**Level 1**

| **Testing types** | **What to check** | **When to use** | **Restrictions** | **Features** |
| --- | --- | --- | --- | --- |
| **Functional** | Testing of main functional activities of the product according to requirements (why product was created). Workflow of separate functions and their connections between each other and/or other objects. Insurance that expected functionality of product will be the same as actual | During the whole STLC. Starting from unit testing and finishing with GUI testing we’re always focusing on the functional base that is working as required. | * interaction of functionality - complication of testing due to the high amount of possible integrations which must be taken into account: unit-unit; unit-system; system-system. | * described at requirements stage * splitting possibility (tests could be done as for single module and for system in total) |
| **Non-functional** | Parameters which are not directly connected to functional aspects of the product but could affect performance: view, stability, security, usability, etc. Non-functional testing has the same importance as functional one, because the end user can deny the product due to bad usability, even if functionality is perfect. | When the main functional side of the product is ready. We need to see the whole system “on one table” for effective non-functional testing. | * most non-functional tests could be done only on final stages of production. * no possibility to satisfy all “tastes” - even if all requirements was kept there is chance of claims from end user based on their preference | * described at requirements stage * defects could be fixed with less impact compared with functional ones at the same production stage. |
| **Due to changes** | Testing of product when functionality is added/changed; esurance that previous functionality was not affected and working correctly (regression); defected function was fixed without impact to other segments. | Product function change (adding/removing/updating) or defect fixes, and, due to this, control that rest functionality is not affected and stable. | * high consumption of time and resources | * general performance review * defects solution control |

**Regression testing -** type of testing which occurs according to a settled timeline (for example after each sprint) or due to functionality changes of our product (adding/deleting/updating) with main scope - check if these changes are not affecting our rest functions in the system.

**Re-testing** - insurance that functionality, where defect was found, reported and after confirmed back that it’s fixed, is working according to expectations.

**Level 2**

*Як ти вважаєш, чи можливе для продукту проведення тільки функціонального тестування, без перевірки нефункціональних вимог?*

Так, на мою думку, є випадки коли є можливість тільки для функціонального тестування, наприклад:

* продукти для людей з обмеженими можливостями (коли функціонал є пріоритетним, а, наприклад, зовнішній вигляд - другорядним або взагалі не потрібним).
* продукти, розробка яких обмежена в часі і фінансуванні (зазвичай такі продукти націлені саме на функціональну складову, а performance є дуже слабким) - сюди можна віднести деякі стартап проекти, де на ринок спочатку подається функціональність, а не функціональна складова вдосконалюється згодом.
* так вимагає замовник - тестування нефункціональних вимог відбувається, наприклад, іншою стороною або end users.

*Як ти розумієш необхідність проведення smoke (димового) тестування? Чи завжди воно є доречним?*

**Smoke testing -** це тип тестування, який має на меті поверхнево перевірити працездатність основного функціоналу продукту (чи можна інсталювати продукт, чи можна залогінитися і тд.). Даний тип тестування є вкрай важливим перед будь яким іншим, більш детальним тестуванням, адже немає смислу тестувати додаток, якщо його не можна інсталювати. Smoke тестування є доречним тільки для готового функціоналу, щоб упевнитися що він працює і відповідає вимогам. На стадії розробки функціоналу даний вид тестування є не доречним.

**Level 3**

[**5 basic Functional test-cases for start-up (Kitty application)**](https://docs.google.com/spreadsheets/d/1ZkrnF8Aai5X3TphspC1LpCRLIFl96SWujFZjtIgGisI/edit?usp=sharing)

**Non- functional testing which could be used for current start-up:**

* **Security -** we need to be sure that each post contains only pictures with cats, without any different objects/subjects which could break our requirements. Our AI is checking it on stage of uploading, but it could miss cases when, for example, cats are available with different objects on the same picture or some object or subject “simulates” as a cat.

**Options/check ins:**

Secondary security step like “Reporting” could catch such issues and solve them according to application rules.

User can choose the “problem” picture and click on “Report” button to inform system about divergence.

* **Installation and Compatibility testing -** we’re offering our product at 2 main OS - IOS and Android, so first of all we need to define and confirm which devices, OS versions our product will support.

**Options/check ins:**

* Installation is starting, processing and successfully finishing at the chosen OS/device.
* Application is starting at the chosen OS/device.
* All application functionality is working at the chosen OS/device.
* Cross-action activities are possible between different OS/devices (friends adding/commenting/likes, etc.)
* **Usability testing** - our start-up is an entertaining application. We must be interesting for end users, make them satisfied and do not overload with complications.

**Options/check ins:**

* simple (clear) understanding which button is doing which function.
* colors conception - classic theme or special
* be different, be special - button/text highlights, animations (for example during loading), sounds.