# AI Assisted Coding

## Assignment-8.1

Name:M Varshith Reddy

HTNO:2303A52087

Batch:40

**Task Description #1**

**(Password Strength Validator – Apply AI in Security Context)**

• Task: Apply AI to generate at least 3 assert test cases for is_strong_password(password) and implement the validator function.

• Requirements:

o Password must have at least 8 characters.

o Must include uppercase, lowercase, digit, and special character.

o Must not contain spaces.

**Example Assert Test Cases:**

assert is_strong_password("Abcd@123") == True

assert is_strong_password("abcd123") == False

assert is_strong_password("ABCD@1234") == True

**Code:**

```
Lab_assign-8.1.py > ...
  1   """Lab 8: Test-Driven Development with AI"""
  2   from typing import Dict
  3
  4   # TASK 1: Password Strength Validator | Req: 8+ chars, uppercase, lowercase, digit, special, no spaces
  5   # assert is_strong_password("Abcd@123")==True; assert is_strong_password("abcd123")==False
  6   def is_strong_password(p:str)->bool:
  7       return isinstance(p,str) and len(p)>=8 and ' ' not in p and any(c.isupper() for c in p) and any(c.islower() for c in p) and any(c.
           isdigit() for c in p) and any(c in '!@#$%^&*()_+-=[]{};\'",./<>?\\|`~' for c in p)
  8
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                                      Python  + ∨ ⬚ 🗑 ⋯ ∷

PS C:\Users\varsh\OneDrive\Documents\SEM 3-2\AIAC> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Documents/SEM 3-2/AIAC/Lab_ass
8.3.py"
3. Anagram Checker
4. Inventory Manager
5. Date Validation & Formatting
6. Run All Tests
7. Exit
------------------------------------------------
Enter your choice (1-7): 1
Enter password: Varshith@1256
Password 'Varshith@1256' is STRONG

Enter your choice (1-7): varshith@123
Invalid choice. Please try again.

Enter your choice (1-7): █
```

**Task Description #2**

**(Number Classification with Loops – Apply AI for Edge Case Handling)**

• Task: Use AI to generate at least 3 assert test cases for a classify_number(n) function. Implement using loops.

• Requirements:

o Classify numbers as Positive, Negative, or Zero.

o Handle invalid inputs like strings and None.

o Include boundary conditions (-1, 0, 1).

**Example Assert Test Cases:**

assert classify_number(10) == "Positive"

assert classify_number(-5) == "Negative"

assert classify_number(0) == "Zero"

**Code:**

```
assignment_8.py > ⓒ classify_number
  7          return isinstance(p,str) and len(p)>=8 and ' ' not in p and any(c.isupper() for c i
  8
  9      # TASK 2: Number Classification | Req: Classify as Positive/Negative/Zero, handle inval
 10      # assert classify_number(10)=="Positive"; assert classify_number(-5)=="Negative"; asser
 11      def classify_number(n)->str:
 12          if n is None or isinstance(n,(str,bool)):return "Invalid"
 13          try:num=float(n) if not isinstance(n,(int,float)) else n
 14          except:return "Invalid"
 15          return "Positive" if num>0 else ("Negative" if num<0 else "Zero")
 16
 17      # TASK 3: Anagram Checker | Req: Ignore case/spaces/punctuation, handle edge cases
 18      # assert is_anagram("listen","silent")==True; assert is_anagram("hello","world")==False
 19      def is_anagram(s1:str,s2:str)->bool:
 20          if not isinstance(s1,str) or not isinstance(s2,str):return False
 21          c1=''.join(x.lower() for x in s1 if x.isalnum())
 22          c2=''.join(x.lower() for x in s2 if x.isalnum())
 23          return (len(c1)==0 and len(c2)==0) or (len(c1)>0 and len(c2)>0 and sorted(c1)==sort
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

```
Enter your choice (1-7): 2
Enter a number: 3456789
Number classification: Positive

Enter your choice (1-7): 2
Enter a number: -23456
Number classification: Negative

