AI Assisted Coding Assignment- 9.5

RUDROJU RUPA SRI

2303A51918

BATCH-30

Lab Experiment: Documentation Generation -Automatic documentation and code comments
Lab Objectives

1. To understand automatic documentation generation.
2. To generate code comments and docstrings using AI tools.
3. To learn the importance of documentation in software development.

Lab Outcomes

1. Students will be able to generate documentation automatically for code.
2. Students will be able to add clear comments and docstrings to programs.
3. Students will be able to improve code readability and maintainability using documentation.

Problem 1: String Utilities Function
Consider the following Python function:
def reverse_string(text):
return text[::-1]
Task:

1. Write documentation in:
   o (a) Docstring
   o (b) Inline comments
   o (c) Google-style documentation
2. Compare the three documentation styles.
3. Recommend the most suitable style for a utility-based string library.

Problem 2: Password Strength Checker
Consider the function:
def check_strength(password):

```
return len(password) >= 8
```

Task:

1. Document the function using docstring, inline comments, and Google style.
2. Compare documentation styles for security-related code.
3. Recommend the most appropriate style.

Problem 3: Math Utilities Module
Task:

1. Create a module math_utils.py with functions:
   o square(n)
   o cube(n)
   o factorial(n)
2. Generate docstrings automatically using AI tools.
3. Export documentation as an HTML file.

Problem 4: Attendance Management Module
Task:

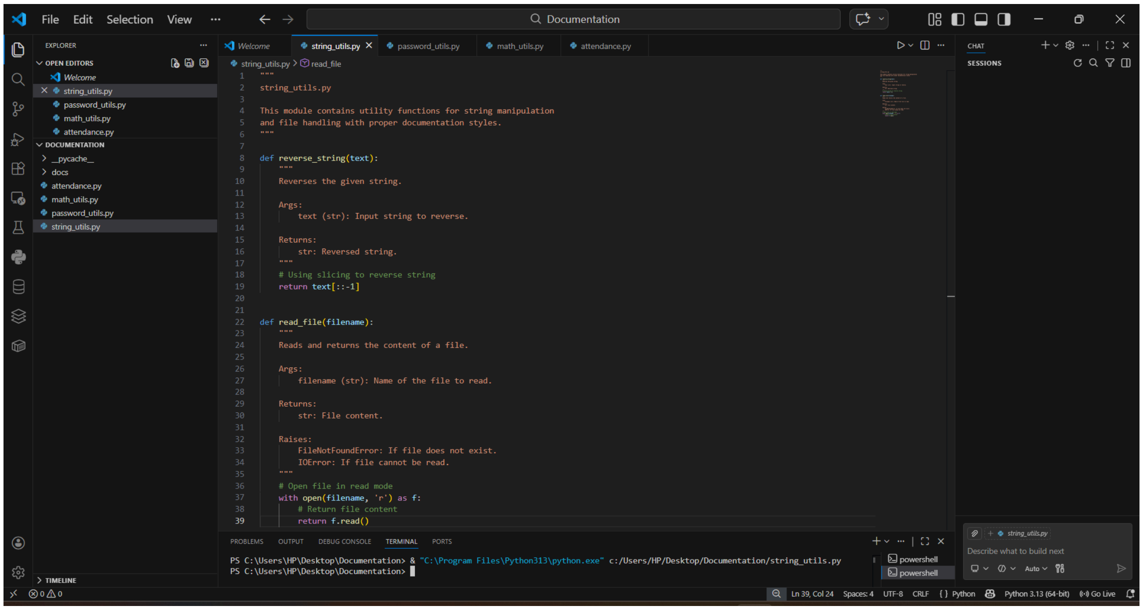
1. Create a module attendance.py with functions:
   o mark_present(student)
   o mark_absent(student)
   o get_attendance(student)
2. Add proper docstrings.
3. Generate and view documentation in terminal and browse

Problem 5: File Handling Function
Consider the function:

```
def read_file(filename):
with open(filename, 'r') as f:
return f.read()
```

