МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Симулятор роботи гаубиці на основі віртуальної реальності»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТм-12 Іванченко Сергій Володимирович

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ « » грудня 2022 р.

Науковий керівник к.т.н., _____ доц. Марченко А.В.
(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Суми-2022

РЕФЕРАТ

Тема кваліфікаційної роботи магістра “Симулятор роботи гаубиці на основі віртуальної реальності”.

Пояснювальна записка складається зі вступу, розділів, висновків, списку використаних джерел, додатків.

Загальний обсяг роботи − 44 сторінки, у тому числі 31 сторінка основного тексту, 2 сторінки списку використаних джерел, 13 сторінок додатків.

Кваліфікаційну роботу магістра присвячено розробці симулятора роботи гаубиці з використанням віртуальної реальності.

В роботі проведено дослідження наявних технологій віртуальної реальності, створення моделі гаубиці в середовищі моделювання Maya 3D. Використано ігровий двигун Unity 3D, та створення на їх основі програми відповідно до завдання.

Результатом проведеної роботи є створення програми відповідно до описаного завдання.

Практичне значення роботи полягає у можливості масової підготовки військових з використання гаубиці без залучення реальної зброї.

Ключові слова: 3D модель, полігональне моделювання, матеріал, текстура, візуалізація, анімація, віртуальна реальність, Unity 3D.

ЗМІСТ

ВСТУП

24 лютого 2022 року Україна зіштовхнулась з небаченим раніше викликом, повномасштабним наступом росії на Україну. Як результат, багато людей загинули, багато втратили місце проживання, майно. Але Українці не здались, були організовані ТРО, які почали боротись з загарбником, тим, що було доступно. ЗСУ використовували запаси зброї виготовлені ще десятки років тому. Прості автомати та вогнепальна зброя неефективна в цій війні, оскільки війська загарбників масово використовували артилерійську зброю, зрівнюючи міста та селища України з землею. Звичайно ЗСУ також використовували артилерію, але нажаль, поповнити снаряди, та ремонтувати артилерію виготовлено ще за часів ссср, було неможливо. Використовувались старі запаси.

Готовність ЄС та США допомогти, надавши артилерійські установки, стикнулась з потребою оперативного навчання військових користуванню останніми.

Дослідження переддипломної практики має на меті розробити рішення проблеми оперативного навчання, за рахунок застосування віртуального спеціалізованого тренажеру з використанням шолому віртуальної реальності HTC Vave та ігрового рушія Unity 3D.

Перевагами впровадження спеціалізованого віртуального тренажера є відсутність потреби пошуку обладнаного полігону та зменшення у десятків разів вартості навчання в порівнянні з використанням реальної гаубиці.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

В останні роки використання тренажерів на основі віртуальної реальності (VR, Virtual Reality) для навчання набуває все більшого поширення, оскільки традиційні методи навчання мають деякі основні недоліки [8]. У військовому контексті цими недоліками, пов’язаними з традиційними методами навчання на основі польових зборів, є висока вартість, складність логістики та просторово-часові обмеження [9]. Стимуляційні середовища на основі VR може допомогти усунути ці недоліки, слугуючи платформою для доповнення сучасних підходів до навчання [8]. Крім того, комп’ютерне моделювання надає більше зручності, доступності і гнучкості [9].

У відкритому доступі систем тренування стрільби з гаубиці на основі віртуальної реальності немає. Але є інші дослідження в цій галузі, наприклад використання доповненої реальності для настільної гри з артилерійської стрільби [1]. Автори пропонують використовувати окуляри доповненої реальності (рис. 1.1), для створення гри, де кожен гравець керує гарматами, для влучання по цілям противника.

Рисунок 1.1 - Ілюстрація компонентів настільної гри

Для поля бою використовується роздрукований QR код, при захопленні якого, окуляри доповненої реальності створюють зображення топографічної карти, де буде відбуватись сам процес гри (рис. 1.2)

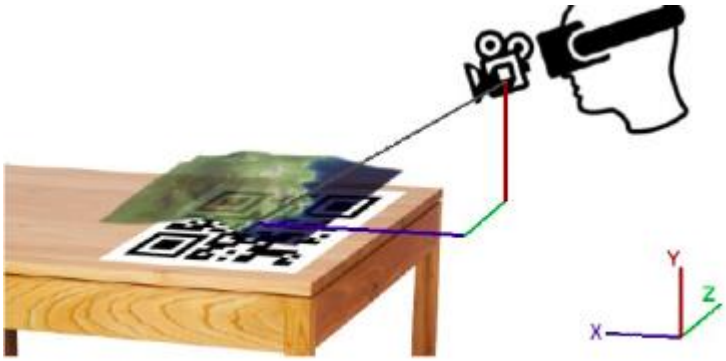

Рисунок 1.2 - Ілюстрація відображення мапи

Дія гри відбувається на 3D-територіях, створених із реальних карт висот. Гравець керує артилерійським підрозділом, гаубицею та може стріляти з нього. Гаубиця була розроблена відповідно до реальної моделі німецької легкої гаубиці, 10,5 cm leFH 18 (рис. 1.4), яка використовувалася під час Другої світової війни.

Рисунок 1.4 - Фотографія гаубиці

Гравець має можливість контролювати орієнтацію ствола гармати та регулювати потужність/тип наступних боєприпасів, якими вона буде стріляти. Це означає, що в грі є кілька типів боєприпасів (снарядів). На місцевості розташовано кілька цільових будівель. Гравець повинен вразити ці цілі, щоб отримати окуляри.

Гра розроблена в Unity3D [7]. Хоча вона розроблена для смарт-окулярів EPSON MoverioBT200, гра може працювати майже на всіх пристроях Android із камерою які підтримують доповнену реальність.

На відміну від VR технологій, даний підхід використовує оточуючі предмети для надання їм певних допоміжних можливостей. Тобто розміщена на столі карта, після обробки комп'ютером, перетворюється для гравця на ігрове тривимірне поле, де можна стріляти з гармат по цілям противника. Нажаль даний підхід не допоможе освоїти використання гаубиці в повноцінному розмірі.

В іншій роботі [2] віртуальне середовище об’єднується з різними датчиками, такими, як інфрачервоний, або акустичний, та використовується для емуляції роботи розумних артилерійських снарядів і створення відповідних 3х вимірних карт на основі карти висот (рис. 1.5). Розраховується та емулюється різна траєкторія руху снаряду (рис. 1.6).

Ця робота доволі цікава, але вона розглядає снаряди окремо від зброї, що використовує цей снаряд. Отже також не допоможе в навчанні майбутніх артилеристів.

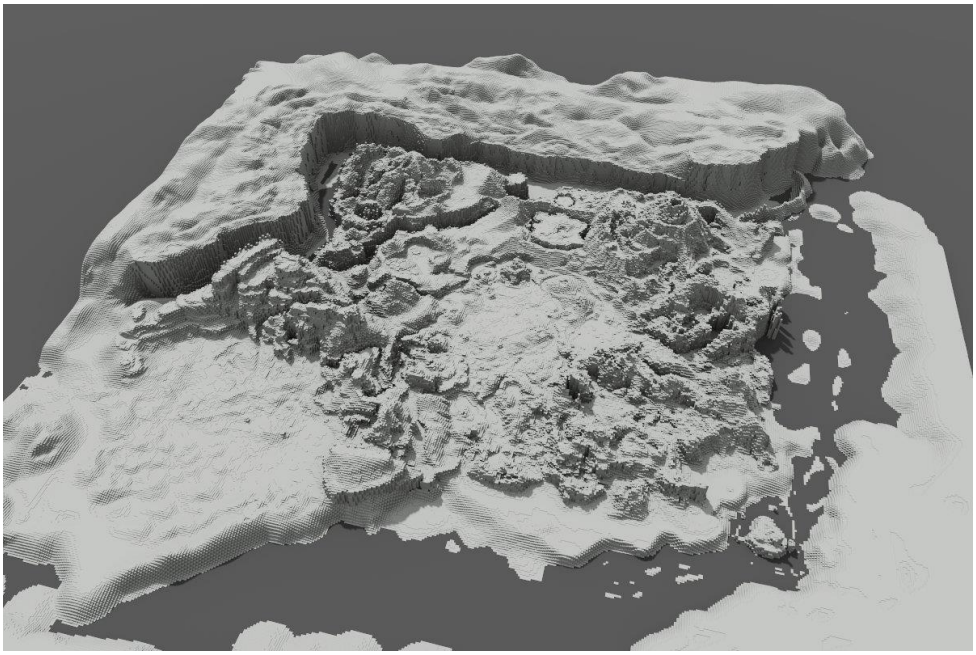

Рисунок 1.5 - Ілюстрація ігрової карти

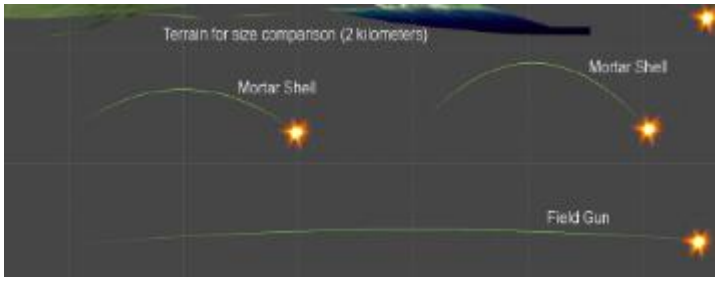

Рисунок 1.6 - Ілюстрація руху снаряду

Також технології віртуальної реальності вже активно та успішно використовуються в армії Бразилії [3], де на основі цієї технології створено цілий фреймворк для навчання корегувальників армійської артилерії. В основі фреймворку є інструктор, які дає команди та керує солдатами, оцінюючи їх дії. Він використовує ноутбук для цих цілей. Та шолом віртуальної реальності (рис. 1.7). Система спілкується між собою за допомогою спеціального протоколу, схема

зображена на рисунку 1.8. Вигляд терміналу інструктора зображений на рисунку 1.9.

