AI ASSISTANT CODING ASSIGNMENT - 2

NAME: JILLELA AKSHAYA

HT.NO: 2303A51629

BATCH: 22

LAB 2

Exploring Additional AI Coding Tools beyond Copilot – Gemini (Colab) and Cursor AI

Task 1: Cleaning Sensor Data

❖ **Scenario:**

❖ **You are cleaning IoT sensor data where negative values are invalid.**

❖ **Task:**

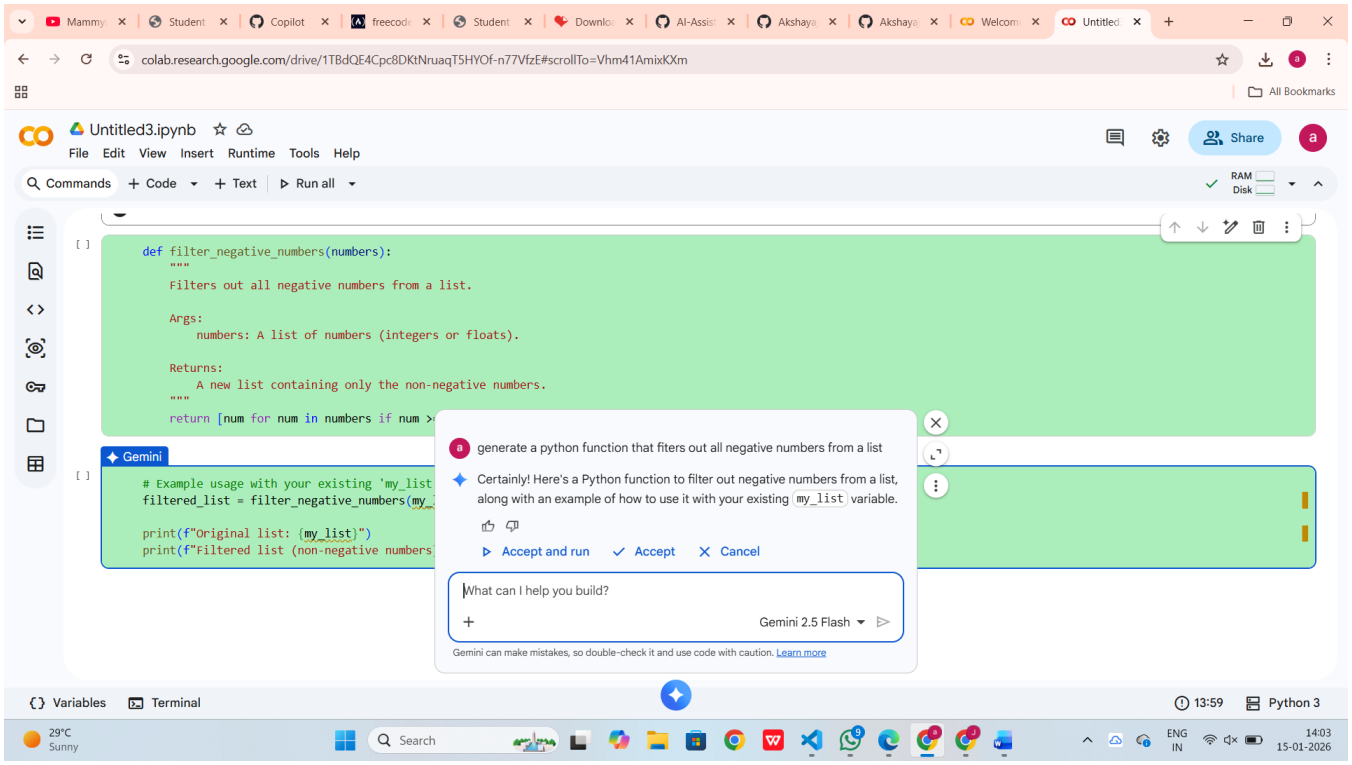
Use Gemini in Colab to generate a function that filters out all negative numbers from a list.

❖ **Expected Output:**

➢ **Before/after list**

➢ **Screenshot of Colab execution**

CODE :



OUTPUT:

```
    Args:
        numbers: A list of numbers (integers or floats).

    Returns:
        A new list containing only the non-negative numbers.
    """
    return [num for num in numbers if num >= 0]
```

```
# Example usage with your existing 'my_list'
filtered_list = filter_negative_numbers(my_list)

print(f"Original list: {my_list}")
print(f"Filtered list (non-negative numbers): {filtered_list}")
```

```
Original list: [-5, 10, -2, 8, 0, -1]
Filtered list (non-negative numbers): [10, 8, 0]
```

Task 2: String Character Analysis

❖ Scenario:

You are building a text-analysis feature.

