

***Tomasz Wojtkiewicz***

***222616***

PRACA DYPLOMOWA

magisterska

na kierunku Informatyka

**Generowanie testów jednostkowych, funkcjonalnych  
i integracyjnych przy użyciu dużego modelu językowego.**

*Katedra Mikroelektroniki i Technik Informatycznych Politechniki Łódzkiej*

**Promotor: dr Inż. Jacek Nazdrowicz**

ŁÓDŹ 2023

# Streszczenie

# Abstract

**Spis treści**

[1 Wstęp 5](#_Toc127743265)

[1.1 Cel i zakres pracy 5](#_Toc127743266)

[1.2 Strategiczne gry turowe 5](#_Toc127743267)

[1.3 Strategiczne gry komputerowe 5](#_Toc127743268)

[1.4 Gry strategiczne kiedyś i dziś 6](#_Toc127743269)

[1.5 Przegląd istniejących rozwiązań 7](#_Toc127743270)

[1.5.1 Civilization VI 7](#_Toc127743271)

[1.5.2 Hearts of Iron IV 8](#_Toc127743272)

[1.5.3 XCOM 2 9](#_Toc127743273)

[2 Projekt 10](#_Toc127743274)

[2.1 Opis produktu – scenariusz gry 10](#_Toc127743275)

[2.1.1 Rozgrywka 10](#_Toc127743276)

[2.1.2 Tereny 10](#_Toc127743277)

[2.1.3 Budynki 11](#_Toc127743278)

[2.1.4 Jednostki 12](#_Toc127743279)

[2.1.5 Zwycięstwo 12](#_Toc127743280)

[2.2 Wymagania 13](#_Toc127743281)

[2.2.1 Rozgrywka 13](#_Toc127743282)

[2.2.2 Interfejs użytkownika 13](#_Toc127743283)

[2.2.3 Aspekty techniczne 13](#_Toc127743284)

[2.3 Diagramy stanów 14](#_Toc127743285)

[3 Implementacja interfejsu użytkownika 16](#_Toc127743286)

[3.1 Ekran rejestracji 16](#_Toc127743287)

[3.2 Menu główne 17](#_Toc127743288)

[3.3 Ekran tworzenia gry 18](#_Toc127743289)

[3.4 Ekran rozgrywki 19](#_Toc127743290)

[3.4.1 Pasek przychodów 19](#_Toc127743291)

[3.4.2 Inspektor pola 20](#_Toc127743292)

[3.4.4 Tryb ataku i przegrupowania 21](#_Toc127743293)

[3.4.5 Powiadomienia 22](#_Toc127743294)

[4 Implementacja 23](#_Toc127743295)

[4.1 Wykorzystane technologie 23](#_Toc127743296)

[4.1.1 Unity 23](#_Toc127743297)

[4.1.2 Język C# 23](#_Toc127743298)

[4.1.3 LAMP 23](#_Toc127743299)

[4.1.4 Git 24](#_Toc127743300)

[4.1.5 Photon Unity Networking 2 24](#_Toc127743301)

[4.2 Animacja w tle menu 25](#_Toc127743302)

[4.3 System powiadomień 26](#_Toc127743303)

[4.4 Opakowanie obiektu przycisku 27](#_Toc127743304)

[4.5 Zachowanie zmiennych między scenami 28](#_Toc127743305)

[4.6 Wyświetlanie jednolitych alertów 28](#_Toc127743306)

[4.7 Odświeżanie wartości paska zasobów 29](#_Toc127743307)

[5 Dalszy rozwój projektu 30](#_Toc127743308)

[5.1 Wersja na platformy mobilne 30](#_Toc127743309)

[5.2 Rozszerzenie zawartości 30](#_Toc127743310)

[5.3 Reklamy 31](#_Toc127743311)

[5.4 Ulepszenie warstwy graficznej 31](#_Toc127743312)

[6 Podsumowanie 32](#_Toc127743313)

[Bibliografia 33](#_Toc127743314)

[Spis ilustracji 34](#_Toc127743315)

# 1 Wstęp

## 1.1 Cel i zakres pracy

Celem pracy jest sprawdzenie skuteczności i jakości testów generowanych przez popularny, prężnie rozwijający się w ostatnim czasie duży model językowy ChatGPT. Zakres pracy obejmuje UZUPEŁNIJ.

## 1.2 Testowanie oprogramowania

Wytworzenie współczesnej aplikacji potrafi wygenerować spore koszty. Oprócz samych programistów należy też często opłacić koszty różnorakich licencji, designu, serwerów, utrzymania aplikacji w dobrej kondycji i inne łatwe do przeoczenia opłaty. Należy też zwrócić uwagę na rodzaj danych, którymi aplikacja ma za zadanie zarządzać. Część z nich może okazać się poufna, a ich wyciek skutkowałby utratą reputacji dla firmy, a co gorsza potencjalnymi sprawami sądowymi i karami finansowymi.

Wymienione wyżej powody to tylko część przykładów odpowiadających na pytanie: „dlaczego należy testować aplikacje?”.

## 1.3 Strategiczne gry komputerowe

Gry strategiczne to gatunek gier komputerowych, w których zwycięstwo uzależnione jest od strategii i taktyki, jaką obierze gracz. Głównym zadaniem użytkownika w tego typu grach jest odpowiednie dowodzenie podległymi zasobami (oddziałami wojskowymi, pracownikami cywilnymi etc.) w celu pokonania przeciwnika lub osiągnięcia innego wyznaczonego celu [1]. To właśnie rozbudowana logika zarządzania jest tym, co wyróżnia grę strategiczną od innych gatunków posiadających tylko jej podstawowe aspekty. Z tego powodu gry strategiczne doskonale idą w parze   
z takimi gatunkami, jak gry ekonomiczne (gdzie gracz wciela się w rolę np. zarządcy parku rozrywki) oraz gry wojenne (w których zadaniem gracza będzie wygrana w bitwie, wykonanie określonej misji lub globalna dominacja).

## 1.4 Gry strategiczne kiedyś i dziś

Gry strategiczne trafiają na dobry odbiór u osób, którym przyjemność z rozgrywki sprawia przewidywanie ruchów przeciwnika i szukania najlepszych możliwych odpowiedzi. Takie nie podyktowane losowością zwycięstwo daje wtedy dużo satysfakcji, ponieważ zapewnia poczucie przewyższenia przeciwnika mądrością i nabytym doświadczeniem. Potrzeba takiego poczucia wyższości, ale też dobrej, umysłowej rozgrywki, towarzyszy nam już od wielu lat. Oprócz popularnych szachów za przykład podać można też starochińską grę Go (rys. 1), której pierwsze udokumentowane zapiski sięgają 1122–221 p.n.e. [2]. Sam program komputerowy do gry w Go zdołał pokonać mistrza Europy Fan Hui dopiero w roku 2015. Wydarzenie to nie oznaczało jednak całkowitej wyższości maszyny nad człowiekiem w tej dziedzinie, ponieważ program   
w dalszym ciągu nie był w stanie za każdym razem doprowadzić do zwycięstwa   
i przegrywał rozgrywki z innymi czołowymi graczami. Fakt ten pokazuje, że skomplikowane gry strategiczne wymagające od gracza zapamiętania wielu zasad oraz zależności nie są wcale wymysłem współczesnym, a jedynie kolejnym krokiem   
w ewolucji idei powstałej kilka tysięcy lat temu.



