

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБЛЕННЯ ЗАСТОСУНКУ ДЛЯ ФІЗИЧНОЇ РЕАБІЛІТАЦІЇ
ПАЦІЄНТІВ З ВИКОРИСТАННЯМ OCULUS RIFT
APPLICATION DEVELOPMENT FOR PHYSICAL REHABILITATION USING
OCULUS RIFT

Виконав: студент(ка) 4 курсу, групи КНТ-227
Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Трунова Д.Д.

(прізвище та ініціали)

Керівник Табунщик Г.В.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 122 Комп'ютерні науки
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Трунової Дар'ї Дмитрівни
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розроблення застосунку для фізичної реабілітації з використанням Oculus Rift. Application Development For Physical Rehabilitation Using Oculus Rift

керівник проєкту (роботи) Табунщик Галина Володимирівна, к.т.н., професор,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) _____

3. Вихідні дані до проєкту (роботи) _____ рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області та стану проблеми. 2. Огляд технологій Oculus. 3. Проектування програмного забезпечення. 4. Конструювання програмного забезпечення.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Табунщик Г.В., професор		
Нормоконтроль	Камінська Ж.К., асистент		

7. Дата видачі завдання “ 28 ” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Трунова Д.Д.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Табунщик Г.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 69 с., 36 рис., 2 табл., 2 додатків, 11 джерел.

ФІЗИЧНА РЕАБІЛІТАЦІЯ, ПЗ, СИСТЕМА, ІГРОВИЙ ОБ'ЄКТ, КОМПОНЕНТ, UNITY, ПРЕФАБ.

Об'єктом дослідження є процеси фізичної реабілітації людини за допомогою програмного застосунку під шолом віртуальної реальності Oculus Rift.

Метою даної роботи є розробка та програмна реалізація реабілітаційного застосунку, що дозволяє слідкувати за правильністю виконання вправ пацієнтом.

Для досягнення мети, в роботі були вирішені наступні задачі:

- розглянуто існуючі методи реабілітації людини після тяжких захворювань;
- проаналізовані методи фізичної реабілітації;
- розроблено віртуальне середовище програми;
- реалізовано систему пересування по віртуальному середовищу, та взаємодії з предметами;
- розроблено методи відстеження дій користувача.

В результаті роботи здійснена програмна реалізація реабілітаційної системи в середовищі Unity, що призначена для фізичної реабілітації людини після тяжких хвороб та травм.

Дана розробка призначена для застосування в медичній галузі реабілітаційного напрямку.

ABSTRACT

Explanatory note to the final qualifying work of the bachelor: 60 pages, 22 figures, 2 tables, 2 appendixes, 11 sources.

PHYSICAL REHABILITATION, SOFTWARE APPLICATION, SYSTEM, GAME OBJECT, COMPONENT, UNITY, PREFAB.

The object of the research is the processes of physical rehabilitation of a person with the help of a software application using Oculus Rift virtual reality helmet.

The purpose of this work is the development and software implementation of a rehabilitation application that allows to monitor the correctness of the exercises performed by the patient.

To achieve this goal, the following tasks were solved in the work:

- the existing methods of rehabilitation of the person after serious diseases are considered;
- analyzed methods of physical rehabilitation;
- implemented a system of movement in the virtual environment, and interaction with objects;
- the methods for tracking user actions were developed.

As a result, the software implementation of the rehabilitation system in the Unity environment, which is designed for physical rehabilitation of a person after serious illness and injury.

This development is intended for use in the medical field of rehabilitation.

ЗМІСТ

	С.
ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	7
ВСТУП.....	8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА СТАНУ ПРОБЛЕМИ	9
1.1 Існуючі види методів реабілітування.....	9
1.2 Методи медичної та фізичної терапії.....	10
1.3 Фізична реабілітація за допомогою віртуальної реальності	14
2 ОГЛЯД ТЕХНОЛОГІЙ OCULUS.....	19
2.1 Шолом віртуальної реальності Oculus Rift	19
2.2 Рекомендовані характеристики комп'ютера.....	21
2.3 Програмне забезпечення для використання Oculus Rift	22
3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	27
3.1 Вибір мови програмування	27
3.2 Графічні рішення Unity 3D	29
3.3 Огляд та налаштування середовища розробки Unity3d	31
3.4 Концепція додатку фізичної реабілітації.	37
4 КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	39
4.1 Алгоритми функціонування	39
4.2 Сценарії використання додатку.....	40
4.3 Опис особливостей програмної реалізації.	42
ВИСНОВКИ	49
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	50
ДОДАТОК А Текст програми	52
ДОДАТОК Б Слайди презентації	63

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

HDRP – High-Definition Render Pipeline

LWRP – Lightweight Render Pipeline

PC – Personal Computer

RAM – Random Access Memory

UI – User Interface

VR – Virtual Reality

ДЦП – Дитячий церебральний параліч

ЛФК – Лікувальна фізкультура

ПЗ – Програмний застосунок

ВСТУП

В усі часи діяльність людини супроводжувалась несприятливими факторами, що впливають на фізичний та психологічний стан.

Через це у світі, з кожним роком, збільшується кількість людей, що потребує медичної, фізичної, психічної або соціально-культурної реабілітації. Сучасні методи медикаментозної терапії часто дають лише тимчасову дію та мають величезну кількість побічних ефектів, які можуть погіршити стан людини. Таким чином розробка методів реабілітації, які не мають фармакологічний характер є особливо актуальною.

Частіше за все медикаментозний та фізичний(фізіотерапія) методи реабілітації застосовуються одночасно. Дослідження показують, що такий підхід значно покращує та прискорює результати.

Одним із найновіших методів є фізична реабілітація за допомогою віртуальної реальності. Цей метод може включати в себе інші методи фізичної реабілітації, як наприклад лікувальна фізкультура. Адже за допомогою датчиків, що знаходяться в контроллерах та шоломі віртуальної реальності можна відстежити дії користувача.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА СТАНУ ПРОБЛЕМИ

1.1 Існуючі види методів реабілітування.

В наш час розвинених технологій існує безліч методів реабілітування людини після хвороб та травм. Найбільш відомим методами є медикаментозна терапія, лікувальна фізкультура, фізіотерапія, тощо. В різних ситуаціях вони мають позитивні а також негативні наслідки.

Методи реабілітаційної терапії можна розділити за такими напрямками [1]:

- медичний та фізичний;
- психо-патологічний;
- соціально культурний.

Перший та основний напрямок реабілітації (медичної та фізичної) - відновлення здоров'я хворого за допомогою комплексного використання різних засобів, спрямованих на максимальне відновлення порушених фізіологічних функцій організму, а у разі неможливості досягнення цього - розвиток компенсаторних і замісних пристосувань (функцій). До медичних методів реабілітації відносяться: фізіотерапія (ультразвук, магнітотерапія, лікарський електрофорез, лазеротерапія, мікрополяризації, електролікування); лікувальна фізкультура з застосуванням надувних предметів, тренажера Гросса, лікувальних костюмів, "Фастон"; медичний масаж; медикаментозна терапія.

Психологічний аспект реабілітації спрямований на корекцію психічного стану пацієнта, а також формування його відносин до лікування, лікарських рекомендацій, виконання реабілітаційних заходів. Необхідно створити умови для психологічної адаптації хворого до нової внаслідок хвороби життєвої ситуації [1].

Соціально-культурна реабілітація направлена на те, щоб повернути потерпілому соціальну повноцінність. Ці завдання вирішуються не тільки медичними установами, а й органами соціального забезпечення. Можуть

організовуватися різні святкові та розважальні заходи, відвідування театрів, музеїв, художніх виставок, тощо.

1.2 Методи медичної та фізичної терапії

1.2.1 Лікувальна фізкультура

Лікувальна фізкультура - це комплекс вправ для реабілітації при різних захворюваннях а також для профілактики цих захворювань. ЛФК не має вікових протипоказів [2].

Правильно підібрані вправи можуть і повинні застосовуватися в лікуванні найрізноманітніших захворювань та ушкоджень. При їх призначенні лікар враховує особливості недуги, характер, ступінь і стадію хворобливого процесу в системах та органах. В основі лікувальної дії фізичних вправ лежать суворо дозовані навантаження стосовно хворих і ослаблених [2].

Самі фізичні вправи можуть бути загальними та спеціальними.

Загальні вправи мають вплив що проявляється в активізації всіх функцій організму, що сприяє одужанню, поліпшує емоційний стан, попереджає ускладнення, зменшує несприятливі наслідки гіпокінезії (зниження рухової активності) під час хвороби. Загальний вплив є неспецифічним, тому різні фізичні вправи для різних м'язових груп можуть позитивно впливати на організм, і ті самі вправи можуть бути ефективними при різних захворюваннях.

Спеціальні вправи у деяких випадках можуть мати специфічну дію на патологічний процес. Наприклад, при атрофії м'язів внаслідок іммобілізації кінцівки, спеціальні вправи, що залучають до рухів відповідні м'язи, відновлюють їхню структуру та обмір речовин.

Головними завданнями лікувальної фізичної культури є [3]:

- стимуляція потенційних захисних сил організму в боротьбі із захворюванням;

- збереження та підтримка хворого організму в найкращому функціональному стані;
- прискорення ліквідації місцевих анатомічних та функціональних проявів хвороби;
- попередження ускладнень;
- відновлення функціональної повноцінності людини за короткий термін [3].

1.2.2 Фізіотерапія

Фізіотерапія - це спосіб лікування різних захворювань за допомогою різних фізичних факторів, таких як тепло, холод, електрострум, магнітне УФ та ІЧ-випромінювання, а також за допомогою лікувальних грязей, масажу, гірудотерапії, тощо [4].

Завдяки одній тільки фізіотерапії складно досягти повного лікування, вона має видимий ефект тільки на перших етапах розвитку захворювання, однак лікарі радять використовувати фізіотерапію як підмога в боротьбі з самими різними захворюваннями. Крім того, практика показує, що фізіотерапевтичні процедури можуть використовуватися в якості профілактики хвороб, оскільки допомагають зміцнити імунітет і активізують важливі біохімічні процеси, що відбуваються в організмі.

Фізіопроцедури мають наступні переваги:

- є безболісними, а ризик розвитку побічних ефектів мінімальний;
- застосування фізіопроцедур на тлі прийому лікарських засобів посилює ефект препаратів, що дозволяє скоротити термін прийому ліків, тим самим зменшуючи медикаментозне навантаження на організм;
- покращують кровопостачання органів і стимулюють місцевий і загальний імунітет;
- мають тривалий терапевтичний ефект, збільшують терміни ремісії хронічних захворювань;

- при відсутності протипоказань є безпечним методом лікування.

Фізіотерапія може бути корисною для людей різного віку та самими різноманітними проблемами зі здоров'ям, наприклад:

- проблеми з кістками, суглобами та м'якими тканинами (болі в спині, плечах, спортивні травми);
- проблеми що впливають на мозок та нервову систему (проблеми з рухом в результаті інсульту, розсіяного склерозу (РС), або хвороби Паркінсона);
- проблеми із серцем та циркуляцією (серцевий напад);
- проблеми з легеньми та диханням (хронічна обструктивна хвороба легень (ХОЗЛ) та муковісцидоз).

Серед розповсюджених процедур фізіотерапії найчастіше зустрічаються наступні [5]:

- TENS-терапія - транскутанна (та, що здійснюється через шкіру) електрична нервова стимуляція - застосовується для відновлення скорочувальної здатності постраждалих нервів та м'язів, усунення болей неврологічного або посттравматичного походження;
- магнітотерапія - метод фізіотерапії, заснований на дії магнітного поля. Стимулює кровопостачання тканин, сприяє зменшенню набряків, хворобливих відчуттів. Використовується при лікуванні наслідків травм, неврологічних та кардіологічних захворювань, захворювань опорно-моторного апарату. Дозволяє підвищити рівень фізичних навантажень;
- міостимуляція - вплив на м'язи слабким імпульсним током, що викликає їх скорочення. Досягається укріплення мускулатури при відсутності активних фізичних навантажень. Відновлюється тонус тканини, поліпшується обмін речовин, знімається м'язове перенапруження;
- імпульсна короткохвильова терапія - допомагає в період посттравматичних станів (при розтягненні, вивихах, ушибах, переломах, гематомах, подряпинах, тощо), сприяє загоєнню ран, часто застосовується після операцій на щелепах і зв'язках та при деяких типах запалення. Дана

терапія також назначається пацієнтам при порушеннях периферичного кровообігу;

- СМТ-терапія (ампліпульстерапія) - різновид електротерапії, при якому вплив на організм здійснюється за допомогою амплітудних пульсацій змінних середньочастотних струмів (частотою від 2 до 5 кГц). Покращує лімфо і кровообіг в ураженій області, сприяє зниженню больових відчуттів;

- лазеротерапія - вплив за допомогою направленого світлового потоку (лазера). В результаті в організмі активізуються найважливіші біохімічні процеси, оновлюються тканини, відновлюється їх життєдіяльність, включаються механізми саморегуляції;

- ультразвукова терапія;

- фонофорез - поєднання ультразвукового впливу з дією лікарських або косметичних засобів. Ультразвук підвищує ефективність місцевого застосування лікарських препаратів;

- електрофорез - введення лікарських препаратів за допомогою постійного електричного струму;

- УВЧ-терапія - фізіотерапія за допомогою ультрависокочастотного електромагнітного поля. Використовується для лікування запалень, травм, радикуліту, невралгії;

- УФ-терапія - вплив на організм ультрафіолетовим випромінюванням. Підвищує опірність організму інфекціям, активізує діяльність ендокринної системи і обмін речовин. Використовується для заповнення дефіциту вітаміну D, лікування дерматологічних захворювань.

1.2.3 Реабілітаційний масаж

Оздоровлення дисфункцій та повернення пацієнтів до активного життя - це основна увага реабілітологів та реабілітаційних масажистів.

Техніка реабілітаційного масажу впливає на організм через шкіру, дихання, клітини та органи, зменшуючи біль, покращуючи роботу

травмованих органів та підвищуючи опірність організму. Терапевтичний масаж може відновити м'язову силу, розширити діапазон рухів, поліпшити витривалість та стійкість до щоденних побутових та робочих навантажень.

Медичний масаж фокусується на лікуванні певних ділянок тіла, пов'язаних з діагнозом. Під час медичного масажу використовуються різні способи, такі як глибокий тканинний масаж, терапія триггерною точкою, нервово-м'язова терапія, вивільнення міофасциальної тканини, сучасна баночна терапія, гідротерапія тощо.

Лікувальний масаж - це ручна маніпуляція м'якими тканинами тіла (м'язи, сполучна тканина, сухожилля та зв'язки) для зміцнення здоров'я та добробуту людини. Люди звертаються до масажної терапії з різних причин - для зменшення стресу та тривоги, розслаблення м'язів, зменшення болю, зміцнення загального стану здоров'я та реабілітації травм. Метою реабілітаційного/медичного масажу є сприяння створенню умов для повернення організму до "нормального стану здоров'я" після травми.

1.3 Фізична реабілітація за допомогою віртуальної реальності

Віртуальна реальність (VR) - це комп'ютерне середовище зі сценами та об'єктами, які видаються справжніми, завдяки чому користувач відчуває, що занурений у своє оточення. Це середовище сприймається через пристрій, відомий як гарнітура або шолом віртуальної реальності. VR дозволяє нам зануритися у відеоігри так, ніби ми є одним із персонажів, навчитися робити операції на серці або покращити якість спортивних тренувань, щоб максимізувати продуктивність.

Використання вже існуючих методів фізичної реабілітації, але з впровадженими сучасними технологіями, значно прискорює та покращує результати терапій. Так нещодавно Віртуальна Реальність стала вагомим доповненням до звичайної терапії шляхом включення реабілітаційних стратегій у новий та недорогий підхід. Терапія на основі віртуальної

реальності може забезпечити позитивний досвід навчання, бути привабливою та мотивуючою.

За допомогою терапії, заснованої на VR, вправи (завдання) можуть бути адаптовані до потреб пацієнтів за допомогою імітації або відеоігор. Перевага віртуальної реальності полягає в тому, що можливості по суті безмежні. Віртуальне середовище можна налаштувати, розробляючи завдання, які відповідають когнітивним та фізичним вадам людини, що є критично важливим для максимізації реорганізації мозку та реактивації тих областей мозку, що беруть участь у плануванні, навчанні та виконанні рухів, а також для підтримки взаємодії.

