МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Острозька академія»
Економічний факультет
Кафедра економіко-математичного моделювання
та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА/ПРОЄКТ
на здобуття освітнього ступеня бакалавра

на тему: **«РОЗРОБКА ІГРОВОГО ЗАСТОСУНКУ**

АРКАДНИХ ПЕРЕГОНІВ ПІД МОБІЛЬНУ СИСТЕМУ ANDROID НА РУШІЮ UNITY»

Виконав: студент 4 курсу, групи КН-41
першого (бакалаврського) рівня вищої освіти
спеціальності 122 Комп'ютерні науки
освітньо-професійної програми «Комп'ютерні науки»
Бунтін Максим Анатолійович

Керівник:
Гаврильчик Леонід Сергійович

Рецензент:
Місай Володимир Віталійович

РОБОТА ДОПУЩЕНА ДО ЗАХИСТУ

Завідувач кафедри економіко-математичного моделювання та інформаційних технологій

_____ (проф., д.е.н. Кривицька О.Р.)

Протокол № ____ від « ___ » _____ 2022 р.

Острог, 2022

Міністерство освіти і науки України
Національний університет «Острозька академія»

Факультет: *економічний*

Кафедра: *економіко-математичного моделювання та інформаційних технологій*

Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри економіко-математичного моделювання
та інформаційних технологій

_____ Ольга КРИВИЦЬКА

«____» _____20__ р.

ЗАВДАННЯ
на кваліфікаційну роботу/проєкт студента

Бунтіна Максима Анатолійовича

1. Тема роботи/проєкту Розробка ігрового застосунку аркадних перегонів під мобільну систему Android на рушію Unity

керівник роботи/проєкту Гаврильчик Леонід Сергійович

Затверджено наказом ректора НаУОА від 29 жовтня 2021року №110

2. Термін здачі студентом закінченої роботи/проєкту: 03 червня 2022 року

3. Вихідні дані до роботи/проєкту: постановка задачі, аналіз аналогів, вихідні дані.

4. Перелік завдань, які належить виконати: опис предметного середовища; огляд наявних аналогів; постановку задачі; опис архітектури рішення, що розроблюється; опис коду та інтерфейсу програми; тестування.

5. Перелік графічного матеріалу: діаграма прецедентів функціональних вимог, діаграма станів ігрового циклу, діаграма станів системи меню, діаграма станів додаткової сцени – пауза.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Гаврильчик Л.С.	01.12.2021р.	01.12.2021р.
2	Гаврильчик Л.С.	01.12.2021р.	01.12.2021р.
3	Гаврильчик Л.С.	01.12.2021р.	01.12.2021р.

7. Дата видачі завдання: 01.12.2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи/проєкту	Строк виконання етапів	Примітка
1	Затвердження теми роботи/проєкту	до 01.11.2021 р.	
2	Постановка завдання	до 01.12. 2021 р.	
3	Розробка архітектури та загальної структури системи	до 01.02.2022 р.	
4	Розробка структур окремих підсистем	до 01.03. 2022 р.	
5	Програмна реалізація системи	до 01.05.2022 р.	
6	Попередній захист кваліфікаційної роботи/проєкту	до 01.06.2022р.	
7	Здача кваліфікаційної роботи/проєкту на кафедру	03.06.2022 р.	

Студент: _____ Максим БУНТІН

Керівник кваліфікаційної роботи/проєкту: _____ Леонід ГАВРИЛЬЧИК

АНОТАЦІЯ
кваліфікаційної роботи/проєкту
на здобуття освітнього ступеня бакалавра

Тема: Розробка ігрового застосунку аркадних перегонів під мобільну систему Android на рушію Unity

Автор: Бунтін Максим Анатолійович

Науковий керівник: Гаврильчик Леонід Сергійович

Захищена «……»…… 20__ року.

Пояснювальна записка до кваліфікаційної роботи/проєкту: *53 с., 39 рис., 4 табл., 6 додатків, 12 джерел.*

Ключові слова: ігровий застосунок, Unity, 3D

Короткий зміст праці:
Кваліфікаційна робота/проєкт на тему: «Розробка ігрового застосунку аркадних перегонів під мобільну систему Android на рушію Unity» присвячена розробці 3D гри з можливістю користування на платформі Android. Актуальність обраної теми для кваліфікаційної роботи/проєкту пов’язана з великою популярністю мобільних ігрових додатків. У процесі роботи використано такі сучасні засоби як: ігровий рушій Unity, мова програмування C#, інтегроване середовище розробки Microsoft Visual Studio, графічний редактор Adobe Photoshop, середовище для створення 3D моделей Blender.

Qualification thesis on the topic: "Development of a game application of arcade racing for the Android mobile system on the Unity engine" is dedicated to the development of 3D games with the ability to use on the Android platform. The relevance of the topic chosen for the qualifying work is due to the great popularity of mobile gaming applications. In the process of work such modern means were used as: Unity game engine, C # programming language, Microsoft Visual Studio integrated development environment, Adobe Photoshop graphic editor, Blender 3D modeling environment.

_____

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

Debug – відлагоджувач, комп'ютерна програма, яка використовується для тестування і виправлення вад інших програм.

IDE – інтегроване середовище розробки – комплексне програмне рішення для розробки програмного забезпечення.

UML – уніфікована мова програмуваня **–** уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування.

Ассет – це компоненти, які є графікою, звуковим супроводом або скриптами. Вони прикріплюються до об'єктів і становлять важливу частину гри.

Векторне зображення – це зображення, що складається з геометричних об'єктів (ліній, кіл, кривих тощо), які описуються математичними рівняннями, **–** графічних примітивів.

ОС – операційна система.

ПЗ – програмне забезпечення.

ПП – програмний продукт.

Префаб – це набір заздалегідь готових ігрових об'єктів та компонентів, які використовуються більш ніж один раз за всю гру.

Растрове зображення – це зображення, що є набором пікселів, кожний із яких має певний колір.

Рендеринг, комп'ютерна візуалізація **–** в комп'ютерній графіці **–** це процес отримання зображення за моделлю з допомогою комп'ютерної програми.

Скрипт – це невелика програма, яка містить послідовність дій, створених для автоматичного виконання завдання.

Спрайт – двовимірне зображення, що застосовується в комп'ютерній графіці. Частіше за все — растрове зображення, що вільно переміщується по екрану.

ВСТУП

У XXI столітті – столітті інформаційного суспільства, людина отримує величезну кількість інформації, яку необхідно проаналізувати та структурувати. Для подальшого використання отриманої інформації необхідно забезпечити її зберігання. На сьогоднішній день це завдання допомагають вирішити мобільні пристрої, такі як смартфони, планшети, електронні книги, ноутбуки тощо. Проте, вищеописані переносні пристрої використовуються не тільки для роботи, але й для розважальної діяльності. Через велике інформаційне навантаження виникла потреба у частій зміні діяльності. На допомогу у вирішенні цієї проблеми створено мобільні ігрові програми, що не несуть важкого смислового навантаження, тим самим сприяючи відпочинку людини та переключенню уваги з однієї діяльності на іншу. Виходячи з вищевикладеного, можемо зробити висновок, що ігри є одним з основних продуктів на ринку інформаційних продуктів та послуг, що дозволяє розробникам розширити коло вироблених додатків. Також слід зазначити, що ігрові додатки роблять великий внесок у розвиток інформаційних технологій, наприклад, у розвиток програмного та апаратного забезпечення. Завдяки зростаючим вимогам користувача до якості ігор збільшуються вимоги до продуктивності пристроїв, що в свою чергу спричиняє розробку нових технологій. Саме тому розробка ігрових програм на сьогодні є найбільш затребуваною серед розробників мобільних додатків.

Мета дослідження – створення програмного продукту.

Задачі дослідження:

— аналіз предметної області;

— проєктування концепту ігрового застосунку;

— вибір інструментарію, методів реалізації та тестування ПП.

Об’єктом дослідження є процес аналізу мобільного геймінгу у жанрі аркадних гонок, вивчення їх ігрових механік.

Предметом дослідження є технології: ігровий рушій Unity, двовимірний графічний редактор Adobe Photoshop, 3D графічний редактор Blender, мова програмування C#.

Для досягнення мети та вирішення завдання реалізації мобільного ігрового додатку необхідно здійснити:

— опис предметного середовища;

— огляд наявних аналогів;

— постановку задачі;

— опис архітектури рішення, що розроблюється;

— опис коду та інтерфейсу програми;

— тестування.

РОЗДІЛ 1
ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1. Обґрунтування вибору середовищ розробки

1.1.1. Вибір ігрового рушія

Ігровий рушій (game engine) – це центральна програмна частина будь-якої комп'ютерної або мобільної відеогри та інших інтерактивних програм з графікою, що обробляється в реальному часі. Вона забезпечує основні технології, спрощує розробку та дає грі можливість запускатися на різних платформах, таких як: ігрові консолі та настільні операційні системи.

Починаючи розробляти будь який продукт, постає питання у якому середовищі варто створювати ігровий додаток. Для створення ігор є велика кількість ігрових рушіїв, яких з кожним роком стає все більше. Одні дозволяють розробляти 2D ігри (такі рушії як CONSTRUCT 2, Corona), інші 3D (Unreal Engine, Unity, Godot). За використання певних програмних продуктів потрібно заплатити місячну підписку або підписку на рік, проте існують й безкоштовні рішення. Деякі дозволяють створювати кросплатформні ігри, це означає, що створивши продукт на одній платформі, його можна перенести на іншу. Є ігрові компанії, які створюють свої власні ігрові рушії, тому що не бажають залежати від технологій інших компаній, хочуть оптимізувати робочий процес під гру, яку розроблюють, а також планують впроваджувати нові технології, які не підтримують існуючі рушії.

Розглянемо такі середовища розробки ігрових додатків як: Unreal Engine, Unity та Godot.

Unreal Engine – розробка компанії Epic Games. Unreal Engine дозволяє створювати ігри для більшості операційних систем, консолей і мобільних платформ. Завдяки підтримці різних систем рендерингу графіки, підтримці різних систем відтворення звуку і можливостей для мережевої гри. Unreal Engine може використовуватися для створення найрізноматніших ігор.

Рисунок 1.1 – Інтерфейс редактора Unreal Engine

Unity – це ігровий рушій, що допомагає створювати 2D та 3D ігри для ПК, Android, IOS, ігрових консолей, VR-пристроїв. Тут використовується компонентно-орієнтований підхід, в рамках якого розробник створює ігрові об'єкти (наприклад, головного героя) і до них додає різні компоненти (наприклад, візуальне відображення персонажа і способи керування ним). Завдяки зручному Drag & Drop інтерфейсу і функціональному графічному редактору рушій дозволяє конструювати карти, розставляти об'єкти в реальному часі і відразу ж тестувати отриманий результат.

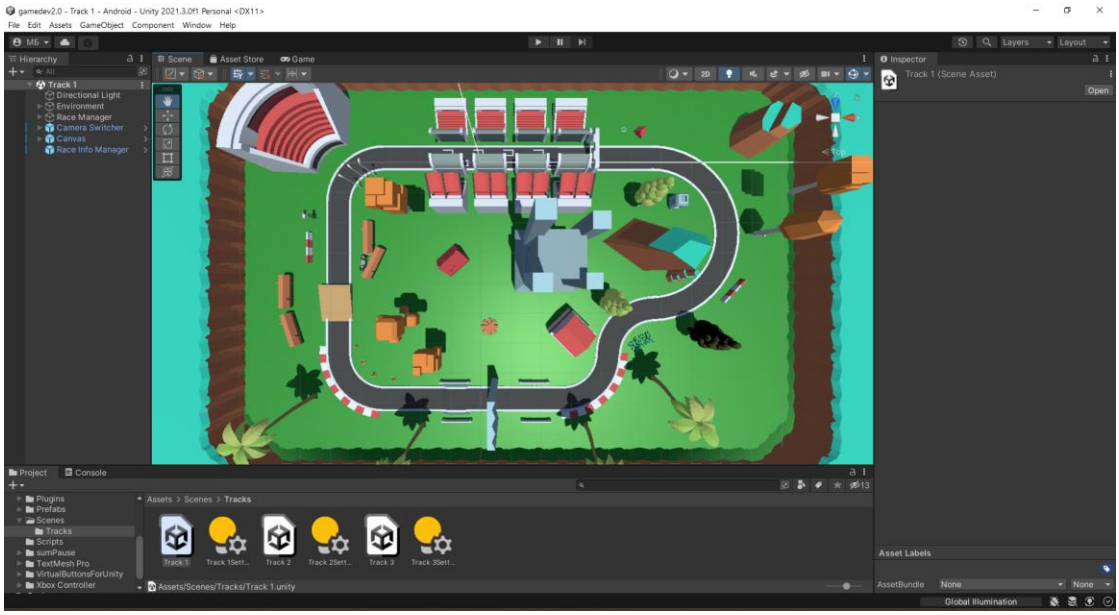

Рисунок 1.2 – Інтерфейс редактора Unity

Godot Engine – це багатофункціональний кросплатформний ігровий рушій для створення 2D і 3D ігор з уніфікованого інтерфейсу. Він надає повний набір поширених інструментів, щоб користувачі могли зосередитися на створенні ігор. Ігри можна експортувати одним кліком на ряд платформ, включаючи основні настільні платформи (Linux, macOS, Windows), мобільні платформи (Android, iOS), а також веб-платформи (HTML5) і консолі.

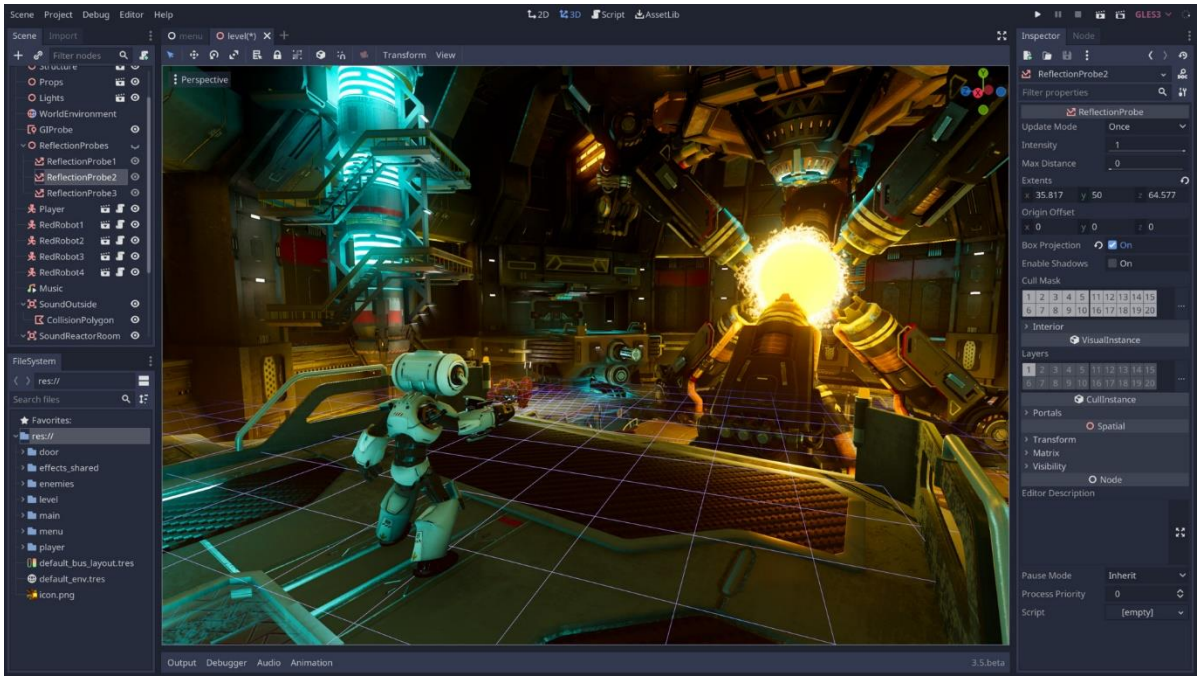

Рисунок 1.3 – Інтерфейс редактора Godot

Godot є повністю безкоштовним і відкритим вихідним кодом за дозволеною ліцензією MIT. Не потребує ніяких зобов’язань, жодних гонорарів. Ігри розробників належать їм до останнього рядка коду рушія. Розробка Godot є повністю незалежною та керованою спільнотою, що дає користувачам можливість допомагати формувати свій рушій відповідно до їхніх очікувань.