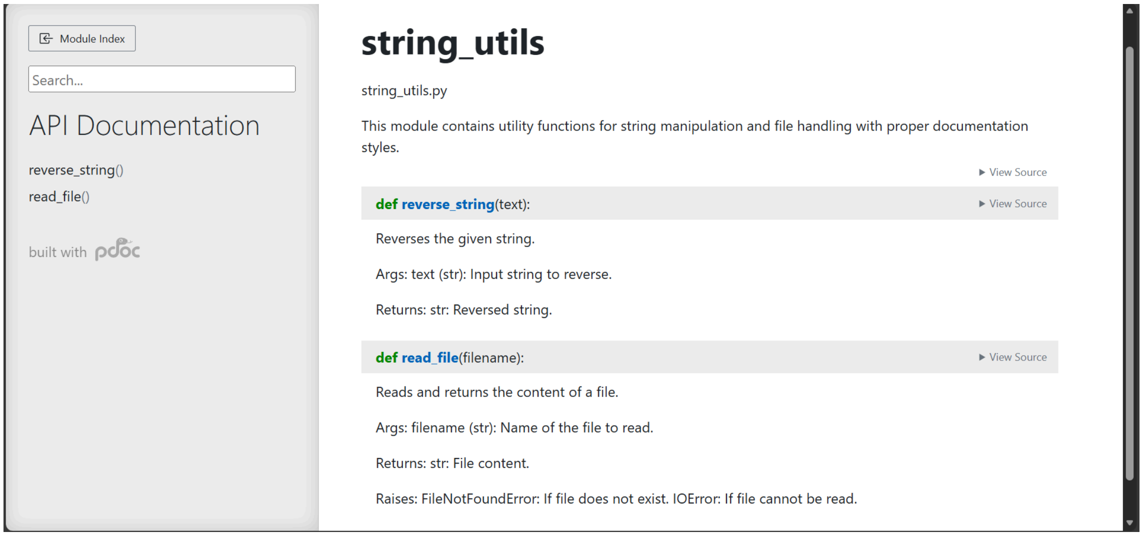
Task:

1. Write documentation using all three formats.
2. Identify which style best explains exception handling.
3. Justify your recommendation.



```python
"""
string_utils.py

This module contains utility functions for string manipulation
and file handling with proper documentation styles.
"""


def reverse_string(text):
    """
    Reverses the given string.

    Args:
        text (str): Input string to reverse.

    Returns:
        str: Reversed string.
    """
    # Using slicing to reverse string
    return text[::-1]


def read_file(filename):
    """
    Reads and returns the content of a file.

    Args:
        filename (str): Name of the file to read.

    Returns:
        str: File content.

    Raises:
        FileNotFoundError: If file does not exist.
        IOError: If file cannot be read.
    """
    # Open file in read mode
    with open(filename, 'r') as f:
        # Return file content
        return f.read()
```

```
PS C:\Users\HP\Desktop\Documentation> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/Documentation/string_utils.py
PS C:\Users\HP\Desktop\Documentation> 
```



Module Index

Search...

API Documentation

reverse_string()

read_file()

built with pdoc

string_utils

string_utils.py

This module contains utility functions for string manipulation and file handling with proper documentation styles.

▶ View Source

def reverse_string(text): ▶ View Source

Reverses the given string.

Args: text (str): Input string to reverse.

Returns: str: Reversed string.

