ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Створення симулятору для тренування медичних процедур мовою C# та з використанням Unity»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Микола АРТЕМЕНКО
(підпис)

Виконав: здобувач вищої освіти групи ПД-43

Микола АРТЕМЕНКО

Керівник: Ігор АВЕРІЧЕВ
к.е.н.

Рецензент: _____

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Артеменку Миколі Анатолійовичу

1. Тема кваліфікаційної роботи: «Створення симулятору для тренування медичних процедур мовою C# та з використанням Unity»
керівник кваліфікаційної роботи к.е.н., доцент кафедри ІПЗ Ігор АВЕРІЧЕВ,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024 р. № 36.

2. Строк подання кваліфікаційної роботи «28» травня 2024 р.

3. Вихідні дані до кваліфікаційної роботи: опис процесу створення симулятора для тренування медичних процедур, аналіз існуючих систем симуляції та управління медичними тренуваннями, план розробки симулятора з включенням етапів розробки та тестування з використанням мови програмування C# та Unity.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Огляд та аналіз існуючих проблем у системах симуляції медичних процедур та визначення вимог до нового симулятора.
2. Проектування симулятора для тренування медичних процедур з використанням мови програмування C# та Unity.

3. Програмна реалізація та опис функціонування симулятору.
4. Тестування симулятору.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Програмні засоби реалізації.
4. Діаграми варіантів використання.
5. Діаграма класів
6. Діаграма послідовності.
7. Екранні форми
8. Відео з демонстрацією медичного симулятору.
9. Апробація результатів дослідження.

6. Дата видачі завдання «28» лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	28.02.-06.03.2024	
2	Аналіз та дослідження існуючих аналогів	07.03-13.03.2024	
3	Вибір інструментів та технологій для розробки симулятора	14.03-20.03.2024	
4	Проектування застосунку для тренування медичних процедур	21.03-29.03.2024	
5	Програмна реалізація симулятора	30.03-08.04.2024	
6	Тестування симулятора	09.04-28.04.2024	
7	Оформлення роботи: вступ, висновки, реферат	29.04-05.05.2024	
8	Розробка демонстраційних матеріалів	06.05-12.05.2024	
9	Попередній захист роботи	13.05-31.05.2024	

Здобувач вищої освіти _____ Микола АРТЕМЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Ігор АВЕРІЧЕВ
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 42 стор., 1 табл., 10 рис., 39 джерел.

Мета роботи – поліпшення навчання студентів-медиків напрямку «загальна медицина» з використанням інтерактивного симулятора для тренування медичних процедур.

Об'єкт дослідження - процес підготовки студентів-медиків напрямку «загальна медицина» до виконання медичних процедур.

Предмет дослідження - інтерактивний симулятор медичних процедур, розроблений за допомогою C# та Unity.

Короткий зміст роботи: у роботі проаналізовано існуючі системи симуляції медичних процедур та визначено їхні недоліки. Встановлено вимоги до нового симулятора, зокрема реалістичність, інтерактивність та можливість багатокористувацької взаємодії. Розроблено симулятор, який включає модулі візуалізації процедур, інтерактивних сценаріїв, оцінки результатів та зворотного зв'язку. Програмно реалізовані ключові функціональні можливості, такі як симуляція різних медичних процедур, інтерактивне керування інструментами та моніторинг прогресу користувача. Проведено тестування симулятора на відповідність вимогам безпеки, продуктивності та зручності використання. Для розробки симулятора використано мову програмування C# та платформу Unity, що забезпечило створення високоякісних візуальних ефектів та інтерактивності.

Сферою використання симулятора є навчальні медичні заклади та центри підготовки медичного персоналу, де необхідне тренування медичних процедур у реалістичних умовах.

КЛЮЧОВІ СЛОВА: СИМУЛЯТОР, МЕДИЧНІ ПРОЦЕДУРИ, C#, UNITY, ІНТЕРАКТИВНЕ ТРЕНУВАННЯ

ЗМІСТ

ВСТУП

У сучасному світі інформаційних технологій симулятори медичних процедур набувають все більшого значення для підготовки медичного персоналу. Розробка ефективного симулятора для тренування медичних процедур є актуальним завданням, яке дозволяє покращити якість медичної освіти, підвищити рівень підготовки медичних працівників та забезпечити безпечне середовище для відпрацювання навичок.

Для реалізації симулятора медичних процедур необхідно обрати відповідні технології та інструменти розробки. Серед популярних інструментів для розробки ігор та інтерактивних додатків виділяються мова програмування C# та платформа Unity. Це поєднання дозволяє створювати реалістичні та інтерактивні симуляції, які відповідають високим вимогам медичної освіти.

Постановка завдання

Завданням дослідження є розробка симулятора для тренування медичних процедур з використанням мови програмування C# та платформи Unity. Система повинна забезпечувати можливість інтерактивного навчання, відпрацювання різних медичних процедур та надання зворотного зв'язку користувачам.

Мета дослідження

Метою дослідження є створення ефективного та зручного у використанні симулятора для тренування медичних процедур, який дозволить підвищити якість підготовки медичних працівників, спростити процес навчання та забезпечити безпечне середовище для відпрацювання навичок.

Результати дослідження

Симулятор медичних процедур - це комплексне програмне рішення, яке дозволяє автоматизувати процеси навчання та тренування медичного персоналу. Така система забезпечує інтерактивне середовище для відпрацювання навичок, зворотний зв'язок та можливість оцінки результатів навчання.

Для розробки симулятора було обрано мову програмування C# та платформу Unity. C# є популярною мовою програмування для розробки ігор та

інтерактивних додатків, яка дозволяє створювати складні логічні структури та взаємодії. Unity - це потужна платформа для розробки 2D та 3D додатків, яка забезпечує високоякісну графіку та зручний інтерфейс для розробки.

Архітектура симулятора передбачає модульну структуру, де кожен модуль відповідає за певну частину функціональності. Основні модулі включають візуалізацію медичних процедур, інтерактивні сценарії, оцінку результатів та зворотний зв'язок. Така архітектура забезпечує високу гнучкість, масштабованість та можливість розширення функціональності.

Симулятор включає кілька основних модулів: модуль візуалізації процедур, модуль інтерактивних сценаріїв, модуль оцінки результатів та модуль зворотного зв'язку. Модуль візуалізації процедур дозволяє користувачам відпрацьовувати різні медичні маніпуляції в реалістичному середовищі. Модуль інтерактивних сценаріїв забезпечує можливість проходження різних навчальних сценаріїв, а модуль оцінки результатів надає користувачам зворотний зв'язок щодо їхньої успішності.

Після впровадження симулятора необхідно забезпечити його супровід та підтримку. Це включає регулярне оновлення програмного забезпечення, усунення виявлених помилок та надання консультацій користувачам. Ефективна підтримка системи дозволяє забезпечити її стабільну роботу та адаптацію до мінливих потреб навчальних закладів.

Висновки та перспективи

Розробка симулятора для тренування медичних процедур з використанням мови програмування C# та платформи Unity є ефективним рішенням для покращення медичної освіти. Такий симулятор дозволяє автоматизувати процеси навчання, забезпечити безпечне середовище для відпрацювання навичок та підвищити якість підготовки медичних працівників.

Використання сучасних технологій, таких як C# та Unity, дозволяє створити реалістичний та інтуїтивно зрозумілий симулятор, який спрощує процес навчання та підвищує його ефективність. Розроблений симулятор

відповідає вимогам безпеки, масштабованості та продуктивності, що дозволяє його впроваджувати у різних навчальних закладах та медичних центрах.

Впровадження симулятора вимагає ретельного планування, тестування та налаштування. Необхідно забезпечити якісну підтримку та супровід системи, регулярно оновлювати програмне забезпечення та адаптувати його до нових вимог. Лише за таких умов симулятор зможе ефективно виконувати свої функції та сприяти підвищенню якості медичної освіти.

1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ І ВИМОГ ДО НОВОГО СИМУЛЯТОРА

1.1. Аналіз сучасних симуляторів медичних процедур

Таблиця 1.1

Аналіз аналогів

Симулятор	TraumaMan	Vimedix	SimMan 3G	Nursing Anne	CureAndLearn
Підтримка навчальних матеріалів	Включає текстові матеріали та відеоуроки	Інтерактивні інструкції, відеоуроки, аналітика	Навчальні матеріали, відеоуроки	Текстові матеріали, інструкції	Інтерактивні інструкції, тестування
Зворотній зв'язок	Реалістична реакція	Аналіз дій користувача	Покрокові інструкції	Тактильний зворотній зв’язок	Інтерактивний зворотній зв'язок
Використання інтерактивних технологій	Обмежені інтерактивні можливості	Використання VR технологій	Інтерактивні сценарії та симуляції	Мінімальне використання інтерактивних технологій	Використання тривимірних моделей
Реалістичність	Висока	Висока	Висока	Середня	Висока
Платформа	Фізичний манекен	ПК, VR	Фізичний манекен	Фізичний манекен	ПК
Набір сценаріїв	Травми, екстрена допомога	Ультразвукова діагностика	Реанімація, кардіологія	Основні медсестринські процедури	Загальні медичні процедури

Сучасні системи симуляції медичних процедур, які використовуються у навчальних медичних закладах, мають низку недоліків, що можуть негативно впливати на ефективність навчального процесу, підготовку студентів та загальний рівень медичної освіти. Ці недоліки можна розділити на кілька основних категорій. Розглянемо їх:

– Нереалістичність та обмежені можливості симуляції

– Висока вартість обладнання та його обслуговування

– Обмежена доступність та мобільність

– Відсутність можливостей для персоналізації та адаптації

– Труднощі у впровадженні та інтеграції з іншими навчальними системами

– Недостатня підтримка зворотного зв’язку та оцінки

Розглянемо ці недоліки більш детально:

Нереалістичність та обмежені можливості симуляції

Багато існуючих симуляторів не можуть забезпечити достатньо реалістичний досвід для студентів. Наприклад, деякі системи не відображають точну анатомію людини або не можуть симулювати складні медичні процедури з високою точністю. Це обмежує можливості студентів для набуття необхідних навичок у реальних умовах. Крім того, багато симуляторів не враховують індивідуальні особливості пацієнтів, такі як анатомічні відмінності, різні стадії хвороб або реакції організму на лікування. Це знижує ефективність навчання та підготовки медичного персоналу.