Enter your choice (1-7): 2
Enter a number: 00000000000
Number classification: Zero
```

**Task Description #3 (Anagram Checker – Apply AI for String Analysis)**

• Task: Use AI to generate at least 3 assert test cases for is_anagram(str1, str2) and implement the function.

• Requirements:

o Ignore case, spaces, and punctuation.

o Handle edge cases (empty strings, identical words).

**Example Assert Test Cases:**

assert is_anagram("listen", "silent") == True

assert is_anagram("hello", "world") == False

assert is_anagram("Dormitory", "Dirty Room") == True

**Code:**

```
assignment_8.py > classify_number
17    # TASK 3: Anagram Checker | Req: Ignore case/spaces/punctuation, handle edge cases
18    # assert is_anagram("listen","silent")==True; assert is_anagram("hello","world")==False
19    def is_anagram(s1:str,s2:str)->bool:
20        if not isinstance(s1,str) or not isinstance(s2,str):return False
21        c1=''.join(x.lower() for x in s1 if x.isalnum())
22        c2=''.join(x.lower() for x in s2 if x.isalnum())
23        return (len(c1)==0 and len(c2)==0) or (len(c1)>0 and len(c2)>0 and sorted(c1)==sorted(c2
24
25    # TASK 4: Inventory Class | Methods: add_item(name, qty), remove_item(name, qty), get_stock(
26    # inv.add_item("Pen",10); assert inv.get_stock("Pen")==10; inv.remove_item("Pen",5); assert
27    class Inventory:
28        def __init__(self):self.stock:Dict[str,int]={}
29        def add_item(self,n:str,q:int)->None:
30            if not isinstance(n,str) or not isinstance(q,int) or isinstance(q,bool):raise TypeEr
31            if q<0:raise ValueError("Quantity cannot be negative")
32            self.stock[n]=self.stock.get(n,0)+q
33        def remove_item(self,n:str,q:int)->None:

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS                                    + v

Enter a number: 00000000000
Number classification: Zero

Enter your choice (1-7): 3
Enter first string: sri kanth
Enter second string: sri kanth
'sri kanth' and 'sri kanth' are ANAGRAMS