На етапі вибору середовища розробки, ми зупинимось між Godot Engine та Unity, так як Unreal Engine не є легким для вивчення, потребує набагато більше часу та ресурсів для роботи.

Godot має багато плюсів, він повністю безкоштовний, доступний інтерфейс. Godot порівняно простий у освоєнні та використанні ігровий рушій. Працювати з ним можуть як досвідчені розробники, так і ентузіасти-початківці. Вбудована мова програмування GDScript має майже такий самий синтаксис, що і

Python, і проста для вивчення. Інтуїтивно зрозумілий редактор, можливість інтеграції зі сторонніми плагінами та додатками також знижують поріг входження.

Якщо дивитись у бік Unity, тут зовсім інша ситуація. Для того, щоб писати гру у ігровому рушії Unity, потрібно вміти програмувати на мові C#. Це дуже потужна мова програмування, її функціонала більше ніж достатньо для виконання тих, або інших функцій. Хоча він не відноситься до низькорівневих мов програмування, це не заважає йому справлятися із своєю задачею.

Беручи до уваги всі плюси та мінуси обох рушіїв, було прийнято рішення працювати з Unity. Даний ігровий рушій ми детально розглядали під час вивчення навчальної дисципліни «Розробки ігрових додатків на Unity», це дало загальні поняття, практику користування інтерфейсом та подальшу розробку ігрового додатку.

1.1.2. Вибір середовища для розробки 2D графіки

Графічний редактор – прикладна програма (пакет програм), що дозволяє її користувачеві створювати й редагувати зображення на екрані комп'ютера і зберігати їх в графічних форматах файлів.

Серед графічних редакторів можна виділити два типи, що спеціалізуються на створенні растрових зображень та створені для роботи з векторними зображеннями. Для аналізу графічних редакторів було обрано три популярні програми: Adobe Photoshop, illustrator та GIMP.

Adobe Photoshop – багатофункціональний редактор для роботи з фото (растрові зображення та кілька векторних інструментів). Програмний продукт працює на ПК з операційними системами macOS, Windows та в мобільних версіях iOS та Android. Графічний редактор використовується для комерційних цілей (телебачення, кінематограф, реклама, ігри, ретуш і так далі).

Adobe illustrator – це універсальне середовище роботи з векторною графікою, що включає набір неперевершених інструментів для роботи, що дозволяє створювати логотипи, значки, малюнки та ілюстрації для друкованих видань, веб-публікацій, відео та мобільних пристроїв.

GIMP – це кросплатформний редактор зображень, доступний для Linux, macOS, Windows та інших операційних систем. Це безкоштовне програмне забезпечення, де користувач може змінювати його вихідний код і поширювати свої зміни.

Таблиця 1.1

Порівняльна таблиця графічних редакторів

Критерії	Програмне забезпечення		
	Photoshop	Illustrator	Gimp
Локалізація (українська мова)	Присутня	Присутня	Присутня
Вартість продукту	Безкоштовний пробний період	Безкоштовний пробний період	Безкоштовна
Можливість роботи з векторними зображеннями	Обмежена	Присутня	Присутня
Можливість роботи з растровими зображеннями	Присутня	Обмежена	Обмежена
Можливість постобробки зображення	Присутня	Обмежена	Обмежена

В результаті аналізу трьох графічних редакторів (табл.1.1) було обрано Adobe Photoshop. Програмне забезпечення, яке підходить для розробки графічного дизайну, редагування зображень та ретушування фотографій. Програма має великий функціонал, спеціалізується на роботі з растровими зображеннями та використовується художниками-ілюстраторами для створення графіки.

1.1.3. Вибір середовища для розробки 3D моделей

У кваліфікаційній роботі нам знадобиться середовище для моделювання 3D графіки. Серед безлічі програм для 3D дизайну, потрібно обрати правильне програмне забезпечення, яке буде відповідати декільком потребам: підтримка імпортування моделей у Unity, невисокі вимоги до характеристик ПК, зручний та зрозумілий інтерфейс.

Серед наявних програмних продуктів, які можуть підійти під вимоги до проєкту, можна виділити Blender та 3ds Max.

3ds Max – тривимірний графічний редактор, повнофункціональний професійний застосунок, система для створення і редагування об’єктів та створення візуалізацій, розроблена компанією Autodesk. Містить найсучасніші засоби для архітекторів, дизайнерів, художників і фахівців в області мультимедіа. Працює в операційних системах Microsoft Windows та Mac OS.

3ds Max використовують для візуалізації моделей будівель, комп’ютерних ігор, тривимірних анімаційних мультфільмів, рекламних роликів тощо. За допомогою цього редактора зроблено безліч анімованих моделей для кінофільмів.

Blender – безкоштовний програмний продукт, призначений для створення та редагування тривимірної графіки. Програма поширена на всіх популярних платформах, має відкритий вихідний код і доступна абсолютно безкоштовно всім охочим, а також є версія українською мовою.

Ці особливості зробили її вкрай популярною як серед користувачів-початківців, так і серед справжніх професіоналів моделювання. Дизайнери нерідко вибирають Blender як основний робочий інструмент для великих і серйозних проєктів.

Таблиця 1.2

Порівняльна таблиця ПЗ для тривимірної графіки

Критерії	Програмне забезпечення	
	Blender	3ds Max
Системні характеристики	Середні	Високі
Локалізація (українська мова)	Присутня	Немає
Вартість продукту	Безкоштовний	Платний
Сумісність з Unity	Присутня	Присутня
Швидкість рендерингу	Швидша, порівнюючи з 3ds Max	Повільніша, порівнюючи з Blender

Кожний тривимірний графічний редактор має свої особливості та унікальний набір характеристик. Не завжди корпоративний інструмент буде кращим вибором, ніж його безкоштовний аналог, але й навпаки.

Проаналізувавши вимоги, потреби та доцільність, для створення моделей оточення та треку (табл. 1.2) буде застосовано програмне забезпечення для створення і редагування тривимірної графіки – Blender. Blender є досить потужним 3D редактором, який активно розвивається. Навіть зараз він є відмінною альтернативою дорогим додаткам і цілком виконує поставлені завдання.

1.1.4. Висновки

Беручи до уваги вище описані програми та їх переваги найкращим рушієм для створення продукту є використання Unity із застосуванням мови програмування C#.

Розробка програмного коду буде відбуватися у середовищі Visual Studio. Для створення 2D графіки буде використана програма Adobe Photoshop, а для 3D моделей сцени та автомобілів буде використано Blender та Unity Asset Store. Короткий порівняльний аналіз обраного програмного забезпечення наведено в таблиці 1.3.

Таблиця 1.3

Порівняльний аналіз програмних продуктів для розробки мобільного додатку

Категорія	**Варіанти програм**	**Висновок**
Ігровий рушій	Unity, Godot, Unreal Engine	Unity дає можливість розробляти ігри, не вимагаючи якихось особливих знань. Завдяки зручному Drag & Drop інтерфейсу рушій дозволяє малювати карти і розставляти об'єкти в реальному часі. Наявність бібліотеки ассетів та плагінів, за допомогою яких можна значно прискорити процес розробки гри
Мова програмування	C#, C++, JavaScript	Ігровий рушій визначив мову програмування (C#)
Середовище розробки програмного коду	Microsoft Visual Studio, VS Code	Visual Studio – це IDE для роботи з мовою програмування C#. Використання IntelliSense та навігації по коду спрощує переміщення по скриптам, а Debug дозволяє швидко виявити проблеми

Створення 2D графіки	Adobe Photoshop, Gimp, illustrator	Photoshop є найпопулярнішим редактором для 2D графіки. Його інструменти дозволяють редагувати існуючі зображення, накладати фільтри, або ж малювати їх з нуля. Підтримується велика кількість форматів, які використовуються в Unity
Створення 3D моделей	Blender, 3ds Max	Blender – безкоштовна програма яка містить основні необхідні інструменти для створення 3D моделей, не вимоглива до ресурсів комп’ютера
Додаткові ресурси	Unity Asset Store	Unity Asset Store – це бібліотека ресурсів, що постійно зростає. Доступні різні типи ресурсів, від текстур, анімації та моделей до повних прикладів проєктів, навчальних посібників та розширень редактора

1.2. Опис предметного середовища

Unity – кросплатформенний рушій розробки комп'ютерних ігор. Unity дозволяє створювати програми, що працюють під більш ніж 20 різними операційними системами, що включають персональні комп'ютери, ігрові консолі, мобільні пристрої, інтернет-програми та інші.

Як правило, Unity надає безліч функціональних можливостей, що дозволяють їх задіяти в різних іграх, в які входять моделювання фізичних середовищ, карти нормалей, динамічні тіні та багато іншого. На відміну від багатьох ігрових рушіїв, у Unity є дві основні переваги: наявність величезної бібліотеки ассетів та плагінів та кросплатформенна підтримка. Перший фактор включає не тільки інструментарій візуального моделювання, а й інтегроване середовище, ланцюжок складання, що спрямоване на підвищення продуктивності розробників, зокрема етапів створення прототипів та тестування. Під кросплатформенною підтримкою мається на увазі не тільки місце розташування (установка на персональному комп'ютері, на мобільному пристрої, консолі тощо), але й наявність інструментарію розробки (інтегроване середовище може використовуватись під Windows та MacOS).

До недоліків відносять: відсутність підтримки Unity посилань на зовнішні бібліотеки, роботу з якими програмістам доводиться налаштовувати самостійно, що ускладнює командну роботу, складнощі при роботі з багатокомпонентними схемами та затруднення при підключенні зовнішніх бібліотек.

1.3. Огляд існуючих платформ

Наявність операційної системи (ОС) – головна особливість, яка відрізняє смартфон від звичайного мобільного телефону. При виборі конкретної моделі телефону або пристрою, операційна система часто є визначальним фактором.

Найпопулярнішими операційними системами для мобільних пристроїв нині є Android та iOS.

Android – це мобільна операційна система, яка існує вже майже 15 років. Користувачі знаходять дану платформу як базову операційну систему телефонів і планшетів у всьому світі. Крім того, існують інші операційні системи, які підтримують програми Android, зокрема ОС Chrome і Windows 11.

Android на сьогоднішній день є найпопулярнішою операційною системою у світі. За оцінками, вона працює на 2,5 мільярдах активних пристроїв по всьому світу з понад 3 мільярдами користувачів або приблизно 39% усього населення планети. Це значно перевершує iOS від Apple, яка є другою за популярністю операційною системою в усьому світі.

Переваги розробки мобільних додатків під ОС Android:

— швидка публікація та менше обмежень;
— більше користувачів і завантажень;
— ефективна монетизація за допомогою реклам.

Недоліки розробки мобільних додатків під ОС Android:

— довший час розробки;
— нижчий середній дохід;
— більше можливих помилок.

iOS – одна з найпопулярніших мобільних операційних систем, розроблена і створена Apple Inc. Пристрій iOS – це електронний гаджет, який працює на iOS. До пристроїв Apple iOS належать: iPad, iPod Touch та iPhone. iOS – друга за популярністю ОС після Android. Протягом багатьох років пристрої Android та iOS дуже сильно змагаються за більшу частку ринку.

Переглянемо переваги розробки мобільних додатків під iOS:

— вищий середній дохід;

— швидший розвиток;

— менше можливих помилок.

Недоліками ж є:

— у два рази менше користувачів;

— тривала публікація з багатьма обмеженнями;

— необхідність придбання спеціального обладнання.

В App Store (платформа для пошуку та завантаженню додатків для iOS пристроїв), щоб мати змогу публікувати додатки потрібно зробити грошовий внесок у розмірі 99$. Реєстрація фізичної особи, яка в майбутньому викладатиме додатки у вільний доступ, складає від 3 тижнів. Далі розробника чекають довгі та не зовсім прості процеси.

Платформа Android теж не дозволяє безкоштовно викласти гру у вільний доступ, для цього також потрібно внести разову оплату розробника у розмірі 25$, однак, Google надає набагато менше вимог до публікованих програм, ніж це робить Apple.

Також ключовим є варіант монетизації, який теж набагато простіший для приладів під керуванням ОС Android, та беручи до уваги кількість девайсів під даною платформою, можна сміло заявити, що реклама принесе більший дохід.

Розробка додатків для Android проти додатків iOS має свої переваги та нюанси залежно від цілей, аудиторії та варіантів монетизації, які розробник збирається реалізувати.

У кваліфікаційній роботі вирішальними стали пункти: «тривала публікація з багатьма обмеженнями» та «необхідність придбання спеціального обладнання». Так як, на останньому етапі розробки кваліфікаційна робота повинна бути кінцевим робочим проєктом, який буде можливим завантажити у будь-яку крамницю для того, щоб користувачі могли мати вільний та легкий доступ для завантаження додатку на свій прилад, потрібно визначитись з операційною системою мобільного пристрою.

Розглянувши і проаналізувавши дві найбільш популярні платформи для розробки мобільних додатків, можна зробити висновок, що Android є найбільш підходящою операційною системою для створення програм з наступних причин:

— посідає перше місце за поширеністю на ринку мобільних пристроїв;
— дозволяє завантажувати програми з інших джерел, крім офіційного магазину програм;
— обліковий запис розробника має найнижчу вартість.

1.4. Огляд наявних аналогів

На перших етапах розробки власного ігрового додатку необхідно проаналізувати схожі мобільні додатки їх механіки, цілі, дизайн. До уваги були взяті наступні додатки:

— Super Arcade Racing
— SpotRacers
— Table Top Racing

Super Arcade Racing – це гоночна гра у стилі ретро, яка занурить вас у атмосферу справжніх гральних автоматів. У цій класичній 2D-грі є все: просте управління, ігровий принцип «згори донизу», пікселований дизайн, класичний саундтрек.

