|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **ProgramName:**B. Tech | | | | **Assignment Type: Lab** | | | **AcademicYear:**2025-2026 | | |
| **CourseCoordinatorName** | | | | Venkataramana Veeramsetty | | | | | |
| **Instructor(s)Name** | | | | |  | | --- | | Dr. V. Venkataramana (Co-ordinator) | | Dr. T. Sampath Kumar | | Dr. Pramoda Patro | | Dr. Brij Kishor Tiwari | | Dr.J.Ravichander | | Dr. Mohammand Ali Shaik | | Dr. Anirodh Kumar | | Mr. S.Naresh Kumar | | Dr. RAJESH VELPULA | | Mr. Kundhan Kumar | | Ms. Ch.Rajitha | | Mr. M Prakash | | Mr. B.Raju | | Intern 1 (Dharma teja) | | Intern 2 (Sai Prasad) | | Intern 3 (Sowmya) | | NS\_2 ( Mounika) | | | | | | |
| **CourseCode** | | | 24CS002PC215 | **CourseTitle** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/I | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week4 - Wednesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:9.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 8: Documentation Generation: Automatic documentation and code comments  **Lab Objectives:**   * To understand the importance of documentation and code comments in software development. * To explore how AI-assisted coding tools can generate meaningful documentation and inline comments. * To practice generating function-level and module-level docstrings automatically. * To evaluate the quality, accuracy, and limitations of AI-generated documentation. * To develop a small automated tool for documentation generation in Python..     **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Apply AI-assisted coding tools to generate docstrings and inline comments for Python code. * Critically analyze AI-generated documentation for correctness, completeness, and readability. * Create structured documentation (function-level, module-level) following standard formats. * Design and implement a mini documentation generator tool to automate code commenting and docstring creation.   **Task Description#1 Basic Docstring Generation**   * Write python function to return sum of even and odd numbers in the given list. * Incorporate manual **docstring** in code with Google Style * Use an AI-assisted tool (e.g., Copilot, Cursor AI) to generate a docstring describing the function. * Compare the AI-generated docstring with your manually written one.   **USER CODE:**    **USER DOC STRING:**  **def sum\_even\_odd(numbers):**  **"""**  **This module provides a function to calculate the sum of even and odd numbers in a given list.**  **Args: numbers (list of int): List of integers to process.**  **Returns:**  **sum\_even\_odd(numbers): Returns a tuple containing the sum of even and odd**  **"""**    **# Comparison:**  **# Comparison of the code and the docstring provided:**  **# 1. The code defines a function sum\_even\_odd(numbers) that takes a list of integers and returns a tuple (sum\_even, sum\_odd).**  **#    - sum\_even: sum of all even numbers in the list.**  **#    - sum\_odd: sum of all odd numbers in the list.**  **# 2. The code includes a main block that demonstrates usage:**  **#    - It creates a list nums = [1, 2, 3, 4, 5, 6]**  **#    - Calls the function and prints the results.**  **# 3. The docstring at the end describes the module:**  **#    - States that the module provides a function to calculate the sum of even and odd numbers.**  **#    - Lists the function and its purpose.**  **# Conclusion:**  **# - The code and the docstring are consistent.**  **# - Both describe and implement a function that returns the sum of even and odd numbers from a list.**  **# - The docstring accurately reflects the functionality provided by the code.**  **Expected Outcome#1:** Students understand how AI can produce function-level documentation.  **Task Description#2 Automatic Inline Comments**   * Write python program for **sru\_student** class with attributes like name, roll no., hostel\_status and **fee\_update** method and **display\_details** method. * Write comments manually for each line/code block * Ask an AI tool to add inline comments explaining each line/step. * Compare the AI-generated comments with your manually written one.   **USER CODE:**    **USER DOC STRING:**  **def fee\_update(self, status):**  **"""**  **Updates the fee payment status.**  **Args:**  **status (bool): True if fee is paid, False otherwise.**  **"""**  **self.fee\_paid = status**  **def display\_details(self):**  **"""**  **Displays the student's details.**  **"""**  **print(f"Name: {self.name}")**  **print(f"Roll No.: {self.roll\_no}")**  **print(f"Hostel Status: {self.hostel\_status}")**  **print(f"Fee Paid: {'Yes' if self.fee\_paid else 'No'}")**    **Expected Output#2:** Students critically analyze AI-generated code comments.    **COMPARISION:**  **# First Implementation:**  **"""**  **A class to represent an SRU student.**  **Attributes:**  **name (str): Name of the student.**  **roll\_no (str): Roll number of the student.**  **hostel\_status (bool): True if staying in hostel, False otherwise.**  **fee\_paid (bool): True if fee is paid, False otherwise.**  **"""**  **# Second Implementation:**  **# No class-level docstring is present.**  **# \_\_init\_\_ method docstrings:**  **# First Implementation:**  **"""**  **Initialize the sru\_student object.**  **Args:**  **name (str): Student's name.**  **roll\_no (str): Student's roll number.**  **hostel\_status (bool): Hostel status.**  **"""**  **# Second Implementation:**  **# No docstring for \_\_init\_\_ method.**  **# fee\_update method docstrings:**  **# First Implementation:**  **"""**  **Update the fee payment status.**  **Args:**  **status (bool): True if fee is paid, False otherwise.**  **"""**  **# Second Implementation:**  **"""**  **Updates the fee payment status.**  **Args:**  **status (bool): True if fee is paid, False otherwise.**  **"""**  **# display\_details method docstrings:**  **# First Implementation:**  **"""**  **Display the details of the student.**  **"""**  **# Second Implementation:**  **"""**  **Displays the student's details.**  **"""**  **# CONCLUSION:**  **# - The first implementation provides a class-level docstring and docstrings for \_\_init\_\_, fee\_update, and display\_details methods.**  **# - The second implementation only provides docstrings for fee\_update and display\_details methods, and omits the class-level and \_\_init\_\_ docstrings.**  **# - The first implementation's docstrings are more descriptive and follow a structured format.**  **Task Description#3**   * Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide). * Incorporate manual **docstring** in code with NumPy Style * Use AI assistance to generate a module-level docstring + individual function docstrings. * Compare the AI-generated docstring with your manually written one.   **Expected Output#3:** Students learn structured documentation for multi-function scripts  **USER CODE:**    **USER DOC STRING:**  **"""**  **This script provides basic calculator functions: add, subtract, multiply, and divide.**  **Functions:**  **add(a, b): Returns the sum of a and b.**  **subtract(a, b): Returns the difference of a and b.**  **multiply(a, b): Returns the product of a and b.**  **divide(a, b): Returns the division of a by b. Raises ValueError if b is zero.**  **"""**        **COMPARISION:**  **"""**  **This script provides basic calculator functions: add, subtract, multiply, and divide.**  **Functions:**  **add(a, b): Returns the sum of a and b.**  **subtract(a, b): Returns the difference of a and b.**  **multiply(a, b): Returns the product of a and b.**  **divide(a, b): Returns the division of a by b. Raises ValueError if b is zero.**  **"""**  **Push documentation whole workspace as .md file in GitHub Repository**  **Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots** | | | | | | Week4 - Wednesday |  |