Висока вартість обладнання та його обслуговування

Сучасні симулятори часто вимагають значних фінансових вкладень для придбання та обслуговування. Це включає вартість самого обладнання, його встановлення, регулярне технічне обслуговування та оновлення програмного забезпечення. Такі витрати можуть бути непідйомними для багатьох навчальних закладів, особливо у країнах з обмеженим бюджетом на освіту. Крім того, необхідність постійного оновлення обладнання та програмного забезпечення може створювати додаткові фінансові труднощі та перешкоди для впровадження нових технологій.

Обмежена доступність та мобільність

Багато симуляторів є стаціонарними та вимагають спеціально обладнаних приміщень. Це обмежує можливості для студентів використовувати симулятори поза межами навчального закладу або в різних локаціях. У випадках, коли навчальні програми потребують мобільності та гнучкості, така обмеженість може стати значною перешкодою. Студенти, які навчаються дистанційно або працюють у віддалених місцях, можуть мати обмежений доступ до необхідних ресурсів та обладнання для симуляційного навчання.

Відсутність можливостей для персоналізації та адаптації

Існуючі симулятори часто не дозволяють налаштовувати сценарії під конкретні потреби студентів або навчальних програм. Відсутність можливостей для адаптації навчального процесу під індивідуальні потреби студентів може

призводити до менш ефективного навчання та не дозволяє максимально використовувати потенціал симуляційного навчання. Наприклад, деякі симулятори можуть не враховувати різні рівні підготовки студентів або специфічні вимоги до навчальних програм, що обмежує можливості для гнучкого та ефективного навчання.

Труднощі у впровадженні та інтеграції з іншими навчальними системами

Інтеграція симуляторів з існуючими навчальними платформами та системами управління навчальним процесом може бути складною та витратною. Це включає в себе проблеми з сумісністю програмного забезпечення, необхідність додаткових ресурсів для налаштування та технічної підтримки, а також складність в управлінні навчальними даними. Наприклад, у випадках, коли навчальні заклади використовують різні платформи для управління навчальним процесом, інтеграція симуляторів може вимагати значних зусиль та ресурсів для забезпечення їхньої сумісності та ефективної роботи.

Недостатня підтримка зворотного зв’язку та оцінки

Багато симуляторів не забезпечують належний рівень зворотного зв’язку та оцінки результатів навчання. Це може ускладнювати процес оцінювання прогресу студентів, аналізу їхніх помилок та надання рекомендацій для подальшого вдосконалення. Відсутність детального зворотного зв’язку обмежує можливості для самостійного навчання та підвищення кваліфікації студентів. Наприклад, деякі симулятори можуть не надавати докладних даних про помилки студентів або не забезпечувати можливість для аналізу їхніх дій у процесі симуляції, що ускладнює процес навчання та підготовки.

Перспективи вдосконалення

Розробка симулятора для тренування медичних процедур з використанням мови програмування C# та платформи Unity може значно подолати зазначені недоліки. Завдяки Unity можна створити реалістичні 3D-моделі анатомії та медичних процедур, що дозволить студентам зануритися у реалістичне навчальне середовище. Використання C# забезпечує високу гнучкість та можливості для налаштування симулятора під конкретні навчальні потреби.

Медичний симулятор також може бути доступним на різних пристроях, включаючи персональні комп’ютери та мобільні пристрої, що підвищить його доступність та мобільність. Інтеграція з іншими навчальними платформами та системами управління дозволить створити єдиний навчальний простір, що спростить процес управління навчальним процесом та оцінки результатів.

Забезпечення можливостей для персоналізації навчального процесу, інтерактивного зворотного зв’язку та детального оцінювання сприятиме підвищенню ефективності навчання та підготовки медичних працівників. Впровадження сучасних технологій у симуляційне навчання відкриває нові горизонти для розвитку медичної освіти та підвищення рівня підготовки майбутніх фахівців.

Зокрема, використання віртуальної реальності (VR) та доповненої реальності (AR) може забезпечити додатковий рівень інтерактивності та занурення, що сприятиме більш ефективному засвоєнню знань та навичок. VR та AR технології можуть використовуватись для створення різноманітних сценаріїв медичних процедур, які будуть максимально наближені до реальних умов.

Інтеграція системи з платформами для управління навчальним процесом також дозволить автоматизувати процеси збору та аналізу даних про успішність студентів, що сприятиме покращенню якості навчання та підвищенню рівня підготовки майбутніх медичних працівників. Відстеження прогресу студентів у режимі реального часу та надання персоналізованих рекомендацій для подальшого навчання дозволить більш ефективно використовувати ресурси навчальних закладів та підвищити якість підготовки фахівців.

Крім того, використання платформ для спільної роботи та обміну досвідом між студентами та викладачами може сприяти підвищенню ефективності навчального процесу та покращенню комунікації між усіма учасниками навчального процесу. Це дозволить студентам отримувати зворотний зв’язок та рекомендації від викладачів та колег у режимі реального часу, що сприятиме покращенню якості навчання та підвищенню рівня підготовки медичних працівників.

Таким чином, впровадження сучасних технологій та підходів до розробки симуляторів для тренування медичних процедур з використанням C# та Unity відкриває нові можливості для покращення якості медичної освіти та підготовки майбутніх фахівців.

1.2. Дослідження закордонного досвіду в області медичних симуляторів

Розвиток медичних симуляторів у закордонних країнах демонструє значні досягнення та інновації, які можуть служити прикладом для вдосконалення медичної освіти в інших країнах. Розглянемо основні напрями та успішні приклади використання симуляторів у медичній практиці за кордоном.

США

Сполучені Штати Америки є одним із лідерів у впровадженні медичних симуляторів у навчальні програми медичних шкіл та університетів. Одним із найбільш відомих центрів є Центр симуляції у медичній освіті при Гарвардському університеті, де використовуються передові технології для навчання студентів. Цей центр має широкі можливості для проведення симуляцій різних медичних процедур, від базових навичок до складних хірургічних втручань.

У США також активно використовуються віртуальні симулятори, такі як симуляційна платформа "SimMan", розроблена компанією Laerdal. Ця платформа дозволяє студентам відпрацьовувати навички на інтерактивних манекенах, які можуть відтворювати різні фізіологічні стани пацієнтів. Такий підхід забезпечує реалістичні умови навчання та допомагає студентам краще підготуватися до роботи в реальних клінічних умовах.

Канада

У Канаді симуляційні центри також широко використовуються для підготовки медичних працівників. Один із провідних центрів - Центр симуляції у медичній освіті при Університеті Торонто, який оснащений найсучаснішими симуляторами для навчання студентів медичних спеціальностей. В цьому центрі

використовуються як фізичні манекени, так і віртуальні симуляції для відпрацювання навичок у різних медичних дисциплінах.

Канадські медичні школи активно впроваджують міждисциплінарні симуляційні програми, де студенти з різних спеціальностей працюють разом у команді. Це дозволяє майбутнім лікарям, медсестрам і іншим медичним працівникам навчитися ефективно співпрацювати і комунікувати під час надання медичної допомоги.

Європа

У Європі також є багато успішних прикладів використання медичних симуляторів у навчальних програмах. У Великобританії одним із провідних центрів є Центр симуляції у медичній освіті при Королівському коледжі Лондона. Цей центр пропонує студентам широкий спектр симуляційних програм, включаючи як базові, так і складні медичні процедури.

У Німеччині симуляційні центри також активно впроваджуються у навчальні програми медичних університетів. Наприклад, Університет Гайдельберга має центр симуляції, де використовуються сучасні технології для навчання студентів. В цьому центрі особлива увага приділяється симуляції хірургічних втручань, що дозволяє студентам отримати необхідні навички у безпечних умовах.

Австралія

Австралія також є однією з країн, де активно використовуються медичні симулятори у навчальних програмах. Наприклад, Університет Мельбурна має центр симуляції, де студенти можуть відпрацьовувати різні медичні навички. У цьому центрі використовуються як фізичні манекени, так і віртуальні симуляції для забезпечення реалістичних умов навчання.

В Австралії також впроваджуються телемедичні симуляції, які дозволяють студентам з різних регіонів брати участь у навчальних програмах. Це особливо корисно для студентів, які живуть у віддалених районах і не мають можливості регулярно відвідувати навчальні заклади.

Азія

В Азії медичні симулятори також знаходять широке застосування у навчальних програмах. У Японії, наприклад, медичні університети активно використовують симулятори для навчання студентів. Університет Токіо має сучасний симуляційний центр, де використовуються як фізичні манекени, так і віртуальні симуляції. В цьому центрі особлива увага приділяється симуляції невідкладної медичної допомоги та хірургічних втручань.

У Сінгапурі медичні школи також активно впроваджують симуляційні програми у навчальні курси. Університет Сінгапуру має один із найсучасніших симуляційних центрів у регіоні, де студенти можуть відпрацьовувати різні медичні процедури у реалістичних умовах.