Рисунок 1.7 - Апаратна частина фреймворку

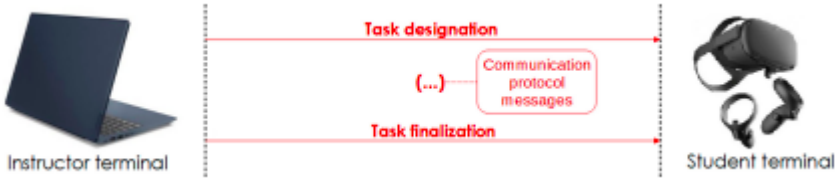

Рисунок 1.8 - Схема комунікації фреймворку

Рисунок 1.9 - Інтерфейс фреймворку

Також є ігрові версії симуляторів [4], але в них використовується спрощені моделі які не дозволяють повноцінно використовувати ігри для навчання.

Аналіз існуючих технологій дозволив визначити такі характеристики розроблюваного віртуального симулятора:

- зрозумілість інтерфейсу,
- гнучкість для розширення,
- генерація результатів навчання.

2. ПОСТАНОВКА ЗАДАЧІ

2.1 Мета та задачі дослідження

Метою дослідження є створення віртуального середовища для навчання військових користуватись гаубичною артилерією, не використовуючи при цьому реальні зразки зброї.

Об’єкт роботи це технології віртуального навчального середовища ,які дозволяють опанувати практичні навички використання віртуальних аналогів реальних об’єктів.

Предмет дослідження це віртуальна платформа, яка реалізує процес опанування практичними навичками ефективного використання причепної артилерії, контроль над процесом навчання та отримання результатів для аналізу ефективності навчання.

Для досягнення поставленої мети необхідно виконати такі задачі:

1. Аналіз предметної області, визначення актуальності дослідження, і опис вимог до системи.

2. Аналіз існуючих рішень.

3. Формулювання чітких вимог до створюваної системи.

4. Моделювання системи.

5. Вибір засобів імплементації.

Віртуальний тренажер повинен мати такі функціональні властивості:

− надавати користувачу мінімально необхідні знання в користуванні гаубицями;

− створювати реалістичні відчуття присутності на полігоні, та полегшити користування реальною гаубицею;

– контролювати процес навчання, для визначення придатності користувача до використання реальних зразків зброї.

Для реалізації функціонального призначення віртуального тренажеру необхідно:

– створити систему для побудови віртуального середовища;
– створити систему керування та влучання пострілів в ціль;
– систему побудови зворотного зв’язку для контролю за навчанням;
– створити моделі гаубиці та цілей.

2.2 Вибір засобів реалізації

Зазначений проект складається з двох основних компонент: апаратної та програмної.

Апаратна компонента являє собою окуляри віртуальної реальності, та комп’ютер. Окуляри слугують для відображення віртуального середовища і відповідно сприйняття цього середовища учнем. Для взаємодії з віртуальним світом, використовуються спеціальні маніпулятори. Для обчислення різних процесів всередині віртуального світу, використовується комп’ютер, спеціально адаптований під використання окулярів віртуальної реальності. Окуляри, маніпулятор та комп’ютер взаємодіють між собою за допомогою низькорівневих програм драйверів. Ці програми як правило поставляються разом з окулярами.

Комп’ютер повинен відповідати високим обчислювальним характеристикам, оскільки робота з віртуальною реальністю, доволі ресурсоємка. Крім цього комп’ютер повинен бути оснащений адаптованою під віртуальну реальність відеокартою.

Для емуляції віртуального середовища в даний час існує декілька апаратних версій обладнання, такі як HTC Vave, Oculus Rift, Index One, Sony VR. Всі вони подібні один до одного, та як правило влючають шолом, та маніпулятори, і інколи спеціальні базові станції, які відслідковують рух шолома та маніпуляторів для відображення переміщення в віртуальному середовищі (рис 2.1).

Рисунок 2.1 − Ілюстрація окулярів віртуальної реальності

Для реалізації поставленої мети обрані такі інструментальні засоби HTC Vave Cosmo [5], оскільки його ціна невисока, він має декілька вбудованих камер в шоломі, які використовуються для трекінгу рухів користувача в реальному світі, та відображенні цих рухів на віртуальний світ.

Обладнання для побудови віртуальної реальності лише дозволяє відображати віртуальну реальність, але для побудови та обчислень використовуються спеціалізовані програми. Для створення моделей, найбільш популярні: Maya 3D, 3D Max, 3D Builder, Cinema 4D, та багато інших. Кожна з них

має як свої плюси та мінуси, але в решті решт не впливає на кінцевий результат, і вибір полягає лише в зручності та навичках користувача для створення моделі.

Для досягнення поставленої мети буде використано Maya 3D [6], оскільки маю найбільше досвіду роботи саме з ним.

Також створені моделі повинні бути обернені програмною логікою в віртуальній реальності. Є два шляхи для вирішення цієї задачі. Перший полягає в побудові з чистої сторінки, всього коду, починаючи від взаємодії з драйверами окулярів, і завершуючи парсерами 3D моделей, та різноманітних анімацій. Як середовище для цього можна використовувати Visual Studion та мову C++. Але даний підхід потребує величезних витрат часу, ресурсів, та знань.

Інший шлях, використати ігрові рушії, які уже мають набір низькорівневих інтерфейсів для взаємодії з апаратним забезпеченням, спрощують взаємодію з моделями, налаштування анімації, та навіть мають в собі фізичні рушії.

Найбільш популярні в даний час це Unity 3D та Unreal Engine. Кожен з них має графічний інтерфейс, засоби для роботи з моделями, анімаціями, додавання різноманітних програмних моделей поведінки. Також обидва мають можливість роботи з окулярами віртуальної реальності.

Щодо відмінностей, то, по-перше, Unity 3D для створення програмної логіки може використовувати такі мови як C# та JavaScript, Unreal Engine використовую C++. На мою думку більш потужним є Unreal Engine. Але ця потужність також робить його більш складнішим через специфіку управління пам’яттю в C++. Тож доволі легко отримати складно відслідковуванні помилки. Unity 3D в свою чергу використовує JavaScript, який є дуже розповсюдженою мовою, і використовується в Web програмуванні.

Для вирішення поставленої задачі використана Unity 3D [7], оскільки JavaScript більш розповсюджена, отже простіше знайти спеціалістів для майбутньої підтримки продукту та її використання зменшує кількість помилок під час написання коду.

3. ПРОЕКТУВАННЯ СИМУЛЯТОРА РОБОТИ ГАУБИЦІ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ VR

3.1 Структурно-функціональне моделювання системи

Структурно-функціональне моделювання системи являє собою, створення моделі у вигляді схеми, що описує логіку роботи системи з врахуванням наявної інформації.

Для структурно-функціонального моделювання використовується стандарт IDEF0. Цей графічний стандарт є частиною методології структурного аналізу та проектування. IDEF0 називають методологією графічного опису систем і процесів діяльності організації як безлічі взаємозалежних функцій [10].

Рисунок 3.1 – IDEF0 процесу роботи симулятора роботи гаубиці

Центральним елементом моделі IDEF0 є функціональний блок, він відображає функцію системи. Кожен функціональний блок містить 4 ключові потоки, які мають фіксований напрямок. З лівої сторони відображаються ресурси які потрібні для виконання функції. З правої сторони результат роботи функції.

Зверху — керуюча логіка. Знизу вказується хто і чим виконує функції. [10]

Дані характеристик гаубиці, характеристик снаряду та мапу полігону використовуються системою для отримання результату. Стрілка управління містить фізичну модель яка використовується для розрахунку руху снаряду.

Ігровий рушій керує апаратною частиною, та відображає віртуальний світ.

