**Name: Akshaya Neerati**

**Enrollment No: 2403A510B8**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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:** Generate a python function to return sum of even and odd numbers in the given list.    **AI generated comments:**    **Observation:**  The Python program defines a function to calculate the sum of even and odd numbers from a given list. It uses a loop to iterate through the list, checks whether each number is even or odd, and accumulates the sums separately. The function then returns both sums, which are displayed as output. The AI-generated comments provide detailed documentation, including function description, arguments, return values, and inline explanations for each step, making the code more readable and easier to understand.    **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:** Generate a python program for sru\_student class with attributes like name, roll no., hostel\_status and fee\_update method and display\_details method.    **AI generated comments:**    **Observation:**  The program defines a class sru\_student with attributes like name, roll number, hostel status, and fee status. The constructor initializes these attributes, and the fee\_update() method allows updating the fee payment status. The display\_details() method prints the student’s details in a structured format. The AI-generated comments provide clear explanations of attributes, methods, parameters, and return values, which make the code more understandable and maintainable. The program successfully demonstrates object-oriented programming concepts like class, object creation, attributes, and methods.  **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:** Write a Python script that defines a simple calculator with 4 functions: add, subtract, multiply, and divide. Use NumPy-style docstrings for each function and example usage.  **Manual:**    **AI generated numpy style:**  def add(a, b):      """      Add two numbers.+      Parameters      ----------      a : float or int          First number.      b : float or int          Second number.      Returns      -------      float or int          The sum of a and b.      Examples      --------      >>> add(2, 3)      5      """      return a + b  def subtract(a, b):      """      Subtract one number from another.      Parameters      ----------      a : float or int          Number to subtract from.      b : float or int          Number to subtract.      Returns      -------      float or int          The result of a - b.      Examples      --------      >>> subtract(5, 2)      3      """      return a - b  def multiply(a, b):      """      Multiply two numbers.      Parameters      ----------      a : float or int          First number.      b : float or int          Second number.      Returns      -------      float or int          The product of a and b.      Examples      --------      >>> multiply(4, 3)      12      """      return a \* b  def divide(a, b):      """      Divide one number by another.      Parameters      ----------      a : float or int          Numerator.      b : float or int          Denominator.      Returns      -------      float          The result of a / b.      Raises      ------      ZeroDivisionError          If b is zero.      Examples      --------      >>> divide(10, 2)      5.0      """      if b == 0:          raise ZeroDivisionError("Cannot divide by zero.")      return a / b  if \_\_name\_\_ == "\_\_main\_\_":      print("Add: ", add(2, 3))      print("Subtract: ", subtract(5, 2))      print("Multiply: ", multiply(4, 3))  print("Divide: ", divide(10, 2))  **Observation:**  The Python script implements a simple calculator using four functions: add, subtract, multiply, and divide. Each function is documented with NumPy-style docstrings, which describe the purpose, parameters, return values, examples, and possible exceptions (for divide). The script demonstrates function modularity, good documentation practices, and error handling for division by zero. The program outputs correct results when executed, making it both user-friendly and maintainable.  **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 |  |