Рисунок 1.4 – Процес гри в Super Arcade Racing

SpotRacers – це гра, в якій гравцю доведеться сканувати та збирати реальні машини з вулиці та використовувати їх для гонки проти друзів. Гравцеві пропонується створити власну колекцію автомобілів та налаштувати кожну машину відповідно до стилю гри та особистостей.

Рисунок 1.5 – Процес гри в SpotRacers

Table Top Racing містить усі види божевільних автомобілів, що борються один з одним у світі настільних гоночних трас та складних перешкод. Гравець може оновлювати автомобілі в гаражі, досліджувати межі можливостей траси та відкривати приховані найкоротші шляхи!

Рисунок 1.6 – Процес гри в Table Top Racing

Жанр гонок досить популярний у сфері мобільного геймінгу. Загальні механіки успішно реалізуються на кожній з платформ, але спосіб реалізації може відрізнятися.

Розглянувши популярних представників ігор жанру аркадних перегонів, були обрані основні та допоміжні механіки гри, стилі зовнішнього вигляду та предметів оточення для майбутнього додатку. Також обрано вигляд камери типу TopDown для спрощення ігрового процесу, адже гравцю так буде зручніше контролювати оточення та опонентів.

Висновок до розділу 1

У першому розділі виконано вибір ігрового рушія, середовища для розробки 2D і 3D моделей, здійснено опис предметного середовища, а також оглянуто існуючі платформи для розробки і наявні аналоги.

РОЗДІЛ 2
ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Аналіз предметної області

Перегони – жанр комп'ютерних ігор з видом від першої або третьої особи, в яких гравець бере участь у гоночному змаганні серед наземних, водних, повітряних або космічних транспортних засобів. Загалом, перегонами можуть бути ігри різних жанрів: від автосимуляторів високого рівня реалізму до звичайних аркадних перегонів.

Перегони в стилі аркад беруть за основу захоплюючий ігровий процес і динамічний досвід, оскільки автомобілі зазвичай змагаються унікальними способами. Ключовою особливістю перегонів у аркадному стилі, яка відрізняє їх від автосимуляторів, є набагато ліберальніша фізика. У той час як у реальних перегонах (отже, у симуляторах) водій повинен значно знизити швидкість, щоб зробити більшість поворотів, гоночні ігри в аркадному стилі зазвичай спонукають гравця «ковзати під дією сили», щоб дозволити зберегти швидкість дрифтуючи через поворот. Зіткнення з суперниками, перешкодами на трасі або транспортними засобами зазвичай набагато перебільшені, ніж у автосимуляторах. Здебільшого гравці в аркадному стилі просто прибирають точність і строгість, необхідні для симуляції, і зосереджуються виключно на самому гоночному елементі.

Ігровий додаток повинен містити наступні механіки:

— відрахунок (кращий час кругу, поточний час кругу);
— дрифт (керований занос автівки);
— відкриття наступного рівня після успішного проходження попереднього.

2.2. Постановка задачі

У результаті аналізу предметної області було сформульовано мету кваліфікаційної роботи: розробка мобільної гри аркадні перегони «MiniArcadeRace» у

стилі ізометрія, щоб дати змогу гравцям відчути емоції, які притаманні аркадним іграм, також підвищити швидкість рефлексів, покращити координації рухів.

Для досягнення мети кваліфікаційної роботи був визначений перелік основних завдань:

— виконати аналіз предметної області та огляд наявних аналогів;
— визначити інструментарій розробки;
— спроєктувати функціонал додатку;
— розробити дизайн ігрового продукту;
— розробити механіки;
— налаштувати сцени та розробити ігрові рівні;
— протестувати ігровий додаток.

Мобільний додаток повинен забезпечувати корисне проведення часу в грі та мати можливість покращувати рефлекси чи тактичні навички в грі.

2.3. Розробка архітектури ігрового додатку

Створення ігрового додатку дуже трудомісткий і складний процес. Даний процес об’єднує у собі величезну кількість елементів у єдину програму, призначену для того, щоб розважати геймера.

Для гри необхідно реалізувати модуль, який відповідає за ігрову логіку і фізику, який буде управляти взаємодією ігрових об'єктів на сцені, перемикати сцени в залежності від обраного пункту меню, завантажувати карти рівнів і так далі.

Розгорнутий опис функціональних вимог здійснюється на етапі проєктування програми. Для того, щоб деталізувати вимоги, необхідно виділити процеси, що відбуваються в заданій предметній області. Опис таких процесів на UML виконується у вигляді прецедентів (рис. 2.1). Прецеденти є сценарієм або варіантом використання ігрового програми при взаємодії із зовнішнім середовищем. За допомогою прецедентів описується функціонування гри з точки зору користувача, який називається в UML актором (Гравець). Актор є будь-якою

зовнішньою по відношенню до модельованої системі сутністю (людина, штучний інтелект), яка взаємодіє з системою і використовує її функціональні можливості для досягнення певної мети.

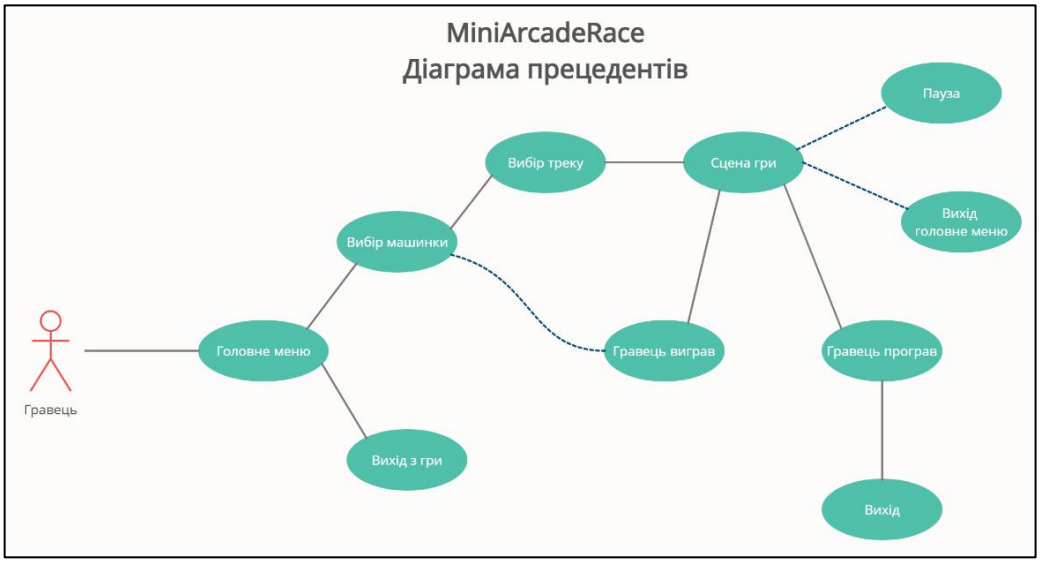

Рисунок 2.1 – Діаграма прецедентів

2.4. Game Loop

Game Loop (ігровий цикл) – це система, яка здійснює послідовний виклик модулів у правильній послідовності. Ігровий цикл буде оформлений у вигляді діаграми, реалізований у скриптах та підключений до MainCamera.

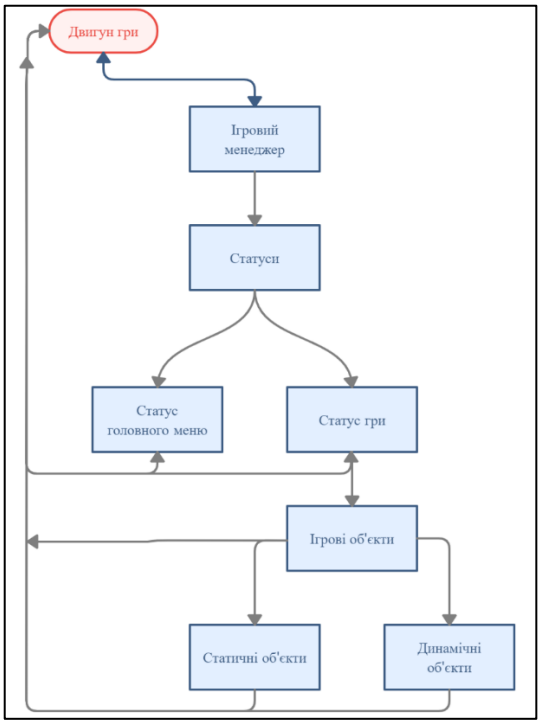

Рисунок 2.2 – Діаграма Game Loop

2.5. Проєктування інтерфейсу гри

В результаті аналізу предметної області та вимог, був розроблений інтерфейс ігрового додатка. Кожне меню або ігровий режим є окремим станом, який має свій клас, призначений для його перемикання.

Основне ігрове поле (рис. 2.3) містить ігрові об'єкти логіки представлення. На екрані присутні лічильники часу кругу (поточний час, час найкращого кругу), поточна позиція гравця, показник поточного/усіх кругів. Також у нижній частині розташовані функціональні кнопки, призначені для керування автомобілем та лічильник відліку до початку перегонів. Кнопки розташовані так, щоб користувач міг одним пальцем натиснути будь-яку кнопку.

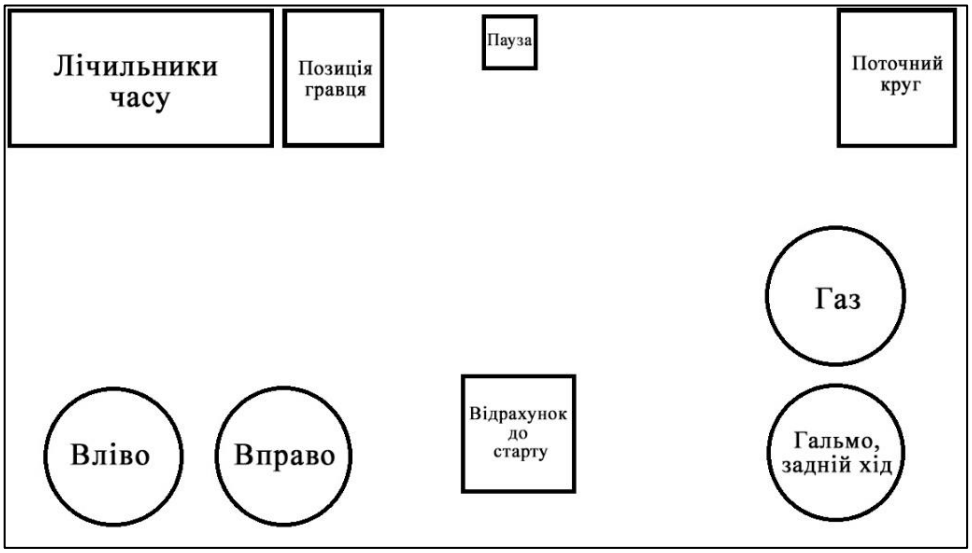

Рисунок 2.3 – Прототип інтерфейсу ігрової сцени

2.6. Система меню

Кожна гра починається з головного меню. Дана система дозволяє гравцеві запускати гру, виходити з програми. Для її проєктування вкрай зручно використати діаграму станів.

Головне меню складається з:

— Кнопка «Грати»

— Вибір авто

— Вибір треку

— Кнопка «Вихід»

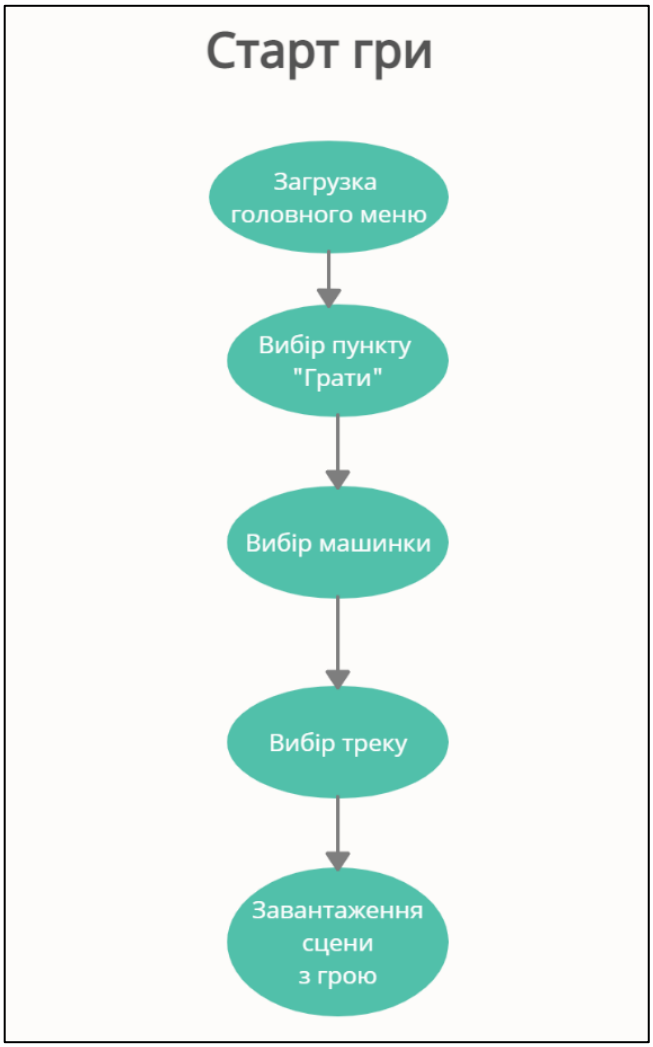

Рисунок 2.4 – Діаграма станів – Старт гри

Рисунок 2.5 – Прототип головного меню гри

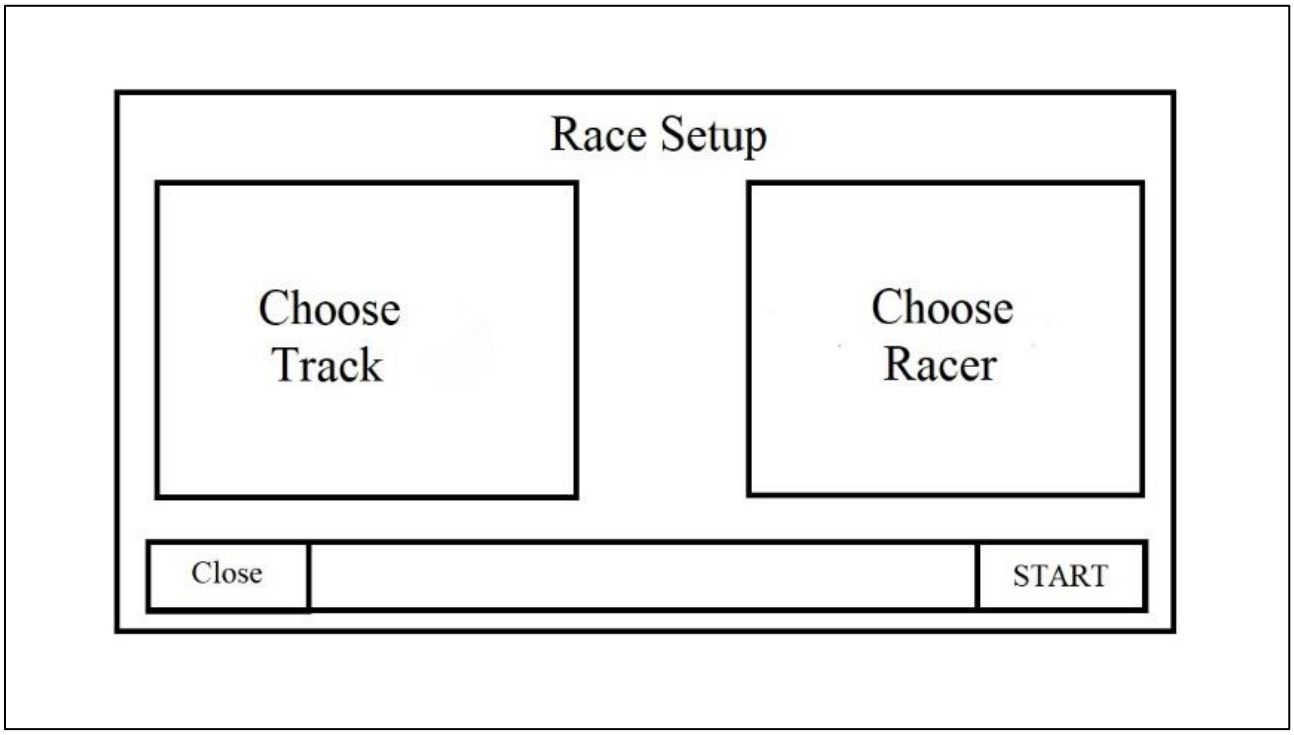

Рисунок 2.6 – Прототип меню вибору треку та автомобіля

Рисунок 2.7 – Діаграма станів – Вихід з гри

Додаткова сцена – Пауза, під час процесу гри

Висновок до розділу 2

У другому розділі було проведено аналіз предметної області, підготовлено постановку задачі, виконано розробку архітектури ігрового додатку, яка включає: ігровий цикл, проєктування прототипів інтерфейсів додатку.