Інновації у використанні симуляторів

Зарубіжний досвід також демонструє значні інновації у використанні симуляторів для медичної освіти. Наприклад, у багатьох країнах активно використовуються технології віртуальної (VR) та доповненої реальності (AR) для створення реалістичних навчальних сценаріїв. Використання VR та AR дозволяє студентам занурюватися у віртуальні операційні зали або кабінети, де вони можуть відпрацьовувати різні медичні процедури без ризику для пацієнтів.

Крім того, впровадження штучного інтелекту (AI) у симуляційні програми дозволяє створювати адаптивні навчальні середовища, які можуть змінюватися залежно від дій студентів. Це сприяє більш гнучкому та персоналізованому навчанню, яке враховує індивідуальні потреби та рівень підготовки студентів.

У закордонних центрах також активно використовуються міждисциплінарні симуляційні програми, де студенти з різних медичних спеціальностей працюють разом у команді. Це дозволяє їм навчитися ефективно співпрацювати та комунікувати під час надання медичної допомоги, що є критично важливим у реальних клінічних умовах.

Дослідження закордонного досвіду у сфері медичних симуляторів показує, що впровадження передових технологій та інноваційних підходів до навчання може значно підвищити якість медичної освіти. Використання віртуальних симуляторів, VR та AR технологій, штучного інтелекту та міждисциплінарних

програм сприяє більш ефективному навчанню студентів та підвищенню їхньої готовності до роботи у реальних клінічних умовах.

Впровадження таких підходів у навчальні програми медичних закладів дозволить підвищити рівень підготовки майбутніх медичних працівників, забезпечити їм необхідні навички та знання для надання якісної медичної допомоги. Це, у свою чергу, сприятиме покращенню загального рівня медичного обслуговування та підвищенню якості життя пацієнтів.

1.3. Проектування архітектури симулятора

Проектування архітектури симулятора медичних процедур є важливим етапом у створенні ефективної та гнучкої системи для навчання медичних працівників. Враховуючи сучасні вимоги до медичної освіти та можливості технологій, архітектура симулятора повинна забезпечувати реалістичність, інтерактивність, масштабованість та безпеку. Основні компоненти архітектури включають модуль візуалізації, модуль сценаріїв, модуль зворотного зв'язку, базу даних та інтерфейс користувача.

Модуль візуалізації

Модуль візуалізації відповідає за відображення реалістичних 3D-моделей медичних процедур та анатомії людини. Використовуючи можливості Unity, можна створити детальні та інтерактивні моделі, що дозволяють студентам занурюватися у віртуальне середовище. Основні функції модуля візуалізації включають:

– Відтворення анатомічних структур та медичних інструментів у реальному часі.

– Візуалізація фізіологічних процесів, таких як кровообіг, дихання та реакції організму на медичні втручання.

– Інтерактивні елементи для маніпулювання інструментами та виконання процедур.

Модуль сценаріїв

Модуль сценаріїв дозволяє створювати та керувати навчальними сценаріями для відпрацювання різних медичних процедур. Цей модуль забезпечує гнучкість у налаштуванні різних навчальних ситуацій, включаючи екстрені випадки та рідкісні патології. Основні функції модуля сценаріїв включають:

– Створення сценаріїв з різним рівнем складності та умовами.

– Адаптація сценаріїв під індивідуальні потреби студентів.

Модуль зворотного зв'язку

Модуль зворотного зв'язку забезпечує студентів інформацією про їхні дії та результати виконання процедур. Це допомагає виявляти помилки та надає рекомендації для покращення навичок. Основні функції модуля зворотного зв'язку включають:

– Відстеження дій студентів у режимі реального часу.

– Надання докладного аналізу виконаних процедур та виявлених помилок.

– Автоматичне створення звітів про успішність навчання та рекомендацій для подальшого розвитку.

База даних

База даних є центральним елементом архітектури симулятора, яка зберігає інформацію про навчальні сценарії, результати студентів та інші важливі дані. Основні функції бази даних включають:

– Зберігання даних про анатомічні моделі, сценарії та результати навчання.

– Забезпечення швидкого доступу до необхідної інформації для всіх модулів симулятора.

– Забезпечення безпеки та конфіденційності даних студентів.

.Інтеграція компонентів

Інтеграція всіх компонентів архітектури є ключовим етапом у створенні ефективного симулятора. Використання Unity дозволяє об'єднати модулі візуалізації, сценаріїв та зворотного зв'язку в єдину систему. Це забезпечує

безперебійну роботу симулятора та надає студентам можливість навчатися у реалістичному та інтерактивному середовищі.

Для забезпечення високої продуктивності та масштабованості симулятора, архітектура повинна підтримувати багатокористувацький режим, що дозволить студентам працювати разом у команді. Крім того, необхідно забезпечити можливість інтеграції симулятора з іншими навчальними системами та платформами для управління навчальним процесом.

Проектування архітектури симулятора медичних процедур з використанням C# та Unity є складним, але важливим процесом для створення ефективної та гнучкої навчальної системи. Реалістичність, інтерактивність, масштабованість та безпека є ключовими аспектами, які повинні бути враховані при розробці архітектури симулятора. Впровадження таких систем у навчальні програми медичних закладів сприятиме покращенню якості медичної освіти та підготовки майбутніх медичних працівників.

2 ВИБІР ТЕХНОЛОГІЙ І ПРОЕКТУВАННЯ СИСТЕМИ

2.1. Огляд технологій для розробки симуляторів

Розробка симулятора для тренування медичних процедур потребує вибору відповідних технологій, які забезпечать необхідний рівень реалістичності, інтерактивності та гнучкості. Основними аспектами, які слід враховувати при виборі технологій, є мови програмування, платформи для розробки, інструменти для візуалізації та інтеграції, а також засоби для забезпечення безпеки та продуктивності системи.

Мови програмування

Для розробки симулятора медичних процедур було обрано мову програмування C#. Ця мова є потужним інструментом для розробки ігор та інтерактивних додатків, що робить її ідеальною для створення реалістичних та інтерактивних симуляцій. Основні переваги C# включають:

– Об'єктно-орієнтований підхід: C# є об'єктно-орієнтованою мовою програмування, що дозволяє створювати модульний, зрозумілий і легко підтримуваний код. Це особливо важливо для розробки складних систем, таких як симулятори медичних процедур, де важлива чітка структура та взаємодія між компонентами.

– Широка підтримка бібліотек: C# має велику кількість бібліотек і фреймворків, що дозволяють вирішувати різноманітні завдання, такі як робота з графікою, анімацією, фізикою та мережею. Це значно спрощує розробку і розширює можливості симулятора.

– Інтеграція з Unity: C# є основною мовою програмування для Unity. Це забезпечує тісну інтеграцію між кодом і функціональністю платформи, що дозволяє ефективно використовувати всі можливості Unity для створення реалістичних і інтерактивних симуляцій.

Платформи для розробки

Платформа Unity була обрана для розробки симулятора з кількох причин:

– Підтримка 3D та 2D графіки: Unity забезпечує високоякісну візуалізацію як для 3D, так і для 2D моделей, що є критично важливим для створення реалістичних медичних симуляцій.

– Інструменти для анімації: Unity має потужні інструменти для створення анімацій, які дозволяють відтворювати складні медичні процедури.

– Мультиплатформенність: Unity підтримує розробку для різних платформ, включаючи ПК, мобільні пристрої та VR, що дозволяє створювати гнучкі та доступні симулятори.

– Велика спільнота розробників: Unity має велику та активну спільноту, що забезпечує доступ до ресурсів, документації та підтримки.

Інструменти для візуалізації та інтеграції

Для забезпечення високої якості візуалізації медичних процедур використовуються наступні інструменти та бібліотеки:

– Blender: Вільний інструмент для створення та редагування 3D моделей. Blender дозволяє створювати детальні анатомічні моделі, які можуть бути інтегровані в Unity.

– Autodesk Maya: Потужний інструмент для 3D моделювання та анімації, який може використовуватися для створення складних моделей та анімацій медичних процедур.

Інтеграція з іншими системами

Для забезпечення інтеграції симулятора з іншими навчальними платформами та системами управління навчальним процесом використовуються наступні технології:

– REST API: Дозволяє забезпечити зв'язок між симулятором та зовнішніми системами, такими як системи управління навчальним процесом (LMS).

– SQL та NoSQL бази даних: Використовуються для зберігання даних про навчальні сценарії, результати студентів та іншу інформацію.

Вибір відповідних технологій для розробки симулятора медичних процедур є критично важливим для створення ефективної та гнучкої системи

навчання. Використання мови програмування C# та платформи Unity забезпечує високий рівень реалістичності та інтерактивності симуляцій, а інтеграція з іншими системами та забезпечення безпеки даних сприяє ефективному використанню симулятора у навчальних закладах.

2.2. Проектування інтерфейсу користувача

Проектування інтерфейсу користувача (UI) є ключовим аспектом розробки симулятора для тренування медичних процедур. Ефективний і зручний інтерфейс забезпечує легкість у використанні системи, швидкий доступ до необхідних функцій і підвищує загальну продуктивність навчання.

Основні принципи проектування інтерфейсу користувача включають інтуїтивність, простоту, функціональність та адаптивність.

Основні принципи проектування інтерфейсу користувача

– Інтуїтивність: інтерфейс повинен бути зрозумілим для користувача без необхідності додаткового навчання або довготривалого ознайомлення. Всі елементи управління мають бути розташовані логічно і відповідати загальним стандартам UI/UX дизайну.

