**Dokumentacja projektu Cryland**

**1. Opis projektu**

Cryland to tekstowa gra RPG napisana w PHP, która pozwala graczom na zanurzenie się w fabularnym świecie, wykonując różnorodne zadania i zdobywając doświadczenie.

**2. Cele projektu**

1. **Zaprojektowanie i wdrożenie systemu zarządzania użytkownikami**: Umożliwienie graczom rejestrację, logowanie i zarządzanie swoimi kontami.
2. **Stworzenie interfejsu gry**: Zapewnienie intuicyjnego i responsywnego interfejsu użytkownika do interakcji z grą.

**3. Interesariusze**

1. **Gracze**: Osoby korzystające z gry w celu rozrywki.
2. **Administratorzy**: Osoby zarządzające grą i dbające o jej prawidłowe funkcjonowanie.

**4. Aktorzy i obiekty systemu**

**Aktorzy**:

1. Gracz
2. Administrator

**Obiekty**:

1. Konto użytkownika
2. Postać w grze
3. Lokacje
4. Ekwipunek
5. Ranking

**5. Wymagania funkcjonalne**

1. **Rejestracja użytkownika**: Umożliwia nowym graczom założenie konta.
2. **Logowanie użytkownika**: Pozwala zarejestrowanym użytkownikom na zalogowanie się do systemu.
3. **Eksploracja i walka z potworami**: Możliwość walki z potworami w celu zdobycia złota i doświadczenia.
4. **Walka z bossami**: Specjalnie zaprojektowane walki o większym poziomie trudności.
5. **Zarządzanie postacią**: Gracze mogą przeglądać i zarządzać swoimi statystykami oraz swoim ekwipunkiem.
6. **Wyświetlanie rankingu**: System wyświetla ranking graczy.
7. **Zarządzanie grą**: Administratorzy mogą zarządzać dowolną zmienną w grze opartą o bazę danych (użytkownicy, konta, przedmioty, potwory, bossy).

**6. Wymagania niefunkcjonalne**

1. **Wydajność**: System powinien obsługiwać co najmniej 100 jednoczesnych użytkowników bez zauważalnych opóźnień.
2. **Bezpieczeństwo**: Dane użytkowników powinny być zabezpieczone przed nieautoryzowanym dostępem za pomocą użycia funkcji hashujących.

**7. Diagram kontenerów**

Diagram kontenerów pokazuje główne elementy systemu oraz sposób, w jaki są one rozmieszczone i połączone. Ponieważ aplikacja Cryland działa w środowisku serwera WWW, diagram kontenerów będzie zawierał następujące elementy:

* **Przeglądarka użytkownika**: Interfejs użytkownika dostępny przez przeglądarkę internetową.
* **Serwer WWW (Apache)**: Obsługuje żądania HTTP i serwuje pliki PHP oraz inne zasoby.
* **Aplikacja PHP**: Skrypty PHP, które przetwarzają logikę gry.
* **Baza danych MySQL**: Przechowuje dane użytkowników, rodzaje przeciwników, ekwipunek, rankingi itp.

**Diagram Kontenerów:**

+--------------------+

| Przeglądarka |

| |

| - HTML/CSS/JS |

+---------+----------+

|

| HTTP

|

+---------v----------+

| Serwer WWW (Apache)|

| |

| - PHP |

| - index.php |

| - login.php |

+---------+----------+

|

| PHP

|

+---------v----------+

| Aplikacja PHP |

| |

| - Moduły gry |

| - Zarządzanie |

| użytkownikami |

+---------+----------+

|

| SQL

|

+---------v----------+

| Baza danych MySQL |

| |

| - Tabele użytkowników|

| - Tabele zadań |

| - Tabele ekwipunku |

+--------------------+

**8. Diagram komponentów**

Diagram komponentów przedstawia szczegóły dotyczące odpowiedzialności i technologii/implementacji wybranego kontenera. W tym przypadku skupimy się na aplikacji PHP, która jest rdzeniem gry Cryland.

**Diagram Komponentów Aplikacji PHP:**

+----------------------------+

| Aplikacja PHP |

| |

| +------------------------+ |

| | Moduł Użytkownika | |

| | | |

| | - Rejestracja | |

| | - Logowanie | |

| | - Zarządzanie profilem | |

| +------------------------+ |

| |

| +------------------------+ |

| | Moduł Gry | |

| | | |

| | - Wyświetlanie zadań | |

| | - Wykonywanie zadań | |

| | - Zarządzanie ekwipunkiem| |

| +------------------------+ |

| |

| +------------------------+ |

| | Moduł Rankingu | |

| | | |

| | - Wyświetlanie rankingu| |

| +------------------------+ |

| |

| +------------------------+ |

| | Moduł Administracji | |

| | | |

| | - Zarządzanie użytkownikami |

| +------------------------+ |

+----------------------------+

**10. Implementacja funkcjonalności**

Opis kluczowych funkcjonalności zaimplementowanych w projekcie:

1. **Rejestracja użytkownika**:
   * Plik: register.php
   * Opis: Formularz rejestracyjny umożliwiający nowym użytkownikom założenie konta. Zapisuje dane użytkownika w bazie danych MySQL.
2. **Logowanie użytkownika**:
   * Plik: login.php
   * Opis: Formularz logowania umożliwiający zarejestrowanym użytkownikom dostęp do systemu. Sprawdza dane logowania z bazą danych.
3. **System walki**:
   * Plik: game/battle\_controller.php
   * Opis: Plik z napisaną logiką na podstawie której odbywa się walka w grze
4. **Walki z bossami**:
   * Plik: game/boss/boss\_controller.php
   * Opis: Specjalnie oskryptowane pojedynki z trudniejszymi przeciwnikami.
5. **Zarządzanie postacią**:
   * Plik: game/character\_controller.php
   * Opis: Strona umożliwiająca graczom przeglądanie i zarządzanie swoim ekwipunkiem. Dane są pobierane i aktualizowane w bazie danych.
6. **Eksploracja świata**:
   * Plik: game/world/exploration.php
   * Opis: System sterujący m.in. jakich przeciwników gracz może napotkać.

**11. Scenariusze testowe**

Przykładowy scenariusz testowy dla rejestracji użytkownika:

* **Warunki wstępne**: Brak, test wykonuje nowy użytkownik.
* **Kroki**:
  1. Przejdź do strony rejestracji.
  2. Wypełnij formularz rejestracyjny danymi użytkownika.
  3. Kliknij przycisk rejestracji.
* **Oczekiwany wynik**: Nowe konto użytkownika zostaje utworzone i użytkownik zostaje zalogowany do systemu.

Inny scenariusz testowy dla logowania:

* **Warunki wstępne**: Użytkownik musi mieć założone konto.
* **Kroki**:
  1. Przejdź do strony logowania.
  2. Wprowadź poprawne dane logowania.
  3. Kliknij przycisk logowania.
* **Oczekiwany wynik**: Użytkownik zostaje zalogowany do systemu i przekierowany na stronę główną.

**12. Wersjonowanie i zarządzanie kodem**

Projekt korzysta z systemu kontroli wersji Git. Repozytorium znajduje się na serwerze zdalnym, a praca nad projektem odbywa się z wykorzystaniem Git Flow, co zapewnia uporządkowany sposób zarządzania rozwojem i utrzymaniem aplikacji.