❖ Task:

Use Gemini to generate a Python function that counts vowels, consonants,

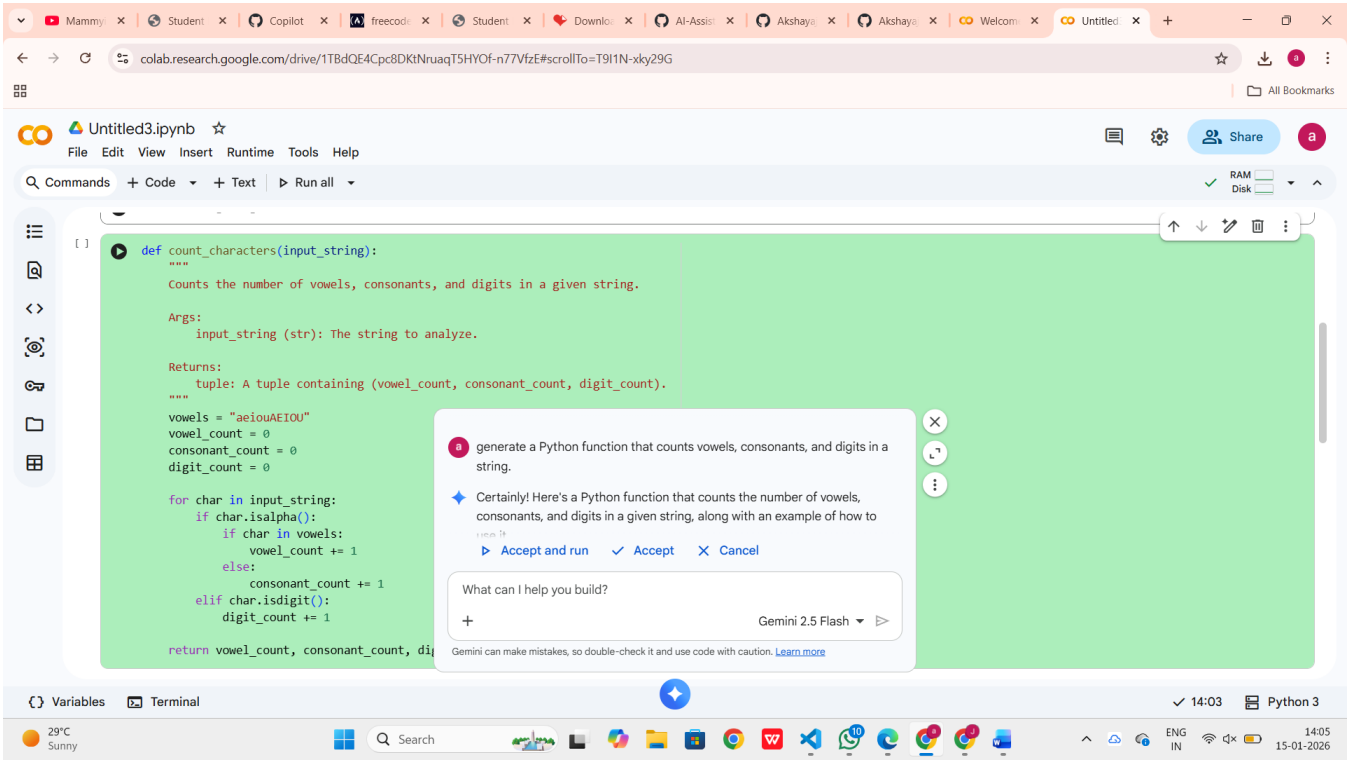
and digits in a string.

❖ **Expected Output:**

➢ **Working function**

➢ **Sample inputs and outputs**

CODE :



OUTPUT:

```
return vowel_count, consonant_count, digit_count
```

```
# Example usage:
sample_string = "Hello World 123!"
v, c, d = count_characters(sample_string)

print(f"Original string: '{sample_string}'")
print(f"Number of vowels: {v}")
print(f"Number of consonants: {c}")
print(f"Number of digits: {d}")
```

Task 3: Palindrome Check – Tool Comparison

❖ **Scenario:**

You must decide which AI tool is clearer for string logic.

Task:

