White Box Testing of the Authentication Module for ShopEase

As a testing engineer, our goal is to analyze the authentication module's source code, identify potential vulnerabilities, and design test cases to ensure its robustness and security. Whites Box Testing is a software testing technique that involves examining the internal structure, logic, and workings of the code. It is also known as Clear Box Testing, Open Box Testing, or Glass Box Testing because the internal code and system design are transparent to the tester.

1. Code Analysis:

- Examine the source code of the authentication module.
- Identify key functionalities such as:
- User input validation.
- Password hashing and storage mechanisms.
- Session management.
- Role-based access control.

2. Identify Potential Vulnerabilities:

- SQL Injection: Check for parameterized queries and proper input sanitization.
- Weak Password Policies: Verify if strong password criteria (e.g., length, complexity) are enforced.
- Session Management Flaws: Validate secure session creation and termination.

3. Design Test Cases:

Develop targeted test cases to validate the identified functionalities and safeguard against vulnerabilities.

Examples:

1.Submit a valid username and password:

Expected Result: User is authenticated successfully.

2.Submit an empty username or password field:

Expected Result: Authentication fails with a relevant error message.

3.Enter SQL injection payload (e.g., OR 1=1;):

Expected Result: Authentication fails, and no database leakage occurs.

4.Use a weak password (e.g., "12345"):

Expected Result: The system prompts the user to create a stronger password.

5.Attempt brute force login:

Expected Result: The account is locked after a predefined number of failed attempts.

6.Manually tamper with session tokens:

Expected Result: The session is invalidated, and the user is logged out.

7.Input a username exceeding the expected length:

Expected Result: The application handles the input gracefully without crashing

Expected Learning Outcomes:

- Identification of Code Vulnerabilities: By diving into the code, you'll uncover potential issues that compromise security.
- Strengthening Security: By crafting and executing detailed test cases, you'll enhance the robustness of the authentication module.
- Understanding Testing Nuances: Gain hands-on experience with white box testing techniques in a real-world scenario.