– Простота: інтерфейс повинен бути максимально простим та не перевантаженим зайвими елементами. Користувачі мають швидко знаходити необхідні функції та виконувати свої завдання з мінімальними зусиллями.

– Функціональність: інтерфейс має забезпечувати доступ до всіх необхідних функцій симулятора, включаючи налаштування, запуск і зупинку симуляцій, збереження і завантаження даних, а також отримання зворотного зв’язку.

– Компоненти інтерфейсу користувача

– Головне меню: головне меню є основним навігаційним елементом, що забезпечує доступ до основних розділів симулятора, таких як початок симуляції, налаштування, довідка та вихід із програми.

– Панель управління симуляцією: ця панель забезпечує управління симуляцією в реальному часі. Вона включає кнопки для запуску, паузи та зупинки симуляції, а також елементи для вибору сценаріїв і налаштувань симуляції.

– Інформаційна панель: ця панель відображає важливу інформацію про поточний стан симуляції, а також результати виконаних процедур.

– Інтерактивні підказки: інтерактивні підказки допомагають користувачам швидко орієнтуватися у функціях симулятора. Вони надають інформацію про функції кнопок та інших елементів інтерфейсу, що допомагає користувачам легко зрозуміти, як працювати із системою.

– Зворотний зв’язок: система зворотного зв’язку дозволяє користувачам отримувати інформацію про їхні дії та результати симуляції. Це може включати повідомлення про помилки, підказки для покращення та оцінку виконаних процедур.

Процес проектування інтерфейсу

– Аналіз вимог користувачів: проектування інтерфейсу починається з аналізу вимог користувачів, щоб зрозуміти їхні потреби та очікування від системи. Це включає проведення опитувань, інтерв’ю та тестування з потенційними користувачами.

– Розробка прототипів: на основі аналізу вимог створюються прототипи інтерфейсу. Прототипи можуть бути як низької деталізації (скетчі на папері), так і високої деталізації (цифрові макети). Прототипи дозволяють візуалізувати ідеї та отримати початковий зворотний зв'язок від користувачів.

– Тестування прототипів: прототипи тестуються з реальними користувачами для виявлення проблем і збору зворотного зв'язку. На основі результатів тестування прототипи коригуються і вдосконалюються.

– Реалізація дизайну: після затвердження прототипів розпочинається реалізація інтерфейсу в середовищі розробки. Це включає створення графічних елементів, написання коду для взаємодії користувача з системою та інтеграцію з іншими компонентами симулятора.

– Тестування і вдосконалення: реалізований інтерфейс тестується для виявлення можливих помилок і проблем. Тестування проводиться як розробниками, так і кінцевими користувачами. На основі зворотного зв'язку інтерфейс вдосконалюється.

Приклади візуальних рішень

Для створення привабливого та функціонального інтерфейсу використовуються сучасні інструменти дизайну, такі як Adobe XD, Figma та Sketch. Ці інструменти дозволяють створювати інтерактивні макети та прототипи, які можна тестувати та вдосконалювати до початку реалізації.

Основні кроки для створення ефективного UI:

– Вибір кольорової схеми: Кольорова схема повинна бути приємною для очей і забезпечувати контрастність для легкої читабельності тексту та інших елементів.

– Типографіка: Вибір шрифтів і розмірів тексту має забезпечувати зручність читання та естетичний вигляд.

– Іконки та графічні елементи: Використання іконок та графічних елементів повинно допомагати орієнтуванню користувача та спрощувати взаємодію з системою.

Проектування інтерфейсу користувача є критично важливим для успіху симулятора медичних процедур. Ефективний UI забезпечує зручність, доступність та функціональність системи, що сприяє покращенню навчального процесу та підвищенню рівня підготовки медичних працівників. Врахування потреб користувачів, тестування прототипів та постійне вдосконалення інтерфейсу є ключовими аспектами, що дозволяють створити зручний та ефективний продукт.

3 РОЗРОБКА СИМУЛЯТОРА МЕДИЧНИХ ПРОЦЕДУР

3.1. Створення модулів для візуалізації медичних процедур

Візуалізація є ключовим компонентом симулятора медичних процедур, оскільки вона забезпечує реалістичне та інтерактивне навчальне середовище для студентів. Основною метою створення модулів для візуалізації є розробка детальних 3D-моделей анатомії та процедур, які дозволять студентам відпрацьовувати навички у реалістичних умовах.

Основні етапи створення модулів візуалізації

– Аналіз вимог та проектування Збір вимог: Визначення потреб користувачів (студентів та викладачів) щодо візуалізації медичних процедур.

– Проектування модулів: Розробка плану проектування, що включає визначення основних компонентів, функціональних можливостей і взаємодії між ними.

–

Створення 3D-моделей

– Моделювання анатомії: Використання Blender та Autodesk Maya для створення детальних 3D-моделей анатомічних структур, таких як органи, кістки та м'язи. Особлива увага приділяється точності та реалістичності моделей.

– Моделювання медичних інструментів: Створення моделей медичних інструментів, які будуть використовуватися в симуляції. Це включає шприци, скальпелі, катетери та інші інструменти, необхідні для виконання процедур.

Текстурування та освітлення

– Текстурування моделей: Використання Substance Painter для створення реалістичних текстур, що відображають зовнішній вигляд шкіри, органів та інших анатомічних структур.

– Налаштування освітлення: Створення сцен в Unity з реалістичним освітленням для забезпечення правильної візуалізації моделей. Це включає налаштування джерел світла, тіней та відблисків.

Анімація

– Анімація процедур: Розробка анімацій для різних медичних процедур, таких як перев'язка ран, введення ін'єкцій, введення катетера тощо. Використання Autodesk Maya для створення складних анімацій рухів та взаємодій.

– Інтерактивні анімації: Використання Unity для створення інтерактивних анімацій, що реагують на дії користувача. Це включає інтерактивні підказки та зворотний зв'язок під час виконання процедур.

Інтеграція з симулятором

– Імпорт моделей в Unity: Імпорт створених моделей та анімацій в Unity. Забезпечення правильної інтеграції з іншими компонентами симулятора, такими як фізика та система управління.

– Налаштування взаємодії: Розробка механізмів взаємодії користувача з моделями та анімаціями. Це включає налаштування системи введення, що дозволяє користувачам взаємодіяти з моделями за допомогою клавіатури або миші.

Тестування та оптимізація

– Тестування моделей та анімацій: Проведення тестування для забезпечення коректності роботи моделей та анімацій. Це включає тестування на різних пристроях і платформах для забезпечення сумісності та продуктивності.

– Оптимізація продуктивності: Виявлення та усунення проблем, що можуть впливати на продуктивність симулятора. Це включає оптимізацію моделей, зменшення кількості полігонів, покращення текстур та налаштування освітлення.

3.2. Розробка інтерактивних сценаріїв тренування

Розробка інтерактивних сценаріїв тренування є критичним аспектом створення ефективного симулятора медичних процедур. Інтерактивні сценарії дозволяють студентам відпрацьовувати навички у реалістичних умовах,

реагуючи на змінні ситуації та отримуючи зворотний зв'язок у режимі реального часу. Це підвищує їхню готовність до роботи у клінічних умовах і забезпечує всебічну підготовку.

Основні компоненти інтерактивних сценаріїв

Сценарії процедур

Сценарії процедур включають послідовність дій, які студент повинен виконати для успішного завершення медичної процедури. Це можуть бути сценарії введення ін'єкцій, перев'язки ран, введення катетера тощо.

Зміна стану пацієнта

Стан пацієнта може змінюватися у відповідь на дії студента. Це включає реакції організму на процедури, зміни життєвих показників та виникнення ускладнень. Такі зміни додають реалістичності та сприяють розвитку критичного мислення.

Зворотний зв'язок та оцінка

Система повинна надавати зворотний зв'язок студентам щодо їхніх дій, вказуючи на помилки та пропонуючи рекомендації для покращення. Оцінка результатів може включати як кількісні показники (балли, відсотки), так і якісний аналіз (описові коментарі).

Варіативність сценаріїв

Сценарії повинні включати різні варіанти розвитку подій, щоб студенти могли практикуватися у різних умовах і навчатися реагувати на непередбачувані ситуації. Це можуть бути різні ступені складності процедур, зміни стану пацієнта тощо.

Етапи розробки інтерактивних сценаріїв

Проектування сценаріїв: Розробка детальних сценаріїв для кожної процедури, включаючи послідовність дій, можливі ускладнення та варіанти розвитку подій.

Розробка сценаріїв

Написання скриптів: Використання C# для написання скриптів, що керують поведінкою системи під час виконання сценаріїв. Це включає

управління зміною стану пацієнта, реагування на дії студента та надання зворотного зв'язку.

Налаштування подій: Створення подій у Unity, які запускаються у відповідь на дії користувача. Це можуть бути зміни життєвих показників пацієнта, поява підказок або повідомлень про помилки.

Тестування та вдосконалення

Тестування сценаріїв: Проведення тестування сценаріїв для виявлення помилок та проблем у логіці. Це включає тестування різних варіантів розвитку подій та реакцій системи.

Збір зворотного зв'язку: Отримання зворотного зв'язку від студентів та викладачів щодо ефективності та реалістичності сценаріїв. На основі цього зворотного зв'язку сценарії можуть бути вдосконалені.

Впровадження у навчальний процес

Інтеграція сценаріїв: Інтеграція розроблених сценаріїв у симулятор. Забезпечення доступності сценаріїв для студентів та викладачів через інтерфейс користувача.

