Specyfikacja wymagań

Aplikacja wieloosobowej platformy gier

wersja 2025-04-04

Anna Czart, Julia Guga

1. Wstęp

Niniejszy dokument zawiera specyfikację wymagań systemu aplikacji desktopowej będącej wieloosobową platformą gier inspirowaną serwisem kurnik.pl. Specyfikacja wymagań opisuje oczekiwane cele, zakres oraz wymagania funkcjonalne i niefunkcjonalne jakie powinien spełniać system platformy gier, wymagania dotyczące jakości modelowanego systemu, bazy danych, stosowanych standardów oraz opis nałożonych na tworzony system ograniczeń.

2. Cele specyfikacji

Specyfikacja wymagań ma na celu znalezienie odpowiedzi na poniższe pytania:

● W jakim celu klient zamawia dany system?

● W jakim środowisku będzie pracował dany system?

● Kto będzie użytkownikiem systemu?

● Jakie procesy funkcjonalne system powinien obsługiwać?

● Jakie są procesy i funkcje systemu?

● Kiedy system ma być dostarczony?

● Inne ograniczenia na system.

3. Zakres specyfikacji

Zakres specyfikacji obejmuje funkcjonalności związane z zarządzaniem wieloosobową platformą gier oraz niefunkcjonalne wymagania dotyczące m.in. jakości, użyteczności, wydajności czy bezpieczeństwa. Do grupy użytkowników systemu należą gracze korzystający z platformy. System zapewnia płynne połączenia, zarządzanie sesjami gry oraz monitorowanie statystyk, tworząc kompleksowe środowisko dla graczy online.

4. Wymagania funkcjonalne

Wymagania funkcjonalne definiują, co system musi robić dla użytkowników. Opisują czynności, operacje, usługi wykonywane przez system. Lista wymagań funkcjonalnych jest stworzona na podstawie dotychczas zgromadzonej wiedzy, a w szczególności na podstawie scenariuszy użycia.

● Użytkownik musi mieć możliwość rejestracji poprzez podanie danych osobowych oraz stworzenie unikalnego loginu i hasła.

● Użytkownik musi mieć możliwość logowania się do systemu za pomocą loginu i hasła.

● Użytkownik musi mieć możliwość przeglądania listy dostępnych gier (Kółko-krzyżyk, Wisielec, Statki)

● Użytkownik musi mieć możliwość odzyskiwania hasła poprzez podanie adresu e-mail powiązanego z kontem.

● Użytkownik może wprowadzić maksymalnie 3 razy błędne hasło przed zablokowaniem konta na określony czas (np. 30 minut).

● Hasło musi się składać z minimum 8 znaków, nie mogą być one spacjami ani znakami specjalnymi.

● Na jeden adres e-mail można założyć tylko jedno konto.

● Użytkownik musi mieć możliwość rozpoczęcia nowej gry, dołączenia do istniejącej rozgrywki lub zaproszenia innych graczy do wspólnej gry.

● Użytkownik musi mieć możliwość przeglądania historii swoich gier, wyników oraz statystyk.

● Użytkownik musi mieć możliwość dostosowania swojego profilu, w tym zmiany awatara i nazwy użytkownika.

● Użytkownik musi mieć możliwość wylogowania się z systemu w dowolnym momencie.

● System musi zapewniać odpowiednią ochronę danych osobowych oraz zabezpieczenia przed nieautoryzowanym dostępem do konta użytkownika.

5. Wymagania niefunkcjonalne

Wymagania niefunkcjonalne definiują, w jakich warunkach system musi realizować wymagania funkcjonalne. Nie odnoszą się one bezpośrednio do cech systemu, ale określają bezwzględnie konieczne wymogi wynikające z potrzeb użytkownika, budżetu, strategii firmy, czynników zewnętrznych oraz współpracy z innymi systemami. Dotyczą one systemu jako całości, a nie poszczególnych jego cech.

● Aplikacja powinna być przygotowana w wersji graficznej, z przyjaznym i intuicyjnym interfejsem użytkownika.

