**Звіт**

Виконання робіт з тестування, сертифікація та аудит

Київ 2023

**ЗМІСТ**

1. План тестування
2. Тести для кожної вимоги ТЗ та елементу проекту
3. Сертифікація
4. Аудит

**1. Види тестування**

***1.1 Функціональне тестування***

Функціональне тестування зазвичай включає наступні етапи:

1.Підготовка тестових сценаріїв. Тестувачі створюють тестові сценарії, які відображають очікувану поведінку програмного забезпечення. Тестові сценарії можуть бути створені на основі функціональних вимог, специфікацій, або на основі досвіду використання аналогічного програмного забезпечення.

2. Виконання тестових сценаріїв. Тестувачі виконують тестові сценарії на програмному забезпеченні, щоб перевірити, чи відбувається очікувана поведінка.

3. Реєстрація результатів. Після виконання тестових сценаріїв, тестувачі реєструють результати у спеціальній системі тестування. Результати можуть бути представлені у вигляді звіту або списку помилок.

4. Аналіз результатів. Після виконання тестових сценаріїв тестувачі аналізують результати тестування, щоб виявити помилки або проблеми в програмному забезпеченні.

5. Виправлення помилок. Після виявлення помилок тестувачі повідомляють розробникам програмного забезпечення про проблему, після чого розробники виправляють помилки та випускають нову версію програмного забезпечення.

***Повторне тестування. Після виправ***

* 1. ***Тестування зручності користування***

Тестування зручності користування дає оцінку рівня зручності використання програми за наступними пунктами:

* продуктивність, ефективність (efficiency) — скільки часу і кроків знадобиться користувачеві для завершення основних завдань програми, наприклад, розміщення новини, реєстрації, покупки тощо (менше — краще);
* правильність (accuracy) — скільки помилок зробив користувач під час роботи з додатком (менше — краще);
* активізація в пам'яті (recall) — як багато користувач пам'ятає про роботу програми після припинення роботи з нею на тривалий період часу (повторне виконання операцій після перерви має проходити швидше ніж у нового користувача);
* емоційна реакція (emotional response) — як користувач почувається після завершення завдання — розгублений, знаходиться у стані стресу? Чи порекомендує користувач систему своїм друзям (позитивна реакція — краще)?
  1. ***Тестування продуктивності***

В основному це тестування навантаження. Тестування навантаження програми перевіряється в більшості випадків автоматом, тобто спеціальними програмами. Це дає шанс перевірити, наскільки вона буде працювати під певним навантаженням.

Мета цього тестування, полягає в кількості віртуальних користувачів, які задають n кількість запитів, в один час (будь це секунди навіть). Тим самим результат дає те, чи зміг наш проект витримати, наприклад, 100 користувачів, які одночасно замовляють товар або авторизувалися в програмі , відповідь показує, чи реально витримати програмі таке навантаження.

* 1. ***Тестування інтерфейсу користувача***

UI testing- це тестування графічного інтерфейсу користувача, яка передбачає перевірити програму на відповідність вимогам до графічного інтерфейсу, професійного вигляду, виконання в єдиному стилі.

У більшості випадків, тестування інтерфейсу користувача, здійснюється разом з наступними видами тестування (UI):

1. Тестування на відповідність стандартам графічних інтерфейсів.
2. Тестування з різними дозволами екрану
3. Тестування локаційних версій: точність перекладу, перевірка довжини назв елементів інтерфейсів і т.д
4. Тестування графічного інтерфейсу користувача на цільових пристроях
   1. ***Тестування програми на уразливість***

Це ключ до надійності програм. Основні правила цього тестування – це перевірка на уразливість різних видів атак. Так як це програма “Рука допомоги”, то , слід перевіряти запити на Sql ін’єкцію( запити до баз даних).

SQL-ін’єкцію

XSS ін'єкція. Дає можливість впровадити довільний код, і атакувати комп'ютер користувачів, які переглядають заражені сторінки.   
1. Перевірка можливості роботи клієнта після короткочасного зникнення з’єднання із сервером додатку

* Під час виконання певної операції, яка потребує безпосередньої взаємодії з сервером додатку
* Під час не виконання операцій, що вимагають безпосередньої взаємодії із сервером додатку

2. Перевірка можливості роботи потребуючого після перезавантаження сервера додатку

3. Перевірка можливості роботи клієнта після короткочасного зникнення з’єднання між сервером додатку та базою даних

4. Перевірка можливості роботи клієнта та сервера програми після перезавантаження бази даних

**Рівні критичності помилок**

1 рівень – програма не запускаєтся

2 рівень - не можливо здійснити реєстрацію;

3 не можливо увійти в систему, якщо пароль та логін вірний

4 рівень отримуєтся не вірна інформація про користувача

**Формат опису знайдених помилок**

Опис знайдених помилок (дії користувача, та інші помилки) буде відбуватись у вигляді текстового опису.