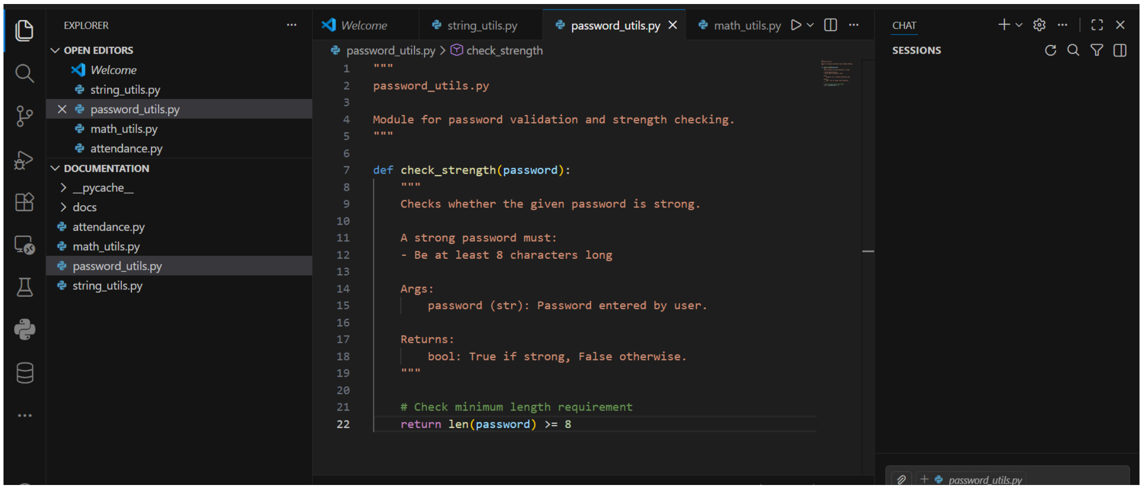
def read_file(filename): ▶ View Source

Reads and returns the content of a file.

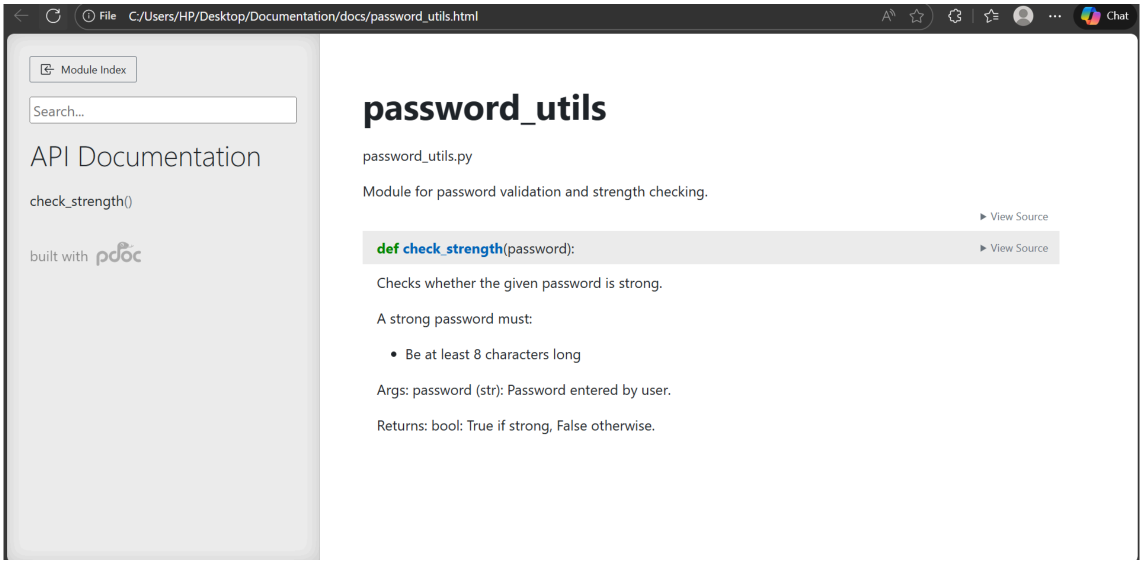
Args: filename (str): Name of the file to read.

Returns: str: File content.