● Interfejs użytkownika oraz dane powinny być dostępne w dwóch wersjach językowych: angielskim oraz polskim, z możliwością łatwej zmiany języka w ustawieniach.

● System powinien obsługiwać co najmniej 2 równoczesnych użytkowników, zapewniając płynność działania aplikacji bez opóźnień.

● Dane użytkowników oraz wyniki gier muszą być chronione przed nieautoryzowanym dostępem za pomocą nowoczesnych metod szyfrowania i zabezpieczeń.

● Aplikacja powinna być wysoce elastyczna, aby mogła być używana przez każdego użytkownika bez trudności, niezależnie od poziomu doświadczenia w grach komputerowych.

● Informacje dotyczące użytkowników, takie jak dane rejestracyjne, historia gier oraz statystyki, powinny być przechowywane w bazie danych, zapewniając ich integralność i łatwy dostęp w przyszłości.

5.1 Interfejsy

Rozdział ten poświęcony został opisom interfejsów jakie będą implementowane w tworzonym systemie tj. Interfejsy użytkownika, sprzętowe, komunikacyjne oraz interfejsy programowe.

5.1.1 Interfejsy użytkownika

Po uruchomieniu aplikacji powinna pojawić się formatka logowania, na której użytkownik będzie mógł wprowadzić dane do logowania. Po poprawnym zalogowaniu się pojawi się główna formatka aplikacji. W przypadku zapomnienia hasła użytkownik powinien mieć możliwość jego przypomnienia za pośrednictwem formatki do zmiany hasła. Jeśli użytkownik nie posiada konta, powinien mieć możliwość zarejestrowania się za pośrednictwem formatki do rejestracji.

Po wykonaniu opisanych wcześniej czynności, użytkownik zostanie przeniesiony do formatki z głównym menu aplikacji. Z tego poziomu będzie mógł przejść do panelu użytkownika, wybrać jedną z trzech gier lub wylogować się z aplikacji. Formatka panelu użytkownika powinna zawierać informacje dotyczące danych logowania.
Po wybraniu jednej z gier powinno pojawić się wspólne dla wszystkich gier okno, służące do zaproszenia gracza o określonym nicku. Po wysłaniu zaproszenia i jego akceptacji powinno otworzyć się okno gry.

5.1.2 Interfejsy sprzętowe

Aplikacja desktopowa zazwyczaj nie korzysta bezpośrednio z interfejsów sprzętowych. Jednakże korzysta z różnych komponentów sprzętowych, takich jak monitor wyświetlający graficzny interfejs użytkownika aplikacji, klawiatura pozwalająca użytkownikowi wprowadzać tekst oraz mysz umożliwiająca interakcję z interfejsem graficznym.

5.1.3 Interfejsy komunikacyjne

Interfejsy komunikacyjne w aplikacji desktopowej platformy gier wieloosobowych obejmują zarówno graficzne interfejsy użytkownika, interfejsy bazy danych, jak i mechanizmy komunikacji klient-serwer.
Interfejsy GUI, takie jak JavaFX, Swing czy SWT, umożliwiają graczom interakcję z aplikacją poprzez różnorodne elementy graficzne — takie jak menu główne, okna zaproszeń do gry czy panele ustawień.
Z kolei interfejsy bazy danych (w wypadku tej aplikacji)JDBC (Java Database Connectivity), zapewniają integralność i efektywność komunikacji z bazą danych SQL, wspierając operacje związane z obsługą kont graczy, historią rozgrywek czy zapisem stanu gry.
W przypadku rozgrywek wieloosobowych kluczowa jest również komunikacja klient-serwer, realizowana za pomocą technologii takich jak gniazda sieciowe (ang. sockets), protokoły TCP/IP lub WebSockety. Dzięki niej możliwa jest wymiana danych między graczami w czasie

rzeczywistym, synchronizacja stanu gry oraz obsługa zaproszeń, dołączania do rozgrywek i przesyłania ruchów graczy.

