Analiza specyfikacji wymagań

Sklep z grami komputerowymi

Wersja 2024-04-16

|  |  |  |
| --- | --- | --- |
|  | **Imię nazwisko** | **Data** |
| Autor: | Adam Bień | 25.06.2024r. |
| Autor: | Patryk Słomian | 25.06.2024r. |

Spis treści

[1. Wstęp 3](#_Toc168770104)

[2. Cel analizy specyfikacji 3](#_Toc168770105)

[3. Zakres specyfikacji 3](#_Toc168770106)

[4. Analiza wymagań funkcjonalnych – model usecase’ów 3](#_Toc168770107)

[4.1. Specyfikacja aktorów 3](#_Toc168770108)

[4.2. Lista usecase’ów 3](#_Toc168770109)

[4.3. Diagram usecase’ów UML 3](#_Toc168770110)

[4.4. Specyfikacja usecase’ów 3](#_Toc168770111)

[4.5. Diagramy aktywności UML dla usecase’ów 3](#_Toc168770112)

[5. Analiza wymagań niefunkcjonalnych 3](#_Toc168770113)

[5.1. Interfejsy użytkownika 3](#_Toc168770114)

[5.2. Interfejsy sprzętowe 3](#_Toc168770115)

[5.3. Interfejsy komunikacyjne 3](#_Toc168770116)

[5.4. Interfejsy programowe 3](#_Toc168770117)

[6. Analiza wymagań dotyczących jakości modelowego systemu 3](#_Toc168770118)

[7. Analiza serwisowania 3](#_Toc168770119)

[8. Analiza ograniczeń architektury systemu 3](#_Toc168770120)

[9. Model bazy danych 3](#_Toc168770121)

[9.1. Specyfikacja atrybutów bazy danych 3](#_Toc168770122)

[9.2. Specyfikacja operacji na danych 3](#_Toc168770123)

[9.3. Specyfikacja reguł poprawności i zgodności typów danych 3](#_Toc168770124)

# Wstęp

Specyfikacja wymagań dotyczy aplikacji "Sklep z grami komputerowymi" i opisuje funkcjonalności, interfejsy oraz wymagania dotyczące jakości systemu. Celem dokumentu jest dostarczenie wytycznych dla zespołu programistycznego oraz interesariuszy projektu, aby zapewnić prawidłowe zrozumienie i realizację wymagań projektu.

# Cel analizy specyfikacji

Celem analizy specyfikacji wymagań jest szczegółowe zrozumienie i ocena zdefiniowanych wymagań funkcjonalnych, niefunkcjonalnych oraz jakościowych. Analiza ta ma na celu zapewnienie, że wszystkie wymagania są jasne, spójne i kompletne oraz że system spełni oczekiwania klienta i użytkowników końcowych.

# Zakres specyfikacji

Specyfikacja obejmuje:

* Wymagania funkcjonalne, takie jak zarządzanie użytkownikami, przeglądanie katalogu gier, dokonywanie zakupów i obsługę płatności.
* Wymagania niefunkcjonalne, w tym interfejsy użytkownika, interfejsy sprzętowe, komunikacyjne i programowe.
* Wymagania dotyczące serwisowania, licencji oraz jakości systemu, takie jak wydajność, niezawodność, skalowalność i bezpieczeństwo.
* Model bazy danych, w tym strukturę i operacje na danych.

# Analiza wymagań funkcjonalnych – model usecase’ów

# Specyfikacja aktorów

* **Użytkownik (User):** Może przeglądać ofertę, wyszukiwać gry, dodawać gry do koszyka, finalizować zakupy oraz zarządzać swoim kontem.
* **Administrator (Admin):** Może zarządzać asortymentem sklepu, dodawać nowe gry, usuwać nieaktualne gry, zarządzać promocjami i użytkownikami.
* **System Płatności (Payment System):** Zewnętrzny system odpowiedzialny za realizację transakcji finansowych.
* **System Autoryzacji:** Zewnętrzny system odpowiedzialny za autoryzację użytkowników, np. logowanie, rejestracja.

# Lista usecase’ów

* + 1. Zakup gry
    2. Dodanie Nowej Gry do Sklepu
    3. Pobranie opłaty

# Diagram usecase’ów UML

Diagram use case’ów UML przedstawia aktorów i przypadki użycia systemu, ilustrując relacje pomiędzy nimi. (Ze względu na brak możliwości wizualizacji bezpośrednio w tym dokumencie, diagram ten powinien być stworzony w narzędziu UML takim jak StarUML, Visual Paradigm lub inne).

Załącznik – Use\_Case.docx

# Specyfikacja usecase’ów

Każdy przypadek użycia jest opisany w formacie:

* Nazwa: Opis przypadku użycia.
* Aktorzy: Użytkownicy biorący udział w danym przypadku użycia.
* Opis: Krótkie wprowadzenie do przypadku użycia.
* Przebieg główny: Krok po kroku opis przebiegu.
* Przebieg alternatywny: Opis przypadków, które mogą wystąpić w trakcie głównego scenariusza.

Na przykład:

* Use Case: Zakup gry
  + Aktorzy: Użytkownik
  + Opis: Użytkownik dodaje wybraną grę do koszyka i dokonuje zakupu.
  + Przebieg główny:

Użytkownik przegląda katalog gier.

Użytkownik wybiera grę, którą chce dodać do koszyka.

Użytkownik klika przycisk "Dodaj do koszyka".

System dodaje grę do koszyka i wyświetla potwierdzenie.

Klient przechodzi do procesu płatności.

System prezentuje dostępne metody płatności.

