**MODULE - 12**

**Assignment 23: Integration Testing**

**Objective**

Conduct integration testing for the IELTS Speaking Test platform to ensure seamless

functionality across all features and modules. Document test results, issues identified, and fixes applied during the process.

**Step-by-Step Approach to Conduct Integration Testing for the IELTS Speaking Test Platform**

**Understand the System and Requirements:**

Review the platform features and their interactions.

Familiarize yourself with the technology stack (React frontend, Flask backend, database, APIs).

Understand the test scope and objectives mentioned in the assignment.

**Prepare Testing Environment:**

Set up a controlled environment that mirrors the production setup.

Ensure all components (frontend, backend, database) are properly configured and connected.

**Define Integration Test Cases:**

Create detailed test cases covering all aspects of the platform:

Authentication:

Login with valid and invalid credentials.

Token issue and expiration handling.

Logout functionality.

**Role-Based Access Control:**

Test workflows for different user roles (admin, test taker).

Ensure restricted access to admin functionalities.

**API and UI Interaction:**

Validate correct data flow from API to UI components.

Test CRUD operations via frontend interactions.

**Critical Workflows:**

Validate complete end-to-end workflows (e.g., taking a test, submitting responses, and viewing results).

**Error Scenarios:**

Handle scenarios such as expired tokens, invalid login attempts, and failed API requests.

Test system behavior under high load (multiple concurrent users).

**Use Testing Tools:**

**API Testing:**

Utilize Postman to independently test API calls.

Validate request and response formats, including error handling.

**UI Testing:**

Use Browser DevTools to monitor network activity and ensure accurate data rendering.

Employ tools like Lighthouse or axe for accessibility testing of UI interactions.

Performance Testing:

Implement tools like JMeter or Locust to test system stability and performance under load.

**Execute Integration Tests:**

Conduct tests in a structured format:

Document each test case with test description, expected result, actual result, and status (e.g., passed, failed).

Take screenshots or logs as evidence of tested features.

Simulate real user interactions to validate frontend-backend integration and the flow of features.

**Identify and Resolve Issues:**

Identify any bugs or inconsistencies during testing.

Document the issues with clear descriptions and steps to reproduce them.

Collaborate with developers to apply fixes to identified issues.

Re-test after applying fixes to ensure the issues are resolved.

**Document Test Results:**

Create a comprehensive integration test report:

Summarize test cases with descriptions, expected and actual results, and status.

Provide evidence of testing like logs or screenshots.

Document any fixes applied along with a summary of identified and resolved issues.

**Submission:**

Compile the test report in a structured format (e.g., table or checklist).

Ensure to include all evidence of testing, such as logs, screenshots, and detailed descriptions of fixes.

Verify that the submission meets the guidelines provided and includes all required files and evidence.

**Deliverables:**

Integration Test Report:

Detailed documentation of each test case.

Results, including expected and actual outcomes.

Evidence of testing (screenshots/logs).

Summary of issues identified and fixes applied.

Evidence of Testing:

Screenshots of the platform during testing.

Logs demonstrating successful integration.

Evaluation Criteria Breakdown:

Comprehensive Test Coverage (40%):

Ensure all major features and interactions are tested thoroughly.

Bug Identification and Resolution (30%):

Effectively identify, document, and resolve issues.

Test Documentation (20%):

Clearly and thoroughly document test results and fixes.

Submission Completeness (10%):