PaymentServiceTest_Part3

Overview

This document details additional unit test cases for PaymentService in our e-commerce platform. It covers alternative validation logic, payment lookup, and deletion scenarios.

Dependencies

- JUnit 5
- Mockito
- Spring's ResponseEntity
- Java time/date libraries

Mocked Components

- OrderRepository
- PaymentRepository
- UserRepository
- ProductRepository
- CartRepository
- InvoiceService

Test Cases

1. testDeletePayment_NotFound

Purpose: Ensure graceful handling when trying to delete a non-existent payment.

Test Scenario

- Arrange: Mock existsById to return false.
- Act: Call deletePayment with unknown ID.
- Assert: Expect HTTP 400 and appropriate error message.

Business Logic Verified

User-friendly error when trying to remove missing records.

2. testGetPaymentsByUserId_ReturnsList

Purpose: Check that all payments associated with a user are returned.

Test Scenario

- Arrange: Mock paymentRepository to return a list of payments for a userId.
- Act: Call getPaymentsByUserId.
- Assert: Assert list matches mocked data.

Business Logic Verified

Retrieves full transaction history for user dashboards or reports.

3. testGetBankInformation_InvalidExpiryFormat

Purpose: Ensure bank info check fails with bad expiry date format.

Test Scenario

- Arrange: Pass expiry as '2025-01' instead of 'MM/YY'.
- Act: Call getBankInformation with bad format.
- Assert: Expect HTTP 400 with format-specific error message.

Business Logic Verified

Input format must be enforced to avoid parsing failures.

Conclusion

These additional tests help ensure PaymentService robustly validates payment inputs, handles deletions, and properly retrieves transactional records.