Politechnika Wrocławska

Wydział Informatyki i Zarządzania
Kierunek studiów: Informatyka

Praca dyplomowa - inżynierska

Projekt i implementacja gry zręcznościowej na urządzenia mobilne opartej o tematykę endless runner z systemem rankingowym i osiągnięciami

Adrian Ślimak

słowa kluczowe:
android, gra mobilna, endless runner

krótkie streszczenie:
Niniejsza praca obejmuje najistotniejsze elementy powstawania zręcznościowej gry mobilnej, przybliża tematykę i pokazuje jej pozycję na rynku oraz przedstawia zastosowane rozwiązania i użyte technologie.

Opiekun pracy dyplomowej	dr inż. Bernadetta Maleszka	.....	.....
	Tytuł/stopień naukowy/imię i nazwisko	*ocena*	*podpis*
Ostateczna ocena za pracę dyplomową			
Przewodniczący Komisji egzaminu dyplomowego	.....	.....	.....
	Tytuł/stopień naukowy/imię i nazwisko	*ocena*	*podpis*

*Do celów archiwalnych pracę dyplomową zakwalifikowano do:**

a) *kategorii A (akta wieczyste)*
b) *kategorii BE 50 (po 50 latach podlegające ekspertyzie)*

* *niepotrzebne skreślić*

pieczątka wydziałowa

Wrocław, 2018

Streszczenie

Niniejsza praca ma na celu przedstawienie procesu tworzenia gry, przeznaczonej na urządzenia mobilne oparte o system Android. Jest to zręcznościowa gra platformowa z gatunku endless runner. Rozgrywka dostępna jest w dwóch trybach, normalnym, opierającym się na zmianie sposobu sterowania pojazdem oraz treningowym, który dodatkowo umożliwia zatrzymywanie i cofanie czasu. Oprócz tego dostępny jest również sklep, pozwalający nabyć nowy wygląd i kolory pojazdów. Użytkownicy mogą konkurować ze sobą za pomocą systemu rankingowego oraz możliwych do zdobycia osiągnięć za postępy w grze.

Na początku pracy znajduje się wprowadzenie do tematyki oraz prezentacja rozwiązań konkurencyjnych, obecnych na rynku. Następnie przedstawiono wykorzystywane narzędzia i omówiono projekt gry, wraz z wymaganiami. Później przyszedł czas na rozdział poświęcony implementacji, zawierający rozwiązania części funkcjonalności. Na końcu znalazły się testy wykonywane w czasie powstawania aplikacji, a także podsumowanie i perspektywy rozwoju.

Abstract

This thesis aims to present the process of developing game, designed for mobile devices with Android system. It is an arcade platform game of the endless runner genre. The game is available in two modes, normal, based on the change of the vehicle control method and training, which also allows to stop and reverse time. In addition, there is a store that allows to purchase new skins and colors of vehicles. Users can compete with each other using a ranking system and achievements earned for progress in game.

At the beginning of the thesis, there is an introduction to the subject and the presentation of competitive solutions, present on the market. Next, the tools used were presented and the game project along with requirements was discussed. Later, the time came for the chapter devoted to implementation, including solutions for part of the functionality. Finally, there were tests performed during application development, as well as a summary and development perspectives.

Spis treści

1. Wstęp

Niniejszy rozdział stanowi wstęp do tematyki pracy. Skupia się na przybliżeniu pojęcia gier i ich rozwoju, a także przedstawieniu ich znaczenia w dzisiejszym świecie oraz pozycji na rynku. Następnie znajduje się krótki opis gry będącej podmiotem pracy, a także gatunku, do którego przynależy.

1.1 Wprowadzenie do tematyki

Gry są obecne wśród ludzi od bardzo dawna. Na początku miały one charakter fizyczny, były to przede wszystkim gry sportowe, ale także umysłowe, a nawet hazardowe. Wraz z rozwojem cywilizacji i postępem technologicznym, zyskały one zupełnie nowe znaczenie i powstały pierwsze urządzenia przeznaczone do grania.

W dzisiejszych czasach, słowo „gra" jest kojarzone głównie z grą komputerową. Nazwa ta niekoniecznie odnosi się do tradycyjnego komputera, ponieważ obecnie gry dostępne są również na wielu urządzeniach przenośnych, takich jak dedykowane temu celu konsole, a także znacznie bardziej popularne, które większość nosi na co dzień w kieszeni – telefony i współczesne smartfony.

Gry komputerowe można podzielić na wiele rodzajów, takich jak: zręcznościowe, strategiczne, edukacyjne, karciane, rozrywkowe, platformowe… Ważne jest jednak, że bardzo często nie są one ograniczone ściśle do jednej kategorii, lecz łączą w sobie cechy z różnych gatunków. Przy współczesnym etapie rozwoju techniki, niekiedy odchodzi się również od tradycyjnych metod sterowania, czyli myszek, klawiatur i padów, na rzecz bezprzewodowych kontrolerów ruchu i okularów wirtualnej rzeczywistości. Dzięki takim rozwiązaniom, zupełnie nowego znaczenia zyskują gry zręcznościowe, które wymagają rzeczywistego wysiłku fizycznego i cieszą się rosnącą popularnością.

Gry najczęściej utożsamiane są z rozrywką, pewną formą spędzania wolnego czasu. Jest to ich nieodłączny element, ponieważ z taką myślą zostały stworzone. Istnieją jednak produkcje, które znalazły odpowiednie zastosowanie w wielu różnych dziedzinach. W medycynie są wykorzystywane na przykład jako skuteczna forma walki z Alzheimerem i demencją u starszych osób. Dzieci już od najmłodszych lat mogą korzystać ze specjalnych programów edukacyjnych, zawierających również elementy zabawy, co znacznie przyspiesza i usprawnia ich naukę. Dzięki znacznemu rozwojowi gier wieloosobowych i rozgrywek sieciowych w ostatnich latach, obecnie w wielu krajach gry uznawane są jako sport i organizowane są specjalne turnieje, w których udział biorą profesjonalne drużyny, a ich zmagania oglądają miliony osób z całego świata.

1.2 Obecność gier na rynku

Popularność gier komputerowych stale rośnie, a co za tym idzie, umacnia się również ich obecność na rynku. Początkowo miało to związek z postępem technologicznym, który pozwalał na coraz większą miniaturyzację i wpłynął na zmniejszenie kosztów produkcji, co zwiększało dostępność komputerów osobistych. Już w latach 70. XX wieku powstały pierwsze przenośne konsole, które były prostymi urządzeniami z niewielkimi wyświetlaczami, oferując tylko jedną grę. Z czasem stały się one bardziej złożone i miały znacznie więcej do zaoferowania, na przykład klasyczny GameBoy od Nintendo lub PSP firmy Sony.

Sytuację diametralnie zmieniło pojawienie się pierwszych smartfonów z systemem Android i IOS. Przenośne konsole przestały zyskiwać na popularności, a aktualnie są już wypierane przez smartfony i tablety. Te z kolei z roku na rok zdobywają coraz większą popularność, a w obecnym 2018 r., po raz pierwszy przychód z segmentu mobilnego będzie stanowił ponad połowę wszystkich przychodów z gier. Zmianę tego udziału na przestrzeni poprzednich lat oraz prognozę na przyszłość można zaobserwować na rysunku 1.1, znajdującym się poniżej.

Rys. 1.1. Przychody z rynku gier na świecie w latach 2012-2021.
Źródło: [16]

Biorąc pod uwagę wyłącznie segment mobilny, najpopularniejszym systemem jest obecnie Android. Zyskuje on uznanie wśród użytkowników, nieustannie już od czasu swojej pierwszej wersji w 2008r. W drugim kwartale 2018r., prawie 88% sprzedanych smartfonów było wyposażonych właśnie w rozwiązanie firmy Google. Biorąc pod uwagę te dane, zdaje się on być dobrym kierunkiem rozwoju wśród programistów.

1.3 Cel i zakres pracy

Celem niniejszej pracy inżynierskiej jest opracowanie gry na urządzenia mobilne z systemem Android. Będzie to zręcznościowa, dwuwymiarowa gra platformowa, należąca do gatunku Endless Runner. Wyróżnia się on tym, że podczas rozgrywki przejmujemy kontrolę nad postacią lub pojazdem, który nieustannie i automatycznie porusza się do przodu. Zadaniem użytkownika natomiast jest wykonywanie pewnych akcji, zazwyczaj dość prostych, mających na celu omijanie różnorodnych przeszkód i w konsekwencji zajście jak najdalej.

Do zakresu pracy należą:

- Opracowanie pomysłu na grę.
- Zapoznanie się z istniejącymi rozwiązaniami.
- Zapoznanie się ze sposobami tworzenia gier na urządzenia z systemem android.
- Wybór odpowiednich narzędzi i środowiska do implementacji gry.
- Zaprojektowanie i utworzenie interfejsu użytkownika.
- Zaprojektowanie poziomów gry, przygotowanie elementów graficznych.
- Implementacja mechanik sterowania oraz elementów rozgrywki.
- Utworzenie systemu zakupów, pozwalającego na zmianę wyglądu pojazdu w grze.
- Utworzenie systemu osiągnięć.
- Integracja z Google Play Game Services oraz utworzenie systemu rankingowego.
- Zaprojektowanie, utworzenie i połączenie bazy danych SQLite z aplikacją.

2. Przegląd istniejących rozwiązań

Na rynku dostępnych są tysiące rozmaitych gier i ciągle powstają nowe. Przyjrzenie się im wszystkim i przeanalizowanie ich byłoby więc bardzo trudne. W związku z tym, ten rozdział skupi się na przedstawieniu kilku najbardziej popularnych tytułów, należących do tej samej kategorii co gra autorska. Przedstawione zostaną ich główne funkcjonalności i zalety, a także wady.

2.1 Temple Run

Rys. 2.1. Temple Run – gra zręcznościowa.
Źródło: [19]

Temple Run jest darmową grą, dostępną na platformy android oraz IOS, której premiera miała miejsce w 2011 roku. Fabuła gry jest bardzo prosta, bohater jest poszukiwaczem przygód, który wszedł do świątyni, ukradł przeklętą figurkę i teraz jest ścigany przez małpy, które starają się odzyskać swój skarb. Gracz ma za zadanie pomóc mu w ucieczce i wykorzystać swój refleks przy omijaniu przeszkód. Postać sama biegnie do przodu, natomiast wykonując proste akcje polegające na odpowiednim przesunięciu palcem, można skakać, ślizgać się i skręcać w prawo lub w lewo na rozwidleniu drogi. Ponadto, wykorzystując żyroskop zawarty w smartfonach, mamy możliwość zmiany toru ruchu postaci. Poziom generowany jest proceduralnie, więc gra nigdy się nie skończy.

Cel gry, jak i sama rozgrywka jest bardzo prosta. Polega na tym, żeby dobiec jak najdalej i uzyskać jak najwyższy wynik. Po drodze rozrzucone są specjalne monety, które warto zbierać, ponieważ posiadając odpowiednią ich liczbę można kupić cenne przedmioty. Wśród nich znajdują się specjalne ulepszenia, które znacznie ułatwiają grę, jak na przykład magnes przyciągający monety, lub ochrona, która zabezpiecza przed wpadnięciem na przeszkodę. Oprócz tego, możliwe jest odblokowanie nowych postaci, a następnie wykorzystanie ich

podczas rozgrywki. Nie mają one jednak żadnych unikatowych właściwości i różnią się jedynie wyglądem. Aplikacja zawiera również system osiągnięć zdobywanych za postępy w grze oraz statystyki zawierające informacje takie jak najwyższy uzyskany wynik, czas gry i liczbę powtórzeń.

Gra bardzo szybko uzyskała dużą popularność wśród graczy, a obecnie ma ponad 100 milionów pobrań. Zawdzięcza to przede wszystkim oryginalnemu pomysłowi i niewielką konkurencją na rynku, ponieważ rok 2011 był początkiem rozwoju gier mobilnych. Oprócz tego, posiada ciekawą oprawę graficzną i prosty, lecz wciągający system sterowania. Przyczyniła się do powstania wielu, bardzo podobnych produkcji, które zyskały mniejszą lub większą popularność. Opierają się one na tym samym schemacie, a różnią przede wszystkich wizualnie. Doczekała się kilku oficjalnych kontynuacji, które również cieszyły się wysoką popularnością.

Niestety, jak każda produkcja posiada ona również swoje wady. Z biegiem czasu i zyskiwaniem uznania wśród graczy do gry zostały wprowadzone reklamy. Są one bardzo uciążliwe, ponieważ pojawiają się po każdej przegranej grze, kiedy chcemy zagrać jeszcze raz. Za prawdziwe pieniądze można zakupić monety w grze, a następnie wymienić je na przedmioty ze sklepu, lecz niestety nie są to małe wartości, tym bardziej, że gra jest nastawiona wyłącznie na rozgrywkę jednoosobową. Prostota tej gry posiada wiele zalet, ponieważ przyswojenie zasad wymaga mało wysiłku, dzięki czemu jest przystępna dla każdego. Wiąże się to jednak z tym, że gra bardzo szybko się nudzi.

2.2 Jetpack Joyride

Rys. 2.2. Jetpack Joyride – gra platformowa.
Źródło: [20]

Rys. 2.3 Jetpack Joyride – gra platformowa.
Źródło: [20]

Jetpack Joyride jest grą platformową, która zdobyła dużą popularność i osiągnęła ponad 500 milionów pobrań. Swoja premierę miała w 2011 roku na urządzeniach z systemem IOS. Rok później ukazała się w wersji na komputery i system Android. Jej duży sukces sprawił, że zdecydowano się wydać ją również na Windows Phone, a także na konsole firmy Sony, lecz nastąpiło to w późniejszym czasie. Gra jest wspierana poprzez aktualizacje do tej pory, a nawet doczekała się wielu konkurencyjnych rozwiązań, którym jednak nie udało się naruszyć jej pozycji.

Historia gry nie jest skomplikowana, a bohaterem jest pracownik przedsiębiorstwa, które produkuje gramofony. Pewnego razy wracając do domu przechodził obok tajnego laboratorium, gdzie znalazł plecak odrzutowy. Mężczyzna włamuje się, kradnie go i w ten sposób rozpoczyna się rozgrywka. Ta z kolei jest pozornie dość prosta, ponieważ głównym celem jest omijanie przeszkód poruszając się w górę i w dół, a postać sama porusza się w prawą stronę. Dużym urozmaiceniem jest zastosowany tutaj system bonusów, które po zebraniu na krótki okres zmieniają system sterowania. Dzięki temu, gracz oprócz podstawowego plecaka odrzutowego, ma możliwość jazdy samochodem, a nawet wcielenia się w ogromnego smoka lub robota. Po drodze można zbierać monety i strzelać do pracowników laboratorium, co zwiększa liczbę otrzymywanych punktów.

Gra jest ładna wizualnie i posiada system rankingowy, który wprowadza dodatkowy element rywalizacji, co przyciąga graczy. Do tego zawiera rozbudowany system ulepszeń, ułatwiających rozgrywkę, które można zakupić za zdobyte monety. Pozwalają one również na wizualną zmianę wyglądu pewnych elementów takich jak pojazdy i ubiór bohatera. Monety w grze można również zakupić za prawdziwą walutę, niestety nie są to małe sumy jak na tak drobne zmiany, które możemy dzięki nim uzyskać.

2.3 Crossy Road

Rys. 2.4. Crossy Road – gra akcji.
Źródło: [21]

Crossy Road jest najmłodszą grą w tym zestawieniu, gdyż swojej premiery doczekała się pod koniec roku 2014. Na jej ogromną popularność wpłynął ciekawy pomysł, który wziął się z żartu „Dlaczego kurczak przeszedł prze ulicę?", jak również brak podobnych tytułów dostępnych wówczas na rynku. Grafika została wykonana techniką „Low Poly", co oznacza, że modele składają się z niewielkiej liczby wielokątów. Pomimo swojej prostoty, pod względem wizualnym prezentuje się bardzo dobrze. Doczekała się ona kontynuacji, która różni się jedynie stylistyka wzorowaną na świecie bajek Disney'a. Powstało również bardzo wiele kopii tej gry, nieznacznie się różniących.

Gra nie posiada żadnej otoczki fabularnej. Jej jedynym celem jest pomoc w przejściu wybranej postaci jak najdalej. Po drodze należy wykazać się czujnością w celu omijania samochodów, pociągów, skakania po platformach płynących na rzece i wielu innych przeszkód, z których zetknięciem kończy się gra. Na planszy znajdują się również monety, które można kolekcjonować, a następnie wykorzystać w specjalnej maszynie, odblokowującej jedną losową postać. Sterowanie jest bardzo proste, należy przesunąć palcem w odpowiednim kierunku, a nasz bohater poruszy się w przód, tył, prawo lub lewo. Można się również zatrzymać, co wyróżnia tę produkcję z innych przytoczonych, lecz kamera porusza się cały czas i zbyt długa zwłoka zakończy się przegraną.

Wraz ze wzrostem pobrań, w grze nastąpił szereg zmian. Dodano wiele nowych poziomów i postaci. Niestety wprowadzono również możliwość zakupienia konkretnych bohaterów za prawdziwe pieniądze, przez co spadła ich wartość związana z elementem losowości. Jedyną formą rywalizacji jest system rankingowy, który dostępny jest dopiero po połączeniu gry z kontem Facebook.

2.4 Geometry Dash

Rys. 2.5. Geometry Dash – gra zręcznościowa.
Źródło: [22]

Rys. 2.6. Geometry Dash – gra zręcznościowa.
Źródło: [22]

Geometry Dash ukazało się na rynku w 2013 roku. Od tamtej pory pobrano ją ponad milion razy, co jest świetnym wynikiem, lecz znacznie odbiega od poprzednio wymienionych tytułów. Rozbieżności te wynikają głównie z faktu, że jest jedyną płatna grą w tym zestawieniu. Gra doczekała się dwóch darmowych rozszerzeń, wymagających do uruchomienia podstawowej wersji, które zawierają dodatkowe poziomy. Pod względem wizualnym prezentuje się na wysokim poziomie. Zawiera grafiki wysokiej rozdzielczości i dużo ciekawych efektów, z którymi niestety przesadzono na niektórych poziomach.

Gra wyróżnia się od poprzednich, ponieważ nie posiada nieskończonych poziomów. Są one zaprojektowane, a nie generowane proceduralnie, co przekłada się na ich wysoką różnorodność i dopracowanie. System sterowania jest bardzo prosty i polega jedynie na dotknięciu ekranu w odpowiednim momencie, natomiast postać sama porusza się do przodu. Mimo to, rozgrywka jest bardzo urozmaicona ze względu na zastosowanie wielu różnych mechanik poruszania się postaci i płynnym przejściu między nimi w trakcie gry. Nie jest to również pozycja dla każdego, ponieważ gra jest bardzo trudna. Wymaga poświęcenia znacznej ilości czasu na przejście kolejnych poziomów oraz wysokiej zręczności. Gracz ma do dyspozycji specjalny tryb praktyki, służący do nauki i ćwiczeń, w którym po przegranej nie zaczyna się od nowa, lecz od ostatniego punktu kontrolnego. Niestety nie jest on zbyt dopracowany, ponieważ punkty te są tworzone w stałych odstępach czasowych, co zmniejsza naszą kontrolę i często utrudnia grę.

W produkcji znalazł się system rankingowy i zakupowy, gdzie mamy możliwość zmiany wyglądu postaci za punkty zdobyte w grze. Znajdziemy tu również rozbudowane narzędzie do tworzenia nowych poziomów, przeznaczone dla graczy. Mogą oni dzielić się swoimi dziełami, a także grać w te opublikowane przez innych. Znacznie urozmaica to grę i przedłuża jej żywotność na rynku. Funkcjonalności te są niestety dostępne dopiero po utworzeniu dodatkowego konta w grze i zalogowaniu się na nim.

2.5 Podsumowanie przeglądu

Każda z aplikacji wymienionych w tym rozdziale była bardzo popularna na rynku urządzeń mobilnych i posiada ogromną liczbę pobrań. Zawdzięczają to przede wszystkim pojawieniu się na nim w odpowiednim czasie i innowacyjnemu wówczas pomysłowi, co ze względu na brak konkurencji przyczyniło się do szybkiego wzrostu zainteresowania wśród użytkowników. Był to jedynie tymczasowy sukces, ponieważ gry stosunkowo szybko zaczęły się nudzić, a gracze poszukiwali nowych tytułów. Główną przyczyną tego była prosta, niewymagająca rozgrywkę i brak możliwości rozwoju, który polegał jedynie na drobnych zabiegach wizualnych związanych ze zmianą wyglądu postaci i innych elementów, które w żaden sposób nie urozmaicały rozgrywki. Wyjątkiem od tej reguły jest gra Geometry Dash, która oferuje zróżnicowany system sterowania oraz edytor poziomów, co pozwoliło zatrzymać wielu graczy na dłużej i sprawiło, że gra nadal jest dość popularna.

Z biegiem czasu powstały narzędzia, które znacznie ułatwiły tworzenie gier przeznaczonych na systemy mobilne. Wpłynęło to na dużą liczbę dostępnych produkcji, z korzyścią dla użytkowników. Znacznie jednak utrudniło deweloperom wydanie tytułu, który będzie się wyróżniał oraz zdobędzie uznanie i popularność wśród graczy. Dlatego wiele produkcji jest do siebie bardzo podobnych, różniących się głównie elementami wizualizacji i poziomem zaprojektowania poszczególnych elementów takich jak interfejs, osiągnięcia czy system sterowania.

Większość tytułów dostępnych obecnie na rynku, cieszących się wysokim zainteresowaniem, są grami przy których przez długi czas pracowały wieloosobowe studia. W związku z tym, bardzo dobrze prezentują się pod względem wizualnym i zawierają dopracowany system personalizacji z dużą liczbą elementów. Ze względu na te ograniczenia, gra będąca podmiotem tej pracy, konkuruje głównie poprzez oferowane funkcjonalności, prostotę w użytkowaniu i ciekawą rozgrywkę, stawiającą wyzwanie dla gracza.

Istotnym elementem wyróżniającym jest również specjalny tryb praktyki, który umożliwia kontrolę nad czasem, jego zatrzymywanie i cofanie. Ma to na celu umożliwić naukę trudnych fragmentów poziomu, bez konieczności jego powtarzania po każdej przegranej, co jest częstą przyczyną zniechęcenia graczy do dalszej rozgrywki i skutkuje porzuceniem gry. Oprócz tego udało się dostosować aplikację do urządzeń z rożnymi proporcjami wyświetlacza, na przykład niestandardowego 18,5:9. Osiągnięto to zarówno dla interfejsu, jak i widoku rozgrywki, który dostosowuje pole widzenia i umożliwia grę na całym ekranie. W rozwiązaniach konkurencyjnych, częstym zabiegiem w tym przypadku jest wypełnienie obrazu czarnymi paskami po bokach, a nawet jego rozciągnięcie lub ucięcie, co znacznie zmniejsza komfort użytkowania i niekiedy całkowicie odbiera dostęp do niektórych funkcjonalności.

Krótkie podsumowanie najważniejszych funkcjonalności w przytoczonych aplikacjach, przedstawia tabela poniżej.

	Temple Run	Jetpack Joyride	Crossy Road	Geometry Dash	Gra autorska
Płatności	darmowa, mikropłatności	darmowa, mikropłatności	darmowa, mikropłatności	płatna, mikropłatności	w pełni darmowa
Reklamy	wymagane po każdej przegranej	opcjonalne	opcjonalne	brak	brak
Prosty, intuicyjny interfejs	nie	nie	tak	tylko częściowo	tak
System rankingowy	brak	dostępny	dostępny	dostępny	dostępny
System osiągnięć	dostępny	dostępny	dostępny	dostępny	dostępny
System personalizacji postaci gry	dostępny, dość prosty	dostępny	dostępny	dostępny	dostępny
Rozbudowana, wymagająca rozgrywka	nie	tak	nie	tak	tak
Tryb praktyki pozwalający na naukę poszczególnych etapów	brak	brak	brak	dostępny (punkty kontrolne)	dostępny (zatrzymywanie i cofanie czasu)
Sposób logowania	brak	konto Google	konto Facebook	wymaga założenia nowego konta	konto Google

Tabela 2.1. Podsumowanie przeglądu rozwiązań konkurencyjnych

3. Technologie i narzędzia

W tym rozdziale zostaną omówione podstawowe narzędzia, dzięki którym możliwe było utworzenie gry będącej podmiotem tej pracy oraz technologie, które umożliwiły zrealizowanie niektórych funkcjonalności. Oprócz tego przedstawione zostanie główne środowisko, gdzie wykonana została większość prac.

3.1 Android

Android jest systemem operacyjnym opartym na jądrze linuks, przeznaczonym dla urządzeń mobilnych, takich jak smartfony i tablety. Jego oprogramowanie jest wydane na otwartej licencji. Posiada wygodną platformę służącą do pobierania, a także publikowania aplikacji o nazwie Google Play, która oferuje ogromną bibliotekę przyciągającą wielu użytkowników. Dzięki temu, Android cieszy się aktualnie największą popularnością pośród systemów mobilnych. W związku z tym, stanowi on docelową platformę wielu programistów i deweloperów.

Dostępnych jest wiele środowisk, które umożliwiają tworzenie aplikacji na Androida. Niektóre z nich jak na przykład Eclipse IDE i IntelliJ IDEA wymagają pobrania dodatkowych rozszerzeń. W ostatnim czasie straciły one wsparcie i popularności ze względu na wydanie oficjalnego Android Studio, które jest znacznie bardziej rozbudowane i wygodniejsze w użyciu, gdyż nie wymaga dodatkowych konfiguracji. Wszystkie te programy oparte są o Android SDK, czyli udostępniony przez Google zestaw narzędzi, w którego skład wchodzą między innymi dokumentacja, biblioteki, debugger i emulator. Umożliwia on również tworzenie gier na Androida wykorzystując silnik Unity.

3.2 Unity

Unity jest popularnym, wieloplatformowym silnikiem gier, wydanym przez Unity Technologies w 2005 roku, wykorzystywanym głównie do tworzenia gier trójwymiarowych, lecz możliwe jest również produkowanie gier 2D, a nawet wizualizacji i animacji. Można w nim zrealizować każdy etapy budowy gry, co sprawia, że jest kompletne i bardzo elastyczne. Dostarczone w nim narzędzia zapewniają szerokie możliwości, które pozwalają w dodatku na dostosowane ich do własnych potrzeb. Znaleźć w nim można, między innymi narzędzie do projektowania interfejsu, mechanizmy animacji i gotowe komponenty służące do realizacji fizyki. Oferuje również zintegrowane środowisko programistyczne MonoDevelop, przeznaczone dla C#, który jest głównym językiem używanym do pisania skryptów. Nie jest jednak obowiązkowe, ponieważ istnieje możliwość prostej integracji z zewnętrznym narzędziem Visual Studio, które jest znacznie bardziej rozbudowane i wygodniejsze w praktyce.

Unity zapewnia wsparcie aż dla 27 różnych platform docelowych. Oznacza to, że utworzone z jego pomocą gry mogą zostać wydane na komputery osobiste (Windows, Linux), urządzenia mobilne z systemami Android, IOS oraz Windows Phone, konsole Xbox i PlayStation, a nawet telewizory z systemem Android TV i okulary wirtualnej rzeczywistości. Silnik pozwala na kompletne przetestowanie gry w obrębie edytora, bez konieczności posiadania platformy docelowej. Stanowi to ogromne ułatwienie dla programistów, ponieważ gry nie trzeba za

każdym razem budować i instalować na przykład na smartfonie, w celu przetestowania pewnych funkcjonalności lub poprawianych błędów.

W przypadku tego projektu, docelową platformą będzie Android. Zanim możliwe było zbudowanie gry i utworzenie pliku .apk, służącego do instalacji aplikacji na urządzeniu, należało wyposażyć Unity w odpowiednie narzędzia. Pierwszym krokiem było zainstalowanie Java Development Kit (JDK) i Android SDK, a następnie wskazanie ścieżki do folderu zawierającego te biblioteki.

3.3 SQLite

SQLite jest systemem zarządzania relacyjnymi bazami danych, zapewniającym obsługę języka SQL. W przeciwieństwie do innych SQL'owych baz, nie korzysta z oddzielnego procesu pracującego w tle na serwerze, tylko na pojedynczym, fizycznym pliku, w którym zawarta jest kompletna baza danych. Plik ten może być swobodnie kopiowany i wysyłany, a także działać na różnych platformach. Nie wymaga również instalacji, wystarczy obecność w odpowiednim miejscu pliku biblioteki. Ze względu na to, jest świetnym rozwiązaniem dla aplikacji mobilnych, gdzie wymagana jest lokalna baza danych, która musi działać bez nadzoru i wsparcia.

Integracja SQLite z Unity jest bardzo prosta. Wystarczy pobrać odpowiednie pliki i umieścić je w folderze plugin'ów, a następnie w klasie obsługującej bazę zaimportować potrzebne biblioteki Bazę danych można zaprojektować i utworzyć przy pomocy konkretnych narzędzi np. SQLiteAdmin. Pomimo prostoty obsługi, program pozwala na tworzenie i edycję tabel, wykonywanie zapytań w celach testowych, wypełnianie bazy danymi z zewnętrznych źródeł i wiele więcej. Tak powstałą bazę, należy przenieść do folderu z projektem w Unity. Do utworzenia połączenia, w celu komunikacji z bazą, wystarczy znać jej adres URI i nazwę.

3.4 Play Game Services

Play Game Services jest rozszerzeniem przeznaczonym dla Unity, o otwartym charakterze z udostępnionym kodem źródłowym dla każdego. Jego zadaniem jest umożliwienie programistom integracji Google Play Games API z grą utworzoną w silniku Unity. Dzięki temu rozwiązaniu, w łatwy sposób można uzyskać w grze takie funkcjonalności jak logowanie i uwierzytelnianie wykorzystujące konto Google, system rankingowy, system osiągnięć, zapisywanie danych na serwerze, a nawet rozgrywkę wieloosobową w czasie rzeczywistym. Rozwiązanie to jest dostępne wyłącznie na systemy operacyjne Android.

Integracja polega na pobraniu rozszerzenia, a następnie zaimportowaniu go wykorzystując edytor Unity. Później należy przeprowadzić konfigurację gry w serwisie Google Play Console. Po ukończeniu całego etapu, uzyskuje się dostęp do wyżej wymienionych funkcjonalności.

3.5 Pozostałe

Visual Studio

Visual Studio jest zintegrowanym środowiskiem programistycznym, należącym do firmy Microsoft. Jego wszechstronność pozwala między innymi na wytwarzanie konsolowych i graficznych programów komputerowych, stron internetowych, aplikacji i serwisów webowych oraz aplikacji mobilnych. Zapewnia wsparcie aż dla 36 języków programowania, w tym C#, a dzięki zainstalowaniu odpowiedniego rozszerzenia możliwa jest integracja z silnikiem Unity. Połączenie to pozwala na pisanie skryptów i zarządzaniem całym projektem za pomocą Visual Studio, które jest znacznie bardziej rozbudowanym narzędziem, niż wbudowany w Unity, MonoDevelop. Sprawia to, że większość programistów Unity decyduje się na rozwiązanie od firmy Microsoft.

Środowisko dostępne jest w wielu równych wersjach, które różnią się możliwościami. W przypadku tego projektu, wykorzystana została płatna wersja Enterprise 2017. Było to możliwe dzięki programowi Miscrosoft DreamSpark, które zapewnia studentom z całego świata dostęp do swoich narzędzi. Oprogramowanie jednak udostępnione jest jedynie do użytku edukacyjnego.

Gimp

Gimp jest popularnym i bezpłatnym, otwarto źródłowym programem do edycji grafiki rastrowej. Funkcjonuje na wielu różnych systemach operacyjnych i jest rozpowszechniany na licencji GNU GPL. Stanowi odpowiednik płatnego narzędzia Adobe Photoshop.

Jest głównymi cechami są funkcjonalność i prostota obsługi. Umożliwia rysowanie, zmianę rozmiaru, perspektywy oraz obrót obrazów, wykonywanie różnorodnych operacji na kolorach, wykorzystanie warstw i mask służących do łączenia obrazów, a także konwersję pomiędzy różnymi formatami plików graficznych.

W projekcie autorskim wielokrotnie został wykorzystany przy tworzeniu interfejsów, elementów składowych poziomów i grafiki postaci. Znacznie ułatwił również modyfikację gotowych obrazów, które wymagały jedynie dostosowania do tworzonej gry.

Sourcetree

Sourcetree to bezpłatna aplikacja kliencka, która pozwala na obsługę repozytoriów Git, Mercurial oraz BitBucket. Narzędzie znacznie upraszcza zarządzanie repozytoriami, dzięki ich wizualizacji oraz prostemu i intuicyjnemu interfejsowi graficznemu. Ułatwia to pracę w stosunku do aplikacji konsolowych i pozwala skupić się wyłącznie na programowaniu. Jest z powodzeniem wykorzystywane zarówno przy małych, amatorskich projektach programistycznych oraz tych bardziej rozbudowanych, komercyjnych. Program pozwala na wyświetlanie kodu, a także wprowadzanie i odrzucanie zmian na poziomie pliku, fragmentu i linii kodu. Dzięki niemu możliwe jest bezproblemowe rozwiązywanie konfliktów pomiędzy wersjami projektu, klonowanie zdalnych repozytorium przez aplikację i szczegółowe śledzenie postępu prac wykorzystując diagramy gałęzi.

3.6 Podsumowanie

Rys. 3.1. Podsumowanie wykorzystanych narzędzi

4. Projekt aplikacji

Niniejszy rozdział poświęcony został przedstawieniu projektu gry autorskiej. Zaprezentowane zostaną podstawowe funkcjonalności oferowane przez aplikację w postaci diagramu przypadków, wymagania niefunkcjonalne, projekt interfejsu i struktura danych.

4.1 Krótkie przedstawienie gry

Podmiotem pracy jest gra, stworzona z myślą o urządzeniach mobilnych z systemem Android. Jest to dwuwymiarowa, zręcznościowa gra platformowa z gatunku endless runner, charakteryzującym się tym, że pojazd nieustannie i automatycznie porusza się do przodu, natomiast gracz musi unikać przeszkód.

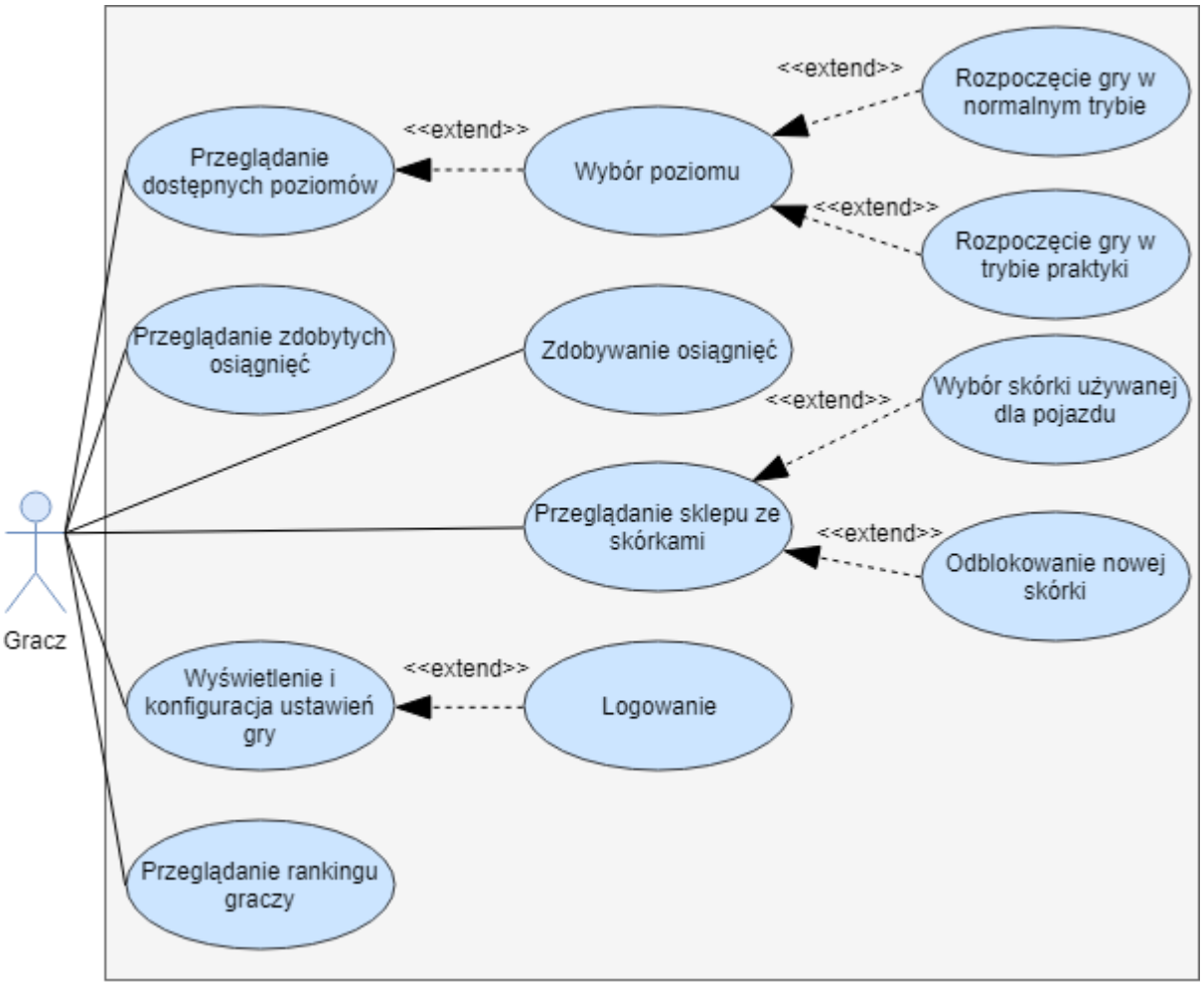
Rozgrywka dostępna jest w dwóch trybach. Normalny opiera się na zmianie mechaniki sterowania, które przypisane są do odpowiednich pojazdów. Przejście pomiędzy nimi jest płynne i następuje po wejściu w specjalny obiekt, zwany portalem. Gracz wykonując określone akcje, ma za zadanie omijać pojawiające się przeszkody. Dodatkowym zadaniem jest zebranie wszystkich ukrytych przedmiotów. Na poziomie znajdują się również obiekty, które pozwalają na specjalną interakcję z pojazdem, jak na przykład podwójny skok. Dostępny jest także tryb praktyki, który dodatkowo umożliwia zatrzymywanie i cofanie czasu. W tym przypadku punkty przyznawane za postęp na danym poziomie, a także zebrane ukryte przedmioty nie będą uwzględniane, ponieważ służy on wyłącznie w celach treningowych.

Gra posiada sklep, pozwalający zmienić, a także zakupić nowy wygląd, czyli tak zwane skórki pojazdów. Oprócz tego gracz może również nabyć i dostosować ich kolory. Przedmioty te można odblokować dzięki punktom oraz osiągnięciom zdobytym za postęp w grze.

Użytkownicy mają dostęp do systemu rankingowego, co wprowadza dodatkowy element rywalizacji. W grze znajduje się również system osiągnięć, które odblokować można za poczynione postępy, jak na przykład zebranie wszystkich ukrytych przedmiotów na danym poziomie.

4.2 Wymagania funkcjonalne

Poniżej znajduje się rysunek przedstawiający diagram przypadków użycia, a następnie tabele, które są uściśleniem poszczególnych przypadków. Jest to forma dokumentacji funkcjonalności aplikacji, bazująca na diagramie i specyfikacji przypadków użycia.



Rys. 4.1. Prezentacja wymagań funkcjonalnych – diagram przypadków użycia

Sekcja	Treść
Przypadek użycia	PU1 - Przeglądanie dostępnych poziomów
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Niski
Warunki wstępne	Aplikacja jest włączona i użytkownik znajduje się w menu głównym
Warunki końcowe	Aplikacja wyświetliła listę dostępnych poziomów
Scenariusz główny	1. Użytkownik wybiera opcję wyświetlenia wszystkich dostępnych poziomów 2. System wczytuje dane z bazy danych 3. System wyświetla listę wszystkich poziomów dostępnych w grze

Tabela 4.1. Specyfikacja przypadków użycia – przeglądanie dostępnych poziomów

Sekcja	Treść
Przypadek użycia	PU2 - Wybór poziomu
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Niski
Warunki wstępne	Użytkownik znajduje się w widoku wyboru poziomów dostępnych w grze
Warunki końcowe	Aplikacja wyświetliła szczegóły dotyczące wybranego poziomu
Scenariusz główny	1. Użytkownik wybiera jeden z dostępnych poziomów, za pomocą listy lub poprzez gest przesunięcia palcem 2. System wyszukuje informacje o wybranym poziomie 3. System wyświetla szczegółowe informacje dotyczące wybranego poziomu

Tabela 4.2. Specyfikacja przypadków użycia – wybór poziomu

Sekcja	Treść
Przypadek użycia	PU3 - Rozpoczęcie gry w normalnym trybie
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Wysoki Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik przegląda szczegóły dotyczące danego poziomu
Warunki końcowe	Aplikacja załadowała scenę wybranego poziomu i wyświetliła widok rozgrywki
Scenariusz główny	1. Użytkownik wybiera rozpoczęcie normalnego trybu rozgrywki na danym poziomie 2. System ładuje scenę odpowiadającą wybranemu poziomowi 3. System wyświetla widok rozgrywki w trybie normalnym dla wybranego przez użytkownika poziomu

Tabela 4.3. Specyfikacja przypadków użycia – rozpoczęcie gry w normalnym trybie

Sekcja	Treść
Przypadek użycia	PU4 - Rozpoczęcie gry w trybie praktyki
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Wysoki Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik przegląda szczegóły dotyczące danego poziomu
Warunki końcowe	Aplikacja załadowała scenę poziomu i wyświetliła widok rozgrywki
Scenariusz główny	1. Użytkownik wybiera rozpoczęcie rozgrywki w trybie praktyki na danym poziomie 2. System ładuje scenę odpowiadającą wybranemu poziomowi 3. System wyświetla widok rozgrywki w trybie normalnym dla wybranego przez użytkownika poziomu

Tabela 4.4. Specyfikacja przypadków użycia – rozpoczęcie gry w trybie praktyki

Sekcja	Treść
Przypadek użycia	PU5 - Przeglądanie zdobytych osiągnięć
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Niski
Warunki wstępne	Aplikacja jest włączona i użytkownik znajduje się w menu głównym
Warunki końcowe	Aplikacja wyświetliła listę osiągnięć dostępnych w grze, z wyszczególnieniem tych już zdobytych i podziałem na kategorie
Scenariusz główny	1. Użytkownik wybiera opcję wyświetlenia osiągnięć 2. System wczytuje dane z bazy danych 3. System wyświetla listę osiągnięć z podziałem na kategorie

Tabela 4.5. Specyfikacja przypadków użycia – przeglądanie zdobytych osiągnięć

Sekcja	Treść
Przypadek użycia	PU6 - Zdobywanie osiągnięć
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik spełnia warunek konieczny do otrzymania jednego z dostępnych osiągnięć, jeszcze nie odblokowanego
Warunki końcowe	Aplikacja przyznaje zdobyte osiągnięcie i wyświetla stosowną informację
Scenariusz główny	1. Użytkownik wykonując pewną akcję spełnia warunek konieczny do przyznania jednego z osiągnięć 2. System upewnia się, czy osiągnięcie nie zostało już odblokowane 3. System wyświetla informację o nowo zdobytym osiągnięciu

Tabela 4.6. Specyfikacja przypadków użycia – zdobywanie osiągnięć

Sekcja	Treść
Przypadek użycia	PU7 - Przeglądanie sklepu ze skórkami
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Wysoki Ryzyko technologiczne: Średni
Warunki wstępne	Aplikacja jest włączona i użytkownik znajduje się w menu głównym
Warunki końcowe	Aplikacja wyświetla widok sklepu
Scenariusz główny	1. Użytkownik wybiera opcję wyświetlenia widoku sklepu ze skórkami 2. System wczytuje dane z bazy danych 3. System wyświetla widok sklepu z listą pojazdów oraz odpowiedniki skórkami i kolorami

Tabela 4.7. Specyfikacja przypadków użycia – przeglądanie sklepu ze skórkami

Sekcja	Treść
Przypadek użycia	PU8 - Wybór skórki używanej dla pojazdu
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Wysoki Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik znajduje się w widoku sklepu i dokonuje zmiany grafiki lub koloru danego pojazdu
Warunki końcowe	Aplikacja wyświetla podgląd aktualnego wyglądu pojazdu i zapisuje wybrane zmiany w bazie danych
Scenariusz główny	1. Użytkownik przegląda listy skórek i kolorów dostępne dla danego pojazdu 2. Użytkownik dokonuje wyboru elementu, wpływającego na wygląd pojazdu 3. System aktualizuje podgląd pojazdu i zapisuje zmiany wprowadzone przez użytkownika

Tabela 4.8. Specyfikacja przypadków użycia – wybór skórki używanej dla pojazdu

Sekcja	Treść
Przypadek użycia	PU9 - Odblokowanie nowej skórki
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Wysoki Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik znajduje się w widoku sklepu i wybiera zablokowany element
Warunki końcowe	Aplikacja odblokowuje element i zapisuje zmiany w bazie danych
Scenariusz główny	1. Użytkownik przegląda listę skórek i kolorów 2. Użytkownik wybiera zablokowany element z listy 3. System wyświetla komunikat o zakupie przedmiotu 4. Użytkownik potwierdza kupno przedmiotu 5. System aktualizuje podgląd pojazdu i zapisuje dokonane zmiany
Scenariusz alternatywny	3.A System wyświetla komunikat o niewystarczającej liczbie punktów, potrzebnej do zakupu przedmiotu 3.A.1 Użytkownik potwierdza wiadomość 3.A.2 Wejście do punktu 1 scenariusza głównego 4.A Użytkownik anuluje kupno przedmiotu 4.A.1 Wejście do punktu 1 scenariusza głównego

Tabela 4.9. Specyfikacja przypadków użycia – odblokowanie nowej skórki

Sekcja	Treść
Przypadek użycia	PU10 - Wyświetlenie i konfiguracja ustawień gry
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Niski
Warunki wstępne	Aplikacja jest włączona i użytkownik znajduje się w menu głównym
Warunki końcowe	Aplikacja zapisuje ustawienia gry
Scenariusz główny	1. System wczytuje aktualne ustawienia i je wyświetla 2. Użytkownik przegląda ustawienia dostępne w grze 3. Użytkownik dokonuje zmian w ustawieniach 4. System zapisuje zmiany dokonane przez użytkownika

Tabela 4.10. Specyfikacja przypadków użycia – wyświetlenie i konfiguracja ustawień gry

Sekcja	Treść
Przypadek użycia	PU11 - Logowanie
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Średni
Warunki wstępne	Użytkownik znajduje się w widoku ustawień gry
Warunki końcowe	Użytkownik jest zalogowany w aplikacji
Scenariusz główny	1. Użytkownik wybiera opcję zalogowania za pomocą konta Google 2. System łączy się z zewnętrznym serwerem w celu weryfikacji użytkownika 3. System loguje użytkownika do aplikacji
Scenariusz wyjątku	Zdarzenie: System nie ma obecnie dostępu do internetu lub użytkownik nie posiada konta Google 1. System wyświetla komunikat o błędzie z zaprzestaje próby logowania 2. Użytkownik potwierdza otrzymany komunikat 3. Następuje powrót do widoku ustawień gry

Tabela 4.11. Specyfikacja przypadków użycia – logowanie

Sekcja	Treść
Przypadek użycia	PU12 - Przeglądanie rankingu graczy
Aktorzy	Gracz
Priorytet	Dla sukcesu projektu: Średni Ryzyko technologiczne: Średni
Warunki wstępne	Aplikacja jest włączona i użytkownik znajduje się w menu głównym
Warunki końcowe	Aplikacja wyświetla widok prezentujący ranking graczy
Scenariusz główny	1. Użytkownik wybiera opcję wyświetlenia widoku z rankingowego 2. System pobiera z serwera niezbędne informacje 3. System wyświetla ranking graczy
Scenariusz wyjątku	Zdarzenie: System nie ma obecnie dostępu do internetu lub użytkownik nie jest zalogowany 1. System nie pobiera danych i wyświetla komunikat o błędzie 2. Użytkownik potwierdza otrzymany komunikat 3. Następuje powrót do widoku menu głównego gry

Tabela 4.12. Specyfikacja przypadków użycia – przeglądanie rankingu graczy

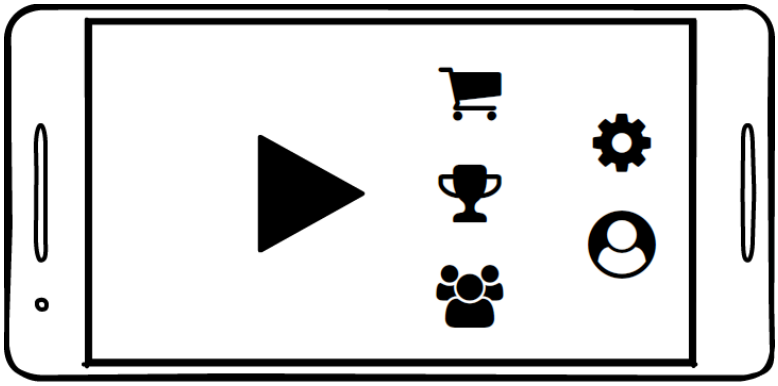
4.3 Wymagania niefunkcjonalne

Wymagania niefunkcjonalne opisują pożądane cechy, które powinien spełniać tworzony produkt. Dla aplikacji będącej podmiotem tej pracy są to:

- Gra powinna działać na urządzeniach mobilnych z systemem Android o minimalnej wersji 4.4 „KitKat" (API 19)
- Gra powinna być zoptymalizowana i działać płynnie również na słabszych urządzeniach
- Sterowanie powinno się odbywać za pomocą ekranu dotykowego
- Gra nie wymaga żadnych dodatkowych uprawnień systemu Android
- Sterowanie podczas rozgrywki powinno być łatwe i pozwolić użytkownikowi na szybkie zapoznanie się z nim
- Aplikacja powinna działać bez dostępu do internetu, oferując użytkownikowi normalną rozgrywkę
- Chwilowy zanik dostępu do internetu nie powinien wpływać na rozgrywkę
- Aplikacja musi zapisywać postęp gracza, zdobyte osiągnięcia, odblokowane przedmioty
- Po uruchomieniu aplikacji powinny załadować się wszystkie niezbędne zasoby, natomiast po wyłączeniu zwolnić
- Interfejs użytkownika powinien być prosty i intuicyjny oraz umożliwiać sprawną nawigację po aplikacji
- Wygląd oraz funkcjonalność interfejsu powinny być zbliżone na każdym urządzeniu, niezależnie od rozdzielczości i współczynnika kształtu ekranu
- Serwer aplikacji z którego pomocą zrealizowany zostanie system rankingowy powinien być dostępny 24 godziny na dobę

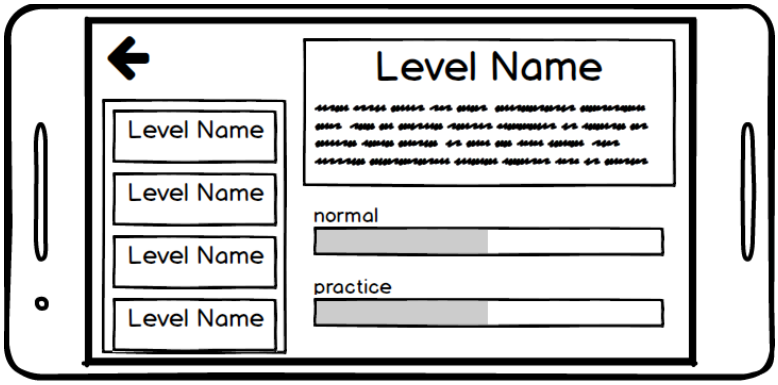
4.4 Projekt interfejsu

Prototypowanie interfejsu jest ważnym elementem wstępnego tworzenia aplikacji lub jeszcze projektowania. Wykorzystując odpowiednie do tego narzędzie, jak na przykład „Balsamiq Mockups" użyty w tym przypadku, można w prosty sposób zaprojektować wygląd poszczególnych ekranów w aplikacji i zawrzeć w nim część funkcjonalności, co ułatwi późniejszą ich implementację. Dużą zaletą jest również szybkość tego procesu, dzięki czemu często można wprowadzać poprawki, zadbać o jak największą przejrzystość i spójną szatę graficzną. Na kolejnych stronach znajdują się prototypy kolejnych widoków w grze autorskiej, wraz z krótkim ich opisem.