Raises: FileNotFoundError: If file does not exist. IOError: If file cannot be read.



```python
"""
password_utils.py

Module for password validation and strength checking.
"""

def check_strength(password):
    """
    Checks whether the given password is strong.

    A strong password must:
    - Be at least 8 characters long

    Args:
        password (str): Password entered by user.

    Returns:
        bool: True if strong, False otherwise.
    """

    # Check minimum length requirement
    return len(password) >= 8
```



API Documentation

check_strength()

password_utils

password_utils.py

Module for password validation and strength checking.

def check_strength(password):

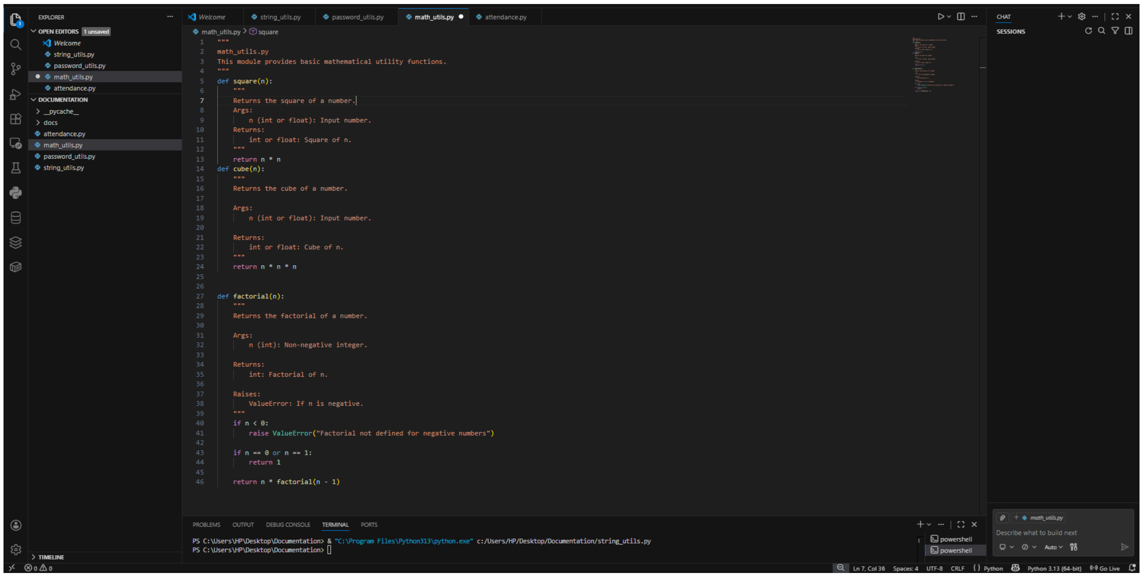
Checks whether the given password is strong.

A strong password must:

- Be at least 8 characters long

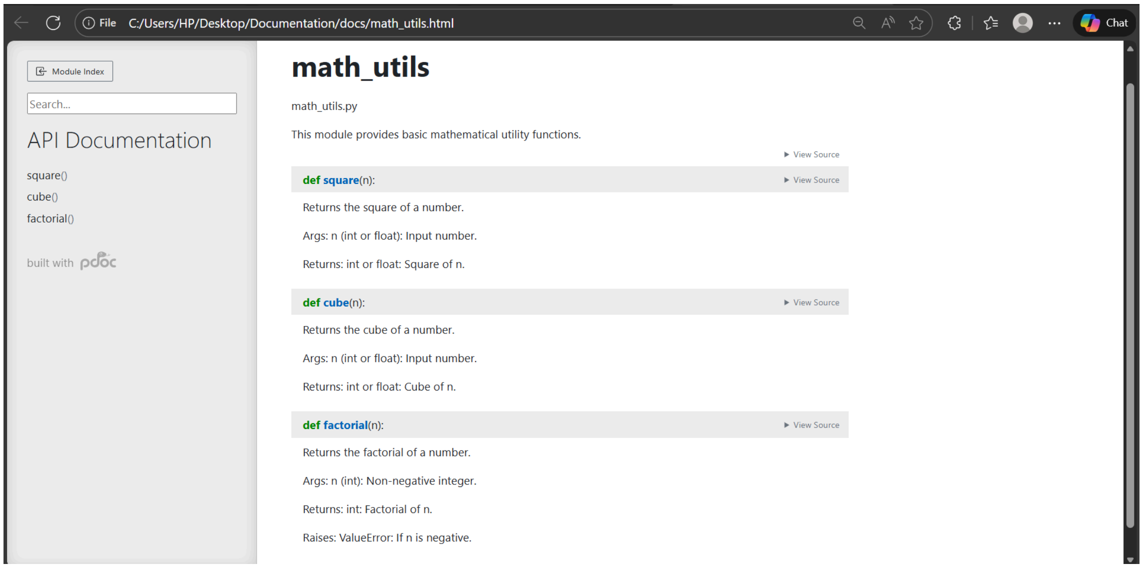
Args: password (str): Password entered by user.

