**What Is Functional Testing? --- verifies what software does**.

* Functional testing **makes sure everything works properly**. It confirms that the **software fulfils functionality requirements**. And it ensures that your software is ready to be shipped.

Typically use **functional testing**to check:

* Core functionality
* Basic usability
* Accessibility
* Error conditions

Types of **Functional Testing / Test levels:**

There are many different types of **functional testing**that**verify software**. the most common ones are:

**Unit Testing:**

* Developers test the most basic parts of code like classes, interfaces, and functions/procedures. They know how their code should respond and can make adjustments depending on output.
* It includes line, path, and method coverage. It can be [static or dynamic](https://www.perforce.com/blog/qac/what-static-code-analysis).
* There are different types of unit testing that are specific to programming languages, such as JUnit tests for Java and NUnit tests for Microsoft .NET.

**Integration Testing:**

* This identifies errors when **modules are integrated**. Different integration tests are **bottom up**, **top down**, and **functional incremental**.

**System Testing:**

* Components of a project are tested as a **whole** in **different environments** with this method.
* It falls under the **black box method** and is one of the final tests in the process. It determines if the **system functions** as it should to **meet business** and **user needs**.

**Acceptance Testing:**

* Acceptance testing can be done by people within the organization. This is known as internal acceptance testing. Or it may be done by people outside of your organization. This is known as **customer acceptance testing** and **user acceptance testing**.
* Also, under the scope of **black box testing**, clients test software to find out if the developer has created the program to desired specifications.

**Smoke Tests:**

* Smoke testing **verifies** that **software works** well enough to **move** into the **testing phase**. It’s typically done after a **build** is **complete**, but **before further tests** are **done**. This reduces defects found later in testing (which makes your testing efforts more cost-effective).
* Smoke testingis often done **manually**. But it **can be automated** to **accelerate** testing process.

**Sanity Tests:**

* Sanity testing **verifies** that **bugs** have been **fixed** after **minor code changes** are made to a build. In DevOps, **sanity tests** are done **during** the **release phase**. And they’re **usually run on just the affected components**.
* You can use sanity tests to verify that the software functions roughly as it’s expected. Like smoke tests, sanity tests provide a way to save time in the testing process.

**Regression Tests:**

* Regression tests **ensure** that **existing functionality works after code changes**. In regression testing, we **simply rerun tests** that you’ve **run before**.
* It takes a lot of time and resources to run regression tests. So, it’s **usually best to select a subset** of **tests** **rather** than **rerun the full test suite**. This minimizes costs while maximizing results.

**Interface Tests:**

* Interface testing **checks** if two **different pieces** of software can **communicate with one another**.

May do interface testing for:

* Workflows.
* Edge cases.
* Performance, load, and network testing.
* Individual systems.