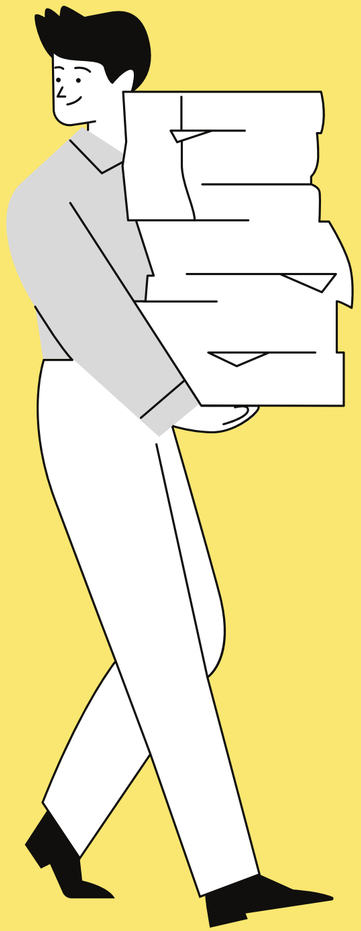
Andersen

Who is a tester? Types of testing

Lecture plan



01 Who is a tester and what does he do?

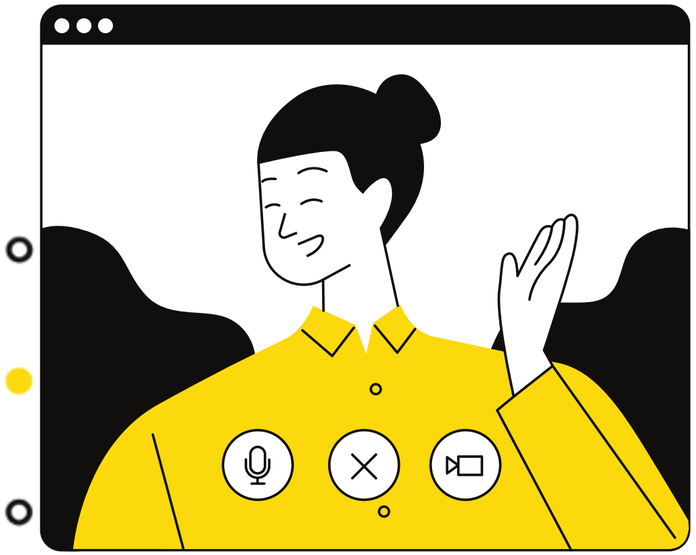
02 Differences between QA,QC, Tester

03 Types of testing

04 Kahoot game

05 Questions

01 Who is a tester and what does he do?



Who is considered to be a tester?

This is an engineer, responsible for quality control of software at all stages of its development.

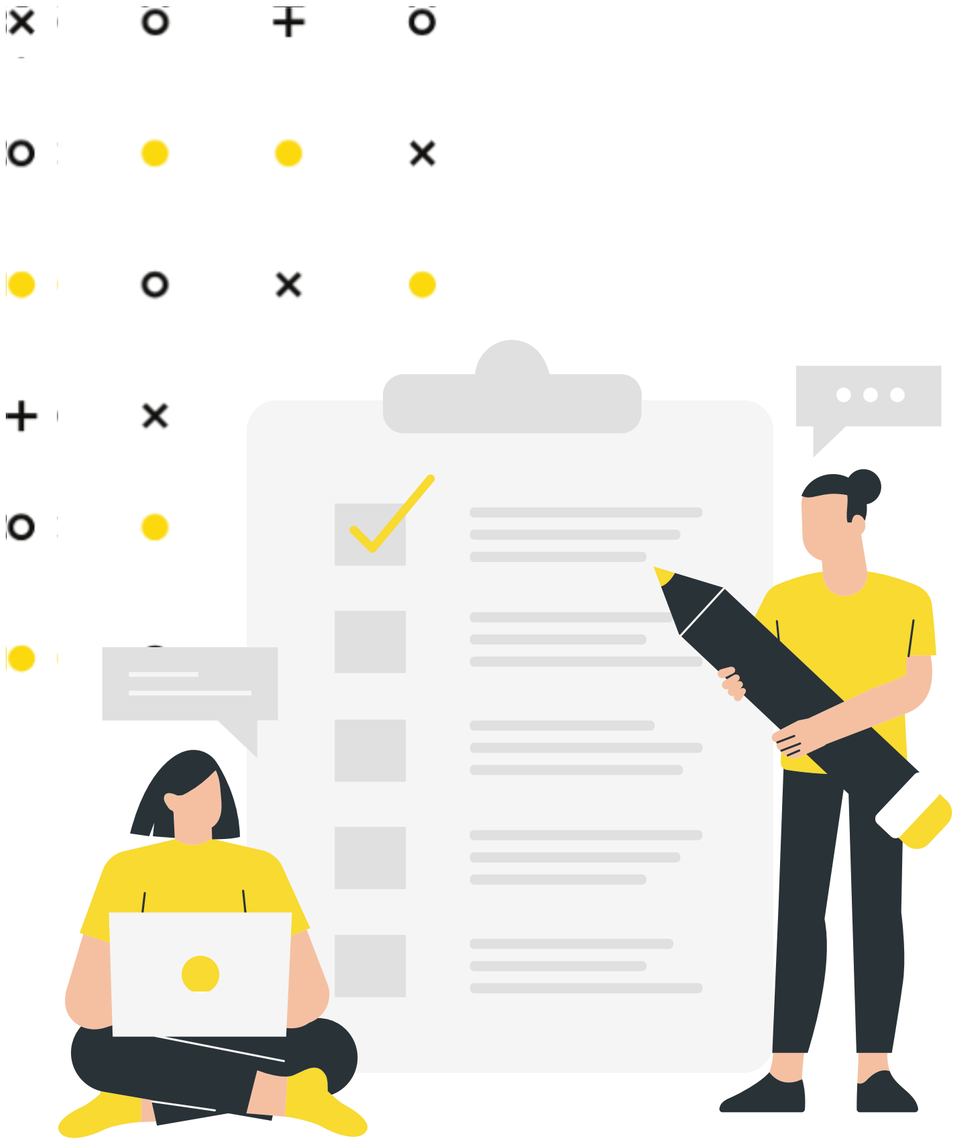
Why the job of a tester has appeared:

- The progress of the software development domain;
- Technical progress;
- Increase in the number of consumers;
- Challenge of controlling everything by yourself.

Mission:
to prevent the appearance of low-quality products.

Goal:
to find and report defects in the software so that it works correctly and safely for users.

