Plan projektu

Sklep z grami komputerowymi

Wersja 2024-04-16

|  |  |  |
| --- | --- | --- |
|  | **Imię nazwisko** | **Data** |
| Autor: | Adam Bień | 16.04.2024r. |
| Autor: | Patryk Słomian | 16.04.2024r. |

Spis treści

[**Sklep z grami komputerowymi** 7](#_Toc174891346)

[1. Wstęp 7](#_Toc174891347)

[2. Opis projektu 7](#_Toc174891349)

[3. Struktury organizacyjne projektu 7](#_Toc174891351)

[4. Role i odpowiedzialność 9](#_Toc174891352)

[5. Harmonogram projektu 12](#_Toc174891437)

[6. Kosztorys projektu 12](#_Toc174891438)

[7. Zasoby projektu 12](#_Toc174891451)

[8. Standardy i narzędzia w projekcie 13](#_Toc174891474)

[9. Procesy zarządzania 13](#_Toc174891483)

[9.1. Plan zarządzania konfiguracją 13](#_Toc174891484)

[9.2. Plan zarządzania ryzykiem 14](#_Toc174891485)

[9.3. Plan zarządzania testami 14](#_Toc174891486)

[9.4. Plan zapewnienia jakości 14](#_Toc174891487)

[9.5. Plan zarządzania zmianami 14](#_Toc174891488)

**Sklep z grami komputerowymi**

1. Wstęp

Celem projektu jest stworzenie zaawansowanego sklepu z grami komputerowymi, który umożliwi użytkownikom przeglądanie, wyszukiwanie oraz zakup gier komputerowych. Program zostanie napisany w języku C# i będzie zawierał zarówno interfejs użytkownika, jak i backend do zarządzania danymi gier oraz transakcjami.

1. Opis projektu

Projekt "Sklep z grami komputerowymi" ma na celu stworzenie nowoczesnej i intuicyjnej platformy zakupowej dla miłośników gier komputerowych. Platforma będzie umożliwiać użytkownikom przeglądanie oferty gier, wyszukiwanie według różnych kryteriów, dodawanie gier do koszyka oraz finalizowanie zakupu w sposób szybki i bezpieczny. Projekt będzie zrealizowany z wykorzystaniem języka programowania C# oraz technologii ASP.NET Core. Projekt ten dostarczy użytkownikom wygodne narzędzie do zakupu gier, a administratorom umożliwi efektywne zarządzanie asortymentem. Dzięki zastosowaniu nowoczesnych technologii i najlepszych praktyk programistycznych, system będzie wydajny, bezpieczny i łatwy w utrzymaniu, co zapewni jego długoterminowy rozwój i możliwość rozbudowy o nowe funkcjonalności w przyszłości.

1. Struktury organizacyjne projektu

**Zespoły w projekcie**

1. Zespół Zarządzający

* Project Manager (PM): Odpowiada za planowanie, organizację i nadzór nad całością projektu. Koordynuje pracę wszystkich zespołów, ustala priorytety i harmonogramy, zarządza ryzykami oraz budżetem projektu.
* Product Owner (PO): Odpowiada za wizję produktu, zbieranie wymagań od interesariuszy oraz priorytetyzację zadań w backlogu. Zapewnia, że zespół pracuje nad najbardziej wartościowymi funkcjonalnościami.

1. Zespół Analityczny

* Analityk Biznesowy: Zbiera i analizuje wymagania funkcjonalne oraz niefunkcjonalne, tworzy dokumentację wymagań oraz specyfikacje systemu. Współpracuje z Product Ownerem w celu zrozumienia potrzeb użytkowników końcowych.

1. Zespół Programistyczny

* Lead Developer: Kieruje zespołem programistów, podejmuje kluczowe decyzje technologiczne i architektoniczne, zapewnia zgodność kodu z najlepszymi praktykami i standardami.
* Backend Developers: Odpowiadają za implementację logiki biznesowej, integrację z bazą danych, implementację API oraz zarządzanie bezpieczeństwem aplikacji.
* Frontend Developers: Zajmują się tworzeniem interfejsu użytkownika, implementacją komponentów interfejsu, zapewnieniem responsywności i optymalizacji wydajności.

1. Zespół Testowy

* Test Manager: Koordynuje prace zespołu testowego, opracowuje strategię testowania, harmonogramy i zarządza raportowaniem wyników testów.
* Testerzy: Przeprowadzają testy funkcjonalne, niefunkcjonalne, regresyjne, integracyjne oraz akceptacyjne. Tworzą scenariusze testowe i raportują błędy.