VR-опосередкована терапія також дала значні поліпшення реабілітації ходьби після інсульту. VR терапії для тренування ходьби часто включають системи тренувань на біговій доріжці в поєднанні з екраном або пристроєм, встановленим на голові (шоломом), для створення захоплюючого віртуального середовища. Додаткові біосенсори, такі як інерційні вимірювальні прилади (ІМС), датчики сили та ЕМГ-датчики, також використовуються для відстеження прогресу в кінематиці пацієнтів, динаміці руху та активації м'язів. Візуалізація цих параметрів у реальному часі дозволяє терапевтам надавати пацієнтам своєчасний зворотний зв'язок про хід та якість виконуваних ними завдань, надаючи їм можливість виявити та виправити можливі помилки.

Після інсульту людині потрібне відновлення, яке, крім усього, поверне і дрібну моторику. Спеціальні ігри мотивують людину виконувати повсякденні дії: підняти руку, взяти склянку, поставити її назад. Очі бачать картинку, на якій віртуальна рука піднімається догори, а мозок «підтягує» реальну руку до такої висоти. У такий спосіб поліпшується моторика рук, координація тіла і підвищується рухова активність.

Таким чином із появою Віртуальної Реальності, почалась нова сторінка у дослідженні методів реабілітації хворих після хвороб та травм.

1.4 Програмні аналоги с реалізованим методом фізичної реабілітаційної терапії за допомогою VR

1.4.1 ATTILAN VR

ATTILAN - це відеогра під VR-шолом з послідовним сюжетом. Гравцям належить побувати на орбіті Марса, всередині міжнародної космічної станції ATTILAN і підготуватися до майбутньої колонізації планети.

Але завдання і труднощі, які очікують їх поруч з Червоною планетою, побудовані таким чином, щоб навчити користуватися біонічної протезами. Гравці тут гуляють по станції, виконують дрібні побутові завдання, керують вантажами і стріляють. Для різних заходів реабілітації передбачені різні ігрові персонажі: інженер, учений, пілот космічного корабля.

Управління відбувається за допомогою тих самих м'язів, що управляють і протезом. Спеціальні датчики фіксують напругу м'язових волокон передпліччя і передають дані в програму. Гравець бачить перед собою, наприклад, одну «живу» руку і один футуристичний протез, яким рухає, стискає і розтискає долоню, бере і переносить предмети завдяки правильній роботі м'язів. Саме так в майбутньому гравець буде управляти своїм біонічним протезом в реальному світі (рис 1.1).



Рисунок 1.1 - Ілюстрація до реабілітаційної гри ATTILAN

1.4.2 ReviMOTION

Програма ReviMOTION розроблена для дітей з ДЦП або іншими порушеннями рухової активності.

Реабілітація з нейротренажером ReviMOTION ще більше схожа на комп'ютерну гру. З її допомогою пацієнти виконують курс фізичних вправ в легкій ігровій формі. Правильно виконане рух змушує рухатися віртуального персонажа і проходити рівень за рівнем.

У той же час лікар за допомогою системи оптичного трекінгу і аналітики рухів, передбачених в тренажері, може відстежувати прогрес, а вбудована інформаційно-аналітична система допоможе йому «вести» пацієнта від сеансу до сеансу протягом усього реабілітаційного курсу (рис 1.2).



Рисунок 1.2 - Ілюстрація до нейротренажера ReviMOTION

1.4.3 Програма віртуальної реабілітації "Девірта Делфі"

«Девірта Делфі» - це віртуальний кіберпростір з власним 3D-світом, для занурення в який, також як в і ATTILAN, необхідні VR-шолом і сенсорні датчики, що забезпечують біологічний зворотний зв'язок. Тут технологія ще більш інтерактивна - комп'ютерний аватар пацієнта повторює рухи людини в віртуальному світі, що створює видимість реальної взаємодії з навколишнім простором і додатково мотивує.

Людям, які пережили пошкодження мозку, призначають медичну реабілітацію в формі простих щоденних вправ. «Девірта Делфі» контролює їх процес і дозволяє пацієнтові бачити результат своїх зусиль, відчуючи себе головним персонажем кожного заняття курсу (рис 1.3).

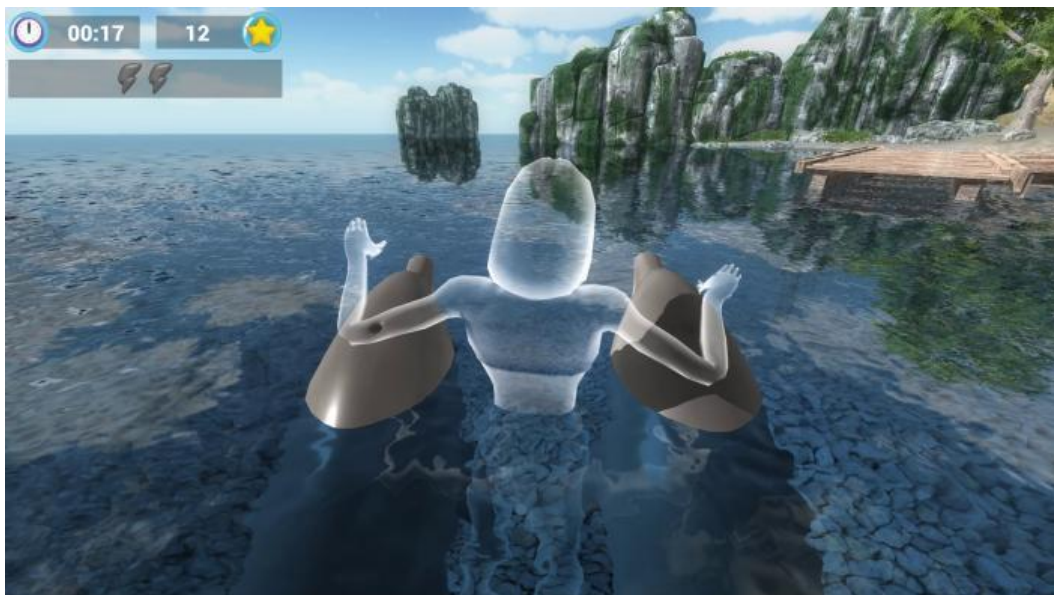


Рисунок 1.3 - Програма віртуальної реабілітації "Девірта Делфі"

2 ОГЛЯД ТЕХНОЛОГІЙ OCULUS

2.1 Шолом віртуальної реальності Oculus Rift

Oculus Rift - шолом віртуальної реальності від компанії Oculus, перша версія якого була презентована у 2013 році (Oculus Rift DK1).

Версія для розробників почала надходити першим замовникам з 30 березня 2013 року. На червень 2013 року розробникам було розіслано понад 5 тис. комплектів.

Загальні характеристики Oculus Rift DK1[6]:

- розміри екранів: 1280x800 (640x800 на око);
- тип екрану: LCD;
- 3D ефект: так, технологія side-by-side;
- горизонталь видимості: 90 градусів;
- діагональ видимості: 110 градусів;
- дротове підключення: Роз'єми DVI / HDMI і USB;
- платформи: PC (персональні комп'ютери);
- відстеження руху: поворот голови;
- трекер руху голови: 240 Гц;
- ступені свободи: 3 ступеня;
- маса: 380 г.

У червні 2013 року на EEE 2013 (Electronic Entertainment Expo 2013) був продемонстрований покращений прототип пристрою з роздільною здатністю Full HD (1080p).

Усього, за два роки (2013 - 2014) було випущено три версії для розробників: Oculus Rift DK1, Oculus Rift DK2 та Oculus Rift Crescent Bay. Остання версія має більш високе розширення, ніж DK1 та DK2, меншу вагу, вбудовані знімні аудіо-навушники, завдяки наявності додаткових ІК-світлодіодів в задній частині пристрою краще відстежує повороти аж до 360 градусів.

Споживча (наступна) версія Oculus Rift CV1 допрацьовувалася протягом довгого часу. У ній планувалися поліпшення трекера, збільшення розширення до деякого рівня «вище, ніж 1080р». Характеристики шолома версії CV1:

- розширення: 1080 × 1200 пікселів на кожне око;
- частота оновлення: 90 Гц, 75 Гц, 72 Гц, 60 Гц;
- кут огляду номінальний: 110 °;
- час відгуку: 3 мс, full;
- частота оновлення вбудованого трекера: 1000 Гц (1000 разів в секунду);
- датчики: гіроскоп, акселерометр, магнітометр, інфрачервоні датчики;
- частота оновлення трекера, вбудованого в камеру: 60 Гц;
- вага: 450 г без кабеля.

Гіроскоп - датчик положення у просторі, відстежує положення голови та рук у віртуальному середовищі.

Акселерометр - датчик переміщення у просторі.

Магнітометр - з його допомогою можна наводити погляд на об'єкт у віртуальній реальності (наприклад, кнопку меню) і, утримуючи погляд на ньому кілька секунд, активувати його.

У комплект до шолому також входять контролери Oculus Touch та сенсори. Характеристики контролерів Oculus Touch:

- датчики: акселерометри;
- з'єднання по радіоканалу 2,404-2,478 GHz, 2,89 mW одностороннє;
- вібромотор: є;
- управління: кнопки, джойстик, курок;
- сумісність: Oculus Rift.

Сенсор - датчик положення, що збільшує ігрову зону і підвищує стабільність роботи шлему віртуальної реальності Oculus Rift CV1.

Наступна споживча версія, Oculus Rift S була випущена весною 2019 року. У новій версії VR-Шолома, oculus позбулися відстеження за допомогою док-станцій і перейшли на відстеження за допомогою камер на поверхні шолома. Другою відмінністю стало нове кріплення, виготовлене за ліцензією Sony, запозичене у шолома PlayStation-VR. Третім відмінністю є спрощення вбудованої звукової апаратури, навушники були замінені на вбудовані динаміки. Четверта відмінність полягає в новій версії контролерів Oculus Touch [6].

2.2 Рекомендовані характеристики комп'ютера

Комп'ютер - це двигун на базі якого працює Oculus Rift. Для оптимальної роботи ігор та застосунків необхідно використовувати комп'ютер з наступними мінімальними характеристиками:

- графічна карта: NVIDIA GTX 1050Ti / AMD Radeon RX 470 або більш потужні;
- альтернативні моделі графічної карти: NVIDIA GTX 960 4 ГБ / AMD Radeon R9 290 або більш потужні;
- процесор: Intel i3-6100 / AMD Ryzen 3 1200, FX4350 або більш потужний;
- оперативна пам'ять: RAM 8 ГБ або більше;
- відеовихід: сумісний відеовихід HDMI 1.3;
- порти USB: 1 порт USB 3.0 і 2 порти USB 2.0;
- операційна система: Windows 10.

Рекомендовані характеристики комп'ютера:

- графічна карта: NVIDIA GTX 1060 / AMD Radeon RX 480 або більш потужні;
- альтернативний варіант графічної карти: NVIDIA GTX 970 / AMD Radeon R9 290 або більш потужні;

- процесор: Intel i5-4590 / AMD Ryzen 5 1500X або більш потужний;
- пам'ять: RAM 8 ГБ або більше;
- відеовихід: Сумісний відеовихід HDMI 1.3;
- порти USB: 3 порти USB 3.0 і 1 порт USB 2.0;
- операційна система: Windows 10.

Також для використання деякого контенту на Rift необхідна комп'ютерна система, що перевищує мінімальні або рекомендовані вимоги Oculus. Тому перед використанням необхідно вивчити рекомендовані характеристики системи для тих додатків на Rift, які ви збираєтеся використовувати.

2.3 Програмне забезпечення для використання Oculus Rift

Для початку роботи з шоломом віртуальної реальності Oculus Rift необхідно перейти на офіційний сайт [7] та завантажити ПЗ для потрібної версії шолому.

Після завантаження необхідно відкрити програмне забезпечення Oculus (рис. 2.1) та налаштувати Oculus Rift. Застосунок сам запропонує налаштувати шолом. Якщо цього не відбулось, необхідно перейти до вкладки “Устройства” та додати гарнітуру. (рис. 2.2 - 2.3).

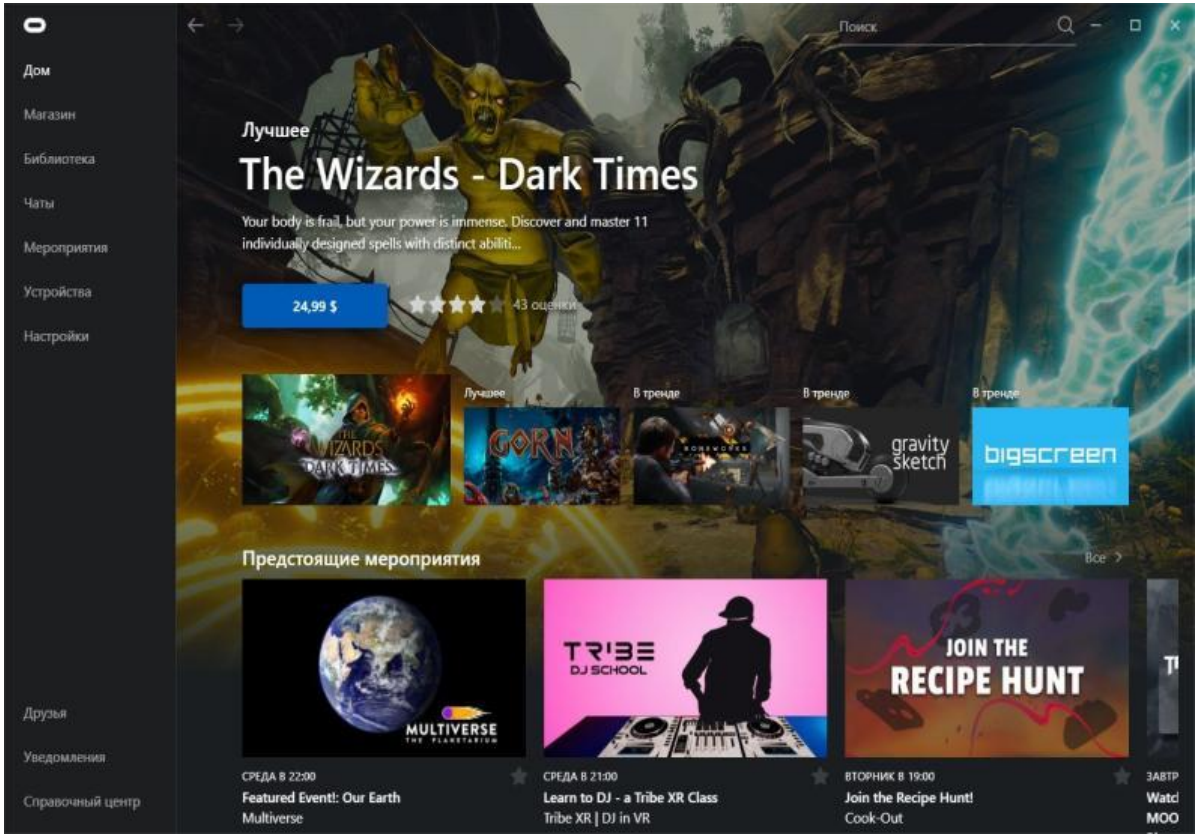


Рисунок 2.1 - Интерфейс программы Oculus

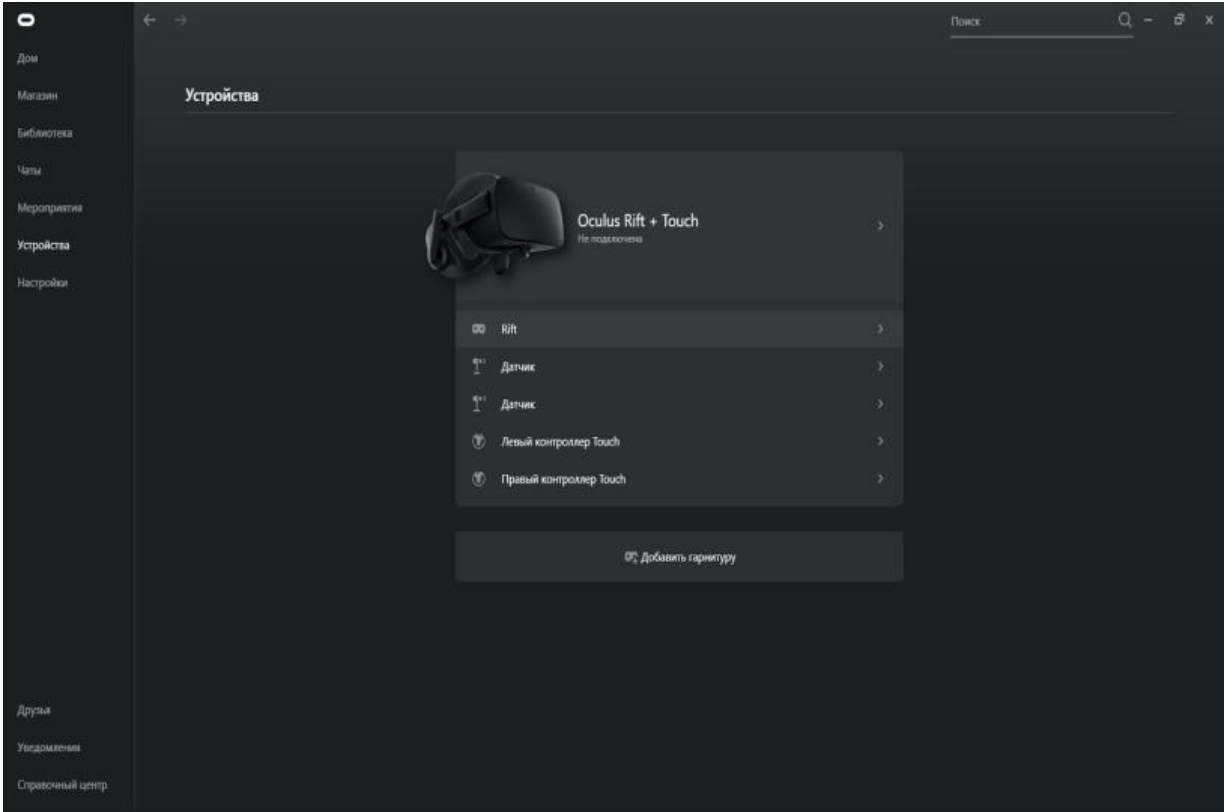


Рисунок 2.2 - Вкладка “Устройства”