РОЗДІЛ 3
ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Засоби розробки

Засоби розробки включають технічну документацію таких середовищ: Unity, Microsoft Visual Studio, Adobe Photoshop та Blender, – для роботи з інтерфейсом і бібліотеками, які використовуються при створенні ігрового продукту; публікації на відеохостингу YouTube, а також статті в Інтернеті.

3.2. Вимоги до технічного та програмного забезпечення

Для створення ігрового застосунку впроваджено наступні вимоги до програмного забезпечення:

— Unity – ігровий рушій;
— C# – мова програмування;
— Microsoft Visual Studio – середовище програмування;
— Unity Asset Store – бібліотека моделей та плагінів;
— Adobe Photoshop – створення двовимірної графіки;
— Blender – створення тривимірної графіки.

Вимоги до технічного забезпечення пристрою для розробки:

— Процесор: Intel Core i5-3470
— Відеокарта: NVIDIA® GeForce® GTX 660 2 ГБ
— Оперативна пам'ять: 8 ГБ
— Операційна система: Windows 10/11 64-розрядна

Вимоги до технічного забезпечення мобільного пристрою:

— Процесор: 4-ядерний
— Оперативна пам'ять: 2 ГБ
— Місце на диску: 100 МБ
— Операційна система: Android 5.0

3.3. Опис програмної реалізації

3.3.1. Створення меню

Щоб створити меню в Unity необхідно небагато часу, для цього потрібно створити окрему сцену, під назвою MainMenu.

Сцена MainMenu включає налаштування камери, саме головне меню та подальші налаштування перегонів (вибір авто та траси), аудіо супроводження, а також скрипт RaceInfoManager, про який ми розповімо пізніше.

Рисунок 3.1 – Сцена головного меню

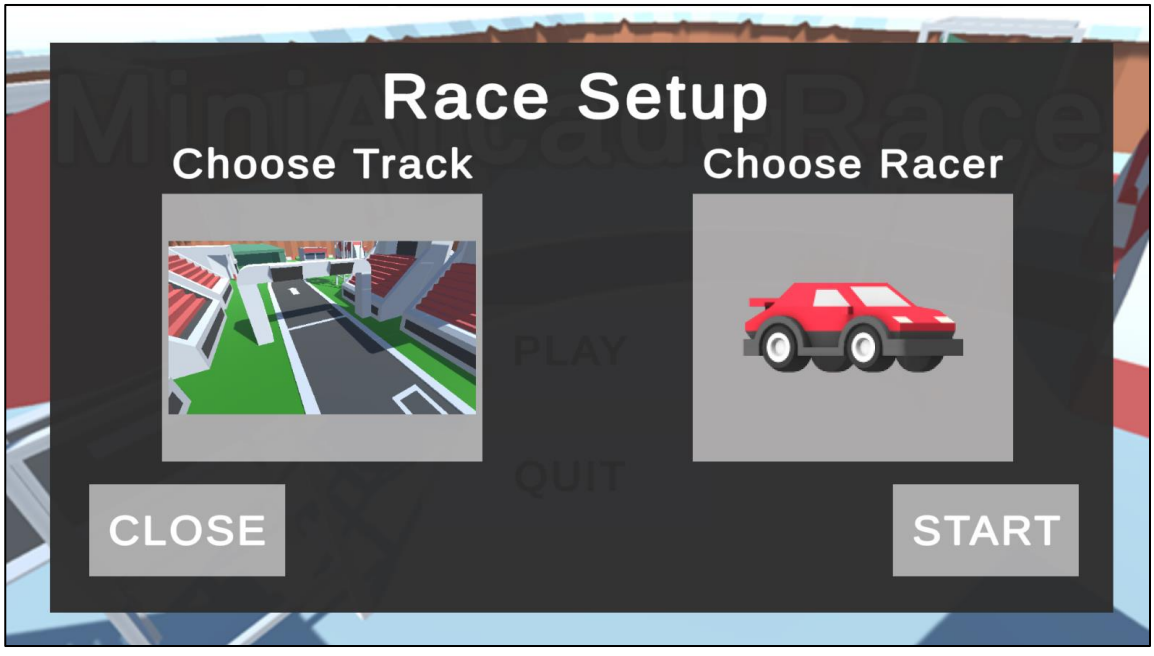

Рисунок 3.2 – Меню Race Setup

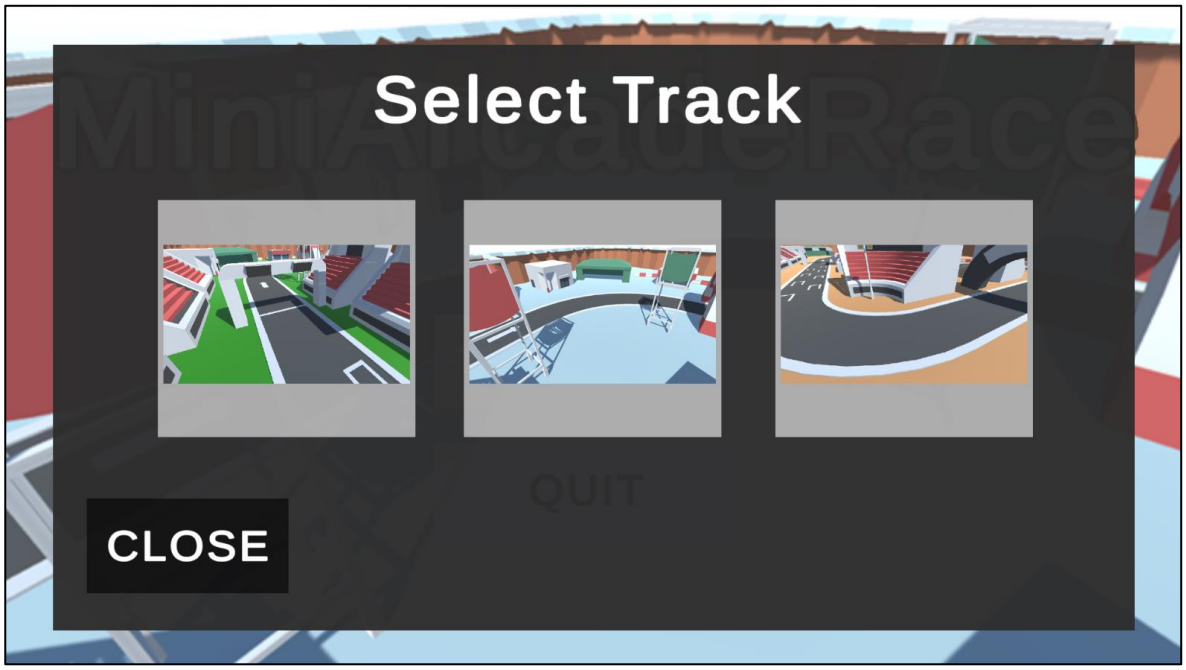

Рисунок 3.3 – Меню Select Track

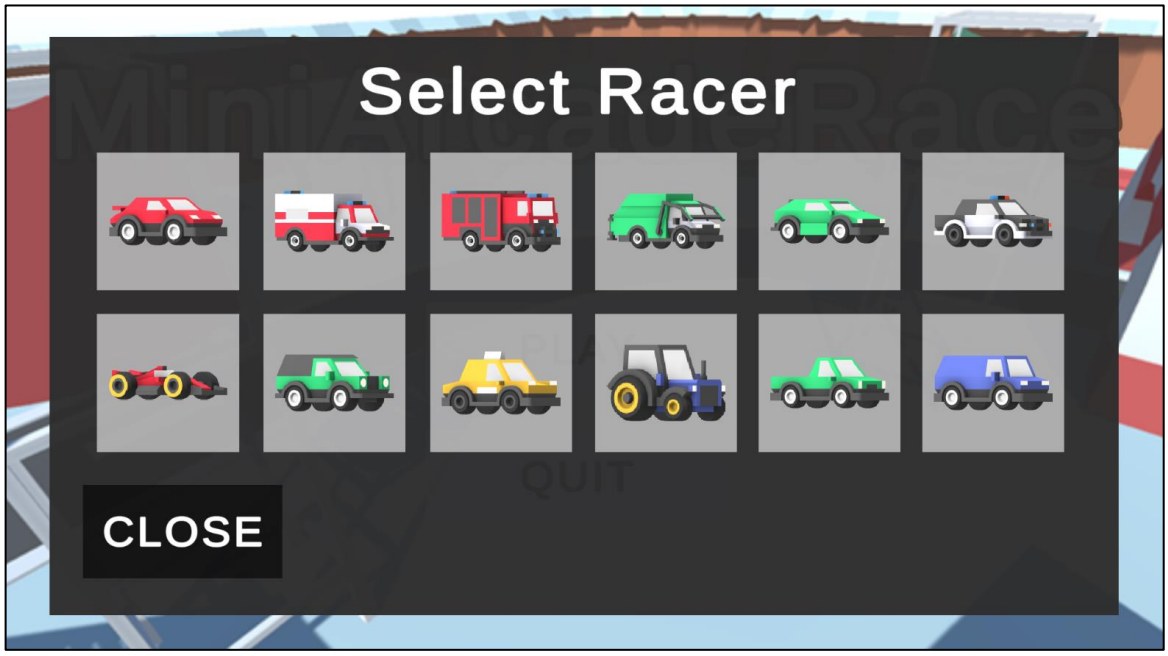

Рисунок 3.4 – Select Racer

Після вибору автомобілю, та треку, натискаємо кнопку «START» у меню Race Setup, після чого автоматично включиться наступна сцена, під назвою Track1.

Скрипт MainMenu має такі елементи:

— вибір траси;

— вибір авто;

— зображення відповідних елементів.

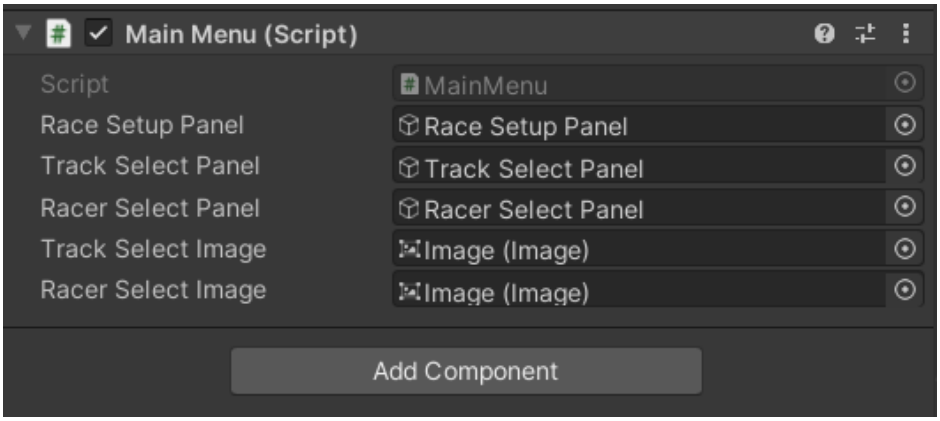

Рисунок 3.5 – MainMenu

3.3.2. Моделювання тестового треку

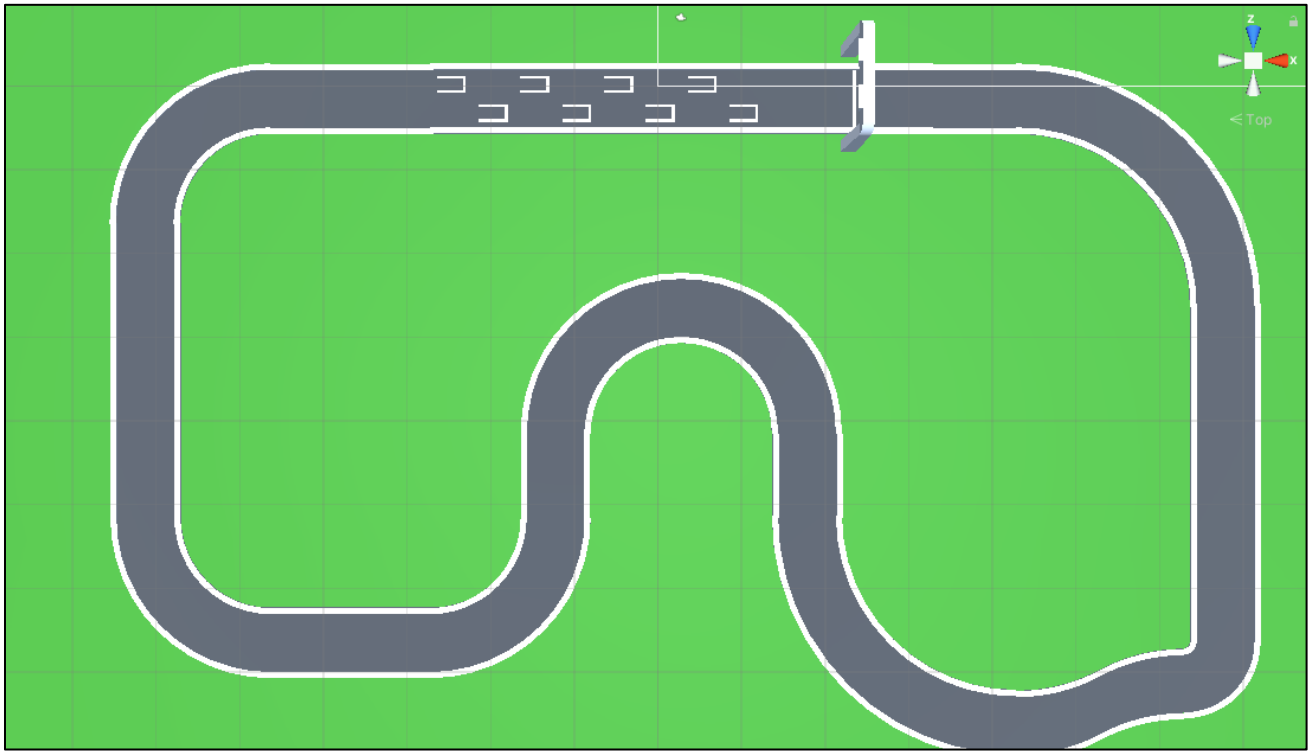

Рисунок 3.6 – Сцена Track1

Дана сцена розроблена у початковому форматі. Трек 1, на якому розміщені стартові позиції, кілька поворотів для тестування авто, смуга СТАРТ/ФІНІШ.

