String Reversal Code



```python
"""Task 1
#Reverse a string
user_input = input("Enter a string: ")
reversed_string = user_input[::-1]
print("Reversed string:", reversed_string)"""

# Task 2
# simplify this string reversal code and imorve readability
def reverse_string(s):
    return s[::-1]
user_input = input("Enter a string: ")
print("Reversed string:", reverse_string(user_input))
```

```
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python>
```

Explanation of Optimization

- The loop and extra variable were removed
- Python slicing reverses the string in a single step
- Code is shorter, cleaner, and easier to understand

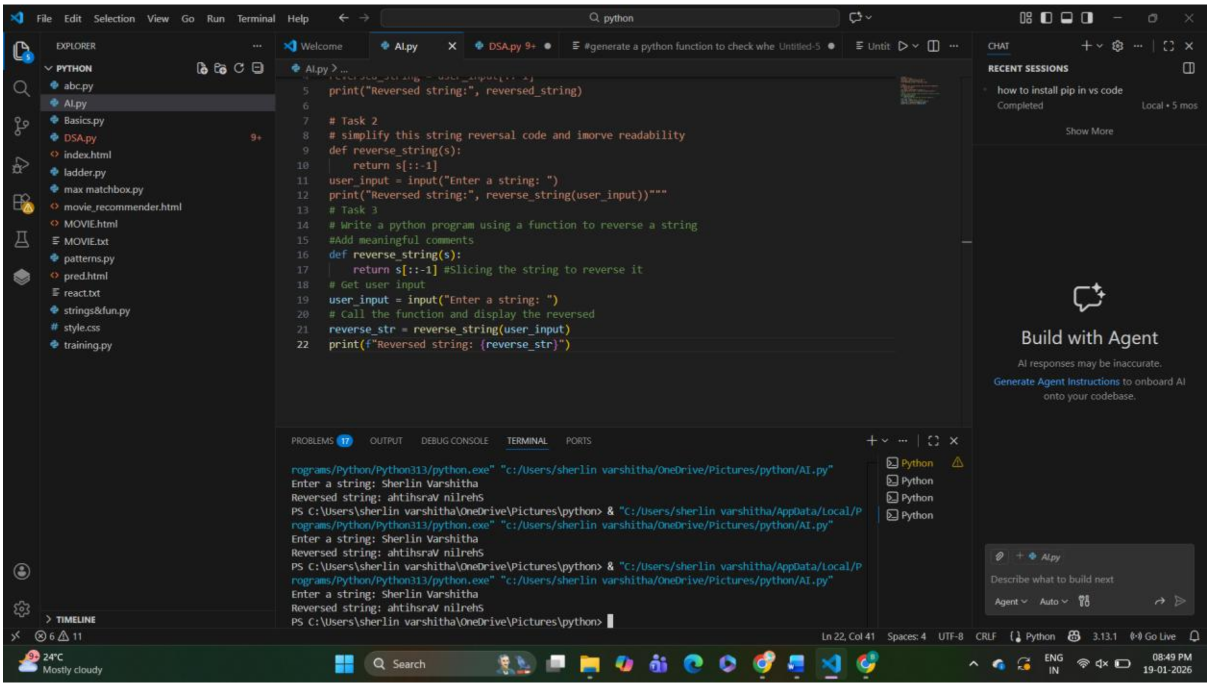
Time Complexity Explanation

- Original code: **O(n)** (manual loop)

- Optimized code: **O(n)** (built-in slicing)
- Although complexity remains the same, slicing is **faster in practice** due to internal optimization

Task 3: Modular Design Using AI Assistance (String Reversal Using Functions)

# Write a Python function to reverse a string



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

# Task 2
# simplify this string reversal code and imorve readability
def reverse_string(s):
    return s[::-1]
user_input = input("Enter a string: ")
print("Reversed string:", reverse_string(user_input))"""
# Task 3
# Write a python program using a function to reverse a string
#Add meaningful comments
def reverse_string(s):
    return s[::-1] #Slicing the string to reverse it
# Get user input
user_input = input("Enter a string: ")
# Call the function and display the reversed
reverse_str = reverse_string(user_input)
print(f"Reversed string: {reverse_str}")
```