Якщо необхідно повторно налаштувати гарнітуру, можна також перейти на вкладинку “Устройства”, натиснути на гарнітуру (в моєму випадку Oculus Rift + Touch), після цього зліва з’явиться меню налаштувань (рис. 2.4). Далі необхідно перейти до “Настройка устройства”.

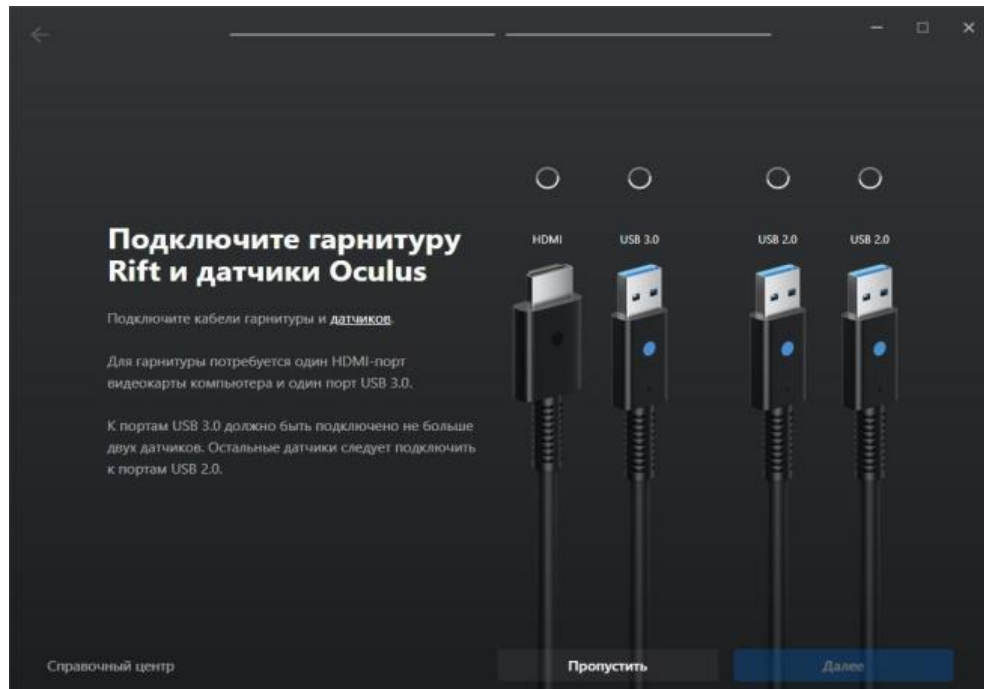


Рисунок 2.3 - Налаштування гарнітури

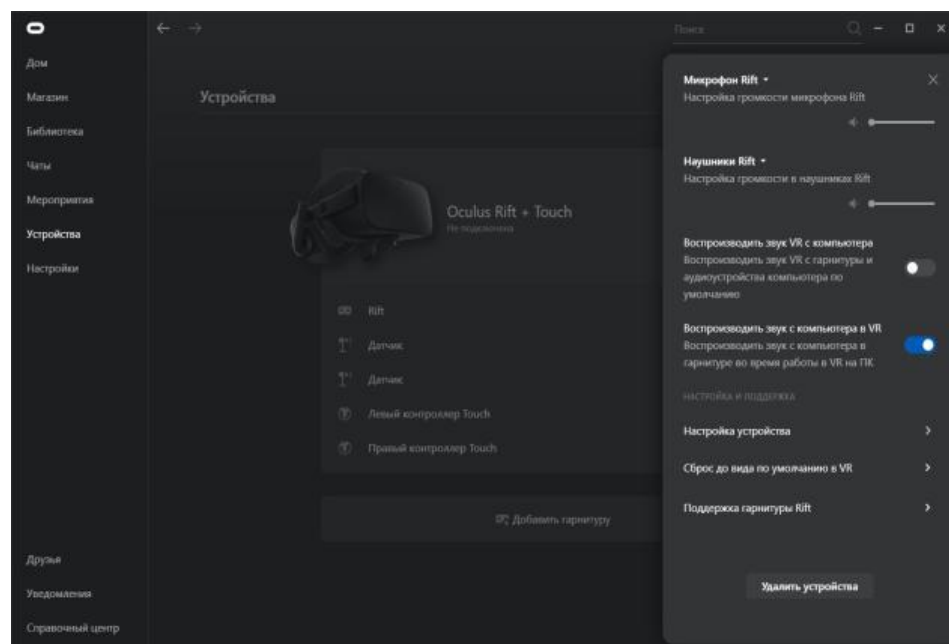


Рисунок 2.4 - Меню налаштування гарнітури

Після закінчення налаштування можна переходити до використання гарнітури.

Також в застосунку Oculus можна переглянути список ігор (зазвичай шутери, хоррори або гонки) та додатки, наприклад для перегляду відео. Для цього необхідно перейти до вкладки “Магазин” (рис. 2.5) та обрати додаток, що потрібен. Більшість ігор та додатків - платні.

Список вже встановлених додатків можна переглянути у розділі “Бібліотека” (рис. 2.6).

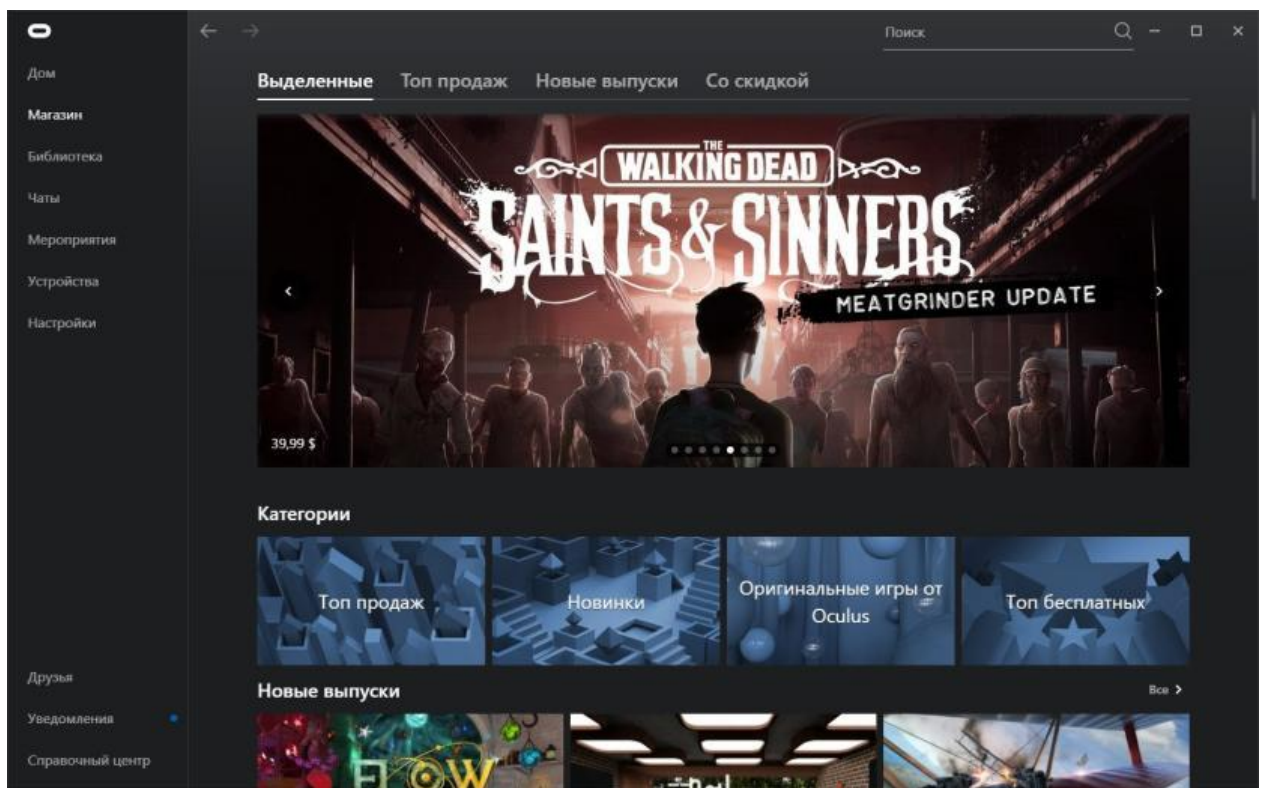


Рисунок 2.5 - Розділ “Магазин”

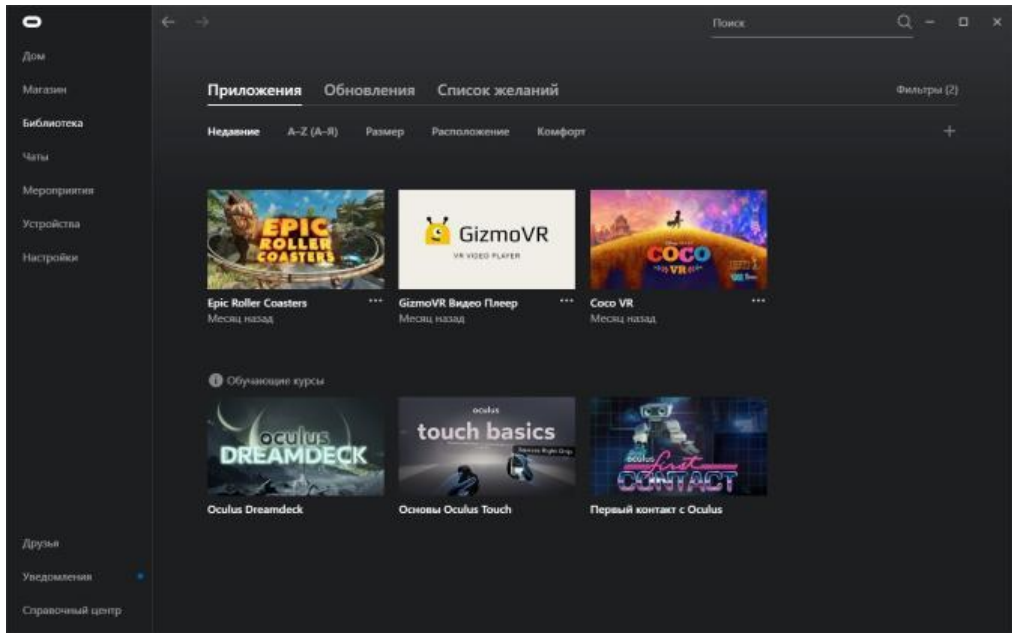


Рисунок 2.6 - Розділ “Бібліотека”

Для того щоб було можливо використовувати сторонні додатки разом з Oculus Rift, необхідно увімкнути відповідне налаштування Невідомі джерела у розділі “Налаштування” -> “Загальні” (рис. 2.7).

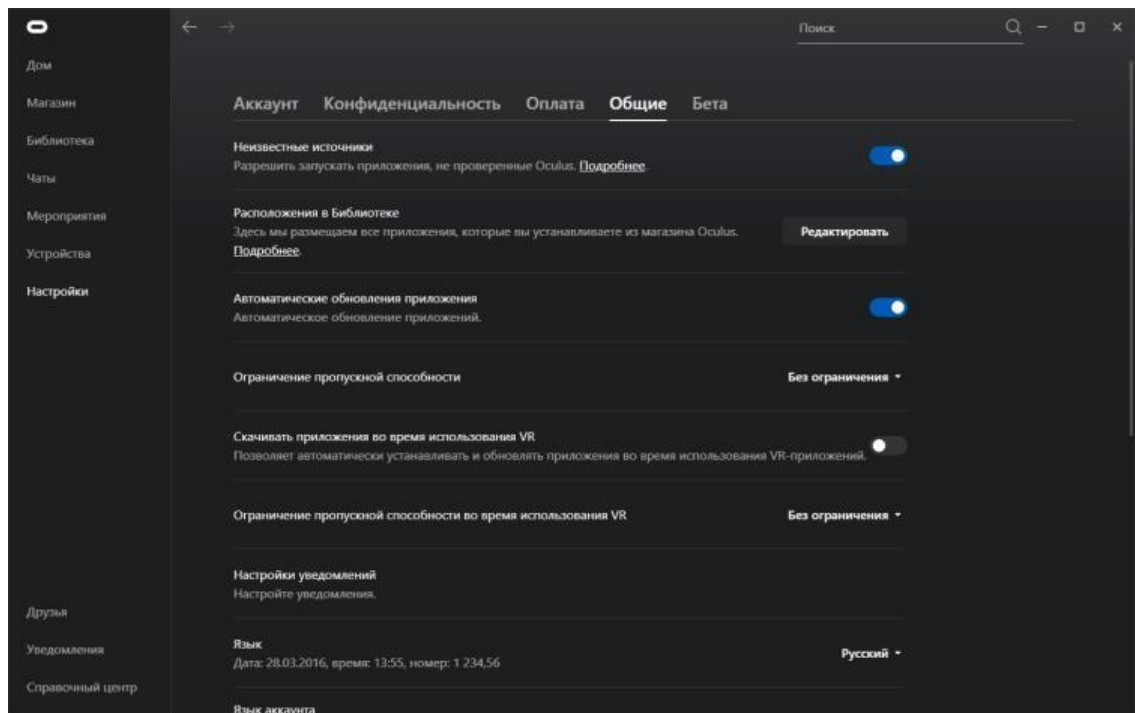


Рисунок 2.7 - Ввімкнення налаштування “Невідомі джерела”

3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір мови програмування

Розробку додатків в індустрії VR можна умовно розділити за напрямками:

- розробка для віртуальних шоломів;
- розробка для мобільних застосунків;
- вебробробка.