3.3.3. Інтерфейс та органи керування

На рисунку 3.7 продемонстровано інтерфейс органів керування, а також ігрової статистики (час кругу, найкращий час кругу, позиція гравця, кількість та значення поточного кругу).

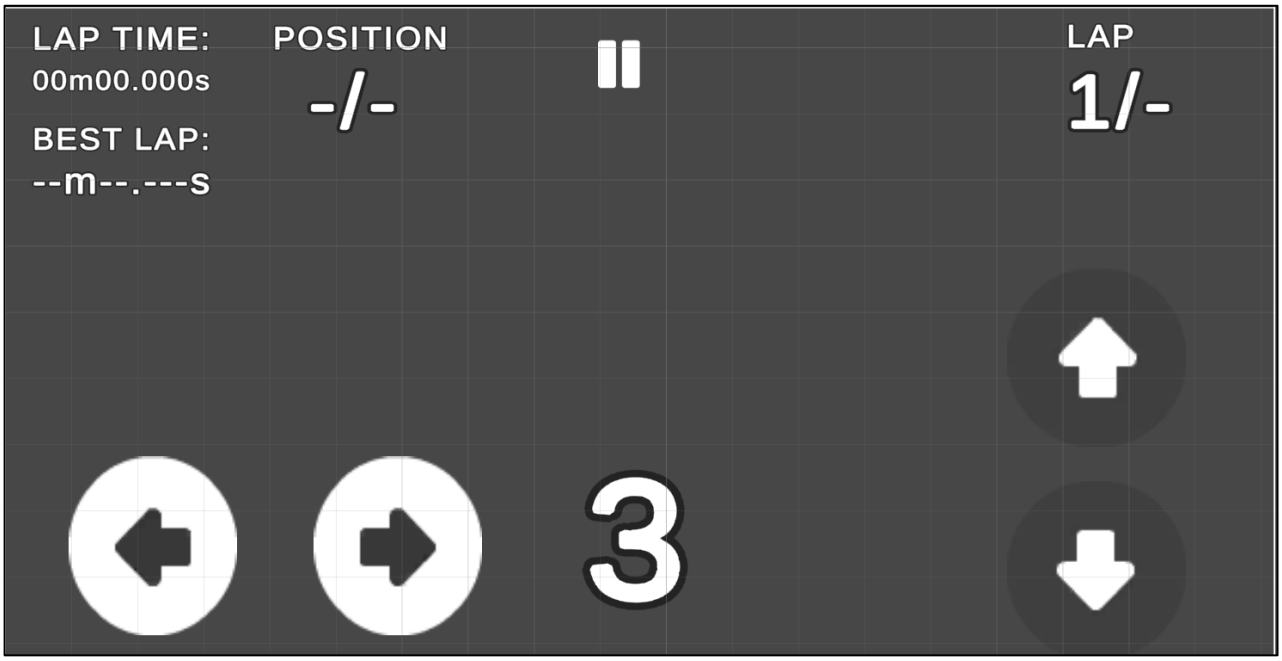

Рисунок 3.7 – Інтерфейс ігрової сцени

Даний інтерфейс не буде працювати без відповідних скриптів, тому було створено UI Manager, який оброблює всю статистику (рис. 3.8).

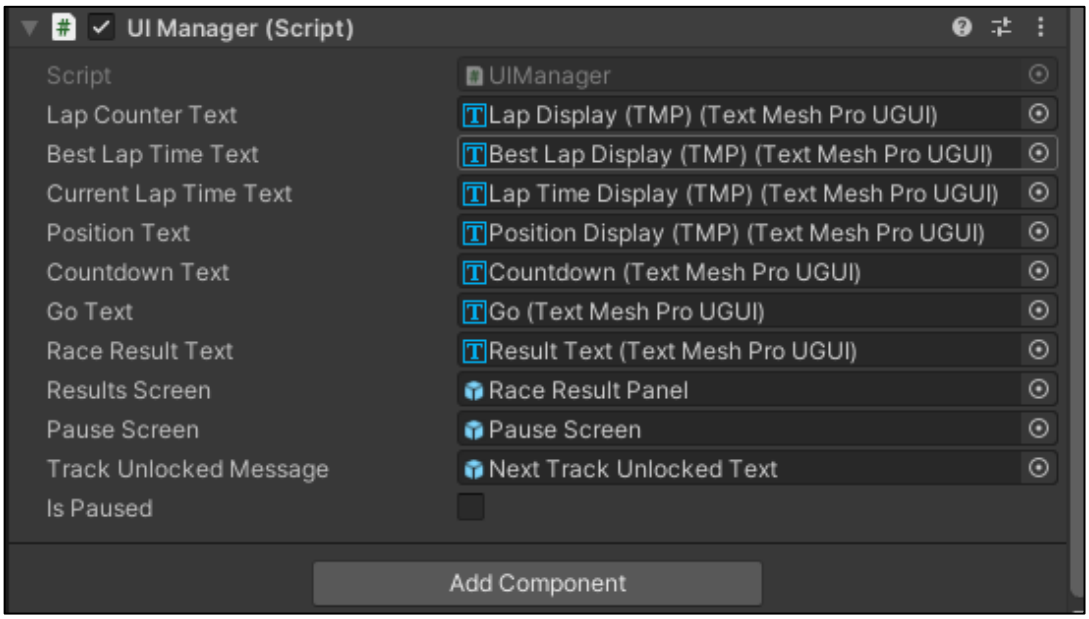

Рисунок 3.8 – Робота скрипту UI Manager

Також на рисунку інтерфейсу ігрової сцени містяться елементи керування автомобілем, у вигляді вертикальних та горизонтальних стрілок, а також кнопка «Pause». Ці органи керування входять до безкоштовного пакету ассетів Simple Input у магазині Unity AssetStore.

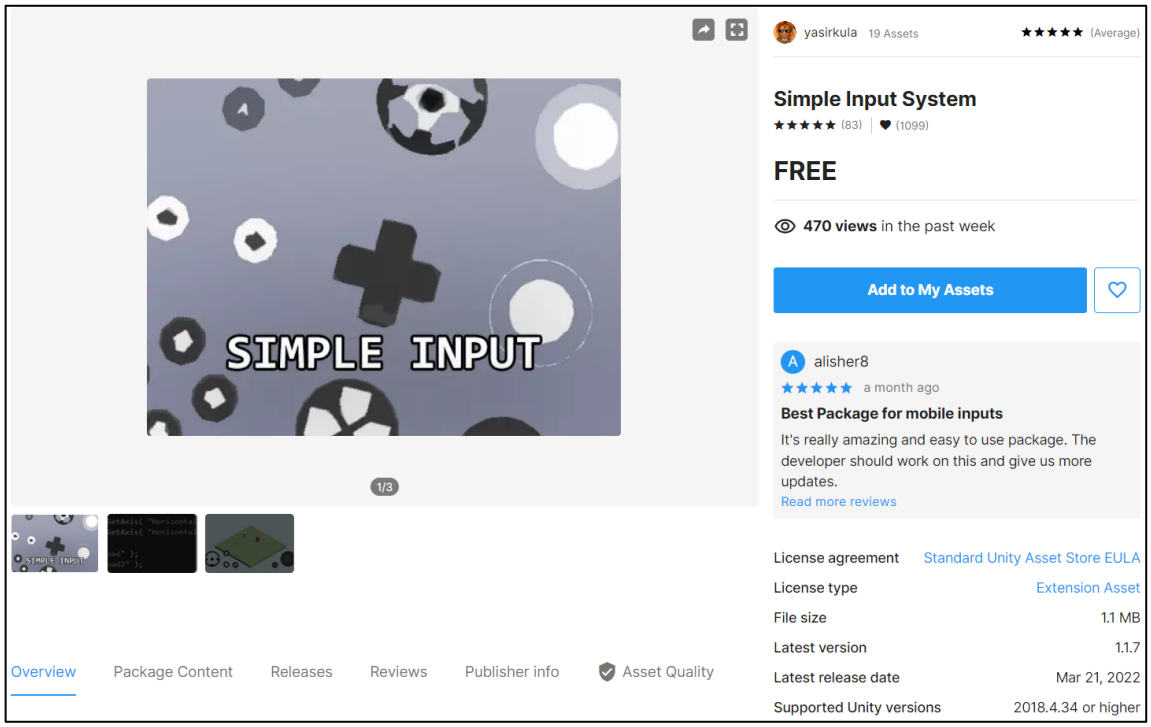

Рисунок 3.9 – Simple Input у магазині Unity AssetStore

3.3.4. Розробка CarController

Щодо управління автомобілем, написання та реалізація скрипту CarController була трудомісткою та зайняла досить багато часу.

Префаб моделі авто (рис. 3.10) включає тривимірну колізію, аудіо супроводження (звуки двигуна, зіткнення, скрегіт шин), а також дим, який формується під час дрифтингу та старту з місця.

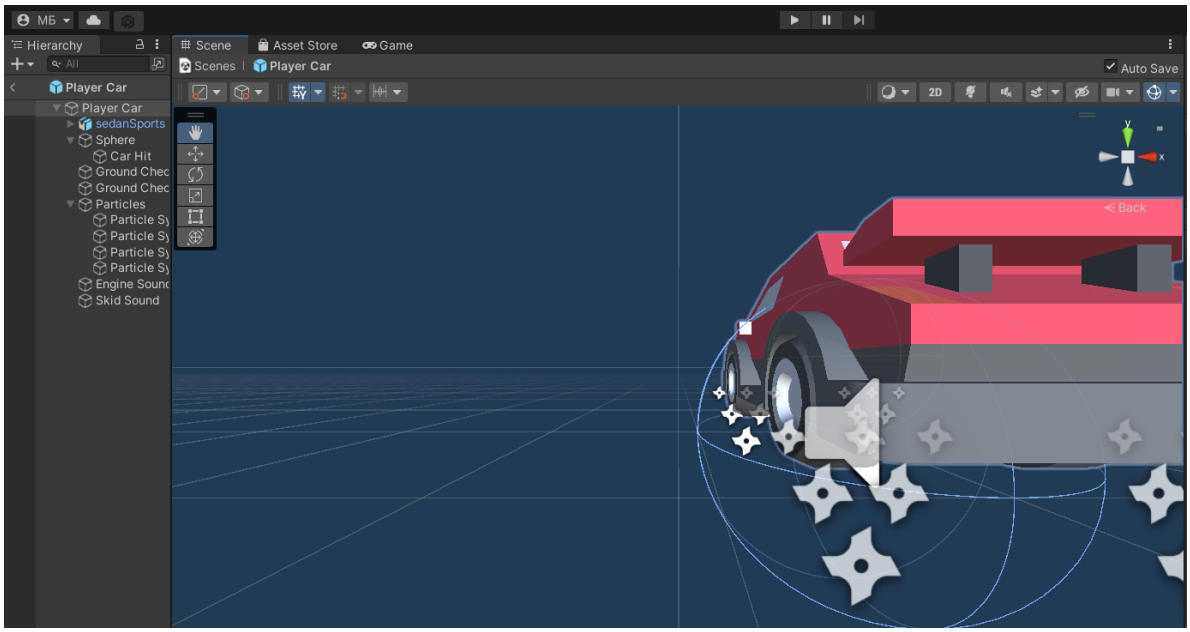

Рисунок 3.10 – Префаб автівки

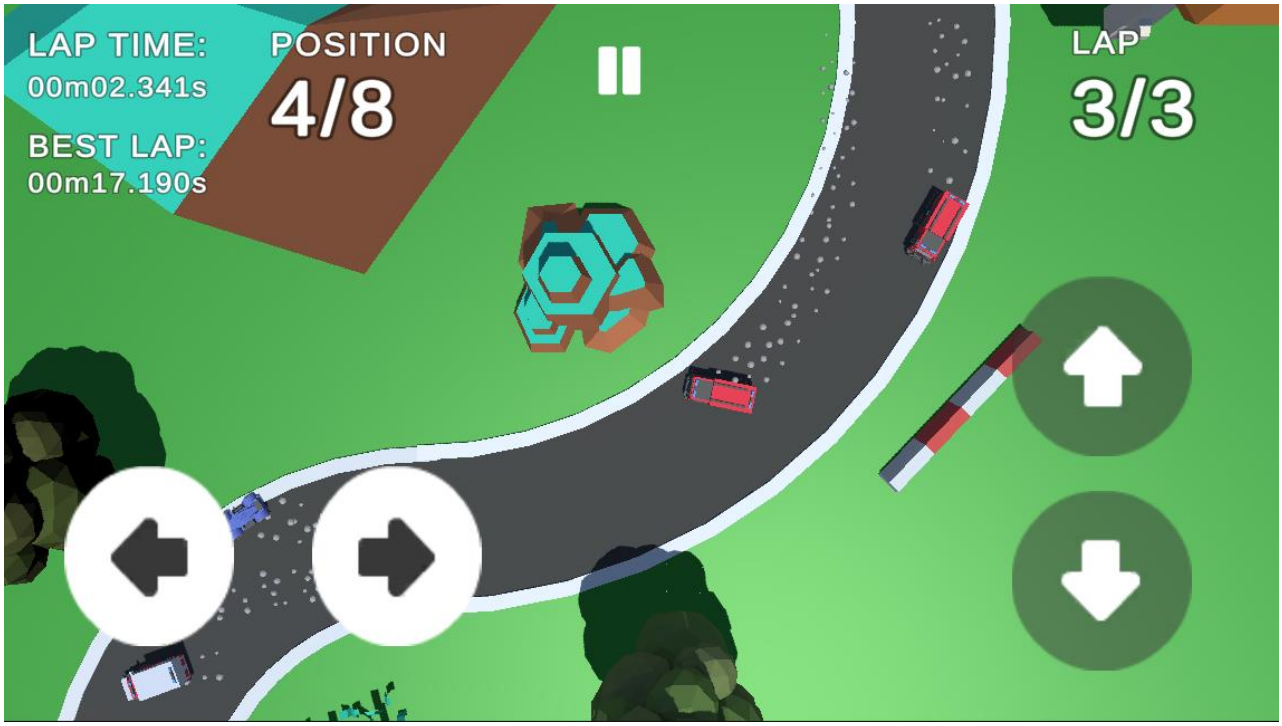

Рисунок 3.11 – Реалізація диму з коліс

Цей скрипт містить звичайний код для контролера автомобіля. У коді автомобільний об’єкт просто рухається відповідно до заданих введень. Контролер під’єднаний до органів керування (наекранних кнопок: газ, гальмо, вправо, вліво).