Приклади інтерактивних сценаріїв

– Сценарій введення ін'єкції

Студенту пропонується вибрати правильне місце для введення ін'єкції, підготувати шкіру пацієнта та виконати процедуру. Система відстежує кожен крок та надає зворотний зв'язок щодо правильності дій.

```
using UnityEngine;

[CreateAssetMenu(fileName = "InjectionScenario", menuName = "Scenarios/Injection")]
public class InjectionScenario : Scenario
{
    public GameObject patientModel;
    public Transform injectionSite;

    private GameObject patientInstance;

    public override void Initialize()
    {
        // Initialize the injection scenario
        if (patientInstance != null)
        {
            Destroy(patientInstance);
        }
        patientInstance = Instantiate(patientModel);
        // Add other initialization code here
    }

    public void PerformInjection()
    {
        // Code to handle injection
        Debug.Log("Injection performed at: " + injectionSite.position);
    }
}
```

Рис. 3.1 Сценарій введення ін'єкції

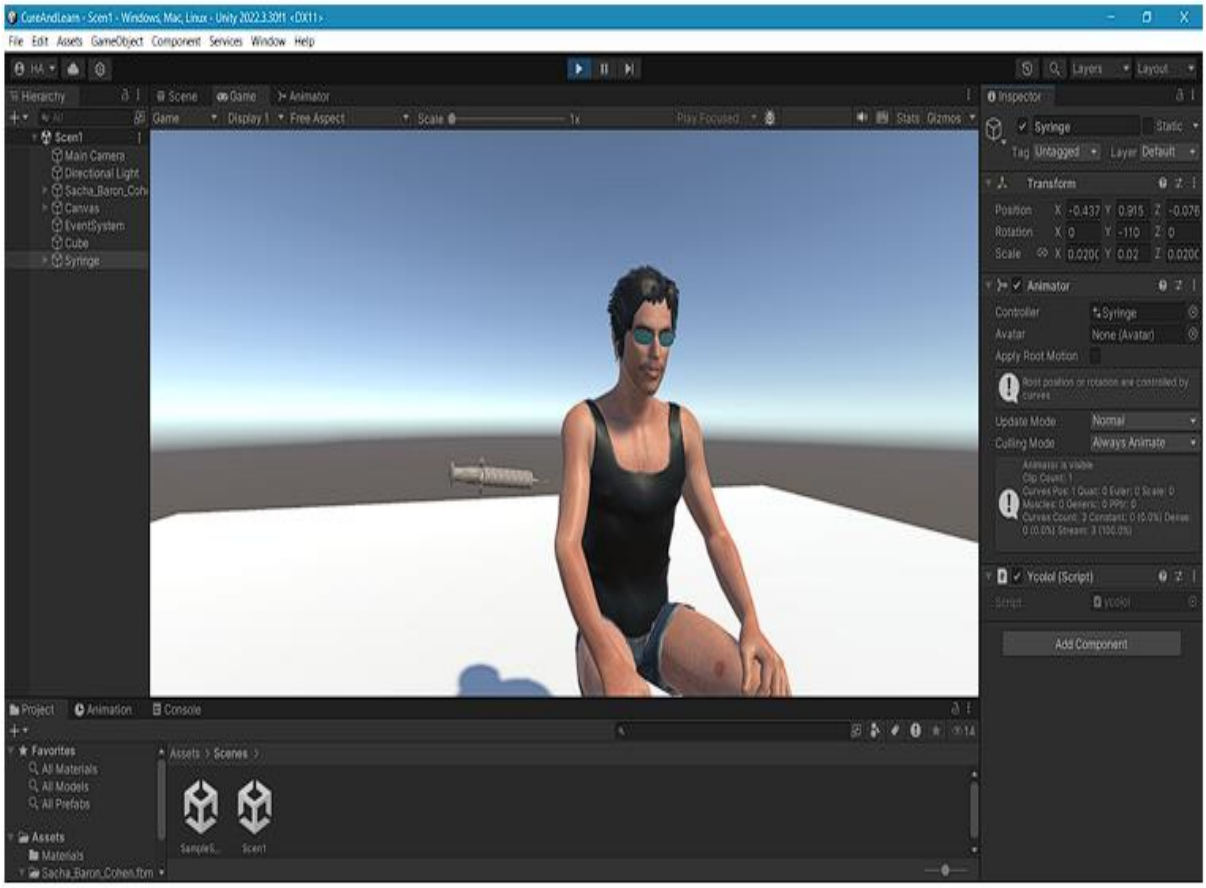


Рис. 3.2 Сцена 1 «Ін’єкція»

Сценарій перев'язки рани

– Студенту необхідно провести оцінку стану рани, вибрати відповідні матеріали для перев'язки та виконати процедуру. Система відстежує час виконання, точність дій та надає рекомендації для покращення.

```
using UnityEngine;

[CreateAssetMenu(fileName = "BandageScenario", menuName = "Scenarios/Bandage")]
public class BandageScenario : Scenario
{
    public GameObject patientModel;
    public Transform woundSite;
    public GameObject bandagePrefab;

    private GameObject patientInstance;

    public override void Initialize()
    {
        // Initialize the bandage scenario
        if (patientInstance != null)
        {
            Destroy(patientInstance);
        }
        patientInstance = Instantiate(patientModel);
        // Add other initialization code here
    }

    public void ApplyBandage()
    {
        // Code to handle bandage application
        Instantiate(bandagePrefab, woundSite.position, woundSite.rotation);
        Debug.Log("Bandage applied to: " + woundSite.position);
    }
}
```

Рис 3.3 Сценарій перев'язки рани

Сценарій введення катетера

– Студенту потрібно підготувати пацієнта, вибрати правильний тип катетера та виконати процедуру введення. Система контролює дії студента, виявляє помилки та надає зворотний зв'язок у режимі реального часу.

```
using UnityEngine;

[CreateAssetMenu(fileName = "CatheterScenario", menuName = "Scenarios/Catheter")]
public class CatheterScenario : Scenario
{
    public GameObject patientModel;
    public Transform catheterSite;

    private GameObject patientInstance;

    public override void Initialize()
    {
        // Initialize the catheter scenario
        if (patientInstance != null)
        {
            Destroy(patientInstance);
        }
        patientInstance = Instantiate(patientModel);
        // Add other initialization code here
    }

    public void InsertCatheter()
    {
        // Code to handle catheter insertion
        Debug.Log("Catheter inserted at: " + catheterSite.position);
    }
}
```

Рис 3.4 Сценарій введення катетера

3.3.Реалізація користувацького інтерфейсу

Реалізація користувацького інтерфейсу (UI) є важливим етапом розробки симулятора медичних процедур, оскільки саме від нього залежить зручність та ефективність використання системи студентами та викладачами. Інтерфейс повинен бути інтуїтивно зрозумілим, функціональним та адаптивним, забезпечуючи доступ до всіх необхідних функцій симулятора. Основні елементи користувацького інтерфейсу

Головне меню

– Функції: Доступ до основних розділів симулятора, таких як початок симуляції, налаштування, довідка та вихід із програми.

– Реалізація: Використання кнопок та випадаючих меню для організації доступу до різних функцій симулятора.

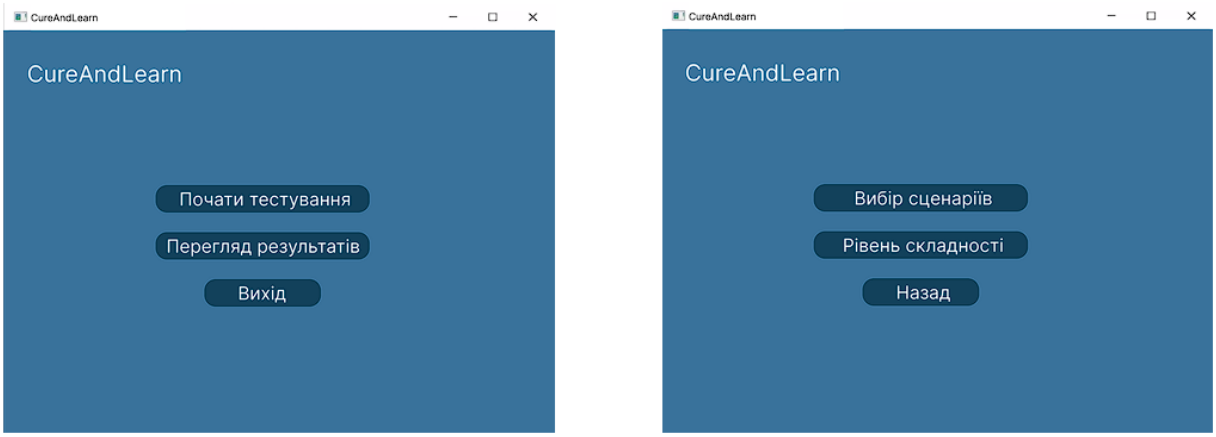


Рис. 3.5 Головне меню

Панель управління симуляцією

– Функції: Управління симуляцією в реальному часі, включаючи запуск, паузу та зупинку симуляції, а також вибір сценаріїв і налаштувань.

– Реалізація: Використання кнопок, слайдерів та випадаючих меню для забезпечення контролю над симуляцією.

Інформаційна панель

– Функції: Відображення важливої інформації про поточний стан симуляції, включаючи параметри пацієнта, а також результати виконаних процедур.

– Реалізація: Використання текстових полів, графіків та діаграм для надання інформації у зрозумілому вигляді.

Інтерактивні підказки

– Функції: Допомога користувачам у розумінні функцій інтерфейсу та виконанні процедур. Інтерактивні підказки можуть надавати інформацію про функції кнопок та інших елементів інтерфейсу.

