|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 - Thursday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | | |
| **AssignmentNumber:10.4**(Present assignment number)/**24**(Total number of assignments)  NAME: P. HEMAN ROLL NO: 2403A510F5  BATCH-06 DATE: 10-09-2025 | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | **Lab 10 – Code Review and Quality: Using AI to Improve Code Quality and Readability**  **Lab Objectives**   * Use AI for automated code review and quality enhancement. * Identify and fix syntax, logical, performance, and security issues in Python code. * Improve readability and maintainability through structured refactoring and comments. * Apply prompt engineering for targeted improvements. * Evaluate AI-generated suggestions against PEP 8 standards and software engineering best practices   **Task 1: Syntax and Error Detection**  **Task:** Identify and fix syntax, indentation, and variable errors in the given script.  # buggy\_code\_task1.py  def add\_numbers(a, b)  result = a + b  return reslt  print(add\_numbers(10 20))    **Expected Output**:   * Corrected code with proper syntax (: after function, fixed variable name, corrected function call). * AI should explain what was fixed.   CORRECTED CODE:    OBSERVATION:  Here are the errors in 10.3.py:   1. Missing colon : after the function definition (def add\_numbers(a, b)). 2. The function body is not indented. 3. Typo in the return statement: reslt should be result. 4. The function call is missing a comma between arguments: add\_numbers(10 20) should be add\_numbers(10, 20)   **Task 2: Logical and Performance Issue Review**  **Task**: Optimize inefficient logic while keeping the result correct.  # buggy\_code\_task2.py  def find\_duplicates(nums):  duplicates = []  for i in range(len(nums)):  for j in range(len(nums)):  if i != j and nums[i] == nums[j] and nums[i] not in duplicates:  duplicates.append(nums[i])  return duplicates  numbers = [1,2,3,2,4,5,1,6,1,2]  print(find\_duplicates(numbers))  **Expected Output**:   * More efficient duplicate detection (e.g., using sets). * AI should explain the optimization.   CORRECTED CODE:    OBSERVATION:  Here are the errors and inefficiencies in your code:   1. **Inefficient Logic:**    * The code uses a nested loop (O(n²)), which is slow for large lists. 2. **Redundant Comparisons:**    * It compares every pair twice (i, j) and (j, i), which is unnecessary. 3. **Incorrect Duplicate Detection:**    * The logic may add the same duplicate multiple times if not careful, but the if nums[i] not in duplicates check prevents this. 4. **Not Handling All Edge Cases:**    * If a number appears more than twice, it still only gets added once, but the nested loop is wasteful. 5. **No Syntax Error:**    * The code will run, but is not optimal.   **Task 3: Code Refactoring for Readability**  **Task**: Refactor messy code into clean, PEP 8–compliant, well-structured code.  # buggy\_code\_task3.py  def c(n):  x=1  for i in range(1,n+1):  x=x\*i  return x  print(c(5))  Expected Output:  Function renamed to calculate\_factorial.  Proper indentation, variable naming, docstrings, and formatting.  AI should provide a more readable version.  CORRECTED CODE:    OBSERVATION:  **Errors in your code:**   1. Indentation is inconsistent (mix of tabs and spaces). 2. The code works, but for best practice, use consistent indentation (usually 4 spaces per level in Python)   **Task 4: Security and Error Handling Enhancement**  **Task:** Add security practices and exception handling to the code.  # buggy\_code\_task4.py  import sqlite3  def get\_user\_data(user\_id):  conn = sqlite3.connect("users.db")  cursor = conn.cursor()  query = f"SELECT \* FROM users WHERE id = {user\_id};" # Potential SQL injection risk  cursor.execute(query)  result = cursor.fetchall()  conn.close()  return result  user\_input = input("Enter user ID: ")  print(get\_user\_data(user\_input))  **Expected Output:**  Safe query using parameterized SQL (? placeholders).  Try-except block for database errors.  Input validation before query execution.  CORRECTED CODE:    OBSERVATION:  **Errors in your code:**   1. **SQL Injection Risk:** The query uses string formatting with user input, which is unsafe. 2. **Type Issue:** [user\_input](vscode-file://vscode-app/c:/Users/heman/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html" \o ") from [input()](vscode-file://vscode-app/c:/Users/heman/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) is a string, but the query expects an integer. 3. **Best Practice:** Use parameterized queries to prevent SQL injection.   **Task 5: Automated Code Review Report Generation**  **Task**: Generate a **review report** for this messy code.  # buggy\_code\_task5.py  def calc(x,y,z):  if z=="add":  return x+y  elif z=="sub": return x-y  elif z=="mul":  return x\*y  elif z=="div":  return x/y  else: print("wrong")    print(calc(10,5,"add"))  print(calc(10,0,"div"))  **Expected Output**:  AI-generated **review report** should mention:   * + Missing docstrings   + Inconsistent formatting (indentation, inline return)   + Missing error handling for division by zero   + Non-descriptive function/variable names   + Suggestions for readability and PEP 8 compliance   CORRECTED CODE:    OBSERVATION:  **1. Documentation Issues** The function has no docstring explaining purpose, parameters, or return value. A proper docstring (PEP 257) should be added for clarity.  **2. Formatting & Style Issues** Indentation and inline returns are inconsistent, reducing readability. PEP 8 violations like missing spaces and vague function names exist.  **3. Error Handling** Division by zero is not handled, causing a runtime error. Add a check or exception handling to prevent crashes.  **4. Naming Conventions** Function and variable names are non-descriptive. Use clear names like calculate, num1, num2, and operation.  **5. Readability Suggestions** Avoid else: print("wrong"), raise a ValueError instead. Keep formatting consistent and follow PEP 8 guidelines. | | | | | | Week5 - Thursday |  |