|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | | Dr. Rishabh Mittal | | | | | |
| **Instructor(s) Name** | | | | |  | | --- | | Mr. S Naresh Kumar | | Ms. B. Swathi | | Dr. Sasanko Shekhar Gantayat | | Mr. Md Sallauddin | | Dr. Mathivanan | | Mr. Y Srikanth | | Ms. N Shilpa | | Dr. Rishabh Mittal (Coordinator) | | Dr. R. Prashant Kumar | | Mr. Ankushavali MD | | Mr. B Viswanath | | Ms. Sujitha Reddy | | Ms. A. Anitha | | Ms. M.Madhuri | | Ms. Katherashala Swetha | | Ms. Velpula sumalatha | | Mr. Bingi Raju | | | | | | |
| **CourseCode** | | | 23CS002PC304 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | III/II | **Regulation** | | R23 | | | |
| **Date and Day**  **of Assignment** | | | **Week4 – Friday** | **Time(s)** | | 23CSBTB01 To 23CSBTB52 | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | | All batches | | | |
| **Assignment Number: 8.5**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | Lab 8: Test-Driven Development with AI – Generating and Working with Test Cases  **Lab Objectives:**   * To introduce students to test-driven development (TDD) using AI code generation tools. * To enable the generation of test cases before writing code implementations. * To reinforce the importance of testing, validation, and error handling. * To encourage writing clean and reliable code based on AI-generated test expectations.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use AI tools to write test cases for Python functions and classes. * Implement functions based on test cases in a test-first development style. * Analyze the completeness and coverage of AI-generated tests. * Compare AI-generated and manually written test cases for quality and logic   **Task Description #1 (Username Validator – Apply AI in Authentication Context)**   * Task: Use AI to generate at least 3 assert test cases for a function is\_valid\_username(username) and then implement the function using Test-Driven Development principles. * Requirements:   + Username length must be between 5 and 15 characters.   + Must contain only alphabets and digits.   + Must not start with a digit.   + No spaces allowed.   Example Assert Test Cases:  assert is\_valid\_username("User123") == True  assert is\_valid\_username("12User") == False  assert is\_valid\_username("Us er") == False  Expected Output #1:   * Username validation logic successfully passing all AI-generated test cases.   **Task Description #2 (Even–Odd & Type Classification – Apply AI for Robust Input Handling)**   * Task: Use AI to generate at least 3 assert test cases for a function classify\_value(x) and implement it using conditional logic and loops. * Requirements:   + If input is an integer, classify as "Even" or "Odd".   + If input is 0, return "Zero".   + If input is non-numeric, return "Invalid Input".   Example Assert Test Cases:  assert classify\_value(8) == "Even"  assert classify\_value(7) == "Odd"  assert classify\_value("abc") == "Invalid Input"  Expected Output #2:   * Function correctly classifying values and passing all test cases.   **Task Description #3 (Palindrome Checker – Apply AI for String Normalization)**   * Task: Use AI to generate at least 3 assert test cases for a function is\_palindrome(text) and implement the function. * Requirements:   + Ignore case, spaces, and punctuation.   + Handle edge cases such as empty strings and single characters.   Example Assert Test Cases:  assert is\_palindrome("Madam") == True  assert is\_palindrome("A man a plan a canal Panama") == True  assert is\_palindrome("Python") == False  Expected Output #3:   * Function correctly identifying palindromes and passing all AI-generated tests.   **Task Description #4 (BankAccount Class – Apply AI for Object-Oriented Test-Driven Development)**   * Task: Ask AI to generate at least 3 assert-based test cases for a BankAccount class and then implement the class. * Methods:   + deposit(amount)   + withdraw(amount)   + get\_balance()   Example Assert Test Cases:  acc = BankAccount(1000)  acc.deposit(500)  assert acc.get\_balance() == 1500  acc.withdraw(300)  assert acc.get\_balance() == 1200  Expected Output #4:   * Fully functional class that passes all AI-generated assertions.   **Task Description #5 (Email ID Validation – Apply AI for Data Validation)**   * Task: Use AI to generate at least 3 assert test cases for a function validate\_email(email) and implement the function. * Requirements:   + Must contain @ and .   + Must not start or end with special characters.   + Should handle invalid formats gracefully.   Example Assert Test Cases:  assert validate\_email("user@example.com") == True  assert validate\_email("userexample.com") == False  assert validate\_email("@gmail.com") == False  Expected Output #5:   * Email validation function passing all AI-generated test cases and handling edge cases correctly. | | | | | | Week4 -Friday |  |