**Karta wymagań dla aplikacji.**

//Opis aplikacji: aplikacja licząca miejsca zerowe funkcji kwadratowej z silnikiem w Spring Boot i GUI

**1. Wymagania funkcjonalne (Functional Requirements)**

// Definiują, **co** system musi robić. Zgodnie z ISO, skupiamy się na kompletności i poprawności funkcjonalnej.

Tabela 1

Wymagania funkcjonalne

|  |  |  |
| --- | --- | --- |
| ID | Nazwa | Opis |
| RF-01 | Wprowadzanie parametrów | System musi umożliwiać użytkownikowi wprowadzenie współczynników $a, b, c$ równania kwadratowego ($ax^2 + bx + c = 0$) przez interfejs graficzny (GUI). |
| RF-02 | Walidacja danych | System musi blokować możliwość obliczeń, jeśli współczynnik $a = 0$ (równanie nie jest kwadratowe) i informować o tym użytkownika. |
| RF-03 | Obliczanie wyróżnika | Silnik (Spring Boot Service) musi poprawnie obliczać deltę wg wzoru: $\Delta = b^2 - 4ac$. |
| RF-04 | Wyznaczanie pierwiastków | System musi wyznaczać liczbę miejsc zerowych (0, 1 lub 2) oraz ich wartości w zbiorze liczb rzeczywistych. |
| RF-05 | Prezentacja wyników | GUI musi wyświetlać wyniki obliczeń (wartości $x\_1, x\_2$ lub informację o braku miejsc zerowych) z precyzją do 2 miejsc po przecinku. |
| RF-06 | Resetowanie formularza | System musi oferować funkcję "Wyczyść", która przywraca stan początkowy pól tekstowych. |

*Źródło: opracowanie własne*

**2. Wymagania niefunkcjonalne (Non-functional Requirements)**

Tabela 1

Wymagania niefunkcjonalne

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Kategoria (ISO 25010) | Nazwa | Opis |
| RNF-01 | Wydajność (Performance) | Czas reakcji interfejsu | Czas od zatwierdzenia danych (kliknięcia przycisku) do wyświetlenia wyniku nie może przekroczyć 200 ms. |
| RNF-02 | Kompatybilność (Compatibility) | Środowisko uruchomieniowe | Aplikacja musi być kompatybilna ze środowiskiem Java JRE/JDK w wersji minimum 17. |
| RNF-03 | Użyteczność (Usability) | Czytelność interfejsu | Interfejs użytkownika musi być intuicyjny; użytkownik powinien móc wykonać obliczenie bez korzystania z instrukcji obsługi. |
| RNF-04 | Niezawodność (Reliability) | Obsługa wyjątków | System musi przechwytywać i obsługiwać błędy typu NumberFormatException (wpisanie liter zamiast cyfr) bez przerywania działania aplikacji. |
| RNF-05 | Konserwowalność (Maintainability) | Architektura warstwowa | Logika biznesowa (obliczenia) musi być oddzielona od warstwy prezentacji (GUI) przy pomocy serwisów zarządzanych przez Spring Boot. |
| RNF-06 | Przenaszalność (Portability) | Formuła dostarczenia | Aplikacja musi być skompilowana do postaci pojedynczego, wykonywalnego pliku .jar (tzw. fat-jar). |
| RNF-07 | Bezpieczeństwo (Security) | Walidacja wejścia | Każdy parametr wejściowy musi być sprawdzany pod kątem poprawności typu danych przed przekazaniem do silnika obliczeniowego. |
| RNF-08 | Konserwowalność (Maintainability) | Testowalność | Kod silnika obliczeniowego musi umożliwiać pokrycie testami jednostkowymi (JUnit) na poziomie minimum 90% linii kodu logicznego. |
| RNF-09 | Wydajność (Performance) | Zużycie zasobów | Aplikacja w stanie spoczynku nie powinna zajmować więcej niż 256 MB pamięci RAM (standard dla małych kontekstów Spring Boot). |

*Źródło: opracowanie własne*

**3. Wymagania jakościowe (wg ISO/IEC 25010)**

Tabela 2

Wymagania jakościowe.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Charakterystyka ISO | Nazwa wymagania | Opis / Kryterium sukcesu |
| RQ-01 | Poprawność funkcjonalna | Precyzja obliczeniowa | System musi wykonywać obliczenia na typach danych o wysokiej precyzji (double lub BigDecimal), zapewniając dokładność wyniku do min. 10 miejsc po przecinku przed zaokrągleniem w GUI. |
| RQ-02 | Użyteczność | Estetyka i komunikatywność | Interfejs musi jasno sygnalizować stan aplikacji: np. kolor czerwony dla błędów walidacji, wyraźne pole wynikowe dla miejsc zerowych. |
| RQ-03 | Użyteczność | Ochrona przed błędami | System musi uniemożliwić wykonanie obliczeń („wyszarzony” przycisk lub komunikat), jeśli pola wejściowe są puste lub zawierają znaki nie-numeryczne. |
| RQ-04 | Niezawodność | Stabilność (Recoverability) | W przypadku wystąpienia nieoczekiwanego błędu w silniku Spring Boot, aplikacja musi przechwycić wyjątek i powrócić do stanu gotowości, zamiast kończyć proces (crash). |
| RQ-05 | Wydajność | Czas startu (Startup time) | Czas inicjalizacji kontekstu Spring Boot oraz uruchomienia GUI nie może przekraczać 5 sekund na standardowej stacji roboczej. |
| RQ-06 | Konserwowalność | Analizowalność (Logging) | Aplikacja musi logować kluczowe zdarzenia (start, błędy walidacji, wykonane obliczenia) do konsoli przy użyciu mechanizmu SLF4J/Logback typowego dla Spring Boot. |
| RQ-07 | Konserwowalność | Modułowość (Modularity) | Zmiana silnika obliczeniowego (np. na obsługujący liczby zespolone) nie może wymagać modyfikacji kodu odpowiedzialnego za interfejs graficzny (implementacja wzorca *Strategy* lub *Service*). |
| RQ-08 | Przenaszalność | Niezależność systemowa | Dzięki wykorzystaniu JVM i Spring Boot, aplikacja musi zachowywać identyczną logikę obliczeń i wygląd GUI na systemach Windows, Linux oraz macOS. |
| RQ-09 | Bezpieczeństwo | Integralność danych | System nie może pozwalać na wstrzyknięcie kodu lub niebezpiecznych znaków poprzez pola formularza (podstawowa walidacja typu danych wejściowych). |

*Źródło: opracowanie własne*