Responsibilities and skills of a tester



Responsibilities:

- Testing software;
- Creating testing documentation;
- Tracking defects;
- Analyzing the reasons for their occurrence;
- Retesting of fixed defects.

Required skills:

- Fundamentals of network administration;
- Knowledge of database work;
- A descent English level;
- Fundamentals of programming language (will be a plus).

Testing Glossary

BUG (bug, defect) is a situation when the actual result doesn't match the expected result. Any bugs found should be documented in bug reports.

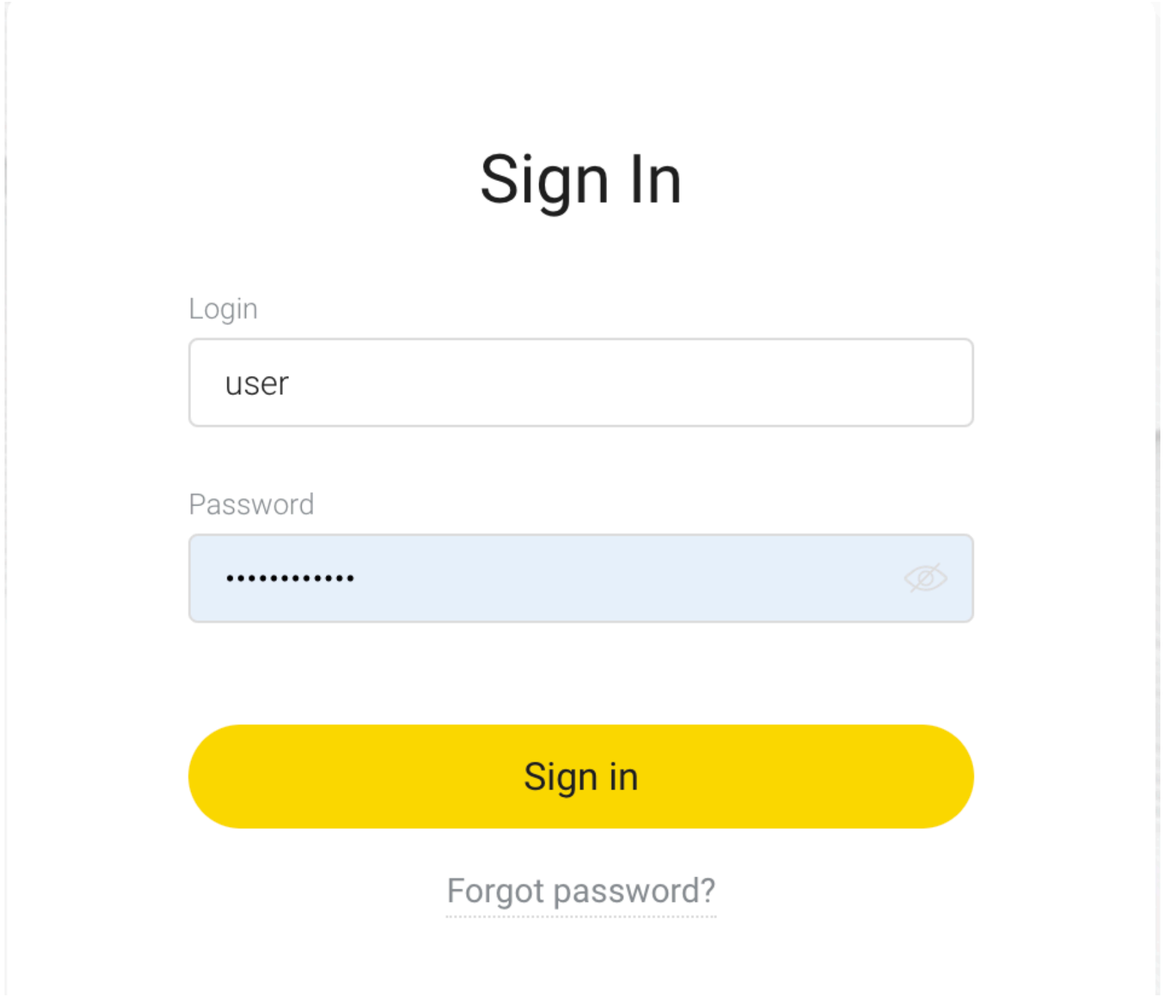
Expected result is a description of how the system should function according to the requirements.

Actual result is the outcome obtained by the tester during the testing process, showing how the system really works.

Testing Glossary

Verification is a **static** process that confirms and ensures that what has been created corresponds with what was planned.

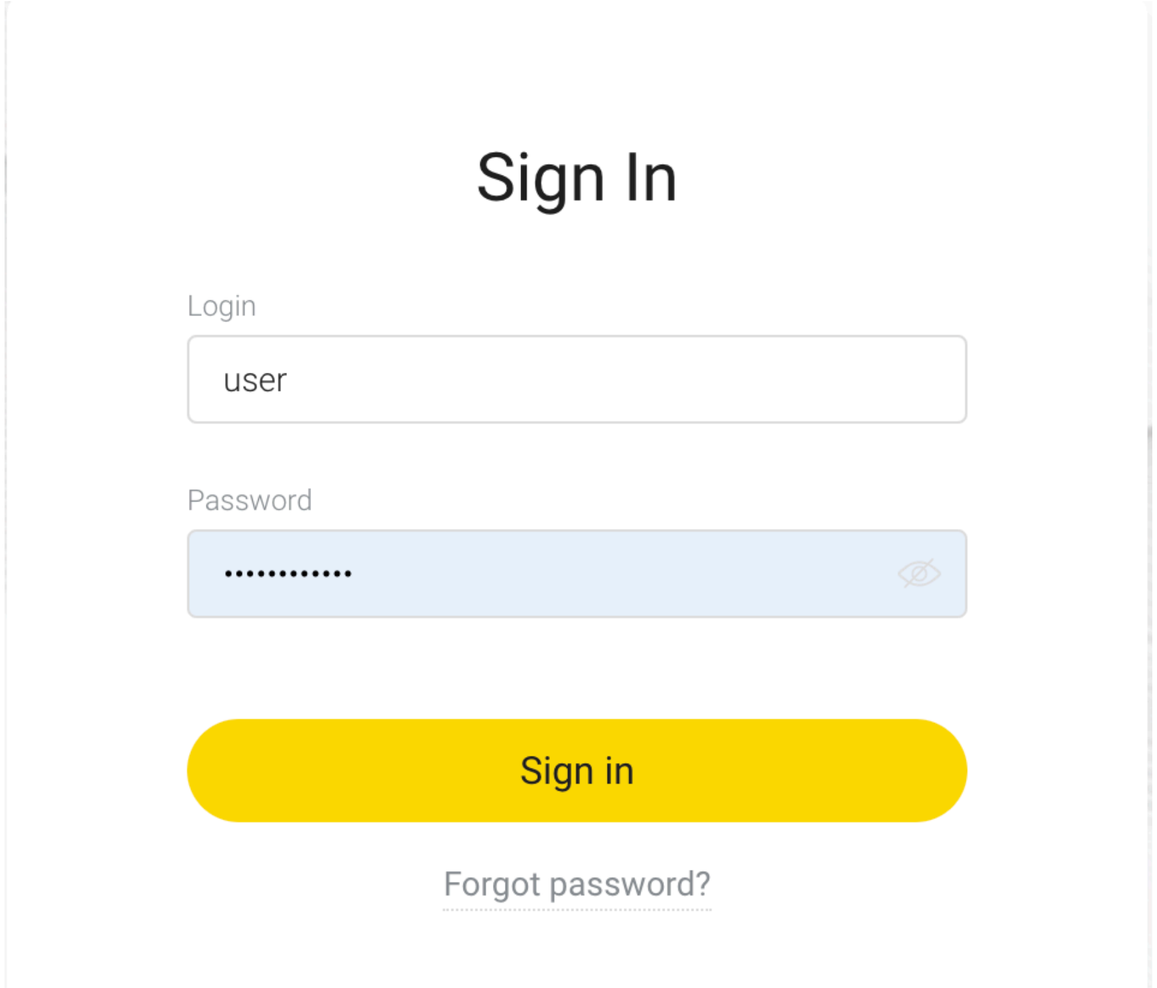
For example: according to the requirements, it is necessary to create a user login form on a website. In this case, checking the presence, sizes, and colours of form elements such as fields and buttons according to the requirements is **verification**. In other words, we confirm that the look of the form matches the design and requirements.