Rys. 1 Model planszy do gry w Go z grobowca dynastii Sui, 581–618 n.e.

.

## 1.5 Przegląd istniejących rozwiązań

### 1.5.1 Civilization VI

Gra programisty Sida Meiera, twórcy wielu uznanych tytułów strategicznych   
a w konsekwencji dyrektora kreatywnego w firmie Firaxis Games. Polega na rozwijaniu wybranej cywilizacji poprzez zakładanie miast, rekrutowanie jednostek, budowę obiektów oraz prowadzenie działań dyplomatycznych, w tym wojny, z innymi takimi cywilizacjami. Do 18 lutego 2018 sprzedano łącznie ponad 33 miliony egzemplarzy gier serii [3].



Rys. 2 Zrzut ekranu z gry Civilization VI

Zalety

* wspaniała warstwa audiowizualna (rys. 2),
* stosunkowo niski próg wejścia,
* mnogość wyboru cywilizacji,
* wiele możliwości zwycięstwa,
* rozbudowana dodatkowa płatna zawartość,
* inspiracja cudami świata i wydarzeniami, które miały miejsce w rzeczywistości.

Wady

* spore wymagania sprzętowe,
* duża ilość miast w końcowej fazie gry znacząco utrudnia optymalne zarządzanie nimi wszystkimi.

### 1.5.2 Hearts of Iron IV

Gra z gatunku strategii czasu rzeczywistego osadzona w realiach II wojny światowej. Stworzona i wydana przez studio Paradox Interactive. Jej premiera miała miejsce 6 czerwca 2016 roku [4]. Cechą charakterystyczną tej serii jest wierne odzwierciedlenie realiów historycznych. Gracz ma za zadanie wcielić się w lidera jednego z państw w okresie na krótko przed rozpoczęciem drugiej wojny światowej   
i poprowadzenie go ku zwycięstwu przez całkowitą dominację wszystkich innych nacji.



Rys. 3 Zrzut ekranu z gry Hearts of Iron IV

Zalety

* dobra warstwa audiowizualna (rys. 3),
* aspekt historyczny,
* stosunkowo niewielkie wymagania sprzętowe,
* tryb rozgrywki wieloosobowej,
* rozbudowane drzewka polityczne i technologiczne.

Wady

* bardzo wysoki próg wejścia,
* tylko jedna możliwość wygrania gry.

### 1.5.3 XCOM 2

Strategiczna gra turowa stworzona przez Firaxis Games, polegająca na wykonywaniu określonych misji grupką kilku oddzielnie kierowanych żołnierzy przyszłości. W świecie przedstawionym gry tajna organizacja wojskowa XCOM próbuje walczyć z inwazją obcych, przegrała wojnę, a teraz stawia opór okupacji Ziemi [5].   
W ciągu tygodnia od premiery na platformie Steam sprzedano ponad pięćset tysięcy cyfrowych kopii gry [6].



Rys. 4 Zrzut ekranu z gry XCOM 2

Zalety

* wspaniała warstwa audiowizualna (rys. 4),
* proceduralnie generowane poziomy,
* zarządzanie bazą, w tym jej rozbudowa,
* niekończący się wyścig z czasem- odwlekanie wykonywania misji prowadzi do ostatecznej porażki ludzkości,
* rozwój technologii i umiejętności przeciwników wraz z postępem kampanii.

Wady

* dość wysoki poziom trudności dla nowych graczy,
* po nauczeniu się pewnych schematów i mechanik porażka jest praktycznie niemożliwa,
* irytujący element losowości,
* duże wymagania sprzętowe.

# 2 Projekt

## 2.1 Opis produktu – scenariusz gry

Aplikacja zaprojektowana w ramach części technicznej jest strategiczną grą turową, przeznaczoną do użytku na komputerach osobistych z systemem operacyjnym Windows lub MacOS. Gra odbywa się przez Internet, przez co wymagane jest też połączenie sieciowe. Do realizacji użyto silnika Unity, wykorzystując tryb tworzenia gier 2D. Tytuł roboczy produktu to „Legendary Alliance”.

### 2.1.1 Rozgrywka

Rozgrywka toczy się na dwuwymiarowej mapie świata. Zadaniem gracza jest przejęcie wszystkich terenów innych graczy, zebranie dziesięciu tysięcy sztuk waluty lub pięciu tysięcy punktów nauki.

### 2.1.2 Tereny

Mapa gry podzielona jest na tereny, które swoje nazewnictwo zawdzięczają grze „Ryzyko”. Z uwagi na fakt, że gra ta była jedną z inspiracji do stworzenia części technicznej tej pracy, zostały one nazwane dokładnie tak, jak w tej znanej grze planszowej (rys. 5).

Obraz zawierający mapa

Opis wygenerowany automatycznie

Rys. 5 Projekt planszy do gry w "Ryzyko"

Wszystkie tereny są „puste” po rozpoczęciu gry. Gracze poprzez wskazanie  
w pierwszej turze wybranego miejsca na rozpoczęcie rozgrywki zajmują dany teren   
i budują tam swoją stolicę.

### 2.1.3 Budynki

Posiadane pola można ulepszać, budując trzy typy budynków: koszary, farmę oraz laboratorium (rys. 6). Każdy z nich, który jest wybudowany, zapewnia graczowi pewne korzyści.

Rys. 6 Projekt budynków. Od lewej: koszary, laboratorium i farma.

Laboratorium daje graczowi przychód w postaci jednego punktu nauki na turę. Maksymalnie na jednym terenie można zbudować pięć tego rodzaju budynków.   
Koszt wybudowania laboratorium to:

Natomiast sprzedaż zwróci graczowi ilość waluty wg wzoru:

Dla obu powyższych wzorów oznaczenia są takie same, czyli:

*b* - bazowy koszt laboratorium,

mk - modyfikator kary za ilość posiadanych laboratoriów na danym terenie

S - ilość posiadanych na danym terenie laboratoriów przed kupnem bądź sprzedażą

Farma pozwala graczowi na posiadanie większej ilości jednostek w kontekście całej mapy. Koszary pozwalają graczowi na rekrutowanie jednostek na danym terenie. Możliwe jest posiadanie tylko jednej farmy i koszar na danym terenie. W każdej podbitej stolicy gracz automatycznie posiada wybudowany jeden budynek farmy oraz koszar.   
Na tych specjalnych polach można budować i sprzedawać tylko laboratoria. Na innych posiadanych terenach koszary i farma zawsze sprzedawane są po stałych cenach ze względu na ich ograniczenie do jednego budynku danego typu na teren.

### 2.1.4 Jednostki

Po wybudowaniu koszar (lub na polu ze stolicą) gracz zyskuje możliwość rekrutowania jednostek (rys. 7). Koszt rekrutacji jest stały. Sumaryczną ilość jednostek dostępnych do posiadania na całej mapie dla danego gracza określa wzór:

gdzie:

sf – suma posiadanych przez gracza farm

bf – bonus do jednostek za każdą posiadaną farmę

ss – suma posiadanych przez gracza stolic

bs – bonus do jednostek za każdą posiadaną stolicę

Obraz zawierający tekst, zabawka

Opis wygenerowany automatycznie

Rys. 7 Jednostka gracza czerwonego

Jednostki służą graczom do przejmowania innych terenów. Aby tego dokonać, gracz podczas atakowania przeciwnika musi wystawić do walki więcej jednostek niż tych obecnych na atakowanym polu. Nie jest to takie proste, ponieważ liczba stacjonujących jednostek jest podana w przybliżeniu. Jednostki służą też do kontrolowania pól. Teren bez jednostki jest bowiem traktowany jako „porzucony” przez gracza, zatem nie można otrzymać już z niego żadnych przychodów aż do momentu jego ponownego przejęcia.

