|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **]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) | | | | | | |
| **rCourseCode** | | | 24CS002PC215 | **CourseTitle** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/I | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week5- 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*:Can you give a pyhton function that collects the list from the user and returns the sum of even and odd numbers in the given list that Python function Contains Incorporate manual docstring in code with style.**      **Manual code:**      **Comparision :**    **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.**   **Prompt:Write a Python program that defines an sru\_student class with attributes: name, roll\_no, and hostel\_status. Include two methods: fee\_update() to update fee status, and display\_details() to print student information. Add manual inline comments explaining each line or code block. Then, use an AI tool to generate inline comments for the same code and compare both sets of comments.**        **Expected Output#2: Students critically analyze AI-generated code comments.**  **Manual Code:**      **Comparision of manual and Ai generated code:**  **· **Class Definition** Manual: Clearly states the purpose of the class (# Define a class to represent a student at SRU)AI: No comment at all for the class definition  Manual is more informative**  ****Constructor Method (**\_\_init\_\_**)**Manual: Describes the method as a constructor and explains each attribute's role**  **AI: Only adds brief comments next to each attribute assignment Manual provides context and intent; AI is minimal**  **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.**   **Prompt:Can you give a python function that generates a python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide) and maintain Incorporate manual docstring in code with NumPy Style.**          **Manual :**      **Expected Output#3: Students learn structured documentation for multi-function scripts**  **Comparision of Manual and Ai generated docstring code:**  The manual code is clean, direct, and easier to maintain, while the AI-generated version uses dynamic scripting for automation. Both use NumPy-style docstrings, but manual formatting offers better control and readability. The AI version embeds docstrings in a string block, which can be harder to lint or debug. Execution in the manual version is straightforward; the AI version requires file creation and import logic. Overall, the manual script is ideal for clarity and testing, while the AI version suits templated or programmatic generation.  **Push documentation whole workspace as .md file in GitHub Repository**  **Note: Report should be submitted a word document for** | | | | | | **Week4 - Wednesday** |  |

**all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots**