**SSW 555 Agile Methods for Software Development**

**Quiz 4: Testing**

**Name: Mrunal Phadke**

**-------------------------------------------------------------------------------------------------------------**

**1.What is the difference between unit testing and acceptance testing?**

**Answer:**

* **Unit Testing:**
* A unit test is a test written by a programmer to verify that a relatively small piece of code is doing what it is intended to do.
* They are narrow in scope, they should be easy to write and execute, and their effectiveness depends on what the programmer considers to be useful.
* The tests are intended for the use of the programmer, they are not directly useful to anybody else, though, if they do their job, testers and users downstream should benefit from seeing fewer bugs.
* Unit tests shouldn’t have dependencies on outside systems. They test internal consistency as opposed to proving that they play nicely with some outside system.
* With Unit tests you can typically test each method in each class to ensure functionality at a very low level. You decide what the unit is by looking for the smallest testable part.
* **Acceptance Testing**
* Acceptance tests make sure a feature or use case is correctly implemented.
* It is similar to an integration test, but with a focus on the use case rather than on the components involved.
* There are many ways to perform acceptance tests. The customer or user can manually test the application to make sure it meets each of the business requirements. Or you can use automated testing tools, such as Cucumber and Selenium.
* Acceptance tests act as the final verification of the required business functionality and proper functioning of the system, emulating real-world usage conditions on behalf of the paying client or a large customer.

**2.What types of feedback do tests provide?**

**Answer:** Tests can be useful for providing different feedbacks. These feedbacks depend upon types of tests performed. The types of feedbacks can be:

* Unit tests provide a feedback whether a module (a small function or class) is performing according to the decided functionality.
* Integration tests provide a feedback whether different pieces of the system work together as expected by the customer.
* Acceptance tests provide with a feedback whether the product or application is correctly implemented and is accepted by the customers.
* Regression tests are performed whenever anything has been changed in the system in order to check that no new bugs have been introduced.

**3. Why should you only write a few tests at a time when practicing TDD?**

* The general rules in TDD are:
  + Write tests first
  + Run the tests (they will probably fail)
  + Write some code
  + Rerun the tests
  + Debug until the tests pass
* So when you write some tests, some code should be implemented to pass the tests so that all the written tests can be tested successfully on that code.
* The purpose is to check whether the code is doing what the programmer wants.
* When you are done with a set of tests you are done implementing that module in the system (testing it and coding it).
* Thus to keep things manageable with respect to what needs to be fixed, it would be useful to have less number of tests at a time.
* Please note that restricting the number of tests does not mean that the tests are incomplete. We can restrict the number of tests by focusing on tests for individual modules only.

**4. Write a user story for some aspect of a university course registration system using the standard BDD template proposed by Dan North.**

* In Behavior Driven Development, the template format is:
  + Given some initial context (the givens),
  + When an event occurs,
  + Then ensure some outcomes.
* Thus, for a course registration system, user story will be:

Given student is registered in the University and we know about his

Term, Master/Bachelors, timings and all necessary details,

When he tries to register for 2 courses whose timings clash,

Then he will be notified that the timings clash and will be redirected to Select Courses page again.

**5. What is the purpose of the fixture in FitNesse?**

* FitNesse is a web server, a wiki and an automated testing tool for software.
* It is based on Ward Cunningham's Framework for Integrated Test and is designed to support acceptance testing rather than unit testing in that it facilitates detailed readable description of system function.
* FitNesse allows users of a developed system to enter specially formatted input (its format is accessible to non-programmers).
* This input is interpreted and tests are created automatically. These tests are then executed by the system and output is returned to the user.
* The advantage of this approach is very fast feedback from users. The developer of the system to be tested needs to provide some support (classes named "fixtures", conforming to certain conventions).
* Testing within the FitNesse system involves four components per test:

1. The wiki page which expresses the test as a decision table.

2. A testing engine, which interprets the wiki page.

3. A test fixture, which is invoked by the testing engine and in turn invokes the system under test.

4. The system under test, which is being tested

* The link between the generic testing engine and the system under test is made by a piece of Java code called a fixture.