Testing Glossary

Validation is a **dynamic** process of ensuring that what has been created is truly useful and works as expected by users, bringing them benefits.

For example: attempting to log in with correct (valid) data, like filling in the fields and clicking the buttons, is **validation** of the form. This indicates that our form serves its main purpose and is helpful to users.



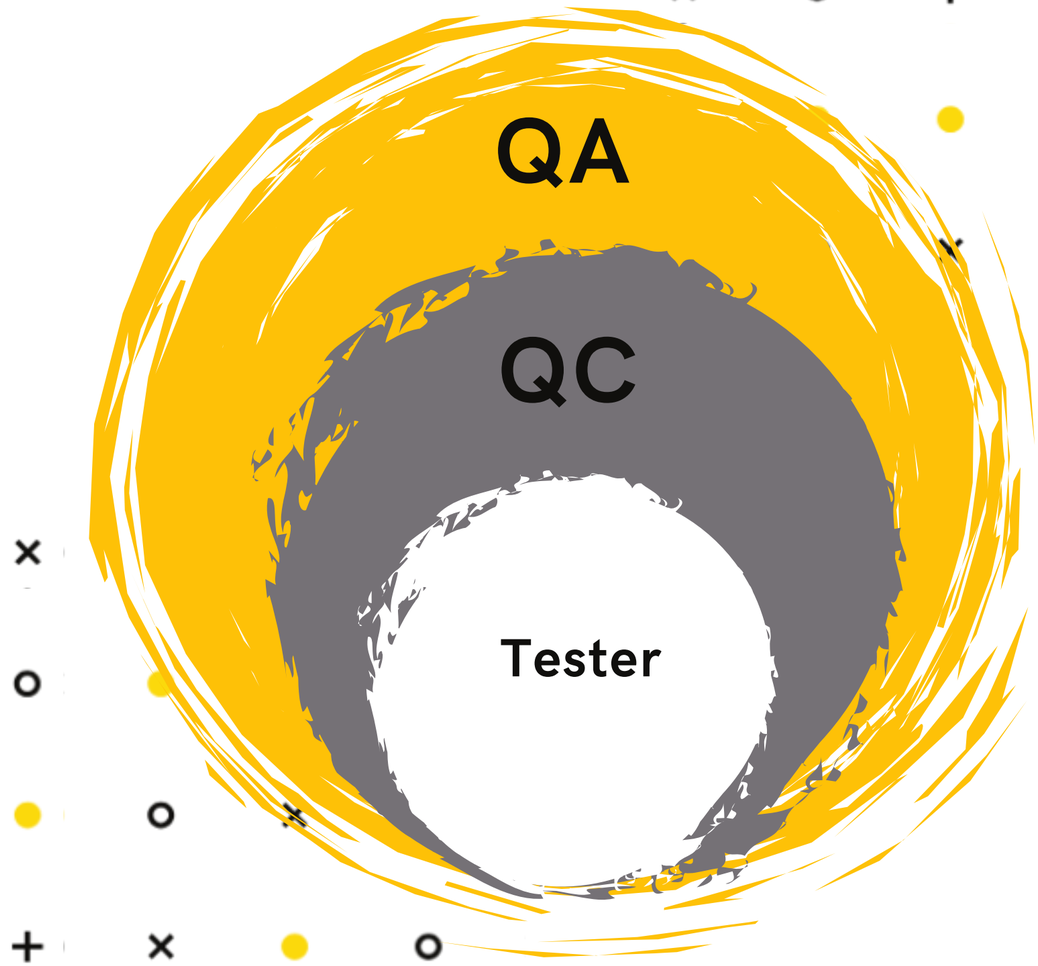
02 Difference between QA,QC, and a Tester

QA,QC, a Tester

Quality Assurance (QA) can be started **before the software is even available.** Its main goal is to develop methods to reduce bugs and guarantee quality at every step of software development.
It includes Quality Control (QC) and testing.

Quality Control (QC) is started whether **requirements are fulfilled**. A QC specialist evaluates the success of testing. QC process takes place when the **software is already available.**

A tester is an engineer who checks software, tries to find defects, and reports them for further correction.