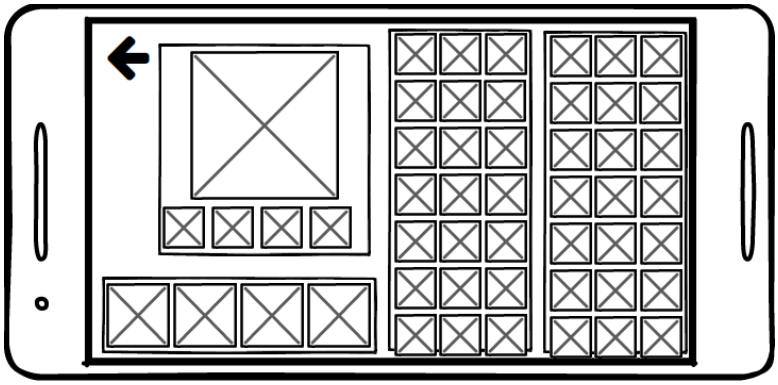
Rys. 4.2. Prototyp interfejsu – menu główne

Ekran startowy aplikacji jest prosty i dzięki temu dość intuicyjny. Po środku znajduje się przycisk rozpoczynający rozgrywkę, później najważniejsze funkcjonalności, czyli sklep, osiągnięcia i rankingi. Po prawej stronie są ustawienia oraz informacje o źródłach.



Rys. 4.3. Prototyp interfejsu – ekran wyboru poziomu

Widok zawiera listę wszystkich poziomów w lewej strony oraz szczegółowe informacje na temat obecnie wybranego. Można również je zmieniać za przesunięciem palca. W lewem górnym rogu znajduje się przycisk powrotu do menu.

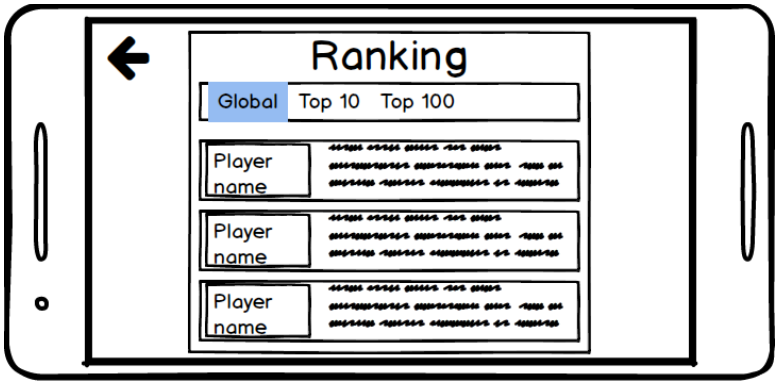


Rys. 4.4. Prototyp interfejsu – ekran sklepu

Po prawej stronie widoczne są 2 listy z dostępnymi do wyboru kolorami. Lista w lewym dolnym rogu służy do wyboru grafiki pojazdu. Centralnym elementem jest prezentacja aktualnego wyglądu pojazdu oraz przyciski zmieniające wybraną mechanikę, do której przypisane są inne pojazdy.

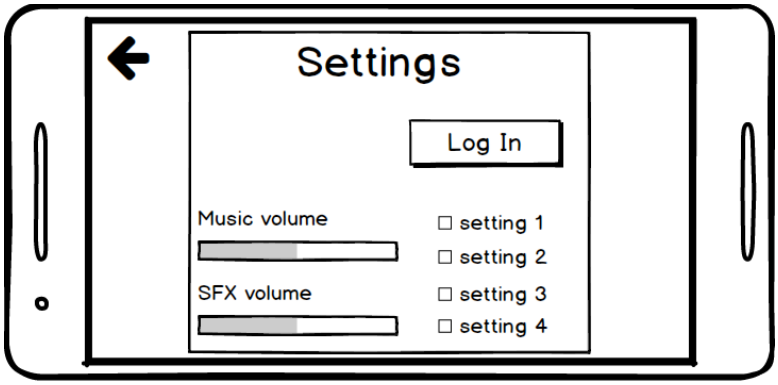
Rys. 4.5. Prototyp interfejsu – panel osiągnięć

Osiągnięcia przedstawione są w formie listy. Dla tych odblokowanych przewidziany jest wyróżniający się wygląd oraz data jego zdobycia. Zostały one podzielone na kategorie, znajdujące się w górnej części. Ułatwi to przeglądanie i znalezienie danego osiągnięcia.



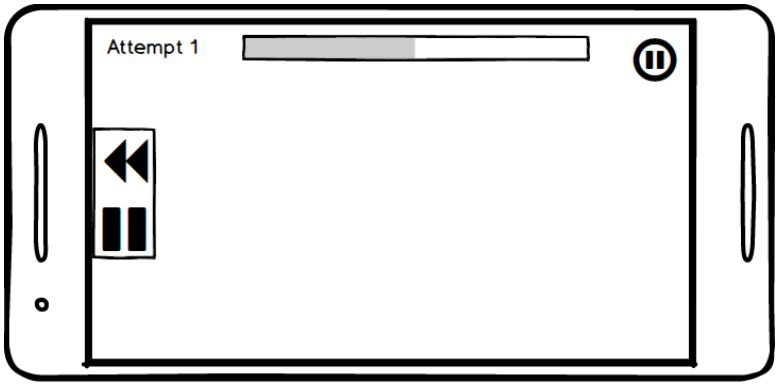
Rys. 4.6. Prototyp interfejsu – panel rankingowy

System rankingowy zrealizowany został z pomocą listy, która zawiera elementy odpowiadające graczom z informacją o ich nazwie oraz liczbą punktów. Dostępne są również zakładki, ogólny ranking z wyszczególnioną pozycją posiadacza aplikacji i najlepsi gracze.



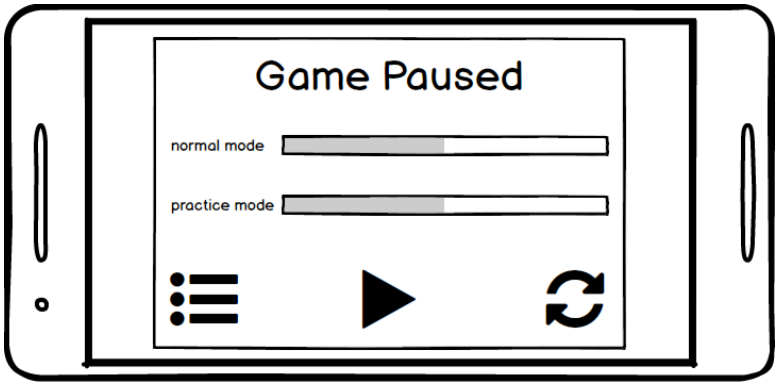
Rys. 4.7. Prototyp interfejsu – ekran ustawień gry

W tym panelu zawarte są elementy pozwalające na konfigurację aplikacji. Po prawej znajdują się suwaki odpowiadające za dźwięk w grze. U góry znalazł się przycisk logowania, a pod nim kolejne ustawienia odpowiadające na przykład za wyświetlania paska postępu podczas rozgrywki lub automatyczny restart poziomu w razie przegranej.



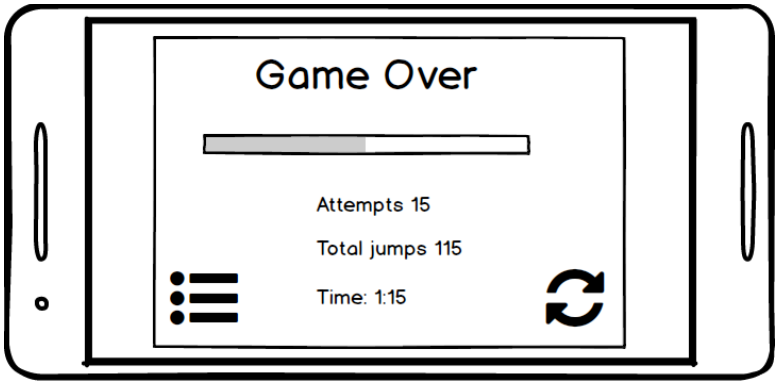
Rys. 4.8. Prototyp interfejsu – widok podczas rozgrywki

Ekran składa się z niewielkiej liczby elementów, ponieważ nie mogą one utrudniać rozgrywki. Po lewej stronie jest panel widoczny tylko w trybie praktyki, służący do zatrzymywania i cofania czasu. U góry można znaleźć liczbę powtórzeń poziomu i pasek postępu w grze oraz przycisk po prawej stronie, pozwalający zatrzymać rozgrywkę.



Rys. 4.9. Prototyp interfejsu – panel zapauzowanej gry

Widok zatrzymanej rozgrywki zawiera przyciski znajdujące się w dolnej części, pozwalające na powrót do wyboru poziomów, wznowienie gry i rozpoczęcie od nowa. W centrum widoczne są paski informujące o najwyższym wyniku na danym poziomie.



Rys. 4.10. Prototyp interfejsu – panel przegranej gry

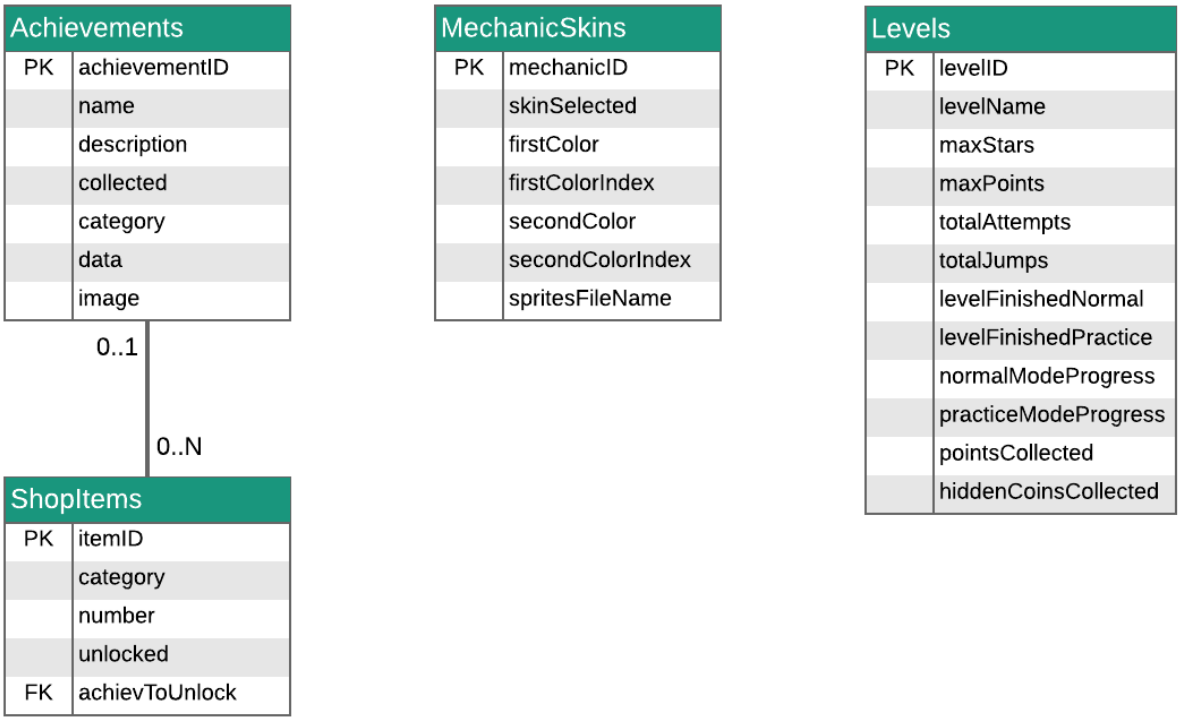
W centralnej części znajdują się informacje o poziomie w chwili przegrania, takie jak postęp, czas i liczba prób. Przyciski w rogach pozwalają na powrót do wyboru poziomów oraz rozpoczęcie gry od nowa.

Rys. 4.11. Prototyp interfejsu – panel wygranej gry

Na środku znajdują się gwiazdki, które symbolizują ukryte obiekty, możliwe do zdobycia na każdym poziomie. Niżej widnieje czas ukończenia poziomu, liczbie powtórzeń oraz wykonanych akcji, nazwanych dla ułatwienia liczbą skoków. Dostępne są również przyciski pozwalające na powrót do wyboru poziomów oraz ponowne uruchomienie bieżącego.

4.5 Struktura przechowywania danych

Baza danych jest obecnie ważnym elementem większości aplikacji. Pozwala w łatwy i zorganizowany sposób przechowywać wiele informacji, do których jest szybki dostęp, a zapis i ich edycja umożliwia zapamiętywanie zmian wprowadzanych przez użytkowników. W przypadku tego projektu została utworzona lokalna struktura, której zadaniem jest przechowywanie postępu w grze, osiągnięć i zdobytych przedmiotów. Jej schemat wraz z opisem znajduje się poniżej.



Rys. 4.12. Schemat struktury przechowywania danych

Powstanie ona z wykorzystaniem SQLite, który jest systemem przeznaczonym do zarządzania relacyjnymi bazami danych. Ze względu na specyfikę projektu, tworzona w ramach niego gra wymaga jedynie bardzo prostej, lokalnej struktury, przeznaczonej tylko dla jednego użytkownika. W swoich tabelach przechowywać będzie niewielką i niezmienną liczbę wierszy, a zmianie podlegać będą jedynie wartości atrybutów. Dodatkowo, biorąc po uwagę to, że nie każda tabela przedstawiona na schemacie posiada zależności łączące je z innymi tabelami, rozwiązania tego nie można wprost nazwać relacyjną bazą danych, a jedynie pewną formą przechowywania informacji i zapisu postępów gracza.

Takie rozwiązanie zostało wybrane głównie ze względu na łatwą integrację SQLite z silnikiem Unity i możliwość jej prostego wykorzystania na urządzeniach mobilnych, dzięki zawartości kompletnej bazy w jednym pliku. Zrezygnowano natomiast z pomysłu przechowywania danych wykorzystując pliki JSON, co byłoby bardziej naturalne, ze względu na prostą strukturę danych i brak relacji między nimi. Głównym ku temu powodem było to, że JSON nie pozwala na edycję danych, przez co każdorazowa modyfikacja wymaga ponownego zapisania całego pliku, co jest bardzo nieefektywne, a dane w grze ulegają zmianie dość często. SQLite natomiast oferuje bardzo szybki dostęp i modyfikację danych, która jest wymagana przy konieczności zachowania płynności rozgrywki.