Returns: bool: True if strong, False otherwise.



```python
"""
math_utils.py
This module provides basic mathematical utility functions.
"""
def square(n):
    """
    Returns the square of a number.
    Args:
        n (int or float): Input number.
    Returns:
        int or float: Square of n.
    """
    return n * n
def cube(n):
    """
    Returns the cube of a number.

    Args:
        n (int or float): Input number.

    Returns:
        int or float: Cube of n.
    """
    return n * n * n


def factorial(n):
    """
    Returns the factorial of a number.

    Args:
        n (int): Non-negative integer.

    Returns:
        int: Factorial of n.

    Raises:
        ValueError: If n is negative.
    """
    if n < 0:
        raise ValueError("Factorial not defined for negative numbers")

    if n == 0 or n == 1:
        return 1

    return n * factorial(n - 1)
```

```
PS C:\Users\HP\Desktop\Documentation> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/Documentation/string_utils.py
PS C:\Users\HP\Desktop\Documentation> 
```



math_utils

math_utils.py

This module provides basic mathematical utility functions.

def square(n):

Returns the square of a number.

Args: n (int or float): Input number.

Returns: int or float: Square of n.

def cube(n):

Returns the cube of a number.

Args: n (int or float): Input number.

Returns: int or float: Cube of n.

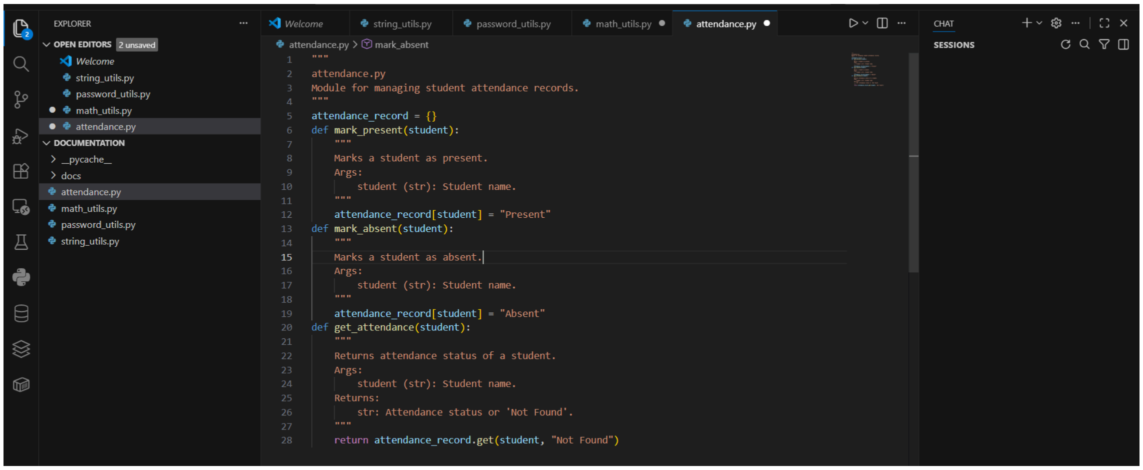
def factorial(n):

Returns the factorial of a number.

