Lab Assignment – 5.5

Hall Ticket No.: 2303A510E6

Name – Anushka Boora

Batch – 29

Task Description #1 (Transparency in Algorithm Optimization)

Task: Use AI to generate two solutions for checking prime numbers:

• Naive approach(basic)

• Optimized approach

Prompt:

“Generate Python code for two prime-checking methods and explain how the optimized version improves performance.”

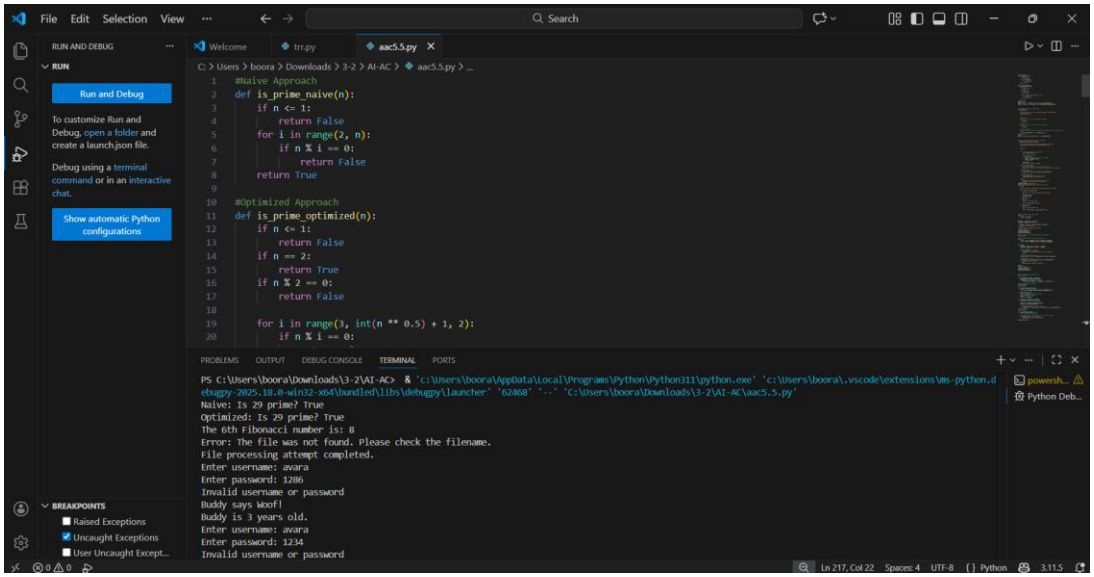
Expected Output:

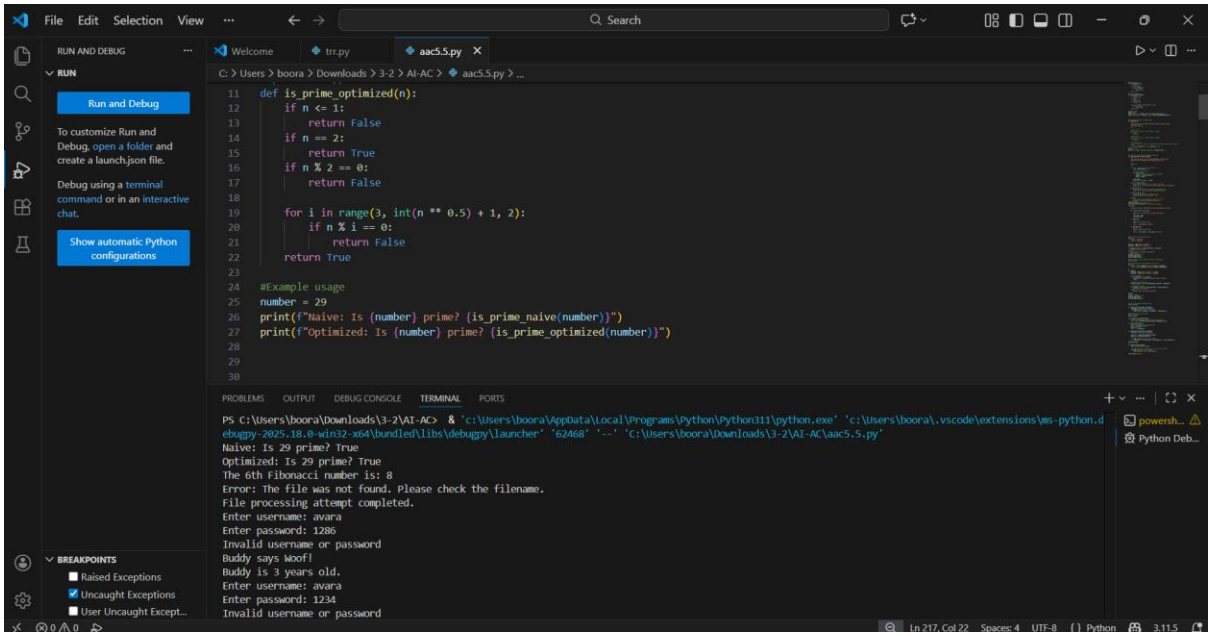
• Code for both methods.