5.1.4 Interfejsy programowe

Interfejsy programowe wykorzystywane w platformie gier wieloosobowych, takie jak JDK, Spring Tools 4 oraz JBoss Tools 4.25.0 Final, odgrywają kluczową rolę w prawidłowym funkcjonowaniu i rozwoju aplikacji.

JDK (Java Development Kit) dostarcza niezbędnych narzędzi do kompilacji i uruchamiania aplikacji w języku Java, co stanowi fundament całego środowiska programistycznego. Spring Tools 4 wspiera tworzenie nowoczesnych aplikacji opartych na frameworku Spring, umożliwiając łatwe zarządzanie komponentami, usługami i bezpieczeństwem aplikacji serwerowej.

Z kolei JBoss Tools 4.25.0 Final ułatwia integrację z serwerami aplikacyjnymi JBoss, co jest istotne w kontekście wdrażania i zarządzania backendem platformy gier, zapewniając stabilną komunikację klient-serwer oraz skalowalność całego systemu.

5.2 Warunki serwisowania (support)

Wsparcie techniczne dla użytkowników systemu powinno być dostępne w trybie 24/7, aby zapewnić pomoc w przypadku jakichkolwiek problemów związanych z korzystaniem z platformy. Użytkownicy powinni mieć możliwość zgłaszania problemów, pytań lub sugestii za pośrednictwem e-maila, a odpowiedzi powinny być udzielane w możliwie najkrótszym czasie. System powinien być regularnie aktualizowany, aby wprowadzać nowe funkcjonalności, poprawiać wydajność oraz eliminować wszelkie zgłoszone błędy, zapewniając tym samym wysoką jakość usługi. System musi regularnie tworzyć kopie zapasowe danych użytkowników oraz wyników gier, aby zapewnić ich bezpieczeństwo oraz możliwość odzyskania w razie awarii. Kopie zapasowe powinny być przechowywane w sposób bezpieczny i dostępne w razie potrzeby przywrócenia danych.

5.3 Ograniczenia architektury systemu

Rozdział ten zawiera szczegółowy opis wymagań i ograniczeń architektury systemu, w tym stosowane standardy programowania, języki programowania, stosowane konwencje nazewnictwa, stosowane klasy, komponenty.

Architektura systemu opiera się na modelu klient-serwer, gdzie komunikacja odbywa się przez REST API do zarządzania danymi oraz WebSocket do komunikacji w czasie rzeczywistym między graczami. Aplikacja będzie napisana w języku Java z użyciem frameworku Spring, a baza danych MySQL posłuży do przechowywania danych użytkowników i wyników gier. System będzie bezpieczny, elastyczny i gotowy do dalszego rozwoju dzięki łatwej integracji nowych funkcji i przestrzeganiu najlepszych praktyk programistycznych. Wszystkie klasy, metody i zmienne powinny być nazwane zgodnie z konwencjami nazewnictwa Java. Klasy powinny mieć nazwy rozpoczynające się od wielkiej litery, metody i zmienne powinny zaczynać się od małej litery, a stałe powinny być zapisane wielkimi literami. System musi

implementować najlepsze praktyki zabezpieczeń, w tym szyfrowanie danych wrażliwych, stosowanie mechanizmów uwierzytelniania i autoryzacji.

5.4 Dokumentacja użytkownika

W ramach systemu dostarczana jest pełna dokumentacja użytkownika, która ma na celu umożliwienie łatwego i intuicyjnego korzystania z platformy. Dokumentacja będzie zawierała szczegółowe instrukcje oraz wytyczne dotyczące wszystkich aspektów korzystania z systemu tj. instrukcja obsługi gier czy zakończenia sesji gry zapewniając użytkownikom pomoc na każdym etapie interakcji z aplikacją.