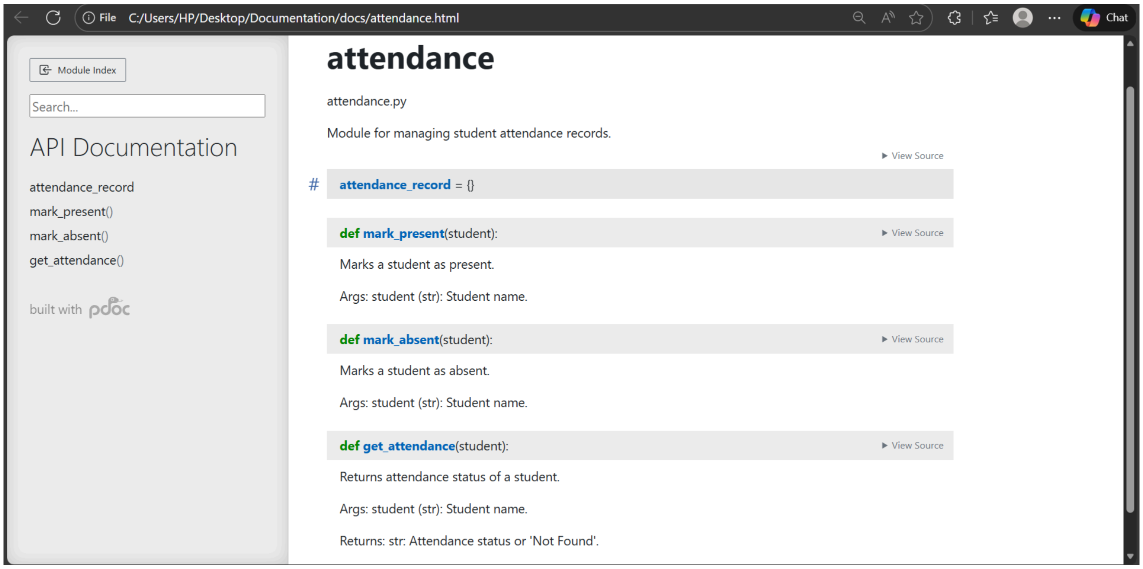
Args: n (int): Non-negative integer.

Returns: int: Factorial of n.

Raises: ValueError: If n is negative.



```python
"""
attendance.py
Module for managing student attendance records.
"""
attendance_record = {}
def mark_present(student):
    """
    Marks a student as present.
    Args:
        student (str): Student name.
    """
    attendance_record[student] = "Present"
def mark_absent(student):
    """
    Marks a student as absent.
    Args:
        student (str): Student name.
    """
    attendance_record[student] = "Absent"
def get_attendance(student):
    """
    Returns attendance status of a student.
    Args:
        student (str): Student name.
    Returns:
        str: Attendance status or 'Not Found'.
    """
    return attendance_record.get(student, "Not Found")
```



attendance

attendance.py

Module for managing student attendance records.

attendance_record = {}

def mark_present(student):

Marks a student as present.

Args: student (str): Student name.

def mark_absent(student):

Marks a student as absent.

