Ciências ULisboa | Informática

VVS (2018/2019)
Assignment 2

1 Introduction

The goal of this second assignment is to apply the integration tools to test some *wepapp demo* functionalities.

The webapp demo is a layered application with:

- a persistence layer (using row data gateways to access the database);
- a business layer organized by services;
- a presentation layer following the MVC pattern with servlets as controllers, JSPs as views and java beans' helpers as business layer models. These beans are populated with information that come from the business layer using Data Transfer Objects (DTO's).

The business layer deals with four concepts: Customers, Addresses, Sales and Sales Deliveries. The presentation layer allows the user to execute the available use cases, like inserting new customers or close existing sales.

2 What to do

Your mission, should you choose to accept it, is to:

1. Use HtmlUnit to perform and test the following narratives in the webapp running on Wildfly:
   (a) insert a new address for an existing customer, then the table of addresses of that client includes that address and its total row size increases by one;
   (b) get the first customer listed in the *List All Customers* use case, then try to insert it again and check if the expected error appears;
   (c) create a new customer, them remove him, and check if the list of all clients does not change;
   (d) create a new sale for an existing customer, insert a delivery for that sale and then show the sale delivery. Check that all intermediate pages have the expected information.
2. Use DbSetup to populate the database with sample data. These data should be organized to test the following tasks:
   (a) after the update of a costumer contact, that information should be properly saved;
   (b) after deleting all but one costumer, the list of all customers should have only that remaining customer;

(c) after deleting a certain costumer, its deliveries should be removed from the database;

(d) after deleting a certain costumer, it's possible to add it back without lifting exceptions;

(e) adding a new delivery increases the total number of all deliveries by one;

Add two extra tests concerning the expected behaviour of sales.

3. Is it possible to mock some SUT business layer's modules (in this case, services) in order to remove dependencies and test these modules separately? If yes, pick two modules A and B with dependency $A \rightarrow B$, mock A and test B using Mockito. If not, explain why you cannot use Mockito as it is, and propose what kind of refactoring would the SUT's business layer need, in order to perform these mocking tests (show an example to explain your methodology).

If necessary, in order to implement what is asked in this assignment, you can adapt the SUT. In the final report, you should list all modifications and explain why was necessary to perform them. Also, if some failures occur, found the respective faults, fix and report them.

3 Deliver

Upload a zip of the group's work into the course's moodle webpage until 23:59 of May 28. The zip file should be named `vvs02_XX.zip` where `XX` is your group's number.

The zip must include:

- the maven project (with just the source code and `pom.xml`, no binaries or Eclipse metadata)
- a pdf report