Хмельницький національний університет

Кафедра комп’ютерних наук

**Лабораторна робота** **№7**

з предмету:

«Основи програмної інженерії та тестування програмного забезпечення»

Виконав:

студент групи КН-22-2

Бекешко Ю.Р.

Перевірив:

Собко О. В.

Хмельницький-2024

**Завдання:**

Завдання 1. Створити Checklist згідно вимог до програмного забезпечення відповідно до варіанту. Варіант виконання завдання призначається викладачем індивідуально.

Завдання 2. Згідно створеного Checklist створити Test Case Suite з 5 Test Case для програмного застосунку з використанням одного із наведених інструментальних засобів створення Test Case.

Завдання 3. Оформити звіт та завантажити його в GitHub

**Виконання завдання:**

Було створено Checklist згідно вимог до програмного забезпечення «Оцінка енергетичної цінності продуктів за фотозображенням нейромережевими засобами». Нижче наведено приклад розробленого Checklist для тестування застосунка:

1. Загальні тести.

Програма запускається та працює коректно.

Програма має зручний і інтуїтивно зрозумілий інтерфейс.

Програма коректно обробляє помилки та винятки.

1. Тести функціональності

Програма дозволяє користувачу завантажувати фото продукту.

Програма коректно розпізнає продукт за фото.

Програма обчислює енергетичну цінність продуктів на 100 грамів.

Програма обчислює енергетичну цінність продуктів на обрану користувачем кількість грамів.

1. Тести безпеки

Програма захищає особисті дані користувачів.

Програма захищає від несанкціонованого доступу.

Програма не дозволяє завантажувати шкідливі файли.

1. Тести продуктивності

Програма працює швидко та без затримок при завантаженні фото та оцінці енергетичної цінності продуктів.

Програма не споживає надто багато ресурсів комп'ютера.

Програма коректно працює з великими обсягами даних (наприклад, фото високої роздільної здатності).

Час обробки фото та оцінки енергетичної цінності продукту відповідає встановленим нормам (не більше декількох секунд).

Згідно створеного Checklist також було створено Test Case Suite з 5 Test Case для програмного застосунку з використанням одного із наведених інструментальних засобів створення Test Case.

Перший тест-кейс виконує завантаження зображення продукту на форму (таблиця 1.1).

Таблиця 1.1 – Тест-кейс TC001

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейс ID**: TC001 | **Пріоритет:** 1 | | **Створено**:28.05.2024, Бекешко Ю.Р. |
| **Назва:** Перевірка завантаження зображення на форму.  **Вхідні дані:** Користувач увійшов в програму. | | | |
| **Кроки** | | **Очікуваний результат** | |
| 1. Запустити програму 2. Натиснути на кнопку «Завантажити фото». 3. З’являється вікно для вибору зображення. 4. Натиснути кнопку «Відкрити». 5. Система виводить обране фото на форму. | | Обране зображення продукту буде виведено на екран. | |
| **Результат виконання тест-кейсу:** пройдено успішно | | | |

Після запуску застосунку відкривається вікно для оцінки енергетичної цінності продуктів. Для цього спочатку потрібно завантажити зображення натиснувши на кнопку «Завантажити фото», що продемонстровано на рисунку 1.1. Після чого з’явиться вікно для вибору зображення, потрібно натиснути на фото та натиснути «Відкрити» (рисунок 1.2). Після цього на екран буде виведено обране фото, що можна побачити на рисунку 1.3.