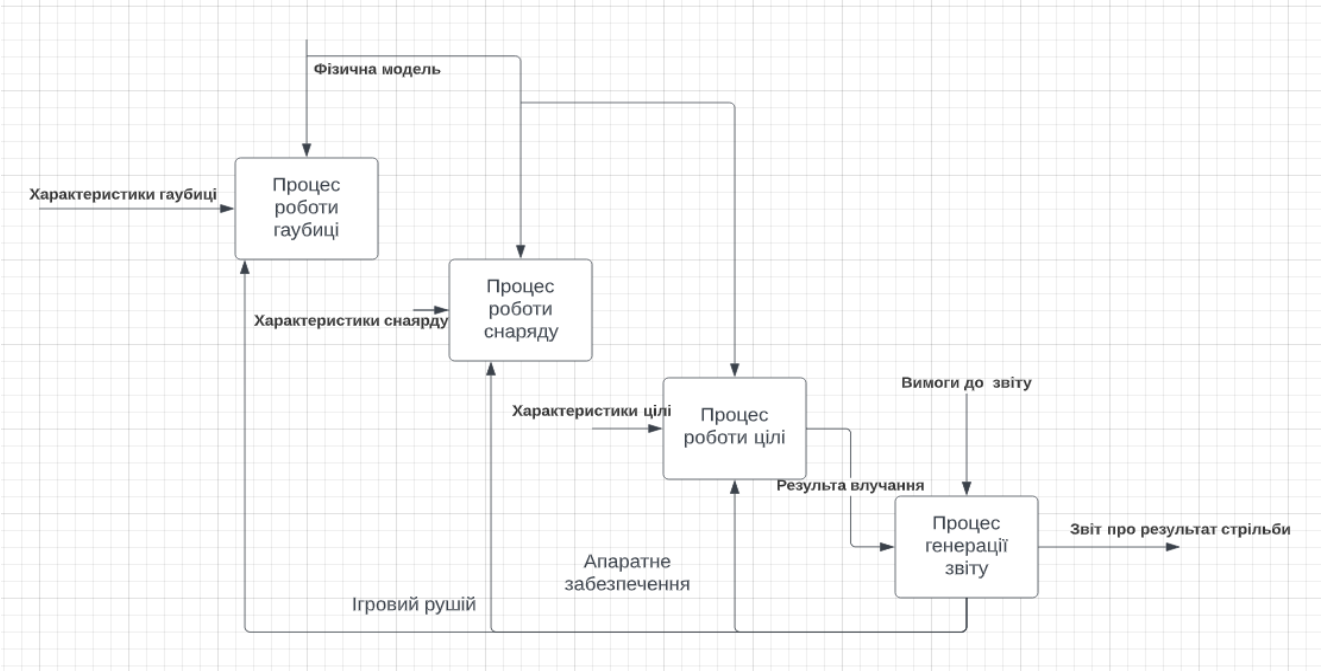

Рисунок 3.2 – Декомпозиція IDEF0-діаграми

Ця діаграма містить наступні процеси:

- поведінку гаубиці;
- поведінку снаряду;

- поведінку цілі на мапі;
- створення звіту по результатам стрільби;

3.2 Моделювання варіантів використання емулятора стрільби з гаубиці

Для полегшення взаємодії учасників проєкту, як один зі стандартів опису бізнес-логіки роботи системи, та, як один з елементів документування, використовують мову графічного опису UML. За допомогою цієї нотації можна визначити, описати процеси, та полегшити проектування системи.[11].

Для визначення сценаріїв роботи системи, використаємо діаграму варіантів використання. Це дасть можливість, створити повний список всіх варіантів використання системи. Вона складається з учасників, так званих акторів, елементів зв’язку, варіантів використання.

Діаграма зображена на рис. 3.3

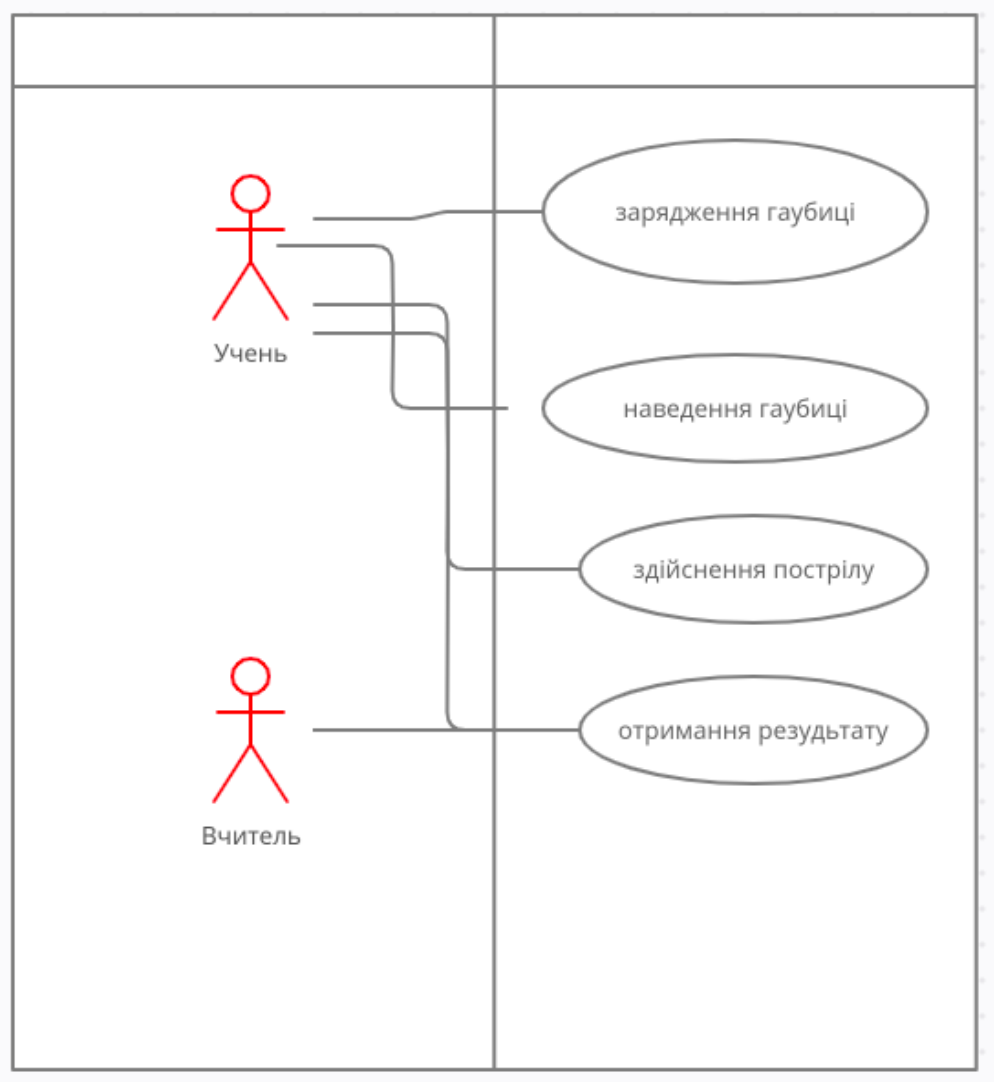

Рисунок 3.3 -Діаграма варіантів використання системи

Варіант використання зарядження гаубиці дозволяє користувачу навчитись заряджати гаубицю.

Варіант використання наведення гаубиці дозволяє користувачу навчитись наводити гаубицю.

Варіант використання здійснення пострілу дозволяє користувачу навчитись здійснювати постріл.

Варіант використання отримання результату дозволяє користувачу отримати результат стрільби.

4. РОЗРОБКА СИМУЛЯТОРА СТРІЛЬБИ З ГАУБИЦІ НА ОСНОВІ ТЕХНОЛОГІЙ VR

4.1 Архітектура системи

Архітектурою називають структурну схему системи та способи взаємодії між елементами системи. Архітектура вміщує в собі елементи, які не можуть бути змінені в майбутньому [23]. Система складається з VR Application, вона вміщує тривимірні моделі, скрипти, які керують логікою системи, налаштування менеджера вводу, та відображення кінцевого результату, вхідні дані для роботи скриптів. Так як при розробці використовується фреймворк Unity 3D, то він накладає свої вимоги до побудови системи, кожен скрипт і модель існують у вигляді окремих об’єктів. Також рушій дозволяє отримувати дані з контролерів, обробляти отримані дані, та побудувати кінцеве зображення передаючи його на шолом віртуальної реальності. Архітектура системи схематично відображена на рис. 4.1.

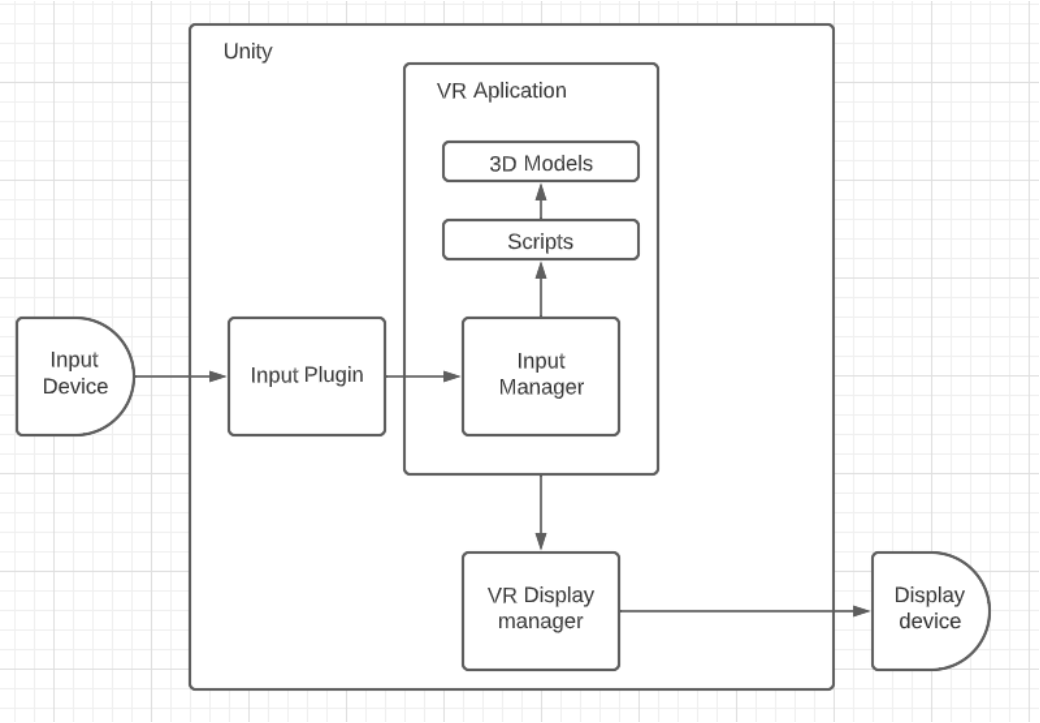

Рисунок 4.1 − Архітектура системи

4.2 Реалізація системи

Першим етапом виконання робіт є створення моделі гаубиці. Для цього використаємо програму для побудови тривимірних моделей Maya. На рис. 4.2 зображено вікно моделювання зі створеною моделлю гаубиці. Після створення моделі, її потрібно експортувати у формат *.3ds для подальшого використання в середовищі Unity. Цей процес зображено на рис. 4.3.