```
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> 
```

Explanation

- A function reverse_string() is defined to reverse a string.
- The function takes one parameter text.
- The slicing method [::-1] is used to reverse the string.
- The reversed string is returned to the caller.
- User input is passed to the function.
- The result is printed.
- This modular approach improves reusability and readability.

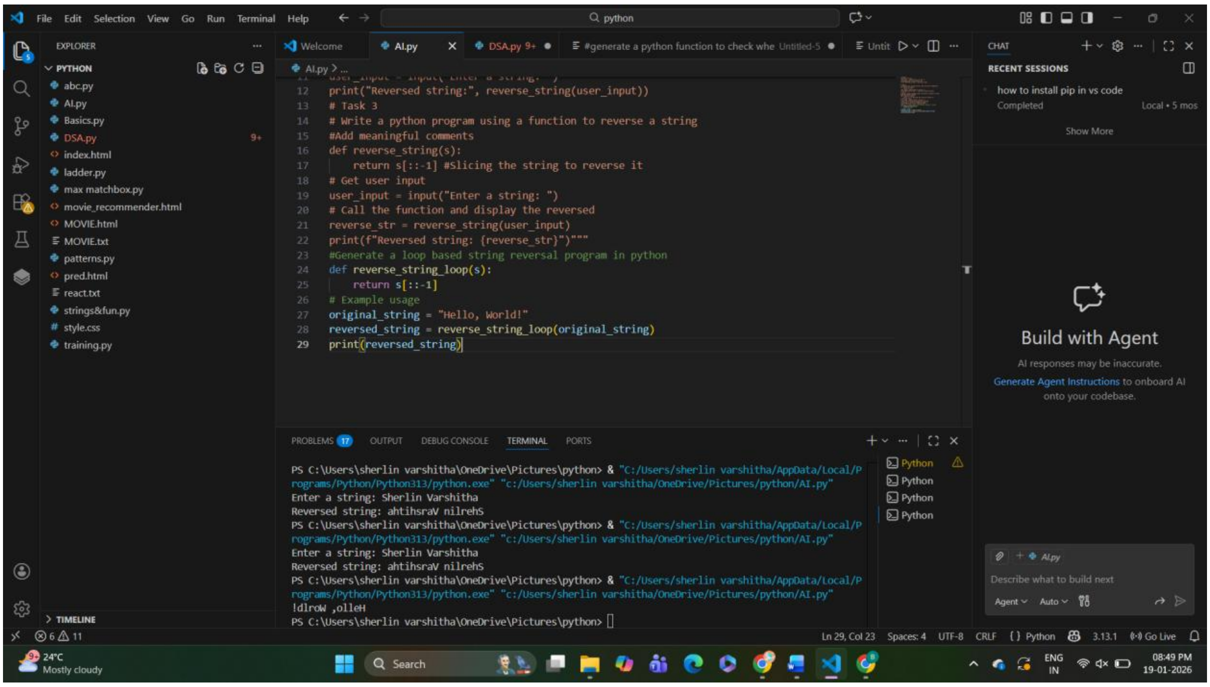
Task 4: Comparative Analysis – Procedural vs Modular Approach (With vs

Without Functions)

Aspect	Without Function (Procedural)	With Function (Modular)
Code Clarity	Moderate	High
Reusability	Not reusable	Highly reusable
Debugging	Difficult	Easier
Maintainability	Low	High
Large-scale Suitability	Poor	Good

Task 5: AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches to String Reversal)

#Generate a loop based string reversal program in Python



