Report project assignment 3
BetterBe Group 3

Security analysis

For our system we are applying security to the employee page of the application. This is the page where items can be added or modified. We do this so only authenticated users who are from the company are able to access that page and no other users are able to modify the items. The accounts of the employees will be saved in the database.

We will hash the passwords of the accounts so if an unauthorized user will access the database it won't have access to other parts of the system. We don't have any security measures for the customer part of the system since everyone is allowed to access those pages. However, we are planning to add an account system for users, so they can save configurations, this will involve inclusion of the same security measurements as with employee accounts.

Software testing

Every method in our system we test thoroughly to see if it behaves as we expect. We have system tests for adding items, getting options of items and for our historical timeline. Methods which require running the system we test by using the console to see if we get the data that we expected and if we parse everything correctly for our methods. Another way that we check our methods is by running the web application and see if our page looks as expected.

Black box testing

To avoid bias, black box testing was performed by front-end developers, because they have fewer in-depth knowledge on back-end structure. Black box tests were simple yet covered most of the functionality, including:

- Adding new vehicle to the catalog (employee side)
- Configuring vehicle (including options combinations management)
- Navigating through web-pages, on different browsers, screen sizes, and devices.
- Browsing historical data (employee side)

All the tests were passed successfully. Nevertheless, some feedback given by testers gave us a better idea on how to improve the responsiveness of the UI, solutions were found and implemented right after the test as they were small.

Unit testing
For unit testing we've created java test:

- AddCarTest
    - Class dedicated to test the addition of new car objects through Dao class.
- Get optionsTest
    - Class created specifically to test the retrieval of option and rules data of a specific car.
- TimelineTest
    - Test class for retrieval of historical data of a certain car using Dao class.

Unit testing supports regression testing as we run them everytime we make any major changes or added functionality. We didn't face any problem except that the addRule method in Dao was working with the options table, instead of the rule table. It was obviously fixed fast, as it required little effort.

User story testing

For the user story testing we look over our user stories on the trello board in the categories doing or to review.

If a user story is in the category doing we check if what we are trying to implement corresponds to what we have to implement according to the user story.

If a user story is in the to review category we review what we have implemented and check if that corresponds to the user story. If that is the case the user story can be moved to the category done. This is how we check if all our user stories will be correctly implemented in our system.