|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week1 - Thursday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | | 24CSBTB01 To 24CSBTB39 | | | |
| **AssignmentNumber:2.4**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 2: Exploring Additional AI Coding Tools – Gemini (Colab) and Cursor AI  **Lab Objectives:**   * To explore and evaluate the functionality of Google Gemini for AI-assisted coding within Google Colab. * To understand and use Cursor AI for code generation, explanation, and refactoring. * To compare outputs and usability between Gemini, GitHub Copilot, and Cursor AI. * To perform code optimization and documentation using AI tools.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Generate Python code using Google Gemini in Google Colab. * Analyze the effectiveness of code explanations and suggestions by Gemini. * Set up and use Cursor AI for AI-powered coding assistance. * Evaluate and refactor code using Cursor AI features. * Compare AI tool behavior and code quality across different platforms.   **Task Description #1**  **•** Open Google Colab and use Google Gemini to generate Python code that performs sorting of a list using both the bubble sort algorithm and Python’s built-in sort() function. Compare the two implementations.  **PROMPT :** Generate Python code that performs sorting of a list using both the bubble sort algorithm and Python’s built-in sort() function.    **Expected Output #1**  **•** Two sorting implementations: Bubble sort (manual logic) and Built-in sort()  **🔍 Observations :**   * **Both sorting methods (Bubble Sort and built-in sort()) return the same final sorted list.** * **The comparison bubble\_sorted == list\_for\_builtin returns True, confirming algorithmic correctness of both methods.**   **Task Description #2**  **•** In Colab, use Google Gemini to generate a Python function that takes a string and returns:  The number of vowels, The number of consonants, The number of digits in the string  PROMPT:  Generate a Python function that takes a string as input and returns:  1. The number of vowels in the string,  2. The number of consonants in the string,  3. The number of digits in the string.  Make sure the function ignores spaces and special characters when counting vowels and consonants. Then provide example usage of the function and print the results.    **Expected Output #2-**  **•** Complete function that Iterates through characters of a string and Counts vowels, consonants, and digits  **Observation :**   1. The function takes a string and checks each character. 2. It counts:    * **Vowels**: a, e, i, o, u (both uppercase and lowercase)    * **Consonants**: any other alphabet letters    * **Digits**: numbers like 0–9 3. It **ignores spaces and special characters** (like @, !, etc.). 4. It uses Python’s built-in functions:    * .isdigit() to find digits    * .isalpha() to find letters 5. The output is the total number of vowels, consonants, and digits in the string.   **Task Description #3**  **•** Install and set up Cursor AI. Use it to generate a Python program that performs file handling:  Create a text file  Write sample text  Read and display the content  PROMPT:  1. Creates a text file named "example.txt"  2. Writes some sample text into the file  3. Reads the content from the file and prints it  4. Uses try-except blocks to handle errors    **Expected Output #3**  **•** Functional code that creates a .txt file, writes content to it, and reads it back.  • Screenshot of Cursor AI interface showing: Prompt used,  Generated code, Output of file operations  **Observation :**  **File Creation**  The code creates a text file named example.txt using Python’s open() function with mode "w" (write).  If the file doesn't exist, it will be created. If it does exist, it will be **overwritten**.  **Task Description #4**  • Ask Google Gemini to generate a Python program that implements a simple calculator using functions (add, subtract, multiply, divide). Then, ask Gemini to explain how the code works.  **Prompt:**  Please write a Python program that implements a simple calculator with four functions: add, subtract, multiply, and divide. Each function should take two numbers as input and return the result. Then create a simple user interface that lets the user choose the operation and enter two numbers.  After you generate the code, please explain how the program works, including what each function does and how the user input is handled.    **Expected Output #4**  **•** Complete calculator code with user input and operation selection.  • Line-by-line explanation or markdown-style explanation provided by Gemini.  • Screenshot of both the code and explanation in Colab.  **Observation:**   * The program has four functions: add, subtract, multiply, and divide. * It asks the user to pick an operation and enter two numbers. * It checks if the user’s choice is valid. * It handles division by zero without crashing. * Then, it shows the result of the calculation. * The code is easy to read and organized.   **Task Description #5** • Use Cursor AI to create a Python program that checks if a given year is a leap year or not. Try different prompt styles and see how Cursor modifies its code suggestions.  **Prompt :**  **Write a Python function to check if a year is a leap year.**  **Make a Python program that asks for a year and tells if it is a leap year or not.**  **Help me write Python code to check if a year is leap or not and print the result**    **Expected Output #5** • A functional program to check leap year with sample input/output • At least two versions of the code (from different prompts) • A short comparison of which version is better and why  **Observation :**   * **The program checks if a year is divisible by 4 but not by 100, unless it’s also divisible by 400.** * **It usually uses a function to keep the code organized.** * **It asks the user to enter a year.** * **It prints a clear message saying whether the year is a leap year or not.** * **The code is easy to read and understand.**   **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**  **Evaluation Criteria:**   | **Criteria** | **Max Marks** | | --- | --- | | Two sorting implementations: Bubble sort (manual logic) and Built-in sort() (Task#1) | 0.5 | | Counts vowels, consonants, and digits(Task#2) | 0.5 | | Functional code that creates a .txt file, writes content to it, and reads it back- Use cursor (Task#3) | 0.5 | | Complete calculator code with user input and operation selection. (Task#4) | 0.5 | | A functional program to check leap year with sample input/output-use Cursor (Task#5) | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week1 - Thursday |  |