Базові мови програмування для створення застосунків для віртуальних шоломів - C++ та C#. В основному розробники використовують спеціальні ігрові движки, які дозволяють створювати інтерактивність в 3D-сценах. Найпопулярніші - Unreal Engine 4 (UE4) і Unity. Обидва мають дуже широкі можливості і є надійними інструментами. Навколо обох склалися активні спільноти з численними інформаційними ресурсами. Обидва движки дозволяють управляти 3D-оточенням, імпортувати власний контент (3D-моделі, зображення, звук, відео), а також програмувати інтерактивність і геймплей [8].

При розробці VR для мобільних застосунків рекомендовано використовувати Swift для iOS і Kotlin - для Android.

Крім ігрових движків, можна також звернутися до розробки інтерактивних VR-веб-сторінок. Це можна робити за допомогою мови розмітки Mozilla's A-Frame, JavaScript (фреймворк Three.js), HTML5 і / або WebGL. Подібні експерименти ведуться в Chrome і Mozilla. Розробка для вебу дозволяє відображати VR-контент прямо на смартфонах користувачів.

Так як розробка буде проходити для шоломів віртуальної реальності, буду обирати поміж Unreal Engine 4 (UE4) і Unity.

3.1.1 Вибір середовища розробки

Unity - це кроссплатформенна середа розробки комп'ютерних ігор, що дозволяє створювати додатки, що підходять більш ніж під 25 різноманітних платформ, включаючи ігрові консолі, ПК, мобільні пристрої та інтернет-додатки. Unity має візуальну середу розробки та модульну систему компонентів.

Unreal Engine - ігровий рушій, створений компанією Epic Games, що був випущений ще в 1996 році. Написаний на мові програмування C++, Unreal Engine дозволяє створювати ігри майже під усі операційні системи та платформи.

Першою відмінністю цих двох движків є те, що в Unreal Engine 4 використовується мова програмування C++, а в Unity3d в основному C# або JavaScript. Unreal Engine 4 також має рішення для людей, які бояться високого порогу входження в C++. Це Blueprint - редактор візуального скриптинга. Технічно там не потрібно писати жодного рядка коду. Це дуже зручно для створення швидких прототипів, і навіть можна створювати цілі гри за допомогою Blueprint [9].

Обидва рушії Unreal Engine 4 і Unity3d мають свій магазин Асетів: з них можна завантажити готові 3d моделі персонажів і оточення, текстури і навіть такі речі, як звуки і системи частинок. Проте, Unity3d однозначно виходить на перше місце з точки зору кількості Асетів в магазині.

Unreal Engine 4 і Unity3d є умовно безкоштовними. Повна версія Unreal Engine 4 буде безкоштовною, поки дохід від гри становить менше \$ 3 000 на квартал, якщо ж більше - доведеться ділитися 5% від своїх доходів.

З Unity3d все складніше. Якщо доходи компанії складають менше \$ 100 000 на рік, можна використовувати безкоштовну версію, але вона не повна.

Проаналізувавши відмінності движків, обираю Unity, так як в цьому движку використовується мова C#, магазин Unity3d має більшу кількість

ассетів, а безкоштовної версії достатньо, щоб розробити повноцінне програмне забезпечення.

3.2 Графічні рішення Unity 3D

Для розробки ігор Unity пропонує декілька готових процесів: High-Definition Render Pipeline (HDRP) та Lightweight Render Pipeline(LWRP). Кожен з них має свою область застосування та технічні вимоги. Також за допомогою Scripting Render Pipeline можна управляти та налаштовувати рендеринг за допомогою скриптів на мові C#. Це дозволяє не тільки вносити невеликі зміни, але й повністю переробляти та налаштовувати процес рендерингу відповідно до потрібних вимог.

Мета LWRP - забезпечити оптимізовану продуктивність в реальному часі на платформах з обмеженою продуктивністю, частично жертвуючи висвітленням та затемненням. Цільова аудиторія Lightweight Render Pipeline - розробники, орієнтовані на широкий спектр мобільних платформ, VR застосунків, та тих, що розробляють ігри з обмеженими потребами в освітленні в режимі реального часу. LWRP виконує однопрохідний (single-pass) прямий рендеринг, з одним тіньовим світлом у реальному часі та відбиванням світла на кожен об'єкт, з перевагою того, що всі вогні затінюються за один прохід. У порівнянні з традиційним конвеєром прямого рендеринга, який виконує додатковий прохід для кожного пиксельного світла в межах діапазону, використання конвеєра LW призведе до меншої кількості викликів малювання. Це відбувається за рахунок деякої додаткової складності шейдерів через більшої кількості проходів. LWRP також підтримується інструментом Shader Graph, що дає деякі додаткові переваги щодо робочого процесу створення шейдерів [10].

HDRP призначений для високопродуктивних платформ, включаючи ПК, Xbox і PlayStation. Він дозволяє забезпечити реалістичну графіку в

складних сценаріях. У HDRP використовуються технологія Compute Shader і можливості сучасних графічних процесорів.

HDRP можна використовувати в розробці ігор з високою роздільною здатністю, автомобільних демонстрацій, архітектурних та інших додатків, де пріоритет віддається якості графіки. HDRP підтримує як попереджуючий, так і відкладений рендеринг, і використовує фізичне освітлення і матеріали.

Один з напрямків HDRP - єдине і логічне висвітлення. Інший напрямок - продуктивність і сучасні технології.

Дизайн HDRP відповідає трьом принципам:

- візуалізація базується на фізичній основі;
- освітлення є уніфікованим і цілісним;
- функції функціонують незалежно від шляху візуалізації.

Візуалізація на фізичній основі базується на трьох опорах: освітленні, матеріалі та камері. Освітлення та матеріал покладаються на фізичну взаємодію, і вони повинні бути чітко роз'єднані, щоб отримати цілісний результат за різних умов освітлення. Камера інтерпретує отримане освітлення, щоб відобразити його на екрані. Мета HDRP - полегшити художникам досягнення реалістичного результату [10].

Уніфіковане освітлення означає, що всі об'єкти та носії інформації, що беруть участь у сцені, повинні отримувати однакове освітлення. Не повинно бути різниці між непрозорими, прозорими або об'ємними матеріалами. Когерентне освітлення означає, що матеріал повинен коректно взаємодіяти з освітленням, виробленим будь-яким джерелом (зондом для відбиття, освітленням на ділянці тощо), навіть якщо він модифікується наклейкою, наприклад. Результат - більш узгоджений вигляд.

Візуалізація в режимі реального часу використовує різні підходи рендерингу: відкладене/пряме, однопрохідне/багатопохідне та тайловий / кластерний. Під час розробки ігор, є нормальним те, що обраний шлях відтворення обмежує графічні функції, якими можна користуватися. Незалежно від обраного шляху візуалізації, з HDRP доступний однаковий

набір графічних функцій. Це означає, що розробник може обрати шлях візуалізації, виходячи виключно з вимог до продуктивності; продуктивність не обмежується графічними функціями [10].

3.3 Огляд та налаштування середовища розробки Unity3d

Для початку роботи з Unity3d необхідно, спочатку зареєструватись, а потім завантажити безкоштовну версію Personal Unity Hub з офіційного сайту Unity за посиланням: [11] (рис. 3.1).

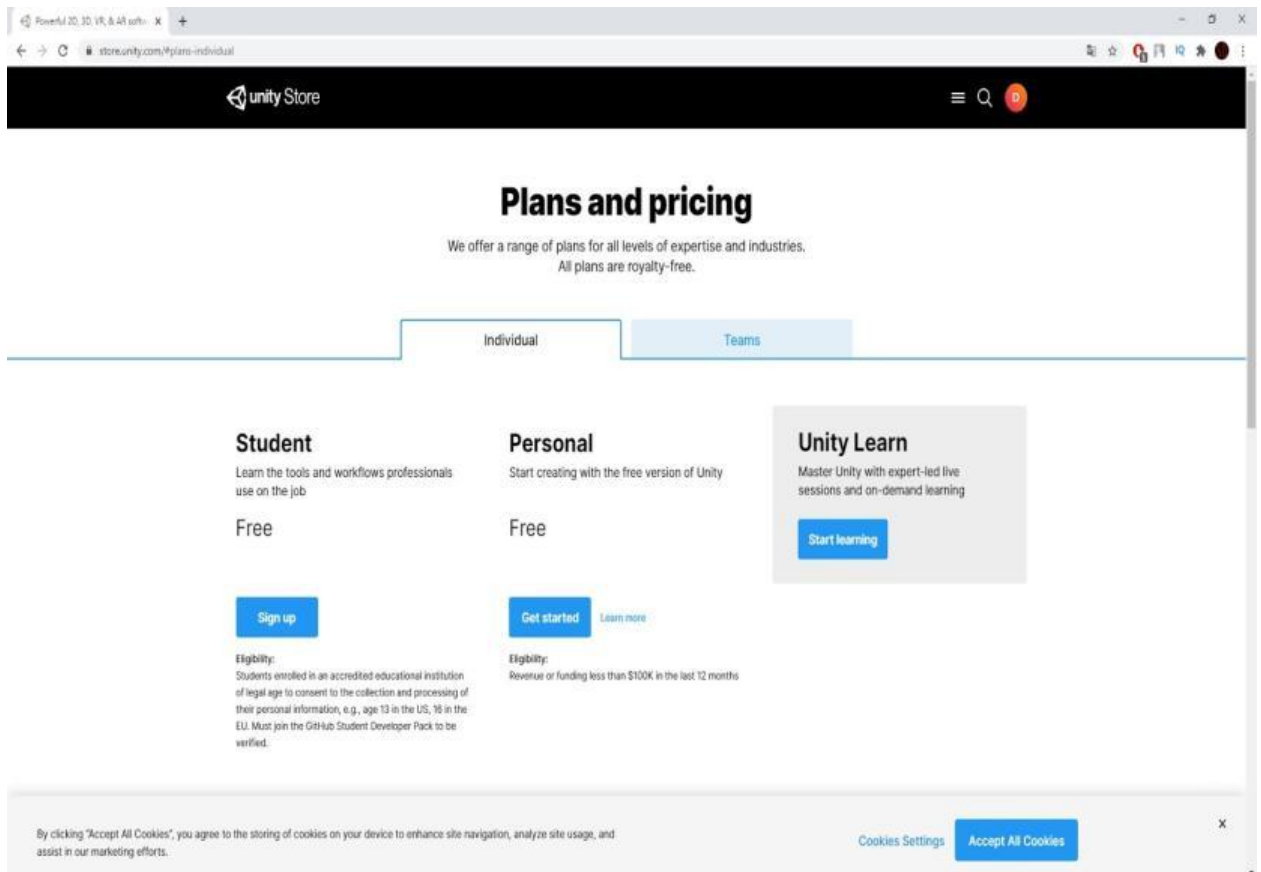


Рисунок 3.1 - Інтерфейс ресурсу для завантаження Unity Hub

Після завантаження та встановлення програми, одразу можемо створити проект. Для цього натискаємо на “New”, обираємо 3D проект, вводимо назву (рис. 3.2-3.3), та натискаємо “Create”.

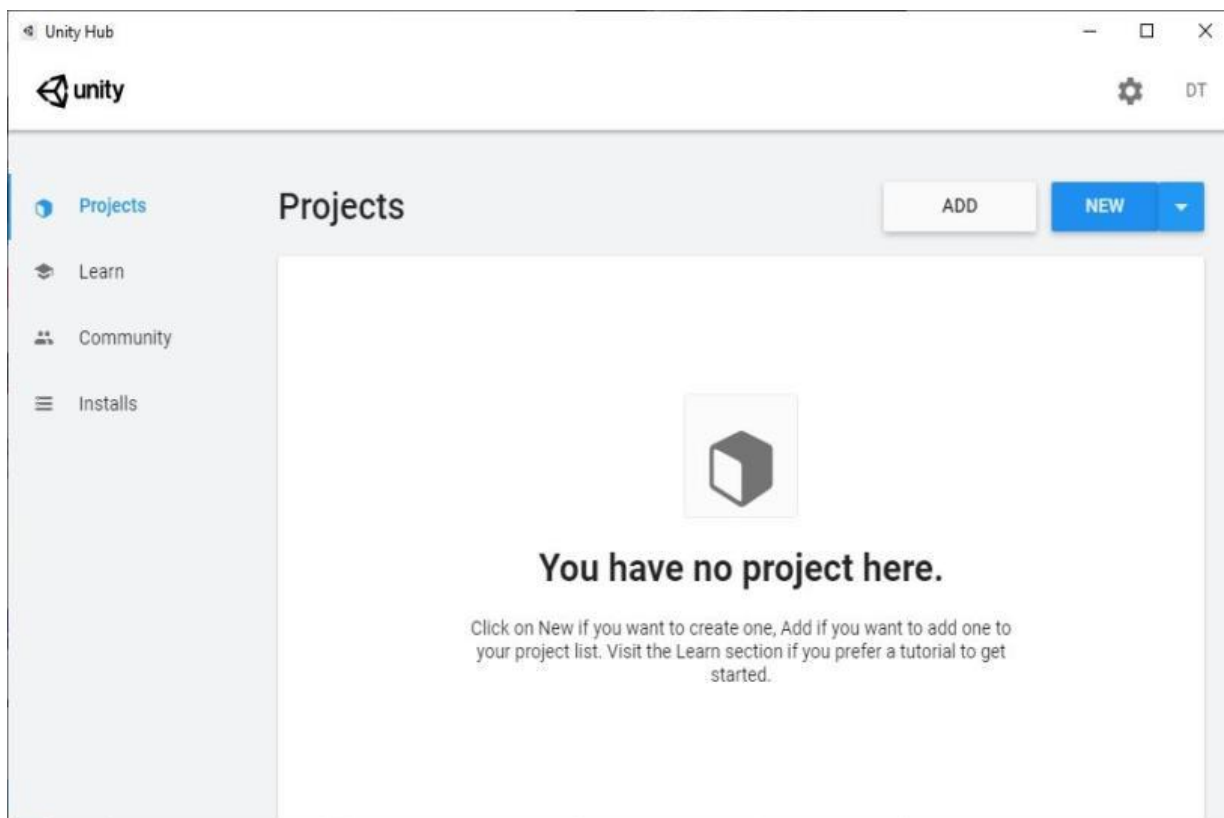


Рисунок 3.2 - Інтерфейс Unity Hub

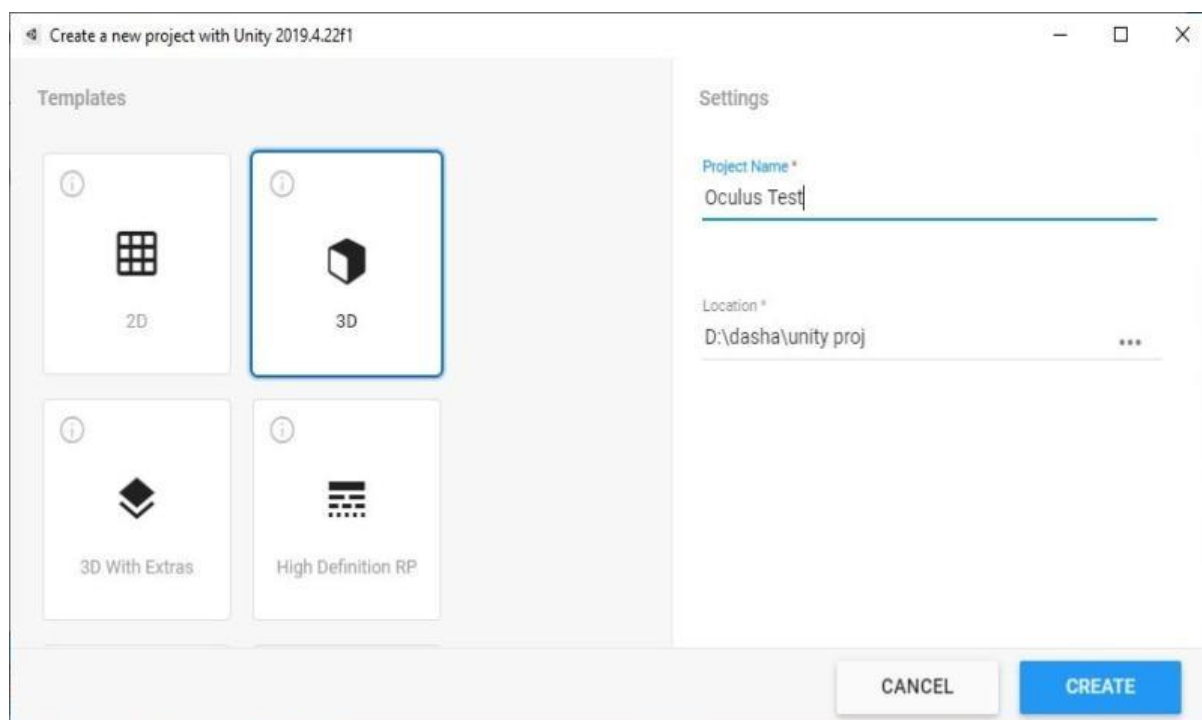


Рисунок 3.3 - Створення 3D проекту

Проект може створюватись на протязі декількох хвилин. Далі необхідно перейти до магазину Асетів від Unity. Для цього у верхньому

меню середі розробки Unity знаходимо “Window” -> “Asset Store” (рис. 3.4). Далі у пошуковій строці вводимо “Oculus Integration” та встановлюємо доповнення (рис 3.5).