1. Zespół DevOps

* DevOps Engineer: Odpowiada za automatyzację procesów wdrożeniowych, zarządzanie środowiskami developerskimi, testowymi i produkcyjnymi oraz monitorowanie wydajności i dostępności aplikacji.

1. Zespół Wsparcia Technicznego

* Support Specialist: Udziela wsparcia użytkownikom końcowym, rozwiązuje zgłaszane problemy techniczne oraz prowadzi dokumentację zgłoszeń.

**Przepływ danych między zespołami**

1. Zespół Zarządzający → Zespół Analityczny

* Przekazywanie wizji produktu, głównych celów biznesowych oraz kluczowych wymagań.

1. Zespół Analityczny → Zespół Programistyczny

* Dokumentacja wymagań, specyfikacje techniczne, przypadki użycia oraz diagramy procesów.

1. Zespół Programistyczny → Zespół Testowy

* Implementacja kodu źródłowego, specyfikacja techniczna oraz informacje o nowych funkcjonalnościach do testowania.

1. Zespół Programistyczny → Zespół DevOps

* Kod aplikacji, skrypty wdrożeniowe, konfiguracje środowisk.

1. Zespół Testowy → Zespół Programistyczny

* Raporty z testów, zgłoszenia błędów, sugestie poprawy kodu.

1. Zespół DevOps → Zespół Programistyczny/Zespół Testowy

* Feedback dotyczący wydajności i stabilności aplikacji, problemy z wdrożeniem.

1. Zespół Wsparcia Technicznego → Zespół Programistyczny

* Zgłoszenia problemów od użytkowników, propozycje poprawek i nowych funkcjonalności.

**Zasady organizacji spotkań**

1. Codzienne spotkania zespołów (Daily Stand-up)
   * Czas: 15 minut
   * Uczestnicy: Wszystkie zespoły
   * Cel: Omówienie postępów, planów na dany dzień oraz ewentualnych blokad.
2. Tygodniowe spotkania zespołów (Sprint Planning)
   * Czas: 1-2 godziny
   * Uczestnicy: Zespół Zarządzający, Zespół Programistyczny, Zespół Testowy
   * Cel: Planowanie zadań na nadchodzący sprint, priorytetyzacja backlogu.
3. Dwutygodniowe spotkania przeglądowe (Sprint Review)
   * Czas: 1-2 godziny
   * Uczestnicy: Wszystkie zespoły, interesariusze
   * Cel: Przegląd ukończonych prac, demonstracja nowych funkcjonalności, zebranie feedbacku.
4. Retrospektywa sprintu
   * Czas: 1 godzina
   * Uczestnicy: Zespół Programistyczny, Zespół Testowy, Project Manager
   * Cel: Omówienie co poszło dobrze, co można poprawić, planowanie usprawnień.
5. Spotkania ad-hoc
   * Uczestnicy: W zależności od potrzeby
   * Cel: Rozwiązywanie nagłych problemów, omawianie kluczowych decyzji technicznych.
6. Miesięczne spotkania strategiczne
   * Czas: 2 godziny
   * Uczestnicy: Zespół Zarządzający, Lead Developer, Test Manager
   * Cel: Przegląd postępów projektu, analiza ryzyk, planowanie długoterminowych działań.
7. Role i odpowiedzialność

**Role w projekcie**

1. Project Manager (PM)

* Zakres obowiązków:
  + Planowanie i organizacja projektu
  + Nadzór nad realizacją zadań
  + Zarządzanie harmonogramem i budżetem
  + Koordynacja prac zespołów
  + Zarządzanie ryzykiem
* Kompetencje:
* Doświadczenie w zarządzaniu projektami IT
* Umiejętności organizacyjne i przywódcze
* Znajomość metodologii zarządzania projektami
* Doskonałe umiejętności komunikacyjne

1. Product Owner (PO)

* Zakres obowiązków:
  + Definiowanie wizji produktu
  + Zbieranie i priorytetyzacja wymagań
  + Współpraca z interesariuszami
* Kompetencje:
  + Znajomość rynku gier komputerowych
  + Umiejętność analizy potrzeb biznesowych
  + Silne umiejętności komunikacyjne i negocjacyjne
  + Doświadczenie w pracy z zespołami deweloperskimi