• Transparent explanation of time complexity.

• Comparison highlighting efficiency improvements.

CODE & OUTPUT






Task Description #2 (Transparency in Recursive Algorithms)

Objective: Use AI to generate a recursive function to calculate

Fibonacci numbers.

Instructions:

1. Ask AI to add clear comments explaining recursion.

2. Ask AI to explain base cases and recursive calls.

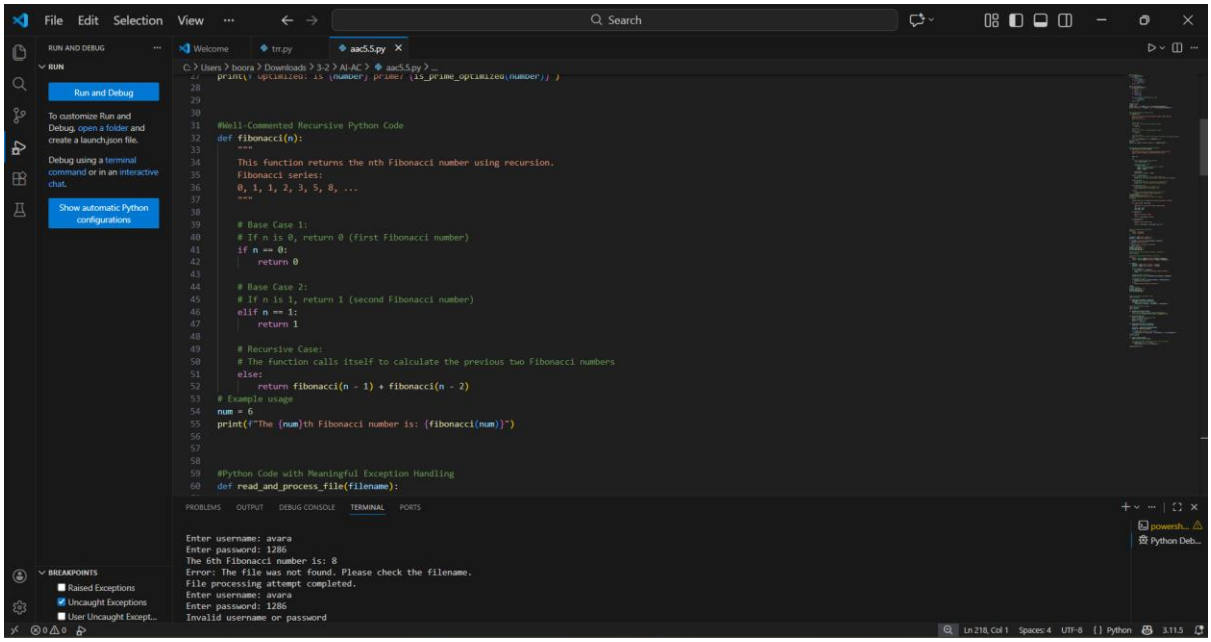
Expected Output:

• Well-commented recursive code.

• Clear explanation of how recursion works.

• Verification that explanation matches actual execution.

CODE & OUTPUT




Task Description #3 (Transparency in Error Handling)

Task: Use AI to generate a Python program that reads a file and processes data.

Prompt:

“Generate code with proper error handling and clear explanations for each exception.”

