Ai Assisted Coding

Name : Navaneeth

Roll NO : 2303A54056

Lab 9.1: Documentation Generation -Automatic documentation and code comments

Problem 1:

Consider the following Python function:

```
def find_max(numbers):
return max(numbers)
```

Task:

• Write documentation for the function in all three formats:

(a) Docstring

(b) Inline comments

(c) Google-style documentation

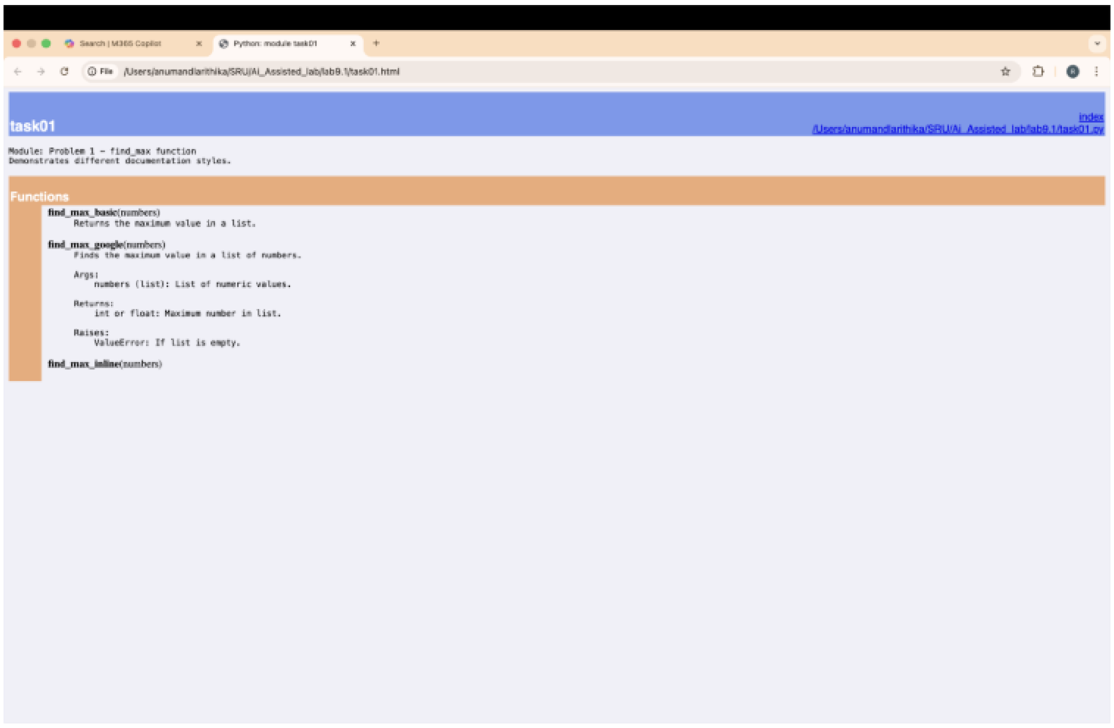
• Critically compare the three approaches. Discuss the advantages, disadvantages, and suitable use cases of each style.

• Recommend which documentation style is most effective for a mathematical utilities library and justify your Answer.

Prompts Used :

```
Python3 –m pydoc task01
Python3 –m pydoc –w task01
Open task01.html
```

Expected output :



Search | M365 Copilot | Python: module task01

/Users/anumandlarithika/SRU/AI_Assisted_lab/lab9.1/task01.html

task01

index
/Users/anumandlarithika/SRU/AI_Assisted_lab/lab9.1/task01.py

```
Module: Problem 1 - find_max function
Demonstrates different documentation styles.
```

Functions

find_max_basic(numbers)
```
    Returns the maximum value in a list.
```

find_max_google(numbers)
```
    Finds the maximum value in a list of numbers.

    Args:
        numbers (list): List of numeric values.

    Returns:
        int or float: Maximum number in list.

    Raises:
        ValueError: If list is empty.
```

find_max_inline(numbers)

Problem 2:

Consider the following Python function:

def login(user, password, credentials):

return credentials.get(user) == password

Task:

1. Write documentation in all three formats.
2. Critically compare the approaches.
3. Recommend which style would be most helpful for new developers onboarding a project, and justify your choice.