– Реалізація: Використання спливаючих вікон, інфографіки та контекстних підказок для надання допомоги користувачам.

Система зворотного зв'язку

– Функції: Надання зворотного зв'язку щодо дій користувачів та результатів симуляції. Це може включати повідомлення про помилки, рекомендації для покращення та оцінку виконаних процедур.

– Реалізація: Використання повідомлень, спливаючих вікон та звітів для надання зворотного зв'язку користувачам.

Процес розробки користувацького інтерфейсу

Аналіз вимог користувачів

Збір вимог: Аналіз потреб користувачів (студентів та викладачів) щодо функціональності та зручності інтерфейсу.

Проектування прототипів: Створення прототипів інтерфейсу на основі зібраних вимог. Прототипи можуть бути як низької, так і високої деталізації, що дозволяє візуалізувати ідеї та отримати початковий зворотний зв'язок.

Приклади візуальних рішень

Головне меню

Дизайн головного меню може включати великі, чітко окреслені кнопки з інтуїтивно зрозумілими іконками та текстовими підказками. Фон може бути нейтральним, щоб не відволікати користувача.

Панель управління симуляцією

Панель управління симуляцією розташовується в нижній частині екрану, забезпечуючи швидкий доступ до основних функцій. Кнопки для запуску, паузи та зупинки симуляції мають бути великими і легко доступними.

Інформаційна панель

Інформаційна панель може бути розташована зліва або справа на екрані і включати текстові поля та графіки для відображення параметрів пацієнта. Використання кольорових індикаторів для різних станів (наприклад, зеленого для нормальних показників та червоного для критичних) допомагає швидко орієнтуватися.

Інтерактивні підказки

Інтерактивні підказки можуть бути реалізовані у вигляді спливаючих вікон або контекстних меню, які з’являються при наведенні курсору на певний елемент. Це допомагає користувачам швидко зрозуміти функції кожного елемента інтерфейсу.

Система зворотного зв'язку

Повідомлення про помилки та рекомендації можуть бути реалізовані у вигляді спливаючих вікон або нотифікацій, які з'являються в верхньому правому куті екрану. Це дозволяє користувачам швидко отримувати інформацію про свої дії та результати симуляції.

3.4 Інтеграція бази даних

Для забезпечення зберігання даних про користувачів, медичні процедури, результати тренувань та інші необхідні дані, в симуляторі використовується реляційна база даних. В якості СКБД (система керування базами даних) обрано MySQL, оскільки вона є надійною, масштабованою та підтримує багатокористувацький режим.

Структура бази даних

База даних складається з кількох основних таблиць:

Users (Користувачі): Зберігає інформацію про користувачів системи.

Procedures (Процедури): Зберігає інформацію про медичні процедури.

TrainingResults (Результати тренувань): Зберігає результати тренувань користувачів.

Settings (Налаштування): Зберігає налаштування системи.

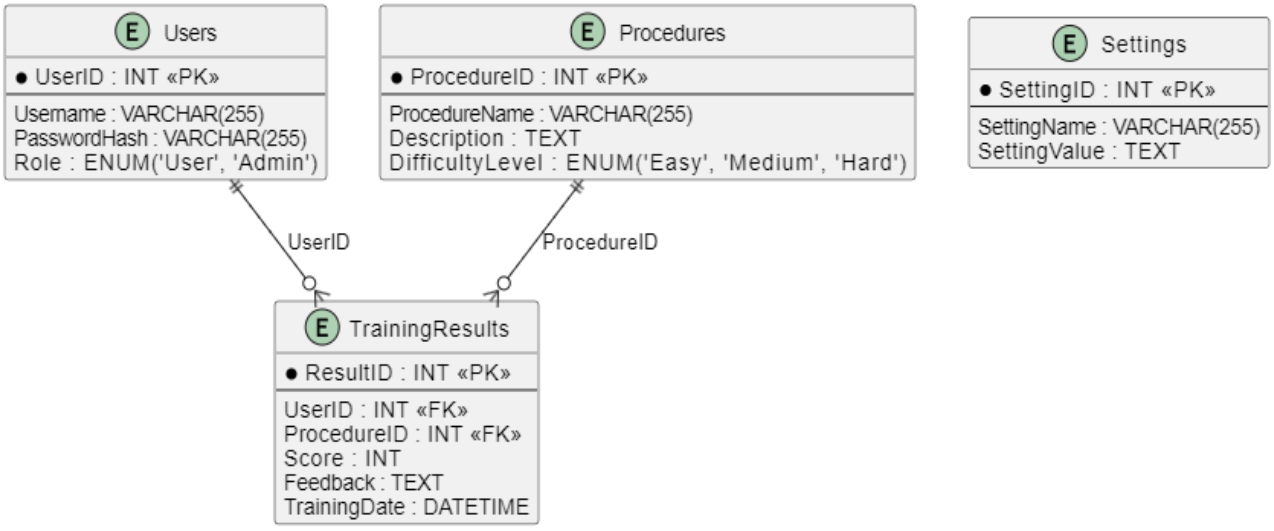


Рис 3.6 ER-діаграма

Виконання SQL-запиту

```
public void AddUser(string username, string passwordHash, string role)
{
    string query = $"INSERT INTO Users (Username, PasswordHash, Role) VALUES ('{username}', '{passwordHash}', '{role}')";
    ExecuteQuery(query);
}
```

3.5. Візуалізація структури та процесів симулятора

Основними завданнями цього етапу було створення діаграм, що ілюструють структуру системи, її функціональні можливості та взаємодію компонентів. Процес візуалізації включав декілька основних аспектів:

– Розробка діаграми сценаріїв використання (Use Case діаграми);
– Створення діаграми діяльності (Activity Diagram);
– Створення діаграми послідовності (Sequence Diagram);
– Створення діаграми класів (Class Diagram).

Рис. 3.7 Діаграма варіантів використання

Діаграма варіантів використання (Use Case діаграма) була створена для відображення основних функцій симулятора та взаємодії користувача з системою. На цій діаграмі користувач має можливість виконувати різні дії, такі як запуск симуляції, налаштування програми, вихід з програми, а також виконання конкретних медичних процедур: ін'єкцій, перев'язок та введення катетера. Крім того, користувач може отримувати підказки та надавати зворотний зв'язок. Адміністратор може додавати нові процедури, редагувати існуючі, видаляти їх та налаштовувати рівень складності.

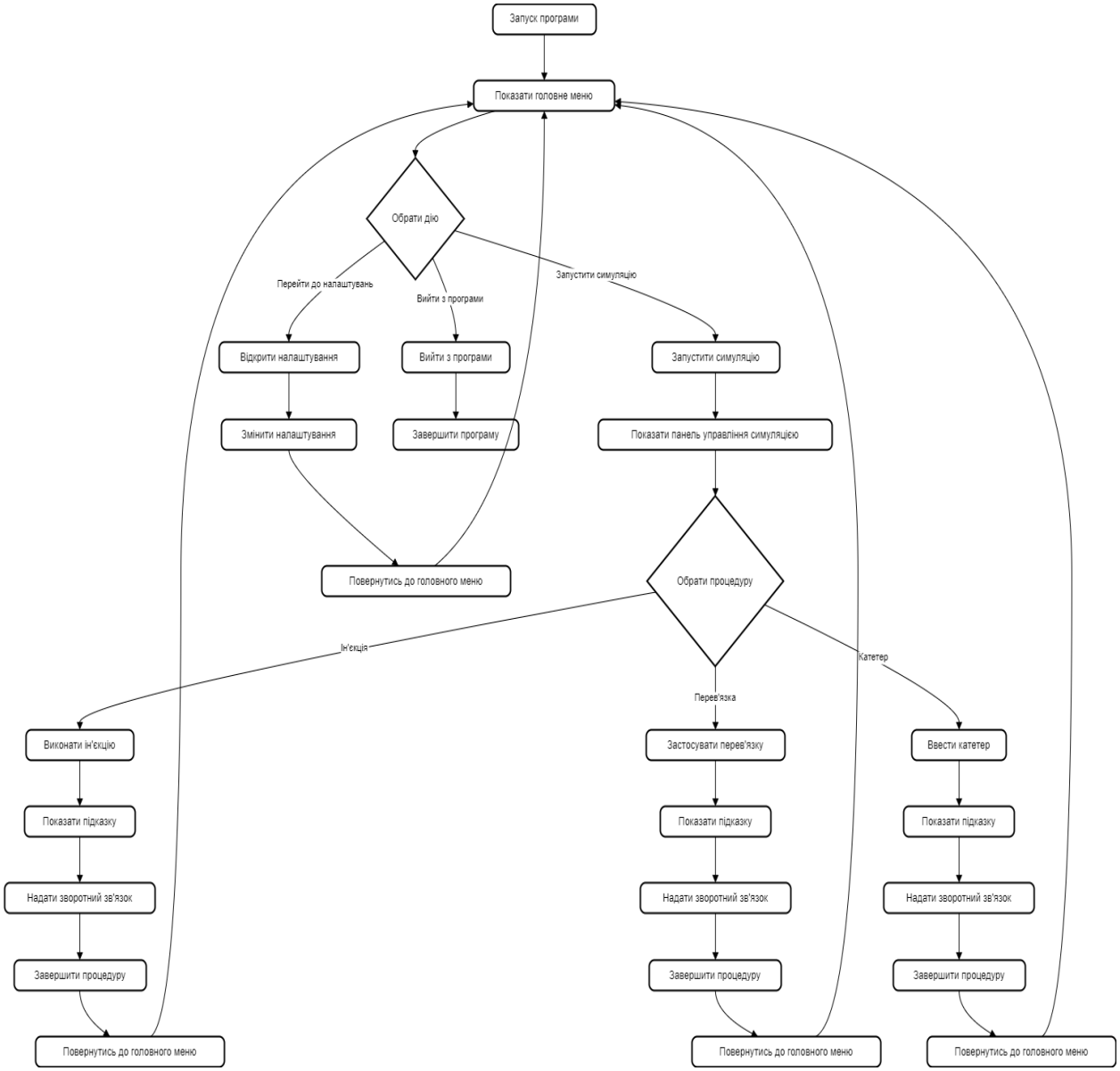


Рис. 3.8 Діаграма діяльності

Діаграма діяльності (Activity Diagram) була створена для відображення послідовності дій та логіки виконання основних процесів у системі. Вона показує, як користувач проходить через різні етапи використання симулятора, починаючи з запуску програми, вибору дії в головному меню, виконання медичних процедур, отримання підказок та зворотного зв'язку, і завершення симуляції.

