|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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.   **VS CODE CODE WITHOUT DOCSTRINGS:**    **OUTPUT:**    **Expected Outcome#1:** Students understand how AI can produce function-level documentation.  **DOCSTRINGS:**    **"""**  This function sum\_even\_odd takes a list of numbers.  Args : It takes input as number with integer datatype.  Returns : It returns the sum of even numbers and the sum of odd numbers.  """  **CODE WITH DOCSTRINGS AND INLINE COMMENTS:**    **ANALYSING DOCSTRINGS:**  My docstring is brief and covers the basic purpose, input, and output of the function.  The docstring github generated is more detailed. It includes:   * A short description of what the function does. * Explicit mention of the argument type (list of int). * Clear explanation of the return value as a tuple with both sums.   My docstring is concise and functional, suitable for quick reference. The generated docstring provides more context, making it helpful for users unfamiliar with the function. It improves clarity by specifying types and the structure of the return value. Overall, a detailed docstring enhances code readability and maintainability, especially in collaborative projects.    **EXTERNAL 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.   **VS CODE CODE WITHOUT DOCSTRINGS:**    **OUTPUT:**    **Expected Output#2:** Students critically analyze AI-generated code comments.  **DOCSTRINGS:**  **”””**  -def \_init\_ function initiates the function from class of SRU\_STUDENT  with the arguments of name,rollno,hostel\_status  -def fee\_update function updates fee and it consist of self and amount entities.  -def display\_details function displays student as output as we filled the above  ”””  **CODE WITH DOCSTRINGS AND INLINE COMMENTS:**    **ANALYSING DOCSTRINGS:**  My docstring is a brief summary of each method’s purpose and arguments. The docstrings in the code are more formal, specifying argument types, return values, and describing the method’s functionality in detail. Overall, the code’s docstrings provide clearer guidance for users and developers, especially for understanding input types and expected behavior.  **EXTERNAL DOCUMENTAION:**    **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.   **VS CODE CODE WITHOUT DOCSTRINGS AND OUTPUT:**    **Expected Output#3:** Students learn structured documentation for multi-function scripts  **DOCSTRINGS:**  **”””**  This program is a simple calculator. It can perform four basic operations: addition, subtraction, multiplication, and division. The add function takes two numbers and gives back their sum. The subtract function finds the difference between two numbers. The multiply function returns the product of two numbers. The divide function divides one number by another, but it will give an error if the second number is zero.  **”””**  **CODE WITH DOCSTRINGS AND INLINE COMMENTS:**    **ANALYSING DOCSTRINGS:**  Your docstring provides a clear, high-level summary of the calculator program and describes the purpose of each function in simple terms. The code docstrings are more detailed and formal, specifying argument types, return values, and exceptions for each function. While your docstring is user-friendly and easy to understand, the code docstrings are more suitable for developers who need precise information about function usage and behavior. Both styles are useful: yours for general documentation and the code’s for technical reference. Combining both approaches can make the program accessible to both users and developers.  Push documentation whole workspace as .md file in GitHub Repository.  **EXTERNAL DOCUMENTATION:**    **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 |  |