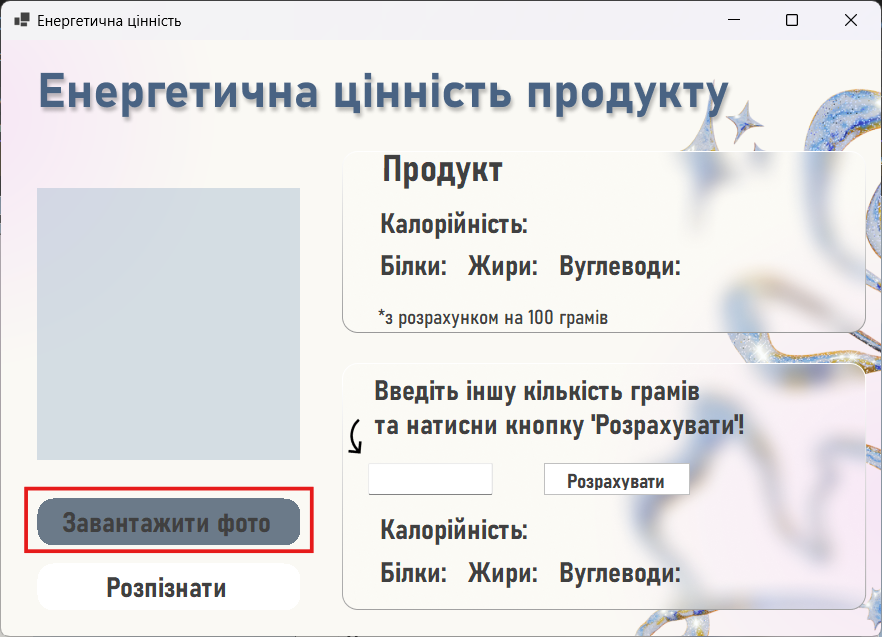


Рисунок 1.1– Демонстрація кнопки для завантаження фото

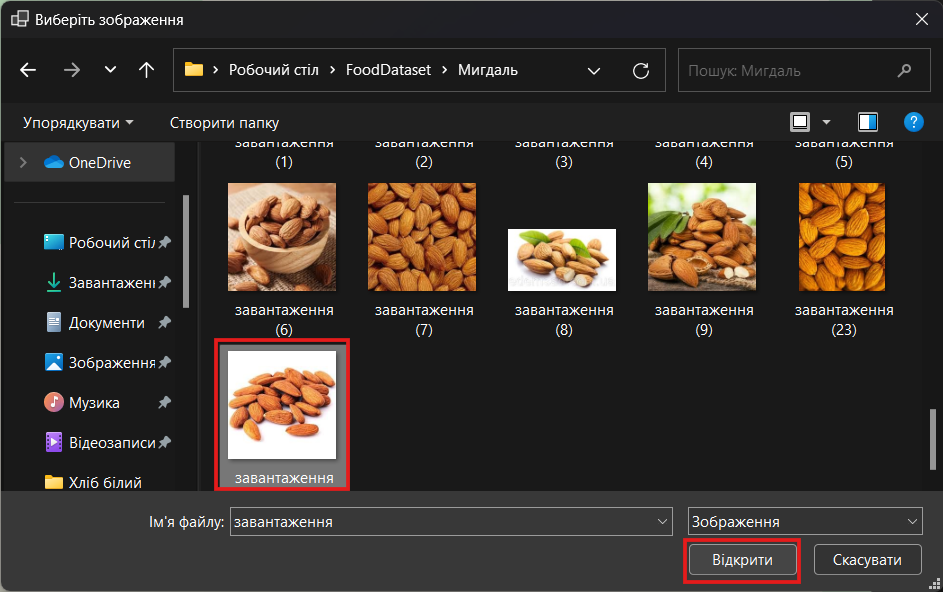


Рисунок 1.2 – Демонстрація вікна вибору зображення

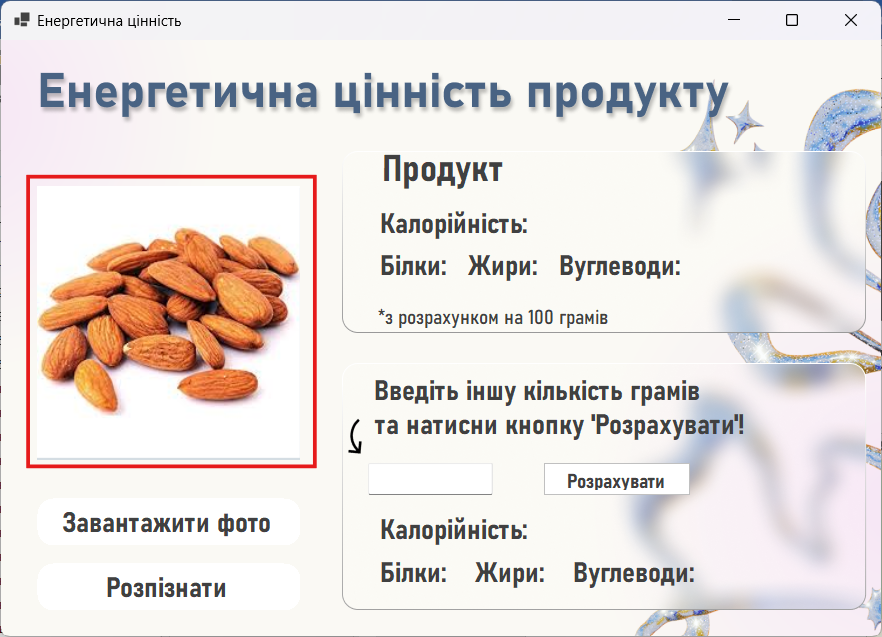


Рисунок 1.3 – Демонстрація виводу фото на форму

Другий тест-кейс демонструє розпізнавання продукту та оцінку його енергетичної цінності з розрахунком на 100 грамів. Тест-кейс представлено у таблиці 1.2.

Таблиця 1.2 – Тест-кейс TC002

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейс ID**: TC002 | **Пріоритет:** 2 | | **Створено**:28.05.2024, Бекешко Ю.Р. |
| **Назва:** Перевірка розпізнавання продукті на фото та оцінка його енергетичної цінності.  **Вхідні дані:** Користувач обрав та завантажив фото продукту. | | | |
| **Кроки** | | **Очікуваний результат** | |
| 1. Натиснути на кнопку «Розпізнати». 2. Система виводить на екран назву продукту. 3. Система виводить енергетичну цінність розпізнаного продукту. | | Назва продукту, його калорійність та БЖВ виведені на екран. | |
| **Результат виконання тест-кейсу:** пройдено успішно | | | |

Після того, як зображення успішно завантажено, можна розпочати розпізнавання, натиснувши на кнопку «Розпізнати», що продемонстровано но рисунку 1.4. Далі назва розпізнаного продукту та його енергетична цінність буде виведена на екран (рисунок 1.5).