1. Analityk Biznesowy

* Zakres obowiązków:
  + Zbieranie i dokumentowanie wymagań
  + Tworzenie specyfikacji funkcjonalnych
  + Współpraca z Product Ownerem i zespołem programistycznym
* Kompetencje:
  + Umiejętność analizy i rozwiązywania problemów
  + Doskonałe umiejętności dokumentacyjne
  + Doświadczenie w analizie systemów IT

1. Lead Developer

* Zakres obowiązków:
  + Kierowanie zespołem programistycznym
  + Podejmowanie decyzji technologicznych i architektonicznych
  + Przegląd i kontrola jakości kodu
* Kompetencje:
  + Zaawansowana znajomość C# i ASP.NET Core
  + Doświadczenie w projektowaniu systemów informatycznych
  + Znajomość wzorców projektowych

1. Backend Developer

* Zakres obowiązków:
  + Implementacja logiki biznesowej
  + Integracja z bazą danych
  + Optymalizacja wydajności backendu
  + Tworzenie i utrzymanie dokumentacji technicznej
* Kompetencje:
  + Bardzo dobra znajomość C# i ASP.NET Core
  + Doświadczenie w pracy z bazami danych
  + Umiejętność rozwiązywania problemów technicznych
  + Znajomość narzędzi do zarządzania wersjami

1. Frontend Developer

* Zakres obowiązków:
  + Tworzenie interfejsu użytkownika
  + Implementacja komponentów frontendowych
  + Zapewnienie responsywności i użyteczności aplikacji
  + Współpraca z zespołem UX/UI
* Kompetencje:
  + Znajomość technologii frontendowych (WPF, WinForms)
  + Doświadczenie w tworzeniu aplikacji webowych
  + Umiejętność pracy z narzędziami do kontroli wersji

1. Test Manager

* Zakres obowiązków:
  + Opracowanie strategii testowania
  + Koordynacja prac zespołu testowego
  + Zarządzanie harmonogramem testów
  + Raportowanie wyników testów
* Kompetencje:
  + Doświadczenie w zarządzaniu testami w projektach IT
  + Znajomość narzędzi i technik testowania
  + Umiejętność analizy i raportowania wyników testów
  + Doskonałe umiejętności komunikacyjne

1. Tester

* Zakres obowiązków:
  + Przeprowadzanie testów funkcjonalnych, niefunkcjonalnych, regresyjnych i integracyjnych
  + Tworzenie scenariuszy testowych
  + Zgłaszanie i śledzenie błędów
  + Współpraca z zespołem programistycznym w celu poprawy jakości kodu
* Kompetencje:
  + Doświadczenie w testowaniu aplikacji webowych i desktopowych
  + Znajomość narzędzi do zarządzania testami
  + Umiejętność tworzenia dokumentacji testowej
  + Znajomość podstaw programowania

**Zasady dokumentowania projektu**

|  |  |
| --- | --- |
| Nazwa dokumentu - szablon | Osoba wypełniająca dokument |
| Plan Projektu | Project Manager |
| Harmonogram projektu | Project Manager |
| Specyfikacja wymagań | Analityk biznesowy, Klient – Product Owner |
| Analiza specyfikacja wymagań | Analityk biznesowy |
| Projekt aplikacji | Lead Developer |
| Struktura use case’a | Analityk biznesowy |
| Analiza ryzyka projektu | Project Manager |
| Opis wersji systemu | Klient – Product Owner |
| Formularz test case’u | Tester |
| Formularz błędu | Tester |