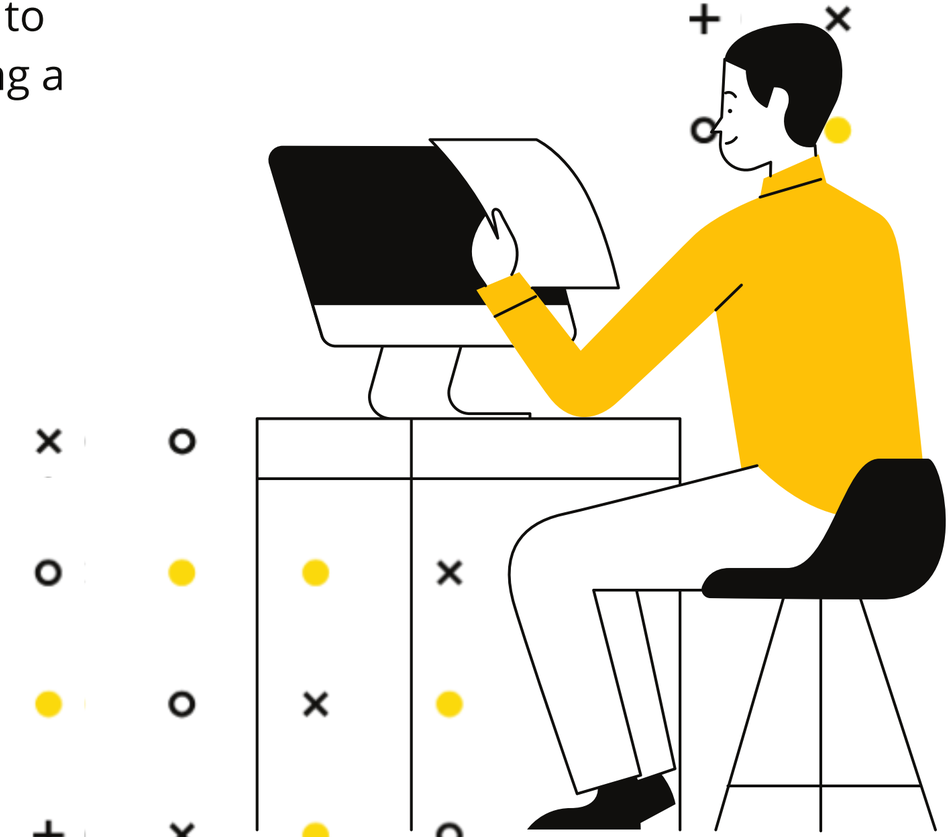
03 Types of Testing

Based on system knowledge

Black-box testing takes place when the tester **lacks access to the internal structure or code** of the application. This could happen due to limited understanding or a deliberate choice to not refer to them during testing. For instance, simply opening a website and testing its interface without diving into its code illustrates **black-box** testing.

White-box testing involves **having access to the internal structure and code** of the application. A tester also understands how it works, usually because he is testing the developer's code directly.

Gray-box testing is a **combination of white-box and black-box methods**. A tester has access to some parts of the code and architecture. For example, this could involve testing an API, using DevTools, or querying a database.



By positivity

Positive testing is verification that the application functions as it was planned.

- If there are requirements - we check according to them.
- If not - we test how the system operates without trying to break anything.

Negative testing is attempting to defy the rules and observe the system's reaction, testing its resilience by trying to disrupt it intentionally.

- Leaving the name input field blank.
- Attempting to upload a 1GB image instead of a 1KB one.

Functional testing

Functional testing - this is testing conducted according to functional requirements, defining the behavior of the software product.
It addresses questions like:

- **What** should **happen**?
- **What** is the **expected behavior** of the system?

Functional types:

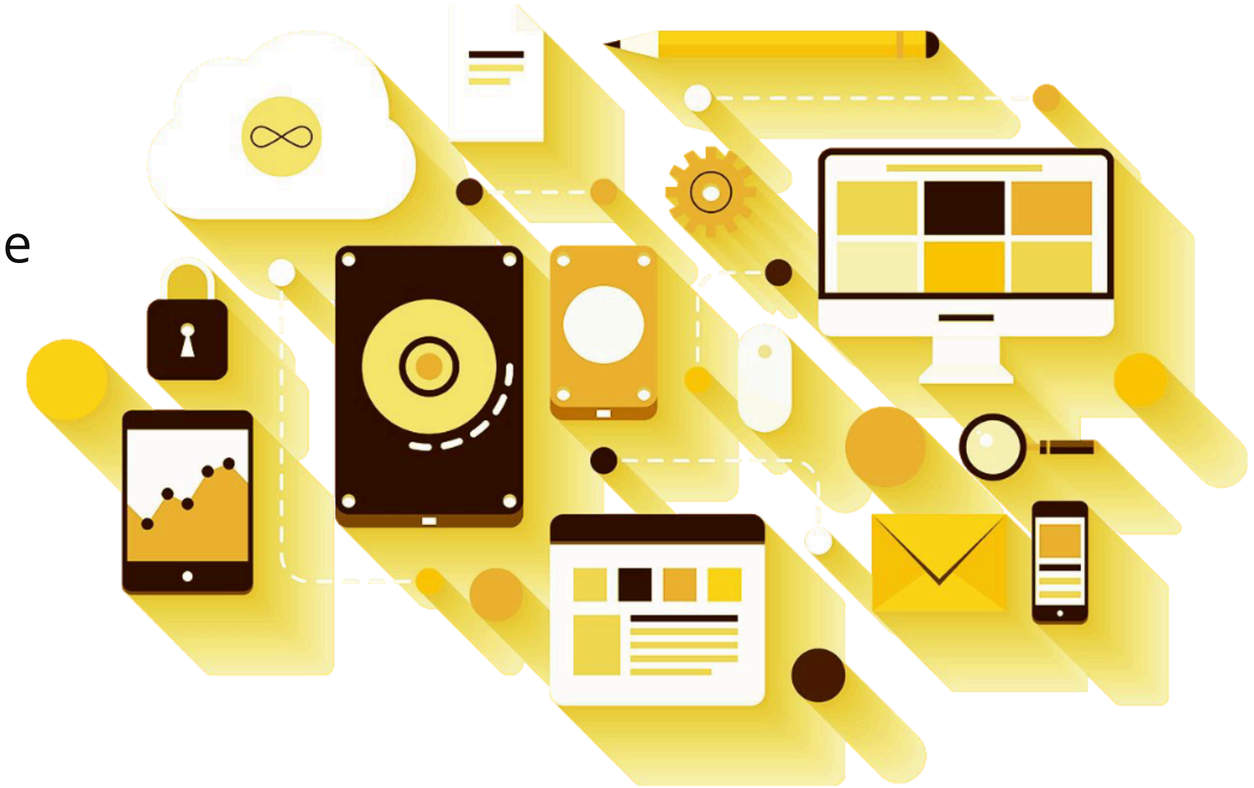
- Functional Tests;
- Interoperability Testing (Interaction Testing).

Functional tests

Functional tests are based on the functions performed by the system and can be conducted at all levels of testing (component, integration, system, acceptance). Typically, these functions are described in requirements, functional specifications, or as use cases.