**План дії в залежності від критичності знайденої помилки**

Помилки у ПЗ можна класифікувати відповідно до їхнього розподілу за етапами життєвого циклу і джерел їхнього виникнення:

* 1) ненавмисне відхилення розробників від робочих стандартів або планів реалізації;
* 2) специфікації функціональних та інтерфейсних вимог без дотримання стандартів розроблення;
* 3) недосконала організація процесу розроблення.

Після знайдення помилок, потрібно занести до CQ :

* Вказати підсистему, в якій було знайдено помилку
* Рівень важливості
* Версія продукту
* Описати, знайдену(і) помилку
* Описати, після яких дій з’являється помилка
* Добавити скріншоти , які демонструють помилку

**Тестування**

Тестувальник виконує тестування в ролі корстувача. Він перевіряє наявність помилок, чи відповідає програма визначеним вимогам. Якщо виникають помилки або виявлені певні дефекти, тестувальник робить їх опис та повідомляє розробника для їх усунення.

**Unit-тести**

Unit-тести застосовувались до функції вікна реєстрації, вікна входу, вікна користувача

**2. Тести для кожної вимоги з ТЗ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Підсистема** | **Вимога** | **Спосіб тестування** | **Критерій проходження тесту** |
| 1 | Реєстрація | Ввести дані для реєстрації | Ввід даних у відповідні поля | З'являється вікно з написом «Ви зареєстровані» |
| 2 | Вхід в систему | Вхід в обліковий запис користувача | Ввід логіну та паролю | Вхід в обліковий запис.  Якщо ведено не вірний логін або пароль з’являється повідомлення про помилку. |
| 3 | Пошук за назвою необхідного товару | Вивід даних про товар | Ввести назву товару у відповідне поле та натиснути «Пошук» | Якщо товар існує з'явиться рядок інформації про нього, а якщо не існує - з'явиться відповідний текст |
| 4 | Корзина | Інформація про вибрані товари | Натиснути кнопку «Корзина» | З'явиться інформація про вибрані користувачем товари |
| 5 | Історія замовлень | Інформація про раніше зроблені замовлення | Натиснути кнопку «Історія» | З'явиться інформація про раніше зроблені замовлення |
| 6 | HELP | Відкриття інструкції до програми | Натиснути кнопку «HELP» | Відкривається інструкція до програми |

**3.Сертифікація**

* 1. ***Оцінка системи на захищеність від НСД.***

Проведена оцінка системи на захищеність від НСД і виконані наступні критерії оцінки:

* Гарантія безпеки
* Послуги безпеки
* Модель порушника
* Концепція диспетчера доступу
* Визначення несанкціонованого доступу
* Постановка проблеми захисту інформації в комп’ютерних системах від несанкціонованого доступу за основними напрямами захисту
* Основні принципи забезпечення захисту інформації: планування захисту і керування системою захисту
* Основні принципи керування доступом
* Основні принципи реалізації програмно-технічних засобів: функції і механізми захисту;
  1. ***Профіль захищеності відповідно до НД ТЗІ 2.5-004-99***

Розроблені критерії захищеності відповідно до НД ТЗІ 2.5-004-99:

* Критерії конфіденційності
* Критерії цілісності
* Критерії доступності
* Критерії гарантій

**4.Аудит**

З моменту виявлення першого багу, тестування програмного забезпечення пройшло великий шлях. Як всякий новий практичний напрям, воно динамічно розвивалося, не уникнувши невдалих спроб адаптації та перенесення методологій, стандартів і концепцій з вже існуючих областей. Додатковою особливістю цього процесу стала залежність тестування від власне програмного забезпечення, чиї технології, методи та інструменти самі переживають період стрімкого й інтенсивного вдосконалення

В середовищі розробки коду, який ми використовуємо є будовані механізми рефакторингу коду. Тобто таких проблем не виникає як:

1. [Код дублюється](https://uk.wikipedia.org/wiki/%D0%94%D1%83%D0%B1%D0%BB%D1%8E%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BA%D0%BE%D0%B4%D1%83)
2. [Підпрограма](https://uk.wikipedia.org/wiki/%D0%9F%D1%96%D0%B4%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0) занадто довга.
3. [Тіло циклу](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D0%BA%D0%BB_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) занадто довге або рівень вкладеності циклів занадто великий.
4. [Клас](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) має багато обов'язків, слабко пов'язаних між собою, що порушує [принцип єдиного обов'язку](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BD%D1%86%D0%B8%D0%BF_%D1%94%D0%B4%D0%B8%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BE%D0%B2%27%D1%8F%D0%B7%D0%BA%D1%83)
5. [Інтерфейс](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) класу не забезпечує достатній рівень абстракції.
6. Назва класу чи [методу](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) недостатньо точно відповідає його змісту.
7. Пов'язані дані, які використовуються разом, не організовані в клас.
8. Функція має занадто багато параметрів.
9. У ланцюжку виклику методів передається багато зайвих даних.
10. Потрібно одночасно змінювати кілька паралельних ієрархій класів.
11. Клас не виконує ніякої роботи самостійно, а тільки передоручає її іншим класам.
12. Клас має занадто багато відкритих (public) членів.
13. Нестатичний клас складається тільки з даних або тільки з методів.
14. Занадто широке застосування глобальних змінних.