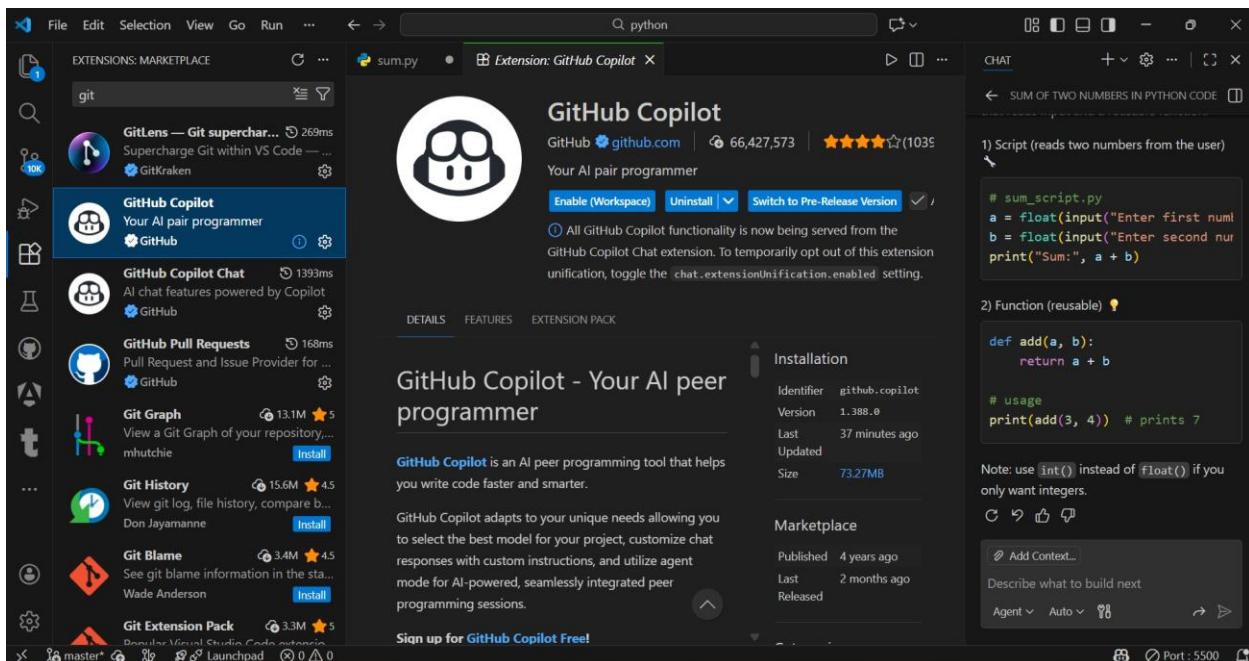
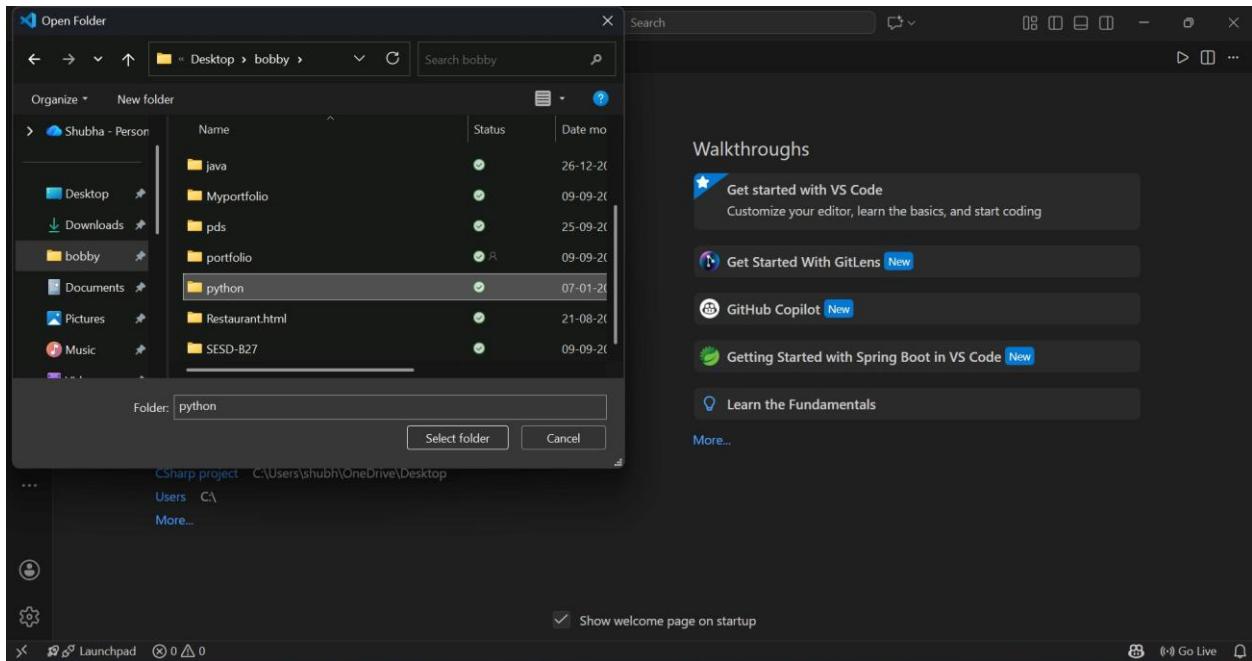


