Raportowanie błędów

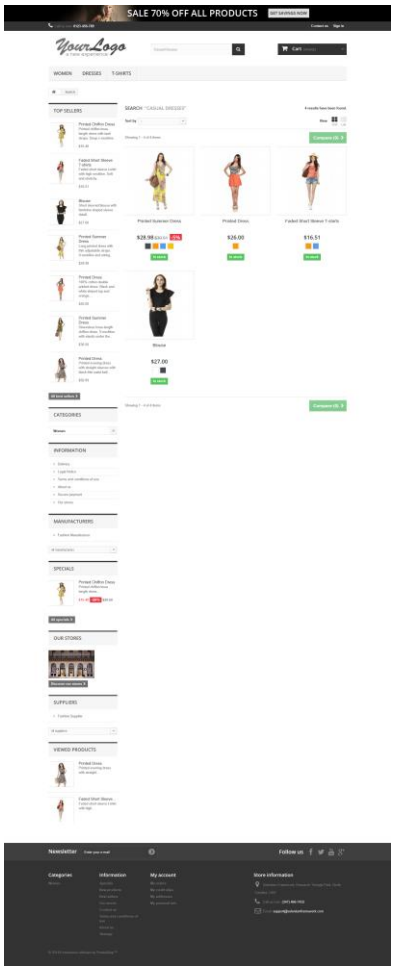
1. **Błąd dotyczy:** TC - 1.2 - Sprawdzenie działania wyszukiwania produktów - poprawne dane
2. **Ważność błędu:** Normal
3. **Środowisko testowe:**
   http://automationpractice.com/index.php
   Windows 10
   Przeglądarka Firefox 72.0.2 (64-bit)
4. **Kroki wywołania błędu:**

   Znajduje się na stronie głównej aplikacji
   Wypełniam pole "search" danymi testowymi (Casual Dresses)
   Wybieram przycisk "lupa"

5. **Oczekiwany rezultat:**

- Użytkownik zostaje przekierowany na stronę z rezultatem wyszukiwania
- System wyświetla produkty zgodnie z kluczem wprowadzonym w polu “search”

6. **Faktyczny rezultat:**
   Na stronie z rezultatem wyszukiwania są wyświetlane produkty, których nie powinno być w kategorii “Casual Dresses” (Blouse).



Blocker

Jest to najpoważniejszy błąd. Wartość “blocker” ustawiamy tylko w wypadku, kiedy aplikacja nie nadaje się do dalszych testów czy użytkowania. Innymi słowy proces testowy jest zablokowany i tylko poprawka może go udrożnić. Przy tego typu błędach blokujemy release aplikacji aż do momentu poprawienia usterek (przejścia review, retestu i zamknięcia jako poprawione).
Przykład: *Po zalogowaniu do systemu nie jest możliwe wykonanie w nim żadnych czynności.*

Critical

Krytyczną ważność otrzymują defekty, które uniemożliwiają nam korzystanie w pełni z funkcjonalności w systemie, natomiast nie blokują nam procesu testowego. Tak samo jak w przypadku błędów typu „blocker” nie powinniśmy dopuszczać do release produktu przed poprawieniem błędów z tą ważnością.
Przykład: *Po zalogowaniu do systemu nie jest możliwe generowanie faktur, ale zakupy i sprzedaż działają i można je testować.*

Major

Priorytet “major” otrzymują błędy poważne w kontekście testowanego komponentu. Wpływają one na większą część testowanej funkcjonalności, natomiast nie uniemożliwiają korzystania z niej. To znaczy, że istnieje obejście problemu w ramach funkcjonalności, w której dany defekt występuje.
Przykład: *Nie jest możliwe generowanie faktury przez zalogowanego klienta, ale administrator sklepu może to robić.*

Normal

Taki błąd nie wpływa w większym stopniu na działanie testowanej funkcjonalności, użytkownik jest w stanie skorzystać z pełnych możliwości aplikacji.
Przykład: *Wygenerowana faktura ma na końcu pustą stronę.*

Minor

Defekty, które są marginalne z punktu widzenia systemu. Użytkownik nawet jeżeli się na nie natknie to nie powinien odczuwać dyskomfortu podczas używania systemu. Przeważnie są to literówki w głównych częściach aplikacji.
Przykład: *Literówka na fakturze w słowie „Lista zakupów”.*

Trivial

Jak sama nazwa wskazuje, są to błędy nieistotne z punktu widzenia aplikacji. Tak jak w przypadku wartości “minor”, wartością “triviwal” określamy głównie literówki. Różnica między obiema wartościami polega na tym, że wartość “trivial” ustawiamy dla błędów w częściach aplikacji, które są praktycznie nie używane przez użytkownika.
Przykład: *Literówka na stronie z regulaminem sklepu.*

Priorytet ustanawia, w jakiej kolejności błędy będą poprawiane

Podając wartość w polu **Priorytet (eng. Priority)** powinniśmy się zastanowić nad tym **jak pilne jest wyeliminowanie defektu z punktu widzenia klienta**. Z tego powodu **zdaża się, że Priorytet nie jest tożsamy z Ważnością**. Podumowując Priorytet stanowi informację na temat tego, które zadanie powinniśmy rozwiązać w pierwszej kolejności, a które w dalszej.
Do dyspozycji mamy następujące wartości.

Natychmiastowy

Wskazuje, że danym defektem powinniśmy się zająć w pierwszej kolejności. Zwykle są to błędy mające duży wpływ na aspekt biznesowy aplikacji.
Przykład: *Po zalogowaniu do aplikacji nie można z niej korzystać.*

Wysoki

Poprawienie błędu jest istotne z punktu widzenia biznesowego i powinniśmy się nim zająć możliwie jak najszybciej.

Przykład: *Ikony umożliwiające udostępnianie treści w mediach społecznościowych działają ale wyglądają jakby były nieaktywne*.