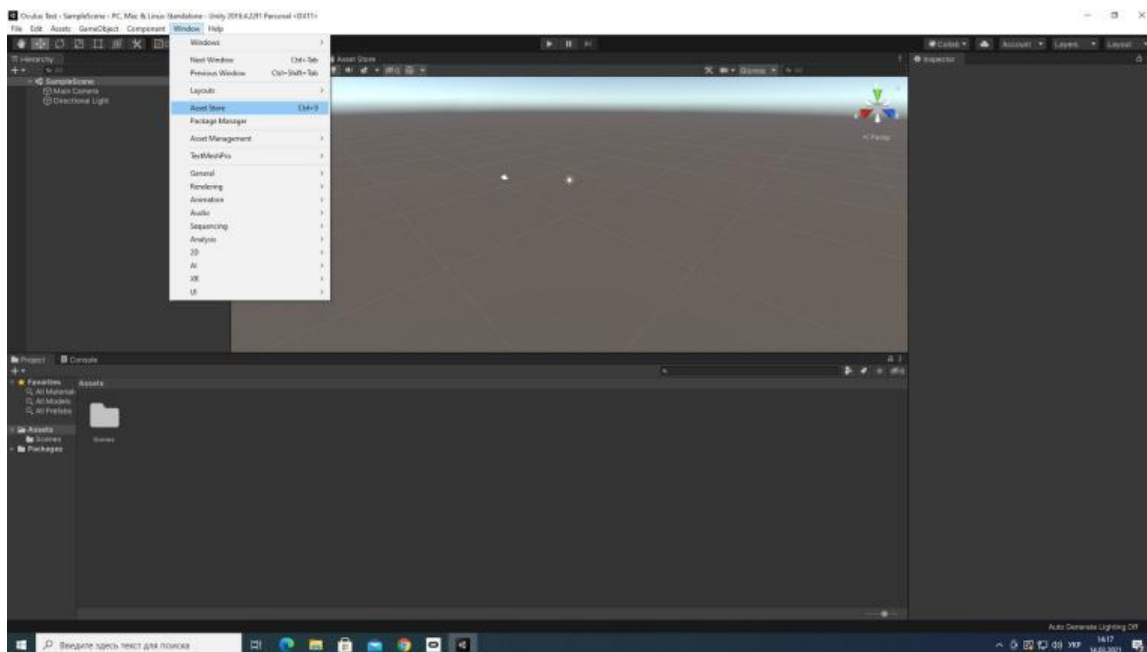


Рисунок 3.4 - “Window” -> “Asset Store”

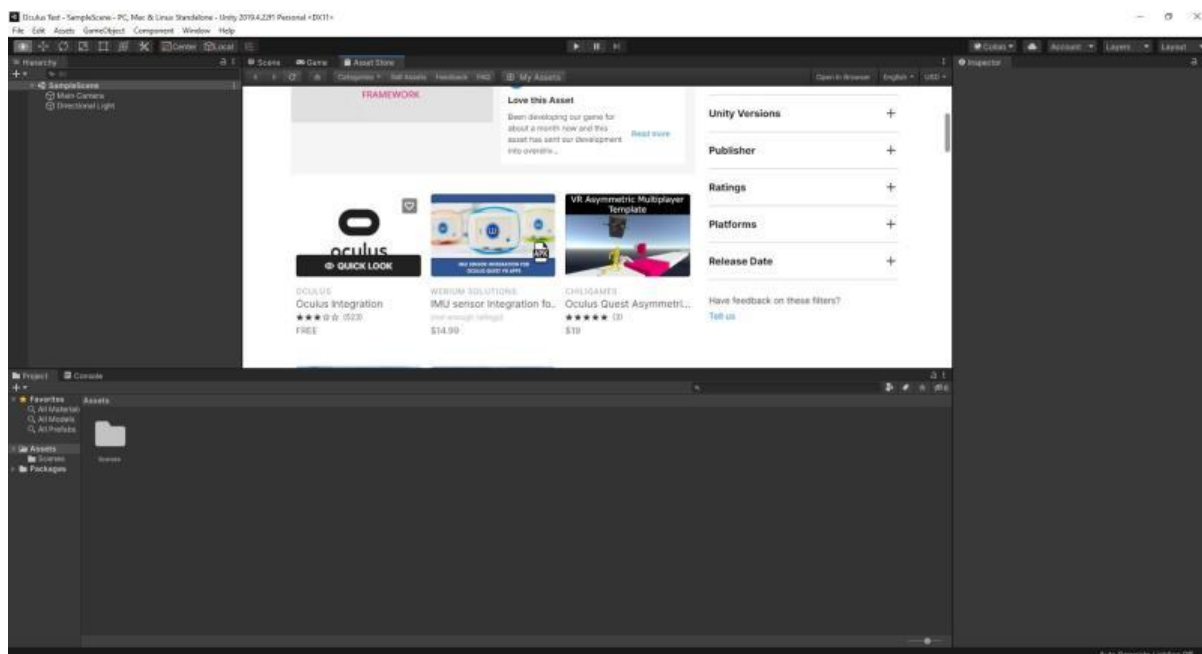


Рисунок 3.5 - магазин Ассетів

Після імпорту бачимо, що в нас з'явилися додаткові папки з засобами для створення VR середовища.

Також для правильного функціонування необхідно дозволити підтримку віртуальної реальності. Для цього необхідно перейти до Edit -> Project Settings -> Player -> XR Settings -> Virtual Reality Support. Далі необхідно натиснути на “+” в лівій верхній частині блоку та обрати Oculus.

Для того, щоб проект використовував для запуску шолом віртуальної реальності, необхідно створити VR камеру. Це можна зробити двома способами:

- знайти готову OVR камеру в пакеті Oculus, та перетягнути її до проекту;
- створити самостійно. Для цього необхідно виконати наступні дії: створити пустий об'єкт (XRRig), в інспекторі додати компонент XR Rig; далі необхідно створити ще один пустий об'єкт, Camera Offset, після цього натискаємо ПКМ по Camera Offset, та додаємо об'єкт Camera. В інспекторі XRRig, в поле Camera Floor Offset Object перетягуємо Camera Offset, а в Camera Game Object - об'єкт Camera. Далі у інспекторі камери додаємо компонент Tracked Pose Driver.

Також серед компонентів, що найчастіше використовуються: XR Controller - для визначення поведінки контролерів; Box/Sphere Collider - для визначення поведінки квадратних та круглих об'єктів; XR Direct Interactor та XR Grab Interactable - для взаємодії з об'єктами, тощо.

Для більшого комфорту проектування, використовувались засоби пакету Virtual Reality Toolkit(VRTK).

VRTK має велику кількість префабів та скриптів для зручного проектування ігор.

Для правильного функціонування пакету, необхідно спочатку прописати залежності у manifest.json, а потім завантажити його останню версію з репозиторію. Для цього необхідно:

- у верхній частині файлу manifest.json, після першої { дописати код, зазначений у Лістингу 3.1;
- у розділі dependencies section також необхідно вписати "io.extendreality.vrtk.prefabs": "1.1.3";
- повернутися до середи розробки, почнеться завантаження пакетів;
- відкрити папку Assets проекту, натиснути ПКМ -> Git Bash Here;
- у командному рядку Git необхідно клонувати репозиторій. Для цього потрібно вписати команду: `git clone https://github.com/ExtendRealityLtd/VRTK.Tutorials.OculusIntegration.git`;
- повернутися до середи розробки, в папці Assets з'явиться пакет VRTK.Tutorials.OculusIntegration.

Лістинг 3.1 - Налаштування manifest.json

```
"scopedRegistries": [
  {
    "name": "npmjs",
    "url": "https://registry.npmjs.org/",
    "scopes": [
      "io.extendreality"
    ]
  }
],
```

Також при проектуванні віртуального середовища було використано ассет Sci-Fi Styled Modular Pack. Він дозволяє створювати сцени різної складності, має безліч оригінальних префабів, текстур, анімацій, завдяки чому можна легко створити цікаву середу.

3.3.1 Огляд інструментів розробки Unity

Основою створення ігор на Unity є ігровий об'єкт (GameObject) - фундаментальний об'єкт, за допомогою якого створюються персонажі, реквізит, декорації. Ігрові об'єкти самі по собі не несуть багато функціоналу, але вони виступають контейнерами для компонентів, які задають їх поведінку. За замовчуванням, у всіх ігрових об'єктів є компонент Transform (рис. 3.6). Цей компонент задає положення, кут оберту та масштаб ігрового об'єкту.

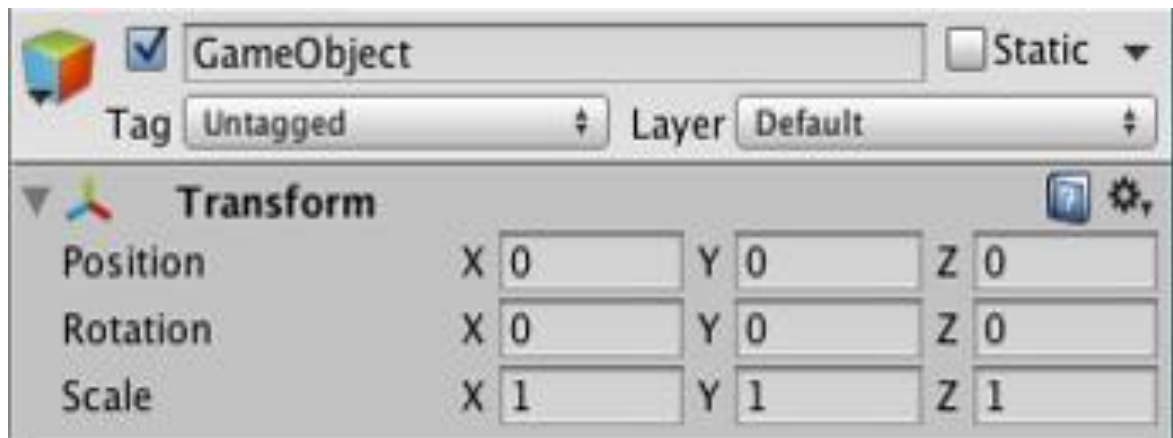


Рисунок 3.6 - Компонент Transform

До будь-якого ігрового об'єкту, через меню Components, або через браузер компонентів, можна додавати компоненти. Компоненти можна створювати та редагувати.

Для створення компоненту, необхідно створити скрипт на C#, та прописати в ньому потрібну логіку. Усі створені скрипти, за замовчуванням, наслідують клас MonoBehaviour, та мають два основні методи: Start та Update. Start - метод, що відпрацьовує один раз при створенні компоненту (ініціалізація). Update - спрацьовує при кожному відновленні кадру.

Компоненти - дуже гнукі, їх значення та властивості, при підключенні до ігрового об'єкту, можуть змінюватися в редакторі при створенні гри, або в

скриптах під час роботи гри. Є два основних типи властивостей: значення (Values) і посилання (References).

3.4 Концепція додатку фізичної реабілітації.

Проаналізувавши програмні аналоги с реалізованим методом фізичної реабілітаційної терапії за допомогою VR, можемо побачити, що найбільш ефективним підходом є імітація повсякденних дій у віртуальному середовищі.

За допомогою контролерів Oculus Touch можна відслідковувати положення рук, зчитуючі їх положення відносно шолому. Також використовуючи комбінації кнопок та джойстиків людина може тренувати дрібну моторику рук. Таким чином, створюючи додаток, необхідно розробити такі завдання, у яких будуть задіяні маніпуляції об'єктів, що потребують певного положення рук та керуються за допомогою комбінацій кнопок.

Дослідження показують, що за допомогою VR можна обдурити мозок. Якщо пацієнт після інсульту може підняти ослаблену, частково паралізовану руку до певного рівня, і це межа його уявних можливостей, то у віртуальній реальності він може підняти її високо, повторюючи за кимось від імені свого "аватара".

Як саме це працює? За словами лікарів, нейрони первинної моторної кори особливо активізуються за допомогою такого феномена, як візуальна дисторсія: віртуальне зображення спотворюється, як і реальне при зоровому сприйнятті. Таким чином, по суті, мозок обманюється.

Важливо, що потім ця навичка закріплюється після виходу з віртуальної реальності.

Було вирішено створити віртуальну середу з багатою кількістю об'єктів, якими можна маніпулювати певним чином. Для повного занурення,

необхідно створити цікаву, приємну віртуальну середу, інноваційну віртуальну кімнату з відображенням вимог до вправ, що виконує пацієнт.

Для проходження завдань необхідно переміщуватись по місцевості та взаємодіяти з об'єктами.

При запуску додатку система визначає список певних завдань, та починає контролювати їх виконання. Система дає певні підказки щодо виконання завдань пацієнтом.

Застосунок дає можливість переміщуватись по місцевості за допомогою проміню телепортації, взаємодіяти з предметами та переглядати список завдань.

4 КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Алгоритми функціонування

Алгоритм роботи програми (рис 4.1) можна розділити на наступні етапи.

Етап 1. Початок роботи. Користувач надягає шолом віртуальної реальності та вмикає додаток.

Етап 2. Початок роботи компоненту TaskManager.

Етап 3. Створення переліку задач. Встановлення флагу поточної задачі.

Етап 4. Постійне відстеження дій користувача. Відображення підказки щодо виконання завдання.

Етап 5. Користувач виконує правильну дію, TaskManager помічає поточне завдання виконаним.

Етап 6. Перевірка стану завдань. Якщо було виконано останнє завдання – переходимо до Етапу 7.1. Якщо невиконані завдання ще є – до Етапу 7.2.

Етап 7.1. Відображення повідомлення про успіх.

Етап 7.2. Компонент змінює поточне завдання на наступне. Перехід до Етапу 3.

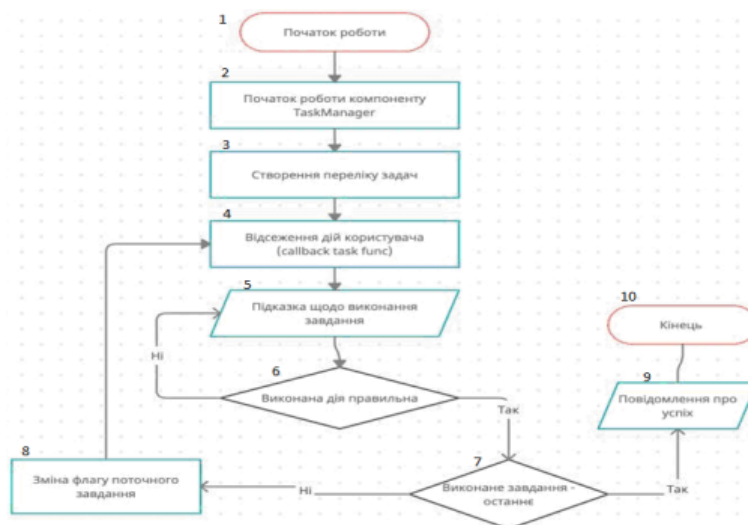


Рисунок 4.1 - Алгоритм роботи програми

4.2 Сценарії використання додатку.

4.2.1 Переміщення по місцевості

Прецедент: зміна позиції користувача в сцені з допомогою телепорту.

Актор: пацієнт.