5.6 Wymagania licencyjne

Nie dotyczy

5.7 Prawa autorskie i inne zagadnienia prawne

Platforma gier wieloosobowych powinna wyświetlać informacje dotyczące zrzeczenia się odpowiedzialności, praw autorskich, znaków słownych i towarowych oraz gwarancji związanych z wykorzystywanymi elementami graficznymi, dźwiękowymi i markami obecnymi w grze. Informacje te powinny zawierać szczegółowe dane o twórcach i właścicielach licencjonowanych treści, takich jak silniki graficzne, postacie, logotypy czy nazwy gier, a także wskazywać wszelkie obowiązujące warunki i ograniczenia dotyczące ich wykorzystania.
System powinien również spełniać wszystkie aktualne przepisy związane z ochroną danych osobowych użytkowników, w tym wymogi RODO, zapewniając odpowiedni poziom bezpieczeństwa danych graczy oraz przejrzystość polityki prywatności.

6. Wymagania dotyczące jakości modelowanego systemu

Rozdział ten zawiera szczegółowy opis wymagań dotyczących jakości modelowanego systemu określonych według modelu owych wymagań tj. niezawodność (reliability), dostępność (availability), użyteczność (usability), wydajność (efficiency), obsługiwalność (maintainability), bezpieczeństwo danych (data security), bezpieczne użycie (safety), przeniosność (portability).

Użyteczność (Usability) - System powinien charakteryzować się łatwością użycia i przejrzystością. Interfejs użytkownika musi być intuicyjny, aby gracze, niezależnie od doświadczenia, mogli szybko zrozumieć, jak zacząć grę, zapisać wyniki, czy zarządzać swoim profilem. Przejrzystość interfejsu pozwoli użytkownikom na łatwe przechodzenie między różnymi trybami gry, w tym "Kółko-krzyżyk", "Wisielec" oraz "Statki", bez potrzeby skomplikowanej nauki obsługi platformy.

Dostępność (Availability) - Platforma gier musi zapewniać, że dostęp do gier i funkcji jest możliwy w określonym czasie, zwłaszcza podczas sesji gry w trybie wieloosobowym. Gracze powinni mieć pewność, że mogą zagrać w grę w zaplanowanym czasie, a system będzie gotowy do rozpoczęcia nowych sesji, w tym dostępu do gier w czasie rzeczywistym.

Rejestracje do gier i sesje nie powinny kolidować, a czas oczekiwania na dostępność gry powinien być minimalny.

Bezpieczeństwo Danych (Data Security) - Wszystkie dane użytkowników, takie jak dane konta, statystyki, historia gier oraz wyniki, muszą być chronione przed nieautoryzowanym dostępem. Platforma powinna wykorzystywać zaawansowane metody szyfrowania do ochrony tych danych oraz zapewnić mechanizmy uwierzytelniania użytkowników, aby zapobiec nieautoryzowanemu dostępowi do konta. Ponadto, ważne jest, aby dane użytkowników były przechowywane zgodnie z obowiązującymi przepisami o ochronie danych.

Niezawodność (Reliability) - System musi być niezawodny podczas przeglądania dostępnych gier, rozpoczynania sesji, a także w trakcie rozgrywek. Aplikacja powinna działać stabilnie, bez awarii, błędów czy zawieszania się, zapewniając płynność rozgrywki. System musi być przygotowany na obsługę dużej liczby graczy w tym samym czasie, a wszelkie mechanizmy komunikacji, takie jak WebSocket, muszą działać bez zakłóceń, umożliwiając płynne interakcje między graczami w czasie rzeczywistym.

Obsługiwalność (Maintainability) - Platforma musi być łatwa do utrzymania, aby w przypadku pojawienia się jakichkolwiek problemów technicznych lub potrzeby rozwoju nowych funkcji, zespół wsparcia mógł szybko wprowadzić niezbędne poprawki. Kod źródłowy systemu musi być dobrze udokumentowany i łatwy do modyfikacji, co zapewni możliwość dodawania nowych gier, trybów rozgrywki lub funkcjonalności bez zakłócania istniejącego działania systemu.

