|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week2 - Monday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | | |
| **Assignment Number:4.1**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques  **Lab Objectives:**   * To explore and apply different levels of prompt examples in AI-assisted code generation. * To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality. * To evaluate the impact of context richness and example quantity on AI performance. * To build awareness of prompt strategy effectiveness for different problem types.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use zero-shot prompting to instruct AI with minimal context. * Use one-shot prompting with a single example to guide AI code generation. * Apply few-shot prompting using multiple examples to improve AI responses. * Compare AI outputs across the three prompting strategies.   **Task #1 – Zero-Shot Prompting with Conditional Validation**  Objective  Use zero-shot prompting to instruct an AI tool to generate a function that validates an Indian mobile number.  Requirements   * The function must ensure the mobile number:   + Starts with 6, 7, 8, or 9   + Contains exactly 10 digits   Expected Output   * A valid Python function that performs all required validations without using any input-output examples in the prompt.   Prompt: write a python function that validates an Indian mobile  Number (start with 6,7,8 or 9) with user input.    Output :      Explanation: this code was generated by using zero shot prompt in that def function was used in the end if-else statements to produces the invalid mobile in india.  **Task #2 – One-Shot Prompting with Edge Case Handling**  Objective  Use one-shot prompting to generate a Python function that calculates the factorial of a number.  Requirements   * Provide one sample input-output pair in the prompt to guide the AI. * The function should handle:   + 0! correctly   + Negative input by returning an appropriate message   Prompt: write a python function that generated the calculates the factorial of a number with user input.  Eg : 5 = 5\*4\*3\*2\*1=120.    Expected Output   * A Python function with correct factorial logic and edge case handling, generated from a single example.   Output:    Explanation: this code was generated by using one shot prompt mean with an a ‘one-eg’ that ai can understand the code without using translate prompt with the ‘one -eg’ ai will understand the code and gives the code similary with given eg.  **Task #3 – Few-Shot Prompting for Nested Dictionary Extraction**  Objective  Use few-shot prompting (2–3 examples) to instruct the AI to create a function that parses a nested dictionary representing student information.  Requirements   * The function should extract and return:   + Full Name   + Branch   + SGPA   Expected Output   * A reusable Python function that correctly navigates and extracts values from nested dictionaries based on the provided examples.   Prompt : write a python function that generated the parases a nested dictionary representing student information with user input.  Eg 1: Full name = kim rose  Branch : cse  Sgpa: 9.5  Eg2: Full name: Min josh  Branch : EEE  Sgpa : 9.4    Output:    Explanation: this code was generated by using few shot prompt mean with an a ‘two to three-egs’ that ai can understand the code without using translate prompt with the ‘two to three-egs’ ai will understand the code and gives the code similary with given egs.  **Task #4 – Comparing Prompting Styles for File Analysis**  Objective  Experiment with zero-shot, one-shot, and few-shot prompting to generate functions for CSV file analysis.  Requirements   * Each generated function should:   + Read a .csv file   + Return the total number of rows   + Count the number of empty rows   + Count the number of words across the file   Expected Output   * Working Python functions for each prompting style, with a brief reflection comparing their accuracy, clarity, and efficiency.   Prompt : zero shot : write a python function that reads a.csv file and return .(the total number of rows, the number of empty rows, the total number of words across all cell in the file).  Prompt : one-shot: write a python function that reads a.csv file and return(total number of rows, number of empty rows, total number of words across all cells).  Eg: def count\_lines(file\_path):  With open(file\_path) as f:  return len(f.readlines())  prompt : write the python function (that reads a.csv file, return the total number of rows, count how many rows are completely  empty, calculates the total number of words across all cells).  Eg: 1: def count\_words(text):  return len(text.split())  2: def is\_empty(row):  return not any(cell.strip() for cell in row)            **Task #5 – Few-Shot Prompting for Text Processing and Word** **Frequency**  Objective  Use few-shot prompting (with at least 3 examples) to generate a Python function that processes text and analyzes word frequency.  Requirements  The function must:   * Accept a paragraph as input * Convert all text to lowercase * Remove punctuation * Return the most frequently used word   Expected Output   * A functional Python script that performs text cleaning, tokenization, and returns the most common word using only the examples provided in the prompt   Prompt : write the python function that processes text and analyses word frequency.  Eg1: praragraph1 = “Hello world! Hello again”  Output= ‘hello’  Eg2:paragraph2=”python is great. Python is dynamic.”  Output=’python’  Eg3: paragraph3=”cats, dogs and birds. Dogs are friendly”  Output=’dogs’  I was done this code in cursor because of some network issues in vsc.    Output:    Explanation:  Imports regex and defines function to find most frequent word in text.  Converts text to lowercase and extracts only words using regex pattern.  Counts word frequencies by storing each word and its count in a dictionary.Finds maximum frequency and collects all words that appear that many times.Returns the first most frequentword from the list (handles ties by picking first one).  **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** | | --- | --- | | Zero Shot (Task #1) | 0.5 | | One Shot (Task#2) | 0.5 | | Few Shot (Task#3, Task#4 & Task #5) | 1.5 | | **Total** | **2.5 Marks** | | | | | | | Week2 - Monday |  |