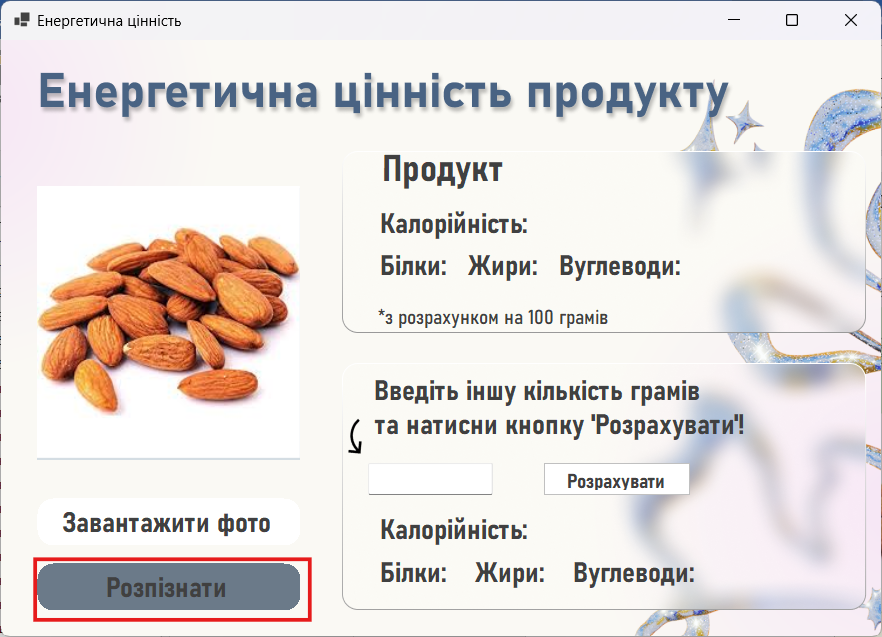


Рисунок 1.4 – Демонстрація кнопки для розпізнавання

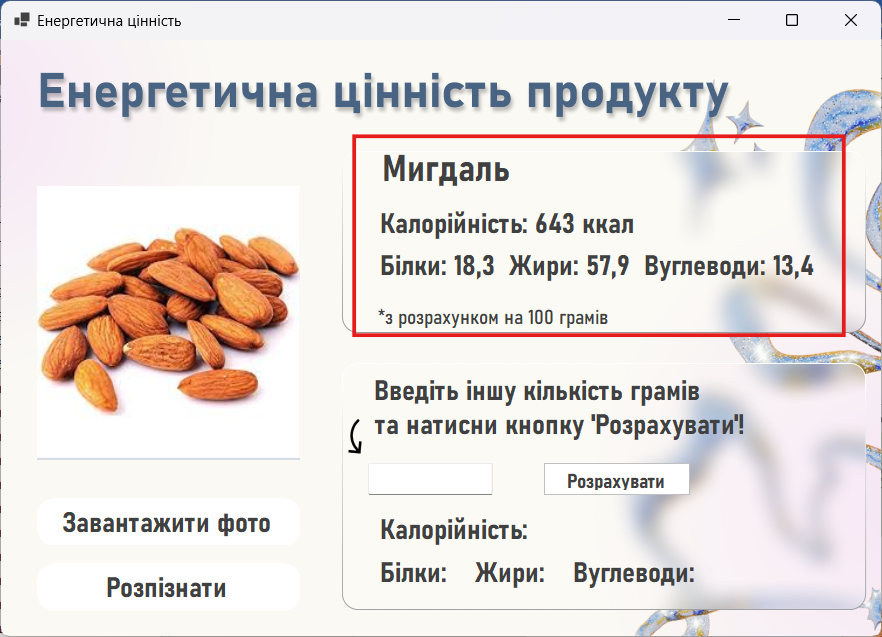


Рисунок 1.5 – Демонстрація результату

Третій тест-кейс демонструє розрахунок енергетичної цінності продукту на обрану користувачем кількість грамів. Тест-кейс представлено у таблиці 1.3.

Таблиця 1.3 – Тест-кейс TC003

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейс ID**: TC003 | **Пріоритет:** 3 | | **Створено**:28.05.2024, Бекешко Ю.Р. |
| **Назва:** Перевірка розрахунку енергетичної цінності розпізнаного продукту на введену кількості грамів.  **Вхідні дані:** Фото продукту обраного користувачем було успішно розпізнано. | | | |
| **Кроки** | | **Очікуваний результат** | |
| 1. Ввести кількість грамів у відповідне поле. 2. Натиснути кнопку «Розрахувати». 3. Система виводить результат розрахунків на екран. | | Розрахунок енергетичної цінності продукту на обрану кількість грамів. | |
| **Результат виконання тест-кейсу:** пройдено успішно | | | |

Після успішного завантаження та розпізнавання зображення, є можливість розрахувати енергетичну цінність продукту за бажаною кількістю грамів. Для цього потрібно ввести числом кількість грамів у відповідне поле та натиснути кнопку «Розрахувати», що продемонстровано на рисунку 1.6. Після чого результат має бути виведено на екран (рисунок 1.7).

