Test Plan Strategies

Unit Testing - It is an essential component for testing the e-learning platform. We plan to use unit tests to thoroughly test every individual component and function used in our code. The aim for unit testing is to isolate each function and method and test their functionality without any interference from other code components. We essentially want to verify that each function or method can run individually and give correct results before we integrate the component pieces together. We plan to have a unit test for every single feature and function used in our e-learning platform project. This will include unit tests for creating courses, enrolling in courses, submitting assignments, posting grades, viewing grades, login to the account, creating an account, etc.

Integrated Testing - After running unit tests on all components of the code, we plan to then test the interaction of these components with each other in the code. We need to make sure that we can piece all the code components together so that they can interact with each other to impart functionality to the system. Hence, integration testing is the next important part of our testing plan for the code.
We plan to map out every single interaction between the functions and methods of the code and test them using integration testing. For example, we plan to have integration tests for the interaction between instructors posting grades and students being able to view these grades and students submitting assignments and instructors being able to view the submitted assignments. So essentially, the student should be able to view the correct grade only after it has been released by the instructor. Similarly, the instructor should not be able to view the students' assignment to grade it until after it has been submitted by the student.

Black Box Testing - It is a crucial component of our testing strategy for the e-learning platform project. This approach ensures that the software is tested from an end-user perspective, without requiring knowledge of its internal code structure. Given that our target audience includes students with varying levels of technical expertise, conducting black box testing aligns well with our goal of delivering a user-friendly and accessible platform.

We aim to thoroughly test all functionalities of the e-learning platform, including course enrollment, assignment submission, grade viewing, user authentication, navigation, content presentation, and interaction with multimedia elements. To ensure unbiased testing, we will create a comprehensive set of test cases covering typical user interactions, edge cases, error conditions, and boundary conditions.

Each test case will be documented, detailing the steps, expected outcomes, preconditions, and assumptions. Additionally, we will utilise realistic test data to simulate diverse user scenarios and inputs. Finally, we will document the results of black box testing in test reports, providing insights into the software's performance, any observed deviations from expected behaviour, and any defects encountered during testing.

Usability Testing Plan - We plan to conduct a usability testing in the mid way and final prototype stage
-In a controlled environment, either in-person or remotely.
Provide participants with clear instructions and scenarios to perform on the platform.

- Encourage participants to think aloud and express their thoughts, opinions, and frustrations during the testing sessions.
- Record participants' interactions, behaviours, and feedback using appropriate tools and techniques.

The aim is to identify usability issues, pain points, and areas for improvement based on observed user behaviours and feedback.

Here are some examples :

1. Login Page Unit Tests:

Test Scenario 1: Valid Login Credentials

Objective: Verify that users can successfully login with correct credentials.
Test Steps:
Enter valid username and password.
Submit the login form.
Verify that the user is redirected to the dashboard or designated landing page.
Expected Result: User should be logged in and redirected to the appropriate page.

Test Scenario 2: Invalid Login Credentials

Objective: Ensure that users cannot login with incorrect credentials.

Test Steps:
Enter invalid username and/or password.
Submit the login form.
Verify that the appropriate error message is displayed.
Expected Result: User should not be logged in and an error message indicating invalid credentials should be displayed.

2. Course Management Unit Tests:

Test Scenario: Admin Course Creation

Objective: Verify that admin can successfully create courses.
Test Steps:
Admin logs in with valid credentials.
Navigate to the course creation page.
Enter course details (e.g., title, description, duration).
Submit the form to create the course.
Expected Result: Course should be created successfully and appear in the list of available courses.

Test Scenario: Admin Enrollment of Students

Objective: Ensure that admin can enrol students in courses.
Test Steps:
Admin logs in with valid credentials.
Navigate to the student enrollment page for a specific course.
Select students to enrol in the course.
Confirm the enrollment.
Expected Result: Selected students should be successfully enrolled in the course.

3. Assignment Creation Unit Tests:

Test Scenario: Teacher Assignment Creation

Objective: Validate that teachers can create different types of assignments (quizzes and essays).
Test Steps:
Teacher logs in with valid credentials.
Navigate to the assignment creation page.
Choose the assignment type (quiz or essay).
Enter assignment details (e.g., title, instructions, deadline).
Save or submit the assignment.
Expected Result: Assignment should be created successfully and accessible to students.
Test Environment:

4) UNIT TEST FOR TEACHER CREATING THE ASSIGNMENT

Test Scenario 1: Quiz Creation

Test Steps:
Log in as a teacher using valid credentials.
Navigate to the assignment creation page.
Choose the quiz assignment type.
Enter assignment details such as title, instructions, and deadline.
Save or submit the assignment.
Expected Result: The quiz assignment should be successfully created and visible on the teacher's dashboard.

5. **UNIT TEST FOR STUDENTS ENROLLING IN COURSE**

Test Scenario : Adding a Course
Test Steps: Student logins with valid credentials.
Navigate to the “Add a Course” button.
Click on the Search for Course Option.
Select the desired course.
Click on the Register button.

Expected Result: The course should be successfully added and become visible on the student’s dashboard.

6. **UNIT TEST FOR STUDENTS SUBMITTING ASSIGNMENTS**

Test Scenario: Submitting Assignment
Test Steps:
The students clicks on Assignments button visible on the Dashboard
Click on the Assignment to be submitted.
Click upload Assignment.
Select the assignment to be uploaded.
Click on the Submit button.
Expected Result: The assignment should be successfully submitted and become visible under submitted assignments.

7. **UNIT TEST FOR TEACHERS GRADING ASSIGNMENTS:**
Test Scenario; grading an assignment
Test steps:
Teacher logins with valid credentials.
Click on the assignment to be graded.
A list of names of the students in the class shows up.
The professor is able to enter a grade for the student.
Expected result; the assignment should be successfully graded and the grades should show up on the students canvas.

8. **UNIT TEST FOR STUDENTS TO VIEW GRADES;**
Test scenario: viewing grades

INTEGRATION TESTING -
Test Scenario 1: Professor Grading Assignment

Test Steps:
Log in as a professor using valid credentials.
Access the assignment grading interface.
Select an assignment to grade.
Enter grades and feedback for students.
Save or submit the grades.
Expected Result: The grades and feedback should be successfully saved for the selected assignment.

Test Scenario 2: Student Viewing Grades

Test Steps:
Log in as a student using valid credentials.
Navigate to the grades or assignments section.

Find the relevant assignment and view grades and feedback provided by the professor.
Expected Result: The student should be able to view their grades and feedback for the assignment.

Tools: Python unittest framework.
Setup: Mock data for user accounts, courses, and assignments.