### 2.1.5 Zwycięstwo

Aby zwyciężyć należy spełnić jeden z trzech warunków:

1. wszystkie pola na mapie muszą należeć do jednego gracza (z wyjątkiem pól niepodbitych),
2. jeden z graczy musi zgromadzić ponad 5000 punktów nauki,
3. jeden z graczy musi zgromadzić ponad 10000 sztuk złota (waluty).

W przypadku remisu punktowego obowiązuje zasada „kto pierwszy, ten lepszy”. Wygrana jest bowiem sprawdzana na początku tury każdego z graczy. W szczególnym przypadku, kiedy gracze będą posiadali po jednym sąsiadującym terenie zaś wynik walki pozbawi ich obu wszystkich jednostek, gra poinformuje graczy o zaistniałym remisie. Jest to jedyny taki przypadek, w którym żaden z graczy nie osiąga wygranej na końcu gry.

## 2.2 Wymagania

### 2.2.1 Rozgrywka

* gra posiada swoje sprecyzowane warunki zwycięstwa: możliwość ukończenia gry,
* możliwość rekrutowania jednostek,
* ulepszenia pól w postaci budynków,
* możliwość przedwczesnego opuszczenia rozgrywki,
* możliwość wygrania gry na więcej niż jeden sposób.

### 2.2.2 Interfejs użytkownika

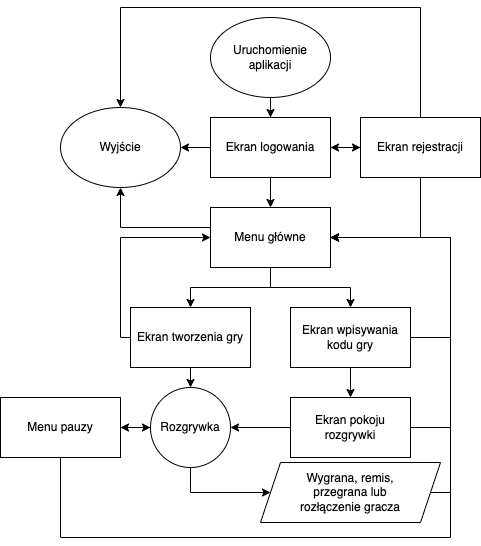
* niestandardowe tekstury przycisków i pól tekstowych,
* animacja tła,
* ekran pokoju z losowo wygenerowanym identyfikatorem i możliwością zmiany statusu gotowości graczy do rozpoczęcia rozgrywki,
* ekran logowania i rejestracji z obsługą przycisku „enter” i tabulatora.

### 2.2.3 Aspekty techniczne

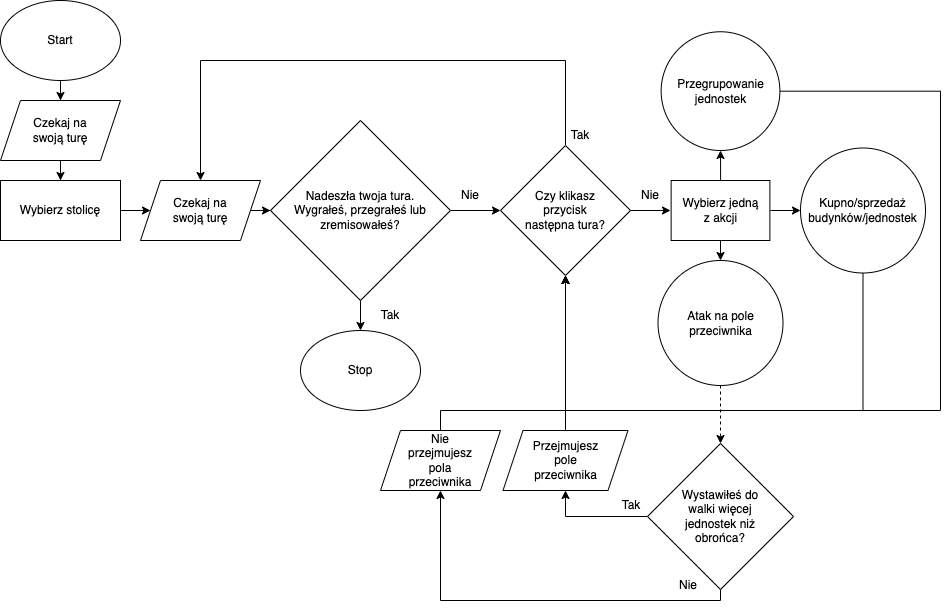
* dostęp do gry tylko poprzez zalogowanie się do systemu,
* możliwość rejestracji nowego konta,
* gra z innymi użytkownikami przez sieć internetową,
* możliwość uruchomienia i rozgrywki między systemami Windows i MacOS.

## 2.3 Diagramy stanów

Poniższe rysunki przedstawiają diagramy stanów. Pierwszy z nich opisuje interakcje interfejsu użytkownika w menu (rys. 8), drugi zaś przedstawia przebieg stanu „Rozgrywka” z pierwszego diagramu (rys. 9).



Rys. 8 Diagram stanów- interakcje interfejsu użytkownika



Rys. 9 Diagram stanów- stan "Rozgrywka"

# 3 Implementacja interfejsu użytkownika

W silniku Unity zastosowano podział gry na „sceny”. Pozwala to na umieszczenie różniących się układów interfejsu użytkownika w osobnych miejscach tak, aby ułatwić organizację pracy. Te sceny zostaną w tym rozdziale omówione. W scenie rozgrywki, ze względu na jej skomplikowanie, każdy ważniejszy element interfejsu zostanie omówiony osobno. Niektóre sceny zostaną pominięte ze względu na zastosowanie w nich podobnych rozwiązań i niewielkie różnice wizualne w porównaniu do omówionych już scen. Wszystkim ekranom interfejsu użytkownika towarzyszy prosta animacja, której szczegóły omówione zostały w rozdziale czwartym pracy. Jest ona pustą mapą świata, nad którą przemieszcza się kamera, i której ruch został zainspirowany klasycznym ruchem logo DVD na odtwarzaczach.

## 3.1 Ekran rejestracji

Ekran rejestracji jest drugim ekranem, obok ekranu logowania, który użytkownik może zobaczyć bez poprawnego zalogowania się do systemu. Do omówienia wybrano właśnie ekran rejestracji (rys. 10) ze względu na jego większy poziom skomplikowania. Ekran ten zawiera formularz pozwalający użytkownikowi na rejestrację w bazie danych nowego konta.



Rys. 10 Ekran rejestracji

Jest on zabezpieczony przed podaniem niepoprawnych danych, za które uznaje się:

* za długie lub za krótkie dane,
* nazwa użytkownika, jeśli pojawiają się znaki specjalne,
* nazwa użytkownika, jeśli podana nazwa jest już zajęta,
* hasło, jeśli nie zawiera co najmniej jednego znaku specjalnego, wielkiej litery, małej litery i cyfry,
* hasło, jeśli zawiera białe znaki,
* powtórzone hasło, jeśli nie jest identyczne z podanym wcześniej hasłem.