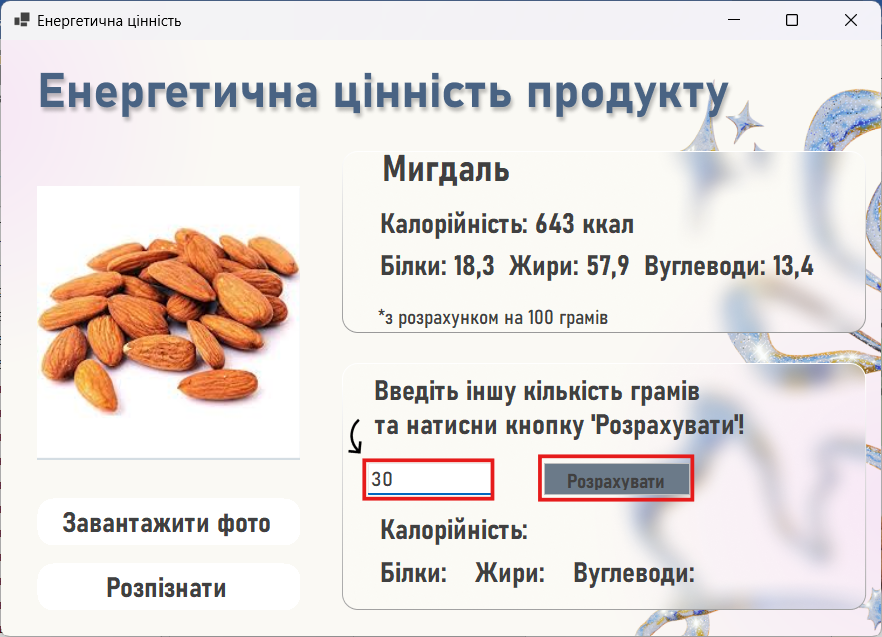


Рисунок 1.6 – Демонстрація процесу для розрахунку



Рисунок 1.7 – Демонстрація результатів розрахунку

Четвертий тест-кейс демонструє виведення повідомлень про помилки при некоректному введенні даних під час розрахунку енергетичної цінності продукту на обрану користувачем кількість грамів. Тест-кейс представлено у таблиці 1.4.

Таблиця 1.4 – Тест-кейс TC004

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейс ID**: TC004 | **Пріоритет:** 4 | | **Створено**:28.05.2024, Бекешко Ю.Р. |
| **Назва:** Перевірка виведення повідомлень про помилки при некоректному введенні даних.  **Вхідні дані:** Фото продукту обраного користувачем було успішно розпізнано. | | | |
| **Кроки** | | **Очікуваний результат** | |
| 1. Ввести некоректне значення (не число) у поле для введення кількості грамів. 2. Натиснути кнопку «Розрахувати». | | Виведення повідомлення "Будь ласка, введіть число для кількості грамів." | |
| 1. Ввести порожнє значення у поле для введення кількості грамів. 2. Натиснути кнопку «Розрахувати». | | Виведення повідомлення "Будь ласка, введіть кількість грам продукту." | |
| **Результат виконання тест-кейсу:** пройдено успішно | | | |

Після успішного завантаження та розпізнавання зображення, є можливість розрахувати енергетичну цінність продукту за бажаною кількістю грамів. Якщо в поле для введення кількості грамів, буде введено не число та натиснути кнопку «Розрахувати», буде виведена помилка: "Будь ласка, введіть число для кількості грамів." (рисунок 1.8). Також, якщо ввести порожнє значення у текстове поле для введення кількості грамів та натиснути кнопку «Розрахувати», система виведе повідомлення: "Будь ласка, введіть кількість грам продукту." (рисунок 1.9).