Zgodność z Przepisami (Regulatory) - Wszystkie działania w ramach platformy gier, takie jak rejestracja użytkowników, przechowywanie wyników oraz danych osobowych, muszą być zgodne z obowiązującymi przepisami prawa dotyczącymi ochrony danych osobowych, w tym RODO. Platforma powinna umożliwiać użytkownikom łatwy dostęp do informacji o tym, jak ich dane są przechowywane i wykorzystywane, oraz zapewniać im możliwość wyrażenia zgody na przetwarzanie danych.

Przenośność (Portability) - Platforma powinna być przenośna i działać na różnych systemach operacyjnych, takich jak Windows oraz macOS, zapewniając graczom możliwość korzystania z aplikacji bez względu na to, jaki system operacyjny jest zainstalowany na ich urządzeniu. Aplikacja powinna być również dostosowana do różnych rozdzielczości ekranów oraz urządzeń, zapewniając płynne działanie na komputerach stacjonarnych, laptopach i innych urządzeniach.

7. Bazy danych

W systemie platformy gier, dla zapewnienia prawidłowego przechowywania danych użytkowników, wyników gier, statystyk oraz historii rozgrywek, muszą zostać zaimplementowane relacyjne bazy danych. Baza danych powinna być odpowiedzialna za przechowywanie informacji o użytkownikach, ich profilach, historii gier, wynikach, a także o sesjach gry. Ponadto, system musi zapewniać odpowiednią integralność danych, ich

bezpieczeństwo oraz optymalizację pod kątem wydajności w przypadku dużej liczby użytkowników.

8. Stosowane standardy

Aplikacja powinna być zgodna ze stosowanymi standardami przemysłowymi i technologicznymi.

9. Opis więzów

Zgodność z systemami operacyjnymi - Aplikacja powinna być zaprojektowana i rozwinięta w sposób zapewniający jej pełną kompatybilność z systemami operacyjnymi Windows oraz macOS. Oznacza to, że użytkownicy korzystający z tych platform powinni móc korzystać z aplikacji bez jakichkolwiek problemów technicznych, takich jak błędy w działaniu interfejsu czy problemy z wydajnością. Aplikacja musi być również przetestowana na różnych wersjach tych systemów, aby zapewnić optymalną kompatybilność.
Dokumentacja systemu - Dokumentacja, czyli zarówno specyfikacje techniczne, jak i dokumentacja użytkowa, musi być dostępna w dwóch językach: polskim i angielskim. Zapewnienie dwujęzycznej dokumentacji jest kluczowe, aby użytkownicy z różnych krajów, językami mogli w pełni zrozumieć, jak korzystać z systemu oraz jakie funkcje oferuje. Dokumentacja powinna zawierać instrukcje obsługi, opis funkcji aplikacji, jak również wskazówki dotyczące konfiguracji i rozwiązywania problemów.

Weryfikacja wymagań systemowych i funkcjonalnych - realizowana będzie w oparciu o zdefiniowane scenariusze testowe. Scenariusze testowe muszą być dokładnie opracowane przed rozpoczęciem testów, aby zapewnić, że wszystkie funkcjonalności systemu działają zgodnie z oczekiwaniami. Testy będą obejmować zarówno testy funkcjonalne, jak i niefunkcjonalne, w tym testy wydajnościowe, bezpieczeństwa oraz kompatybilności. Każdy scenariusz testowy powinien zawierać jasno określone kroki, dane wejściowe, oczekiwane wyniki oraz warunki zakończenia testu.

Ograniczony dostęp do bazy danych - Baza danych systemu musi być odpowiednio zabezpieczona i nie może być dostępna dla wszystkich użytkowników aplikacji. Dostęp do bazy danych powinien być ograniczony tylko do autoryzowanych użytkowników i procesów systemowych, w celu zapobieżenia nieautoryzowanemu dostępowi do wrażliwych danych. Tylko uprawnieni administratorzy oraz systemy, które pełnią określone funkcje (np. generowanie raportów, monitorowanie działania systemu), powinni mieć dostęp do danych przechowywanych w bazie. Poza tym odpowiednie pola w bazie danych, takie jak hasła użytkowników powinny być zahaszowane.