**PLAN TEstów akceptacyjnych**

**dla GPWplus**

Wersja 1.12

Autorzy: jakub olejniczak (jakubkuba997@gmail.com)

**Historia zmian dokumentu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Wersja** | **Data** | **Osoba** | **Komentarz** |
| 1.0 | 08.06.2019 | Jakub Olejniczak | Pierwszy zarys dokumentu |
| 1.1 | 11.06.2019 | Jakub Olejniczak | Skorygowano nieścisłości w tytułach |
| 1.11 | 13.06.2019 | Jakub Olejniczak | Aktualizacja ze względu na restrukturyzację projektu |
| 1.12 | 24.06.2019 | Jakub Olejniczak | Uzupełnienie informacji o podejściu do testów |

**Spis treści**

[1. Wprowadzenie 3](#_Toc12282558)

[1.1. Cele testów i ich ograniczenia 3](#_Toc12282559)

[1.2. podejście do testów 3](#_Toc12282560)

[1.3. wymagania systemowe 3](#_Toc12282561)

[1.4. środowisko testowe 3](#_Toc12282562)

[1.5. podstawa testów 3](#_Toc12282563)

[1.6. główny przedmiot testów 3](#_Toc12282564)

[1.7. poboczne przedmioty testów 3](#_Toc12282565)

[2. funkcjonalność 5](#_Toc12282566)

[2.1. funkcjonalność do przetestowania 5](#_Toc12282567)

[2.2. funkcjonalność nietestowana 5](#_Toc12282568)

[3. dostawy testowe 6](#_Toc12282569)

[3.1. plan testów 6](#_Toc12282570)

[3.2. scenariusz testowy 6](#_Toc12282571)

[3.2.1. Przypadki testowe 6](#_Toc12282572)

[3.3. historia konsoli 6](#_Toc12282573)

[3.4. skrypty funkcji i skrypty testów 6](#_Toc12282574)

[3.5. matryca błędów 6](#_Toc12282575)

[3.6. raport końcowy 6](#_Toc12282576)

[4. kryteria akceptacji 7](#_Toc12282577)

1. Wprowadzenie
   1. Cele testów i ich ograniczenia

Podstawowym celem testowania jest znalezienie jak największej ilości defektów – niedogodności związanych z błędami podczas procesu tworzenia oprogramowania, mogących prowadzić do powstawania przeróżnych awarii, uniemożliwiających prawidłowe korzystanie z systemu.

Wykonywanie testów jest kluczem w procesie budowania zaufania odbiorcy do jakości używanego oprogramowania.

Celem zaplanowanych testów będzie przetestowanie końcowego produktu, bez wdrażania się w jego strukturę wewnętrzną.

* 1. podejście do testów

Testy bazują na podstawie testów i zamieszczonych w niej historyjkach użytkownika i mają charakter dynamiczny. Testowanie akceptacyjne skupiać się będzie na funkcjonalności badanego oprogramowania, będzie przeprowadzane w całości metodami automatycznymi aby ułatwić późniejsze testy regresji i testy potwierdzające.

Ze względu na przedstawione przedmioty testów, będą się one odbywały bezpośrednio na końcowym pliku oprogramowania metodą czarnoskrzynkową.

Wszystkie zaplanowane testy są pierwszymi testami przeprowadzonymi na oprogramowaniu, przed ich wykonaniem nie istnieją żadne dostępne testy regresyjne lub informacje o znanych defektach.

* 1. wymagania systemowe

Testowane oprogramowanie wymaga użycia środowiska systemowego Windows10®. W specyfikacji wymagań nie określono żadnych dodatkowych wymogów.

* 1. środowisko testowe

System operacyjny: Windows 10 Home

Procesor: Intel Core i7-5500U CPU @ 2.40GHz

Pamięć RAM: 8.00 GB

Rodzaj systemu: 64-bit Operating System, x64-based procesor

Ping: średnia 30ms

Pobieranie: średnia 13.17 Mb/s

* 1. podstawa testów

Za podstawę wykonywanych testów służyć będzie plik specyfikacji wymagań programu GPWPlus, zamieszczony w folderze dokumentacji tego projektu.

* 1. główny przedmiot testów

Głównym przedmiotem wykonywanych testów jest plik egzekucyjny(.exe), będący wersją finalną testowanego systemu. Jest on zamieszczony w folderze dokumentacji tego projektu, jako przedmiot testów.

* 1. poboczne przedmioty testów

W trakcie wykonywania testów, za sprawą testowanego systemu, mogą powstać dodatkowe pliki, będące niezbędną częścią poprawnego działania oprogramowania. Będą one zapisywane w folderze głównego przedmiotu testów i również mogą one służyć za przedmiot testów.