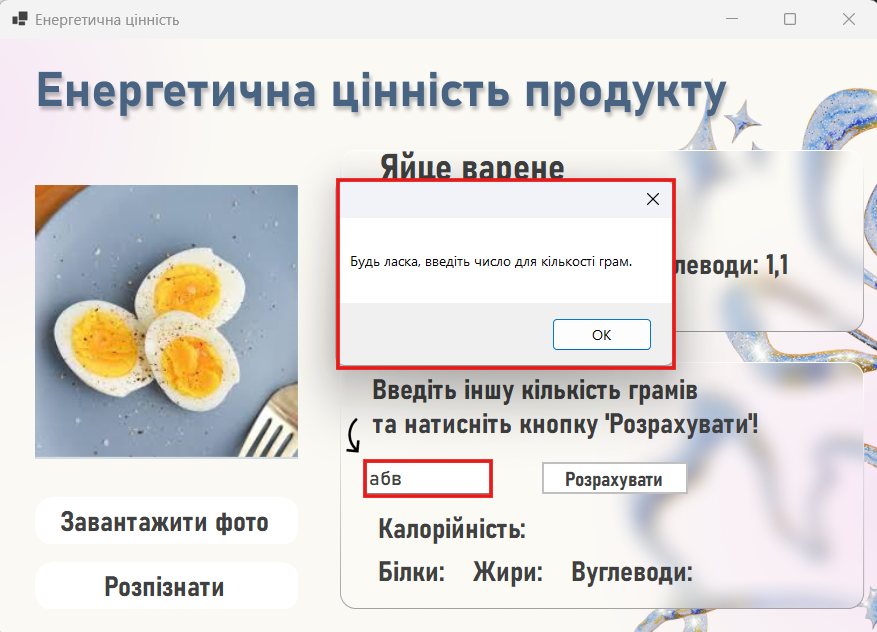


Рисунок 1.8 – Демонстрація виведення вікна помилки

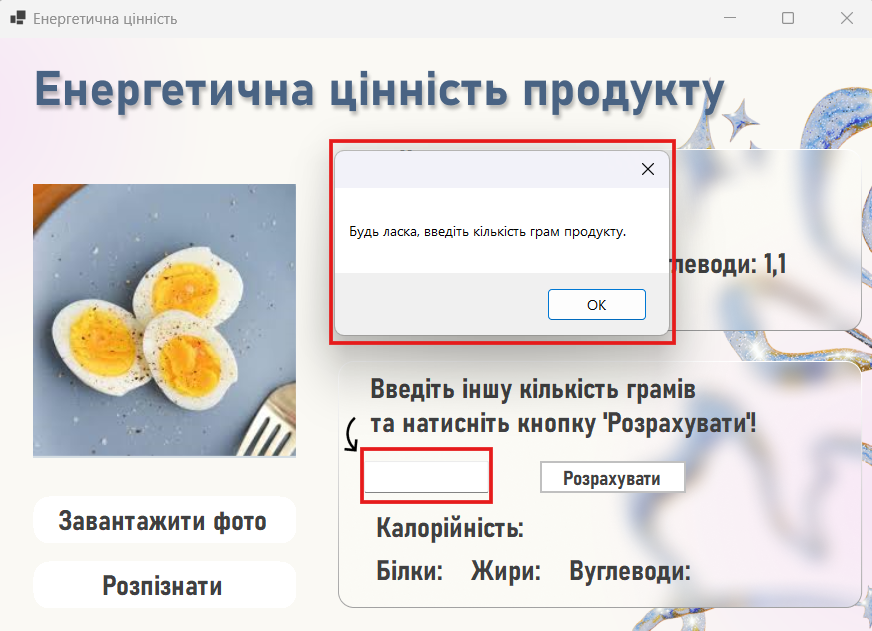


Рисунок 1.9 – Демонстрація виведення вікна помилки

П’ятий тест-кейс демонструє Перевірка повідомлень про помилки при відсутності розпізнаного продукту в базі даних. Тест-кейс представлено у таблиці 1.5.

Таблиця 1.5 – Тест-кейс TC005

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейс ID**: TC005 | **Пріоритет:** 5 | | **Створено**:28.05.2024, Бекешко Ю.Р. |
| **Назва:** Перевірка повідомлень про помилки при відсутності розпізнаного продукту в базі даних.  **Вхідні дані:** Зображення продукту, якого немає в базі даних. | | | |
| **Кроки** | | **Очікуваний результат** | |
| 1. Програма виводить повідомлення "Продукт не знайдено в базі даних". 2. Натиснути кнопку «Розпізнати». | | Назва продукту виводиться на екран, але оцінка йог енергетичної цінності не відбувається і програма виводить повідомлення "Продукт не знайдено в базі даних". | |
| **Результат виконання тест-кейсу:** пройдено успішно | | | |

Цей тест-кейс перевіряє коректність обробки випадків, коли зображення продукту не відповідає жодному продукту в джерелі даних про розпізнаний продукт. Очікується, що програма виведе відповідне повідомлення про помилку, зберігаючи свою стабільність та коректну роботу.