Achievements	
Nazwa atrybutu	Opis
achievementID	Identyfikator osiągnięcia
name	Nazwa osiągnięcia
description	Krótki opis warunków wymaganych do odblokowania osiągnięcia
collected	Wartość określająca, czy osiągnięcie zostało zdobyte
category	Kategoria, do której przynależy osiągnięcie
data	Data zdobycia osiągnięcia
image	Nazwa ikony przypisanej do osiągnięcia, umożliwia wczytanie jej z plików gry

Tabela 4.13. Schemat bazy danych – tabela „Achievements”

ShopItems	
Nazwa atrybutu	Opis
itemID	Identyfikator przedmiotu
category	Kategoria, do której należy dany przedmiot
number	Numer określający przedmiot i identyfikujący go w danej kategorii, wyznaczający jego pozycję na liście w sklepie gry
unlocked	Wartość określająca, czy przedmiot został odblokowany
achievToUnlock	Identyfikator osiągnięcia, które należy zdobyć, aby odblokować dany przedmiot

Tabela 4.14. Schemat bazy danych – tabela „ShopItems”

MechanicSkins	
Nazwa atrybutu	Opis
mechanicID	Identyfikator mechaniki sterowania
skinSelected	Wartość określająca wybrany przez użytkownika wygląd pojazdu, związanego z daną mechaniką
firstColor	Pierwszy kolor wybrany przez użytkownika, reprezentowany poprzez ciąg znaków, będący heksadecymalnym zapisem koloru RGB
firstColorIndex	Indeks pierwszego koloru, określający jego pozycję na liście w sklepie gry
secondColor	Drugi kolor wybrany przez użytkownika, reprezentowany poprzez ciąg znaków, będący heksadecymalnym zapisem koloru RGB
secondColorIndex	Indeks drugiego koloru, określający jego pozycję na liście w sklepie gry
spritesFileName	Nazwa pliku zawierającego grafiki dla pojazdu, identyfikującego się z daną mechaniką, umożliwia ich wczytanie z plików gry

Tabela 4.15. Schemat bazy danych – tabela „MechanicSkins”

Levels	
Nazwa atrybutu	Opis
levelID	Identyfikator poziomu
levelName	Nazwa poziomu
maxStars	Liczba gwiazdek za ukończenie poziomu
maxPoints	Liczba do zdobycia za ukończenie całego poziomu
totalAttempts	Całkowita liczba prób przejścia poziomu
totalJumps	Całkowita liczba podskoków, akcji użytkownika
levelFinishedNormal	Wartość określająca, czy poziom ukończono w normalnym trybie
levelFinishedPractice	Wartość określająca, czy poziom ukończono w trybie praktyki
normalModeProgress	Postęp osiągnięty w normalnym trybie rozgrywki, w procentach
practiceModeProgress	Postęp osiągnięty w trybie praktyki, w procentach
pointsCollected	Liczba przyznanych punktów, zależna od aktualnego postępu na danym poziomie
hiddenCoinsCollected	Liczba odnalezionych monet, ukrytych na poziomie

Tabela 4.16. Schemat bazy danych – tabela „Levels”

5. Implementacja

Ten rozdział skupi się na kwestiach związanych z procesem implementacji gry będącej podmiotem tej pracy. Przedstawione zostaną obiekty zarządzające głównymi funkcjonalnościami aplikacji, sposób ich działania oraz znaczenie. Następnie zaprezentowane będą elementy rozgrywki, obiekty specjalne i ich realizacja. Przedstawiony zostanie również końcowy interfejs użytkownika wraz ze sposobem przejścia między kolejnymi widokami.

5.1 Singletony

Singleton jest wzorcem projektowym, którego zadanie polega na ograniczeniu tworzenia obiektów należących do danej klasy tylko do jednej instancji oraz zagwarantowaniu globalnego dostępu do tak utworzonego obiektu.

Singletony odgrywają znaczącą rolę w działaniu aplikacji autorskiej. Obiekty utworzone w tej sposób sprawują funkcję menadżerów i są odpowiedzialne za podstawowe funkcjonalności aplikacji, między innymi takie jak łączność z bazą danych, zarządzanie rozgrywką czy osiągnięciami. W silniku Unity realizacja singletonu możliwa jest na wiele sposobów, lecz najprostsza i najbardziej popularna polega na dodaniu do klasy kodu znajdującego się poniżej.

```
19      private void Awake()
20      {
21          if (Instance == null)
22          {
23              Instance = this;
24              DontDestroyOnLoad(this.gameObject);
25          }
26          else Destroy(this);
27      }
```

Rys. 5.1. Fragment kodu realizujący wzorzec singletonu

Funkcja „Awake” jest metodą specyficzną dla Unity, wywoływaną tuż przed startem gry, kiedy instancja klasy jest tworzona. Wykorzystuje się ją głównie do inicjalizacji zmiennych, ale można jej użyć również dla singletonu. Podany fragment kodu sprawdza czy istnieje już obiekt danej klasy. Jeżeli nie to do statycznego pola „Instance”, przypisywany jest bieżący obiekt, dla którego wywołana została metoda „Awake”. W przeciwnym wypadku nadmiarowa instancja klasy jest natychmiast usuwana. Funkcja „DontDestroyOnLoad” sprawia, że obiekt objęty jej działaniem nie zostanie zniszczony podczas załadowania nowej sceny. Jest to ważne w celu zachowania spójnych informacji podczas użytkowania aplikacji, gdzie każdy poziom, menu główne i sklep stanowi osobną scenę, a przejścia między nimi skutkują zniszczeniem wszystkich znajdujących się na niej obiektów.

W aplikacji znajduje się 9 singletonów, z których każdy posiada własne zasadnicze zadanie i sprawuje kontrolę nad pewną funkcjonalnością. Rysunek poniżej przedstawia ich budowę. Część z nich, te oznaczone na zielono są obiektami tworzonymi jednorazowo podczas cyklu życia całej aplikacji. Pozostałe, czyli niebieskie, to kontrolery biorące udział podczas rozgrywki, pełnią swoje funkcje wyłącznie lokalnie, w obrębie danej sceny.

Rys. 5.2. Klasy będące singletonami

LevelManager

Instancja tej klasy przechowuje informacje dotyczące wszystkich poziomów dostępnych w grze. Przy starcie aplikacji niezbędne dane pobierane są z lokalnej bazy danych i następnie przechowywane w słowniku, co ułatwia dostęp do nich. Odpowiada również za zapis do bazy wszelkich wprowadzonych zmian, jeżeli zajdzie taka potrzeba.

AchievementManager

Menadżer, który zarządza osiągnięciami dostępnymi w grze. Przy starcie aplikacji tworzy listę wszystkich osiągnięć, które pobierane są z bazy danych. Posiada metody pozwalające na sprawdzenie czy osiągnięcie zostało już odblokowane, popraniu informacji o nim, a także danych o wszystkich osiągnięciach z konkretnej kategorii. Do jego zadań należy również kontrola i przyznawanie osiągnięć związanych ze zmiennymi wartościami takimi jak całkowita ilość zdobytych punktów, ukrytych monet lub cofniętego czasu. Ponad to zajmuje się powiadamianiem użytkownika o nowo zdobytym osiągnięciu poprzez ukazanie panelu z odpowiednim komunikatem.

GooglePlayManager

Jest obiektem, który bezpośrednio wykorzystuje wtyczkę Play Game Services. Odpowiada za połączenie aplikacji z kontem Google, a także tak zwane ciche logowanie, które dokonuje automatycznej autoryzacji przy uruchomieniu gry, po wcześniejszym, udanym połączeniu. Z jego pomocą został zrealizowany system rankingowy, ponieważ zawiera metody pozwalające na pobranie wymaganych informacji z serwera gry.

SceneLoader

Ta klasa posiada metody, wykorzystywane przez inne obiekty aplikacji w celu załadowania nowej sceny lub ponownym uruchomieniu bieżącej. Odpowiada również za płynne przejście pomiędzy nimi oraz wyświetlanie animacji, polegającej na przyciemnieniu i rozjaśnieniu ekranu przy zmianie widoku w aplikacji.

SQLDataBase

Jest jednym z ważniejszych elementów całej aplikacji, a swoje działanie zawdzięcza wtyczce do SQLite dla Unity. Przy starcie nawiązuje połączenie z lokalną bazą danych oraz rozłącza je, wykorzystując w tym celu specjalną funkcję „OnApplicationQuit" dostarczaną przez silnik Unity, która jest wywoływana przy wyłączaniu aplikacji. Ponad to posiada szereg metod pozwalających na wczytanie informacji z bazy danych, z których korzysta wiele innych obiektów, takich jak kontroler osiągnięć i poziomów. Z jego pomocą można również dane zaktualizować i zapisać.

GameSettings

Instancja tej klasy zarządza ustawieniami dostępnymi w grze, które wczytywane są podczas włączenia aplikacji. Informacje te wykorzystywane są przez inne obiekty, na przykład kontroler poziomu używa ustawień związanych z rozgrywką, a sklep danych o zapisanym wyglądzie postaci. Dostępne są również metody pozwalające na ich zmianę i zapis przez użytkownika, możliwe w specjalnym widoku z dostępnymi ustawieniami gry.

PlayerStatistics

W tym obiekcie przechowywane są podstawowe statystyki gracza, będące ogólnym postępem w grze, który stanowi podstawę przy wyliczaniu punktów w rankingu. Przy starcie aplikacji są one wczytywane, a następnie globalnie dostępne dla innych obiektów, umożliwiając również ich zapis.

LevelController

Kontroler poziomu jest najważniejszym elementem w funkcjonowaniu rozgrywki. Obsługuje wydarzenia związane z zatrzymaniem i wznowieniem gry oraz jej zakończenia w przypadku przegranej lub ukończenia poziomu. Obsługuje i zarządza obiektami znajdującymi się na scenie, jest odpowiedzialny za kontrolę postępu w grze i poprawny jego zapis, a także wywoływanie procesów sprawdzających warunki potrzebne do przyznania osiągnięć. W przeciwieństwie do poprzednich przypadków, obiekt ten jest tworzony przy każdym włączeniu poziomu, a następnie usuwany pod koniec, więc sprawuje kontrolę w obrębie tylko jednej sceny, a nie podczas całego cyklu życia aplikacji.

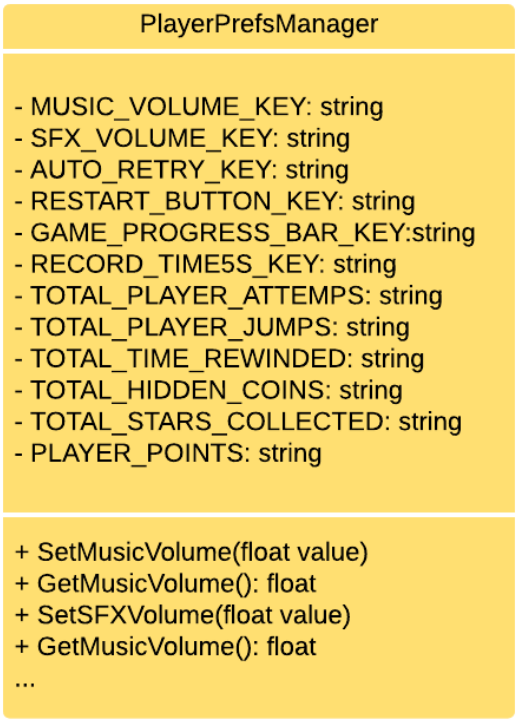
TimeController

Kontroler czasu również istnieje wyłącznie dla danego poziomu, lecz tylko w przypadku, gdy jest on uruchamiany w trybie praktyki. Jego głównym zadaniem jest ciągły zapis aktualnego stanu pojazdu oraz umożliwienie użytkownikowi cofania czasu i jego zatrzymanie.

PlayerPrefsManager

Menadżer preferencji użytkownika jest wyjątkową klasą, ponieważ w przeciwieństwie do wszystkich wyżej wymienionych, przy pełnieniu swoich funkcjonalności nie wymaga tworzenia ani jednej instancji. Jego pola są stałymi wartościami, a metody statyczne i publicznie dostępne. Jest to specyficzny dla Unity sposób przechowywania danych z wykorzystaniem dostarczonych przez nie bibliotek. Pozwala na zapis, odczyt i modyfikacje pojedynczych zmiennych, wykorzystując do tego celu specjalne klucze służące do ich identyfikacji, co w działaniu przypomina trochę strukturę słownika. Z jego pomocą zapamiętywane są informacje dotyczące statystyk użytkownika i ustawień gry, których zapis w bazie danych byłby nie praktyczny. Klucze są przechowywane w postaci stałych, statycznych pól, co znacznie ułatwia zarządzanie nimi. Klasa posiada szereg metod, dostępnych bezpośrednio dla każdego obiektu, co pozwala na łatwy odczyt i zapis niezbędnych danych.

Poniżej przedstawiony został schemat tej klasy. Nie wszystkie metody zostały na nim uwzględnione ze względu na ich dużą ilość i powtarzalność, dla każdego pola występuje metoda odczytująca i modyfikująca wartość z nim powiązaną.



Rys. 5.3. Klasa „PlayerPrefsManager”

5.2 Elementy rozgrywki

Gra zaprojektowana została jako dwuwymiarowa gra platformowa, wyświetlana horyzontalnie, bez możliwości obrócenia ekranu. Jej głównym celem jest omijanie przeszkód i zajście jak najdalej, aby ostatecznie ukończyć poziom. Na rozgrywkę składa się wiele elementów, takich jak pojazdy, obiekty specjalne i portale, które mogą działać niezależnie, natomiast po umieszczeniu na zaprojektowanym poziomie tworzą kompletną całość.

Tryby rozgrywki

Podczas normalnego trybu rozgrywki, gracz sprawuje kontrolę nad jednym z pojazdów, różniących się mechaniką sterowania. Kontrola nad nimi jest bardzo prosta i polega jedynie na dotknięciu lub przytrzymaniu ekranu w odpowiednim momencie, natomiast pojazd sam porusza się do przodu. Dzięki płynnemu przejściu między nimi w trakcie gry i zastosowaniu obiektów specjalnych, znajdujących się na poziomach, rozgrywka została znacznie urozmaicona pomimo prostoty sterowana. Z czasem stanowi coraz większe wyzwanie, zwłaszcza przy często zmieniających się mechanikach sterowania oraz wzrastającej liczbie przeszkód i obiektów. W związku z tym potrzebne okazuje się liczne powtarzanie poziomów, w celu poprawienia zręczności i zdobyciu umiejętności pozwalających zajść dalej.

Tryb praktyki został stworzony, aby nieco ułatwić graczom ten proces. Zbyt wygórowany poziom trudności i konieczność rozpoczynania poziomu od początku, kiedy wyzwanie stanowi pewien końcowy jego etap, mógłby skutkować zniechęceniem wśród użytkowników, a nawet porzuceniem dalszej gry.

Podczas tego trybu, rozgrywka przebiega zupełnie standardowo, a poziom swoją budową niczym nie różni się od normalnego trybu. Istotną zmianę natomiast stanowi możliwość kontroli czasu, czyli jego zatrzymywania i cofania, dzięki czemu przegrana nie jest ostateczna, a użytkownik ma możliwość dowolnego powtarzania etapu sprawiającego mu trudność. Dostęp do tych funkcjonalności zapewnia dodatkowy panel, zawierający dwa przyciski, które wciśnięte podlegają specjalnej animacji informującej o aktualnym stanie. Znajduje się on na obrazku poniżej.

Rys. 5.4. Panel kontroli czasu.