Функції скрипту:

— Реалізація керування автомобілем;
— Налаштування параметрами автомобіля (максимальна швидкість, максимальне прискорення, викиди диму, рівень гравітації і так далі);
— Налаштування оберту коліс навколо своєї осі (колеса змінюють свій кут, коли вони змінюють напрямок);
— Контроль та налаштування штучного інтелекту (налаштування базової максимальної швидкості, прискорення та максимальний кут повороту;
— Аудіо супроводження автомобіля.

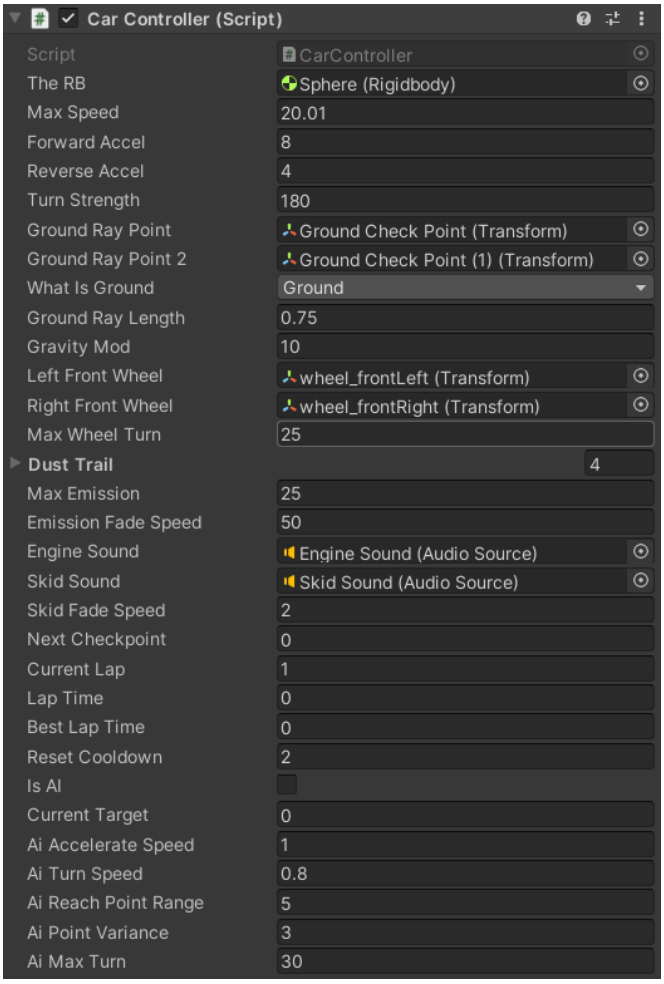

Рисунок 3.12 – Скрипт у інспекторі Unity

3.3.5. Розробка CameraController

Так як ігровий проєкт типу TopDown (вигляд зверху), потрібно налаштувати вид камери, під час проходження рівню.

Рисунок 3.13 – Вигляд ігрової сцени зі сторони гравця

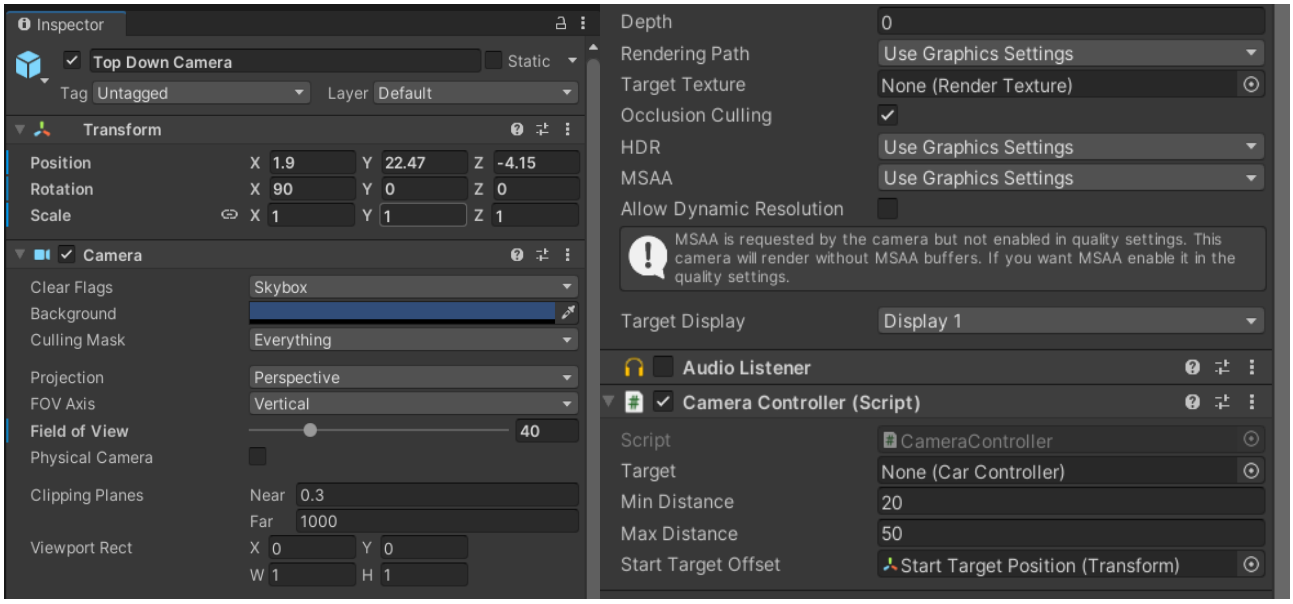

Рисунок 3.14 – Налаштування камери у інспекторі

3.3.6. Розробка RaceManager

RaceManager потрібний проєкту для керування штучним інтелектом, стартовими позиціями, кількістю опонентів на трасі, які саме опоненти будуть на початкових позиціях, а також контролю чекпоінтів.

RaceManager – це клас траси. Налаштовується відповідно до сцени траси (кількість стартових позицій, різноманітність опонентів та їх параметри за замовчуванням). Інспектор легко дозволяє оглянути його властивості.

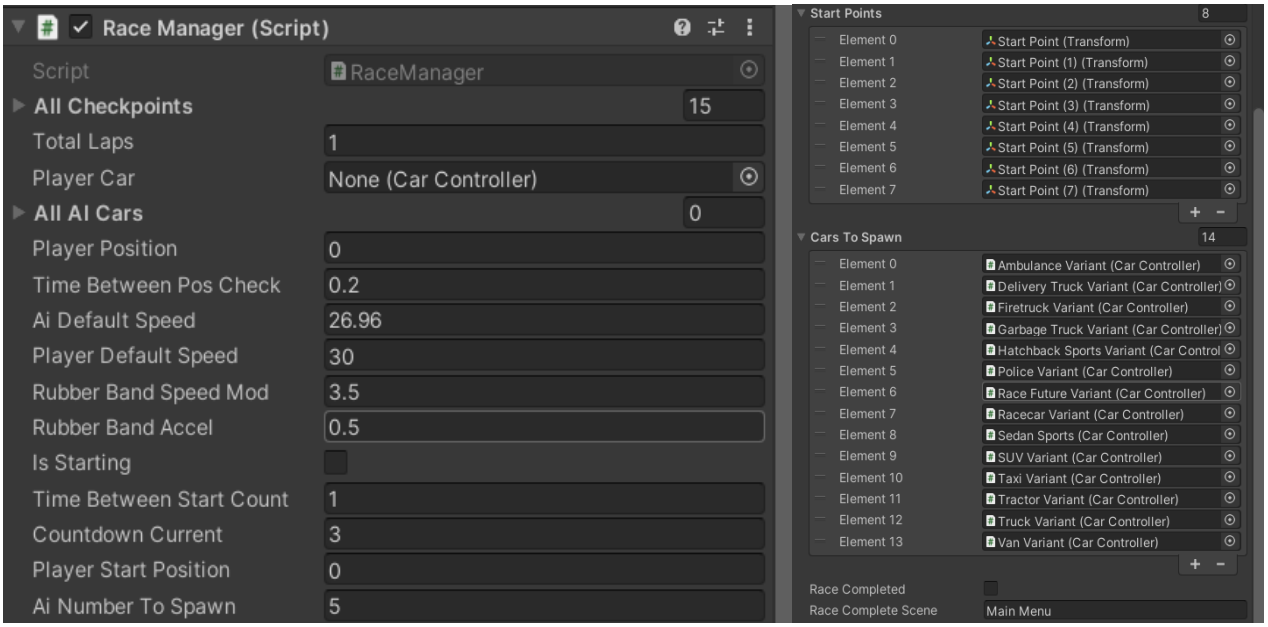

Рисунок 3.15 – RaceManager

3.3.7. Розробка RaceInfoManager

Допоміжний до RaceManager скрипт, який завантажує необхідний рівень, і якщо рівень пройдений відмінно (1 місце) – відкриває наступний. Працює у зв’язці зі скриптом Track Select Button.

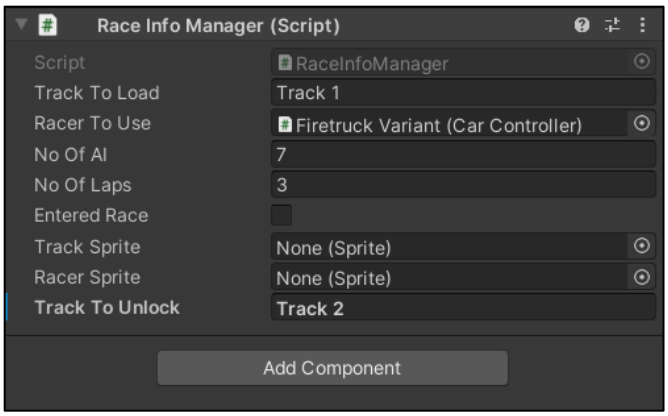

Рисунок 3.16 – RaceInfoManager

3.3.8. Розробка Checkpoints

Чекпоінти у грі представлені у виді елементів з компонентом Box Collider Їх скрипт вкрай примітивний, але тісно пов’язаний з CheckpontChecker.

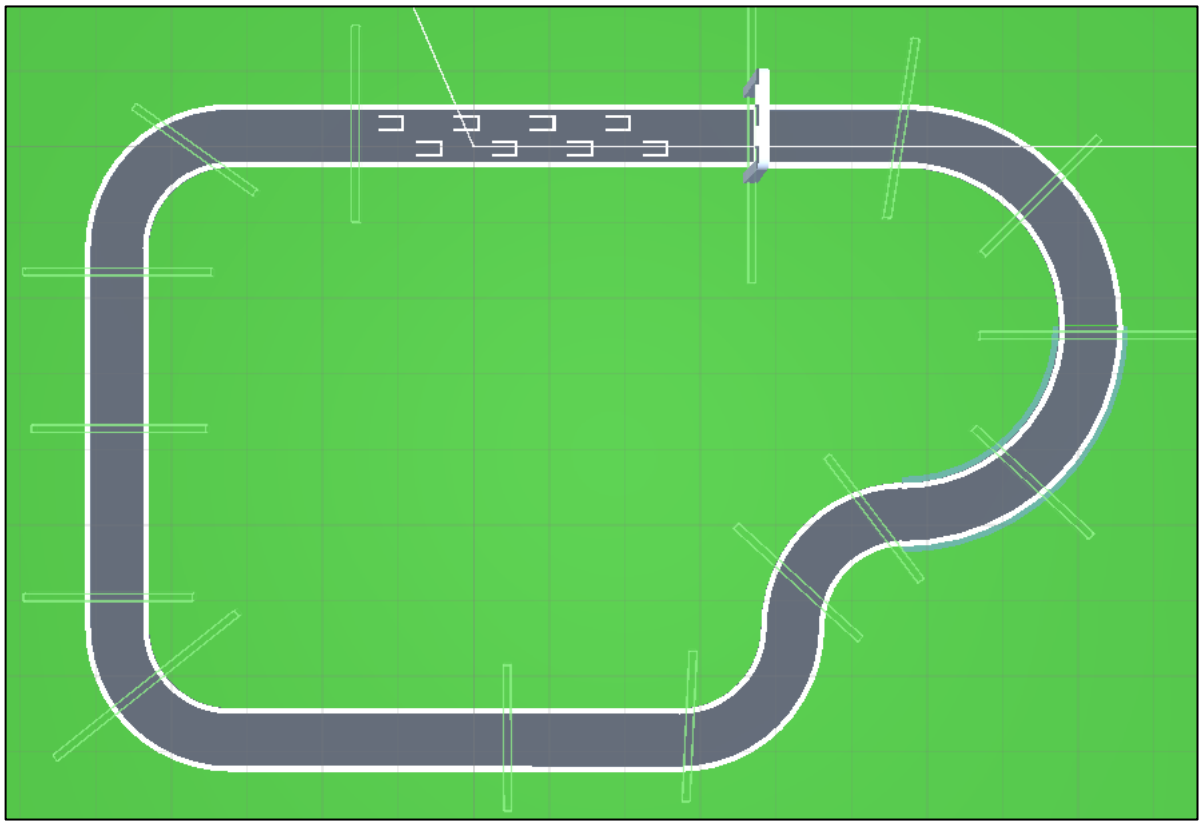

Рисунок 3.17 – Огляд Checkpoints

Також реалізований допоміжний скрипт CheckpointChecker, який перевіряє прохождення машини через контрольну точку. Працює за допомогою скрипту CarController.

3.3.9. Розробка MusicManager

MusicManager потрібен, щоб відтворювати аудіо у проєкті. Це звук двигуна, скрегіт шин, зіткнення, а також музика у головному меню. Використані аудіо ліцензовані, мають автора. Автор асоціює їх як безкоштовні у використанні у будь якому проєкті персональному чи комерційному. Автор - kenney.nl.

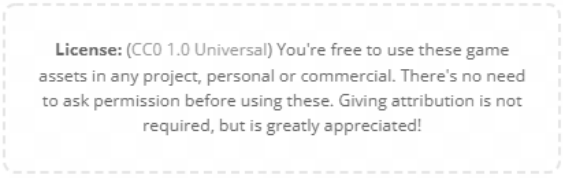

Рисунок 3.18 – Ліцензія на використання

Unity Audio Mixer дозволяє змішувати різні джерела звуку, застосовувати до них ефекти та виконувати мастеринг.