1. funkcjonalność
   1. funkcjonalność do przetestowania

Wymagania funkcjonalne objęte testowaniem, znajdujące się w specyfikacji wymagań:

|  |  |
| --- | --- |
| ID: | **REQ-F001** |
| Nazwa: | Odczyt funkcji pomocy i informacji o produkcie |

|  |  |
| --- | --- |
| ID: | **REQ-F002** |
| Nazwa: | Poprawny, permanentny zapis podanych spółek |

|  |  |
| --- | --- |
| ID: | **REQ-F004** |
| Nazwa: | Możliwość poprawnego i czytelnego korzystania z konsoli |

|  |  |
| --- | --- |
| ID: | **REQ-F005** |
| Nazwa: | Sprawdzanie poprawności podanych spółek |

|  |  |
| --- | --- |
| ID: | **REQ-F006** |
| Nazwa: | Działania na zapisanych spółkach |

|  |  |
| --- | --- |
| ID: | **REQ-F007** |
| Nazwa: | Poprawna komunikacja z użytkownikiem |

* 1. funkcjonalność nietestowana

Poza wszystkimi wymaganiami niefunkcjonalnymi zawartymi w podstawie testów, następujące wymagania funkcjonalne nie będą objęte testowaniem:

|  |  |
| --- | --- |
| ID: | **REQ-F003** |
| Nazwa: | **Możliwość manualnego zapisywania akcji** |
| Powód: | Niekrytyczny priorytet, połączony z możliwością łatwego sprawdzenia danego wymagania za pomocą testów manualnych |

1. dostawy testowe

Poniżej wymienione zostały zależne od siebie rzeczy dostarczone wraz z testami, które powinny się znajdować w tym samym projekcie.

* 1. plan testów

Plan testów jest dokumentem obszernie wyjaśniającym sposób podejścia do testowania i najważniejsze rzeczy związane z pracą.

* 1. scenariusz testowy

Scenariusz testowy jest dokumentem określającym ciąg akcji umożliwiających wykonanie danych testów. Są w nim zawarte wszystkie niezbędne informacje potrzebne do manualnego odtworzenia wykonywanych testów oraz ich powiązanie do wymogów zawartych w podstawie testów.

* + 1. Przypadki testowe

Każdy test znajdujący się w scenariuszu testowym składa się z indywidualnych przypadków testowych.

* 1. historia konsoli

Folder zawierający potencjalne historie konsoli, zapisywane w przypadku znalezienia defektu, powinien znajdować się razem z dokumentacją projektu. Takie teksty ułatwiają pracę nad debugowaniem oraz pozwalają poprawnie zidentyfikować napotkany defekt. Każda historia konsoli nazywana jest po ID scenariuszu testowego którego dotyczy, w ramach ułatwienia późniejszego poszukiwania powiązań.

* 1. skrypty funkcji i skrypty testów

Skrypty funkcji są nieodłącznym obiektem testowania automatycznego i służą do automatycznego używania danego systemu. Zostały one stworzone z myślą o działaniu niezależnym od siebie nawzajem, w przypadku wymagania wykonania powtórzonych czynności wielokrotnie, tworzone są niezależne skrypty z funkcjami, pozwalające łatwo usunąć ewentualne błędy bądź zmiany w kodzie.

Skrypty funkcji zapisywane są w folderze nazwanym ‘Function<nazwa\_funkcji>’. W ramach ułatwienia późniejszego wykonywania testów należy je zapisywać pod nazwą ‘\_\_init\_\_’.

Skrypty testów korzystają ze skryptów funkcji aby zdobyć dane dotyczące powodzenia bądź niepowodzenia danych wydarzeń. Zapisywane są w tym samym folderze co skrypty funkcji, pod nazwą ‘Test<nazwa\_testu>’.

* 1. matryca błędów

Matryca błędów umożliwia łatwe i szybkie znalezienie powiązań pomiędzy wykonywanymi testami, a wymaganiami testowymi.

* 1. raport końcowy

Raport końcowy sporządzany jest po wykonaniu wszystkich testów, na bazie kryteriów akceptacji.

1. kryteria akceptacji

Oprogramowanie przejdzie pomyślnie przez fazę testowania tylko i wyłącznie gdy:

1. Wszystkie testowane wymagania funkcjonalne zostaną pokryte co najmniej dwoma testami
2. Wszystkie wymagania funkcjonalne o priorytecie wysokim pomyślnie przejdą testy
3. Iloraz liczby znalezionych defektów przez liczbę wykonanych testów będzie mniejszy niż 0.15