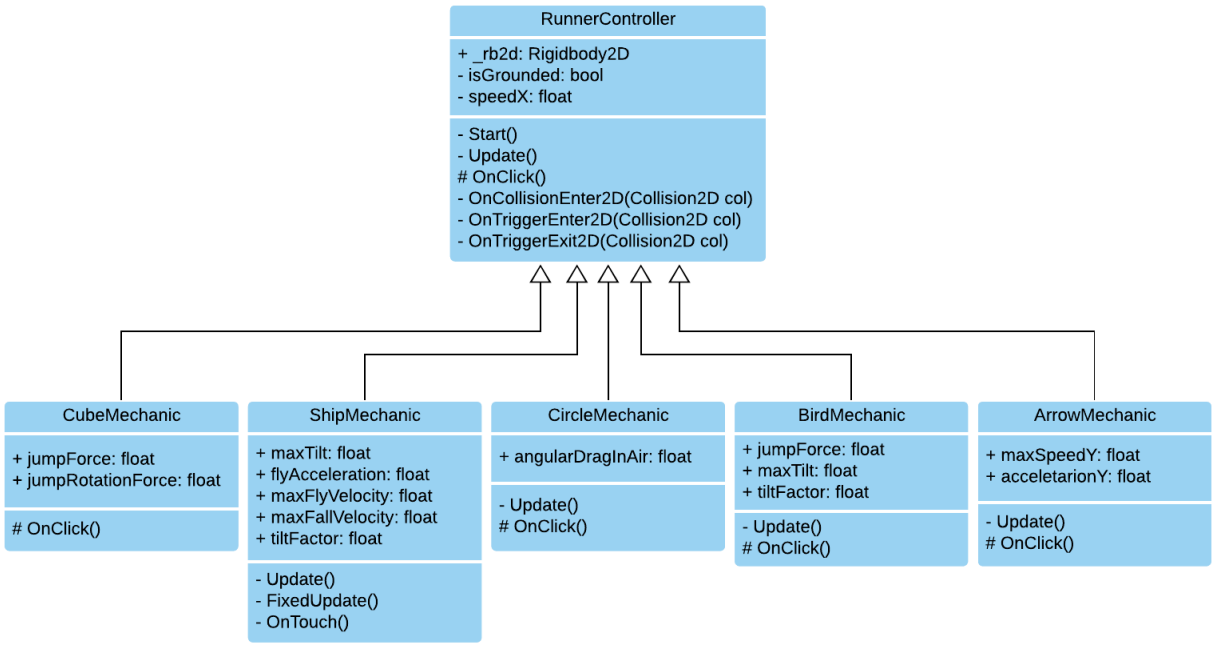
W celu umożliwienia użytkownikowi cofania czasu, należy najpierw go nagrywać. Wymaga to zapamiętania poprzednich wartości dotyczących pojazdu, czyli pozycji, rotacji, prędkości liniowej i kątowej, przy każdej ich aktualizacji. Informacje te zapisywane są za pomocą stosu, który dobrze sprawdza się w tym przypadku, ponieważ najwcześniejsze dane przechowuje na szczycie, a cofanie czasu polega na zdejmowaniu z niego kolejnych wartości, dzięki funkcji „Pop”. Zwracany element zawiera poprzedni stan pojazdu, do którego następuje powrót. Procedura powtarza się co każdą kolejną klatkę, zapewniając płynny efekt. Z racji tego, że cały proces wymaga zapamiętania dużej liczby zmiennych oraz ograniczonych możliwości urządzeń mobilnych, przewijanie czasu dotyczy wyłącznie pojazdu sterowanego przez gracza, elementy dekoracyjne tła, które również się poruszają, są pomijane.

Zatrzymywanie czasu jest możliwe dzięki wbudowanemu w silnik Unity systemu, który pozwala na zmianę skali czasu. W przypadku ustawienia jej na ‘0’, zatrzymane zostają aktualizacje wszystkich obiektów znajdujących się na scenie. Po ponownym ustawieniu wartości na ‘1’, rozgrywka jest wznawiana i obiekty znowu zaczynają się poruszać.

Mechaniki sterowania

W grze dostępnych jest kilka pojazdów różniących się wyglądem, ale przede wszystkim sposobem poruszania się i sprawowanej nad nimi kontroli przez gracza. Mają one wiele cech wspólnych, dlatego naturalnym rozwiązaniem w ich implementacji było zastosowanie dziedziczenia. Schemat tej relacji oraz jej opis znajduje się poniżej.

Wszystkie nazwy zaprezentowanych mechanik są wyłącznie poglądowe i nadane w celu łatwego ich rozróżnienia z pomocą skojarzeń.

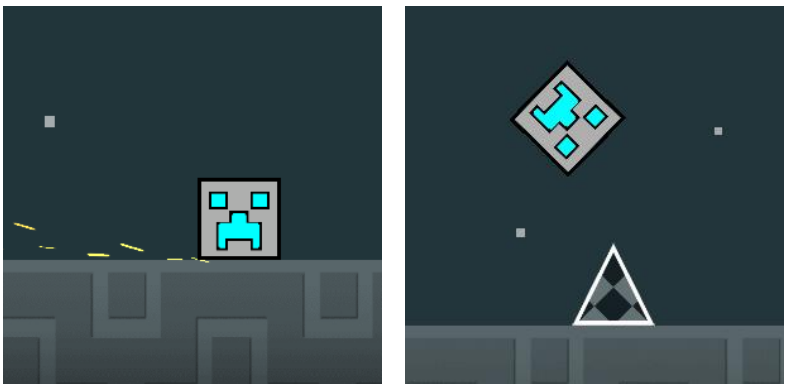


Rys. 5.5. Schemat przedstawiający wykorzystanie dziedziczenia w implementacji pojazdów

RunnerController

Klasa bazowa, która zawiera wartości i metody będące podstawą funkcjonowania każdej mechaniki. Pole typu Rigidbody2D jest odniesieniem do komponentu dostarczanego przez silnik Unity, który odpowiada za nadanie obiektowi cech fizycznych, takich jak grawitacja, prędkość i przyspieszenie oraz pozwala na ich modyfikację w kodzie. Tutaj jest również zdefiniowana wartość prędkości w osi poziomej, która odpowiada za niezmienny i automatyczny ruch pojazdów. Wirtualna funkcja OnClick umożliwia po jej przeciążeniu w klasach pochodnych, obsługę dotknięcia ekranu przez użytkownika i odpowiednią reakcję na nie zdefiniowaną przez poszczególne pojazdy. Klasa posiada także metody obsługujące kolizję ze światem gry i wyznacza przegraną w przypadku zatrzymania pojazdu lub kontaktu z kolcami. Sprawdza również, kiedy pojazd znajduje się na podłożu, a kiedy w powietrzu, co umożliwia zablokowanie skoku w drugim przypadku.

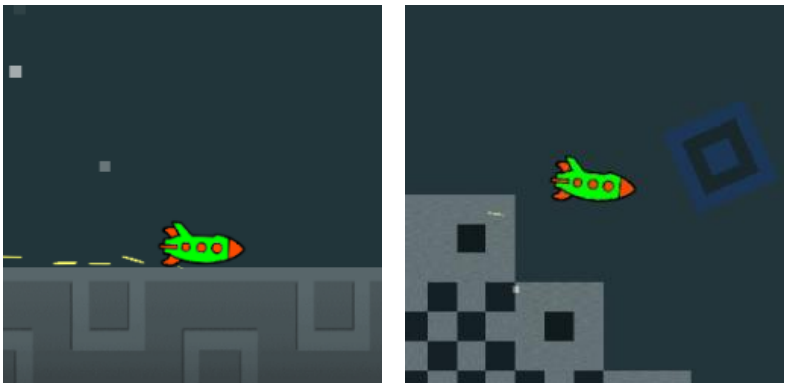
CubeMechanic



Rys. 5.6. Mechanika kostki

Mechanika kostki jest podstawowym pojazdem i pierwszym z którym gracz się zapoznaje. Sterowanie nią jest bardzo proste i polega na skoku przy dotknięciu ekranu urządzenia. W jego trakcie następuje również jej obrót. Skok możliwy jest wyłącznie wówczas, gdy pojazd znajduje się na podłożu. Siła skoku oraz obrotu zdefiniowana jest w odpowiednich polach i można nią dowolnie manipulować, co pozwoliło na ich precyzyjne dobranie i dostosowanie do gry.

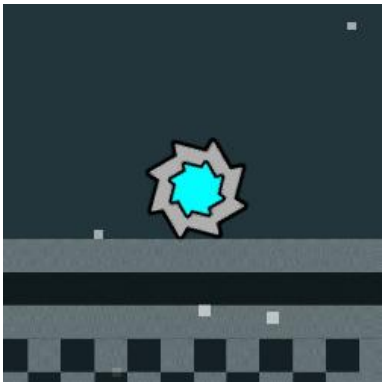
ShipMechanic



Rys. 5.7. Mechanika statku

Mechanika statku kosmicznego jest nieco inna od pozostałych, ponieważ swoje działanie opiera na utrzymaniu dotyku ekranu, a nie wyłącznie na naciśnięciu, dlatego nie przeciąża metody OnClick, lecz definiuje własną OnTouch. Podczas tego zdarzenia do pojazdu przykładana jest siła, która nadaje mu przyspieszenie w górnym kierunku. W przeciwnym wypadku, działająca grawitacja powoduje jego spadek. Przód pojazdu podąża nieco w kierunku ruchu, co powoduje uczucie płynnego sterowania.

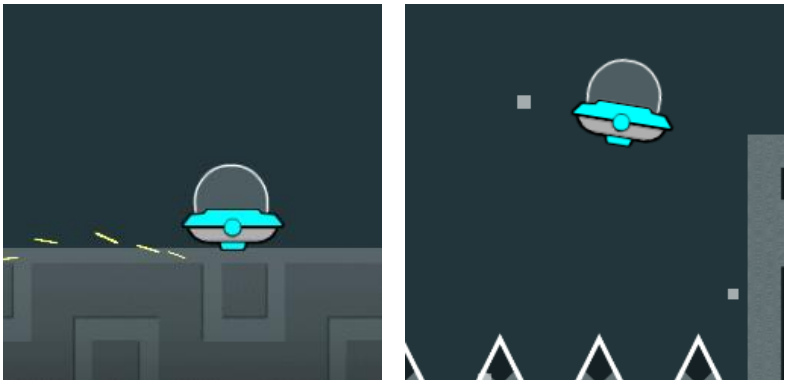
CircleMechanic



Rys. 5.8. Mechanika koła

Mechanika koła polega na odwróceniu grawitacji. Pojazd toczy się po powierzchni, a w momencie dotknięcia ekranu następuje zmiana siły na niego działającej, co powoduje ruch w osi pionowej w kierunku przeciwnym. Podczas gry pojazd znajduje się w powietrzu, zmiana grawitacji nie jest już możliwa, natomiast obraca się on nadal, powoli wytracając prędkość, aż znowu dotknie podłoża.

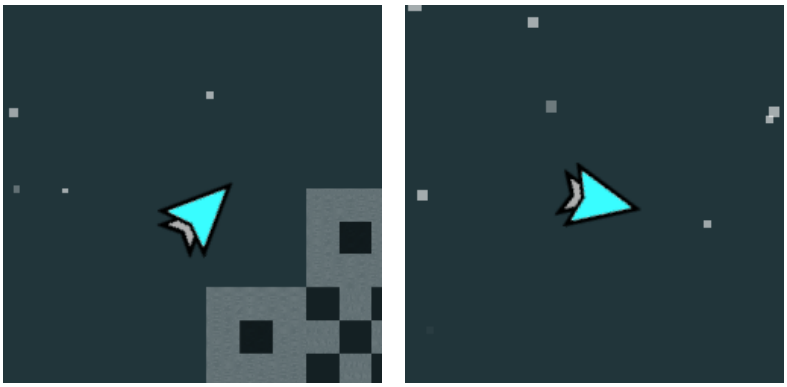
BirdMechanic



Rys. 5.9. Mechanika ptaka

Mechanika ptaka przypomina sposób sterownia w popularnej grze „FlappyBird”. Przy każdym kliknięciu, do pojazdu dodawana jest pewna siła w górnym kierunku, która powoduje jego wznoszenie. Nie jest ono tak płynne jak w przypadku statku, lecz gwałtowne, mniej precyzyjne i przez to trudniejsze, co wymaga zupełnie innego sposobu w pokonywaniu przeszkód. Przy każdym wzlocie przód pojazdu wznosi się do góry, a następnie powoli obniża wraz z pojazdem, co zapewnia efekt płynności.

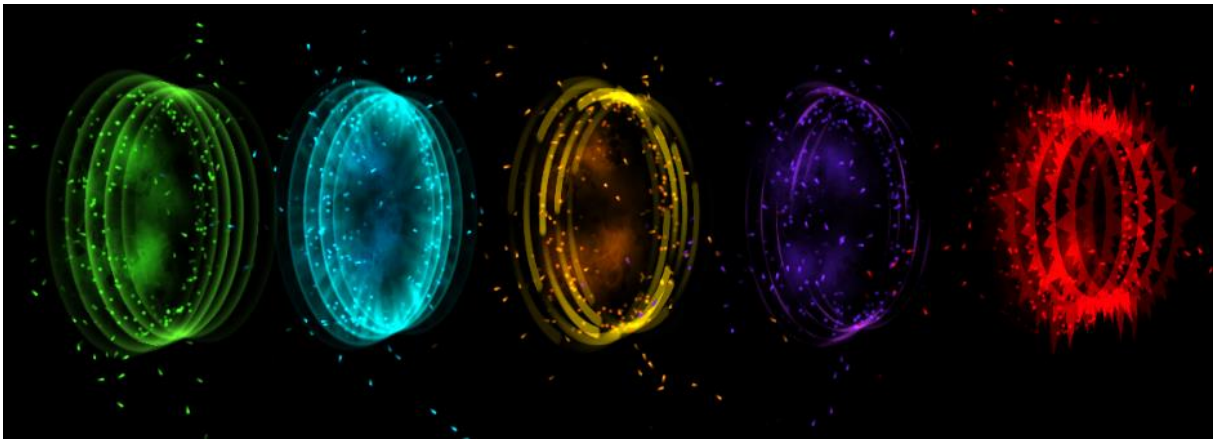
ArrowMechanic



Rys. 5.10. Mechanika strzały

Mechanika strzały inaczej niż pozostałe nie porusza się w prawo, lecz po ukosie pod kątem 45 stopni. Gracz ma możliwość zmiany jedynie kierunku poruszania, w górę lub w dół, przy zachowaniu odpowiedniego kąta, w dowolnym momencie, nie tylko gdy dotyka podłoża. Zmiana ta nie zachodzi jednak natychmiastowo, lecz jest płynnym przejściem pomiędzy dwoma kierunkami.

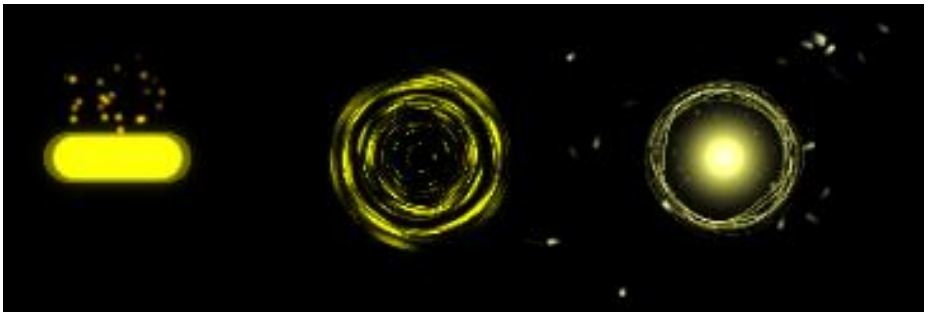
Portale



Rys. 5.11. Grafika przedstawiająca portale w grze

Portale są obiektami służącymi do zmiany mechaniki sterowania, kiedy pojazd znajdzie się w ich obszarze. Ich działanie opiera się na usuwaniu obiektu, który w nie wlatuje, zapamiętywaniu jego pozycji, prędkości i rotacji, a następnie utworzeniu nowego obiektu, będącego określonym pojazdem i nadaniu mu zapisanych cech poprzedniego. W ten sposób możliwe jest płynne przejście pomiędzy nimi w trakcie trwania rozgrywki. Aby umożliwić rozpoznanie docelowego pojazdu, w który nastąpi przemiana, portale posiadają różne kształty oraz kolory. Kolejno od lewej strony są to, pojazd kostki, statku, koła, ptaka oraz strzały.

Obiekty specjalne



Rys. 5.12. Obiekty specjalne w grze

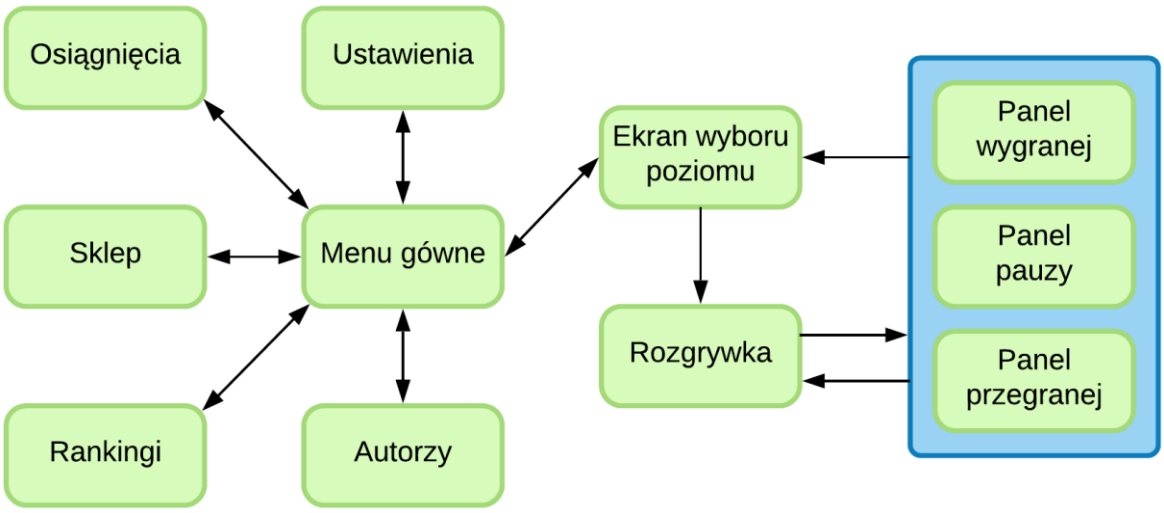
W grze występują specjalne obiekty, które odpowiednio rozmieszczone na poziomie stanowią element dodatkowy rozgrywki. Pierwszy z nich pełni funkcję wyrzutni i kiedy pojazd znajdzie się w jego zasięgu zostaje niezależnie wyrzucony do góry. Umożliwia to pokonywanie większych dystansów niż w przypadku zwykłego skoku, a także pewnego rodzaju pułapek, gdy umieści się nad nim kolec. Kolejnym jest lśniąca sfera, która wpływa na pojazd tak jakby miał styczność z podłożem, co umożliwia wykonanie dostępnych wówczas akcji. Dzięki temu pojazd kostki może dokonać skoku w powietrzu, a mechanika kółka ponowną zmianę grawitacji. Ostatni z nich jest elementem kolekcjonerskim. Na każdym poziomie ukryte są 3 takie obiekty, a zadaniem gracza jako cel dodatkowy, jest znalezienie i zdobycie ich wszystkich, co ze względu na ich umiejscowienie nie jest prostym zadaniem, lecz stanowi wyzwanie.