Ekran rejestracji zawiera przycisk „SIGN UP”, który po kliknięciu sprawdza poprawność danych według powyższej listy. Następnie, jeżeli okażą się one poprawne, wysyła zapytanie do serwera, który dodaje użytkownika do bazy danych. Jeżeli przy tej operacji serwer zwróci kod sukcesu, to użytkownik zostanie przekierowany do ekranu głównego. W przeciwnym wypadku ukaże mu się pole tekstowe z wiadomością błędu przesłaną przez serwer. Aby upodobnić proces wypełniania formularza do tego znanego ze stron internetowych, opracowano proste rozwiązanie dające możliwość podjęcia próby rejestracji przyciskiem „enter” oraz przeskakiwania między polami formularza przyciskiem tab.

## 3.2 Menu główne

Ekran menu głównego (rys. 11) daje użytkownikowi możliwość wyboru - gracz może stworzyć lub dołączyć do już istniejącego pokoju gry. To również na ten ekran będzie wracał po każdej rozgrywce. Na tym ekranie gracz ma też możliwość wylogowania się lub zakończenia działania aplikacji.



Rys. 11 Ekran menu głównego

## 3.3 Ekran tworzenia gry

Ekran tworzenia gry (rys. 12) jest dostępny dla gracza, który zdecydował się utworzyć rozgrywkę i zostać „hostem”. Jego zadaniem będzie udostępnienie pozostałym graczom kodu pokoju (*room id*) aby mogli do niego dołączyć. Przycisk „ready” pozwala na zmianę statusu gracza. Rozgrywki nie będzie można uruchomić aż do momentu, kiedy wszyscy gracze ustawią swój status na gotowy do gry.

Obraz zawierający mapa

Opis wygenerowany automatycznie

Rys. 12 Ekran tworzenia rozgrywki

## 3.4 Ekran rozgrywki

Ekran rozgrywki (rys. 13) to najbardziej skomplikowany ekran wśród wszystkich obecnych w programie. To na nim gracze, po wcześniejszym dołączeniu do wspólnego pokoju, będą starać się o zwycięstwo. Ze względu na ilość ważnych elementów, każdy   
z nich został omówiony w osobnym dla siebie rozdziale.

Obraz zawierający tekst, osoba

Opis wygenerowany automatycznie

Rys. 13 Ekran rozgrywki w testowym trybie dla jednego gracza

Gracz już na pierwszy rzut oka uzyskuje dwie bardzo ważne informacje strategiczne   
o dostępnych terenach - ich sąsiedztwo oraz połączenia morskie. Po wybraniu stolicy, kolejną taką informacją staje się zmieniający się kolor pola w zależności od koloru aktualnie władającego nim gracza. Warto też wspomnieć o lewej stronie ekranu, na której wyświetla się kolejka ruchów graczy. Gracz, który obecnie wykonuje swój ruch, ma   
w niej swoją nazwę użytkownika przesuniętą bardziej w prawą stronę w stosunku do innych nazw użytkowników. Dzięki temu zabiegowi gracze dokładnie wiedzą, kto obecnie wykonuje swoje akcje.

### 3.4.1 Pasek przychodów

Na górze ekranu rozgrywki (rys. 13) znajduje się stale widoczny pasek przychodów. Zawiera on cenne dla gracza informacje, bo oprócz jego nazwy użytkownika są to odpowiednio:

* na żółtym tle- ilość posiadanej waluty i jej przychód,
* na niebieskim tle- ilość posiadanych punktów nauki i ich przychód,
* na zielonym tle - ilość posiadanych farm, a po znaku równości ilość posiadanych jednostek oraz górna granica ich liczności.

Pasek przychodów automatycznie odświeża swoje wartości po każdej zmieniającej je akcji gracza, na przykład po wybudowaniu farmy.

### 3.4.2 Inspektor pola

Inspektor pola (rys. 14) pojawia się, kiedy w swojej turze gracz naciśnie myszą na pole. Jego przynależność zdecyduje o tym, jaki wariant inspektora pola wyświetli się użytkownikowi. Jeżeli pole należy do gracza to inspektor pola pokaże mu, ile posiada na nim budynków danej kategorii i jednostek. Umożliwi też kupno nowych obiektów   
i przeprowadzenia rekrutacji na danym terenie. W przypadku, gdy teren należy do przeciwnika, inspektor pola również pokaże informacje o ulepszeniach pola. Tym razem będą to jednak ich oszacowane wartości z pewnego ustalonego odgórnie zakresu.   
Gracz otrzyma zatem przykładową informację: „na polu znajduje się od pięciu do ośmiu żołnierzy”. Ma to na celu wprowadzenie elementu niepewności do rozgrywki. Jeżeli pole nie należy do żadnego z graczy, a są na nim zbudowane obiekty, to inspektor pola pokaże ich dokładną ilość. Wyświetli on też kolorystyczną informację o obecnym właścicielu danego terenu.

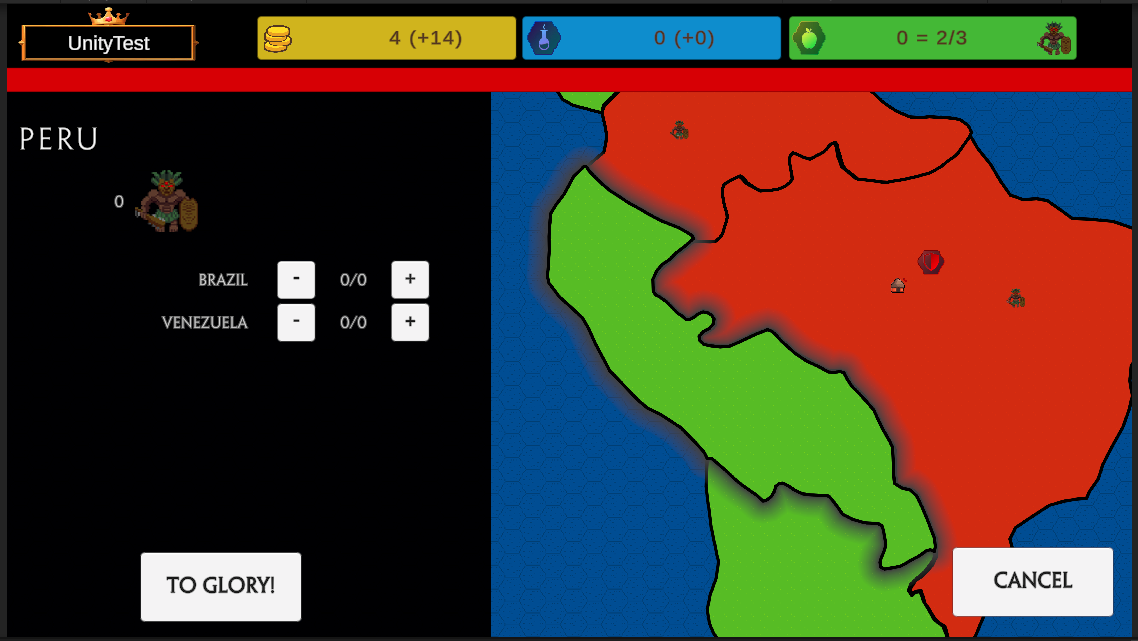
Obraz zawierający tekst, zewnętrzne, zrzut ekranu

Opis wygenerowany automatycznie

Rys. 14 Inspektor pola w wariancie dla posiadanego terenu

### 3.4.4 Tryb ataku i przegrupowania

Tryb ataku (rys. 15) staje się widoczny, gdy gracz w inspektorze pola sąsiadującego z jego terenem naciśnie odpowiednio przycisk „ATTACK”. Tryb ten pokazuje graczowi, iloma jednostkami sumarycznie atakuje pole przeciwnika, z jakiego sąsiadującego terenu pobiera swoje jednostki i w jakiej ilości. Jest bowiem możliwy atak z kilku terenów jednocześnie. Jeżeli gracz faktycznie zdecyduje się na atak, to po wciśnięciu przez niego przycisku „TO GLORY!” gra przeliczy odpowiednio, który   
z graczy wygrał bitwę i przypisze to pole odpowiedniemu z nich. Odświeży też informacje o jednostkach znajdujących się na atakowanym i atakujących terenach.



