**Завдання 1.**

**Функціональні види тестування:**

**Функціональне** **тестування** – перевіряє, як виконуються функції програми, як програма відповідає на дії користувача. Застосовується завжди, так як коректне виконання функцій системою є ключовим критерієм якості продукту.

**Тестування UI/UX (інтерфейсу)** – тестування взаємодії користувача з системою через інтерфейс. Оцінювання зовнішнього виду інтерфейсу програми, відповідність дизайну згідно заданим вимогам, та аналіз того, чи зручно взагалі користуватися таким інтерфейсом згідно користувацького досвіду самого тестувальника. Використовується постійно, так як візуальне сприйняття інтерфейсу продукту може впливати в цілому на його успіх серед користувачів.

**Тестування взаємодії (сумісності)** перевіряє здатність програми взаємодіяти з різними системами, тобто чи буде ПЗ правильно працювати на різних операційних системах, браузерах, різних розширеннях екранів і т. д.

**Нефункціональні види тестування:**

**Performance Testing –** тестування продуктивностіпроводиться для визначення наскільки швидко працює програма чи система при певній запланованій нагрузці (наприклад, кількість людей, які одночасно зайшли на сайт і т.д.).

**Stability/Reliability Testing** (Тестування стабільності та надійності) – різновид тестування продуктивності. Перевіряє наскільки стабільно буде працювати система при звичайній нагрузці, але при тривалій роботі програми.

**Load Testing** – різновид тестування продуктивності, але в цьому випадку тестувальник має перевірити наскільки швидко буде працювати система, якщо нагрузка на неї буде більша, ніж запланова.

**Stress Testing** також відноситься до тестування продуктивності системи, але в цьому випадку ми тестуємо систему при екстремальній нагрузці, та перевіряємо наскільки велику нагрузку може витримати продукт.

**Usability Testing (Тестування зручності)** – перекликається з тестуванням інтерфейсу, але тут пріоритет надається не візуальному оцінюванню інтерфейсу, а взагалі зручності користування програмою, інтерфейсом і т. д.

**Installation Testing (тестування установки) –** перевіряє наскільки коректно ПЗ встановлюється на девайс, наскільки коректно його можна видалити і що відбувається при цьому з даними.

**Security and Access Control Testing (Тестування безпеки)** – перевіряє, наскільки безпечно користуватися програмою, чи безпечно зберігаються конфіденційні дані користувача, наскільки система захищена від витіку даних, атак хакерів, вірусів і т.п.

**Volume Testing (Об’ємне тестування) –** також вид тестування продуктивності, але тут в якості нагрузки виступає великий об’єм даних.

**Тестування локалізації –** тестування системи, пов’язане з адаптацією продукту до вимог інших країн, наприклад, це можуть бути нюанси, пов’язані з мовою, особливостями інтерфейсу, забороненими функціями в інших країнах і т. п.

**Failover and Recovery Testing (Тестування на відмову та відновлення)** – тестування, метою якого є виявлення наскільки система стабільна до різних збоїв, пов’язаних з дефектами ПЗ, поломкою обладнання чи відключення від мереж. І наскільки швидко може система відновитися після збоїв, та чи збережуться дані після відновлення.

**Тестування, пов’язане зі змінами:**

**Smoke Testing (Димове тестування)** – тестування, яке проводиться для швидкого оцінювання основних критичних функцій нового продукту, щоб визначити чи варто проводити подальше детальне тестування, чи віддати продукт на доопрацювання. Таке раннє тестування економить час та матеріальні ресурси.

**Re-testing (повторне тестування)** – проводиться для того, щоб переконатися, що дефект, який раніше був знайдений і відправлений на виправлення розробникам, був дійсно усунений.

**Regression Testing** - вид тестування, який проводиться постійно для з’ясування чи виправлені дефекти або якісь додані зміни не спричинили нові зміни у вже працюючому функціоналі.

**Sanity Testing (Санітарне тестування)** – це більш детальне поглиблене тестування певної функції, яка дала збій за певних умов в стабільно працюючій програмі. Санітарне тестування проводиться, як правило, коли є скарга на продукт з боку користувача, тому таку проблему треба вирішити якомога швидше, щоб не понести репутаційних втрат.

**Завдання 2.**

Повторне тестування (Retesting) проводиться для того, щоб перевірити, що знайдений дефект у програмі був виправлений розробником та більше не проявляється у даному компоненті. Тобто тестувальник просто має впевнитися, що дефект був виправлений. На відміну від повторного тестування Regressing Testing ставить собі масштабнішу задачу - перевірити чи виправлений баг або будь-яке внесення змін в код не спричинили за собою інші зміни вже існуючого і нормально працюючого функціоналу програми, який ми вже раніше протестували.

**Завдання 3.**

Я вважаю, що для коректної роботи програми, щоб продукт був якісний та відповідав потребам замовника і користувачів, потрібно намагатися виділяти час і на функціональне, і на нефункціональне тестування. Але тестувальники працюють в різних проектах, умови яких можуть відрізнятися, може бути брак часу та ресурсів, тому потрібно надавати пріоритет функціональному тестуванню, яке перевірить функції програми та візуальне наповнення, а вже якщо вистачає часу, то провести і нефункціональне тестування, хоча б основні його види, наприклад, на безпеку, продуктивність і т. д.

**Завдання 4.**

Димове тестування проводиться, коли ми отримуємо нову версію продукту і нам потрібно оцінити критичні функції програми. Наприклад, при димовому тестуванні сайту інтернет-магазину ми перевіримо чи можна користувачеві зареєструватися. Якщо, наприклад, в цій функції буде знайдено дефект, то програму віддають на доопрацювання, так як немає сенсу заглиблюватися в більш детальне тестування, якщо ключові функції системи не працюють. Тобто задача димового тестування оцінити ключові функції системи на ранньому етапі. Я вважаю, що димове тестування доцільно проводити завжди, бо це не займає багато часу, але врешті економить час та інші ресурси.