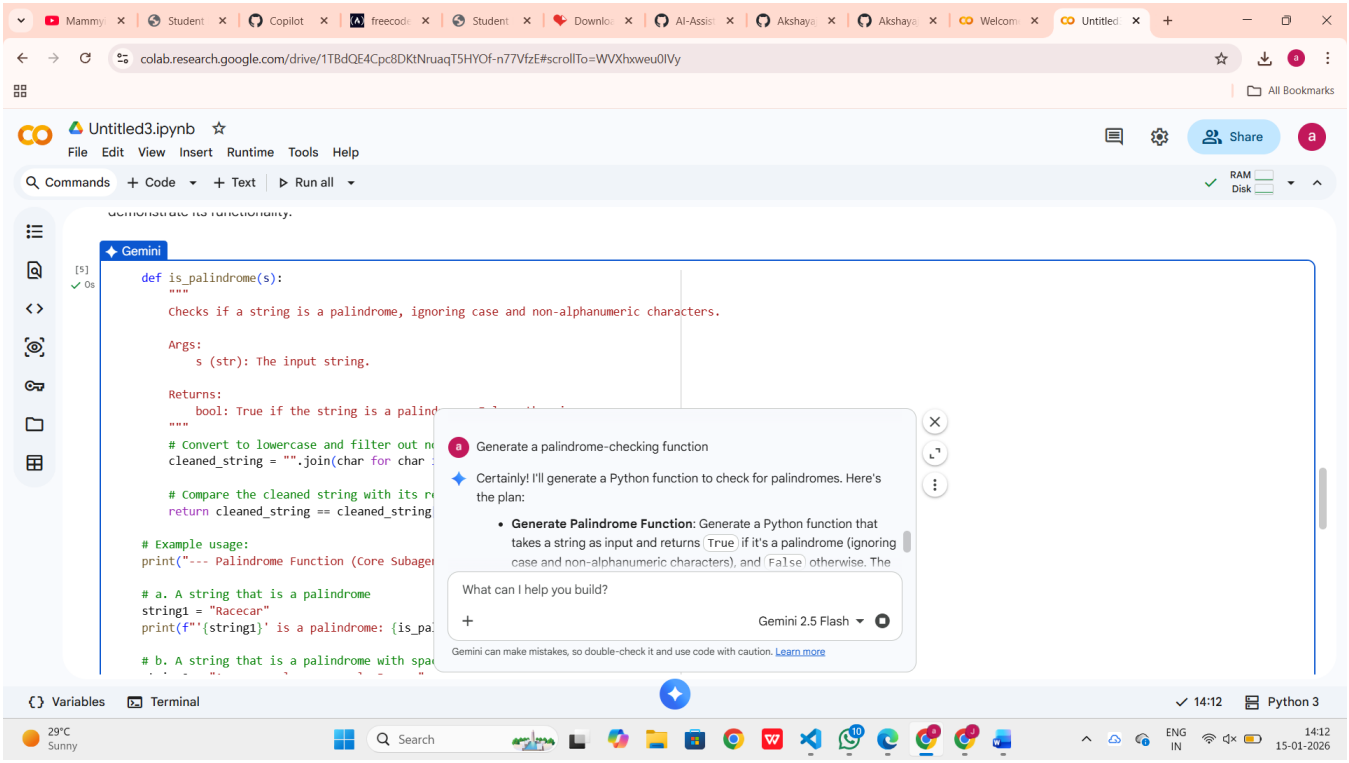
Generate a palindrome-checking function using Gemini and Copilot, then compare the results.

❖ Expected Output:

➢ Side-by-side code comparison

➢ Observations on clarity and structure

CODE:



OUTPUT:

```
+print(is_palindrome("A man, a plan, a canal: Panama")) # Output: True
+print(is_palindrome("race a car"))     # Output: False
+print(is_palindrome("hello"))          # Output: False
+
```

```
True
True
False
False
```

Task 4: Code Explanation Using AI

❖ Scenario:

You are reviewing unfamiliar code written by another developer.

❖ Task:

Ask Gemini to explain a Python function (prime check OR palindrome check) line by line.

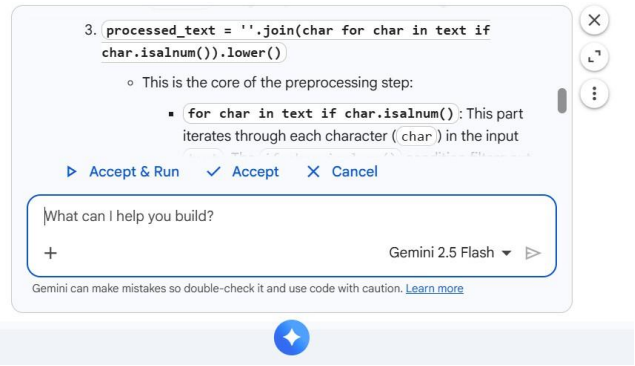
❖ Expected Output:

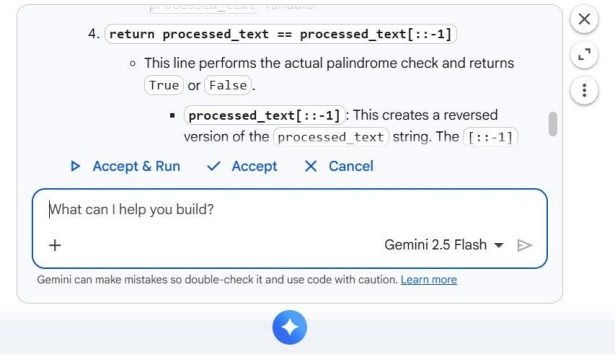
➢ Code snippet

➢ AI explanation

➢ Student comments on understanding

➢ OUPUT





My own experience using **both Gemini and GitHub Copilot**:

While working with **Gemini** in **Google Colab**, I noticed that the explanations were well-structured and easy to follow. It helped me understand the reasoning and flow behind Python programs by breaking them down step by step. This made Gemini particularly helpful for learning, debugging, and analyzing code logic.

On the other hand, **GitHub Copilot** focused more on speed and productivity. By generating code suggestions directly inside the editor, it allowed me to implement solutions quickly and continue coding without interruptions. Copilot felt more effective during active development, especially when working on longer programs.

In summary, Gemini supported deeper conceptual understanding, while GitHub Copilot enhanced coding speed. Using both tools together significantly improved my learning experience as well as overall coding efficiency.