Interoperability Testing

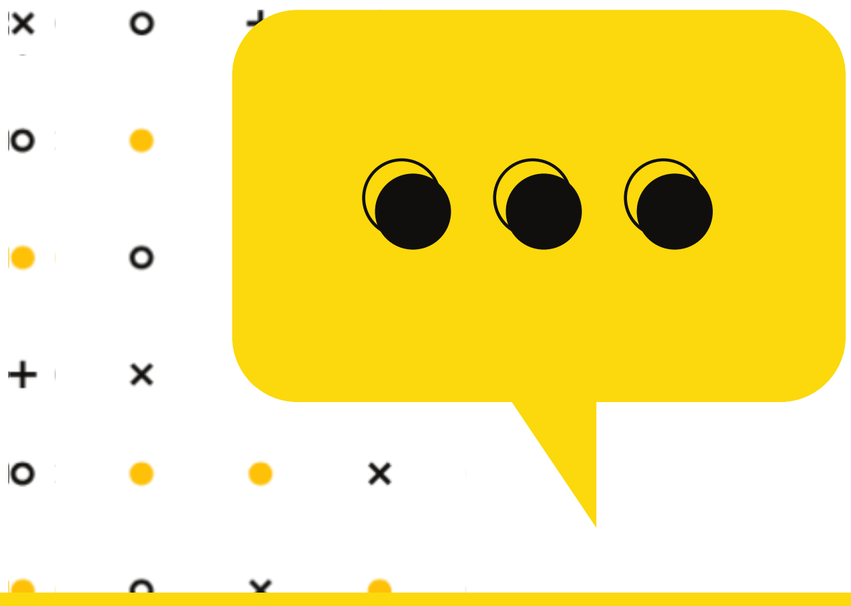
Interoperability Testing (Interaction Testing) is functional testing that verifies the application's ability to interact with one or more components or systems, including compatibility testing and integration testing.



Non-functional types of testing:

Non-functional testing it covers all other requirements not addressed by functional requirements. It describes **HOW** the system operates and addresses questions like:

- **HOW fast** does the application perform?
- **HOW secure** is it?
- **WHAT** is the appearance of the application?



Non-functional types:

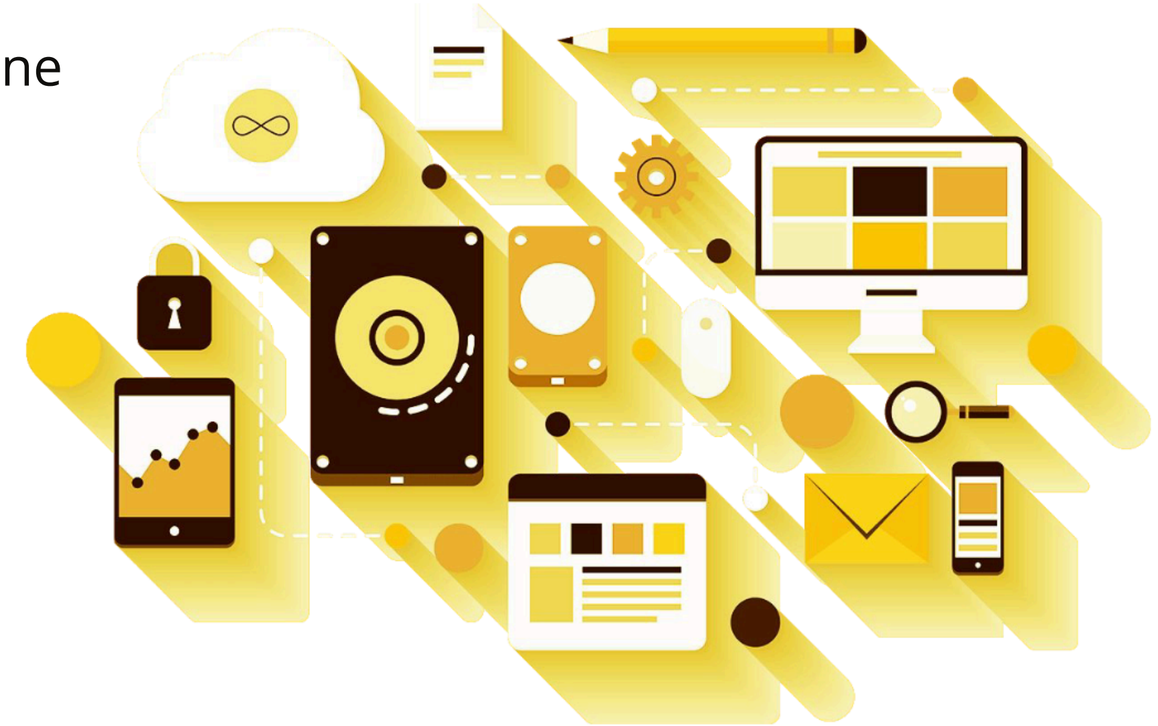
- Performance Testing;
- Installation Testing;
- Usability testing;
- Failover and Recovery Testing;
- Configuration testing;
- Localization Testing;
- Internationalization Testing;
- UI/UX Testing
- Load Testing;
- Security testing;
- Reliability Testing;
- Accessibility Testing.

Performance testing

The goal of performance testing is to determine the scalability of the application under load.

Types include:

- Stress testing;
- Volume testing;
- Stability / Reliability Testing;
- Load testing.