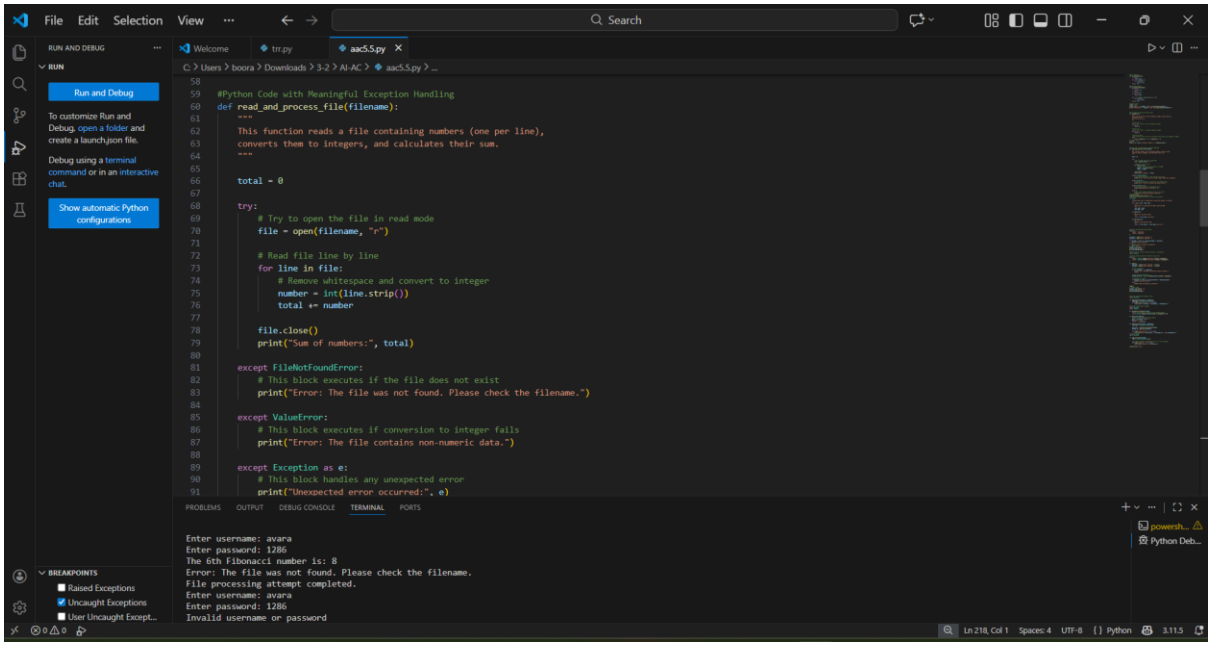
Expected Output:

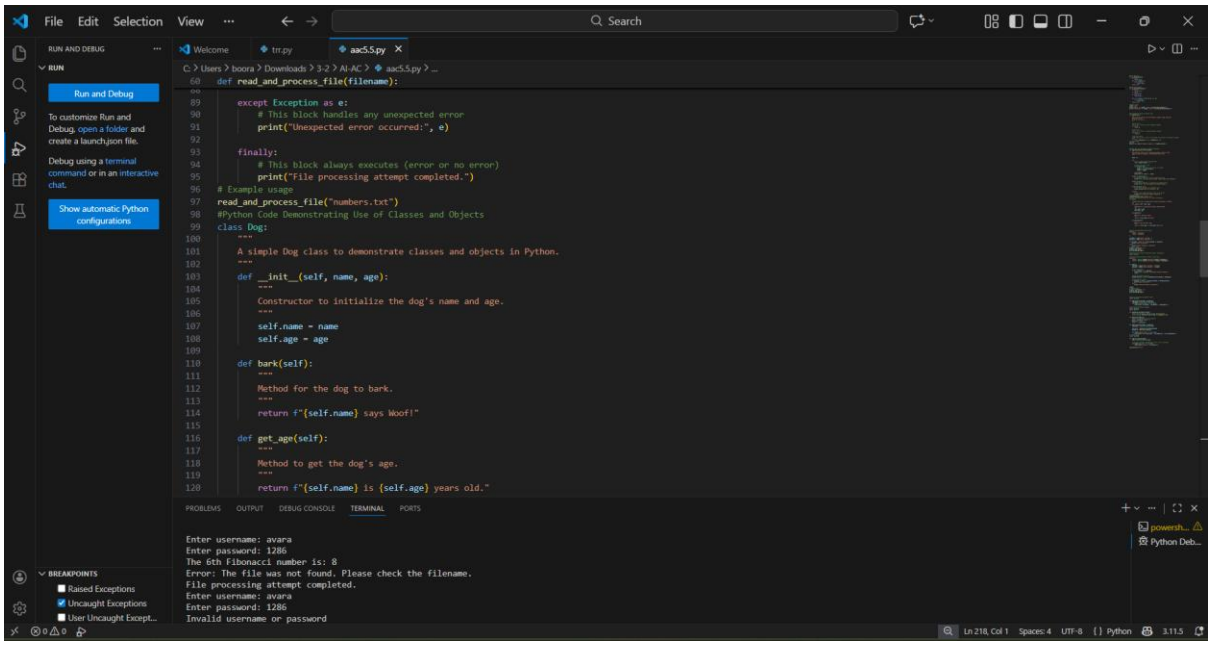
• Code with meaningful exception handling.

