|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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.   **Expected Outcome#1:** Students understand how AI can produce function-level documentation.  **Prompt#1:**   * Write a python code that it has a function which returns sum of even and odd numbers in the given list using dynamic input.   **Code#1:**    **Observation#1:**   * This Python code defines a function sum\_even\_odd that takes a list of numbers, iterates through each number, and accumulates the sum of even numbers in even\_sum and the sum of odd numbers in odd\_sum using a simple conditional check (number % 2 == 0) within a for loop before returning both sums as a tuple; the rest of the code prompts the user for a space-separated string of numbers, converts it into a list of integers, calls the function with this list, and then prints the calculated total sum for both even and odd numbers.   **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.   **Expected Output#2:** Students critically analyze AI-generated code comments.  **Prompt#2:**   * Write a python function for oops such that sru\_students as class and attributes are name,roll no.,hostel\_status and fee update method and display-details method using dynamic input dont use comments**.**   **Code#2:**      **Observation#2:**   * This Python code defines an SRUStudent class to manage student details including name, roll number, hostel status, hostel fee, and college fee, initializing these attributes when a student object is created; it also includes an update\_fees method to modify existing fee amounts and a display\_details method to print all the student's information; the main part of the script dynamically prompts the user for the number of students and then iteratively collects each student's details, including conditional input for hostel fees based on status and input for college fees, creates SRUStudent objects, stores them in a list, displays their initial details, and finally provides an option to dynamically update and display the fees for each student.   **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    **Prompt#3:**   * Write a python code that has 3-4 functions (e.g., calculator: add, subtract, multiply, divide) using dynamic input.   **Code#3:**      **Observation#3:**   * This Python code implements a simple calculator by defining functions for addition, subtraction, multiplication, and division, including basic error handling for division by zero; the main part of the script presents a menu of operations to the user and enters a loop that repeatedly prompts for an operation choice and two numbers, performs the selected calculation using the defined functions, and prints the result, continuing until the user enters 'quit', while also including error handling for invalid numerical input.   **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 |  |