Rys. 15 Tryb ataku w sytuacji możliwego natarcia z wielu terenów jednocześnie

Tryb ataku zmienia się odpowiednio w tryb przegrupowania, jeżeli wybrane przez gracza pole już do niego należy. W tym trybie wygląd interfejsu jest niemalże identyczny, a gra zamiast przeliczać zwycięzcę odpowiednio zmienia ilość jednostek na wybranym terenie i terenach z nim sąsiadujących.

### 3.4.5 Powiadomienia

Ostatnim elementem interfejsu ekranu rozgrywki jest moduł powiadomień. Jest to prostokąt, który wyświetla zakolejkowane dla gracza powiadomienia. Wyświetla się   
w kolorze przypisanym graczowi (rys. 16) i posiada graficzną informację o ilości pozostałych w kolejce powiadomień, jeżeli faktycznie czeka w niej jeszcze jakieś powiadomienie. Powiadomienie wygasa automatycznie, ale kliknięcie w nie przyspiesza jego animację zanikania.

Obraz zawierający mapa

Opis wygenerowany automatycznie

Rys. 16 Powiadomienie wyświetlane graczowi czerwonemu

# 4 Implementacja

W tym rozdziale opisano wykorzystane technologie oraz wybrane problemy napotkane podczas implementacji projektu wraz z ich rozwiązaniem.

## 4.1 Wykorzystane technologie

### 4.1.1 Unity

Unity to silnik do tworzenia gier charakteryzujący się dużą popularnością ze względu na swój niski poziom skomplikowania, niski próg wejścia i dużą ilość materiałów szkoleniowych dostępnych w Internecie. To właśnie te argumenty zadecydowały o wyborze tej technologii do implementacji projektu. Kolejnym argumentem przemawiającym na korzyść Unity jest możliwość kompilacji projektu na wiele platform, takich jak Windows, MacOS, Linux, Android, iOS, WebGL a nawet tvOS. Jako potencjalny kandydat na silnik gry był rozpatrywany też silnik Unreal Engine, ale ze względu na konieczność używania w nim języka C++ ostatecznie użyto właśnie Unity posługującego się łatwiejszym w nauce językiem C#.

### 4.1.2 Język C#

Język C# to język programowania zaprojektowany dla firmy Microsoft. Jego nazwa pochodzi od symbolu krzyżyka ♯. W muzyce dźwięk C♯ jest wyższy niż dźwięk C. Może to sugerować, że język jest rozwinięciem C/C++ [7]. Pomimo tej zależności sam język ma dużo wspólnego z językiem programowania Java. Jego cechy takie, jak wspomaganie programowania obiektowego, wysoki poziom abstrakcji oraz automatyczne usuwanie śmieci z pamięci jeszcze bardziej przemawiają na jego korzyść.

### 4.1.3 LAMP

Zestaw oprogramowania klasycznie używany jako platforma serwerowa dynamicznych stron WWW. Jego nazwa to akronim, który swoje pochodzenie zawdzięcza nazwie użytych narzędzi:

* L - system operacyjny Linux;
* A - serwer Apache;
* M - baza danych w systemie MySQL;
* P - język skryptowy PHP.

W projekcie do postawienia serwera i bazy danych na komputerze z systemem Linux wykorzystano usługę EC2[[1]](#footnote-2) dostępną dzięki Amazon Web Services. W omawianym projekcie ten zestaw oprogramowania został wykorzystany do zdalnego wydawania poleceń bazie danych przez użytkownika i przesłanie mu informacji zwrotnej. Zastosowanie usługi EC2 zapewnia stałe działanie i dostępność serwera,  
w przeciwieństwie do tymczasowego serwera na maszynie lokalnej, co pozwoliło na testowanie aplikacji na urządzeniach z całego świata już we wczesnej fazie rozwoju gry bez konieczności własnoręcznego przekierowywania portów oraz posiadania ciągle uruchomionej maszyny.

### 4.1.4 Git

Git to rozproszony system kontroli wersji. Cechuje go doskonałe wsparcie dla rozgałęzionego procesu programowania, praca bez podłączenia do sieci oraz efektywna praca z projektami dużych rozmiarów. W tym jednoosobowym projekcie sprawdził się doskonale jako narzędzie pozwalające na szybki powrót do ostatniej działającej wersji. Dzięki temu o wiele łatwiej było zlokalizować nowe błędy, ponieważ można było sprawdzić ich istnienie w poprzednich wersjach aplikacji, po czym szybko przenieść się z powrotem do najnowszej wersji. Pozwalał też na dokładne sprawdzenie zmian naniesionych w danym pliku, co również przyczyniło się do sprawniejszego wykrywania błędów.

### 4.1.5 Photon Unity Networking 2

Photon Unity Networking 2 jest wtyczką do silnika Unity ułatwiająca tworzenie gier wymagających wymiany informacji między graczami w czasie rzeczywistym przez sieć [8]. Daje ona dostęp do darmowych dedykowanych serwerów o bardzo dużej wydajności. Serwery te zawierają pokoje, do których mogą podłączać się gracze, aby przesyłać między sobą informacje o aktualnym stanie rozgrywki. Sama wtyczka daje wiele możliwości takiej synchronizacji, przy czym w projekcie wykorzystano system rozsyłania „zdarzeń” wśród graczy w obrębie wspólnego pokoju.

## 4.2 Animacja w tle menu

Poruszając się po interfejsie użytkownika nie jest trudno zauważyć ruchome tło. Zasadza działania tej animacji jest dość prosta - wystarczy sprawdzać, czy kamera dotarła już w odpowiednie miejsce na scenie. Jeśli dotyka jednej z granic, należy przemnożyć jej prędkość przez -1. Dodatkowo, ponieważ silnik Unity po każdej zmianie sceny usuwa wszystkie obiekty z poprzedniej, należy wywołać przy tworzeniu takiego obiektu metodę *DontDestroyOnLoad()*. To rozwiązanie faktycznie zachowuje obiekt animacji   
z poprzedniej sceny, a także zachowuje jej płynność - animacja nie zaczyna się od nowa przy każdej zmianie sceny.

W trakcie dalszej implementacji okazało się, że użytkownik nie zawsze swoją nawigację po interfejsie użytkownika będzie zaczynał od ekranu logowania. Po skończonej rozgrywce zostanie on bowiem przeniesiony na ekran menu głównego. Należało więc zainicjalizować obiekt animacji również i na tym ekranie. Tutaj pojawił się problem. Jeżeli użytkownik, w przerwie pomiędzy rozgrywkami, odwiedzi ekran logowania i ekran menu głównego, to animacja zostanie utworzona dwa razy. Efekt wizualny tego błędu nie był widoczny, dało się jednak usłyszeć dwie instancje odtwarzacza audio, które grają ten sam utwór opóźniony w stosunku do siebie.

