Software Quality Assurance Fall 2025

Bahria University Islamabad Campus

Saira Hameed

Assignment No 01

BS(CS) 8B

Name: Shams ul Islam

Enrollment #: 01-134212-166

Due Date: 05th Oct 2025

Assignment should be in pdf format.

Question No 1 **[Marks 05]**

Explain how a single software error introduced during coding can eventually lead to multiple software failures during system operation. Give one example.

A single software error (also called a fault or defect) is a flaw in the source code introduced during the development phase. When this error is executed under specific conditions, it can trigger multiple software failures, which are observable deviations from expected system behavior. The key distinction is that an error is a static defect in the code, while failures are dynamic manifestations of that error during runtime.

How One Error Leads to Multiple Failures:

A single coding error can cause multiple failures because:

Different Execution Paths: The same error may be encountered through different program flows, causing failures in various scenarios.

Cascading Effects: One error can corrupt data or system state, which then propagates through the system causing subsequent failures in other components.

Different Input Conditions: The same defective code may produce different failure symptoms depending on the input data or user interactions.

Timing Dependencies: In concurrent systems, the same error might manifest differently based on timing, thread scheduling, or race conditions.

Interconnected Components: Modern software systems are highly integrated, so an error in one module can affect multiple dependent modules.

Concrete Example:

Scenario: Banking System - Null Pointer Error in Account Validation

The Error (Single Coding Mistake):

```
public boolean validateAccount(Account account) {
   // ERROR: Missing null check
   if (account.getBalance() > 0) {
      return true;
   }
   return false;
}
```

Multiple Failures Resulting from This Single Error:

Failure 1 - ATM Withdrawal Crash

When a user tries to withdraw money through an ATM, the system calls validateAccount() with a null account object (due to network delay in fetching account details)

Result: NullPointerException → ATM application crashes

User Impact: Transaction fails, card may get stuck

Failure 2 - Online Banking Login Failure

During login, the system attempts to validate the account before displaying the dashboard

If account data hasn't loaded yet, null is passed to the method

Result: User sees error page instead of login success

User Impact: Cannot access online banking

Failure 3 - Mobile App Payment Failure

While processing a mobile payment, the app validates the account

If there's a temporary database connection issue returning null

Result: Payment transaction fails with cryptic error message

User Impact: Payment not processed, potential duplicate charges

Failure 4 - Batch Processing System Halt

Nightly batch job processing interest calculations calls this validation

If any account record is corrupted (null), the entire batch process crashes

Result: Interest not calculated for ANY customers

User Impact: Thousands of accounts affected, system-wide failure

Failure 5 - Customer Service Tool Crash

Bank representative trying to look up customer account information

System calls validation during account retrieval

Result: Customer service application becomes unusable

User Impact: In-branch services disrupted

Conclusion:

This example demonstrates how a single line of missing code (null check) can cascade into multiple distinct failures across different channels (ATM, web, mobile, batch systems, and customer service tools), affecting thousands of users in different ways. Each failure appears different to the end-user, but all stem from the same root cause—the single coding error in the validateAccount() method.