Handover Document

Event Planner and Scheduler

Handover Team – I

Overview

This report aims to discuss the failed test cases and how the relevant bugs/errors were resolved.

White Box Test Cases

Test Case No. 7:

Test Description:

This test case aimed to verify that two User instances with the same attributes should not be considered equal. However, the test failed because the equals method incorrectly returned true, indicating equality between instances that should be distinct.

Solution:

The code was corrected to pass the mentioned test.

Inference:

By correctly implementing and overriding these methods, the User class now accurately compares instances for equality based on specified fields and generates consistent hash codes for equal objects. This approach adheres to best practices for handling object equality and ensures reliable behavior across different parts of the software that rely on these comparisons.

Test Case No. 5:

Test Description:

This test suite aims to validate the DAO operations related to user registration, ensuring that username and email uniqueness constraints are properly enforced. The failure in the "Add User Test" highlights the importance of robust uniqueness validation mechanisms to maintain data integrity and prevent duplicate entries in the system.

Solution:

A temporary solution was devised because the test case is flawed.

Inference:

The temporary solution involves updating the test case with new credentials each time to bypass the flawed constant values, ensuring the test case runs successfully. However, it's noted that in the actual application environment, this functionality operates correctly.

Test Case No. 2:

Test Description:

This test case verifies the Singleton pattern implementation for Session Context, ensuring that only one instance exists and remains consistent throughout the user's session. The failure suggests a problem with resetting or managing the Singleton instance, impacting the reliability of session management and user data access functionalities within the application.

Solution: The Test Case is wrong.

Inference:

1. The test case expects instance1 and instance2 to be different, which contradicts the Singleton pattern. In a correct Singleton implementation, SessionContext.getInstance() should return the same instance (instance1 should equal instance2).
2. It contradicts what it says in the test description.

Test Case No. 4:

Test Description:

This test case validates the Event DAO's functionality for adding events and retrieving them by user ID. The failure highlights a problem with the addEvent operation, which prevents successful addition of events to the database. Resolving this issue is crucial for ensuring reliable event management capabilities within the application, particularly concerning data integrity and user-specific operations.

Solution: The Test Case is wrong.

Inference:

1. When creating an event, it sets user id 0, but retrieves for user id 1.
2. This functionality works correctly in the software
3. The time changes during code run, which returns is as not equal.

Test Case No. 6:

Test Description:

This test case assesses various aspects of event creation and validation. The failure highlights a specific issue where events created with timestamps after 12:00 on the current day were not properly flagged as past events, despite the expectation that they should be. Addressing this issue is crucial to ensure accurate timestamp validation and proper handling of event creation constraints within the application.

Solution: The code is fixed.

Inference:

The problem is fixed, but then 4-5 other test cases failed because they were flawed. For instance, one test case attempted to create an event with a past time and then checked if the event was successfully created. However, creating an event with a past time triggers an exception, so the test case cannot succeed under those conditions.

Black Box Test Cases

Test Case No. 6:

Test Description:

This test case aims to validate the behavior of a sign-in process when username and/or password fields are left empty. The expected outcome includes specific error messages based on which fields are empty. However, the test case fails because the actual error message displayed is "invalid email/password," which does not align with the expected behavior outlined in the test steps. This discrepancy indicates a failure in the system's error handling logic for empty fields during the sign-in process.

Solution: The Test Case is wrong.

Inference:

Revealing whether the username or password is incorrect during login would compromise security. Additionally, due to password uniqueness and potential sharing among users, incorrectly identifying a username as incorrect when a password mismatch occurs could lead to confusion and potentially reveal sensitive information.

Test Case No. 15:

Test Description:

The test case involves attempting to create an event with a time that has already passed on the same day. The user, who is logged in and has access to create events, fills out all required fields and sets a time earlier than the current time on the same day. The expected outcome is to receive an error message stating "An event cannot be added in the past." However, the test fails because the event is successfully added despite the past time setting.

Solution:

The code was corrected to pass the mentioned test.

Inference:

The test case was resolved by introducing an exception for events created with a past time on the same day. This prevents such events from being successfully added, ensuring adherence to the expected behavior where events cannot be scheduled in the past.

Test Case No. 16:

Test Description:

This test case involves attempting to create an event with a description containing over 1000 characters. The user, who is registered and has access to the create event tab, correctly fills out the title, date, and time fields but exceeds the character limit for the description field. Upon clicking "Add event," the expected outcome is for the event to be successfully created. However, the test fails because the program becomes unresponsive, indicating a failure to handle descriptions exceeding the specified character limit effectively.

Solution:

The code was corrected to pass the mentioned test.

Inference:

Increasing the character limit for event descriptions to 2000 resolved the issue of program unresponsiveness when descriptions exceeded 1000 characters, ensuring smooth functionality.

Test Case No. 20:

Test Description:

This test case involves attempting to edit an existing event using the Event Management System. The user, who is registered and logged in, navigates to the "Events" tab and attempts to edit an event by clicking the "Edit" button. The expected outcome is for options to edit the event to appear. However, the test fails because the "Edit" button does not function as expected, resulting in no action being performed when clicked.

Solution:

The code was corrected to pass the mentioned test.

Inference:

The issue with the non-functional "Edit" button in the Event Management System was resolved by implementing the necessary functionality. Users can now successfully edit events as intended, ensuring the system operates as expected for event management.