5.3 Interfejs użytkownika

Interfejs w całości został zaprojektowany przy użyciu narzędzia wbudowanego w silnik Unity. Część ekranów została zrealizowana za pomocą osobnych scen, są to menu główne, sklep, ekran wyboru poziomu, rozgrywka. Pozostałe widoki są wysuwającymi się panelami, które nakładają się na widok podstawowy odpowiedniej ze scen. Zdecydowano się na takie rozwiązanie, ponieważ część z nich jest mocno rozbudowana lub stanowi osobny element, jak w przypadku rozgrywki. Pozostałe natomiast są dość proste i dlatego zamiast osobnych scen, znajdują się na wysuwanych panelach.

Zgodnie z założeniami we wstępie, interfejs udało się zaprojektować tak, aby był poprawnie wyświetlany na urządzeniach z różnymi rozdzielczościami i aspektami ekranu. Było to możliwe dzięki odpowiedniej konfiguracji skalowania ekranów i rozmieszczeniu poszczególnych elementów z zachowaniem położenia względem siebie, na etapie tworzenia widoków w Unity. Część użytych grafik została wykonana samodzielnie, natomiast większość z nich pochodzi z gotowych rozwiązań [23], [24].

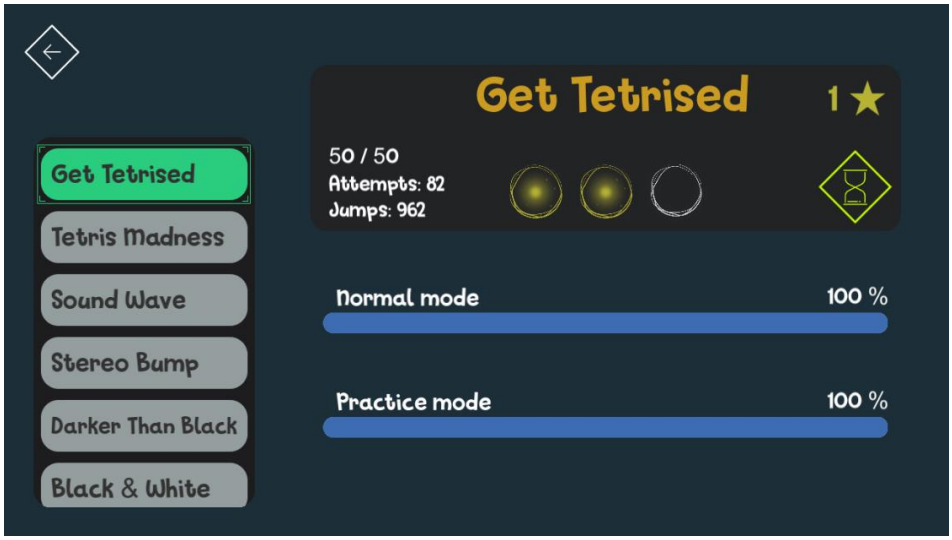
Poniżej zamieszczony został rysunek, przedstawiający przejścia pomiędzy poszczególnymi widokami. Podstawowym ekranem jest menu główne, które pojawia się zaraz po uruchomieniu aplikacji. Z tego miejsca użytkownik ma dostęp do wszystkich pozostałych funkcjonalności. Przejście pomiędzy scenami jest szybie, z obecnym efektem przyciemnienia i rozjaśnienia ekranu, co zapewnia wrażenie płynności. W przypadku widoków zawartych na panelach, towarzyszy im animacja jego wysuwania. Podczas rozgrywki uwidocznić może się jeden z paneli znajdujących się w niebieskim bloku, przy czym ten odpowiadający wygranej i przegranej pokazuje się w przypadku zajścia jednego z tych zdarzeń. Z nich natomiast można powrócić do rozgrywki lub ekranu wyboru poziomu.



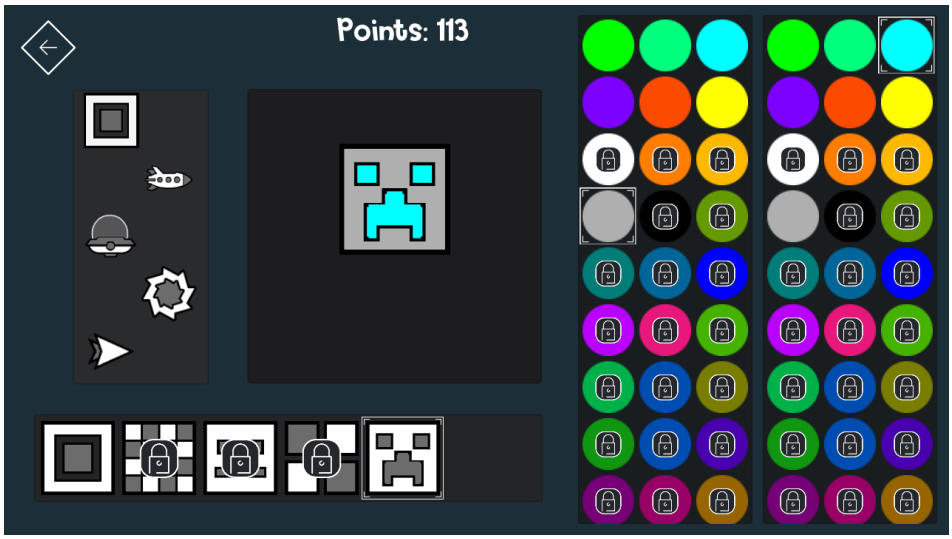
Rys. 5.13. Interfejs użytkownika – schemat przejść między ekranami

Na następnych stronach zamieszczono zrzuty ekranu ze wszystkich widoków w grze. Poszczególne ich elementy zostały już omówione na etapie projektowania aplikacji, przy okazji prototypu interfejsu, dlatego w tym przypadku zaprezentowano jedynie ich finalną wersję, bez szczegółowych opisów.

Rys. 5.14. Interfejs użytkownika – menu główne

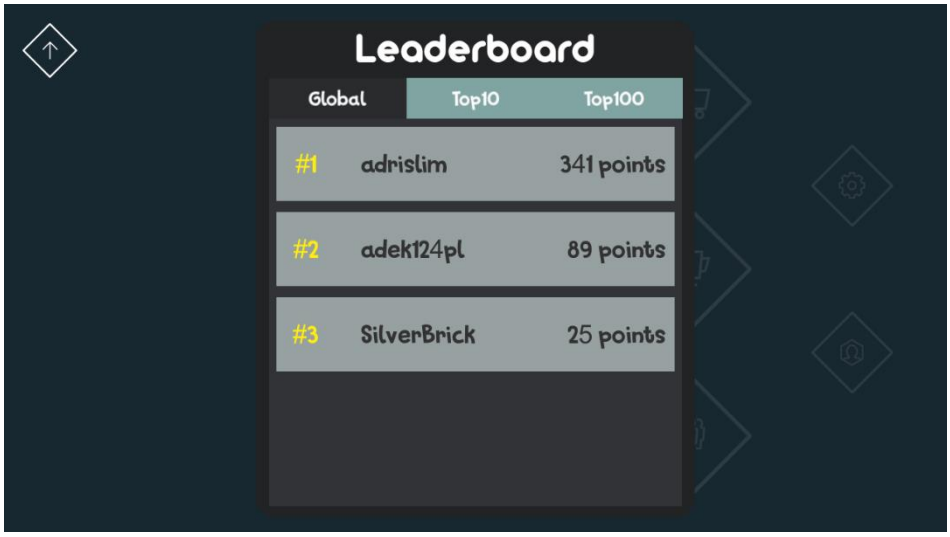


Rys. 5.15. Interfejs użytkownika – ekran wyboru poziomu

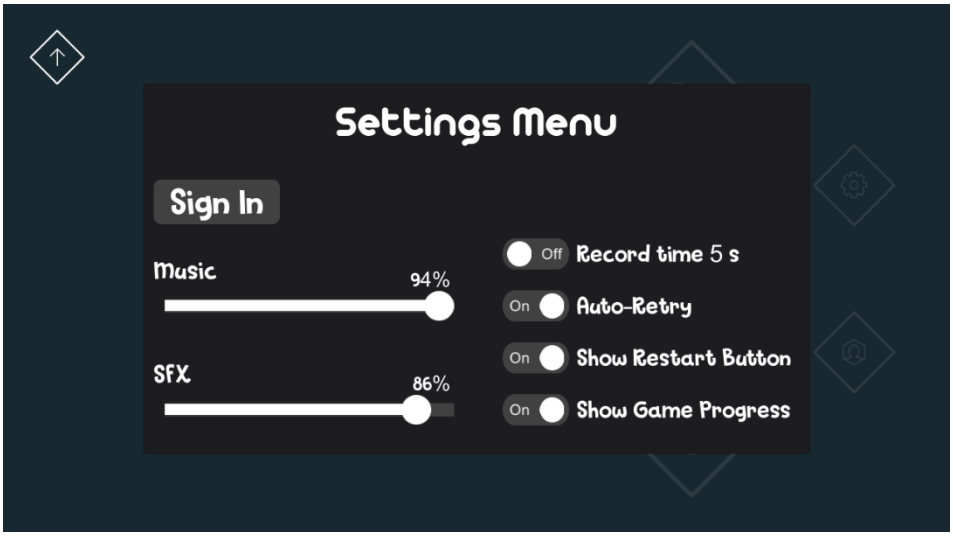


Rys. 5.16. Interfejs użytkownika – widok sklepu

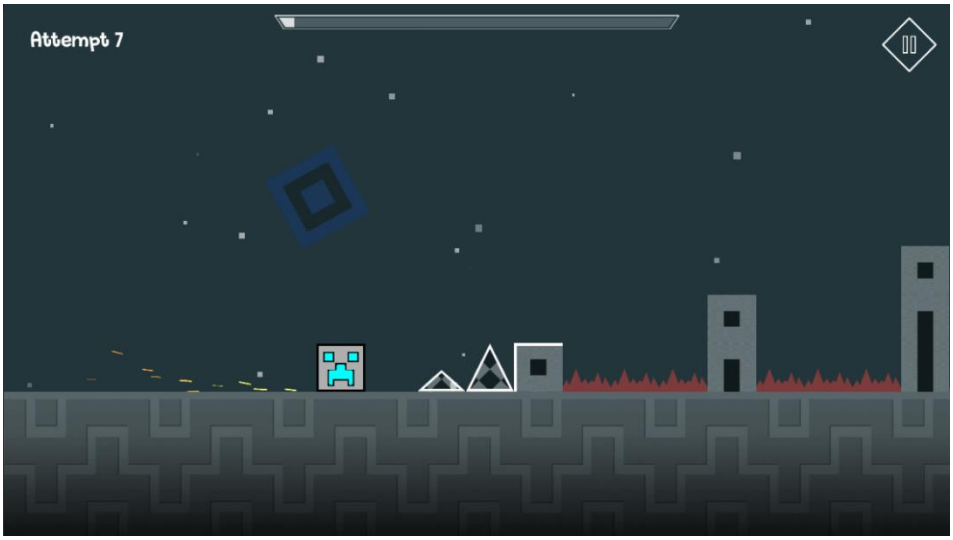
Rys. 5.17. Interfejs użytkownika – widok osiągnięć



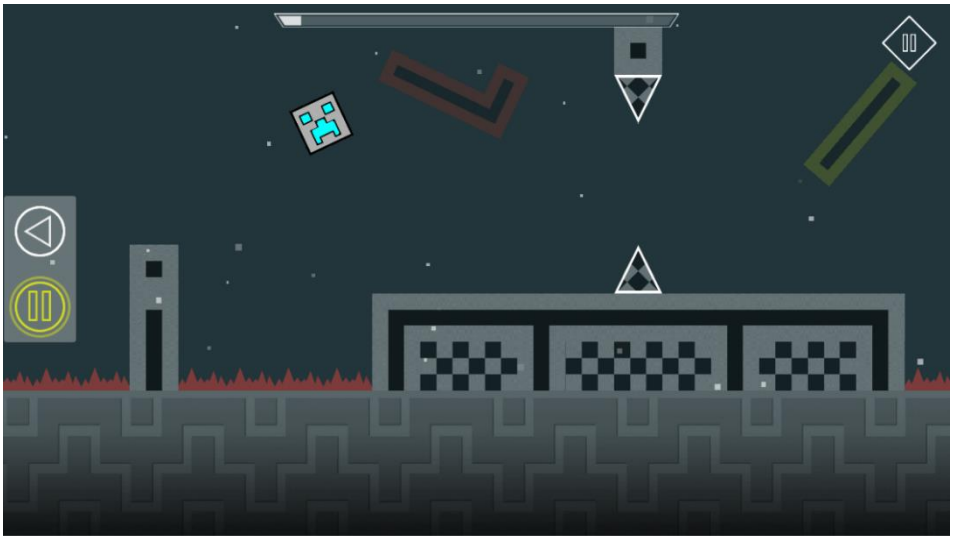
Rys. 5.18. Interfejs użytkownika – widok rankingu

