|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **AcademicYear:**2025-2026 | |
| **Course Coordinator Name** | | | | Venkataramana Veeramsetty | | | | |
| **Instructor(s)Name** | | | | 1. Dr. Mohammed Ali Shaik  2. Dr. T Sampath Kumar  3. Mr. S Naresh Kumar  4. Dr. V. Rajesh  5. Dr. Brij Kishore  6. Dr Pramoda Patro  7. Dr. Venkataramana  8. Dr. Ravi Chander  9. Dr. Jagjeeth Singh | | | | |
| **Course Code** | | | 24CS002PC215 | **Course Title** | | AI Assisted Coding | | |
| **Year/Sem** | | | II/I | **Regulation** | | R24 | | |
| **Date and Day**  **of Assignment** | | | 06-08-2025 | **Time(s)** | |  | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | |
| **AssignmentNumber:4.5**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | |
|  | | | | | | | | |
|  | **Q. No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | **Lab 4: Advanced Prompt Engineering: Zero-shot, one-shot, and few-shot techniques**  **Objective:** To explore and compare Zero-shot, One-shot, and Few-shot prompting techniques for classifying emails into predefined categories using a large language model (LLM).  Suppose that you work for a company that receives hundreds of customer emails daily. Management wants to automatically classify emails into categories like "Billing", "Technical Support", "Feedback", and "Others" before assigning them to appropriate departments. Instead of training a new model, your task is to use prompt engineering techniques with an existing LLM to handle the classification.  Tasks to be completed are as below  1. **Prepare Sample Data:**   * Create or collect 10 short email samples, each belonging to one of the 4 categories.     2. **Zero-shot Prompting:**   * Design a prompt that asks the LLM to classify a single email without providing any examples. * Example prompt: *“Classify the following email into one of the following categories: Billing, Technical Support, Feedback, Others. Email: ‘I have not received my invoice for last month.’”*   PROMPT:  Classify email messages into "Billing", "Technical Support", or "Others" using keyword matching in Python.    **3. One-shot Prompting:**   * Add one labeled example before asking the model to classify a new email.   PROMPT:  Write a Python program that lets the user enter a list of items, sorts them alphabetically, and uses binary search to find a specific item. The program should display clear messages and print the index of the item if found, or say it's not found    4. **Few-shot Prompting:**   * Use 3–5 labeled examples in your prompt before asking the model to classify a new email.   PROMPT:  Write a Python program that classifies email messages into categories based on their content. The program should: – Ask the user to enter an email message – Use keyword matching to classify the message into one of three categories:     • "Billing" (e.g., contains words like "billing", "payment")     • "Technical Support" (e.g., contains words like "error", "bug", "support", "assist")     • "Others" (if no keywords match) – Print the classification result    5. **Evaluation:**   * Run all three techniques on the same set of 5 test emails. * Compare and document the accuracy and clarity of responses.   **OBSERVATION:**  The Python program effectively classifies email messages into three categories—Billing, Technical Support, and Others—based on keyword matching. It prompts the user to input an email message, converts it to lowercase for case-insensitive comparison, and checks for specific keywords like "billing", "payment", "error", "bug", "support", and "assist". The classification logic is accurate and aligns well with the intended purpose. The code is structured clearly, with a dedicated function (classify\_email) that makes it easy to understand and maintain. User interaction is simple and intuitive, and the output is displayed with clear messaging. Sample comments included in the code help illustrate expected behavior, making the program beginner-friendly and practical for basic automation tasks.  **Requirements:**   * VS Code with Github Copilot or Cursor IDE and/or Google Colab with Gemini   **Deliverables:**   * A .txt or .md file showing prompts and model responses. * A comparison table showing classification accuracy for each technique. * A short reflection on which method was most effective and why   . | | | | | | 08.08.2025 EOD |  |