• Clear comments explaining each error scenario.

• Validation that explanations align with runtime behavior.

CODE & OUTPUT




Task Description #4 (Security in User Authentication)

Task: Use an AI tool to generate a Python-based login system.

Analyze: Check whether the AI uses secure password handling practices.

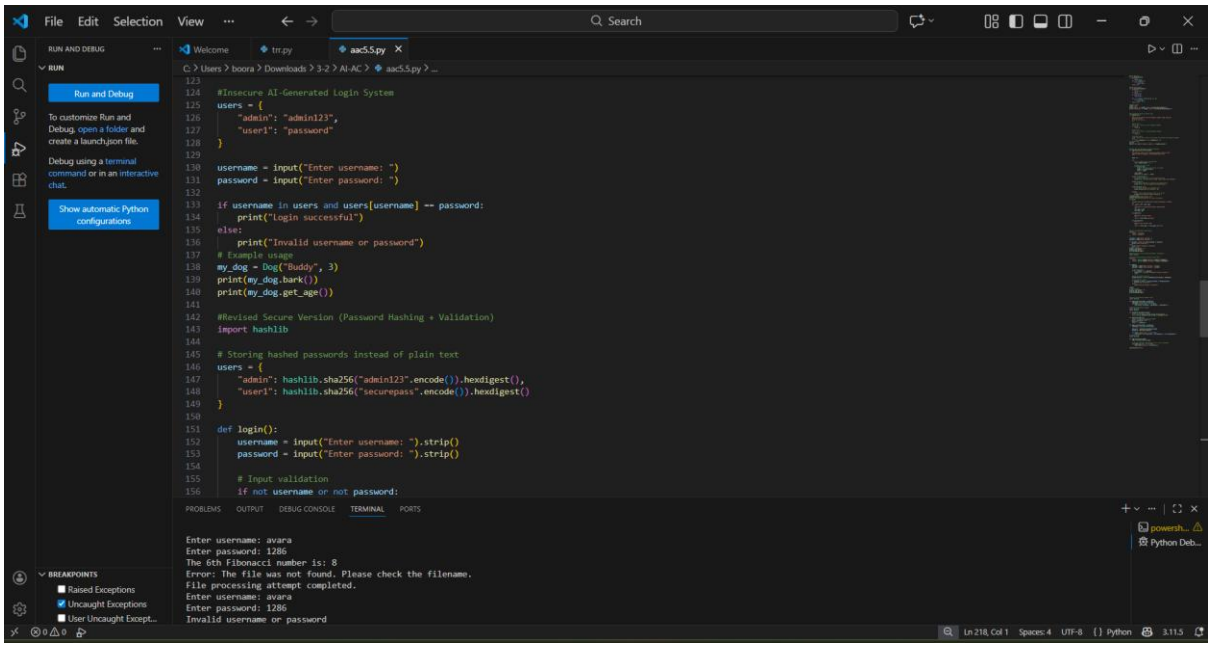
Expected Output:

• Identification of security flaws (plain-text passwords, weak validation).

• Revised version using password hashing and input validation.

• Short note on best practices for secure authentication.

CODE & OUTPUT




Task Description #5 (Privacy in Data Logging)

Task: Use an AI tool to generate a Python script that logs user activity (username, IP address, timestamp).

Analyze: Examine whether sensitive data is logged unnecessarily or insecurely.