1. Harmonogram projektu

|  |  |  |  |
| --- | --- | --- | --- |
| Nazwa grupy zadań | Nazwa zadania | Artefakt zadania  (owoce pracy) | Rola w projekcie  realizująca zadanie |
| Zarządzanie projektem | Przygotowanie planu projektu | Plan projektu | Kierownik Projektu |
| Przygotowanie harmonogramu projektu | Harmonogram projektu | Kierownik Projektu |
| Przygotowanie kosztorysu projektu | Kosztorys projektu | Kierownik Projektu, Analityk |
| Napisanie planu testów | Plan testów | Tester |
| Przygotowanie planu zarządzanie ryzykiem | Plan zarządzania ryzykiem | Kierownik Projektu, Specjalista ds. Zarzadzania Ryzykiem |
| Przygotowanie planu zarządzania konfiguracją | Plan zarządzania konfiguracją | Kierownik Projektu, Specjalista ds. Konfiguracji |
| Akceptacja planów | Zatwierdzone plany projektu | Kierownik Projektu |
| Przygotowanie specyfikacji wymagań | Zbieranie wymagań | Lista wymagań | Analityk |
| Dokumentacja wymagań | Specyfikacja wymagań | Analityk |
| Przegląd wymagań z klientem | Zatwierdzona specyfikacja wymagań | Analityk, Klient |
| Analiza specyfikacji wymagań | Analiza techniczna wymagań | Raport z analizy technicznej | Architekt systemu, Analityk |
| Przegląd analizy technicznej | Zatwierdzony raport z analizy technicznej | Architekt systemu, Kierownik Projektu |
| Przygotowanie projektu aplikacji | Projektowanie architektury systemu | Dokumentacja architektury | Architekt systemu |
| Projektowanie interfejsu użytkownika | Specyfikacja UI | Projektant UI |
| Projektowanie bazy danych | Model bazy danych | Inżynier baz danych |
| Przegląd projektu aplikacji | Zatwierdzona dokumentacja projektu | Architekt systemu, Kierownik Projektu |
| Implementacja aplikacji | Implementacja modułów aplikacji | Kod źródłowy | Programista |
| Integracja modułów | Zintegrowania aplikacja | Programista |
| Kodowanie testów jednostkowych | Skrypty testów jednostkowych | Programista, Tester |
| Testy aplikacji | Przygotowanie środowiska testowego | Skonfigurowane środowisko testowe | Tester |
| Testy funkcjonalne | Raport z testów funkcjonalnych | Tester |
| Testy wydajnościowe | Raport z testów wydajnościowych | Tester |
| Testy akceptacyjne | Raport z testów akceptacyjnych, zatwierdzona aplikacja | Tester, Klient |

1. Kosztorys projektu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Rola | Średnie godziny pracy dziennie | Liczba dni pracy | Łączny nakład godzinowy | Stawka godzinowa (PLN) | Łączny koszt (PLN) |
| Project Manager (PM) | 6 | 60 | 360 | 150 | 54 000 |
| Product Owner (PO) | 4 | 50 | 200 | 180 | 36 000 |
| Analityk Biznesowy | 7 | 45 | 315 | 130 | 40 950 |
| Lead Developer | 8 | 90 | 720 | 200 | 144 00 |
| Backend Developer | 8 | 85 | 680 | 160 | 108 800 |
| Frontend Developer | 7 | 80 | 560 | 160 | 89 600 |
| Test Manager | 6 | 50 | 300 | 140 | 42 000 |
| Tester | 7 | 60 | 420 | 120 | 50 400 |

1. Zasoby projektu
2. Zasoby sprzętowe

* Komputery dla zespołów programistycznych
* Specyfikacja minimalna:
  + Procesor: AMD Ryzen 5 3500U
  + RAM: 8 GB
  + Dysk SSD: 512 GB
  + Karta graficzna: Zintegrowana lub dedykowana (np. AMD Radeon Vega 8 Graphics)
  + System operacyjny: Windows 10/11

1. Inne zasoby sprzętowe

* Monitory: Dla każdego członka zespołu programistycznego i testowego minimum dwa monitory, rozdzielczość Full HD.
* Urządzenia peryferyjne: Klawiatury, myszy.

1. Licencje oprogramowania

* Środowisko IDE
  + Visual Studio: Licencje dla wszystkich deweloperów (preferowana wersja Enterprise dla zaawansowanych funkcji).
* Systemy operacyjne
  + Windows 11 Home: Dla komputerów programistycznych i testowych.
* Bazy danych
  + Microsoft SQL Server: Licencje na odpowiednią edycję (np. Standard lub Enterprise) dla środowisk deweloperskich, testowych i produkcyjnych.
  + SQL Server Management Studio (SSMS): Narzędzie do zarządzania bazami danych.
* Narzędzia do kontroli wersji
  + Git: Oprogramowanie open-source do kontroli wersji.
  + GitHub/GitLab: Licencje na wersje korporacyjne, jeśli używane są prywatne repozytoria.

1. Standardy i narzędzia w projekcie

**Narzędzia**

1. Środowisko programistyczne (IDE)

* Visual Studio 2022: Główne środowisko IDE używane przez programistów do tworzenia aplikacji w C#.

1. System operacyjny

* Windows 11 Home: System operacyjny dla komputerów programistycznych.

