**ASSIGNMENT-10**

**1.What qualities should a good Agile tester have?**

Ans:

Good qualities of an Agile tester:

* Self-organizing: The team is self-directed to make own decisions. The team manages themselves to understand each other and is expected to make the right calls at the right time without external command and control from the managers.
* Open to feedback: Agile testing is about giving feedback early, it means that if you find a bug; report it to the developer quickly.
* Co-located: The team is expected to sit together along with Scrum Master and Product Owner.
* Collaborative: Testers and developers mingle together to behave as ONE Team.
* Empowered: The team takes their own operational and technical decisions regarding development and testing .
* Committed: The teams along with testers are committed to the product quality to satisfy the Product Owner and customer needs.
* Credible: There is always a mutual trust and accountability that must be built with the team of testers and stakeholders as far as the test results are concerned.
* Choose the right set of tools to help the testing.Help the developers in writing automated tests and analyze the results of the automated tests.
* Be active in Meetings:Influence on quality assurance in agile development methodologies is needed to improve the product.Nw ideas for improving the project should be shareamong client in meetings.
* Continuous Integration: Provides frequent verification and notification of changes to the code and application
* End of Life Cycle Testing: Here independent test team validates that the system is ready to go into production.

**2.What is Test Driven Development (TDD)and how much does it help add business value?**

Ans:

Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle: requirements are turned into very specific test cases, then the software is improved to pass the new tests, only.Test-driven development is related to the test-first programming (TFD)concepts of extreme programming, There are two levels of TDD are as follows:-

* **Acceptance TDD (ATDD)** :- With ATDD you write a single acceptance test, or behavioral specification depending on your preferred terminology, and then just enough production functionality/code to fulfill that test.
* **Developer TDD**:- With developer TDD you write a single developer test, sometimes inaccurately referred to as a unit test, and then just enough production code to fulfill that test.

A test is written to define or improve a function. TDD makes the developer focus on the requirements before writing the code to write more robut tests.Since the TDD is done during every short time periods,it gives an efficient outcome. That is , there will not have any bugs. Thus we are able to release feature/features without any bugs to client during each iterations which increase the business value.

**3.What is CI? What is the effect of CI in schedule?**

Ans:

Continuous integration (CI) is a software engineering practice in which isolated changes are immediately tested and reported on when they are added to a larger code base. The goal of CI is to provide rapid feedback so that if a defect is introduced into the code base, it can be identified and corrected as soon as possible.

It involves:

* Frequent code check-ins
* Regularly scheduled, automated builds
* Automated tests
* Immediate feedback to developers detailing any build errors or failed tests

Normally, most of the companies schedule a particular date for releases. All though the coding is done before the relase date, they didn’t get enough time to test all parts. In such cases there may arise conflicts and confusions with the customers and companies.Releasing un-tested code to the customer can have huge consequences.

With Continous Integration,testing of all codes are completed before the release date.

But it can’t produce the whole features in a single release. Only a particular set of features are released without any bugs. So customer must be satisfied.

**4.What is configuration management?**

**a.Name some sample tools available for version control?**

**b.How does version control help in agile projects?**

Ans:

Configuration management (CM) refers to a discipline for evaluating, coordinating, approving or disapproving, and implementing changes in artifacts that are used to construct and maintain software systems. An artifact may be a piece of hardware or software or documentation.

**a) Name some sample tools available for version control?**

* **CVS :-** CVS may very well be where version control systems started. Released initially in 1986, Google still hosts the original Usenet post that announced CVS.
* **SVN :-** SVN, or Subversion as it is sometimes called, is generally the version control system that has the widest adoption.
* **GIT :-** Git is considered to be a newer, and faster emerging star when it comes to version control systems. First developed by the creator of Linux kernel, Linus Torvalds.
* **Bazaar :-** Bazaar is distributed version control system, which also provides a great, friendly user experience. Bazaar is unique that it can be deployed either with a central code base or as a distributed code base.

**b) How does version control help in agile projects?**

It is a repository to persist and track version of code and artifacts.In agile methodology, several teams work on the same modules of a single project. Thismay lead to code integration without proper communication among the teams. Version control can be used to avoid this problem. If in the midst of any project,we can go to an older version will be difficult. Version Control helps to go to an older version of a project.

**5.Describe Automated Testing vs Deployment Automation**

Ans:

**Automated testing**

It is the use of special software to control the execution of tests and the comparison of actual outcomes with predicted outcomes. Automated testing tools are capable of executing tests, reporting outcomes and comparing results with earlier test runs. Tests carried out with these tools can be run repeatedly, at any time of day. The tools are very useful to speed up the test cycle as they can replicate manual testing processes at a much faster rate.

**Unit tests**: Testing of smallest possible piece of code that can operate in isolation

**Integration tests**: Testing interaction between multiple components

**Acceptance tests**: Testing complete segments of a system to ensure it meets the business needs

**Deployment Automation**

Deployment Automation Tools help organizations improve the speed and quality of software releases, and address the challenges of manual software deployment. It helps to reduce errors eliminates rework, wait times and the business impact of failures in production.This is important because with manual deployments, the ability to deploy an update is often in the hands of the one or two people who built it. It can leave the company in a problem if there is any kind of need to take immediate action, such as fixing a bug in production or releasing more frequently to stay competitive in the market

* it reduces the number of errors
* anyone can deploy,experts arent needed
* lower costs
* it reduces complexity by using standardized script
* reduces risk by re-running same script
* reduces cost by speeding up deployment and frees up resources to focus on new features
* Major deployment tools- Saltstack,octopus deploy