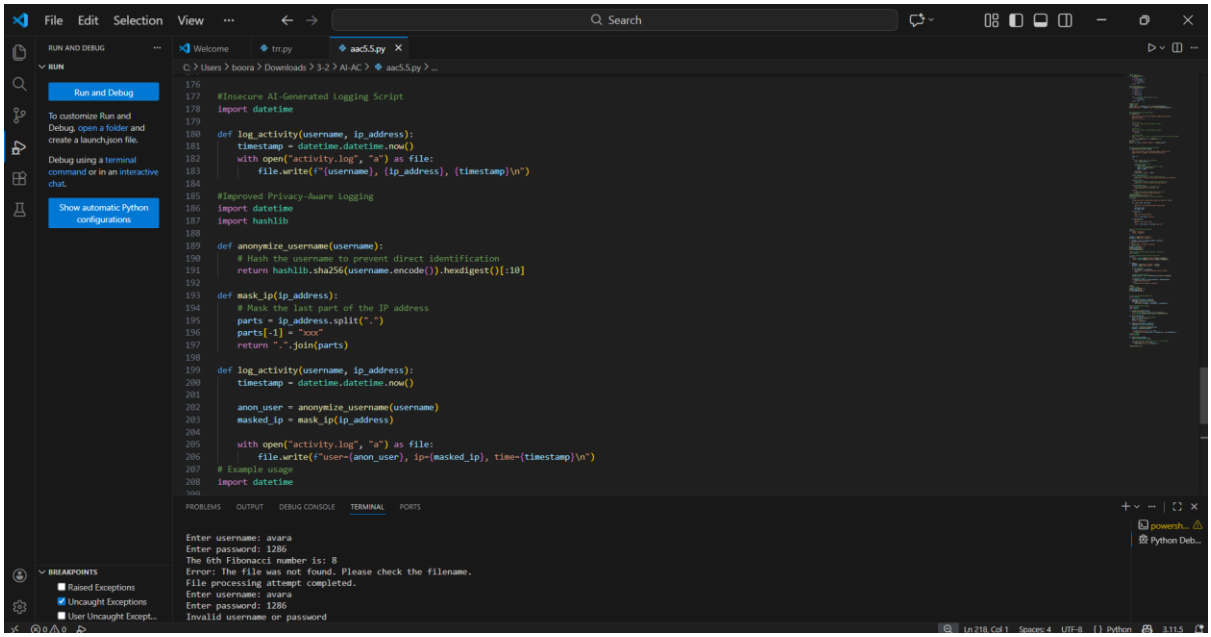
Expected Output:

• Identified privacy risks in logging.

• Improved version with minimal, anonymized, or masked logging.

• Explanation of privacy-aware logging principles.

CODE & OUTPUT


File Edit Selection View
Search
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug, open a folder and create a launch.json file.
Debug using a terminal command or in an interactive chat.
Show automatic Python configurations
BREAKPOINTS
Raised Exceptions
Uncaught Exceptions
User Uncaught Except...
Welcome
trr.py
aac5.5.py
C: > Users > boora > Downloads > 3-2 > AI-AC > aac5.5.py > ...
#Insecure AI-Generated Logging Script
import datetime
def log_activity(username, ip_address):
timestamp = datetime.datetime.now()
with open("activity.log", "a") as file:
file.write(f"{username}, {ip_address}, {timestamp}\n")
#Improved Privacy-Aware Logging
import datetime
import hashlib
def anonymize_username(username):
# Hash the username to prevent direct identification
return hashlib.sha256(username.encode()).hexdigest()[:10]
def mask_ip(ip_address):
# Mask the last part of the IP address
parts = ip_address.split(".")
parts[-1] = "xxx"
return ".".join(parts)
def log_activity(username, ip_address):
timestamp = datetime.datetime.now()
anon_user = anonymize_username(username)
masked_ip = mask_ip(ip_address)
with open("activity.log", "a") as file:
file.write(f"user={anon_user}, ip={masked_ip}, time={timestamp}\n")
# Example usage
import datetime
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter username: avara
Enter password: 1286
The 6th Fibonacci number is: 8
Error: The file was not found. Please check the filename.
File processing attempt completed.
Enter username: avara
Enter password: 1286
Invalid username or password
powersh...
Python Deb...
Ln 218, Col 1 Spaces: 4 UTF-8 Python 3.11.5