|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 function to return sum of even and odd numbers in the given list 2. Give the code without docstring 3. After writing code manually ask AI to give docstring   **Code:**  **AI generated Docstring:**    **Manual Docstring:**  **“““**  The code calculates the sum of odd and even numbers differently  Takes the input from user  Gives output of even and odd number sum separately  **”””**  **Output:**      **Code Explanation :**  This code takes a space-separated list of numbers from the user, converts them to integers (skipping invalid entries), calculates the sum of the even and odd numbers separately using the sum\_even\_odd function, and then prints the results.  **Observation:**   1. Manual docstring is in simple words and AI generated docstring has a more formal structure 2. AI generated docstring structured with explicit Args and Returns sections, detailing input/output types. 3. Manual docstring is more narrative description, mentioning user input relevant to the script's usage. 4. Finally AI generated docstring is in an structured way with detailed type and manual docstring is easy to understand with simple words.     **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:**   1. Write a python program for **sru\_student** class with attributes like name, roll no., hostel\_status and **fee\_update** method and **display\_details** method. 2. Take input from user. 3. Give comment for each line in code.   **AI generated Code with comments :**      **Code with Manual Comments :**    **Output :**      **Code Explanation :**    This code defines a blueprint for a student (sru\_student class), takes information from the user to create a specific student based on that blueprint, displays the details of that student, and then allows for a simulated fee update based on further user input.    **Observation :**   1. AI comments are clearer and more accurate than most manual ones. 2. AI maintains a uniform style, while manual comments often vary. 3. AI explains both what and why; manual usually just covers what. 4. AI uses professional grammar and phrasing; manual may not. 5. AI comments are more educational and easier to update.   **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 :**     1. Generate a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide).with user input.   **AI-generated code:**        **Manual Code:**      **Comparison:**  **1.AI docstring** explains the function in a **formal, structured way** (with Parameters, Returns, Raises).  2. **Manual docstring** explains it in a **short and direct way** (one-liner focusing only on what the function does).  3. **AI docstrings** are best for documentation, while **Manual docstrings** are best for real projects .  4. **AI docstrings** focus on explaining how inputs/outputs work, while **Manual docstrings** focus on what the function does.  **Code Explanation:**  1.Four functions are defined → add, subtract, multiply, divide.  2.The program asks the user to enter two numbers.  3.The user chooses which operation to perform based on that choice ,the result is printed.  4.If the user enters wrong input the program shows an error message.  **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 |  |