RemoveFromCartTest Documentation

Overview

This document details two new unit tests for the `removeFromCart` method in `CartService`, ensuring correct behavior when removing existing items and appropriate handling when attempting to remove non-existent items.

Test Class Details

- **Class name:** `CartServiceTest`
- **Package:** `com.cs308.backend.services`
- **Service under test:** `CartService`
- **Testing frameworks:** JUnit 5, Mockito

Mocked Dependencies

Field	Role
`@Mock CartRepository`	Persists and retrieves `Cart` entities
`@Mock ProductRepository`	Provides product lookup (unused here)
`@InjectMocks CartService`	Contains business logic for cart ops

Test Cases

1. removeFromCart_ExistingItem_Successful

Verifies that an existing item is removed and the cart is saved.

```
@Test
public void removeFromCart_ExistingItem_Successful() {
    // Arrange
    CartItem existing = new CartItem(testProduct.getProductId(), 2, testProduct.getPrice());
    testCart.getItems().add(existing);
    when(cartRepository.findById(cartId)).thenReturn(Optional.of(testCart));
    when(cartRepository.save(any(Cart.class))).thenReturn(testCart);

    // Act
    ResponseEntity<Void> response = cartService.removeFromCart(cartId, testProduct.getProduc

    // Assert
    assertEquals(200, response.getStatusCodeValue());
```

```
    assertTrue(testCart.getItems().isEmpty(), "Cart should be empty after removal");
    verify(cartRepository, times(1)).save(testCart);
}
```

- **Arrange:** Cart contains one item; repository stubbed to return this cart and save it.
- **Act:** Call `removeFromCart` with the existing product ID.
- **Assert:** HTTP **200 OK**; cart's item list is empty; `save()` invoked once.

2. removeFromCart_NonexistentItem_ShouldReturnNotFound

Ensures that attempting to remove an item not in the cart returns 404 and does not save.

```
@Test
public void removeFromCart_NonexistentItem_ShouldReturnNotFound() {
    // Arrange
    when(cartRepository.findById(cartId)).thenReturn(Optional.of(testCart));

    // Act
    ResponseEntity<Void> response = cartService.removeFromCart(cartId, "missingId");

    // Assert
    assertEquals(404, response.getStatusCodeValue());
    verify(cartRepository, never()).save(any(Cart.class));
}
```

- **Arrange:** Cart exists but has no matching item.
- **Act:** Call `removeFromCart` with a non-existent product ID.
- **Assert:** HTTP **404 Not Found**; `save()` is **never** called.

Mocking Strategy

- Use Mockito's `@Mock` to stub `CartRepository` behavior.
- Use `when(...).thenReturn(...)` to simulate cart retrieval and saving.
- Use `verify(...)` to assert repository interactions.