1. Narzędzia do komunikacji

* Microsoft Teams: Platforma do komunikacji i współpracy.
* Discord: Alternatywne narzędzie do komunikacji zespołowej.

1. Procesy zarządzania
   1. Plan zarządzania konfiguracją

Plan zarządzania konfiguracją opisuje procesy związane z kontrolą, rejestrowaniem i śledzeniem zmian w dokumentacji projektowej, kodzie źródłowym, środowiskach testowych i produkcyjnych oraz innych elementach systemu. Główne działania w ramach tego procesu obejmują:

* **Identyfikacja konfiguracji:** Ustalanie, które elementy systemu podlegają kontroli wersji.
* **Kontrola wersji:** Zarządzanie wersjami komponentów i dokumentów, w tym nadawanie unikalnych numerów wersji oraz przechowywanie poprzednich wersji.
* **Audyt konfiguracji:** Regularne przeglądy i audyty w celu zapewnienia zgodności komponentów z założeniami projektu.
* **Raportowanie konfiguracji:** Tworzenie raportów opisujących stan i historię zmian elementów podlegających kontroli konfiguracji.
  1. Plan zarządzania ryzykiem

Plan zarządzania ryzykiem definiuje procesy identyfikacji, oceny, monitorowania i zarządzania ryzykami, które mogą mieć wpływ na projekt. Główne elementy tego procesu to:

* **Identyfikacja ryzyk:** Rozpoznanie i opisanie potencjalnych zagrożeń dla projektu.
* **Ocena ryzyk:** Analiza prawdopodobieństwa wystąpienia oraz wpływu ryzyka na projekt.
* **Planowanie reakcji na ryzyko:** Określenie strategii działania w przypadku wystąpienia ryzyka, np. unikanie, łagodzenie, przeniesienie lub akceptacja ryzyka.
* **Monitorowanie ryzyk:** Regularne przeglądy ryzyk i aktualizacja planów działania w odpowiedzi na zmieniające się warunki projektowe.
  1. Plan zarządzania testami

Plan zarządzania testami opisuje podejście do testowania systemu, określając zakres, harmonogram, zasoby i metody testowania. Kluczowe procesy obejmują:

* **Planowanie testów:** Określenie zakresu testów, kryteriów wejścia i wyjścia, oraz harmonogramu testów.
* **Projektowanie testów:** Tworzenie przypadków testowych, scenariuszy testowych oraz przygotowanie środowiska testowego.
* **Wykonanie testów:** Przeprowadzenie testów zgodnie z planem, dokumentowanie wyników i rejestrowanie błędów.
* **Zarządzanie defektami:** Identyfikacja, raportowanie, priorytetyzacja i śledzenie naprawy błędów.
* **Ocena testów:** Analiza wyników testów w celu weryfikacji, czy system spełnia założenia i jest gotowy do wdrożenia.
  1. Plan zapewnienia jakości

Plan zapewnienia jakości opisuje działania mające na celu zapewnienie, że produkt końcowy spełnia wymagania jakościowe. Procesy te obejmują:

* **Definicja standardów jakości:** Określenie wymagań jakościowych oraz standardów, które muszą być spełnione.
* **Przeglądy i audyty:** Regularne przeglądy dokumentacji, kodu i procesów w celu zapewnienia zgodności z ustalonymi standardami.
* **Testy jakościowe:** Weryfikacja, czy produkt końcowy spełnia określone kryteria jakościowe.
* **Zarządzanie defektami jakości:** Identyfikacja i naprawa problemów związanych z jakością produktu na wczesnym etapie.
  1. Plan zarządzania zmianami

Plan zarządzania zmianami definiuje procedury, które są stosowane w celu kontrolowania i zatwierdzania zmian w projekcie. Procesy zarządzania zmianami obejmują:

* **Identyfikacja zmian:** Zgłaszanie i dokumentowanie proponowanych zmian w projekcie.
* **Ocena zmian:** Analiza wpływu zmiany na zakres, budżet, harmonogram i jakość projektu.
* **Zatwierdzanie zmian:** Przegląd i zatwierdzenie lub odrzucenie zmian przez komitet sterujący lub inne odpowiedzialne osoby.
* **Wdrażanie zmian:** Implementacja zatwierdzonych zmian oraz aktualizacja dokumentacji projektowej.
* **Monitorowanie zmian:** Śledzenie wdrożonych zmian i ocena ich wpływu na projekt.