Klient wybiera metodę płatności i wprowadza wymagane dane.

System Płatności weryfikuje transakcję.

Po pomyślnej weryfikacji, gra zostaje dodana do konta Klienta.

Klient otrzymuje potwierdzenie zakupu.

* + Przebieg alternatywny:
    - Brak środków na koncie użytkownika: System Płatności odrzuca transakcję z powodu braku środków; użytkownik otrzymuje komunikat o błędzie i może wybrać inną metodę płatności lub anulować zakup.

# Diagramy aktywności UML dla usecase’ów

Diagramy aktywności UML ilustrują szczegółowy przepływ działań dla każdego z przypadków użycia. Dla przykładu, diagram aktywności dla "Finalizacja zakupu" zawiera kroki od momentu dodania gry do koszyka, przez wybór metody płatności, aż po otrzymanie potwierdzenia zakupu.

Załącznik – Use\_Case.docx

# Analiza wymagań niefunkcjonalnych

# Interfejsy użytkownika

* Aplikacja musi mieć intuicyjny i responsywny interfejs użytkownika, dostosowany do różnych urządzeń (desktop, tablet, smartfon).
* Interfejs oparty na wzorcach UX/UI, z łatwo dostępnymi funkcjami nawigacyjnymi.

# Interfejsy sprzętowe

* Obsługa aplikacji na standardowych przeglądarkach internetowych (Chrome, Firefox, Safari, Edge).
* Wsparcie dla urządzeń z systemami Windows, macOS, Linux, Android i iOS.

# Interfejsy komunikacyjne

* Aplikacja musi obsługiwać API RESTful do integracji z zewnętrznymi systemami płatności i innymi usługami.
* Wymagane jest szyfrowanie danych przesyłanych przez sieć z wykorzystaniem protokołu HTTPS.

# Interfejsy programowe

* Dokumentacja API dla deweloperów umożliwiająca łatwą integrację zewnętrznych usług i aplikacji.
* Modułowa architektura umożliwiająca łatwe dodawanie nowych funkcjonalności.

1. Zastosowane algorytmy i interfejsy programistyczne (API)

* WebAPI

Aplikacja korzysta z interfejsu WebAPI do realizacji komunikacji między serwerem a aplikacją webową. WebAPI obsługuje operacje CRUD (Create, Read, Update, Delete) dla elementów, takich jak produkty, użytkownicy, zamówienia, itd.

* Obliczanie przeceny/promocji

W systemie stosowany jest algorytm obliczający procentową obniżkę ceny produktu. Przecena obliczana jest na podstawie procentu podanego przez administratora i odjęcia tego procentu od ceny bazowej produktu.

* Operacje na bazie danych

System obsługuje standardowe operacje na bazie danych, takie jak:

* + **Wstawianie:** Dodawanie nowych produktów, użytkowników i zamówień do bazy danych.
  + **Usuwanie:** Usuwanie elementów z bazy danych na podstawie unikalnego identyfikatora.
  + **Aktualizowanie:** Modyfikowanie istniejących rekordów, np. aktualizacja danych użytkownika lub zmiana ceny produktu.
  + **Wczytywanie:** Pobieranie danych z bazy danych, np. w celu wyświetlenia produktów w sklepie.
* Walidacja promocji

System zawiera mechanizm walidacji wartości procentowej promocji. Wartość ta musi mieścić się w przedziale 1-100%. Wprowadzona wartość jest sprawdzana przed zapisem do bazy danych.

* + Walidacja:

**Zasady:** Tylko wartości między 1 a 100 są akceptowane. Jeśli wprowadzona wartość jest spoza tego zakresu, użytkownik otrzymuje komunikat o błędzie.

* Sprawdzanie poprawności danych wejściowych

Przy wprowadzaniu ceny produktu system weryfikuje, czy wprowadzone dane są liczbą. W przypadku, gdy użytkownik poda nieprawidłową wartość (np. tekst zamiast liczby), zostanie wyświetlony odpowiedni komunikat o błędzie.

# Analiza wymagań dotyczących jakości modelowego systemu

* **Wydajność:** Obsługa do 1000 jednoczesnych użytkowników.
* **Niezawodność:** Średni czas bezawaryjnej pracy (MTBF) minimum 999% w skali miesiąca.
* **Skalowalność:** Aplikacja musi być łatwo skalowalna.
* **Bezpieczeństwo:** Dane użytkowników muszą być chronione przed nieautoryzowanym dostępem.

# Analiza serwisowania

* Dostępność wsparcia technicznego dla użytkowników w trybie 24/7.
* System zgłoszeń serwisowych dostępny z poziomu aplikacji oraz strony internetowej.

# Analiza ograniczeń architektury systemu

* Aplikacja będzie działać w chmurze, korzystając z węzłów serwerowych skonfigurowanych w środowisku AWS lub Azure.
* Konfiguracja węzłów obejmuje redundantne serwery aplikacji, serwery baz danych oraz systemy do zarządzania pamięcią podręczną i równoważenia obciążenia.

# Model bazy danych

# Specyfikacja atrybutów bazy danych

* Baza danych będzie przechowywać informacje o użytkownikach, grach, zamówieniach i płatnościach.
* Struktura bazy danych oparta na zasadach normalizacji zapewniających spójność i integralność danych.

# Specyfikacja operacji na danych

* Typowe operacje obejmują dodawanie, aktualizowanie, usuwanie i wyszukiwanie danych.

# Specyfikacja reguł poprawności i zgodności typów danych

* Reguły integralności referencyjnej i typy danych dostosowane do przechowywania różnych typów informacji (np. tekst, liczby, daty).