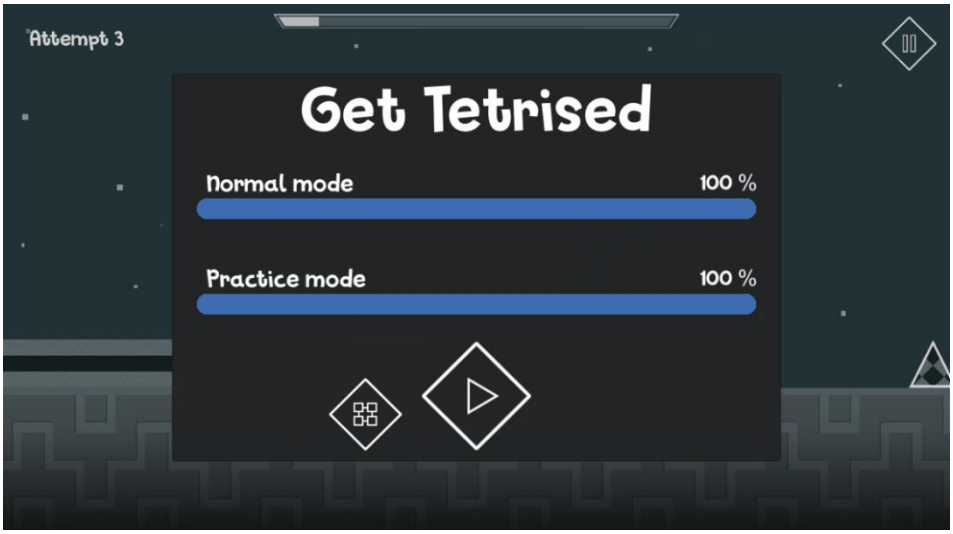


Rys. 5.19. Interfejs użytkownika – ekran ustawień

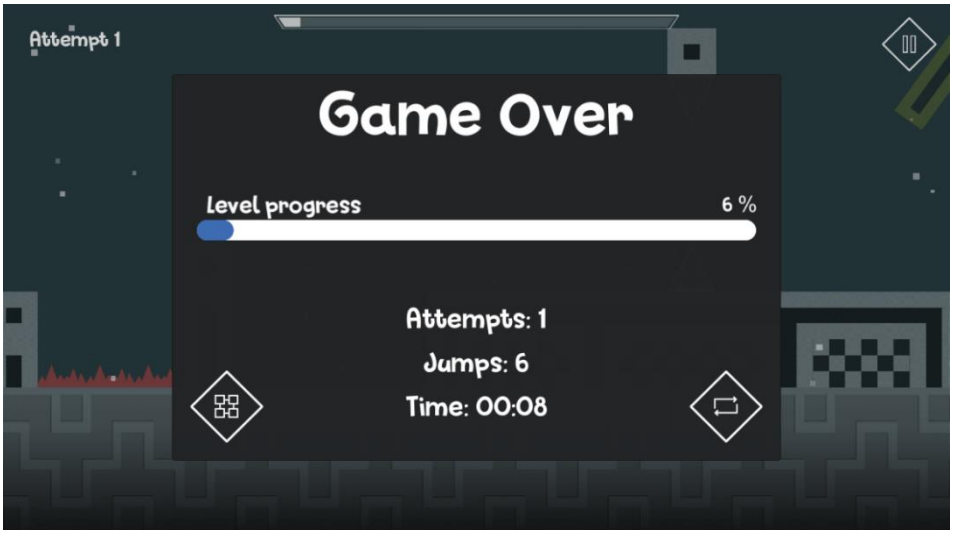


Rys. 5.20. Interfejs użytkownika – widok rozgrywki trybu normalnego

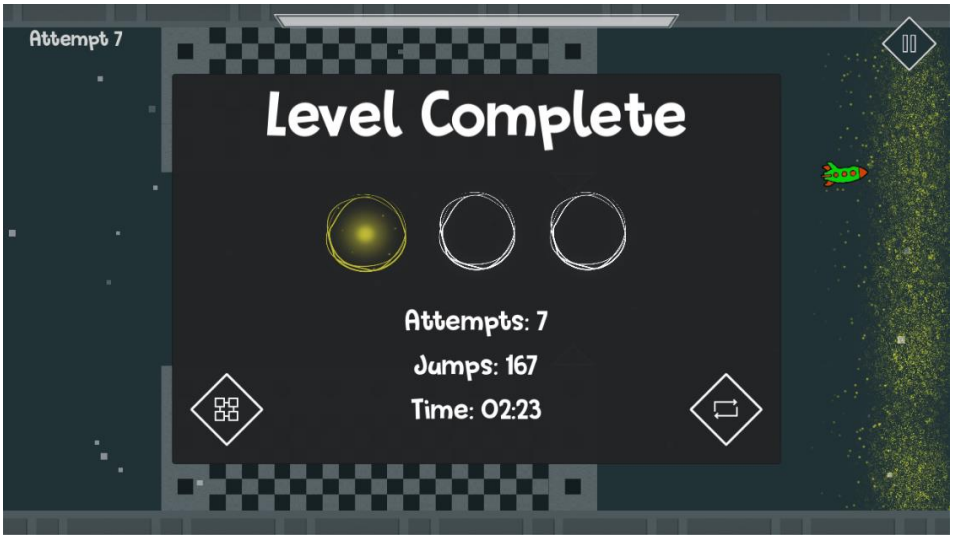

Rys. 5.21. Interfejs użytkownika – widok rozgrywki trybu praktyki



Rys. 5.22. Interfejs użytkownika – panel pauzy



Rys. 5.23. Interfejs użytkownika – panel przegranej



Rys. 5.24. Interfejs użytkownika – panel wygranej

6. Testy

Testowanie aplikacji jest obecnie nieoddzielnym elementem jej powstawania. W zależności od wybranej metody, może się odbywać w dowolnym momencie, a nawet trwać podczas całego cyklu produkcji. Jest to dobre podejście, ponieważ już we wczesnych fazach pozwala znaleźć defekty, które niewykryte, wraz z rozwojem aplikacji mogłyby być trudne do naprawy. Ich cel nie ogranicza się jedynie do wyszukiwania błędów w aplikacji, które powodują niepoprawne jej działanie. Równie ważnym aspektem przeprowadzanych testów jest sprawdzenie zgodności z wymaganiami stawianymi przez jej użytkowników oraz zapewnienie odpowiedniego poziomu prezentowanego przez produkt końcowy. W tym rozdziale zostaną omówione rodzaje testów jakim poddana została gra autorska, a także ich przebieg i znaczenie.

Testy jednostkowe

Testy jednostkowe są metodą testowania polegającą na weryfikacji poprawności działania pojedynczych elementów programu taki jak metody, obiekty czy procedury. Testowanie fragmentu kodu polega na wykonaniu go, a następnie porównaniu wyników z oczekiwanymi wynikami. Ich zaletą jest możliwość przeprowadzenia automatycznych testów na modyfikowanych fragmentach programu, co pozwala na wychwycenie błędu zaraz po jego pojawieniu się i szybką naprawę.

Unity posiada wbudowane narzędzie pozwalające na tworzenie oraz wykonywanie testów jednostkowych. Wykorzystuje ono biblioteki NUnit, która jest otwarto źródłową biblioteką przeznaczoną dla języków technologii .NET. Ze względu na specyfikę projektu, jakim jest gra, istnieją dwa różne sposoby na wykonywanie testów jednostkowych. Pierwszy z nich jest dostępny w trybie edycji, kiedy możliwe jest projektowanie poziomów, dodawanie nowych obiektów i ogółem tworzenie gry. Przebiega dość standardowo, jest włączany ręcznie, a po chwili dostępne są informacje o ewentualnych niepowodzeniach. Drugi natomiast ma miejsce podczas trybu rozgrywki, czyli gry uruchomionej w środowisku Unity w celach testowych. W tym przypadku, specjalnie zdefiniowane testy są wykonywane co określony czas, aby sprawdzić poprawność działania przy zmieniających się warunkach w grze.

Głównym zastosowaniem dla testów jednostkowych podczas realizacji tego projektu, była kontrola działania komunikacji z bazą danych. Ponieważ na pewnym etapie produkcji aplikacji, baza danych oraz jej zawartość była dość często modyfikowana, istotne okazało się sprawdzanie poprawności odczytywanych i zapisywanych danych dotyczących osiągnięć, odblokowanych przedmiotów oraz postępów w grze.

Testy funkcjonalne

Testy funkcjonalne nie opierają się na wewnętrznej budowie programu, lecz założeniach funkcjonalnych, które według wcześniejszych wymagań powinna spełniać aplikacja. Charakteryzują się większą szansą wykrycia niepoprawnych działań, ale nie dostarczają precyzyjnych informacji co do przyczyny wystąpienia błędu.

Istnieje możliwość zautomatyzowania testów, w sytuacjach, gdy pewne czynności są powtarzalne, co pozwala zaoszczędzić czas. W przypadku gier natomiast, bardzo wielu rzeczy nie da się przewidzieć, a błędy pojawić się mogą przy zajściu określonych warunków. Ze względu na to, testy zostały przeprowadzane manualnie przy dodawaniu każdej nowej

funkcjonalności. Wykorzystywane w tym celu było narzędzie silnika Unity, pozwalające na uruchomienie aplikacji bez konieczności jej instalacji na urządzeniu docelowym. Testowanie obejmowało między innymi poprawne działanie interfejsu, wybór i zapis grafiki postaci, a następnie wczytywanie tych elementów z bazy danych, prawidłowe działanie mechaniki rozgrywki, systemu sterowania, automatyczny zapis postępu gracza oraz przyznawanie osiągnięć.

Istotnym elementem każdej gry jest jej rozgrywka. Musi ona działać poprawnie i reagować na akcje przeprowadzane przez użytkownika w oczekiwany sposób. Ponad to, bardzo ważne jest odpowiednie jej zaprojektowanie, tak aby stanowiła dla gracza wyzwanie, lecz nie zniechęcała zbyt wysokim poziomem trudności. Należało tak zaplanować poziomy, aby stopniowo wprowadzać kolejne mechaniki, dając graczowi czas na zapoznanie się z nimi i ich opanowanie. Nie istnieje niestety żadne rozwiązanie, które rozwiązywało by ten problem. Ze względu na to istotne było znalezienie grupy chętnych, którzy zgodzili się przetestować grę przede wszystkim pod względem rozgrywki, ale również innych funkcjonalności.

Testy użyteczności

Głównym celem przeprowadzania testów użyteczności jest wykrycie obszarów utrudniających realizację niektórych funkcjonalności, a także tych wpływających negatywnie na satysfakcję podczas korzystania z aplikacji. Z ich pomocą można określić w jakim stopniu jest zrozumiała, prosta w użyciu i łatwa do nauczenia się, co przekłada się pozytywnie na atrakcyjność w oczach odbiorcy. Ich przebieg polega na wykonywaniu różnych zadań przez obecnych lub potencjalnych użytkowników. Następnie na podstawie uzyskanych informacji można wyciągnąć wnioski, będące podstawą do wprowadzenia zmian i poprawy jakości oprogramowania.

W celu przetestowania gry autorskiej pod względem użyteczności, znaleziono grupę chętnych, pośród których były osoby obeznane z tematem gier mobilnych, jak i takie nie mające z nimi wcześniej do czynienia. Ich zadanie polegało na zapoznaniu się ze wszystkimi funkcjonalnościami, bez wcześniejszych informacji jak tego dokonać oraz zgłaszanie wszelkich uwag na temat nieintuicyjnych i nieczytelnych fragmentów interfejsu, a także ogólnego przedstawienia swoich doświadczeń z użytkowania. Dzięki temu możliwe było wprowadzenie wielu kosmetycznych poprawek, które obejmowały zastąpienie niekiedy niewyraźnej czcionki oraz zmianę jej rozmiaru i koloru. Oprócz tego zdecydowano się na zmianę układu elementów interfejsu na niektórych ekranach, a także ikon i oznaczeń, które wydawały się nie jasne w stosunku do pełnionych przez nich funkcji. Spełnienie oczekiwań wszystkich osób uczestniczących w testach okazało się niemożliwe, ze względu na ich rozbieżne opinie i sugestie co do zmian. Wprowadzono więc rozwiązania, które zgadzały się z uwagami większości oraz wydały się najbardziej rozsądne.

Testy wydajnościowe

Oprócz poprawnego działania i braku błędów, każda aplikacja powinna działać płynnie i nie zawieszać się, dlatego istotne jest przeprowadzanie testów wydajnościowych. Szczególnie ważne jest to w przypadku gier, gdzie spadek płynności, czyli liczby wyświetlanych klatek na sekundę, może znacznie utrudnić, a nawet uniemożliwić rozgrywkę, co negatywnie wpływa na jej ocenę, bez względu na inne walory.

Gra autorska nie jest na tyle rozbudowana i skomplikowana, żeby nie mogły sobie z nią poradzić smartfony dostępne obecnie na rynku. Wszystkie funkcjonalności działały sprawnie, nawet na słabszym urządzeniu, Motorola Moto G z 2013 roku. Pojawił się natomiast problem z zauważalnym spadkiem płynności, dotyczący trybu praktyki. W celu umożliwienia użytkownikowi cofania czasu, należy zapamiętywać poprzednie wartości dotyczące postaci, przy każdej aktualizacji. Ma to miejsce dokładnie co dwie tysięczne sekundy (0,02 s), co wynika z silnika Unity. Dodatkowo należy uwzględnić potrzebę zapisania aż czterech wartości, czyli pozycji, rotacji, prędkości liniowej oraz kątowej. Przekłada się to na szybki wzrost liczby przechowywanych wartości, ponieważ co sekundę zapamiętywanych jest ich aż 200, natomiast po minucie rozgrywki, liczba wzrasta do 12 000. Ze względu na to zdecydowano się wprowadzić pewne ograniczenie, które pozwala nagrywać czas tylko 5 sekund. Jest ono opcjonalne, znajduje się w ustawieniach i to użytkownik ma prawo zdecydować. Podczas testowania na wydajniejszym urządzeniu, Samsung Galaxy S8, nie zauważono problemów ze spadkiem płynności.

7. Podsumowanie

W pracy zaprezentowany został proces tworzenia gry mobilnej, który rozpoczął się od sformułowania początkowych wymagań i założeń oraz rozeznania wśród rozwiązań konkurencyjnych dostępnych na rynku. Kolejnym krokiem był projekt aplikacji, dostarczający podstawowych informacji o funkcjonalnościach i prototyp interfejsu. Ostatnim etapem została implementacja, przybliżająca sposób działania i budowę najważniejszych fragmentów aplikacji, finalny wygląd interfejsu użytkownika oraz przedstawiająca wszystkie elementy rozgrywki, która stanowi najważniejszy aspekt gry.

Początkowo sformułowane cele i wymagania pracy udało się zrealizować. Wymagało to użycia i poznania wielu narzędzi, a ostatecznie utworzyło kompletną grę w silniku Unity. Ze względu na ograniczenia czasowe, pominięte zostały elementy rozbudowujące rozgrywkę, takie jak większa liczba poziomów i grafik pojazdów do wyboru, co nie stanowiło jednak głównego zadania niniejszej pracy.

Aplikacja pozostawia jeszcze wiele ścieżek rozwoju. Zaczynając od rozbudowy istniejących elementów wizualnych i wprowadzenia nowych, na przykład zmiana śladu pozastawianego za pojazdem oraz zaprojektowania większej liczby poziomów, co jest proste dzięki zaimplementowaniu wszystkich mechanik. Możliwe jest również znaczne urozmaicenie rozgrywki, poprzez dodanie kolejnych pojazdów, różnych w sposobie sterowania i obiektów wprowadzających specjalne efekty, takie jak zmiana szybkości poruszania się, chwilowe odwrócenie jego kierunku lub zmiana grawitacji. Wszystkie te zmiany znacznie rozbudują grę i wprowadzą nowe, ciekawe wyzwania, które z pewnością przyciągną użytkowników na dłużej.

8. Bibliografia

[1] Matt Smith, Chico Queiroz. *Unity 5.x Cookbook.* Packt Publishing, 2013
[2] Tommaso Lintrami. *Unity 2017 Game Development Essentials.* Packt Publishing
[3] Joseph Albahari, Ben Albahari. *C# 7.0 w pigułce. Wydanie VII.* Hellion
[4] Greg Lukosek. *Learning C# by Developing Games with Unity 5.x.* Packt Publishing
[5] Sunny Aditya, Vikash Karn. *Android SQLite Essentials.* Packt Publishing
[6] Robert C. Martin. *Czysty kod. Podręcznik dobrego programisty.* Hellion
[7] Dokumentacja silnika Unity. Ostatni dostęp: 02.10.2018 r. *https://docs.unity3d.com/Manual/index.html*
[8] Dokumentacja do obsługi plugin'u Play Game Services. Ostatni dostęp: 02.10.2018 r. *https://github.com/playgameservices/play-games-plugin-for-unity*
[9] Dokumentacja języka C#. Ostatni dostęp: 02.10.2018 r. *https://docs.microsoft.com/en-us/dotnet/csharp/*
[10] Opis środowiska Unity. Ostatni dostęp: 02.10.2018 r. *https://en.wikipedia.org/wiki/Unity_(game_engine)*
[11] Opis programu Visual Studio. Ostatni dostęp: 02.10.2018 r. *https://docs.microsoft.com/en-us/visualstudio/ide/?view=vs-2017*
[12] Opis pakietu Android SDK. Ostatni dostęp: 02.10.2018 r. *https://developer.android.com/sdk/index.html*
[13] Informacje na temat SQLite. Ostatni dostęp: 02.10.2018 r. *https://en.wikipedia.org/wiki/SQLite*
[14] Informacje na temat wzorca projektowego Singleton. Ostatni dostęp: 02.10.2018 r. *https://4programmers.net/In%C5%BCynieria_oprogramowania/Wzorce_projektowe/Singleton*
[15] Informacje na temat rynku gier mobilnych. Ostatni dostęp: 02.10.2018 r. *https://mobirank.pl/2018/05/02/w-2018-r-gry-mobilne-wygeneruja-51-proc-przychodow-na-rynku-gier/*
[16] Artykuł na temat przychodów z rynku gier. Ostatni dostęp: 02.10.2018 r. https://newzoo.com/insights/articles/global-games-market-reaches-137-9-billion-in-2018-mobile-games-take-half/
[17] Krótka historia gier wideo. Ostatni dostęp: 02.10.2018 r. *https://www.ppe.pl/blog/14716/3497/na-poczatku-byl-chaos-historia-gier-wideo-w-pigulce.html*
[18] Najpopularniejsze gry z gatunku Endless Runner. Ostatni dostęp: 02.10.2018 r. *https://www.androidauthority.com/best-endless-runner-games-android-690566/*
[19] Obrazki i informacje gry Temple Run. Ostatni dostęp: 02.10.2018 r. *https://play.google.com/store/apps/details?id=com.imangi.templerun&hl=en*
[20] Obrazki i informacje gry Jetpack Joyride. Ostatni dostęp: 02.10.2018 r. *https://play.google.com/store/apps/details?id=com.halfbrick.jetpackjoyride&hl=en*
[21] Obrazki i informacje gry Crossy Road. Ostatni dostęp: 02.10.2018 r. *https://play.google.com/store/apps/details?id=com.yodo1.crossyroad&hl=en*
[22] Obrazki i informacje o grze Geometry Dash. Ostatni dostęp: 02.10.2018 r. *https://play.google.com/store/apps/details?id=com.robtopx.geometryjump&hl=en*
[23] Źródło ikon interfejsu użytkownika. Ostatni dostęp: 02.10.2018 r. *https://www.flaticon.com/*
[24] Źródło grafik interfejsu użytkownika. Ostatni dostęp: 02.10.2018 r. *https://assetstore.unity.com/packages/tools/gui/ui-builder-29757*