**Agile Methodology**

Task Description:

Write a blog on below question.

QI. Describe the Differences in between:

1. Smoke & Sanity testing

Sanity and smoke testing are both [types of software testing](https://www.orientsoftware.com/blog/what-are-the-different-software-testing-types/) performed during the software development life cycle before release. The purpose of these tests is to decide whether the software works as expected under normal conditions. Both sanity and smoke testing provide general information about how well a system performs, but they mainly differ in their goals and execution.

Smoke tests operate under the assumption that the software is broken and that it will not run. During this process, the software is tested under extreme conditions of input values to determine whether the system responds as expected. While smoke tests can be performed manually, they are often automated with scripting languages.

On the other hand, sanity testing operates on the assumption that the software is working properly and requires no extreme values or conditions. The purpose of the test is to show that the software works under normal conditions and commonly expected values.

To summarize, smoke tests are designed to determine whether a system has broken components, while sanity testing is performed with the assumption that everything is working correctly. Moreover, there are some other minor differences between these two software testing techniques as below:

* Sanity testing is a more lightweight and faster form of testing that is typically used to test basic functionality, while smoke testing is a more comprehensive form of testing that is used to identify critical errors.
* Sanity testing is typically performed by the developer, while smoke testing is typically performed by the [Quality Assurance (QA) team](https://www.orientsoftware.com/services/qa-and-software-testing/).
* Sanity testing is intended to identify major defects, while smoke testing is intended to identify all defects, including minor ones.
* Sanity testing is usually done on a completed system, while smoke testing is done on an in-progress system.
* Sanity testing is done before a system goes into production, while smoke testing is done after a system goes into production.
* Sanity testing is performed more to display that the system will work, while smoke testing is performed to ensure that the system works under real-world conditions.
* Sanity tests typically take less time than smoke tests, although both are intended to be performed quickly.
* Sanity testing can be manual or automated, while smoke testing is always automated.
* Sanity testing provides immediate feedback to the developer, while smoke testing provides the QA team with information about defects.

1. Validation & Verification

* **Verification** is the process of checking that a software achieves its goal without any bugs. It is the process to ensure whether the product that is developed is right or not. It verifies whether the developed product fulfils the requirements that we have. Verification is static testing.
* **Validation** is the process of checking whether the software product is up to the mark or in other words product has high level requirements. It is the process of checking the validation of product i.e., it checks what we are developing is the right product. it is validation of actual and expected product. Validation is the dynamic testing.

Q2. Explain about Agile Methodology.

However, the phrase "agile methodology" is misleading because it implies that agile is a singular approach to software development. Agile is not a set of prescriptions for exactly which actions to take in software development. Instead, it is a way of thinking about collaboration and workflows and it is a set of values which guide our choices in regards to what we make and how we make it.

In practical terms, agile software development methodologies are all about delivering small pieces of working software quickly to improve customer satisfaction. These methodologies use adaptive approaches and teamwork to focus on continuous improvement. Usually, agile software development consists of small, self-organizing teams of software developers and business representatives regularly meeting in-person throughout the [software development life cycle](https://www.redhat.com/en/topics/security/software-development-lifecycle-security).

Q3. Explain about Epic and User Stories?

**Epic**: An epic is a large chunk of work you cannot deliver during one sprint. Instead, you can break it down into smaller, more specific units (called user stories). Of course, there are no set criteria to determine the largeness of an epic, so an epic’s size varies from company to company.

An epic usually encompasses multiple teams working on several projects during a set of sprints.

For example, in a software development project, a user authentication feature is an epic.

A collection of epics that drive toward a common goal form a theme.

**User Stories:** User stories are the smaller units of work in an agile framework. Several similar user stories make up an epic (it’s better to create an epic if there are more than five user stories with the same focus).

Software development is primarily user-centred, and user stories put actual end-users first. Using the epic and deriving a user story from it provides a concise context for the developing team; this way, they will have a distinct outlook on what to build and why build it.

User stories fit in the scrum framework neatly. During sprint planning meetings, teams decide which user stories to take on and discuss their features and requirements (and even determine their sizes and required effort by using [agile estimation techniques](https://winatalent.com/blog/2020/02/agile-project-estimation-and-techniques/)).