AI ASSISTED CODING

ASSIGNMENT 9.2

Name: B.Rakshith

batch no: 11

hall ticket no:2403A51251

**Task 1: Google-Style Docstrings for Python Functions**

**Goal:**

Add **Google-style docstrings** to *all* functions in a script.

**What to do:**

* Give AI the code **without examples or explanations**.
* Ask AI to include:
  + Description of the function
  + Parameters with type hints
  + Return type
  + Example usage

**Expected Output:**

A Python script with **Google-style docstrings**, e.g.:

def calculate\_area(radius: float) -> float:

    """

    Calculates the area of a circle given its radius.

    Args:

        radius (float): The radius of the circle.

    Returns:

        float: The area of the circle.

    Example:

        >>> calculate\_area(3)

        28.27

    """

    return 3.14 \* radius \* radius

**Task 2: Inline Comments for Complex Logic**

**Goal:**

Add **inline comments** only for *complex or non-obvious* parts of the code.

**What to do:**

* Share code with **no comments**.
* Instruct AI to:
  + **Skip basic syntax**
  + Focus on non-trivial, tricky, or unclear logic
* Review for clarity and usefulness.

**Expected Output:**

Code with **targeted comments** like:

# Using a deque to efficiently manage a sliding window

window = deque()

# This loop maintains the max value within the window size

for i in range(len(nums)):

    # Remove smaller values that are no longer useful

    while window and nums[i] > nums[window[-1]]:

        window.pop()

**Task 3: Module-Level Docstring**

**Goal:**

Create a **top-of-file docstring** describing the whole module.

**What to do:**

* Supply the **entire .py file**.
* Ask AI to summarize:
  + Purpose of the file
  + Main functions/classes
  + Dependencies (imports)
  + Usage

**Expected Output:**

At the top of the file:

"""

This module provides utility functions for file I/O and data validation.

It includes:

- read\_csv(): Reads a CSV file into a DataFrame.

- validate\_schema(): Validates data against a predefined schema.

Dependencies:

- pandas

- jsonschema

**Task 4: Convert Inline Comments to Docstrings**

**Goal:**

Turn **existing inline comments** into **Google-style function docstrings**.

**What to do:**

* Provide code with inline comments.
* Ask AI to **move relevant info** into docstrings and **remove** old comments.
* Ensure formatting is clean and meaning preserved.

**Expected Output:**

Before:

# Calculates the average of a list

def average(numbers):

    # Sum all numbers

    total = sum(numbers)

    # Divide by count

    return total / len(numbers)

After:

def average(numbers: List[float]) -> float:

    """

    Calculates the average of a list of numbers.

    Args:

        numbers (List[float]): A list of numeric values.

    Returns:

        float: The average of the list.

    """

    total = sum(numbers)

    return total / len(numbers)

**Task 5: Review and Correct Docstrings**

**Goal:**

Fix **outdated or incorrect** docstrings using AI.

**What to do:**

* Provide AI with code that has **wrong or outdated docstrings**.
* Instruct AI to **analyze function behavior** and update docstrings accordingly.

**Expected Output:**

Before:

def multiply(a, b):

    """Adds two numbers."""

    return a \* b

After:

def multiply(a: int, b: int) -> int:

    """

    Multiplies two numbers and returns the result.

    Args:

        a (int): The first number.

        b (int): The second number.

    Returns:

        int: The product of a and b.

    """

    return a \* b

**Task 6: Prompt Comparison Experiment**

**Goal:**

Compare documentation quality between a **vague** vs **detailed** prompt.

**What to do:**

* Use **the same function** with:
  + Prompt A: "Add comments to this function"
  + Prompt B: "Add Google-style docstrings with parameters, return types, and examples"
* Record and compare AI outputs.

**Expected Output:**

A **comparison table**, like:

| **Prompt** | **Output Quality** | **Completeness** | **Format** | **Observations** |
| --- | --- | --- | --- | --- |
| "Add comments" | Low | Partial | No format | Misses return type and example |
| "Add Google-style..." | High | Full | Google-style | Clear, detailed docstring with example |