Передумови:

- Oculus Rift підключений і працює коректно;
- додаток запущено і працює коректно;
- в налаштуваннях Oculus дозволені сторонні додатки.

Післяумови: користувач переміщений на точку телепортації.

Основний сценарій:

Крок 1. Користувач наводить "лазер" контролера на будь-яке, потрібне йому для переміщення місце.

Крок 2. За допомогою джойстику користувач вибирає напрямок телепортації .

Крок 3. Користувач натискає на джойстик.

Крок 4. Додаток відпрацьовує скрипт по переміщенню персонажа.

Крок 5. Камера змінює розташування на сцені.

Альтернативний сценарій:

- при некоректній роботі телепорту, користувач "залишається на місці" (рис. 4.2).



Рисунок 4.2 – Діаграма прецедентів «Зміна позиції користувача за допомогою телепорту»

4.2.2 Тренування пацієнту

Прецедент: відстеження виконання завдання.

Актор: пацієнт.

Передумови:

- Oculus Rift підключений і працює коректно;
- в налаштуваннях Oculus дозволені сторонні додатки.

Післяумови: користувач виконав реабілітаційні вправи.

Основний сценарій:

Крок 1. Користувач запускає додаток.

Крок 2. Спрацьовує система створення та відстеження поточних задач.

Крок 3. Додаток відображає користувачу поточну задачу.

Крок 4. Система відстежує дії користувача.

Крок 5. У разі виконання вправи, виводиться повідомлення про успіх.

Альтернативний сценарій:

- якщо вправа виконується неправильно, виводиться відповідне повідомлення та інструкції, як це виправити.

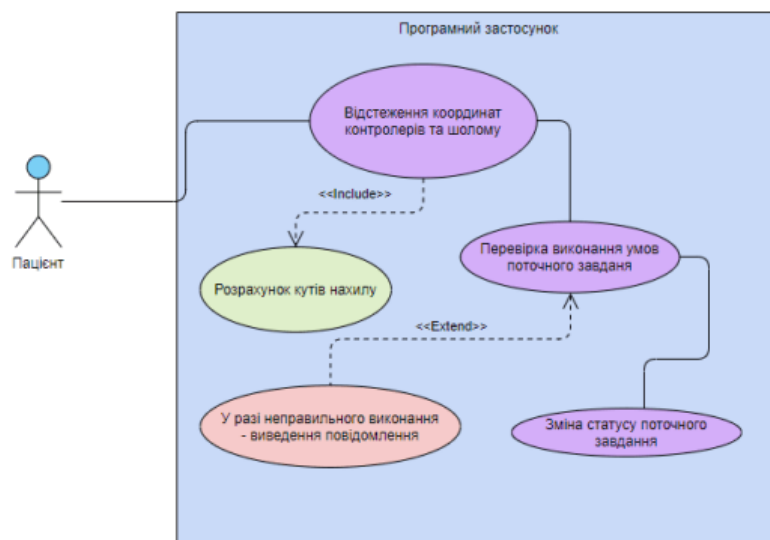


Рисунок 4.3 – Діаграма прецедентів «Відстеження виконання завдання»

4.3 Опис особливостей програмної реалізації.

Для початку реалізації програмного забезпечення, за допомогою ассетів, завантажених з Unity Asset Store, було створено сцену. Була використана велика кількість готових префабів (рис. 4.2). Префаби - готові шаблони ігрових об'єктів на яких вже встановлені компоненти.

Для функціонування засобів віртуальної реальності було створено систему префабів, що працюють з камерою(шоломом) та контроллерами (рис 4.3). Серед них: OVRCameraRig, TrackedAlias, Teleporter.Instant, TeleportCurve, InputHandlers.

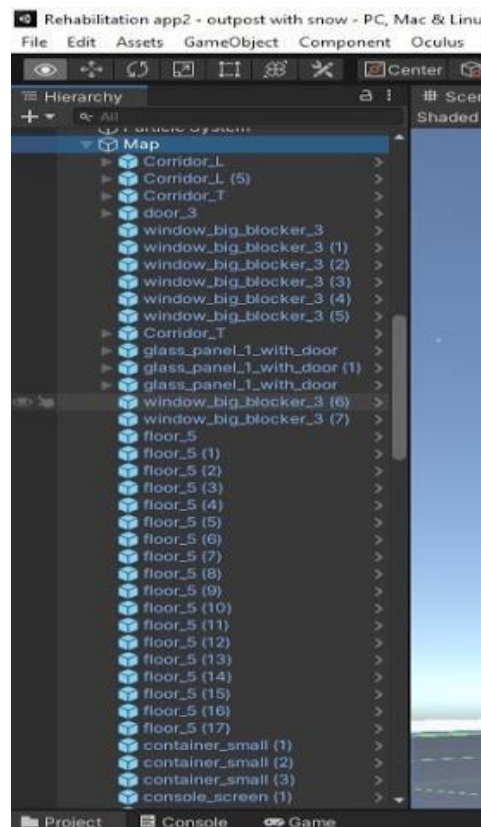


Рисунок 4.2 - Об'єкти сцени

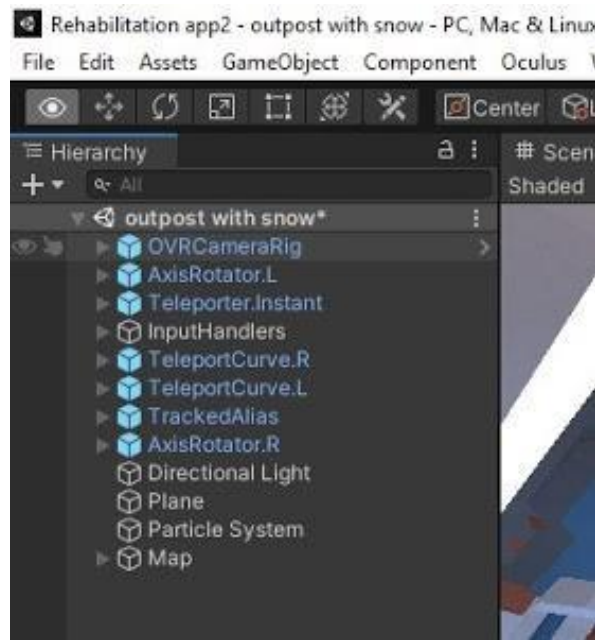


Рисунок 4.3 - Система префабів для правильного функціонування Oculus

OVRCameraRig - ігровий об'єкт, що керує налаштуваннями камери та контролерів. Він включає в себе об'єкт TrakingSpace, який в свою чергу включає об'єкти для функціонування кожної частинки системи: LeftEyeAnchor, CenterEyeAnchor, RightEyeAnchor, TrackedAnchor, LeftHandAnchor, RightHandAnchor (рис. 4.4).

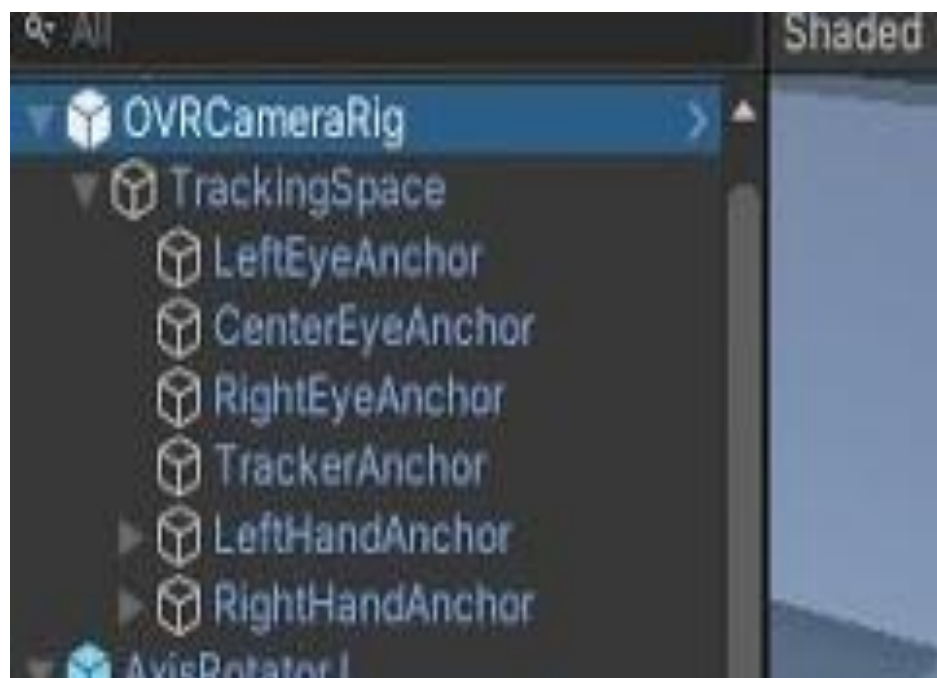


Рисунок 4.4 - OVR Camera Rig

Для правильного функціонування ігрових об'єктів, до кожного з них були підключені первні компоненти. У таблиці 4.1 подано перелік компонентів, що були використані під час розробки додатку.

Таблиця 4.1 – Використані компоненти

Назва	Призначення
1	2
OVR Camera Rig	Призначений для відображення зображення у шоломі віртуальної реальності
OVR Manager	Дозволяє нааштувати «поведінку» шолому, наприклад, мінімальний та максимальний масштаб рендерингу, тип відстежування, тощо.
OVR Headset Emulator	Емулятор шолому
Linked Alias Assosiation	Використовується для зіставлення потрібних камер та якорів з елементами відстеження.
OVR Anchor Velocity Estimator	Використовується для більш реалістичного відображення поведінки предметів, вираховує швидкість прискорення при падінні.
Axis Rotator Facade	Обертання об'єктів
Tracked Alias Façade	Утворює систему відстеження шолому та контроллерів
Teleporter Facade	Дозволяє налаштувати зону телепортації
Pointer Facade	Дозволяє створити телепортаційний вказівник(промінь)

Продовження таблиці 4.1

1	2
Box Collider	Задає поведінку об'єкту кубічної форми при зіткненні з іншими об'єктами
Sphere Collider	Задає поведінку об'єкту сферичної форми при зіткненні з іншими об'єктами
Rigidbody	Задає об'єкту поведінку твердого тіла
Physic Material	Використовується для налаштування ефектів тертя і відскакування об'єктів при зіткненнях.

Після створення ігрової карти та налаштування камери, було реалізовано скрипти, що спрацьовують при натисканні на певні кнопки контролера. Скрипти та їх призначення перераховані у таблиці 4.2.

Також було розроблено клас Task, що має такі поля, як назва (taskName), стан (status), а також колбек функція що відстежує виконання завдання (taskTracking).

Таблиця 4.2 – Створені скрипти для маніпулювання середовищем

Назва	Призначення
1	2
OVR Input Touch Action	Спрацьовує при дотику до перших кнопок. Дозволяє обрати контролер та кнопку при якій він буде спрацьовувати.
OVR Input Button Action	Спрацьовує при натисканні на першу кнопку. Дозволяє обрати контролер та кнопку при якій він буде спрацьовувати.

Продовження таблиці 4.1

1	2
OVR Input Axis 2D Action	Дозволяє працювати з вісью, керуючи значеннями X та Y. Використовується для задання позиції.

Основним двигуном системи завдань є створений скрипт TaskManager. Він відпрацьовує одразу при запуску застосунку. Клас TaskManager наслідує MonoBehaviour, а тому має два основних методи – Start та Update. Метод Start відпрацьовує один раз, при першому запуску скрипту. В цьому методі створюється список завдань для пацієнта, та задається перша, поточна задача. TaskManager має поля tasks типу List<Task>, що являє собою список завдань, currentTask типу Task – що відображає поточну задачу, count типу int, що використовується для підрахування кількості раз.

У методі Update, що спарцьовує на кожен кадр, викликається callback функція поточної задачі, що перевіряє виконання дії кристувачем. У разі правильного виконання кожна така функція повертає значення true. У разі якщо колбек функція повернула значення потрібну кількість разів, поточна задача помічається, як виповнена та «закривається». TaskManager встановлює наступну задачу у статус поточної.

Для відображення поточних задач, було розроблено UI (User Interfase). Це ігровий об'єкт, що знаходиться на сцені. Виконані завдання відображаються в самому низу, мають приглушений колір тексту, та помітку. Поточне завдання відображається вгорі. Користувач не може перейти до наступних завдань, доки не виконає поточне.

Для розробки UI було використано ігровий об'єкт Canvas. Він дозволяє розмістити в собі усі об'єкти UI: панелі, кнопки, малюнки, текстові об'єкти. Під час створення UI було використано панель, текстові об'єкти, та чекбокс (Toggle).

Кожен текстовий об'єкт відображає задачу в списку, а саме її назву – Task.Name. У чекбоксі поряд з текстовим об'єктом відображається статус задачі (рис 4.5).

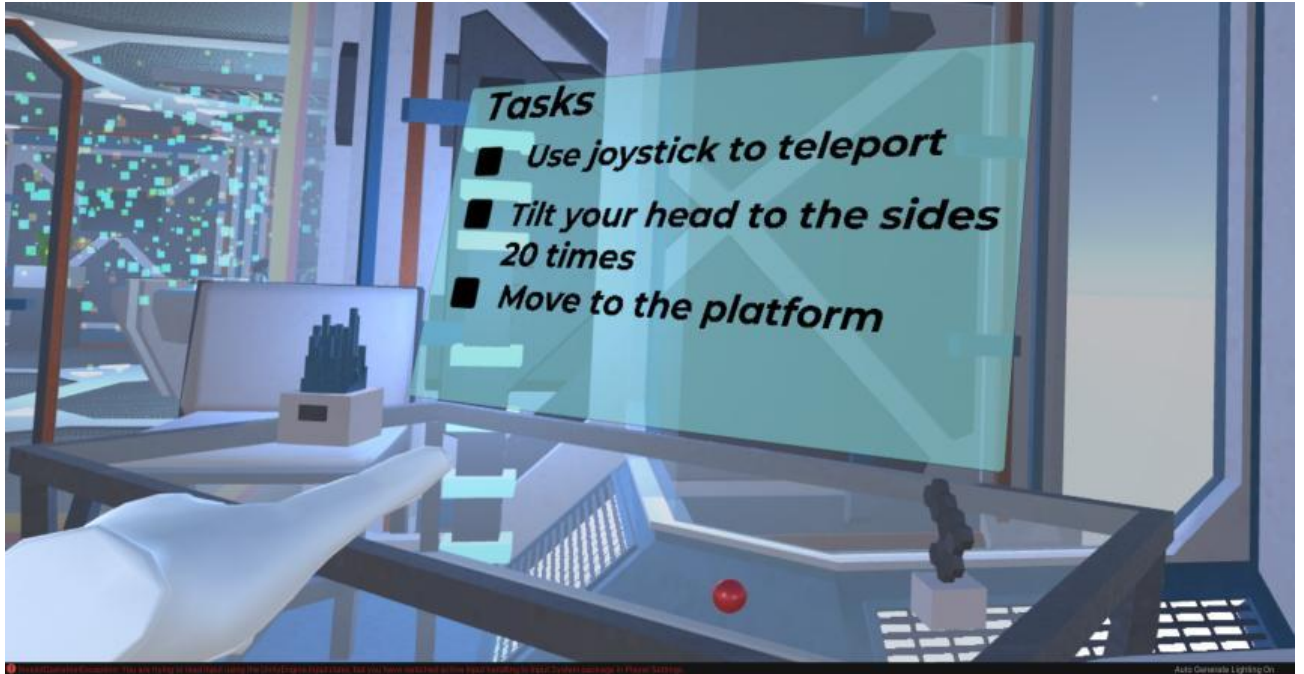


Рисунок 4.5 – UI для відображення задач

Для телепортації використовувались ігрові об'єкти Teleporter.Instant, TeleportCurve.R та TeleportCurve.L (для правого та лівого контролерів). За допомогою цих об'єктів вдалось створити промінь телепортації, що дає можливість, за допомогою джойстику, обрати напрямок телепортації. (рис. 4.6).