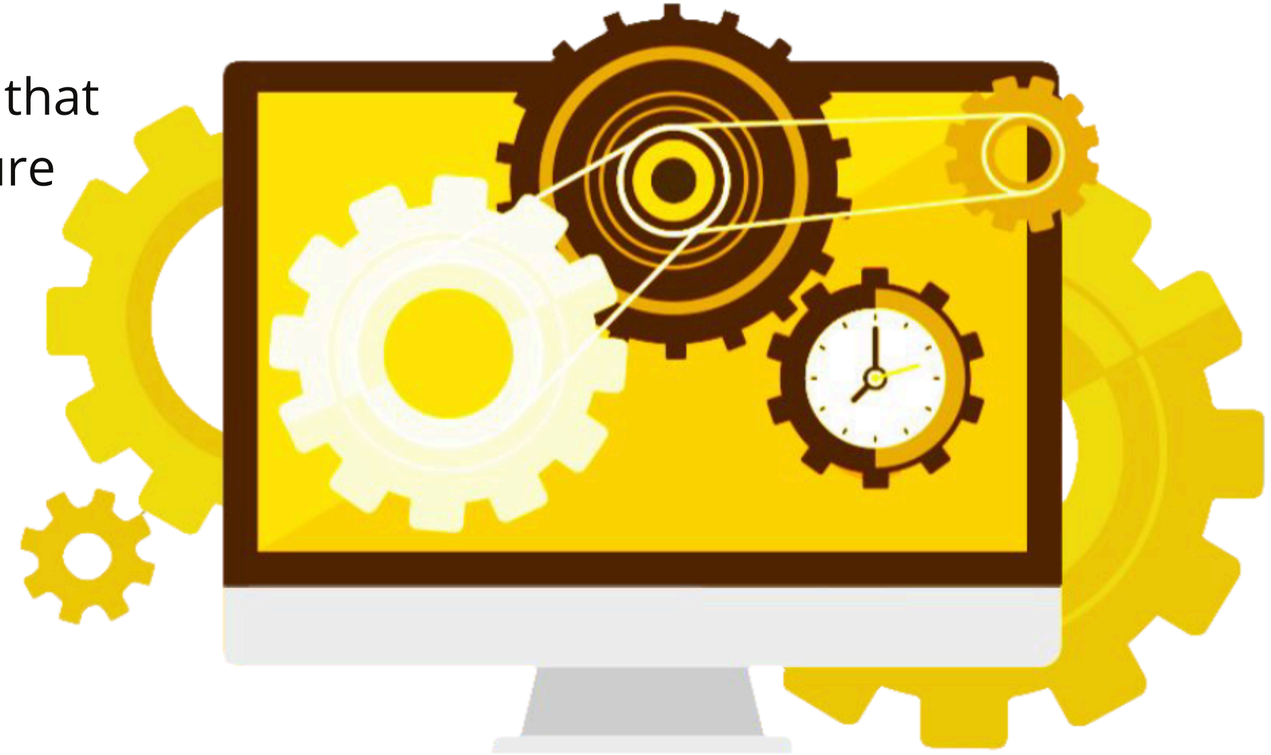


Installation testing

Installation testing focuses on the actions that users need to perform to install and configure software.

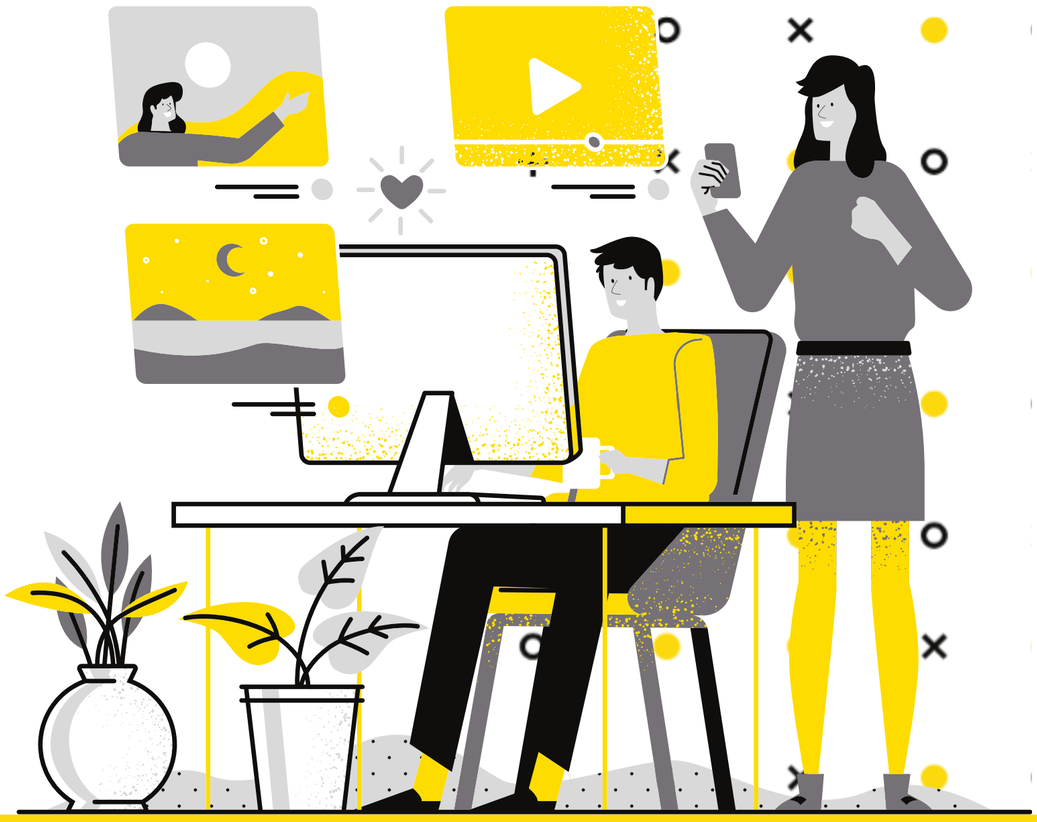
It may include:

- full or partial installation;
- installation/removal modification.



Usability Testing

Usability testing is a testing method aimed at establishing the degree of usability, learnability, understandability, and attractiveness for users of the product in some particular conditions.



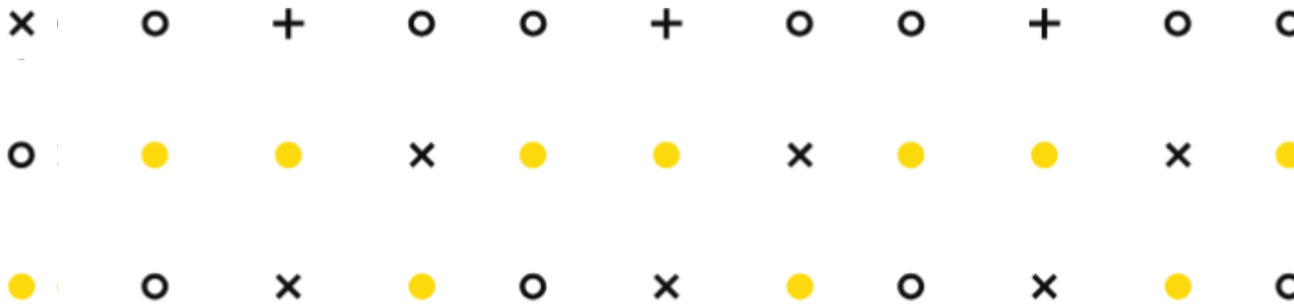
Failover and Recovery Testing

Failover and Recovery Testing validates the product under test for its ability to withstand and successfully recover from possible failures due to software errors, hardware failures, or communication problems (such as a network failure).

The purpose of this type of testing is to check recovery systems (or duplicate the main functionality of systems), which, in the event of a failure, will ensure the safety and integrity of the data of the tested product.