Обирається зображення, після чого розпізнається продукт, але не виводиться інформація про його енергетичну цінність, оскільки розпізнаний продукт не знайдено в джерелі даних про енергетичну цінність продуктів (рисунок 1.10).

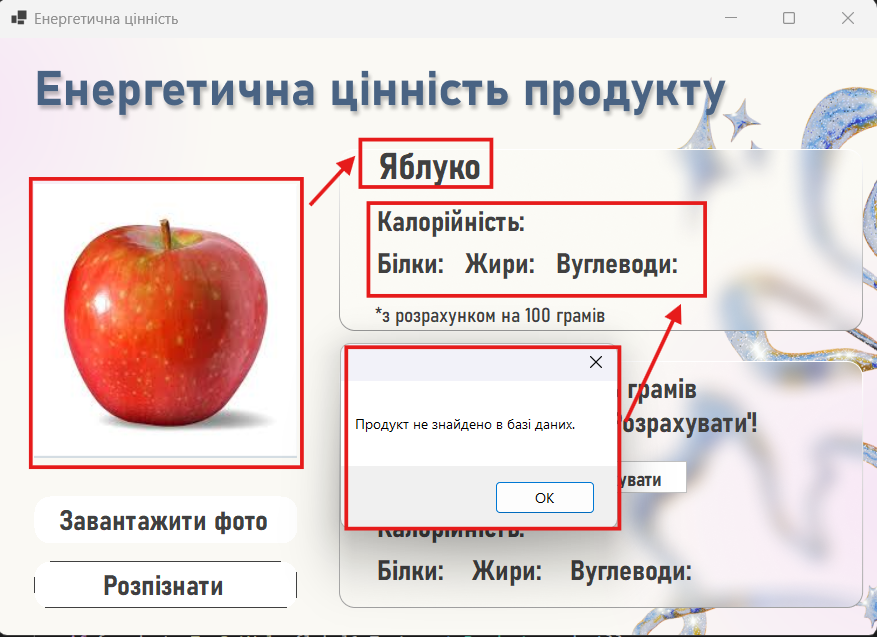


Рисунок 1.10 – Демонстрація виведення вікна помилки

Після проведення тестування програмного продукту не було виявлено неправильно працюючих функцій. Це свідчить про те, що створена програма є відповідною до вимог специфікації та може бути використаною задля полегшення процесу оцінки енергетичної цінності продуктів харчування.

**Висновок**

Під час лабараторної роботи №7 було зроблено тестування розробленого в рамках курсового проєкту, в додатку не було виявлено жодних проблем у його роботі. Це підтверджує, що додаток працює коректно і відповідає очікуванням.

**Контрольні питання**

1. Підходи до тестування ПЗ

Функціональне тестування: перевірка функціональності програмного забезпечення відповідно до вимог.

Нефункціональне тестування: оцінка таких аспектів, як продуктивність, безпека, зручність використання тощо.

Статичне тестування: включає рев'ю коду, аналіз, верифікацію і валідацію без виконання програми.

Динамічне тестування: тестування шляхом виконання коду, аналіз результатів.

Чорний ящик: тестування без знання внутрішньої структури коду.

Білий ящик: тестування з урахуванням внутрішньої структури коду.

Сірий ящик: комбінує методи чорного та білого ящиків.

1. Відмінності між ручним і автоматизованим тестуванням

Ручне тестування: виконання тестових випадків вручну без використання автоматизації.

Переваги: гнучкість, підходить для складних і непередбачуваних випадків.

Недоліки: трудомістке, повільне, схильне до людських помилок.

Автоматизоване тестування: використання спеціальних інструментів для автоматизації виконання тестів.

Переваги: швидке, повторюване, знижує людські помилки.

Недоліки: висока початкова вартість, потребує підтримки скриптів.

1. Відмінності між негативним і позитивним тестуванням

Позитивне тестування: перевірка, чи працює система як очікується при введенні валідних даних.

Негативне тестування: перевірка, як система поводиться при введенні невалідних або непередбачених даних.

1. Відмінності між функціональним і структурним підходами до тестування

Функціональне тестування: тестування функцій програмного забезпечення без знання його внутрішньої структури.