У вікні відображається Audio Mixer, який по суті є деревом груп Audio Mixer. Група Audio Mixer є сигнальним ланцюгом, який дозволяє застосовувати ослаблення гучності та корекцію висоти; він дозволяє вставляти ефекти, які обробляють звуковий сигнал і змінюють параметри ефектів. Також є механізм відправки та повернення для передачі результатів з однієї шини на іншу.

Рисунок 3.19 – AudioMixer

3.3.10. SimpleInput

Цей плагін є вдосконаленням у порівнянні зі застарілою системою введення Unity, яка дозволяє використовувати користувацькі постачальники введення, як екранні джойстики, кнопки інтерфейсу користувача та d-пади. Іншими словами, це дає змогу моделювати, наприклад Input.GetAxis, коли натискається кнопка або перетягується віртуальний джойстик.

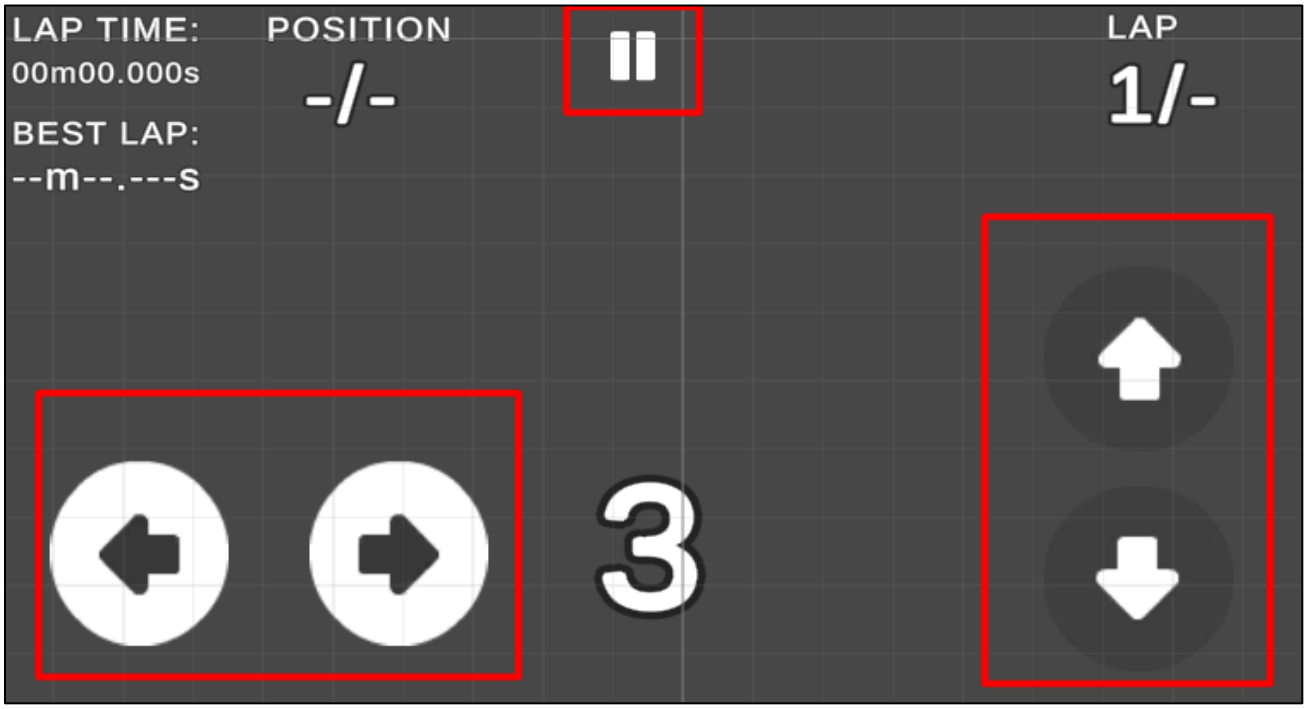

Рисунок 3.20 – SimpleInput

3.3.11. Розробка графічного оформлення

Варто прийняти до уваги, що основні етапи реалізації проєкту, а саме розробка графічного оформлення та розробка самого проєкту, проходили паралельно один одному. Це говорить про те, що для опису програмного коду необхідно наявність готових анімацій і іншого, а для перевірки працездатності самого коду достатньо стандартних примітивів Unity. Такий спосіб реалізації проєкту значно збільшує швидкість роботи.

Для виконання цього завдання були використані такі ПЗ: Adobe Photoshop, Blender, а також Unity Asset Store. У ПЗ Adobe Photoshop були виконані растрові зображення (спрайти) для меню, іконки інтерфейсу ігрового проєкту.

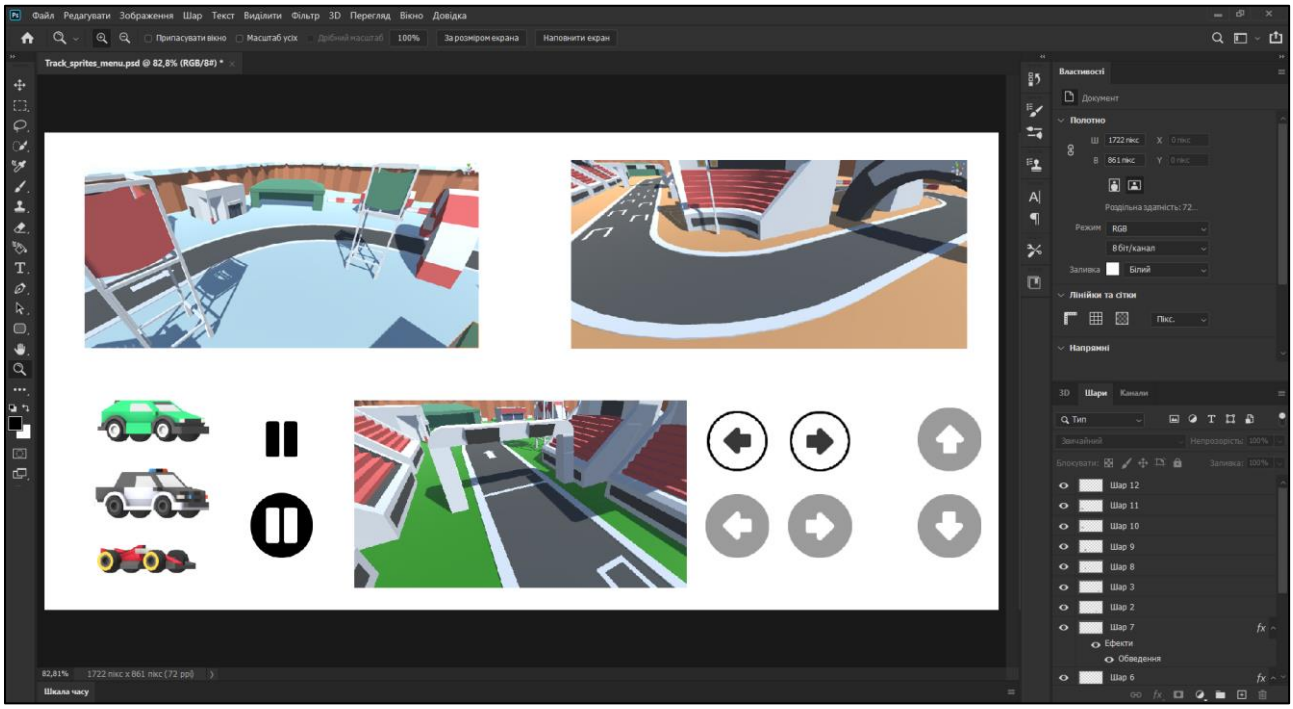

Рисунок 3.21 – Спрайти у Adobe Photoshop

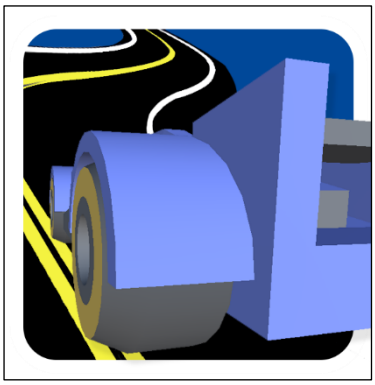

Рисунок 3.22 – Логотип (іконка) ігрового застосунку «MiniArcadeRace»

У Blender були розроблені деякі види оформлення треку такі як: дерева, огорожі, та інші види оформлення.

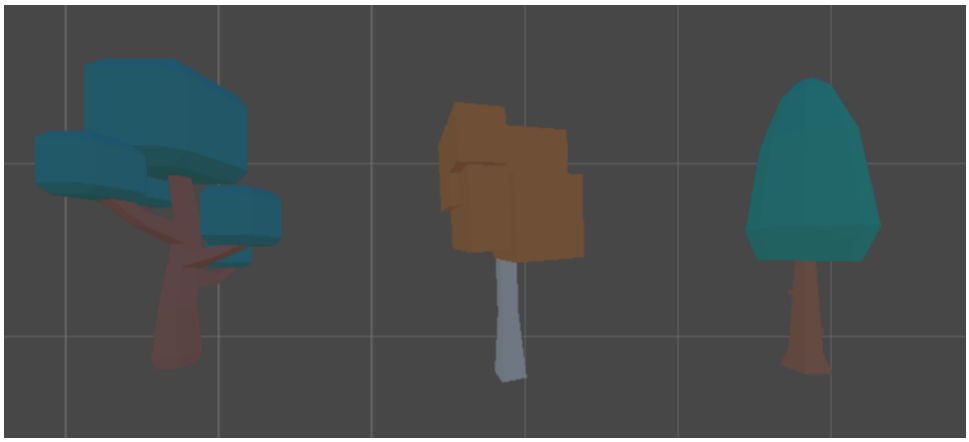

Рисунок 3.23 – Моделі елементів оточення у Blender

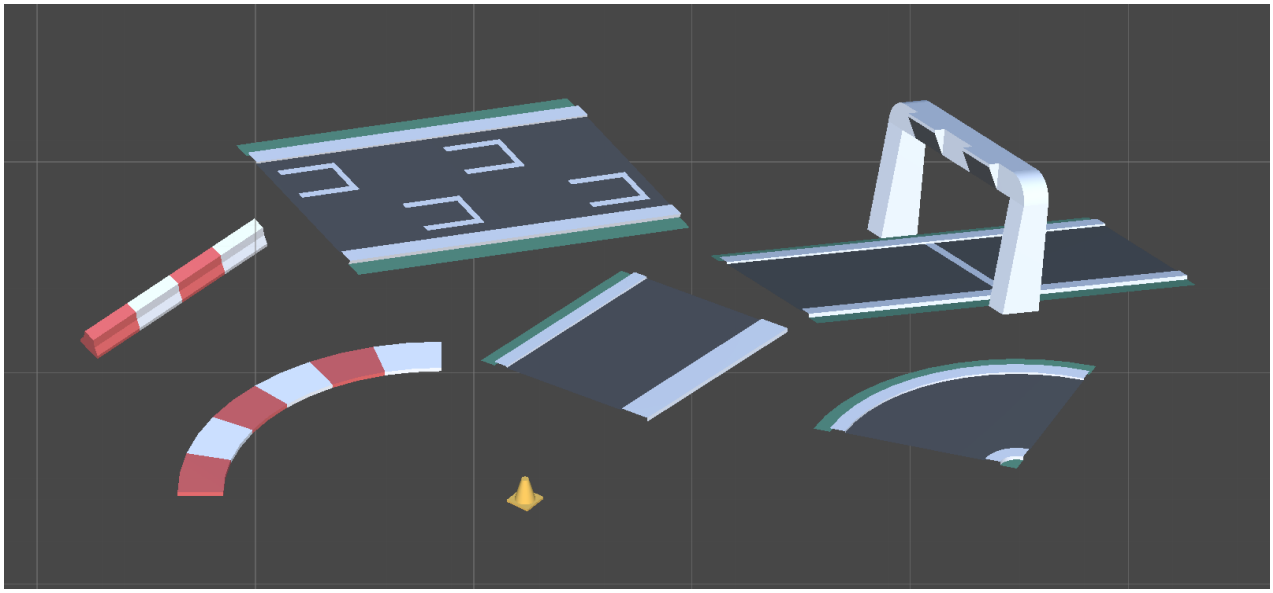

Рисунок 3.24 – 3D моделі траси у Blender

Unity Asset Store пропонує для завантаження безкоштовних kit-ассетів різних моделей. Ресурс kenney.nl, який надає ліцензію на безкоштовне використання власних ассетів, дає змогу завантажити їх та використовувати у будь якому проєкті персональному чи для комерції. На сторінці автора kenney можна знайти набір автомобілів для проєкту.

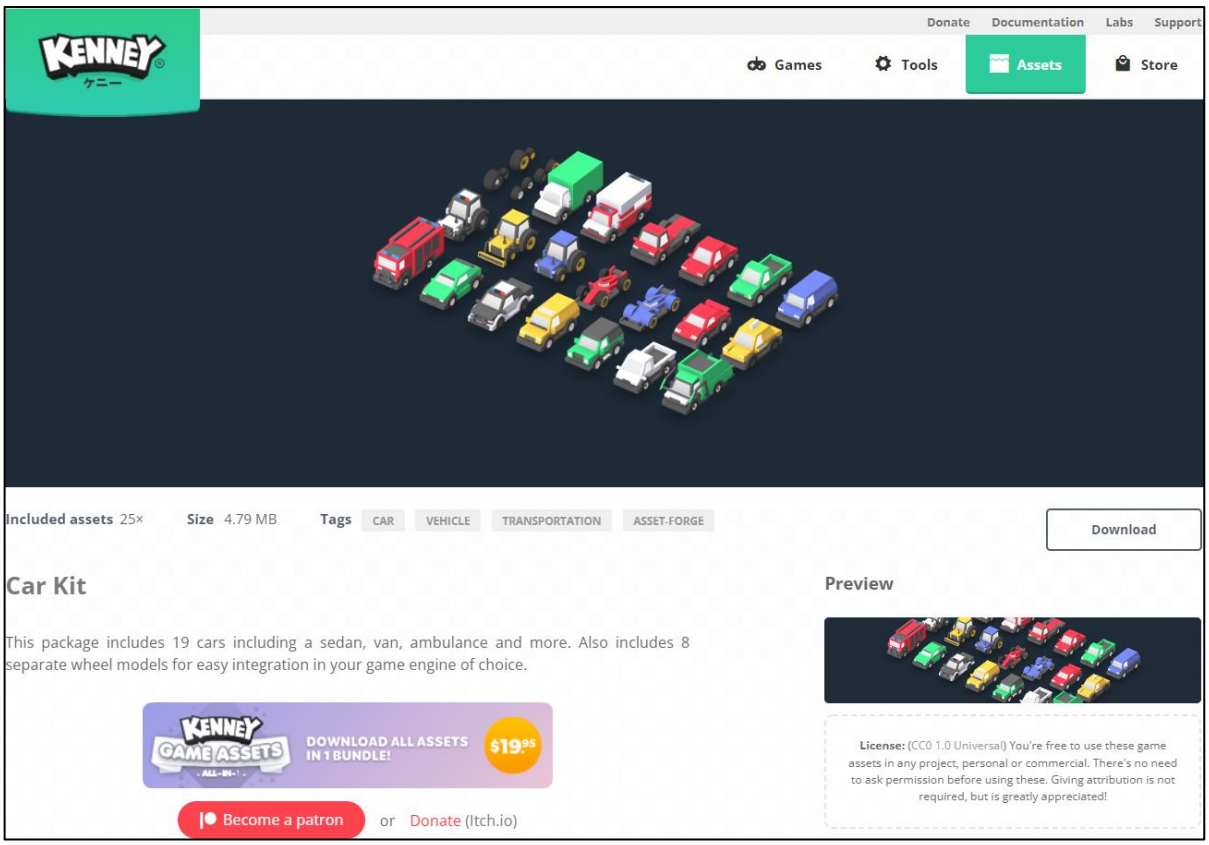

Рисунок 3.25 – Робота з готовими ассетами

3.3.12. Розробка треків для гонок

Після виконання усіх графічних елементів, можна приступити до конструювання та оформлення треків.