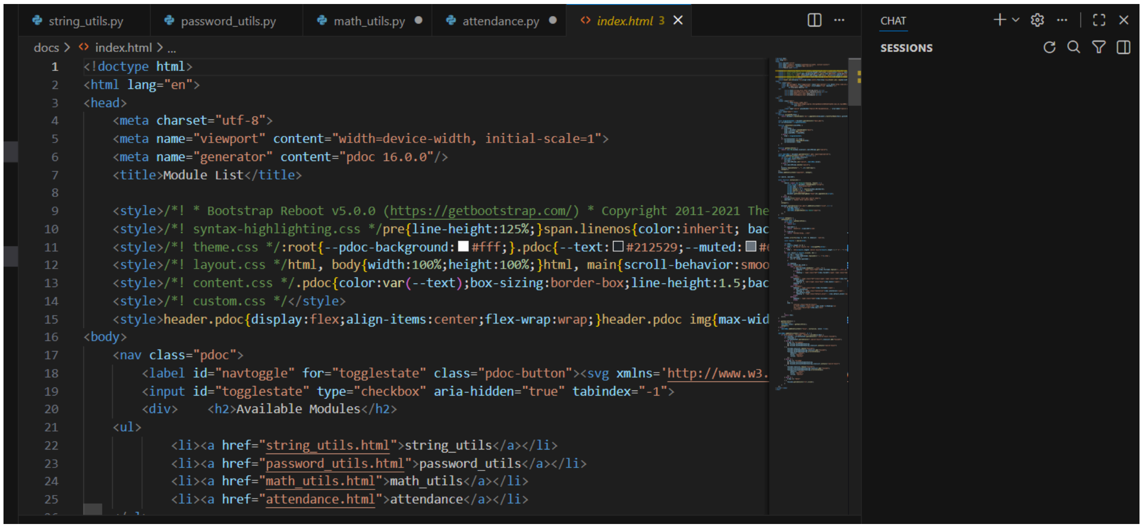
Args: student (str): Student name.

def get_attendance(student):

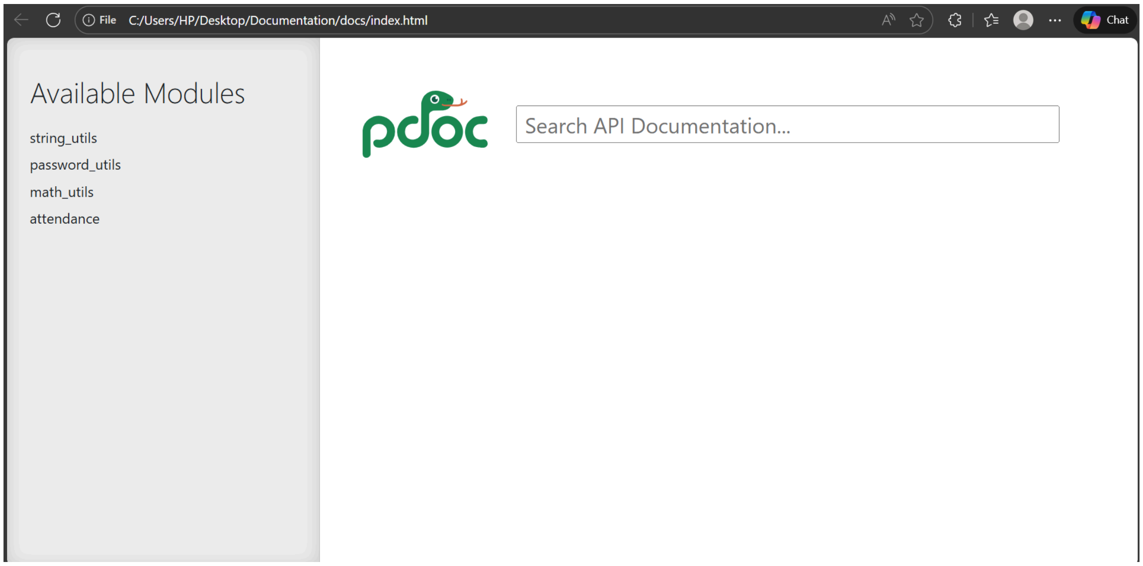
Returns attendance status of a student.

Args: student (str): Student name.

