Assignment 2 Report
Software Verification and Validation
2019–2020

João David
49448

09/06/2020

1 HTML Unit

While defining the HTML Unit tests, some adaptations where made to the JSP files. The $name$ property was added to some HTML elements, in order to be able to get them using HTML Unit's API, those adaptations were made to the following files:

- CustomerInfo.jsp
- SalesInfo.jsp
- ShowSalesDelivery.jsp
- addSaleDelivery.jsp

2 DB Setup

2.1 Sales Behavior

The first Junit test verifies that it is not possible to add a sale to an non existent customer.

```
public void extraSaleBehaviour1() throws
   ApplicationException {
  int vat = 503183504;
  assertFalse(hasClient(vat));
  assertThrows(ApplicationException.class, () -> {
    SaleService.INSTANCE.addSale(vat);
  });
}
```

All new sales created for a new customer must have its date the same as of creation, a total of 0.0, an open status and it needs to be associated to the right vat.

```
public void extraSaleBehaviour2() throws
   ApplicationException {
  int vat = 503183504;
  assertFalse(hasClient(vat));
  CustomerService.INSTANCE.addCustomer(vat, "FCUL",
     217500000);
  assertTrue(hasClient(vat));
  SaleService.INSTANCE.addSale(vat);
  List<SaleDTO> sales =
     SaleService.INSTANCE.getSaleByCustomerVat(vat).sales;
  SimpleDateFormat dateFormat = new
     SimpleDateFormat("yyyy-MM-dd");
  for (SaleDTO curr : sales) {
    assertEquals(dateFormat.format(new Date()),
       curr.data.toString());
    assertEquals(new Double(0.0), curr.total);
    assertEquals("O", curr.statusId);
    assertEquals(vat, curr.customerVat);
  }
}
```

2.2 Sale Deliveries Behavior

After a sale has been closed, it should not be possible to add a delivery for that sale

```
public void extraSaleDeliveryBehaviour1() throws
   ApplicationException {
  int vat = 197672337;
  assertTrue(hasClient(vat));
  SaleService.INSTANCE.addSale(vat);
  assertEquals("O",
     SaleService.INSTANCE.getSaleById(1).statusId);
  SaleService.INSTANCE.updateSale(1);
  assertEquals("C",
     SaleService.INSTANCE.getSaleById(1).statusId);
  assertThrows(ApplicationException.class, () -> {
    SaleService.INSTANCE.addSaleDelivery(1, 1);
  });
}
```

After removing a customer, its sale deliveries should be removed as well

```
public void extraSaleDeliveryBehaviour2() throws
   ApplicationException {
  int vat = 197672337;
  assertTrue(hasClient(vat));
  SaleService.INSTANCE.addSaleDelivery(1, 1);
  assertNotEquals(0, SaleService.INSTANCE
       .getSalesDeliveryByVat(vat).sales_delivery.size());
  CustomerService.INSTANCE.removeCustomer(vat);
  assertFalse(hasClient(vat));
  assertEquals(0, SaleService.INSTANCE
    .getSalesDeliveryByVat(vat).sales_delivery.size());
}
```

3 Mockito

4 Bugs found

4.1 Customer Removal

After removing a registered customer from the system, its addresses, sales and sale deliveries were still kept in the database. This bug can be easily reproduced:

1. Create e new customer
2. Add an address
3. Insert a new sale
4. Insert a new sale delivery using the previous two information
5. Remove the customer
6. Use the customer's vat number to search for sales/sale deliveries

4.1.1 Solution

First implement in the Address, Sale and SaleDelivery RDGW classes, the methods responsible for deleting all addresses, sales and sale deliveries, respectively, given a customer vat number. Then, use them to delete the information in the $removeCustomer$ method in the $CustomerService$ Java class.