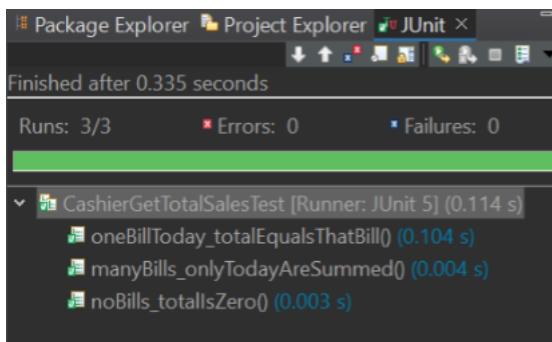


Testing Analysis

Antea Koxherri

1. Boundary Value Testing : Method: Cashier.getTotalSales()



Purpose This test class applies **Boundary Value Testing (BVT)** to verify the correctness of the `getTotalSales()` method. The focus is on edge cases related to the number of bills and the date-based selection of valid sales.

Test Case 1: No bills (Minimum boundary)

Input: A cashier with no bills recorded

Expected Result: The total sales value is 0.0

Explanation: This test represents the minimum input boundary (zero bills). The method should correctly return zero when no sales data is available.

Test Case 2: One bill today (Minimum + 1 boundary)

Input: A single bill created on the current day with a total amount of 150.50

Expected Result: The total sales equals 150.50

Explanation: This test verifies that the method functions correctly when there is exactly one valid bill for today.

Test Case 3: Multiple bills with mixed dates (Date filtering boundary)

Input: Two bills created on the current day with totals 100 and 200

One bill created on the previous day with a total of 999

Expected Result: The total sales is 300.0

Explanation: This test checks the logical boundary between bills that should be included (today's bills) and bills that should be excluded (bills from previous days).

Conclusion

These test cases cover the key boundary conditions of the `getTotalSales()` method: no bills present, a single valid bill, and correct exclusion of bills outside the current day.

As a result, the tests ensure that the method behaves correctly at its boundary values.

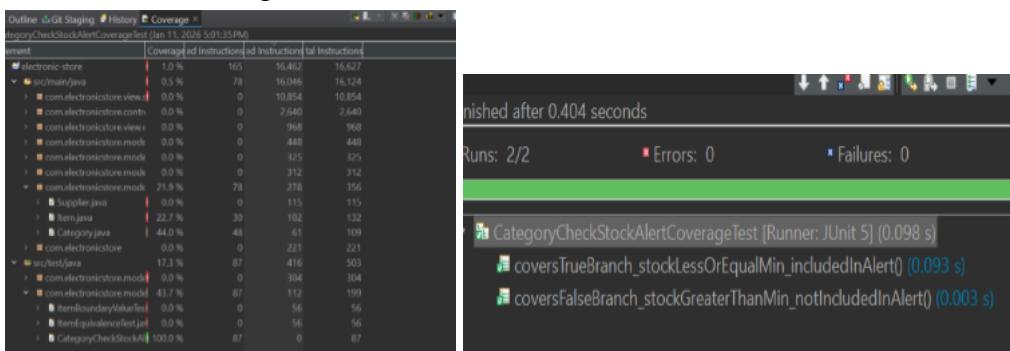
2. Code Coverage Testing – Category.checkStockAlert()

Purpose of testing : The purpose of Code Coverage Testing is to verify that all decision paths in the `checkStockAlert()` method are executed.

The condition `stockQuantity ≤ minStockLevel` was tested for both outcomes:

TRUE: item included in the alert list , **FALSE:** item excluded from the alert list.

Coverage Results : Both branches were executed successfully using two test cases. The coverage analysis shows: Statement Coverage: **100%** , Branch Coverage: **100%** , Condition Coverage: **100%**.



3. Equivalence Class Testing – Cashier.viewDailyBills()

Purpose of Testing

The purpose of this test is to verify that `viewDailyBills()` correctly returns bills created on the current day and ignores bills from previous days. The possible bill dates are divided into equivalence classes so that not every individual date needs to be tested.

Tested Equivalence Classes & Results

1) Today-bills class (`billDate = today`)

Representative values tested: two bills created on the current day.

In this case, the bills were returned in the result list, confirming that the method correctly includes all bills from today.

2) Old-bills class (`billDate < today`)

Representative value tested: one bill created on a previous day.

The method returned an empty list, confirming that bills from earlier dates are correctly excluded.

3) Empty input class (no bills saved)

Representative value tested: a cashier with no bills recorded.

The method returned an empty list, confirming safe handling when no bills are available.

4) Mixed-bills class (today + old bills)

Representative values tested: one bill created today and one bill created on a previous day.

The method returned only the bill from today and excluded the old bill, confirming correct filtering under mixed input conditions.

Conclusion

Equivalence Class Testing confirmed that

`viewDailyBills()` consistently

separates today's bills from older bills and safely returns an empty list when no bills are available.