Recovery testing example:

- If the application is receiving data from the network, unplug the connection cable.
- After a while, reconnect the cable and analyse the ability of the application to continue receiving data from the point where the network connection was broken.
- Reboot the system while the browser has a certain number of sessions open and check if the browser can restore them all or not.

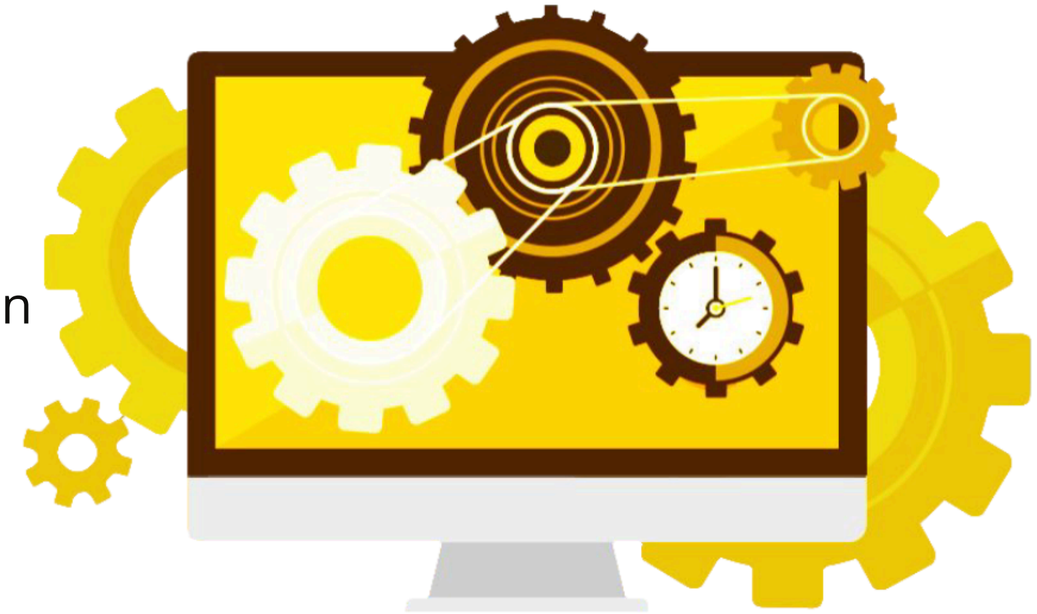


Configuration testing

Configuration Testing is a special type of testing aimed at checking the operation of software under various system configurations (declared platforms, supported drivers, various computer configurations, etc.).

Depending on the project type, configuration testing can have different goals:

- System profiling project.
- A project to migrate a system from one platform to another.



Security testing

A testing strategy is used to test the security of a system, as well as to analyze the risks associated with providing a holistic approach to protecting an application, attacks by hackers, viruses, and unauthorized access to confidential data.

The overall security strategy is based on three main principles:

- confidentiality;
- integrity;
- availability.

Change-related testing

These types of testing are conducted after fixing the defects/errors, identified during the testing process.

The main task is to confirm the resolution of the problem.

Main types:

- **Re-testing -** checks whether a defect has been fixed or not.
- **Regression testing -** verifies if a code change has impacted other functionalities of the product.
- **Smoke Testing -** checks critical functions and system stability prior to more thorough testing.
- **Sanity testing -** validates the functionality of a specific feature.

Static and Dynamic Testing

Based on the criterion of program execution (whether the software code is executed), two types of testing are distinguished:

Static testing is a type of testing that assumes the software code **will not be executed** during testing.
Several types are distinguished, such as:

- Source code review;
- Requirements verification.

Dynamic testing is a type of testing that **involves the execution** of software code.
Thus, the behaviour of the program during its operation is analysed.
It includes various subtypes, each of which depends on:

- Access to the code (Black\White\Grey box testing);
- Level of testing (system, acceptance testing);
- Application usage areas (functional, load testing).

Manual and Automated Testing

During **manual testing,** execution of test cases is performed manually.

Automated testing involves the use of specialized software (in addition to the one being tested) to control test execution and compare the expected and actual results of the program's operation.

There are several types of automated testing:

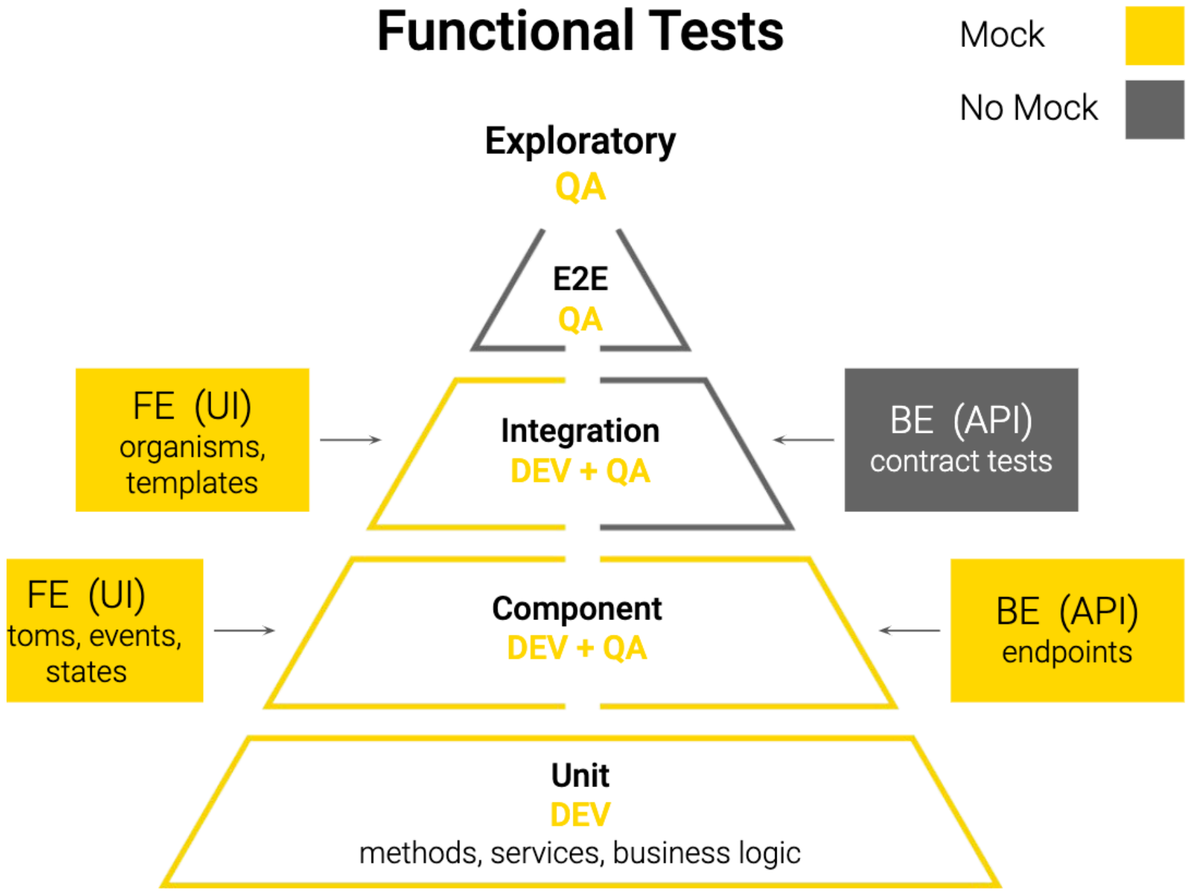
- Code-driven testing;
- Graphical user interface testing;
- API (Application Programming Interface) testing.