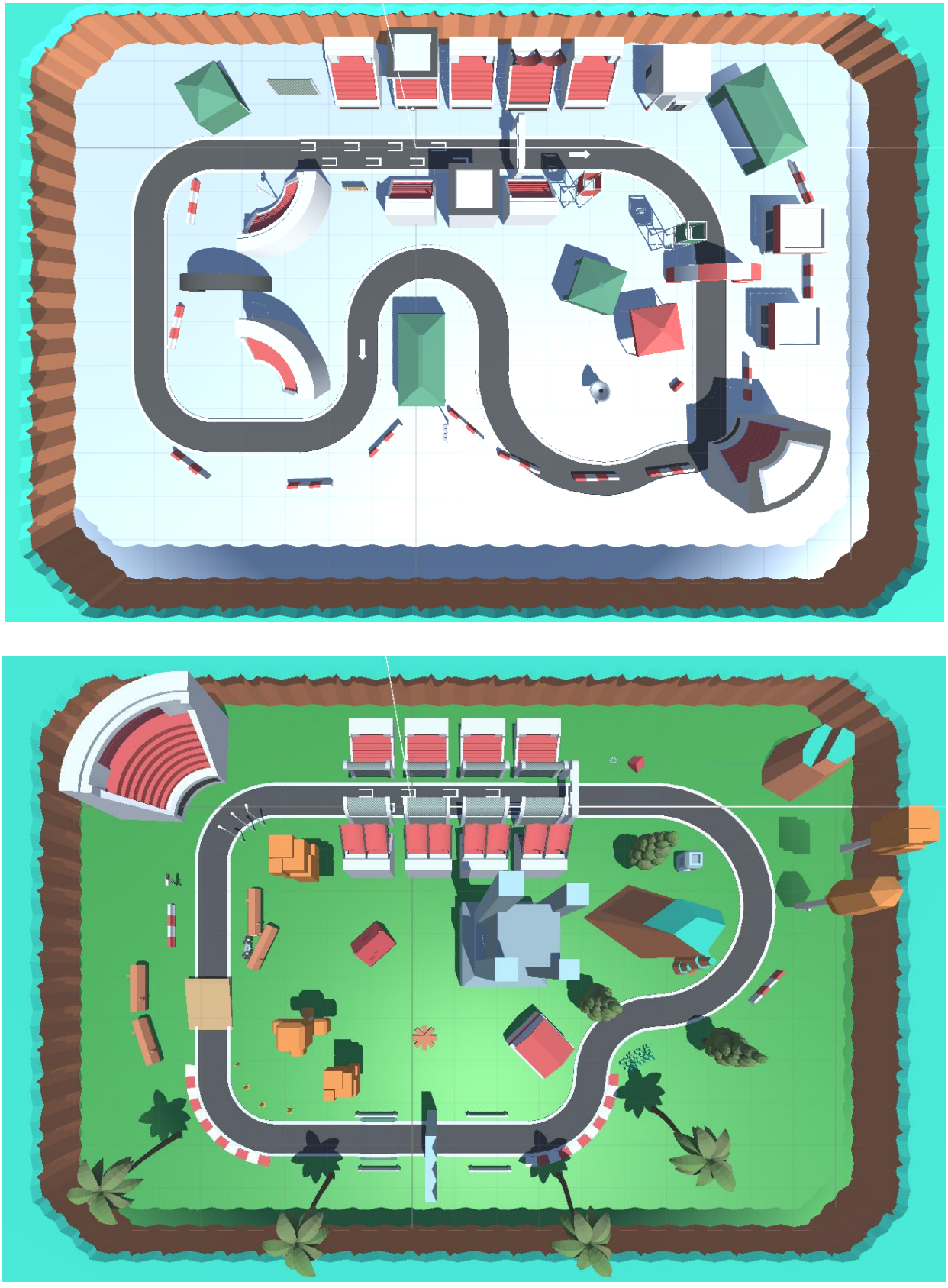

Рисунок 3.26 – Оформлення треків у Unity

Висновок до розділу 3

Під час роботи над кваліфікаційним проєктом було затверджено засоби розробки, які будуть використовуватись під час розробки проєкту, вимоги до технічного та програмного забезпечення, а також описано процес реалізації ігрового застосунку за допомогою засобів розробки.

РОЗДІЛ 4
ТЕСТУВАННЯ

4.1. Функціональне тестування

Функціональне тестування – один із видів тестування, спрямованого на перевірку відповідностей функціональних вимог ПЗ його реальним характеристикам. Основним завданням функціонального тестування є підтвердження того, що програмний продукт, який розробляється, володіє усім необхідним функціоналом. У таблиці 4.1 наведені результати функціонального тестування.

Таблиця 4.1

Результати функціонального тестування

№	Назва тесту	Очікуваний результат	Отриманий результат	Тест пройдений?
1	Відображення головного меню гри	На екрані відображається головне меню гри з кнопками «Play» та «Quit»	На екрані відобразилось головне меню гри з кнопками «Play» та «Quit»	+
2	Робота кнопки «Play»	При натиску на кнопку «Play» очікуємо перехід у наступне меню	При натиску на кнопку «Play» перейшло у наступне меню	+
3	Робота кнопки «Quit»	При натиску на кнопку «Quit» очікуємо вихід з гри	При натиску на кнопку «Quit» вийшли з гри	+
4	Відображення меню «RaceSetup»	На екрані відображається головне меню гри з вікнами вибору «Choose car» та «Choose Track»	На екрані відобразилось головне меню гри з вікнами вибору «Choose car» та «Choose Track»	+

5	Робота меню «Choose car»	Натискаємо на меню «Choose car» очікуємо перехід у меню вибору автомобілів	Натискаємо на меню «Choose car» переходимо у меню вибору автомобілів	+
6	Робота меню «Choose Track»	Натискаємо на меню «Choose truck» очікуємо перехід у меню вибору трека	Натискаємо на меню «Choose truck» переходимо у меню вибору трека	+
7	Робота кнопки «START»	При натиску на кнопку «START» очікуємо завантаження рівня гри	При натиску на кнопку «START» завантажується рівень гри	+
8	Робота MusicMixer	Ігровий проєкт супроводжується аудіо треками у головному меню та під час гри	Ігровий проєкт супроводжується аудіо треками у головному меню та під час гри	+
9	Робота SimpleInput	При натиску на стрілки керування автомобілем виконується переміщення машини по карті	При натиску на стрілки керування автомобілем виконується переміщення машини по карті	+
10	Робота інтерфейсу сцени гри	Працює відлік часу, кількість пройдених/всіх кіл, час найкращого кола, номер позиції, відлік для старту гонки	Відліку часу, кількість пройдених/всіх кіл, час найкращого кола, номер позиції, відлік для старту гонки працює	+
11	Робота RaceManager	При закінченні гонки на першому місці очікується розблокування наступного треку	При закінченні гонки на першому місці наступний трек розблокований	+

4.2. Usability Test

Usability Test – це спосіб тестування, за якого визначається зручність, зрозумілість та привабливість продукту для користувача. Виконується за допомогою залучення користувачів з метою проходження тестів, з метою отримання надалі інформації та висновків від випробувачів.

Учасникам тестування необхідно виконати такі задачі:

— почати гру;

— зупинити гру;

— продовжити гру;

— обрати машину;

— пройти трек.

Висновок до розділу 4

Всі задачі були виконані тестувальниками успішно та без проблем. Респонденти відмітили простоту керування, хороше оформлення та важкість проходження. Usability test пройдено успішно.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було вивчено сучасні інформаційні технології навчання, їх вплив на ігрову індустрію. Показані можливості ігрового рушія Unity у розробці ігор.

Додаток Unity, що є професійним ігровим рушієм, використовується у створенні відеоігор для різних платформ. Це інструмент, яким щодня користуються досвідчені розробники, а також один із найдоступніших інструментів для початківців.

При написанні коду використано мову C#. Ця мова дозволяє легко увійти до розробки гри. Мова C# актуальна на сьогоднішній день, тому що поєднує найкращі ідеї сучасних мов програмування Java, C++, Visual Basic і т. д. Через велику різноманітність синтаксичних конструкцій та можливості працювати з платформою Unity.

Для створення 2D графіки застосовано програмне забезпечення Adobe Photoshop, яке головним чином призначене для створення растрової графіки. Особливості Adobe Photoshop полягають у багатому інструментарії для операції створення і обробки зображень, високій якості обробки графічних зображень, зручності й простоті в експлуатації, широких можливостях до автоматизації обробки растрових зображень

Програмне забезпечення Blender використовувалось для моделювання 3D графіки. Його особливостями пакету є малий розмір, висока швидкість рендерингу, наявність безкоштовної версії.

Таким чином, можна зробити висновок, що використовуючи можливості вищеназваних програм, можна створювати ігрові додатки розважального характеру для різних платформ.

У ході виконання кваліфікаційної роботи бакалавра було реалізовано ігровий застосунок аркадних перегонів під мобільну систему Android на рушію Unity.

Для вирішення поставленої мети було виконано такі завдання:

— проведено аналіз актуальності теми;

— виконано аналіз існуючих аналогів;

— виконано огляд актуальних програмних середовищ для реалізації додатку;

— спроєктовано ігровий додаток;

— реалізовано ігровий додаток;

— проведено тестування ігрового застосунку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Joseph Hocking — Unity in Action. Multiplatform game development in C# with Unity 5, 2015
2. Alan Thorn — Mastering Unity Scripting, 2015
3. Alan Thorn — Unity Animation Essentials, 2015
4. Alan Thorn — How to Cheat in Unity 5: Tips and Tricks for Game Development, 2015
5. Kenny Lammers — Unity Shaders and Effects Cookbook, 2013
6. Low Poly Racing - Making Of - Ep 1 - Unity & Blender [Електронний ресурс] https://www.youtube.com/watch?v=ODVV3eUE5zM&t=451s&ab_channel=Imphenzia
7. How to create an AI Bot Race Car Controller in Unity tutorial Part 1 – Waypoints [Електронний ресурс] — https://www.youtube.com/watch?v=vsPzo7IV-THw&t=4s&ab_channel=PrettyFlyGames
8. Unity Racing Game Essentials Tutorial [Електронний ресурс] — https://www.youtube.com/watch?v=F1JRy8nFTb4&ab_channel=BMo
9. Simple Car Controller in Unity Tutorial [Електронний ресурс] — https://www.youtube.com/watch?v=Z4HA8zJhGEk&t=11s&ab_channel=GameDevChef
10. Arcade Style Car Controller || Unity Tutorial [Електронний ресурс] — https://www.youtube.com/watch?v=TBIYSksI10k&ab_channel=SpawnCampGames
11. Arcade Car Driving in Unity [Електронний ресурс] — https://www.youtube.com/watch?v=cqATTzJmFDY&t=2s&ab_channel=gamesplusjames
12. HOW TO MAKE A DRIVING & RACING GAME IN UNITY C# TUTORIAL BEGINNER/INTERMEDIATE [FULL COURSE] [Електронний ресурс] — https://www.youtube.com/watch?v=cqATTzJmFDY&t=2s&ab_channel=gamesplusjames

ДОДАТКИ

Додаток А

UIManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
public class UIManager : MonoBehaviour
{
    public static UIManager instance;
    public TMP_Text lapCounterText;
    public TMP_Text bestLapTimeText;
    public TMP_Text currentLapTimeText;
    public TMP_Text positionText;
    public TMP_Text countdownText;
    public TMP_Text goText;
    public TMP_Text raceResultText;
    public GameObject resultsScreen;
    public GameObject pauseScreen;
    public GameObject trackUnlockedMessage;
    public bool isPaused;
    void Awake()
    {
        instance = this;
    }
    void Start()
    {
    }
    void Update()
    {
    }
    public void PauseUnpause()
    {
        isPaused = !isPaused;
        pauseScreen.SetActive(isPaused);
        if (isPaused)
        {
            Time.timeScale = 0f;
        }
        else
        {
            Time.timeScale = 1f;
        }
    }
```

```
    public void ExitRace()
    {
        Time.timeScale = 1f;
        RaceManager.instance.ExitRace();
    }
    public void QuitGame()
    {
        Application.Quit();
        Debug.Log("Game Quit");
    }
}
```

Додаток Б

CameraController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraController : MonoBehaviour
{
    public CarController target;

    Vector3 offsetDir;

    public float minDistance = 20f;

    public float maxDistance = 50f;

    float activeDistance;

    public Transform startTargetOffset;

    void Start()
    {
        offsetDir = transform.position - startTargetOffset.transform.position;
        activeDistance = minDistance;
        offsetDir.Normalize();
    }

    void Update()
    {
        activeDistance = minDistance + ((maxDistance - minDistance) *
(target.theRB.velocity.magnitude / target.maxSpeed));
        transform.position = target.transform.position + (offsetDir *
activeDistance);
    }
}
```

Додаток С

RaceInfoManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class RaceInfoManager : MonoBehaviour
{
    public static RaceInfoManager instance;
    public string trackToLoad;
    public CarController racerToUse;
    public int noOfAI;
    public int noOfLaps;

    public bool enteredRace;
    public Sprite trackSprite;
    public Sprite racerSprite;

    public string trackToUnlock;

    void Awake()
    {
        if (instance == null)
        {
            instance = this;
            DontDestroyOnLoad(gameObject);
        }
        else
        {
            Destroy(gameObject);
        }
    }
}
```

Додаток Е

Checkpoint.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Checkpoint : MonoBehaviour
{
    public int cpNumber;
}
```

Додаток Г

CheckpointChecker.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CheckpointChecker : MonoBehaviour
{
    public CarController theCar;

    void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Checkpoint")
        {
            //Debug.Log("Hit cp " + other.GetComponent<Checkpoint>().cpNumber);
            theCar.CheckpointHit(other.GetComponent<Checkpoint>().cpNumber);
        }
    }
}
```

Додаток Д

MusicManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MusicManager : MonoBehaviour
{
    public AudioSource musicPlayer;
    public AudioClip[] potentialMusic;

    void Start()
    {
        musicPlayer.clip = potentialMusic[Random.Range(0,
potentialMusic.Length)];
        musicPlayer.Play();
    }
}
```