# InvoiceServiceCoreTest Documentation

## Overview

This document details the unit tests implemented for the InvoiceService class in our e-commerce application backend. The InvoiceService is responsible for generating and emailing PDF invoices to customers after a purchase is completed.

## Test Class Structure

### Dependencies

- JUnit 5: For test execution and assertions
- Mockito: To mock dependencies and simulate interactions
- Spring Boot Test: For integration with the Spring testing framework

### Mocked Components

- JavaMailSender: Used to simulate email sending
- OrderRepository: Database access for order records
- UserRepository: Database access for user records
- ProductRepository: Fetches ordered products
- PdfInvoiceBuilder: Utility for generating PDF invoice byte stream

## Test Setup

Before each test, the following setup is performed:
1. Initialize mocks using MockitoAnnotations
2. Create a test order with sample data
3. Create a test user with associated details
4. Create a test product representing a purchased item

## Test Cases

### 1. testEmailPdfInvoice_Success

Purpose: Verify that a PDF invoice is generated and emailed when order and user exist.

#### Test Scenario:

- Arrange:
  Mocks return valid order, user, and product data. PDF builder returns a sample byte array.

- Act:
  Call invoiceService.emailPdfInvoice with a valid order ID.

- Assert:
  Verify email was sent, PDF was attached, and invoiceSentDate was saved.

**Business Logic Verified:**

Invoices are sent correctly when data is valid and dependencies are satisfied.

### 2. testEmailPdfInvoice_OrderNotFound

Purpose: Ensure that an exception is thrown if the order is not found.

**Test Scenario:**

- Arrange:
  Mock orderRepo to return Optional.empty().

- Act:
  Call invoiceService.emailPdfInvoice with an invalid order ID.

- Assert:
  Assert that IllegalArgumentException is thrown with 'Order not found'.

**Business Logic Verified:**

System should prevent invoice generation for nonexistent orders.

### 3. testEmailPdfInvoice_UserNotFound

Purpose: Ensure that an exception is thrown if the user is not found.

**Test Scenario:**

- Arrange:
  Mock orderRepo to return valid order, but userRepo to return Optional.empty().

- Act:
  Call invoiceService.emailPdfInvoice with a valid order ID.

- Assert:
  Assert that IllegalArgumentException is thrown with 'User not found'.

**Business Logic Verified:**

System must ensure customer exists before sending invoice.

### 4. testEmailPdfInvoice_PdfBuildFails

Purpose: Ensure that PDF builder failures are propagated.

**Test Scenario:**

- Arrange:
  Mocks return valid order, user, and product. PDF builder throws exception.

- Act:
  Call invoiceService.emailPdfInvoice.

- Assert:
  Assert RuntimeException is thrown with appropriate message.

**Business Logic Verified:**

System should handle failures in PDF generation gracefully.

## Mocking Strategy

The tests use a consistent mocking strategy to isolate the InvoiceService from its dependencies:
1. Mock Responses: Return prepared test objects when repository methods are called
2. Behavior Verification: Verify that the service calls dependent methods as expected
3. State Verification: Validate the responses and outcomes from the service

## Test Coverage

These tests cover the core functionality of the InvoiceService:
- Successful generation and sending of invoices
- Validation of order and user existence
- PDF generation failure handling
- Invoice sent date persistence
- Proper structure and dispatch of email

## Conclusion

The InvoiceService test classes verify critical aspects of the invoice generation and dispatch process. This ensures reliable communication with customers and auditability of financial transactions in the application backend.