Testing levels

Testing at different levels is conducted throughout the entire software development and maintenance lifecycle. The testing level determines what is being tested: individual modules, groups of modules, or the system as a whole. Conducting testing at all levels of the system is the key to successful project implementation and delivery.

Classification according to ISTQB:

- Unit testing;
- Integration testing;
- System testing;
- Acceptance testing.



Object testing

Let's consider the application and sequence of testing types using the example of testing a pencil.
So, the task is to test the pencil.

Algorithm:

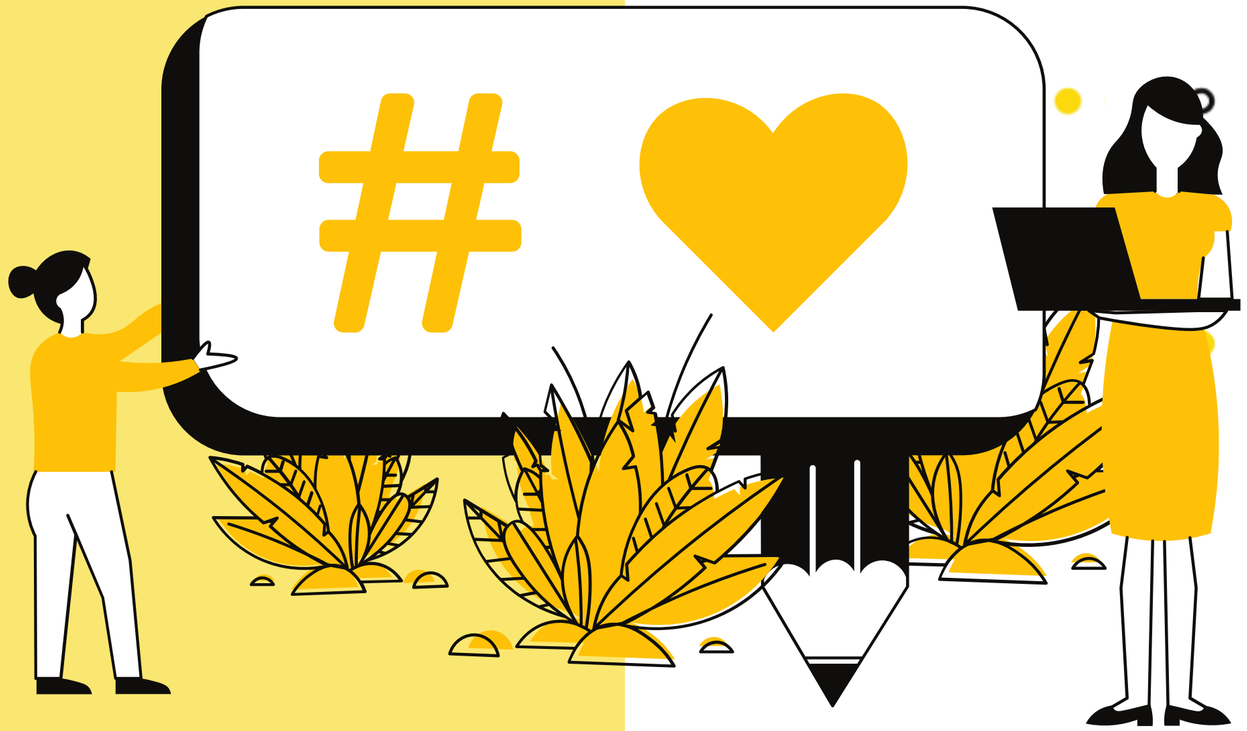
1. ALWAYS start with **positive functional tests**. Check if our object performs its basic functions.
2. Then move on to **non-functional testing**:
    - Check appearance (dimensions, colour) for compliance with requirements;
    - Usability;
    - Load testing;
    - Performance under different conditions;
    - Compatibility with other items;
    - Safety.
3. Finish testing with **negative checks** (crushing, breaking, tapping, bending, etc.)

Example of mind-map

PENCIL

- **Black-box Testing**
  - We don't know how it's built
  - Proceeding blindly
- **Beta Testing**
  - Give it to friends, and let them test it
- **Automated Testing**
  - Build a robot, teach it to write
  - Automatic sharpener
- **Functional Testing**
  - Does it write?
    - Can what's written be erased?
    - What if pressure is applied while writing?
  - Does it need sharpening?
    - From both sides?
  - Will we store or carry it?
    - Is it environmentally friendly?
- **Security Testing**
  - Will it stain if we don't sharpen it?
  - Will we get poisoned if we eat it?
- **Usability Testing**
  - Is it comfortable to write with?
  - Is it easy to sharpen?
  - Does it hold behind the ear?
- **Negative Testing**
  - To chew / To stick it into a socket
- **Compatibility Testing**
  - Writing surfaces
    - On paper / On a fence / But on glass?
  - Sharpening tools
    - Sharpener / knife / razor?
  - Storage spaces
    - Pencil case / cup / box of pencils
- **Stress Testing**
  - To drop / To tap on the floor / To bend
- **Performance Testing**
  - How much can be written?
  - How many times can it be sharpened?
  - What is the expiration date?
- **Documentation Testing**
  - Everything good with instruction of use?
- **Integration Testing**
  - Writing separately with the lead
  - Sharpening the wooden base separately
- **UI Testing**
  - What color?
  - Have pictures?
- **Component Testing**
  - Assembled, the lead doesn't fall out?
- **Enrty Point Testing**
  - New or already sharpened?

04 Kahoot game



The start of the Kahoot game

1. Follow the link **https://kahoot.it/** from your mobile phone or computer.
2. Enter a game code.

05 Questions?