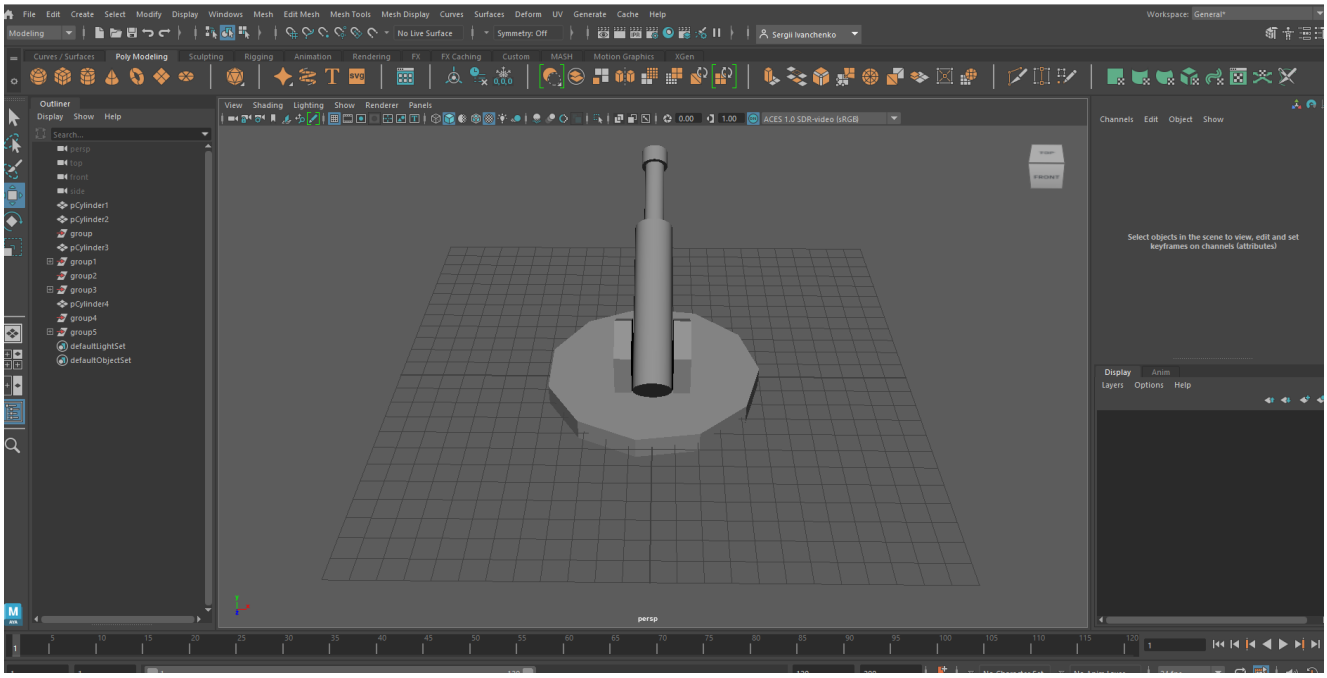

Рисунок 4.2 – Ілюстрація інтерфейсу середовища Maya зі створеною гаубицею

Рисунок 4.3 – Конфігурація плагіну

Наступним кроком є розробка системи в рушії Unity. Процес розбивається на декілька етапів. Перший − підключення і налаштування шолома віртуальної реальності. Для цього встановлюємо плагін SteamVR Plugin [14]. В середовищі Unity створюємо новий проект та приймаємо всі налаштування зображені на рисунку 4.3.

Наступний крок налаштування системи керування, для цього потрібно перейти в конфігурації (рис. 4.4).

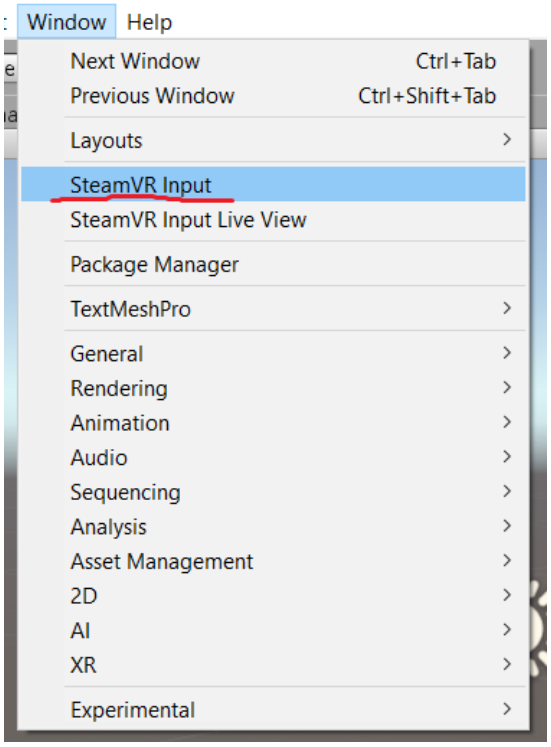

Рисунок 4.4 − Меню конфігурації

У відкритому вікні достатньо натиснути Save and Generate. Це створить новий конфігураційний файл, який в подальшому можна буде змінювати під час роботи програми (рис 4.5).

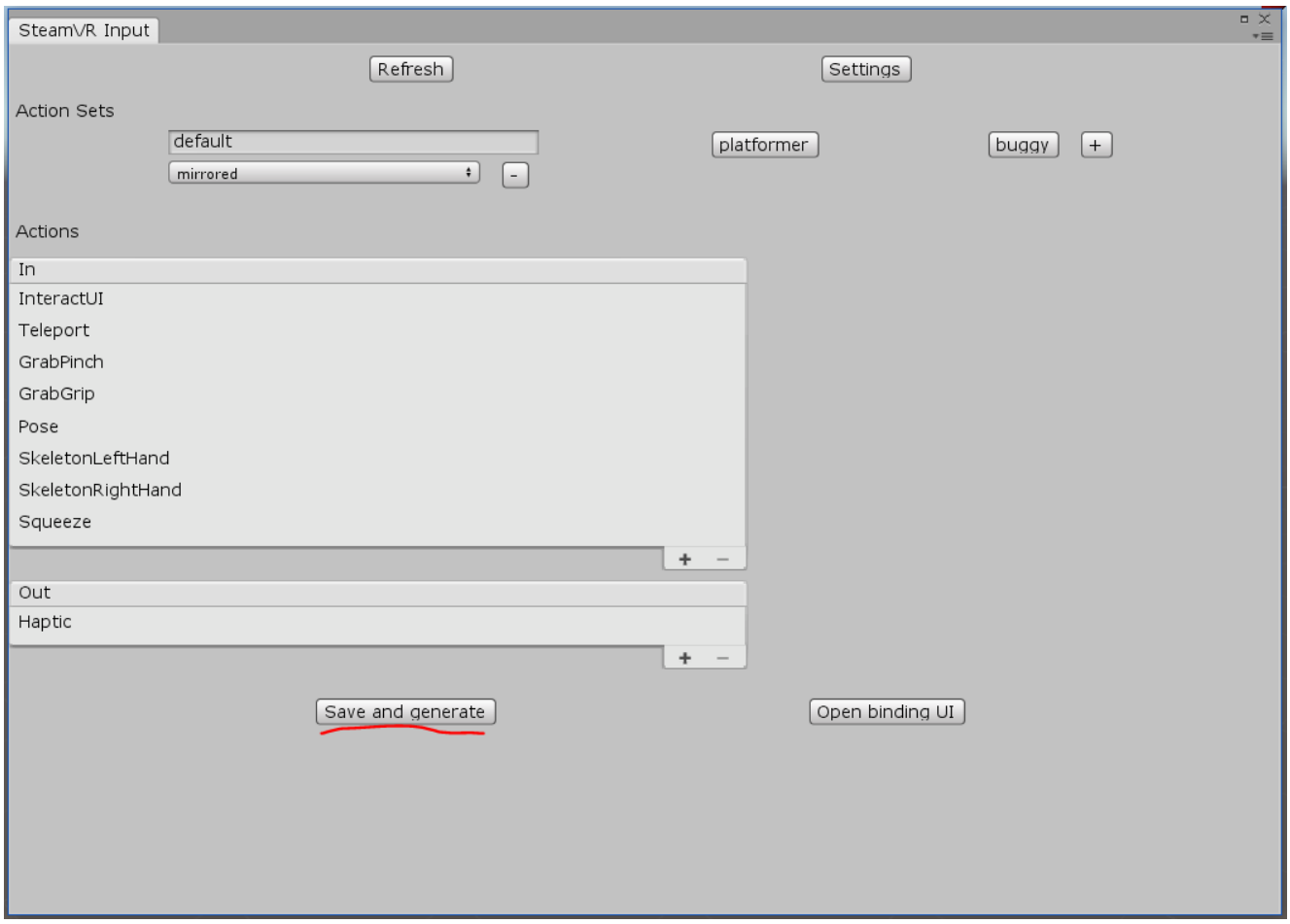

Рисунок 4.5 − Вікно створення конфігурації

Наступним кроком є створення гравця та видалення дефолтної камери оскільки вона буде замінена камерою з плагіну. Щоб додати гравця потрібно з вікна префабів, конфігураційної папки додати префаб гравця (рис 4.6).