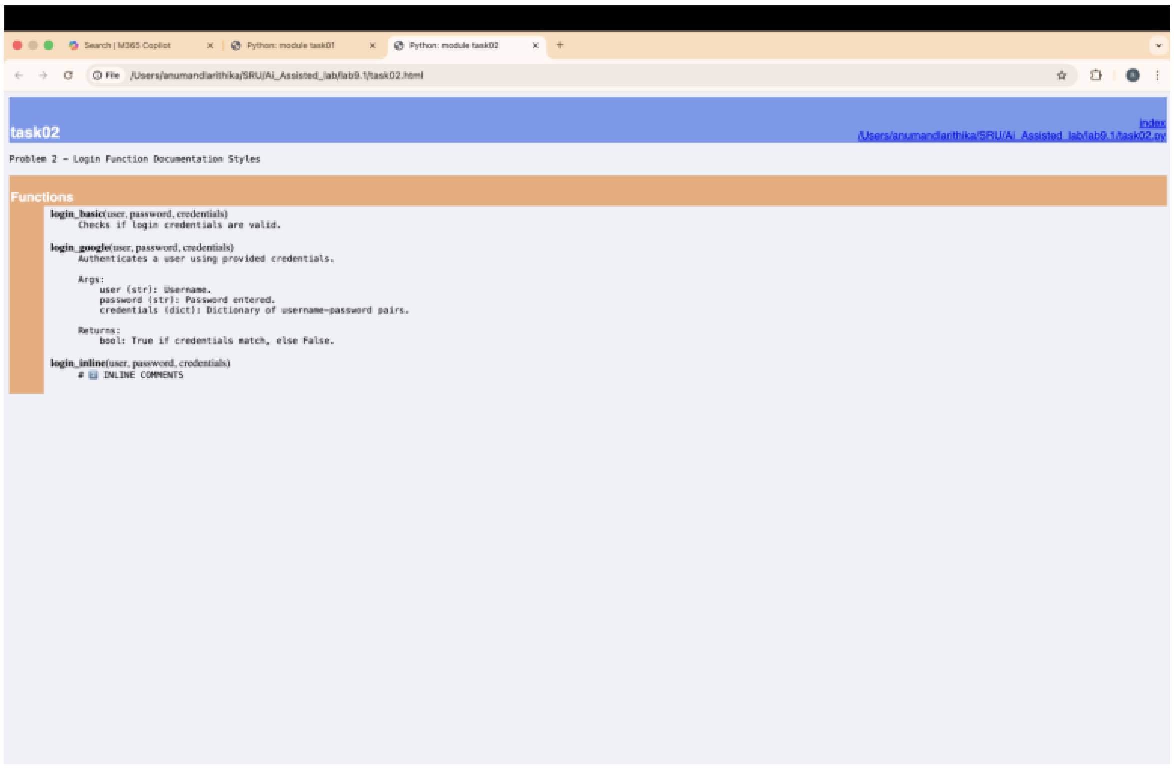
Prompts Used :

Python3 –m pydoc task02

Python –m pydoc –w task02

Open task02.html

Expected Output :


Search | M365 Copilot
Python: module task01
Python: module task02
File /Users/anumandlarithika/SRU/Ai_Assisted_lab/lab9.1/task02.html

```
task02                                                    index
                /Users/anumandlarithika/SRU/Ai_Assisted_lab/lab9.1/task02.py

Problem 2 – Login Function Documentation Styles

Functions
    login_basic(user, password, credentials)
        Checks if login credentials are valid.

    login_google(user, password, credentials)
        Authenticates a user using provided credentials.

        Args:
            user (str): Username.
            password (str): Password entered.
            credentials (dict): Dictionary of username-password pairs.

        Returns:
            bool: True if credentials match, else False.

    login_inline(user, password, credentials)
        # 🔹 INLINE COMMENTS
```

Problem 3: Calculator (Automatic Documentation Generation)

Task: Design a Python module named calculator.py and demonstrate automatic documentation generation.

Instructions:

1. Create a Python module calculator.py that includes the following functions, each written with appropriate docstrings:

o add(a, b) – returns the sum of two numbers

o subtract(a, b) – returns the difference of two numbers

o multiply(a, b) – returns the product of two numbers

o divide(a, b) – returns the quotient of two numbers

2. Display the module documentation in the terminal using Python’s documentation tools.

3. Generate and export the module documentation in HTML format using the pydoc utility, and open the generated HTML

file in a web browser to verify the output.