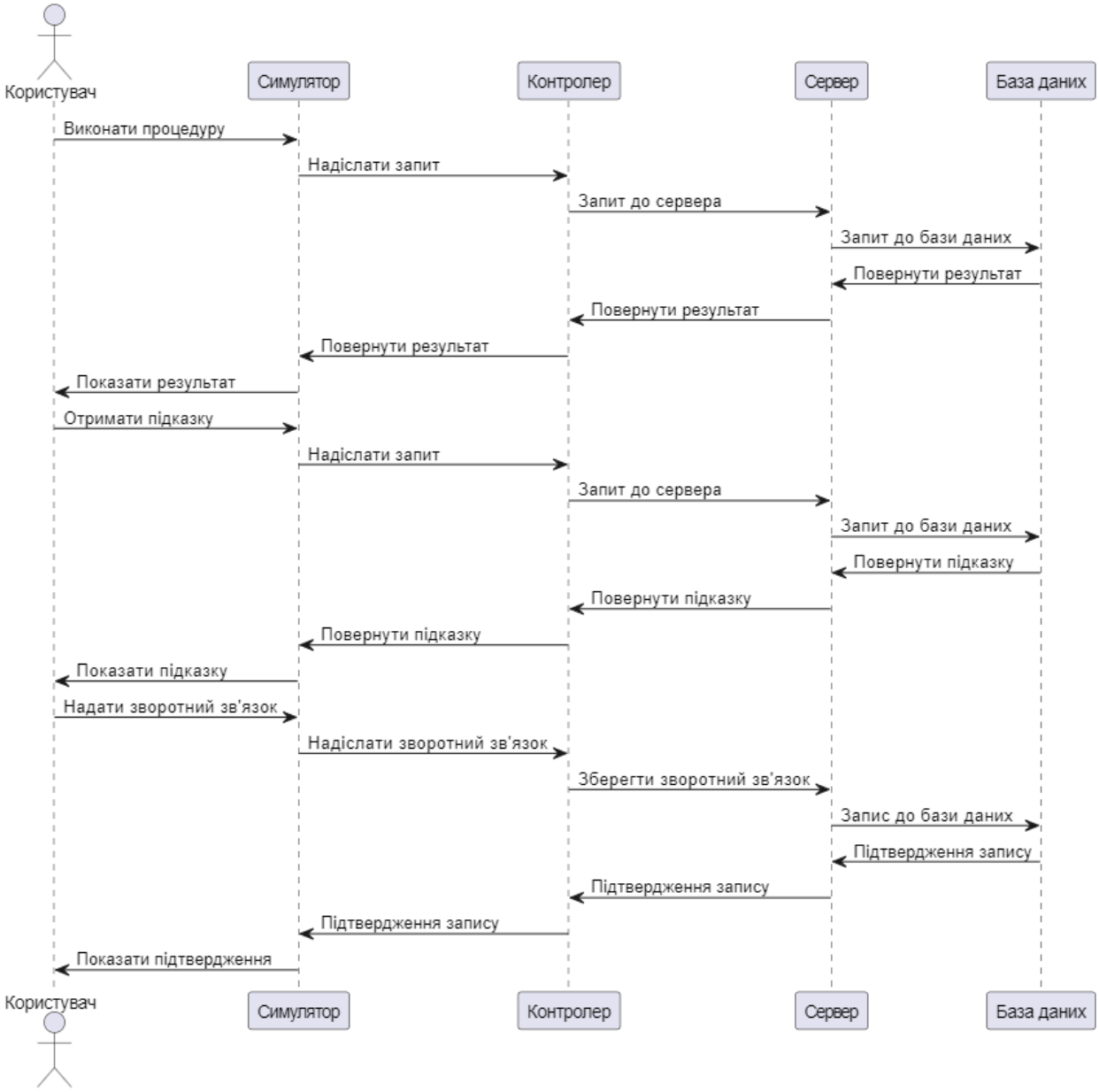


Рис. 3.9 Діаграма послідовності

Діаграма послідовності (Sequence Diagram) ілюструє процес обміну даними між користувачем, симулятором, контролером, сервером і базою даних. Ця діаграма демонструє, як дані передаються через різні компоненти системи для забезпечення виконання медичних процедур, отримання підказок та надання зворотного зв'язку.

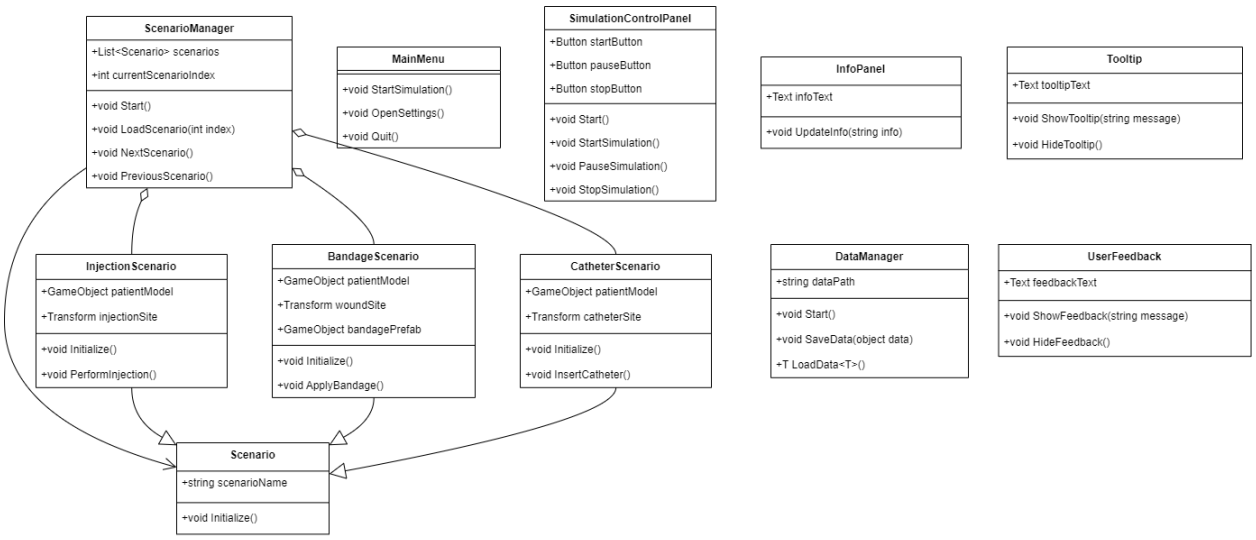


Рис. 3.10 Діаграма класів

Діаграма класів (Class Diagram) була створена для відображення структури системи. Вона показує основні класи, їх атрибути, методи та взаємозв'язки між ними. Ця діаграма ілюструє архітектуру симулятора медичних процедур, включаючи класи для різних сценаріїв, управління інтерфейсом користувача та зберігання даних.

3.6 Перевірка функціональності та користувацького інтерфейсу

Перевірка функціональності та користувацького інтерфейсу є критично важливим етапом розробки симулятора медичних процедур. Цей процес забезпечує виявлення та усунення помилок, підвищення якості системи та гарантує, що кінцевий продукт відповідає вимогам користувачів.

Основні етапи перевірки

- Планування тестування
- Визначення обсягу та методів тестування.
- Встановлення критеріїв прийняття та цілей тестування.
- Розробка тестових сценаріїв та планів.

Функціональне тестування

Тестування основних функцій: Перевірка базових функціональних можливостей симулятора, таких як запуск симуляції, управління процесами та збереження результатів.

Тестування сценаріїв: Перевірка всіх інтерактивних сценаріїв, включаючи введення ін'єкцій, перев'язку ран та введення катетера, на відповідність очікуваним результатам.

Тестування інтеграції: Перевірка взаємодії між різними компонентами симулятора, такими як модулі візуалізації, база даних та серверна частина.

Тестування користувацького інтерфейсу

Юзабіліті тестування: Оцінка зручності користування інтерфейсом. Включає тестування навігації, розташування елементів, зрозумілість та інтуїтивність інтерфейсу.

Тестування адаптивності: Перевірка коректності відображення інтерфейсу на різних пристроях і екранах різного розміру, включаючи ПК, планшети та мобільні пристрої.

Тестування доступності: Оцінка інтерфейсу на відповідність стандартам доступності для користувачів з обмеженими можливостями.

Тестування продуктивності

Тестування навантаження: Визначення, як система працює під високим навантаженням, коли велика кількість користувачів взаємодіє з симулятором одночасно.

Тестування стресу: Перевірка стабільності системи при екстремальних умовах роботи, таких як пікові навантаження або несподівані збої.