Enter your choice (1-7): 3
Enter first string: anagaram
Enter second string: nangr
'anagaram' and 'nangr' are NOT ANAGRAMS
```
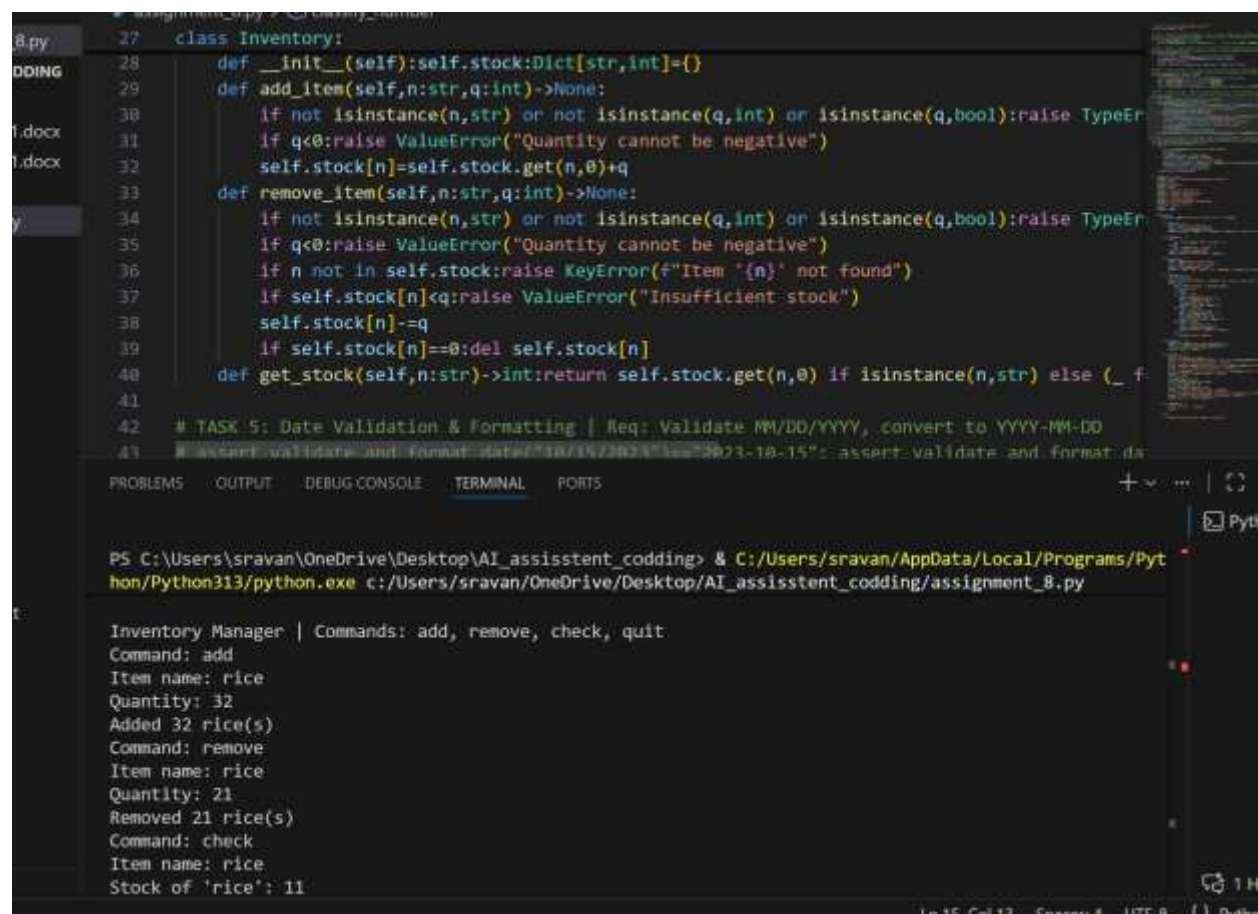
## Task Description #4

**(Inventory Class – Apply AI to Simulate Real- World Inventory System)**

• Task: Ask AI to generate at least 3 assert-based tests for an Inventory class with stock management.

• Methods:

o add_item(name, quantity)

o remove_item(name, quantity)

o get_stock(name)

**Example Assert Test Cases:**

inv = Inventory()

inv.add_item("Pen", 10)

assert inv.get_stock("Pen") == 10

inv.remove_item("Pen", 5)

assert inv.get_stock("Pen") == 5

inv.add_item("Book", 3)

assert inv.get_stock("Book") == 3

**Code:**



**Task Description #5**

**(Date Validation & Formatting – Apply AI for Data Validation)**

• Task: Use AI to generate at least 3 assert test cases for validate_and_format_date(date_str) to check and convert dates.

• Requirements:

o Validate "MM/DD/YYYY" format.

o Handle invalid dates.

o Convert valid dates to "YYYY-MM-DD".

**Example Assert Test Cases:**

assert validate_and_format_date("10/15/2023") == "2023-10-15"

assert validate_and_format_date("02/30/2023") == "Invalid Date"

assert validate_and_format_date("01/01/2024") == "2024-01-01"

**Code:**

```
 assignment_8.py > classify_number
41
42     # TASK 5: Date Validation & Formatting | Req: Validate MM/DD/YYYY, convert to YYYY-MM-DD
43     # assert validate_and_format_date("10/15/2023")=="2023-10-15"; assert validate_and_format_da
44     def validate_and_format_date(d:str)->str:
45         try:
46             if not isinstance(d,str):return "Invalid Date"
47             p=d.split('/')
48             if len(p)!=3:return "Invalid Date"
49             m,da,y=int(p[0]),int(p[1]),int(p[2])
50             if m<1 or m>12 or da<1:return "Invalid Date"
51             dm=[31,29 if y%4==0 and (y%100!=0 or y%400==0) else 28,31,30,31,30,31,31,30,31,30,31
52             if da>dm[m-1]:return "Invalid Date"
53             return f"{y:04d}-{m:02d}-{da:02d}"
54         except:return "Invalid Date"
55
56     if __name__=="__main__":
57         print("\nLab 8: Test-Driven Development with AI\n")
58         print("="*50)
59         print("Main Menu")
60         print("="*50)
61         print("1. Password Validator")
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS


Enter your choice (1-7): 5
Enter date (MM/DD/YYYY): 12122004
Result: Invalid Date

Enter your choice (1-7): 5
Enter date (MM/DD/YYYY): 01122026
Result: Invalid Date

Enter your choice (1-7):
```