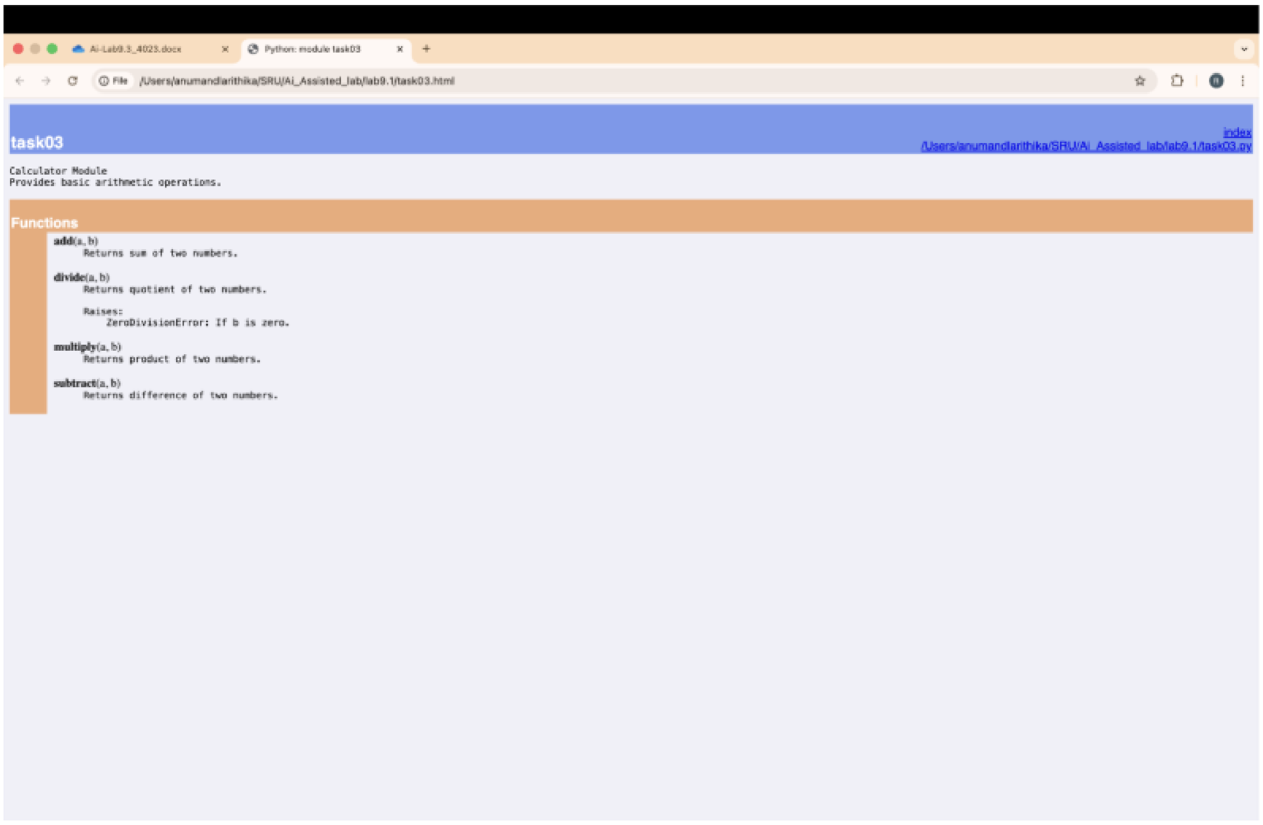
Prompts Used :

Python3 –m pydoc task03

Python –m pydoc –w task03

Open task03.html

Expected Output :



Ai-Lab9.3_4023.docx
Python: module task03
File /Users/anumandlarithika/SRU/Ai_Assisted_lab/lab9.1/task03.html

task03

index
/Users/anumandlarithika/SRU/AI_Assisted_lab/lab9.1/task03.py

```
Calculator Module
Provides basic arithmetic operations.
```

Functions

```
add(a, b)
    Returns sum of two numbers.

divide(a, b)
    Returns quotient of two numbers.

    Raises:
        ZeroDivisionError: If b is zero.

multiply(a, b)
    Returns product of two numbers.

subtract(a, b)
    Returns difference of two numbers.
```

Problem 4: Conversion Utilities Module

Task:

1. Write a module named conversion.py with functions:

o decimal_to_binary(n)

o binary_to_decimal(b)

o decimal_to_hexadecimal(n)

2. Use Copilot for auto-generating docstrings.

3. Generate documentation in the terminal.

4. Export the documentation in HTML format and open it in a

Browser.

