Your emphasis for testing A04 is on the back-end "model" classes - you will have the Java "Bean" (the java version of each database table) and it's error message twin (what I call the "Evil Twin") as an example SystemUser.java and SystemUserErrorMsgs.java.

For these two you have data validation checks and rule checks (here the rule check is associated with the Primary key). You also have SystemUserDAO.java - the difference between SystemUser.java and SystemUserDAO.java is that the former is used when creating a NEW system user and the latter to check system users in the DB.

You will test SystemUser.java and its "evil twin" the SystemUserErrorMsgs.java together as a **single** unit test. Strictly speaking this is an integration level test and not a unit test, but it makes no sense to create the mocks for each individually and then still have to develop an integration test, so we do both to save time.

You will need 100 percent coverage of each Java Bean and its Evil Twin in both JaCoCo and PIT. Remember to set the latter to "All Mutators".

As you are testing SystemUser.java it will call SystemUserDAO.java to make the rule check. Here due to time limitations I am allowing you to directly call SystemUserDAO.java and not mock it with a strict unit test of SystemUser.java. Notice in doing this that you will not get 100% coverage of SystemUserDAO.java. We don't care.

You will get 100% coverage of each DAO and Controller with your Selenium tests. When we put these together we will get 100% automation coverage of the entire application.

The following is the **summary** of my coverage reports so that you can see what your expectations should be. This is only a summary - please see the A04 requirements for a description of the required JaCoCo and PIT test results.