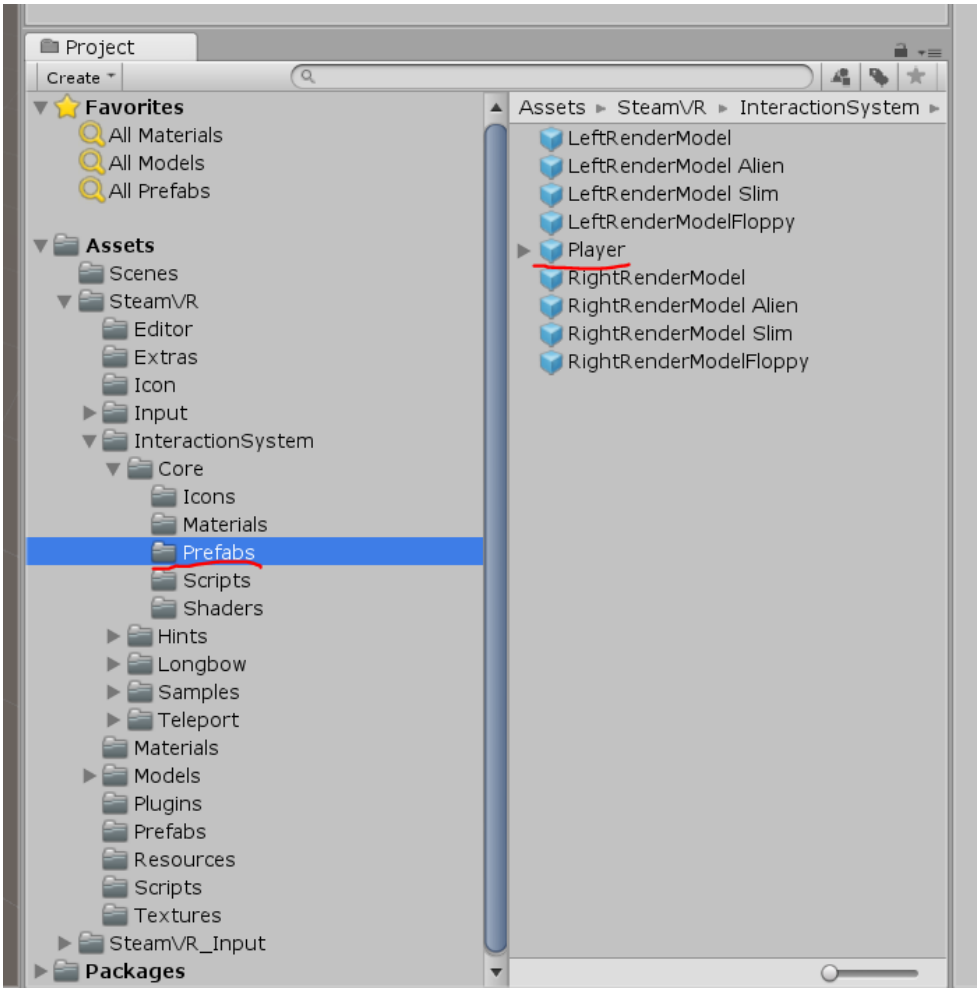

Рисунок 4.6 − Вікно створення гравця.

Після цього в системі будуть створені віртуальні маніпулятори. Але для повноцінної роботи маніпуляторів, потрібно додати відповідний скрипт в Player→NoSteamVRFallbackObjects→FallbackHand (Додаток Б.1)

Наступний етап − імпорт моделі створеної в Maya в середовище Unity. Для цього достатньо перетягнути модель у вікно префабів. І розміщення її на мапі в Unity. Після імпорту моделі це буде статична модель, яка нічого не виконує. Тож для повноцінної роботи приєднуємо до неї відповідний скрипт. Скрипт реагує на взаємодію з відповідною частиною моделі, в нашому випадку таких частин 3.

- перша слугує для калібрування нахилу гаубиці,
- друга керує горизонтальною позицією,
- третя слугує для ініціалізації стрільби.

Для їх побудови, додаєм колайдер на кожну частину. Колайдер - це об’єкт, що надається рушієм, та реагує на взаємодію з гравцем (Додаток Б.2). OnCollisionStay функція слідкує за реакцією та реалізує відповідну реакцію.

В перших двох випадках ми змінюємо позицію гармати. Після ініціації стрільби, систем на основі кутів положення дула гармати, створює префаб снаряду. Снаряд також отримує початкову швидкість та масу. Оскільки рушій Unity має вбудований дуже точний фізичний рушій, то всю роботу по розрахунку траєкторії бере на себе саме він. Нам потрібно створити лише відповідний об’єкт снаряду, визначити його направлення, та задати потрібні фізичні характеристики. Для емуляції реальних характеристик снаряду, таких як різна маса, відхилення через погодні умови, різний початковий заряд порохового заряду, внесемо випадкові похибки в ці характеристики. Під час руху снаряду, відслідковуються координати. У випадку якщо Y координата досягне 0, снаряд знищується. Оскільки досягнутий рівень поверхні. Також після створення снаряду, гравець переміщується в координати над поверхнею, та вимикається дія гравітації. Це здійснюється для фізичного спостереження за рухом снаряду. Мапу буде створено за допомогою інтегрованого інструменту Terrain (рисунок 4.7), який дає можливість створити величезну карту, засадити її віртуальними деревами, та травою для реалістичності. Зображення сконфігурованого терейна на рисунку 4.8.

Для відображення траєкторії руху снаряду, скористаємось компонентом рушія TrailRenderer [16]. Конфігурація TrailRenderer вказана на рисунку 4.9.

Перед останнім етам є створення цілі. Для цього було побудовано модель будинку в Maya, і імпортовано в середовище рушія. Для цієї моделі створюємо бокс колайдер, який буде реагувати на влучання снаряду. Під час взаємодії снаряду з боксом цілі, нараховуємо очки за кожне влучення.

Після закінчення стрільби, виводимо на екран результат влучання за допомогою компонента рушія TextElement, він надає можливість відображати текст на екрані і слугує для створення репорту (рисунок 4.10).