Returns: str: Attendance status or 'Not Found'.



```html
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="generator" content="pdoc 16.0.0"/>
    <title>Module List</title>

    <style>/*! * Bootstrap Reboot v5.0.0 (https://getbootstrap.com/) * Copyright 2011-2021 The
    <style>/*! syntax-highlighting.css */pre{line-height:125%;}span.linenos{color:inherit; bac
    <style>/*! theme.css */:root{--pdoc-background:#fff;}.pdoc{--text:#212529;--muted:#(
    <style>/*! layout.css */html, body{width:100%;height:100%;}html, main{scroll-behavior:smoo
    <style>/*! content.css */.pdoc{color:var(--text);box-sizing:border-box;line-height:1.5;bac
    <style>/*! custom.css */</style>
    <style>header.pdoc{display:flex;align-items:center;flex-wrap:wrap;}header.pdoc img{max-wid
<body>
    <nav class="pdoc">
        <label id="navtoggle" for="togglestate" class="pdoc-button"><svg xmlns='http://www.w3.
        <input id="togglestate" type="checkbox" aria-hidden="true" tabindex="-1">
        <div>    <h2>Available Modules</h2>
    <ul>
            <li><a href="string_utils.html">string_utils</a></li>
            <li><a href="password_utils.html">password_utils</a></li>
            <li><a href="math_utils.html">math_utils</a></li>
            <li><a href="attendance.html">attendance</a></li>
```



Available Modules

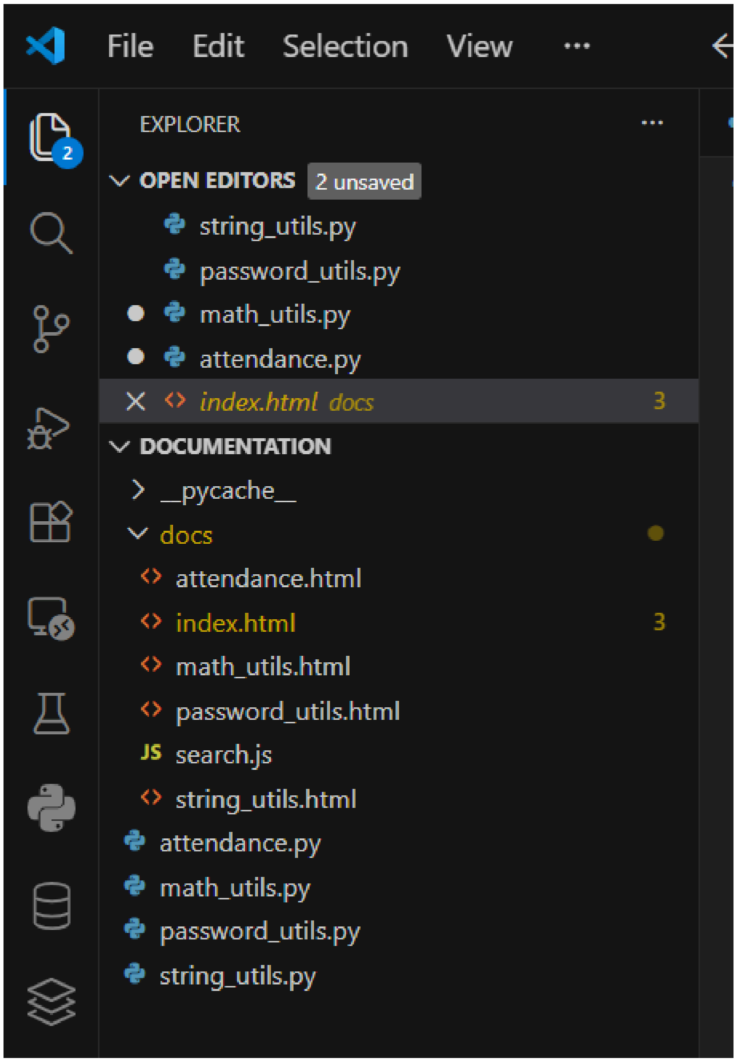
string_utils

password_utils

math_utils

attendance

Search API Documentation...



EXPLORER

OPEN EDITORS 2 unsaved

string_utils.py

password_utils.py

● math_utils.py

● attendance.py

× index.html docs 3

DOCUMENTATION

> __pycache__

docs ●

attendance.html

index.html 3

math_utils.html

password_utils.html

JS search.js

string_utils.html

attendance.py

math_utils.py

password_utils.py

string_utils.py