```python
user_input = input("Enter a string: ")
print("Reversed string:", reverse_string(user_input))
# Task 3
# Write a python program using a function to reverse a string
#Add meaningful comments
def reverse_string(s):
    return s[::-1] #Slicing the string to reverse it
# Get user input
user_input = input("Enter a string: ")
# Call the function and display the reversed
reverse_str = reverse_string(user_input)
print(f"Reversed string: {reverse_str}")"""
#Generate a loop based string reversal program in python
def reverse_string_loop(s):
    return s[::-1]
# Example usage
original_string = "Hello, World!"
reversed_string = reverse_string_loop(original_string)
print(reversed_string)
```

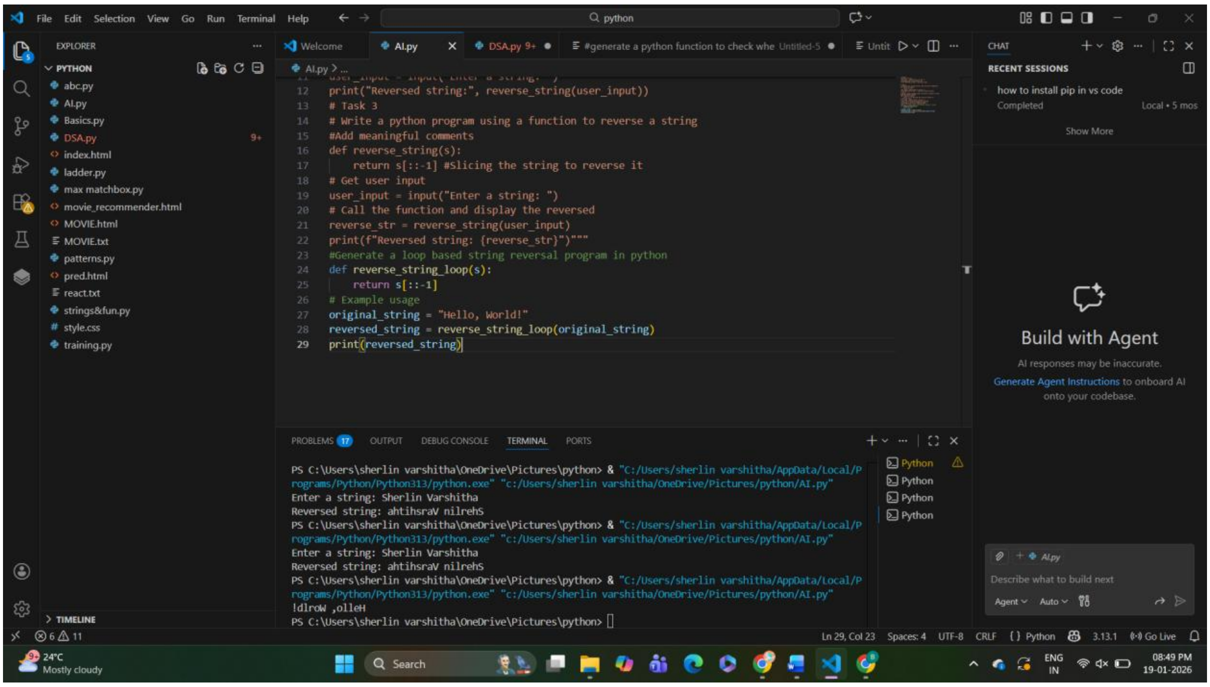
```
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
!dlroW ,olleH
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> 
```

Explanation

- The user inputs a string.
- An empty string rev is created.
- The loop reads each character from left to right.

- Each character is added at the beginning of rev, reversing the order.
- The reversed string is printed.
- This method helps understand string manipulation logic.

#Generate a slicing based string reversal program in Python



```
print("Reversed string:", reverse_string(user_input))
# Task 3
# Write a python program using a function to reverse a string
#Add meaningful comments
def reverse_string(s):
    return s[::-1] #Slicing the string to reverse it
# Get user input
user_input = input("Enter a string: ")
# Call the function and display the reversed
reverse_str = reverse_string(user_input)
print(f"Reversed string: {reverse_str}")"""
#Generate a loop based string reversal program in python
def reverse_string_loop(s):
    return s[::-1]
# Example usage
original_string = "Hello, World!"
reversed_string = reverse_string_loop(original_string)
print(reversed_string)
```

```
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
Enter a string: Sherlin Varshitha
Reversed string: ahtihsraV nilrehS
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python> & "C:/Users/sherlin varshitha/AppData/Local/P
rograms/Python/Python313/python.exe" "c:/Users/sherlin varshitha/OneDrive/Pictures/python/AI.py"
!dlroW ,olleH
PS C:\Users\sherlin varshitha\OneDrive\Pictures\python>
```

Explanation

- The string is taken from the user.
- Python slicing reverses the string efficiently.
- The reversed string is printed directly.
- This approach is best for large inputs and real-world applications.

Comparison of Approaches

Aspect	**Loop-Based**	**Slicing-Based**
Execution Flow	Step-by-step reversal	Single operation
Time Complexity	O(n)	O(n)
Performance for Large Inputs	Slower	Faster
Readability	Moderate	Very High

Aspect	**Loop-Based**	**Slicing-Based**
Best Usage	Learning logic	Production code