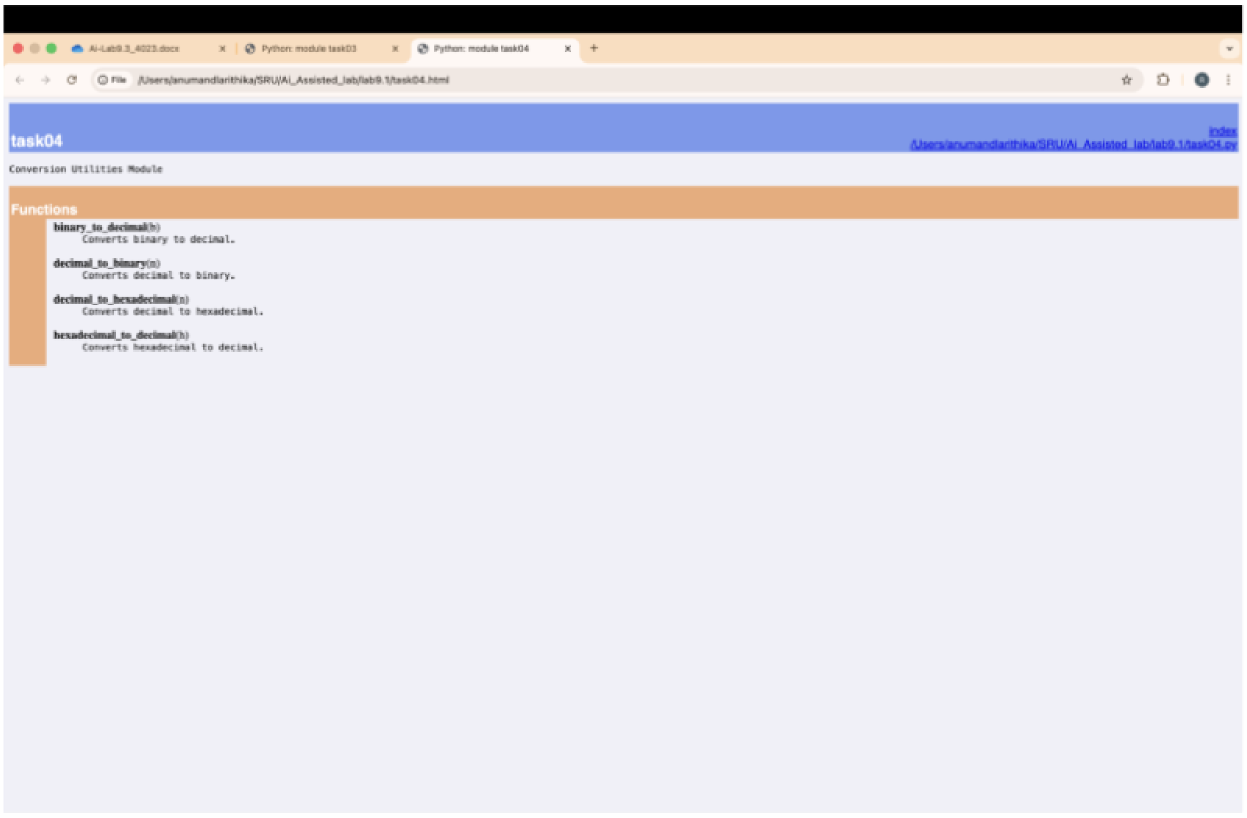
Prompts Used :

Python3 –m pydoc task04

Python –m pydoc –w task04

Open task04.html

Expected Output :



/Users/anumandlarithika/SRU/Ai_Assisted_lab/lab9.1/task04.html

task04

index
/Users/anumandlarithika/SRU/Ai_Assisted_lab/lab9.1/task04.py

```
Conversion Utilities Module
```

Functions

binary_to_decimal(b)
```
    Converts binary to decimal.
```
decimal_to_binary(n)
```
    Converts decimal to binary.
```
decimal_to_hexadecimal(n)
```
    Converts decimal to hexadecimal.
```
hexadecimal_to_decimal(h)
```
    Converts hexadecimal to decimal.
```

Problem 5 – Course Management Module

Task:

1. Create a module course.py with functions:

o add_course(course_id, name, credits)

o remove_course(course_id)

o get_course(course_id)

2. Add docstrings with Copilot.

3. Generate documentation in the terminal.

4. Export the documentation in HTML format and open it in a

Browser.