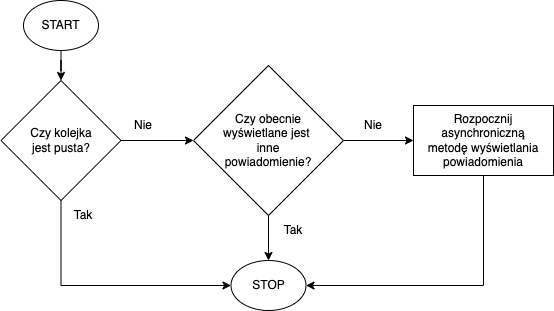
Rozwiązaniem problemu okazało się zaimplementowanie obiektu animacji zgodnie ze wzorcem projektowym Singleton. Wzorzec ten, mimo nazwania go przez niektórych programistów antywzorcem, okazał się idealnym kandydatem na rozwiązanie tego problemu. Zapewnia on bowiem globalną referencję do danego obiektu jednocześnie dbając o to, aby istniała maksymalnie jedna jego instancja w każdym momencie działania aplikacji [9]. Dzięki temu każda próba utworzenia drugiego obiektu animacji skutkowała zakończeniem tej metody, pozostawiając tylko oryginalny obiekt.

## 4.3 System powiadomień

Gracz w przerwie pomiędzy swoimi turami nie widzi dokładnie jakie akcje podejmuje obecnie grający użytkownik. Widzi on jedynie teren, który ten obecnie wybrał, oraz wizualne odzwierciedlenie skutków jego działań takie, jak zmiana koloru pola po jego przejęciu. Aby poinformować wszystkich nie wykonujących obecnie ruchu graczy niezbędne okazało się zaimplementowanie systemu powiadomień. Taki system rozsyłałby do odpowiednich odbiorców wiadomość, która opisuje, co właściwie dzieje się na ekranie podczas ruchu przeciwnika, nie zdradzając jednocześnie wszystkich szczegółów. Użytkownicy wiedzieliby więc, że gracz wybudował laboratorium, ale wciąż nie mieliby pewności, ile dokładnie takich budynków posiada przeciwnik.

Już pierwsze testy wykazały, że powiadomienia często przychodzą w bardzo krótkim odstępie czasowym, przez co możliwe było przeczytanie tylko ostatniego z nich. Zasłaniało ono bowiem wszystkie inne powiadomienia. Potrzebne było znalezienie sposobu na gromadzenie powiadomień i wyświetlanie ich jedno po drugim nawet mimo ich dużej ilości, tak aby gracz był w stanie przeczytać i zareagować odpowiednio na każde z nich.

W tej sytuacji pomocną strukturą danych okazała się kolejka. Wszystkie przychodzące do gracza powiadomienia w pierwszej kolejności trafiają do obiektu kolejki i tam czekają na swoje wywołanie w postaci obiektu łańcucha znaków. Osobna metoda uruchamiana raz na każdą klatkę gry działa według poniższego diagramu (rys. 17).



Rys. 17 Schemat metody decydującej o wyświetleniu kolejnego powiadomienia

## 4.4 Opakowanie obiektu przycisku

Ustawianie parametrów przycisku w kodzie gry okazało się problematyczne. Jest to spowodowane podziałem przez silnik Unity przycisku na trzy klasy: właściwy przycisk, napis na nim oraz jego bitmapa. Problem ten stawał się bardzo irytujący- dla zaledwie trzech przycisków na ekranie trzeba było zadeklarować 9 pól klasy, a następnie na ekranie edytora metodą „przeciągnij i upuść” ustawić pola tej klasy na odpowiednie referencje do interfejsu użytkownika. Przykładowa taka lista referencji w edytorze wygląda jak na rys. 18. Dodanie do takiej przykładowej listy choćby jednego przycisku, a co za tym idzie dopisanie do niej trzech nowych pól, powodowało więc zwiększenie szumu informacyjnego, a czasami nawet było skutkiem błędów.

Obraz zawierający tekst, tabliczka, zrzut ekranu

Opis wygenerowany automatycznie

Rys. 18 Przykładowa lista referencji

Jako rozwiązanie zaimplementowano klasę *ButtonWrapper* przechowującą referencję do wszystkich niezbędnych do operowania przyciskiem. Konstruktor tej klasy przyjmuje tylko jeden parametr i jest nim obiekt przycisku. Na jego podstawie klasa wyłuskuje pozostałe niezbędne obiekty i przechowuje je w polach klasy. Zastosowanie tego rozwiązania pozwoliło na zmniejszenie wykorzystywanej ilości pól klasy w innych skryptach oraz skrócenie metod *Start()* tych skryptów, w których do tej pory odbywał się wyżej wymieniony proces wyłuskiwania elementów. Wszystkie te działania przyczyniły się do zwiększenia przejrzystości kodu i ekranu edytora.

## 4.5 Zachowanie zmiennych między scenami

Jak już wcześniej omówiono w rozdziale „4.2 Animacja w tle menu”, Unity nie zachowuje obiektów między scenami. Z tego powodu zmienne zapisane na jednym ekranie przepadną przy przejściu do następnej sceny. Sytuacja taka występowała, kiedy gracz na ekranie tworzenia gry klikał przycisk rozpoczęcia gry- dane o graczach nie były już dostępne do odczytu przez program.

Przy próbie rozwiązania problemu rozważono zastosowanie metody *DontDestroyOnLoad()*, jednak oprócz zmiennych zachowałaby ona cały obiekt sceny. Podejście to byłoby więc problematyczne przy skalowaniu aplikacji - obiekt sceny mógłby okazać się zbyt duży. Potencjalnym rozwiązaniem byłoby utworzenie nowego obiektu, który nie byłby niszczony przy zmianie sceny, a jego zadaniem byłoby tylko przechowanie tych niezbędnych informacji.

W końcowej wersji produktu zdecydowano się na rozwiązanie, które nie tylko pozwoli na przechowywanie informacji między scenami, ale także umożliwi ich odzyskanie nawet po nieoczekiwanym zamknięciu aplikacji. Wykorzystano klasę *PlayerPrefs*, która udostępnia metody zapisywania par klucz-wartość do pliku z preferencjami gracza. Rozwiązanie to otwiera furtkę do zaimplementowania w przyszłości modułu powrotu do gry gracza, który był zmuszony zamknąć aplikację w trakcie rozgrywki - dane o kodzie pokoju i jego przeciwnikach będą zapisane w pliku zamiast niedostępnej po zamknięciu pamięci programu.

## 4.6 Wyświetlanie jednolitych alertów

Aby poinformować gracza o zaistniałym błędzie z powodu niedostępności serwera czy na przykład niepoprawnych danych wejściowych, potrzebne jest wyświetlenie komunikatu błędu. Unity nie posiada jednak wbudowanej klasy takich alertów; niezbędna okazała się więc jego własna implementacja, tak aby wyświetlane komunikaty zachowały jeden ustalony styl graficzny.