Lab Assignment 1.3

A.Nikhil (2303A51934)





Task 1: AI-Generated Logic Without Modularization (Procedural Fibonacci) :

```
# Fibonacci sequence
```

```

# Fibonacci sequence
n = int(input("Enter the number of terms: "))
a, b = 0, 1
count = 0
while count < n:
    print(a, end=' ')
    a, b = b, a + b
    count += 1

```

```

Active code page: 65001
C:\Users\shubh\OneDrive\Desktop\bobby>C:/Python314/python.exe c:/Users/shubh/OneDrive/Desktop/bobby
/pytho/Fibonacci.py
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby>

```

Task 2: AI Code Optimization & Cleanup :

```
# Optimize this Fibonacci code
```

Simplify variable usage

The screenshot shows a Python code editor in VS Code. The file Fibonacci.py contains the following code:

```
1 # Simplify variable usage
2 n = int(input("Enter the number of terms: "))
3 a, b = 0, 1
4 for _ in range(n):
5     print(a, end=' ')
6     a, b = b, a + b
```

The line "# Simplify variable usage" is highlighted. The terminal below shows the output of running the script:

```
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 5
0 1 1 2 3
```

Task 3: Modular Design Using AI Assistance (Function-Based Fibonacci) :

Write a Python function to generate Fibonacci sequence up to n
Use meaningful comments

The screenshot shows a Python code editor in VS Code. The file Fibonacci.py contains the following code:

```
1 # Write a Python function to generate Fibonacci sequence up to n
2 # Use meaningful comments
3 def fibonacci_sequence(n):
4     """Generate Fibonacci sequence up to n terms."""
5     sequence = []
6     a, b = 0, 1
7     for _ in range(n):
8         sequence.append(a)
9         a, b = b, a + b
10    return sequence
```

The terminal below shows the output of running the script:

```
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
```

Task 4: Comparative Analysis – Procedural vs Modular Code

Criteria	Without Functions	With Functions
Code Clarity	Lower	Higher
Reusability	No	Yes
Debugging	Harder	Easier
Scalability	Poor	Excellent
Suitable for Large Systems	No	Yes

Task 5: Iterative vs Recursive Fibonacci (AI-Generated):

Generate Fibonacci using iterative approach

The screenshot shows a Visual Studio Code (VS Code) interface with a dark theme. The left sidebar has a 'PYTHON' folder containing 'Fibonacci.py' and 'sum.py'. The main editor window displays the code for 'Fibonacci.py', which generates Fibonacci numbers using both iterative and recursive approaches. The terminal at the bottom shows the execution of the script and its output for 4 terms.

```
1 # Generate Fibonacci using iterative approach
2 # Generate Fibonacci using recursive approach
3 def fibonacci_iterative(n):
4     a, b = 0, 1
5     for _ in range(n):
6         print(a, end=' ')
7         a, b = b, a + b
8     print()
```

```
Active code page: 65001

C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby\python>
```

Generate Fibonacci using recursive approach

A screenshot of the Visual Studio Code (VS Code) interface. The window title is "File Edit Selection View Go Run ...". The top bar includes a search field for "python" and standard window controls. The left sidebar has icons for Explorer, Search, and others, with "EXPLORER" expanded to show "PYTHON" and files "Fibonacci.py" and "sum.py". The main area shows a code editor with the following Python code:

```
1 # Generate Fibonacci using recursive approach
2
3 def fibonacci_recursive(n):
4     if n <= 0:
5         return 0
6     elif n == 1:
7         return 1
8     else:
9         return fibonacci_recursive(n - 1) + fibonacci_recursive(n - 2)
```

The status bar at the bottom shows "Active code page: 65001" and a terminal history:

```
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby\python>python -u "c:\Users\shubh\OneDrive\Desktop\bobby\python\Fibonacci.py"
Enter the number of terms: 4
0 1 1 2
C:\Users\shubh\OneDrive\Desktop\bobby\python>
```

The status bar also displays "Ln 1, Col 1 Spaces: 4 UTF-8 LF [] Python 3.14.0 ⓘ Go Live ⚙️ Prettier 🎨".