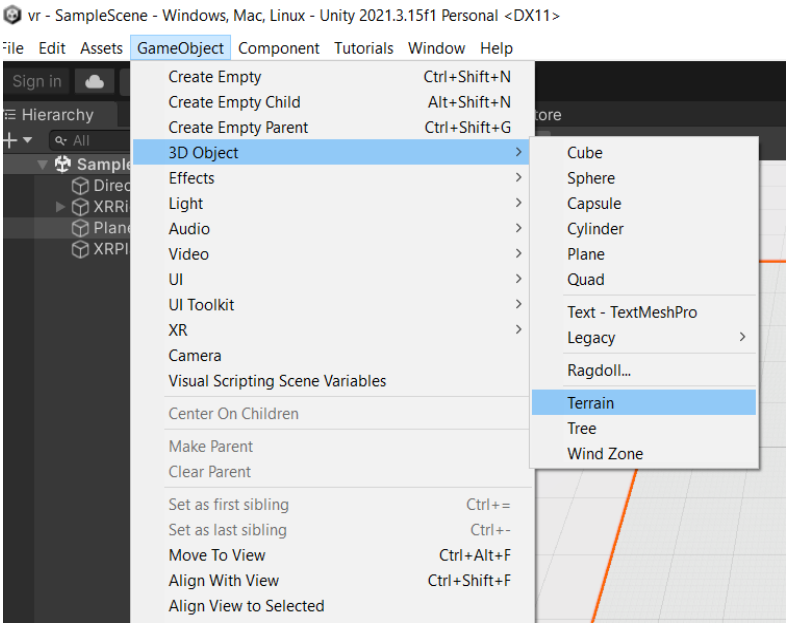

Рисунок 4.7 – Ілюстрація доступу до інструменту Terraine

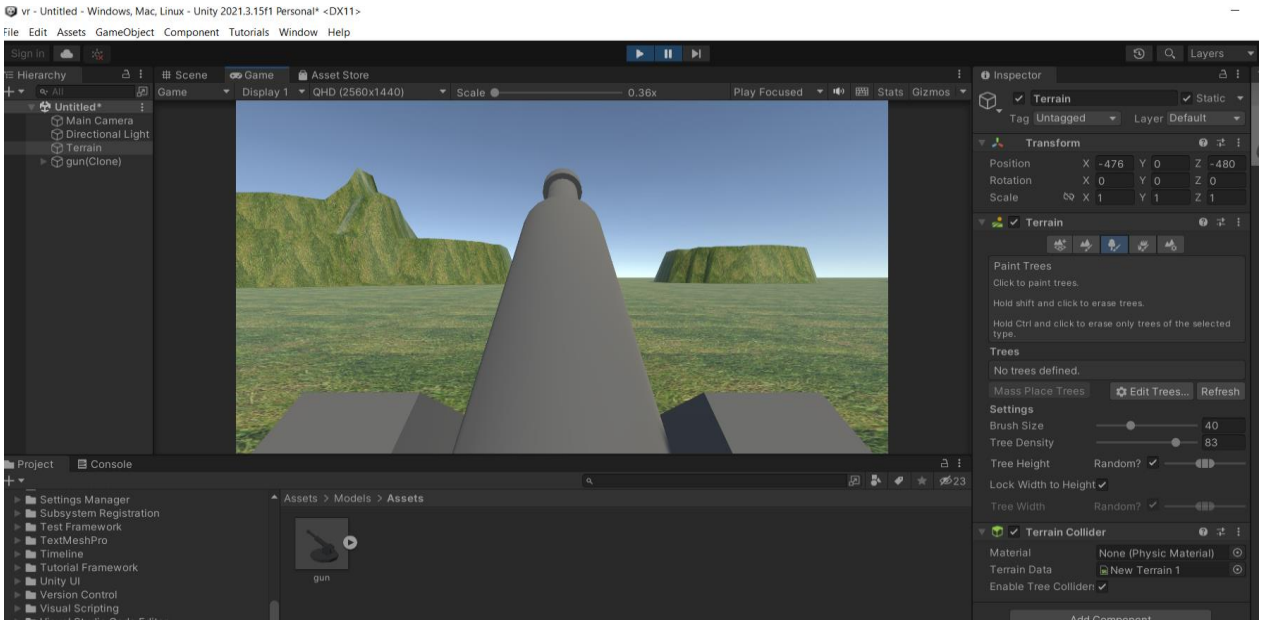

Рисунок 4.8 –Меню Terrain для генерації мапи

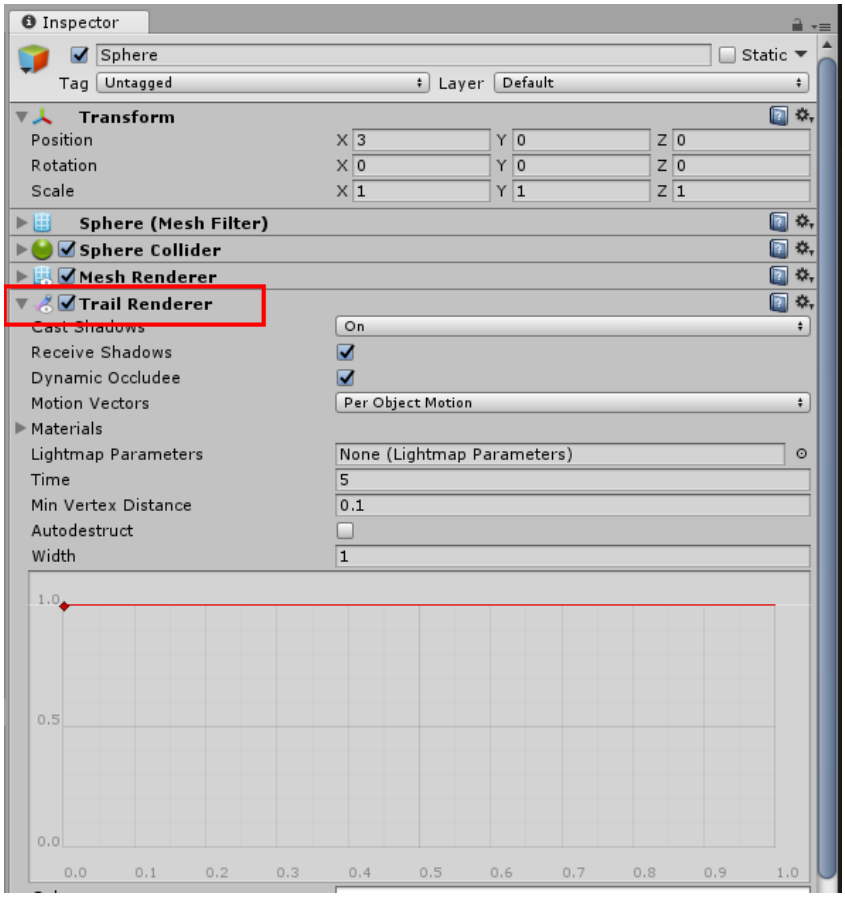

Рисунок 4.9 – Ілюстрація конфігурації компонента TrailRenderer

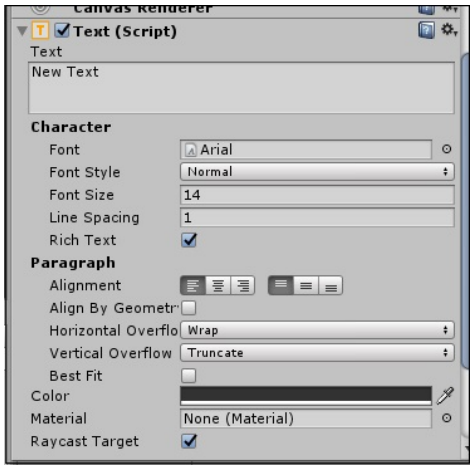

Рисунок 4.10 – Ілюстрація конфігурації TextElement для генерації репорту

ВИСНОВКИ

В наш час цифрові технології дуже глибоко занурились в життя людства. Практично кожна людина має цифрові гаджети, такі як телефони, смарт годинники, та інше. Багато будинків обладнані розумними системами, які завдяки мережі Інтернет, дозволяють керувати будинком з будь-якої точки планети де є Інтернет покриття. Більше того технології постійно розвиваються та рухають прогресом вперед. Швидкість та покриття мережі Інтернет росте. Тож в ці часи віртуальний світ виході на передній план, втілюючи найсміливіші мрії людей в віртуальному просторі. Вже близький час коли люди зможуть побувати там ,де раніше це було неможливо по суті. На інших планетах, на уявних світах. Спуститись в цифрову копію світового океану, чи полетіти до сусідньої галактики. Все це наближається з доступом до віртуального середовища.

Тож потрібно використати подібні можливості для навчання, не тільки для розваг.

Метою дослідження є створення симулятора роботи гаубиці, що в майбутньому може дозволити швидко навчати майбутніх артилеристів, або підвищувати їх кваліфікацію, не використовуючи справжні зразки зброї. Аналіз існуючих технологій дозволив визначити найбільш перспективні, та зручні для виконання даного дослідження.

Результати ІТ проектування представлені діаграмами, матрицею ризиків, графіком роботи.

Для досягнення мети, були використані такі засоби як, середовище тривимірного моделювання Maya, ігровий рушій Unity3D.

Результатом дипломної роботи є створений симулятор роботи гаубиці, який дозволяє навчатись стрільбі без використання реальних зразків зброї:

функціонально керує влучанням пострілів в ціль, система побудови зворотного зв’язку для контролю за навчанням, створені моделі гаубиці та цілей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Yavuz, Mehmet & Sayar, Ridvan & Yilmaz, Bilge & Bostanci, Gazi Erkan & Guzel, Mehmet & Yılmaz, Abdullah. (2019). Desktop Artillery Simulation Using Augmented Reality. 79-85. 10.1109/ICEEE2019.2019.00023.
2. Gozard, Patrick & Bret, Emmanuel & Cathala, Thierry. (2008). Virtual simulation tools for artillery. 10.1117/12.780359.
3. Girardi, Romullo & Oliveira, Jauvane. (2021). IME VR: An MVC Framework for Military Training VR Simulators. 10.1007/978-3-030-77599-5_40.
4. Geospecific Simulation With Game Quality Graphics [Електронний ресурс] – Режим доступу до ресурсу: URL: https://www.mvrsimulation.com/ (дата звернення: 02.10.2022);
5. Viveport SDK [Електронний ресурс] – Режим доступу до ресурсу: URL: https://developer.vive.com/resources/viveport/sdk/documentation/english/viveport-sdk/ (дата звернення: 02.10.2022);
6. Autodesk Maya documetation [Електронний ресурс] – Режим доступу до ресурсу: URL: https://help.autodesk.com/view/MAYAUL/2023/ENU/ (дата звернення: 02.10.2022);
7. Unity3D SDK [Електронний ресурс] – Режим доступу до ресурсу: URL: https://docs.unity3d.com/Manual/index.html (дата звернення: 02.10.2022);
8. Gupta, A., Cecil, J., Pirela-Cruz, M., Ramanathan, P.: A virtual reality enhanced cyber-human framework for orthopedic surgical training. IEEE Syst. J. 13(3), 3501–3512 (2019)
9. Hill, R.R., Miller, J.O.: A history of united states military simulation. In: Proceedings of the 2017 Winter Simulation Conference. WSC 2017. IEEE Press (2017)

10. Syntax and Semantics for IDEF0 [Електронний ресурс] – Режим доступу до ресурсу: URL: https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:31320:-1:ed-1:v1:en (дата звернення: 02.10.2022);
11. Introduction to UML [Електронний ресурс] – Режим доступу до ресурсу: URL: http://www.uml.org/what-is-uml.htm (дата звернення: 02.10.2022);
12. Standard Work Breakdown Structure
13. [Електронний ресурс] – Режим доступу до ресурсу: URL: https://help.sap.com/docs/SAP_S4HANA_ON-PREMISE/04cb6d82d96a43e1947dfa9921687b8e/b972bb53707db44ce10000000a174cb4.html (дата звернення: 02.10.2022);
14. [Електронний ресурс] - Режим доступу до ресурсу: URL: https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647 (дата звернення: 02.10.2022);
15. [Електронний ресурс] - Режим доступу до ресурсу: URL: https://docs.unity3d.com/Manual/class-TrailRenderer.html (дата звернення: 02.10.2022);

ДОДАТОК А

ПЛАНУВАННЯ РОБІТ

для розробки кваліфікаційної роботи магістра

“ ***Симулятор роботи гаубиці на основі віртуальної реальності***”

А.1 ІДЕНТИФІКАЦІЯ МЕТИ ІТ-ПРОЕКТУ

В наш час, питання швидкої і якісної підготовки артилеристів, це питання виживання не тільки людей, а і України, оскільки від вмінь, і навичок цих незламних воїнів, залежить те як далеко зможе підійти ворог, як просуватиметься наша піхота, так як швидко ми звільнимо наші території, адже не дарма артилеристів ще називають “Боги війни”. Як видно з історії, наявність саме артилерії впливає на перебіг війни, адже ніяка піхота не може пробитись через вогняний артилерійський вал, не втративши при цьому тисячі життів.