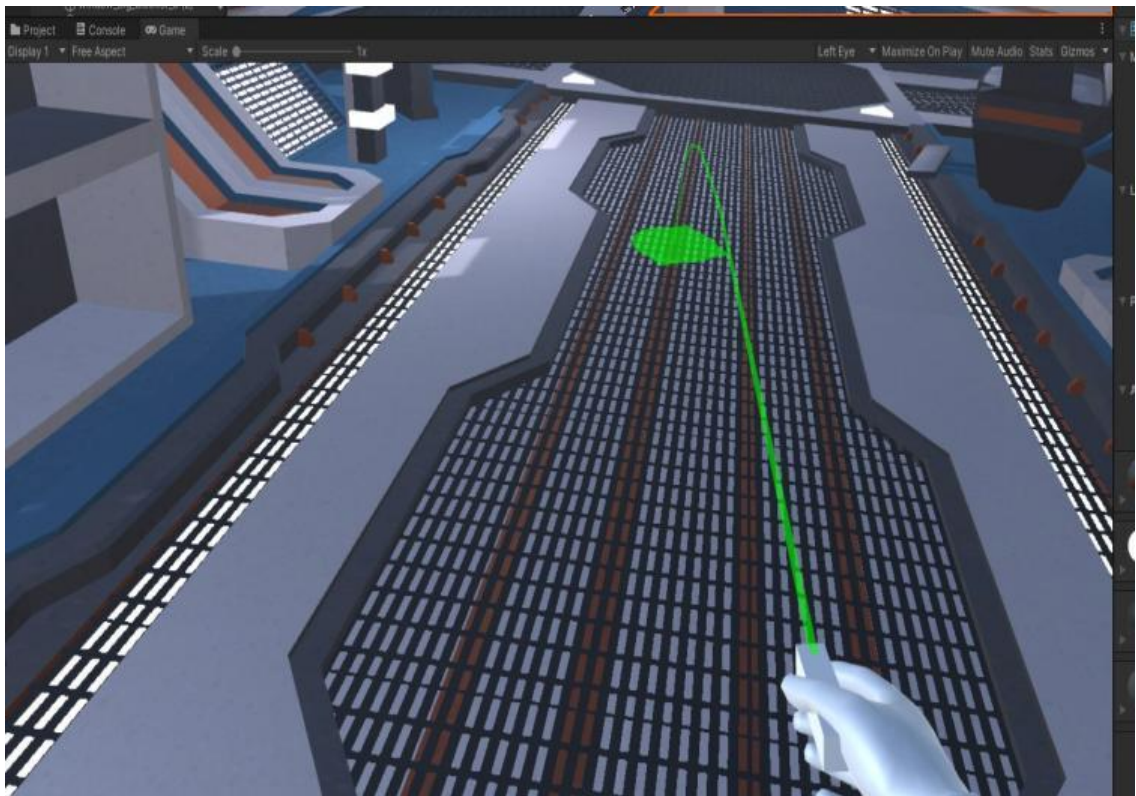


Рисунок 4.6 – Промінь телепортації

ВИСНОВКИ

Під час виконання роботи було виконано огляд та порівняння реабілітаційних напрямків та методів фізичної реабілітації.

Також було проведено огляд сучасних програм-аналогів для фізичної реабілітації людей після хвороб та травм. Характерною Особливістю цих програм є те, що усі вправи відбуваються у якості гри або повсякденних задач.

Для розробки VR-додатку було проаналізовано ігрові движки Unity та Unreal Engine 4, та обрано Unity, через її певні переваги, а саме: велика кількість Asset-ів а також використання мови програмування C#.

Розроблений програмний продукт надає можливість користувачам проводити цікаві тренування, з відстеженням правильності виконання вправ, взаємодіяти з об'єктами на "сцені". Застосунок має користувацький інтерфейс, відображає список завдань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Методичні рекомендації для викладачів на медичному факультеті з дисципліни «Медична реабілітація» [Електрон. ресурс] – Режим доступу: http://kaf-fis-reab.dsmu.edu.ua/?page_id=445.
2. EuroMed: Лікувальна фізкультура [Електроний ресурс] – Режим доступу: <https://euromed.ua/ua/article/travmatologiya-ortopediya/lechebnaya-fizkultura>.
3. Порухнення опорно-рухового апарату. [Електрон. ресурс] – Режим доступу: http://dergachi-irc.kh.sch.in.ua/rekomendacii/blog_vchitelya-reabilitologa/porushennya_oporno-ruhovogo_aparatu_ta_ih_profilaktika.
4. Фізіотерапія. [Електрон. ресурс] – Режим доступу: <https://rodovid.center.ua/direction/physiotherapy>.
5. Фізіотерапевтичне відділення [Електрон. ресурс] – Режим доступу: <https://cimt.com.ua/%D0%B1%D0%B5%D0%B7-%D1%80%D1%83%D0%B1%D1%80%D0%B8%D0%BA%D0%B8/fizioterapev-tichne-viddilennja>.
6. Oculus_Rift. [Электронный ресурс] – Режим доступа: wikiqube.net/wiki/Oculus_Rift.
7. Настройка Oculus Rist [Электронный ресурс] – Режим доступа: <https://www.oculus.com/setup/#rift-setup>.
8. Unity3D или Unreal Engine 4. <https://stfalcon.com/ru/blog/post/unity3d-vs-unreal-engine-4>.
9. The Lightweight Render Pipeline: Optimizing Real Time Performance [Electronic Resource] – Access mode: <https://blogs.unity3d.com/ru/2018/02/21/the-lightweight-render-pipeline-optimizing-real-time-performance>.
10. HDRP: Out of Preview in 2019.3. [Electronic Resource] – Access mode: https://blogs.unity3d.com/2020/02/24/hdrp-out-of-preview-in-2019-3/?_ga=2.102501249.1425026701.1620929117-1280814297.1620929117.

11. Unity Plans and Pricing. Electronic Resource] – Access mode:
<https://store.unity.com/#plans-individual>.

ДОДАТОК А
Текст програми

A.1 Код з файлу OVRInputTouchAction.cs

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

public class OVRInputTouchAction : BooleanAction
{
    public OVRInput.Controller controller =
OVRInput.Controller.Active;
    public OVRInput.Touch touch;

    void Update()
    {
        Receive(OVRInput.Get(touch, controller));
    }
}
```

A.2 Код з файлу OVRInputButtonAction.cs

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

public class OVRInputButtonAction : BooleanAction
{
    public OVRInput.Controller controller =
OVRInput.Controller.Active;
    public OVRInput.Button button;

    void Update()
    {
        Receive(OVRInput.Get(button, controller));
    }
}
```

A.3 Код з файлу OVRInputAxis2DAction.cs

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

```

using UnityEngine;
using Zinnia.Action;

public class OVRInputAxis2DAction : Vector2Action
{
    public OVRInput.Controller controller =
OVRInput.Controller.Active;
    public OVRInput.Axis2D axis;

    void Update() {
        Receive(OVRInput.Get(axis, controller));
    }
}

```

A.4 Код з файлу Task.cs

```

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

public class Task
{
    private bool _taskState;
    private int _countNeed;
    private string _taskName;
    private Func<bool> _callbackFunc;
    public Func<bool> CallbackFunc
    {
        get
        {
            return _callbackFunc;
        }
    }
    public bool State
    {
        get
        {
            return _taskState;
        }
        set
        {
            _taskState = value;
        }
    }
}

```

```

    }
    public string TaskName
    {
        get
        {
            return _taskName;
        }
    }

    public int CountNeed
    {
        get
        {
            return _countNeed;
        }
    }

    public Task()
    {
    }

    public Task(string taskName, int countNeed,
Func<bool> callbackFunc, bool state = false)
    {
        this._countNeed = countNeed;
        this._taskName = taskName;
        this._taskState = state;
        this._callbackFunc = callbackFunc;
    }
}

```

A.5 Код з файлу TaskManager.cs

```

using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.VR;

```

```

public class TaskManager : MonoBehaviour
{
    [SerializeField] Transform taskPanel;
    [SerializeField] GameObject taskToggle;
    [SerializeField] GameObject userText;
    private List<Task> tasks;
    private Task currentTask;
    private int count;

    // Start is called before the first frame update
    void Start()
    {
        this.tasks = new List<Task>();
        this.tasks.Add(new Task("Use joystick to
teleport", 1, TrackerList.trakingTaskUseTeleport));
        this.tasks.Add(new Task("Tilt your head to the
sides", 15, TrackerList.trakingTiltHead));
        this.tasks.Add(new Task("Head tilts forward
backward", 15,
TrackerList.trakingTiltHeadForwardBackward));
        this.tasks.Add(new Task("Raise your hands to the
sides", 10, TrackerList.trakingRiseHands));
        setTasksPanel();
        currentTask = tasks[0];
        count = 0;

        userText.GetComponentInChildren<Text>().text =
currentTask.TaskName + "\nCount:
"+Convert.ToString(count)+"\\"+Convert.ToString(current
Task.CountNeed);
    }
}

```

```

        // Update is called once per frame
        void Update() {
            if (currentTask != null &&
currentTask.CallbackFunc()) {
                this.count++;

                userText.GetComponentInChildren<Text>().text =
currentTask.TaskName+ "\nCount:
"+Convert.ToString(count)+"\\"+Convert.ToString(current
Task.CountNeed);

                if (this.count ==
this.currentTask.CountNeed) {
                    this.currentTask.State = true;
                    sortTasks();
                    setTasksPanel();
                    this.currentTask = this.tasks[0];
                    this.count = 0;

userText.GetComponentInChildren<Text>().text =
currentTask.TaskName+ "\nCount:
"+Convert.ToString(count)+"\\"+Convert.ToString(current
Task.CountNeed);

                    if(currentTask.State == true) {currentTask
= null;

userText.GetComponentInChildren<Text>().text = "";
                }
            }
        }
    }
}

```

```

    }

    void setTasksPanel() {
        foreach (Transform child in taskPanel)
            GameObject.Destroy(child.gameObject);

        for (int i=0; i<3; i++){
            GameObject toggle = (GameObject)
Instantiate(taskToggle, taskPanel);
            toggle.GetComponentInChildren<Text>().text =
tasks[i].TaskName;
            if(tasks[i].State == true)
toggle.GetComponent<Toggle>().isOn = true;
            Debug.Log(tasks[i].TaskName);
        }
    }

    void sortTasks() {
        tasks.Sort(delegate(Task x, Task y)
        {
            if (x.State == false && y.State == false)
return 0;
            else if (x.State == false) return -1;
            else if (y.State == false) return 1;
            else return x.State.CompareTo(y.State);
        });
    }
}
}

```

A.6 Код з файлу TrackerList.cs

```

    using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.VR;

public static class TrackerList
{
    public static bool finalPosition;
    public static bool startPosition;
    public static bool trackingTiltHeadForwardBackward()
    {
        Debug.Log("Rotation:
"+(Vector3)UnityEngine.XR.InputTracking.GetLocalRotation
n (UnityEngine.XR.XRNode.CenterEye).eulerAngles);
        if (startPosition &&
UnityEngine.XR.InputTracking.GetLocalRotation
(UnityEngine.XR.XRNode.CenterEye).eulerAngles.x < 5.0)
            finalPosition = true;
        if
(UnityEngine.XR.InputTracking.GetLocalRotation
(UnityEngine.XR.XRNode.CenterEye).eulerAngles.x >
300.0)
            startPosition = true;
        if(finalPosition && startPosition){
            Debug.Log("COOL!");
            finalPosition = false;

```

```

        startPosition = false;
        return true;
    }
    return false;
}

public static bool trackingTiltHead()
{
    Debug.Log("Rotation:
"+(Vector3)UnityEngine.XR.InputTracking.GetLocalRotation
n (UnityEngine.XR.XRNode.CenterEye).eulerAngles);

    if (startPosition &&
UnityEngine.XR.InputTracking.GetLocalRotation
(UnityEngine.XR.XRNode.CenterEye).eulerAngles.z < 10.0)
        finalPosition = true;
    if (UnityEngine.XR.InputTracking.GetLocalRotation
(UnityEngine.XR.XRNode.CenterEye).eulerAngles.z >
300.0)
        startPosition = true;
    if(finalPosition && startPosition){
        finalPosition = false;
        startPosition = false;
        return true;
    }
    return false;
}

public static bool trackingRiseHands()
{
    float angleLeft =
Vector3.Angle((Vector3)OVRInput.GetLocalControllerPosit

```

```

ion(OVRInput.Controller.LTouch),
(Vector3)UnityEngine.XR.InputTracking.GetLocalPosition(
UnityEngine.XR.XRNode.CenterEye));
    float angleRight =
Vector3.Angle((Vector3)OVRInput.GetLocalControllerPosit
ion(OVRInput.Controller.RTouch),
(Vector3)UnityEngine.XR.InputTracking.GetLocalPosition(
UnityEngine.XR.XRNode.CenterEye));
    Debug.Log("angleLeft: "+ angleLeft);
    Debug.Log("angleRight: " + angleRight);
    if (angleLeft > 6.5&& angleRight > 6.5)
        startPosition = true;
    if (startPosition && angleLeft < 5.0&& angleRight
< 5.0)
        finalPosition = true;
    if(finalPosition && startPosition){
        Debug.Log("COOL!");
        finalPosition = false;
        startPosition = false;
        return true;
    }
    return false;
}
}

```

A.7 Код з файлу AxisRotatorConfigurator.cs

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Zinnia.Action;

```

```
public class OVRInputAxis1DAction : FloatAction
{
    public OVRInput.Controller controller =
OVRInput.Controller.Active;
    public OVRInput.Axis1D axis;