Pierwsza koncepcja polegała na utworzeniu osobnego, niewidocznego przy starcie obiektu *Canvas*, który posiadałby tytuł, komunikat oraz przycisk zamykający alert.   
W celu uniknięcia tworzenia takiego obiektu na każdym ekranie można byłoby zastosować wzorzec Singleton tak jak opisano w rozdziale „4.2 Animacja w tle menu”. Takie rozwiązanie znacznie utrudniłoby jednak tworzenie komunikatów   
z unikatowymi dla nich elementami, ponieważ dla każdego typu komunikatu należałoby stworzyć osobny obiekt. Z tego powodu zdecydowano się na zaprojektowanie klasy *MessageBoxFactory*, która tworzy nowy obiekt *Canvas* wraz z odpowiednią zawartością wyłącznie przy użyciu skryptu, a nie edytora. Pozwoliło to osiągnąć dwa ważne cele. Po pierwsze, wyświetlenie komunikatu błędu w innym skrypcie zajmuje jedną linijkę, co pozytywnie wpływa na przejrzystości kodu. Po drugie, wyświetlenie dodatkowych elementów w komunikacie jest kwestią dodania odpowiednich metod, a nie jak   
w przypadku pierwszej wersji rozwiązania tworzenia osobnych obiektów co niekorzystnie wpłynęłoby na czytelność edytora silnika gry.

## 4.7 Odświeżanie wartości paska zasobów

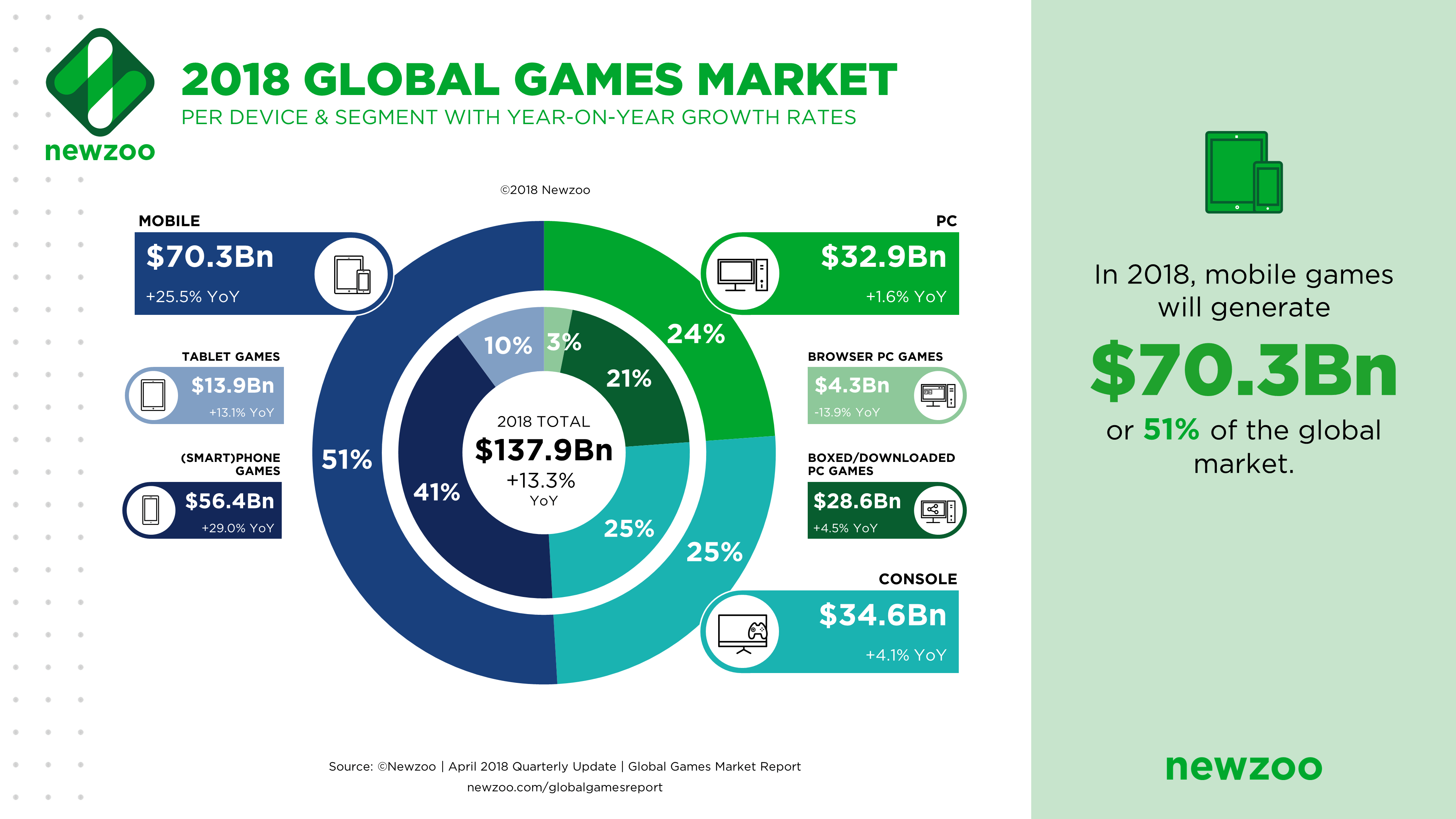
Pasek zasobów to niezwykle ważny element rozgrywki. Jego dane zawsze powinny być aktualne, dlatego kluczowym działaniem było dopilnowanie ich poprawnego odświeżania. Niestety, przy każdym nowo dodanym budynku lub jednostce pojawiały się nowe możliwości na pominięcie tej operacji, co wprowadzałoby gracza   
w błąd.

Aby pozbyć się tego problemu, pasek zasobów automatycznie pobiera obecne wartości co piętnaście klatek gry. Takie rozwiązanie sprawia, że logikę pobierania danych wystarczy wpisać tylko w jednym miejscu. Powoduje ono też odświeżenia paska nawet w turze gracza przeciwnego; gracz po utraceniu terenu z laboratoriami zobaczy w czasie rzeczywistym różnicę w przychodzi punktów nauki bez konieczności implementowania tej funkcji w jakimkolwiek innym miejscu w kodzie.

# 5 Dalszy rozwój projektu

## 5.1 Wersja na platformy mobilne

Silnik Unity pozwala na kompilację projektu dla platformy Android i iOS. Otwiera to furtkę do wejścia na rynek gier mobilnych, który w ostatnich latach przeżywa rozkwit (rys. 19). Decyzja o pominięciu tak dużej liczby odbiorców musiałaby być poparta poważnymi argumentami. W obecnej fazie projektu nie ma żadnych przeciwskazań, aby wydać go na takie platformy. Niezbędnym byłoby jednak przeprojektowanie interfejsu użytkownika tak, aby był kompatybilny z ekranem dotykowym.



Rys. 19 Stan rynku branży gier na kwiecień 2018

## 5.2 Rozszerzenie zawartości

Wiele pomysłów na rozwój projektu musiało zostać odłożonych na bok, głównie ze względu na ograniczenia czasowe. Funkcjonalności takie jak drzewko rozwoju technologicznego, tryb samouczka, przeciwnicy komputerowi, osiągnięcia czy nowe budynki i jednostki uczyniłyby grę ciekawszą.

## 5.3 Reklamy

Popularnym modelem stosowanym w branży gier mobilnych jest wprowadzenie reklam, których można pozbyć się przez wykupienie płatnej subskrypcji. Taki model jest nawet zaimplementowany w silniku Unity. Jeżeli projekt kiedykolwiek miałby zostać wydany na platformy mobilne, to zastosowanie takiego rozwiązania będzie nieuniknione. Pozwoli to na inwestycje w skalowanie aplikacji lub w przypadku ogromnego sukcesu, na zatrudnienie innych pracowników wspomagających dalszy rozwój gry.