Отже підготовка артилеристів в максимально короткі терміни, займає одну з найважливіших ніш в підготовці військових спеціалістів.

А враховуючи, те що проектована система не залежить від місця, оскільки потребує лише наявності комп’ютера та окулярів віртуальної реальності, то, підготовка може здійснюватися в будь-якій точці земної кули, незалежно від локації.

А.2 ПЛАНУВАННЯ ЗМІСТУ СТРУКТУРИ РОБІТ ІТ ПРОЕКТУ

Для планування структури проекту використаєм підхід на основі WBS (Work Breakdown Structure). Згідно з ним вся робота розбиваєтеся на ієрархічну структуру, що допомагає досягти цілей проєкту. Шляхом ітерацій, WBS розбивається на атомарні, логічно повні частини. Результат цього розбиття — мінімальні частини які можуть бути легко оцінені та сплановані.

WBS проекту — це схема, де верхні рівні слугують для розроблення пруф-офф концепт проекту. Для подальшої деталізації потрібні технічні специфікації.

Процес створення WBS має декілька основних принципів:

− правило 100%, кожен підрівень повинен вміщувати 100% роботи батьківського рівня. Порушення цього правила, призводить до порушення термінів реалізації. Зміщення термінів, та втрати проектом актуальності;

− несумісні події, тобто не повинно бути подій, які суперечать одна одній. Порушення цього правила призводить до подвоєння термінів виконання роботи, та до непередбачуваних результатів під час фази розробки проекту;

− заплановані висновки замість дій. Тобто той хто створює WBS повинен думати про кінцевий результат, а не про способи його досягнення. Порушення цього правила може призвести до перевищення 100% порогу з першого правила.

− рівень деталізації. При розробці діаграми, завжди повинна існувати кінцева точка завершення поділу роботи на менші частини. Це правило містить в собі велику кількість підпунктів, основні це — правило 80 годин, згідно з яким, жодна діяльність не може перевищувати 80 робочих годин. Друге правило визначає що жодна діяльність на найнижчому рівні не може бути довшою за звітній період. Останнє підправило — при створенні діяльності потрібно враховувати чи є це доцільним.

Одним з елементів WBS є термінальний елемент, це найнижчий і мінімально неподільний елемент [12]. Розроблена діаграма зображена на рисунку А. 2.1.

Після побудови WBS розробляють організаційну структуру виконавців. Організаційна структура проекту (OBS) є ієрархічною структурою управління проектом. Вона визначається за переліком пакетів робіт нижнього рівня кожної гілки WBS-структури. Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які задіяні в реалізації проекту. Оскільки над цим проектом працює єдина людина, автор то відображати схематично цю структуру немає сенсу. Для її побудови потрібно в кожному квадраті з рисунку А.2.1 замінити текст на відповідальну особу. Діаграма OBS представлена на рис. А.2.2.

