|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week5 - Tuesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:10.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 10: Code Review and Quality: Using AI to improve code quality and readability  **Lab Objectives:**   * To understand the importance of code readability, maintainability, and quality. * To explore how AI-assisted coding tools can review code and suggest improvements. * To practice identifying code smells, redundant code, and poor naming conventions. * To apply AI tools for refactoring and improving readability. * To critically evaluate AI feedback and integrate it into real projects   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use AI-assisted tools (e.g., GitHub Copilot, Cursor AI) to review Python code. * Identify and correct syntax issues, code smells, and inefficient logic. * Improve readability by applying consistent formatting, naming, and comments. * Refactor code with AI suggestions while ensuring functionality is preserved. * Apply best practices for writing clean, maintainable, and professional code.   **Task Description#1 AI-Assisted Code Review (Basic Errors)**   * Write python program as shown below. * Use an AI assistant to review and suggest corrections.     **Expected Outcome#1:** Students need to submit corrected code with comments.  PROMPT: suggest corrections and give corrected code with comments      **Task Description#2 Automatic Inline Comments**   * Write the Python code for Fibonacci as shown below and execute. * Ask AI to improve variable names, add comments, and apply PEP8 formatting (cleaned up). * Students evaluate which suggestions improve readability most. one.     **Expected Output#2:** Clean format python code with much readability.  **PROMPT:**  **can you please improve variable names, add comments, and apply PEP8 formatting.**      **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.   **Common Examples of Code Smells**   * Long Function – A single function tries to do too many things. * Duplicate Code – Copy-pasted logic in multiple places. * Poor Naming – Variables or functions with confusing names (x1, foo, data123). * Unused Variables – Declaring variables but never using them. * Magic Numbers – Using unexplained constants (3.14159 instead of PI). * Deep Nesting – Too many if/else levels, making code hard to read. * Large Class – A single class handling too many responsibilities.   **Why Detecting Code Smells is Important**   * Makes code easier to read and maintain. * Reduces chance of bugs in future updates. * Helps in refactoring (improving structure without changing behavior). * Encourages clean coding practices   **Dead Code – Code that is never executed.**  **Expected Output#3:** Students learn structured documentation for multi-function scripts  **PROMPT:**  **generate module-level and individual function docstrings in NumPy style for the following Python script. I've already written manual docstrings, so I'd like to compare them with AI-generated ones. Keep them concise but clear, and follow NumPy documentation standards.**          **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** | | | | | | Week5 |  |