## 5.4 Ulepszenie warstwy graficznej

Obecny projekt warstwy graficznej jest nieskomplikowany, co niesie swoje wady i zalety. Do pozytywnych stron takiej szaty graficznej należą na pewno niska ilość ładowanych obiektów a także niskie zużycie zasobów obliczeniowych procesora graficznego. W obecnych czasach jest to jednak coraz mniejszy problem, a twórcy gier mogą pozwolić sobie na coraz lepsze efekty graficzne, nawet na urządzeniach mobilnych. Wyłania się więc wada obecnego rozwiązania: produkt może nie wyróżnić się na tle innych, gotowych już do pobrania gier. Z tego powodu, jeżeli projekt byłby kontynuowany, niezbędnym będzie poprawienie warstwy graficznej. Pomocne okaże się poszerzenie wiedzy w temacie komputerowego modelowania trójwymiarowych obiektów w programach takich jak Blender, który jest darmowym narzędziem z dużą społecznością, a co za tym idzie, dużą ilością materiałów szkoleniowych i kursów.

# 6 Podsumowanie

Wynikiem prac jest gra na platformę Windows i MacOS spełniająca wszystkie przyjęte wymagania. Implementacja gry okazała się zadaniem trudnym, ale głównie ze względu na potrzebę tworzenia warstwy audiowizualnej.

Zastosowane rozwiązania sprawdziły się znakomicie. Architektura LAMP pozwala na błyskawiczny i nieprzerwany dostęp do serwerów gry odpowiadających za rejestrację   
i logowanie, jednocześnie pozwalając na bardzo łatwe dodawanie nowych usług. Silnik Unity okazał się narzędziem bardzo intuicyjnym, dobrze udokumentowanym   
i posiadającym wiele materiałów szkoleniowych wyprodukowanych przez jego bogatą społeczność. Wtyczka Photon Unity Networking 2 zdecydowanie ułatwiła synchronizację danych przez sieć internetową dzięki swojemu intuicyjnemu systemowi „zdarzeń”. System kontroli wersji Git doskonale sprawdził się jako narzędzie ostatecznego ratunku, kiedy podczas długiego poszukiwania błędu zaistniała potrzeba powrotu do poprzedniej działającej odsłony programu.

Przeprowadzone na grupie czterech osób testy wykazały, że produkt nie przejawia krytycznych błędów i jest grywalny, ale szybko nudzi graczy. Z tego powodu   
w przyszłości planuje się rozpoczęcie projektu od nowa z wiedzą nabytą podczas projektowania tej wersji aplikacji, tak aby wdrożyć wszystkie funkcjonalności wymienione w rozdziale „5 Dalszy rozwój projektu”, skupiając się głównie na dostępności gry na platformy mobilne oraz poszerzeniu zawartości samej gry. Tak przygotowany hipotetyczny produkt byłby gotowy do wypuszczenia na rynek   
i konkurowania z innymi podobnymi do siebie tytułami gatunku Indie, czyli gier tworzonych przez jednego programistę lub ich niewielki zespół.

# Bibliografia

|  |  |
| --- | --- |
| [1] | GRYOnline.pl, „Encyklopedia gier: Gry Strategiczne,” [Online]. Available: https://www.gry-online.pl/gry/strategiczne/. [Data uzyskania dostępu: 6 Luty 2023]. |
| [2] | B. Mazan, Bez antypodów? Zbliżenia i konfrontacje kultur., Łódź: Fundacja Uniwersytetu Łódzkiego, 2008. |
| [3] | GamesBeat, „Civilization: 25 years, 33M copies sold, 1 billion hours played, and 66 versions,” [Online]. Available: https://venturebeat.com/games/civilization-25-years-66-versions-33m-copies-sold-1-billion-hours-played/. [Data uzyskania dostępu: 7 Luty 2023]. |
| [4] | GRYOnline.pl, „Encyklopedia gier: Hearts of Iron IV PC,” [Online]. Available: https://www.gry-online.pl/gry/hearts-of-iron-iv/z93bd0. [Data uzyskania dostępu: 7 Luty 2023]. |
| [5] | GRYOnline.pl, „Encyklopedia gier: XCOM 2,” [Online]. Available: https://www.gry-online.pl/gry/xcom-2/za42a3. [Data uzyskania dostępu: 7 Luty 2023]. |
| [6] | Dualshockers, „XCOM 2 Sells Over 500,000 Copies in Its First Week,” [Online]. Available: https://web.archive.org/web/20161222082930/http://www.dualshockers.com/2016/02/10/xcom-2-sells-over-50000-copies-in-its-first-week/. [Data uzyskania dostępu: 07 Luty 2023]. |
| [7] | Computerworld, „The A-Z of Programming Languages: C#,” [Online]. Available: https://web.archive.org/web/20150402120752/https://www.computerworld.com.au/article/261958/a-z\_programming\_languages\_c\_/?pp=2. [Data uzyskania dostępu: 10 Luty 2023]. |
| [8] | Exit Games, „Photon Unity Networking 2,” [Online]. Available: https://doc-api.photonengine.com/en/pun/v2/index.html. [Data uzyskania dostępu: 10 Luty 2023]. |
| [9] | Refactoring guru, „Singleton,” [Online]. Available: https://refactoring.guru/design-patterns/singleton. [Data uzyskania dostępu: 11 Luty 2023]. |

# Spis ilustracji

[Rys. 1 Model planszy do gry w Go z grobowca dynastii Sui, 581–618 n.e. 6](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742589)

[Rys. 2 Zrzut ekranu z gry Civilization VI 7](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742590)

[Rys. 3 Zrzut ekranu z gry Hearts of Iron IV 8](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742591)

[Rys. 4 Zrzut ekranu z gry XCOM 2 9](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742592)

[Rys. 5 Projekt planszy do gry w "Ryzyko" 10](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742593)

[Rys. 6 Projekt budynków. Od lewej: koszary, laboratorium i farma. 11](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742594)

[Rys. 7 Jednostka gracza czerwonego 12](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742595)

[Rys. 8 Diagram stanów- interakcje interfejsu użytkownika 15](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742596)

[Rys. 9 Diagram stanów- stan "Rozgrywka" 15](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742597)

[Rys. 10 Ekran rejestracji 16](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742598)

[Rys. 11 Ekran menu głównego 18](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742599)

[Rys. 12 Ekran tworzenia rozgrywki 18](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742600)

[Rys. 13 Ekran rozgrywki w testowym trybie dla jednego gracza 19](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742601)

[Rys. 14 Inspektor pola w wariancie dla posiadanego terenu 20](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742602)

[Rys. 15 Tryb ataku w sytuacji możliwego natarcia z wielu terenów jednocześnie 21](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742603)

[Rys. 16 Powiadomienie wyświetlane graczowi czerwonemu 22](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742604)

[Rys. 17 Schemat metody decydującej o wyświetleniu kolejnego powiadomienia 26](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742605)

[Rys. 18 Przykładowa lista referencji 27](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742606)

[Rys. 19 Stan rynku branży gier na kwiecień 2018 30](https://tulodz-my.sharepoint.com/personal/222616_edu_p_lodz_pl/Documents/Praca%20inżynierska%20Tomasz%20Wojtkiewicz.docx#_Toc127742607)

1. Serwis WWW dostarczający skalowalną moc obliczeniową w chmurze obliczeniowej [↑](#footnote-ref-2)