|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **Academic Year:**2025-2026 | | |
| **Course Coordinator Name** | | | | 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) | | | | | | |
| **Course Code** | | | 24CS002PC215 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/I | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week5 - Monday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | | |
| **AssignmentNumber: 9.1**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | **Lab 9 – Documentation Generation: Automatic Documentation and Code Comments**  **Lab Objectives**   * Inline comments * Docstrings * Auto-documentation tools * AI-assisted summarization   **Task Description #1** (Automatic Code Commenting)  **Scenario:** You have been given a Python function without comments.  def calculate\_discount(price, discount\_rate):  return price - (price \* discount\_rate / 100)   * Use an AI tool (or manually simulate it) to generate line-by-line comments for the function. * Modify the function so that it includes a docstring in Google-style or NumPy-style format. * Compare the auto-generated comments with your manually written version.   **Inline Comments:**      **Doc String:**      **Manually Written Comments:**    **Auto-generated:**   * Usually adds generic comments like “# Function to calculate discount”. * Might miss details like parameters, return type, or examples.   **Manual version:**   * More specific and tailored. * Includes explanation of arguments, return value, and usage examples.   **Task Description #2** (API Documentation Generator)  **Scenario:** A team is building a **Library Management System** with multiple functions.  def add\_book(title, author, year):  # code to add book  pass  def issue\_book(book\_id, user\_id):  # code to issue book  Pass   * Write a Python script that uses docstrings for each function (with input, output, and description). * Use a documentation generator tool (like pdoc, Sphinx, or MkDocs) to automatically create HTML documentation. * Submit both the code and the generated documentation as output.   **Code and Generated Documentation:**        **Task Description #3** (AI-Assisted Code Summarization)  **Scenario:** You are reviewing a colleague’s codebase containing long functions.  def process\_sensor\_data(data):  cleaned = [x for x in data if x is not None]  avg = sum(cleaned)/len(cleaned)  anomalies = [x for x in cleaned if abs(x - avg) > 10]  return {"average": avg, "anomalies": anomalies}   * Generate a summary comment explaining the purpose of the function in 2–3 lines. * Create a flow-style comment (step-by-step explanation). * Write a short paragraph of documentation describing possible use cases of this function in real-world scenarios.   **Summary Comment:**    **Flow-Style Comment:**      **Use in the Real-World:**    **Task Description #4** (Real-Time Project Documentation)  **Scenario:** You are part of a project team that develops a Chatbot Application. The team needs documentation for maintainability.   * Write a README.md file for the chatbot project (include project description, installation steps, usage, and example). * Add inline comments in the chatbot’s main Python script (focus on explaining logic, not trivial code). * Use an AI-assisted tool (or simulate it) to generate a usage guide in plain English from your code comments. * Reflect: How does automated documentation help in real-time projects compared to manual documentation?   **Python Script:**    **README.md File:**    **Output:**    **Adding Inline Comments:**        **Usage Guide:**   * Run the program to start chatting. * Type a message — the bot looks for keywords like "hi", "hello", or "bye". * If it finds a match, it replies with a preset response. * If not, it says it doesn’t understand. * Type "bye" to end the chat.   **Automated vs Manual Documentation:**   * Manual documentation requires deliberate effort and time, often becoming outdated if the code evolves rapidly. * AI-assisted tools can generate and update documentation from code comments in real-time, which helps teams stay in sync and reduces maintenance overhead. * However, manual documentation offers more control and clarity when explaining complex logic or high-level design decisions. * Combining both methods yields the best results in fast-moving projects. | | | | | | Week5 - Monday |  |