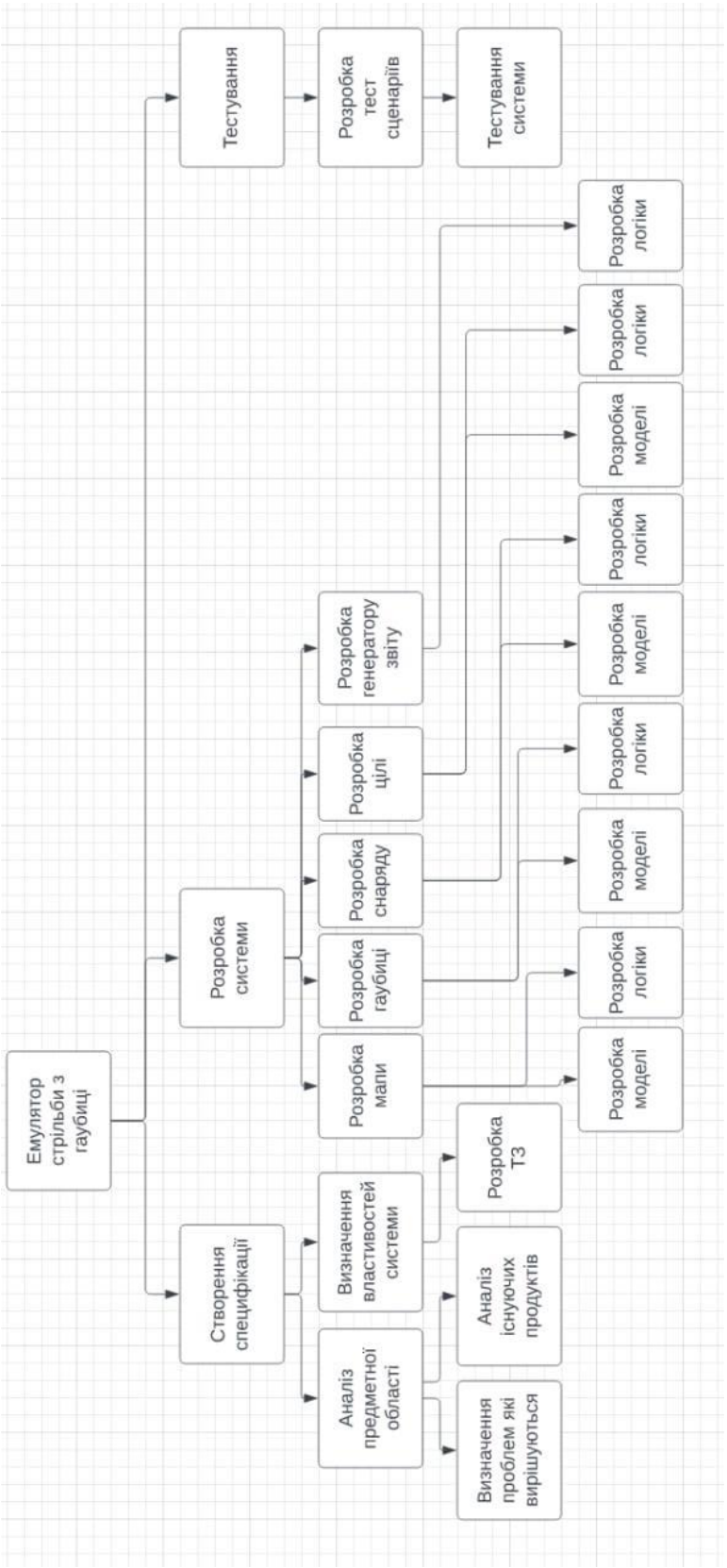

Рисунок А. 2.1 – WBS діаграма

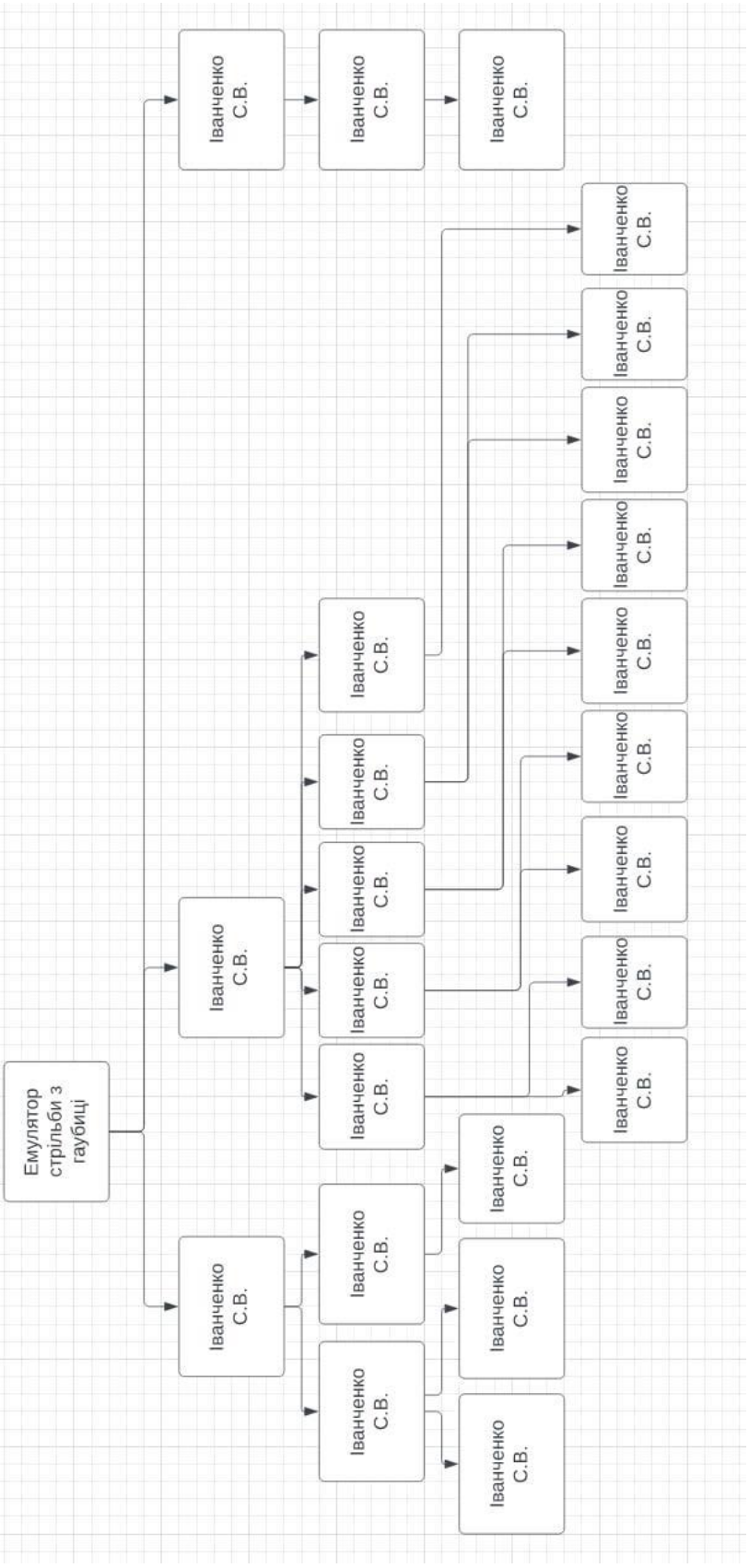

Рисунок А.2.2 – OBS Діаграма

А.3 ПОБУДОВА КАЛЕНДАРНОГО ГРАФІКУ ВИКОНАННЯ ІТ-ПРОЕКТУ

Для визначення часу робіт та заходів, які потрібно виконати для завершення проекту, і встановлення взаємозв'язків між роботами та цілями з урахуванням ризиків, складається календарний план проекту. Календарне планування полягає у створенні та подальшому уточненні розкладу, який враховує склад робіт, ризики, обмеження. Графік зображено на рисунку А.3.1.

Назва задачі	Тривалість	Початок	Кінець
1. Створення емуляторм стрільби з гаубиці	30	01.10.22	30.10.22
1.1 Створення специфікації	5	1.10.22	01.05.22
1.1.1 Аналіз предметної області	2	1.10.22	2.10.22
1.1.1.1 Визначення проблем які вирішуються	1	1.10.22	1.10.22
1.1.1.2 Аналіз Існуючих продуктів	1	2.10.22	2.10.22
1.1.2. Визначення властивостей системи	3	2.10.22	2.10.22
1.1.2.1 Розробка ТЗ	3	2.10.22	5.10.22
1.2 Розробка системи	20	5.10.22	25.10.22
1.2.1 Розробка мапи	5	5.10.22	10.10.22
1.2.1.1 Розробка моделі	3	5.10.22	8.10.22
1.2.1.2 Розробка логіки	2	8.10.22	10.10.22
1.2.2 Розробка гаубиці	5	10.10.22	15.10.22
1.2.2.1 Розробка моделі	3	10.10.22	13.10.22
1.2.2.2 Розробка логіки	2	13.10.22	15.10.22
1.2.3 Розробка снаряду	5	15.10.22	20.10.22
1.2.3.1 Розробка моделі	3	15.10.22	18.10.22
1.2.3.2 Розробка логіки	2	18.10.22	20.10.22
1.2.4 Розробка цілі	5	20.10.22	25.10.22
1.2.4.1 Розробка моделі	3	20.10.22	23.10.22
1.2.4.2 Розробка логіки	2	23.10.22	25.10.22
1.2.5 Розробка генератору звіту	1	25.10.22	25.10.22
1.2.5.1 Розробка логіки	1	25.10.22	25.10.22
1.3 Тестування	5	25.10.22	30.10.22
1.3.1 Розробка тестценаріїв	2	25.10.22	27.10.22
1.3.1.1 Тестуванная	3	27.10.22	30.10.22

Рисунок А.3.1 – Календарний план робіт

А.4 ПЛАНУВАННЯ РИЗИКІВ ПРОЕКТУ

Ризиком це подія, вірогідність якої дуже висока. В результаті цієї події суб’єкт, який прийняв рішення, втрачає можливість досягти запланованих результатів проекту або його окремих параметрів, що мають тимчасову, кількісну та вартісну оцінку. У кожного ризика є свої джерелами чи причинами та обов’язково є наслідки. Ризик безпосередньо впливає на досягнення результатів проекту.

Процес управління ризиками включає наступні етапи:

- ідентифікація;
- процес оцінювання ризиків,
- моніторинг заходів і ризиків.

Ризики проекту представлені у таблиці А.4.1.

Таблиця А.4.1 Ризики проекту

Ризик	Назва
1	Недостатні знання в певній області
2	Проблеми з відсутністю фізичної можливості працювати над проектом (хвороба. віялові вимкнення електроенергії)
3	Фінансування

Реагування на ризики представлені у таблиці А.4.2.

Таблиця А.4.2 Реагування на ризики

Ризик	Назва	реагування
1	Недостатні знання в певній області	Пошук курсів.Джерел. допоміжної інформації.
2	Проблеми з відсутністю фізичної можливості працювати над проектом (хвороба. віялові вимкнення електроенергії)	корегування часу на виконання проекту
3	Фінансування	Припинення розробки до відновлення повного фінансування

ДОДАТОК Б

Б1. СКРИПТ ПОВЕДІНКИ ВІРТУАЛЬНИХ МАНІПУЛЯТОРІВ

```
public class VrSimulatorHandFixer156 : SteamVR_Behaviour_Pose
{
   Valve.VR.InteractionSystem.Hand _hand;
   protected override void Start()
   {

      base.Start();
      _hand = this.gameObject.GetComponent<Valve.VR.InteractionSystem.Hand>();
      _hand.handType = SteamVR_Input_Sources.RightHand;

         GameObject broHand = GameObject.Instantiate(_hand.gameObject);
         Destroy(broHand.GetComponent<VrSimulatorHandFixer156>());
         broHand.SetActive(false);
      _hand.otherHand =
broHand.GetComponent<Valve.VR.InteractionSystem.Hand>();
      _hand.otherHand.handType = SteamVR_Input_Sources.LeftHand;

      var spoofMouse = new SpoofMouseAction();
      _hand.grabGripAction = spoofMouse;
      spoofMouse.InitializeDictionariesExposed(_hand.handType);

      this.poseAction = new Poser_SteamVR_Action_Pose();
   }

   class SpoofMouseAction : SteamVR_Action_Boolean
   {


      public override void UpdateValue(SteamVR_Input_Sources inputSource)
      {
         lastActionData[inputSource] = actionData[inputSource];
         tempActionData.bState = Input.GetMouseButton(0) ||
Input.GetMouseButtonDown(0);
         tempActionData.bChanged = Input.GetMouseButtonDown(0) ||
Input.GetMouseButtonUp(0);
```

```
            tempActionData.bActive = true;

            actionData[inputSource] = tempActionData;
            changed[inputSource] = tempActionData.bChanged;
            active[inputSource] = tempActionData.bActive;
            activeOrigin[inputSource] = tempActionData.activeOrigin;
            updateTime[inputSource] = Time.time;

            if (changed[inputSource])
               lastChanged[inputSource] = Time.time;

            if (onStateDown[inputSource] != null && GetStateDown(inputSource))
               onStateDown[inputSource].Invoke(this);

            if (onStateUp[inputSource] != null && GetStateUp(inputSource))
               onStateUp[inputSource].Invoke(this);

            if (onChange[inputSource] != null && GetChanged(inputSource))
               onChange[inputSource].Invoke(this);

            if (onUpdate[inputSource] != null)
               onUpdate[inputSource].Invoke(this);

            if (onActiveChange[inputSource] != null &&
lastActionData[inputSource].bActive != active[inputSource])
               onActiveChange[inputSource].Invoke(this, active[inputSource]);
         }
      }
         class Poser_SteamVR_Action_Pose : SteamVR_Action_Pose
      {
         public override bool GetActive(SteamVR_Input_Sources inputSource)
         {
            return false;
         }
      }
}
```

Б2. СКРИПТ ГЕНЕРАЦІЇ СНАРЯДУ

```
void Update()
{
	if(Input.GetMouseButton(0))
	{
		curTimeout += Time.deltaTime;
		if(curTimeout > timeout)
		{
			curTimeout = 0;
			Rigidbody bulletInstance = Instantiate(bullet, gunPoint.position,
			Quaternion.identity) as Rigidbody;
			bulletInstance.velocity = gunPoint.forward * bulletSpeed;
		}
	}
	else
	{
		curTimeout = timeout + 1;
	}
}


using UnityEngine;
using System.Collections;

public class Bullet : MonoBehaviour {
	public float damage = 1;
	public string[] targetTags = {"Target_1"};

	void OnTriggerEnter(Collider coll)
	{
		foreach(string currentTag in targetTags)
		{
			if(currentTag == coll.transform.tag)
			{
			coll.transform.GetComponent<EnemyHealth>().AddDamage(damag
			e);
			}
		}
		Destroy(gameObject);
	}
```

```
}
```