Структурне тестування: тестування, що враховує внутрішню структуру коду, часто називають тестуванням білого ящика.

1. Рівні тестування

Модульне тестування: перевірка окремих компонентів або модулів програми.

Інтеграційне тестування: перевірка взаємодії між модулями.

Системне тестування: тестування всієї системи як єдиного цілого.

Приймальне тестування: перевірка системи з точки зору кінцевого користувача або замовника.

1. Регресивне тестування та комбінування рівнів тестування

Регресивне тестування: перевірка, чи не з’явилися нові помилки в раніше протестованих частинах ПЗ після внесення змін або додавання нових функцій.

Комбінування рівнів тестування включає використання різних рівнів (модульне, інтеграційне, системне) в одному циклі тестування для забезпечення покриття всіх аспектів програми.

1. Суть тестування продуктивності ПЗ

Тестування продуктивності: оцінка системи на відповідність вимогам до швидкості, масштабованості, стабільності при навантаженні та інших метрик продуктивності.

1. Інструментальні засоби тестування

Автоматизовані інструменти: Selenium, JUnit, TestNG, LoadRunner.

Інструменти для відстеження помилок: JIRA, Bugzilla.

Інструменти для тестування продуктивності: JMeter, Gatling.

1. Класифікація помилок

Синтаксичні: помилки у використанні мови програмування.

Логічні: помилки у логіці виконання програми.

Функціональні: помилки в реалізації функціональних вимог.

Нефункціональні: проблеми продуктивності, безпеки тощо.

1. Ручне тестування

Ручне тестування: процес перевірки ПЗ без використання автоматизованих інструментів, де тести виконуються вручну.

1. Типи ручного тестування

Дослідницьке тестування: тестування без заздалегідь підготовлених тест-кейсів.

Регресивне тестування: повторне тестування після внесення змін.

Смоук тестування: початкове тестування для перевірки базової працездатності.

Санітарне тестування: перевірка після невеликих змін у коді.

1. Етапи ручного тестування

Планування тестування: визначення мети, обсягу, підходів та ресурсів для тестування.

Аналіз і дизайн тестів: створення тестових сценаріїв, визначення умов тестування.

Виконання тестів: проведення тестування за сценаріями.

Звітність: документування результатів тестування, виявлених дефектів.

1. Правила ручного тестування

Знати вимоги і специфікації ПЗ.

Писати чіткі та зрозумілі тест-кейси.

Виконувати тести в контрольованому середовищі.

Звітувати про помилки детально та коректно.

Перевіряти результати тестів.

1. Інструменти для ручного тестування

Баг-трекери: JIRA, Bugzilla, Redmine.

Тест-менеджмент системи: TestRail, Zephyr.

1. Checklist та його мета

Checklist: перелік завдань або критеріїв, які потрібно перевірити.

Мета: забезпечити, щоб жоден важливий аспект тестування не був пропущений.

1. Правила створення Checklist

Бути конкретним та зрозумілим.

Охоплювати всі критичні аспекти тестування.

Регулярно оновлюватися відповідно до змін у ПЗ.

1. Test Case

Test Case: документ, що описує вхідні дані, умови виконання, очікуваний результат і фактичний результат для конкретного тесту.

1. Життєвий цикл Test Case

Створення: написання тест-кейсу.

Огляд: перевірка тест-кейсу.

Виконання: проведення тесту згідно з тест-кейсом.

Звітність: документування результатів.

Аналіз результатів: оцінка результатів тестування.

Підтримка: оновлення тест-кейсів при зміні вимог.

1. Атрибути Test Case

Ідентифікатор

Назва

Опис

Передумови

Кроки виконання

Очікуваний результат

Фактичний результат

Примітки

1. Test Case Suite

Test Case Suite: група пов'язаних тест-кейсів, об'єднаних для досягнення певної мети тестування.

1. Інструментальні засоби створення Test Case

TestRail: система управління тестуванням.

Zephyr: плагін для JIRA для управління тест-кейсами.

HP ALM: комплексне рішення для управління життєвим циклом тестування.