**Перший рівень**

| **Вид тестування** |  | **Що проверяется** | **Коли застосовуется** | **Обмеження** | **Особливості** |
| --- | --- | --- | --- | --- | --- |
| Функціональне | Функціональне тестування | Функціональності | На ранніх етапах розробки | Початкові етапи розробки | Розглядає задану поведінку |
| Тестування GUI | Кольори, шрифти, розміри | В кінці розробки | Можливо тестувати тільки коли є интерфейс, а це не відразу | |
| Тестування безпеки | Захищеність системи | Періодично на всіх етапах розробки | Проводиться після підключення до зовнішніх сервісів | Це стратегія яку треба періодично й ретельно пройти |
| Тестування взаємодії | Здатність додатку діяти з компонентами системи | Після функціонального тестування | Можливо тільки при наявності великого і цілісного шматку системи | Охоплює також тестування на сумісність та інтеграційне тестування |
| Нефункціональне | Тестування продуктивності | Перевірка усього додатку на витривалість, навантаження | Коли система зібрана | На початку розробки не можна проводити | Проводиться на готовій системі |
| Тестування встановлення | Перевірка встановлення | На фіналі розробки | Може проводитись коли система зібрана до купи |  |
| Тестування зручності використання | Тестується зручність та зрозумілість користувачеві | На фіналі розробки |  | Тестується увесь інтерфейс на всі можливі перевірки що стосуються зручності та зрозумілості використання |
| Тестування на відмову та відновлення | Тестування системи на збої та відновлення після них | Коли система повністю зібрана | Коли система не підключина до зовнішніх джерел це важко зробити | Тестування різних варіантів помилок, а також зовнішніх перешкод в роботі додатку |
| Конфігураційне тестування | Різні конфігурації системи | На фіналі розробки | Поки система не зібрана та не встановлена в середовище роботи не можлива перевірка | Може поєднуватись з іншими видами тестування |
| Пов'язане зі змінами | Димове тестування | Перевірка основних функцій | Після кожної нової збірки | Не глибоке тестування |  |
| Регресійне тестування | Перевірка всіх функцій | Після змін в коді або змін в оточенні |  | Перевіряє усі функції після змін |
| Re-test | Виправлення багів | Після виправлення багу |  | Перевіряє тільки те чи виправлений баг чи ні |
| Тестування збірки | Визначає чи нова збірка відповідає вимогам | У кінці розробки |  | Проводиться практично після випуску фінальної версії і перед альфа тестуванням |
| Sanity тест | Перевіряється одна визначена функція | Після змін у коді або оточенні |  | Перевірка направлена на співвідношення функції та специфікації |

2. Різниця між Регресивним тестуванням та Ре-тестом хоч вони і схожу на перший погляд все ж таки є і в першу чергу у тому що Ре-тест перевіряє що баг виправлений, а от регрессивные тестування перевіряє чи не вплинуло виправлення бага на весь старий функціонал. Також різниця у тому що Ре-тест не підходить для автоматизації тестування коли ж Регресійне тестування практично найгарніший варіант для цього, через велику кількість тестів які потрібно пройти. Також відмінність у тому що Регресійне тестування проводиться на будь якому обладнанні визначеному у вимогах також у будь якій версії, а от Ре-тест проводиться лише у версії в якій саме виправлений баг і саме на тому обладнанні на якому було виявлено баг.

**Другий рівень.**

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

* Я вважаю що це можливо, на початкових етапах розробки щоб перевірити чи відповідає система та її основні функції вимогама, також щоб пересвідчитись чи система взагалі робить те що від неї хоче замовник чи ні. Саме функціональних тестів достатньо для визначення чи працює чи ні модуль.

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

Димове тестування направлено на те щоб перевірити чи працюють основні функції чи ні після виходу нової збірки програми. Тому воно завжди доречно і допомагає одразу після отримання тестувальником нової збірки визначити чи є сенс її тестувати більш поглиблено чи варто повернути на доробку розробникам цю версію продукту. Тож димове тестування полягає у тому щоб швидко проклацати основні функції і якщо щось не працює повернути на доопрацювання.

**Третій рівень**:

Тест-кейси

| Завантаження фото | Відкрити додаток на Андроїд | 1. Натиснути кнопку завантажити |  | Переход на сторінку завантаження |
| --- | --- | --- | --- | --- |
| 2. Натиснути в полі завантаження |  | Відкривається галерея телефону |
| 3. Обрати фото |  | З'являться строка очікування завантаження а потім повідомленні про завантаження |
| Залишити вподобайку |  | 1. Відкрити додаток на Андроид |  | З'явиться стрічка з фото |
| 2. Натиснути двічі на будь яке фото |  | На фото з'явиться тіньове сердечко |
| Додати друга | Відкрити додаток на iPhone | 1. Внизу екрану натиснути кнопку мій профіль |  | Тебе перекине на твій профіль зі світлинами |
| 2. Натиснути кнопку "додати друга" |  | З'явиться список-пошук з зареєстрованими користувачами |
| 3. Через пошук оберт друга якого хочеш додати |  | Поруч з ім'ям користувача з'явиться галочка і з'явиться напис що друга додано |
| Завантажити фото з собачкою | Відкрити додаток на Андроїд | 1. Натиснути кнопку завантажити |  | Перехід на сторінку завантаження |
| 2. Натиснути в полі завантаження |  | Відкривається галерея телефону |
| 3. Обрати не валідне фото(собачки) |  | Висвічується повідомлення про помилку. |
| Перевірити чи зберігаються дані користувача у базі даних |  | 1. Відкрити додаток та вибрати створити нового користувача |  | Відкриється форма для реєстрації |
| 2. Ввести валідні данні | Пароль 1237&влад  Логін malizak | Усі поля приймають дані |
| 3. Натиснути кнопку зберегти |  | Підсвітеться привітання з реєстрацією |
| 4. Перейти до бази даних та подивитись чи збережені данні |  | Інформація збереглась і вірна |