    void Update() {
        Receive(OVRInput.Get(axis, controller));
    }
}
```

ДОДАТОК Б
Слайди презентації

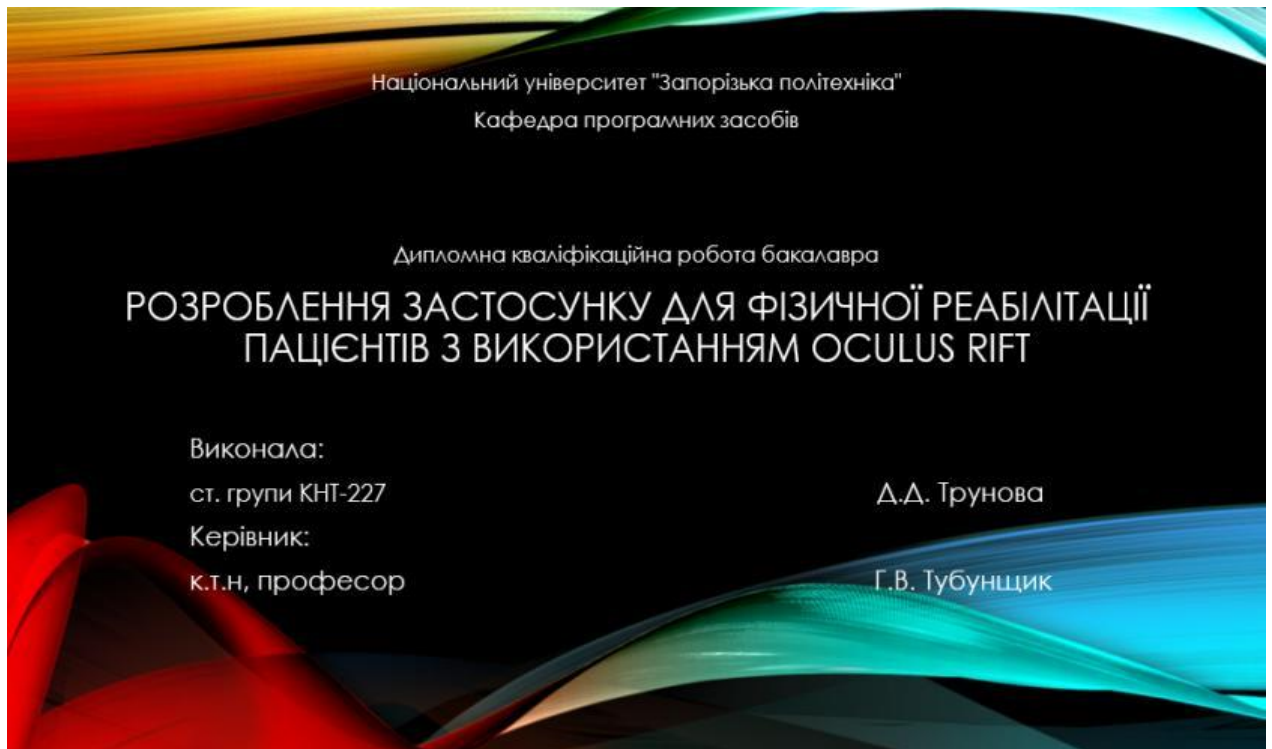


Рисунок Б.1 – Слайд 1

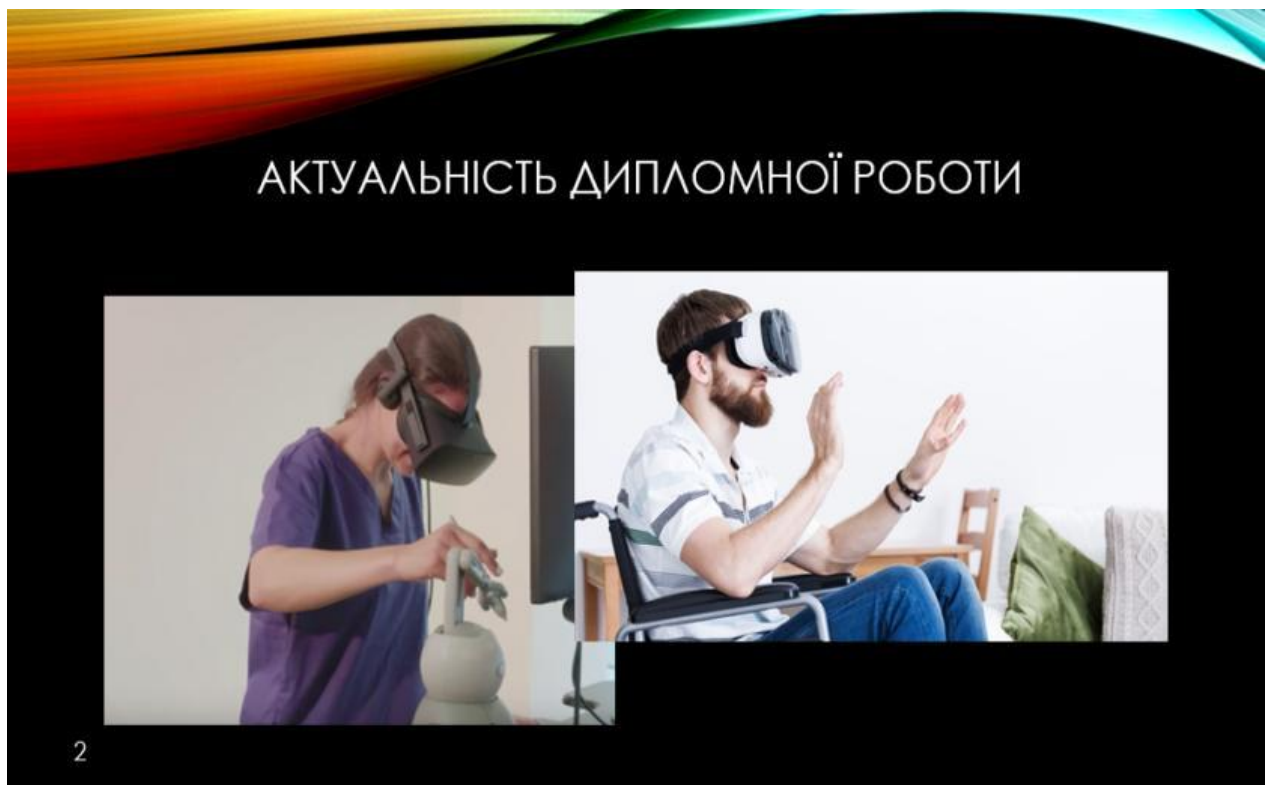


Рисунок Б.2 – Слайд 2

МЕТА ТА ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

Метою даної роботи є розробка та програмна реалізація реабілітаційного застосунку, що дозволяє слідкувати за правильністю виконання вправ пацієнтом.

Для досягнення мети, в роботі були вирішені наступні **задачі**:

- розглянуто існуючі методи реабілітації людини після тяжких захворювань;
- проаналізовані методи фізичної реабілітації;
- розроблено віртуальне середовище програми;
- реалізовано систему пересування по віртуальному середовищу, та взаємодії з предметами;
- розроблено методи відстеження дій користувача.

3

Рисунок Б.3 – Слайд 3

АНАЛІЗ ІГРОВИХ РУШІЇВ

	Unity	Unreal Engine
Мова програмування	C# / JavaScript	C++
Наявність системи візуального скриптингу	-	+
Белика кількість ассетів в Asset Store	+	-
Безкоштовне використання	+	+

4

Рисунок Б.4 – Слайд 4

ДАТЧИКИ OCULUS RIFT

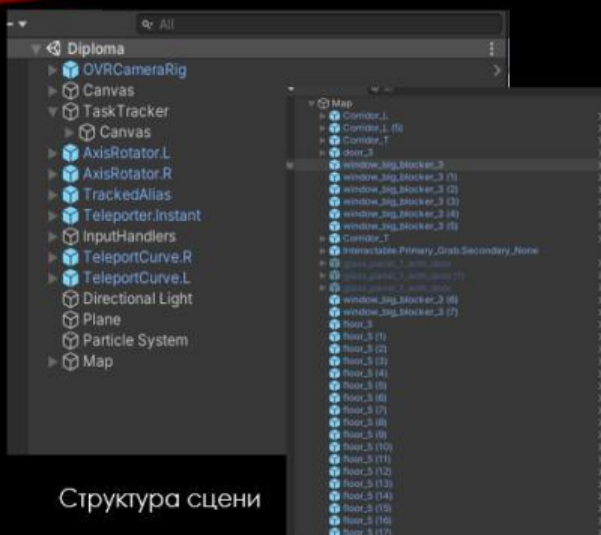
- Гіроскоп - датчик положення у просторі, відстежує положення голови та рук у віртуальному середовищі.
- Акселерометр - датчик переміщення у просторі.
- Магнітометр - з його допомогою можна наводити погляд на об'єкт у віртуальній реальності (наприклад, кнопку меню) і, утримуючи погляд на ньому кілька секунд, активувати його.



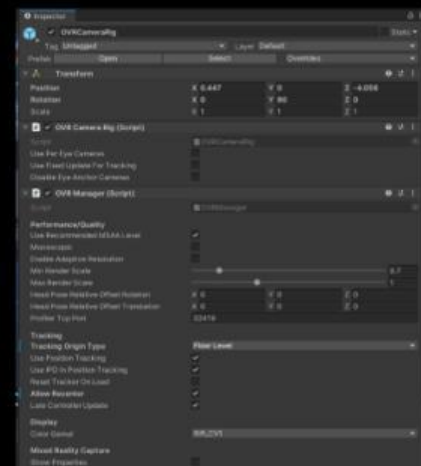
5

Рисунок Б.5 – Слайд 5

СТРУКТУРА ПРОЕКТУ



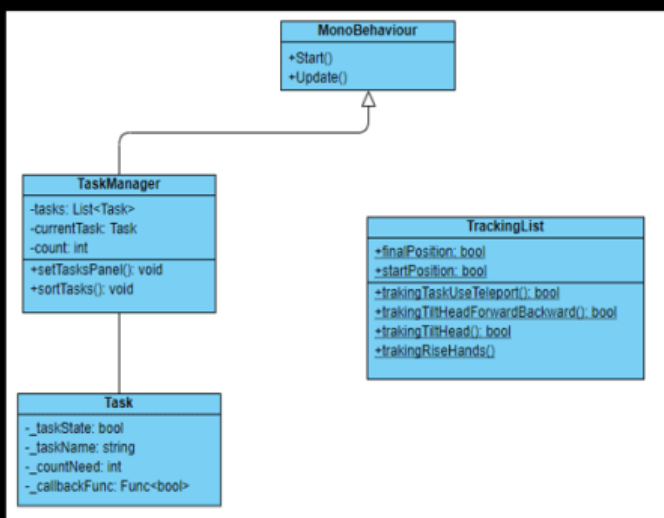
Структура сцени

Компоненти ігрового об'єкту
OVR Camera Rig

6

Рисунок Б.6 – Слайд 6

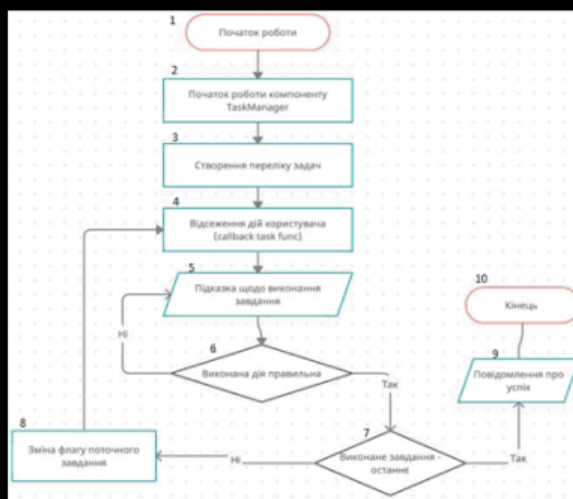
ДІАГРАМА СТВОРЕНИХ КЛАСІВ



7

Рисунок Б.7 – Слайд 7

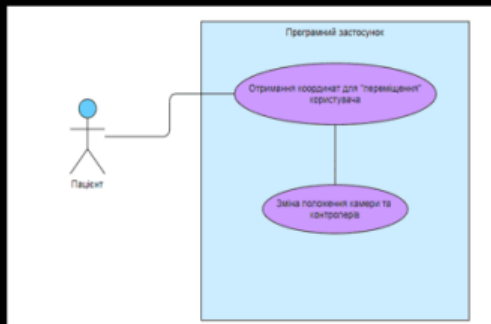
АЛГОРИТМ ФУНКЦІОНУВАННЯ



8

Рисунок Б.8 – Слайд 8

СЦЕНАРІЙ ВИКОРИСТАННЯ



Діаграма прецедентів «Зміна позиції користувача за допомогою телепорту»



Діаграма прецедентів «Відстеження виконання завдання»

9

Рисунок Б.9 – Слайд 9

СЦЕНА ПРОГРАМНОГО ЗАСТОСУНКУ



10

Рисунок Б.10 – Слайд 10

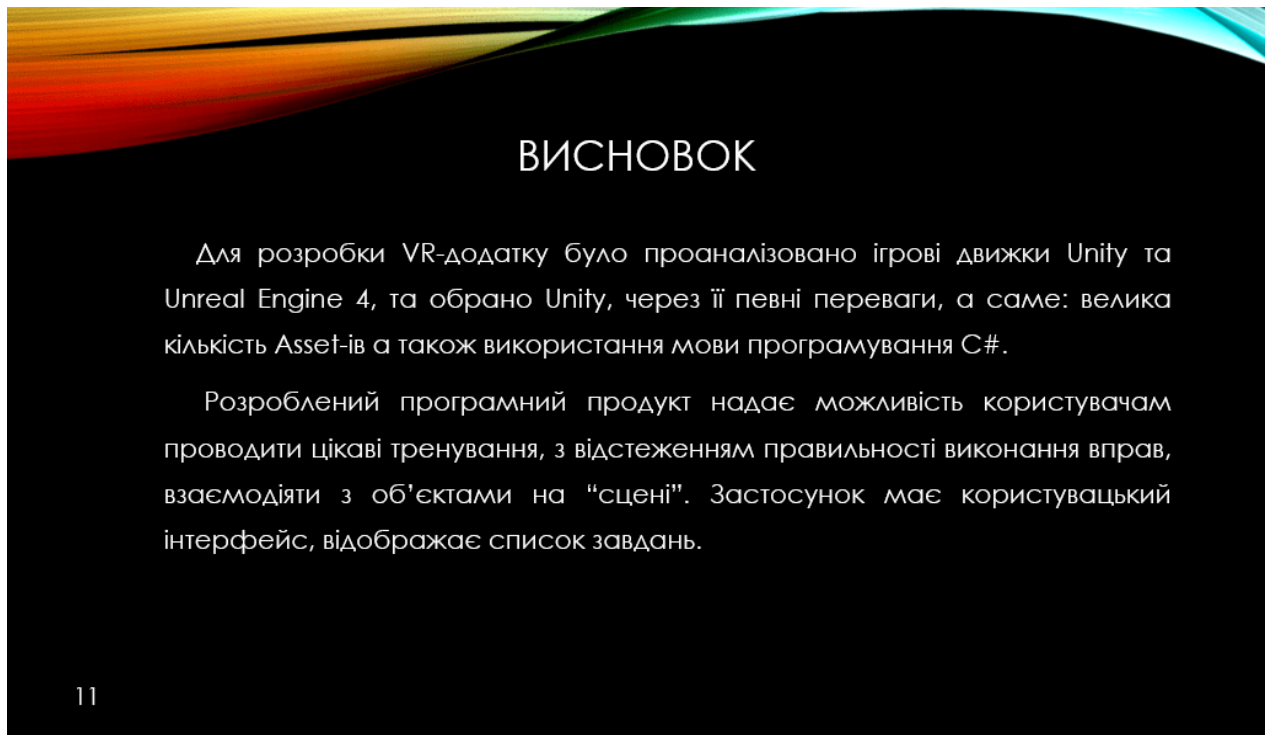


Рисунок Б.11 – Слайд 11

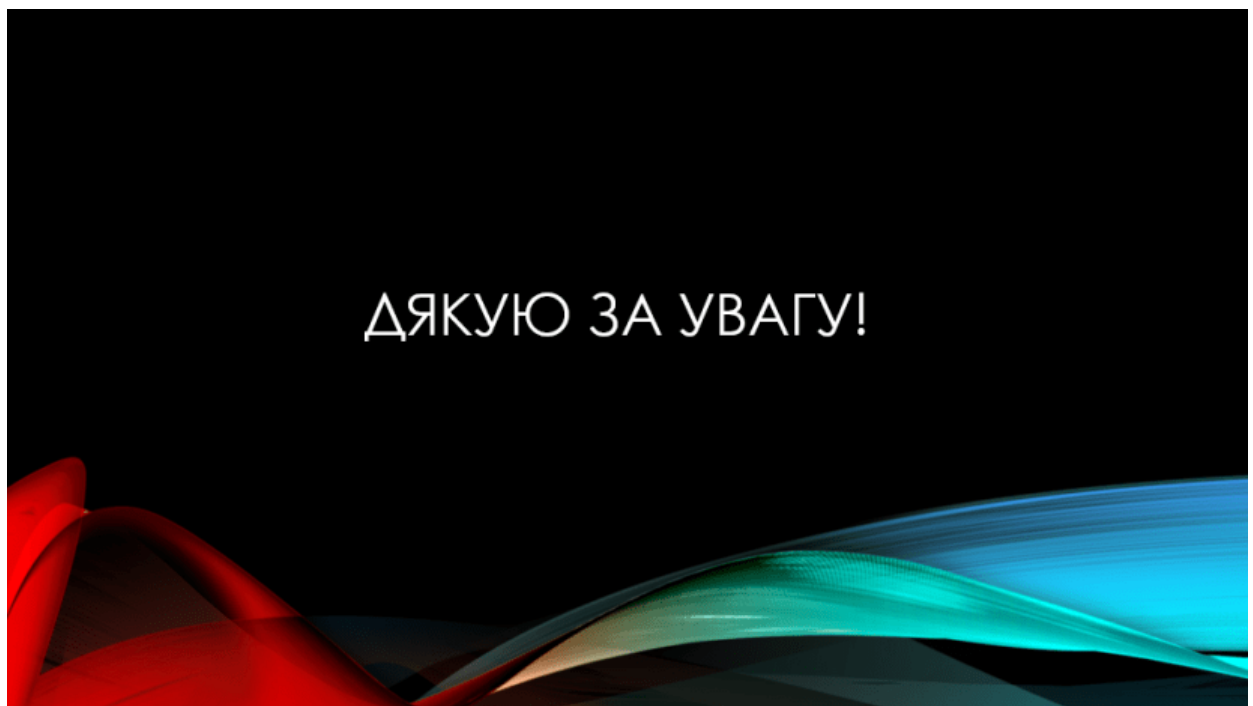


Рисунок Б.12 – Слайд 12