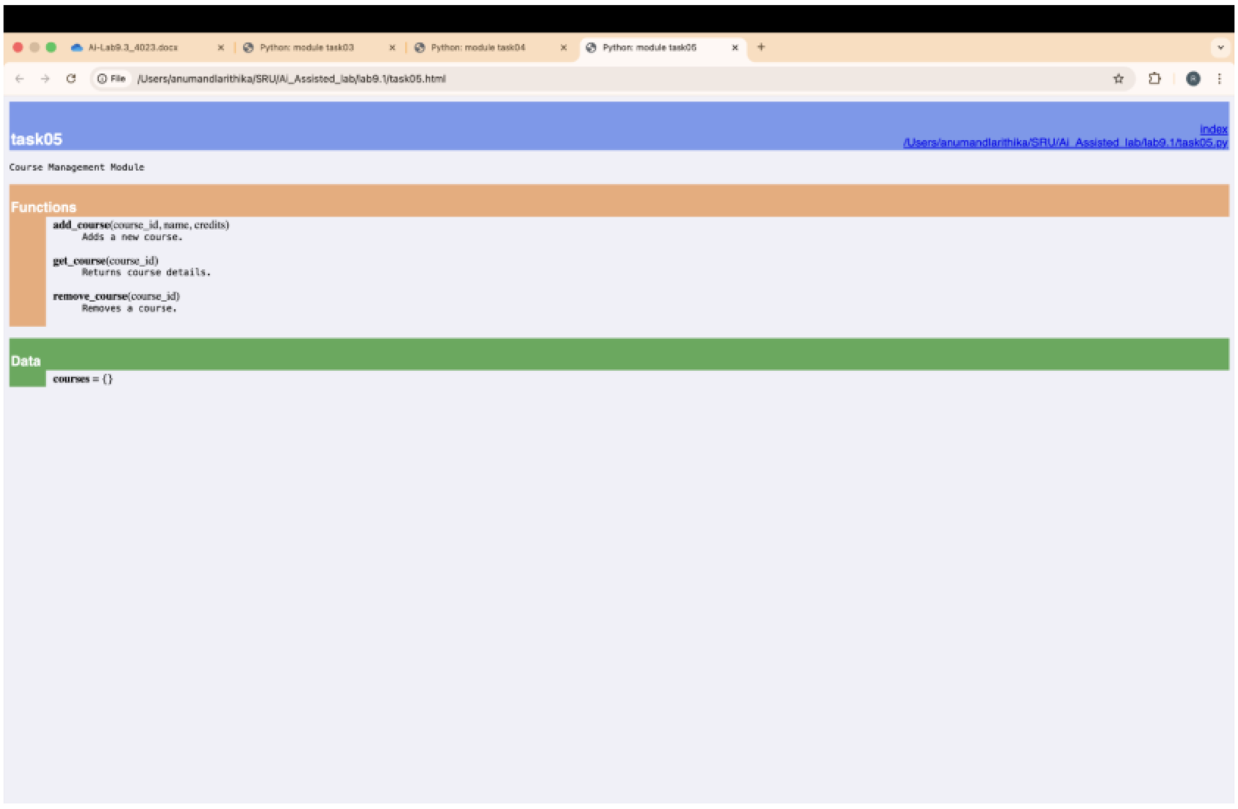
Prompts Used :

Python3 –m pydoc task05

Python –m pydoc –w task05

Open task05.html

Expected Output :



```
AI-Lab9.3_4023.docx | Python: module task03 | Python: module task04 | Python: module task05
File  /Users/anumandlarithika/SRU/AI_Assisted_lab/lab9.1/task05.html

task05                                                                     index
                               /Users/anumandlarithika/SRU/AI_Assisted_lab/lab9.1/task05.py
Course Management Module

Functions
    add_course(course_id, name, credits)
        Adds a new course.

    get_course(course_id)
        Returns course details.

    remove_course(course_id)
        Removes a course.

Data
    courses = {}
```