Процес тестування

Розробка тестових випадків

– Створення детальних тестових випадків, що описують конкретні дії та очікувані результати. Це включає функціональні, юзабіліті, продуктивні та безпекові аспекти.

Автоматизоване тестування

– Використання інструментів для автоматизованого тестування, таких як Selenium для веб-інтерфейсу та NUnit для тестування C# кодів. Це дозволяє швидко та ефективно перевіряти великий обсяг тестових випадків.

Ручне тестування

– Виконання тестових сценаріїв вручну для перевірки функцій, які важко автоматизувати. Це включає оцінку юзабіліті та адаптивності інтерфейсу.

Збір та аналіз результатів

– Збір результатів тестування, аналіз виявлених помилок та їх пріоритезація для подальшого виправлення. Використання баг-трекінгових систем, таких як Jira, для управління процесом виправлення помилок.

Виправлення помилок та повторне тестування

– Виправлення виявлених помилок та повторне тестування для підтвердження, що виправлення не викликали нових проблем і система працює коректно.

Приклади тестових випадків

Функціональний тестовий випадок

– Опис: Перевірка функції запуску симуляції введення ін'єкції.

– Дії: Користувач вибирає сценарій введення ін'єкції, запускає симуляцію, виконує процедуру, завершує симуляцію.

– Очікуваний результат: Система правильно реагує на всі дії користувача, надає зворотний зв'язок та зберігає результати симуляції.

Юзабіліті тестовий випадок

- Опис: Оцінка зручності навігації в головному меню.
- Дії: Користувач переходить між розділами головного меню (налаштування, довідка, запуск симуляції).
- Очікуваний результат: Меню інтуїтивно зрозуміле, всі елементи легко доступні, переходи відбуваються без затримок.

Продуктивний тестовий випадок

- Опис: Перевірка роботи системи під навантаженням.
- Дії: Одночасне виконання симуляції 50 користувачами.
- Очікуваний результат: Система витримує навантаження без значного зниження продуктивності, всі користувачі отримують коректний зворотний зв'язок.

ВИСНОВКИ

Розробка симулятора медичних процедур мовою C# з використанням Unity є важливим кроком у напрямку створення сучасних навчальних інструментів для медичної освіти. Цей проект дозволяє підвищити якість підготовки медичних працівників, забезпечуючи їм можливість тренуватися у виконанні складних медичних процедур в умовах, максимально наближених до реальних.

В процесі роботи були вирішені наступні завдання:

Аналіз існуючих рішень та вимог до системи:

Було проведено дослідження існуючих аналогів симуляторів медичних процедур та визначено їхні переваги і недоліки.

На основі проведеного аналізу були визначені функціональні вимоги до нової системи, що включали в себе необхідність забезпечення високої реалістичності, інтерактивності та можливості зворотного зв'язку.

Вибір технологій та проектування системи:

Для розробки симулятора було обрано мову програмування C# та платформу Unity, що дозволило створити високоякісну тривимірну графіку та забезпечити інтерактивність.

Було створено проектну документацію, яка включала діаграми варіантів використання, діяльності, послідовності та класів. Ці діаграми допомогли визначити структуру системи, її компоненти та їх взаємодію.

Розробка симулятора:

Було реалізовано кілька модулів симулятора, таких як модулі для візуалізації медичних процедур (ін'єкції, перев'язки, введення катетера), модулі управління симуляцією, надання підказок та зворотного зв'язку.

Особлива увага приділялася створенню зручного та інтуїтивно зрозумілого користувацького інтерфейсу, який забезпечував легкий доступ до всіх функцій симулятора.

Тестування та впровадження:

Проведено детальне тестування симулятора для перевірки його функціональності та виявлення можливих помилок.

В результаті тестування були зроблені необхідні корекції та покращення, що дозволило забезпечити стабільну та надійну роботу системи.

Розробка додаткових функцій для адміністраторів:

Було реалізовано можливості для адміністраторів, такі як додавання нових процедур, редагування існуючих, видалення процедур та налаштування рівня складності.

Це дозволило зробити систему більш гнучкою та адаптивною до потреб навчальних закладів.

Переваги розробленого симулятора

Розроблений симулятор має кілька ключових переваг:

Висока реалістичність: Використання Unity дозволяє створити високоякісну тривимірну графіку, що робить симуляції максимально наближеними до реальних умов.

Інтерактивність: Користувачі можуть взаємодіяти з системою в режимі реального часу, виконуючи медичні процедури та отримуючи миттєвий зворотний зв'язок.

Модульність: Система побудована на модульній архітектурі, що дозволяє легко додавати нові функції та процедури без значних змін у вже існуючому коді.

Доступність для адміністраторів: Адміністратори мають можливість налаштовувати систему під конкретні потреби, додаючи або редагуючи медичні процедури та встановлюючи рівні складності.

Впровадження та майбутні перспективи

Розроблений симулятор медичних процедур може бути впроваджений у навчальні програми медичних навчальних закладів для покращення практичної підготовки студентів. Він може використовуватися як інструмент для регулярних тренувань, так і для оцінювання навичок студентів.

Майбутні перспективи розвитку симулятора включають:

Розширення бібліотеки медичних процедур: Додавання нових процедур для покриття більш широкого спектру медичних маніпуляцій.

Покращення зворотного зв'язку: Розробка більш детальних алгоритмів надання зворотного зв'язку, включаючи аналіз помилок та рекомендації щодо покращення.

Мультимедійні навчальні матеріали: Інтеграція з мультимедійними навчальними матеріалами для забезпечення комплексного навчання.

Підтримка багатокористувацького режиму: Впровадження функціоналу для одночасного використання симулятора кількома користувачами, що дозволить проводити командні тренування.

Заключення

Розробка симулятора медичних процедур мовою C# з використанням Unity є значущим внеском у розвиток медичної освіти. Цей проект не тільки дозволяє підвищити рівень практичної підготовки студентів, але й сприяє покращенню якості медичних послуг в цілому. Завдяки високій реалістичності, інтерактивності та гнучкості, розроблений симулятор стане незамінним інструментом для навчання майбутніх медичних працівників.

Робота пройшла апробацію:

Артеменко М.А., Аверічев І.М. Створення симулятору для тренування медичних процедур. Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT», 18 квітня 2024 р., Київ, Державний університет інформаційно-комунікаційних технологій. .Збірник тез К.: ДУІКТ, 2024. С. 94-95

Артеменко М.А., Аверічев І.М. Створення тренажера для навчання медичним процедурам використовуючи «Unity» та «C#» // Всеукраїнська Науково-технічна конференція «Застосування програмного забезпечення в ІКТ», 24 квітня 2024 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез К.: ДУІКТ,2024. С. 24-25

ПЕРЕЛІК ПОСИЛАНЬ

1. Alinier, G. (2007). A typology of educationally focused medical simulation tools. Medical Teacher, 29(8), e243-e250.

2. Bradley, P. (2006). The history of simulation in medical education and possible future directions. Medical Education, 40(3), 254-262.

3. Gaba, D. M. (2004). The future vision of simulation in healthcare. Quality and Safety in Health Care, 13(suppl 1), i2-i10.

4. Issenberg, S. B., McGaghie, W. C., Petrusa, E. R., Lee Gordon, D., & Scalese, R. J. (2005). Features and uses of high-fidelity medical simulations that lead to effective learning: a BEME systematic review. Medical Teacher, 27(1), 10-28.

5. Cook, D. A., Hatala, R., Brydges, R., Zendejas, B., Szostek, J. H., Wang, A. T., ... & Hamstra, S. J. (2011). Technology-enhanced simulation for health professions education: a systematic review and meta-analysis. JAMA, 306(9), 978-988.

6. Dieckmann, P., Gaba, D., & Rall, M. (2007). Deepening the theoretical foundations of patient simulation as social practice. Simulation in Healthcare, 2(3), 183-193.

7. Kneebone, R. (2005). Evaluating clinical simulations for learning procedural skills: a theory-based approach. Academic Medicine, 80(6), 549-553.

8. Ziv, A., Wolpe, P. R., Small, S. D., & Glick, S. (2006). Simulation-based medical education: an ethical imperative. Academic Medicine, 81(9), 819-824.

9. Gaba, D. M., Howard, S. K., Fish, K. J., Smith, B. E., & Sowb, Y. A. (2001). Simulation-based training in anesthesia crisis resource management (ACRM): a decade of experience. Simulation & Gaming, 32(2), 175-193.

10. McGaghie, W. C., Siddall, V. J., Mazmanian, P. E., & Myers, J. (2009). Lessons for continuing medical education from simulation research in undergraduate and graduate medical education: learning outcomes by level of fidelity. Academic Medicine, 84(7), 957-968.

11. Norman, G. (2014). Simulation and the development of clinical reasoning: the role of practice. Medical Education, 48(3), 230-238.

12. Rosen, K. R. (2008). The history of medical simulation. Journal of Critical Care, 23(2), 157-166.

13. Aggarwal, R., & Darzi, A. (2006). Technical-skills training in the 21st century. New England Journal of Medicine, 355(25), 2695-2696.

14. Gorman, P. J., Meier, A. H., & Krummel, T. M. (1999). Simulation and virtual reality in surgical education: real or unreal?. Archives of Surgery, 134(11), 1203-1208.

15. Lateef, F. (2010). Simulation-based learning: Just like the real thing. Journal of Emergencies, Trauma, and Shock, 3(4), 348-352.

16. Okuda, Y., Bryson, E. O., DeMaria, S., Jacobson, L., Quinones, J., Shen, B., & Levine, A. I. (2009). The utility of simulation in medical education: what is the evidence?. Mount Sinai Journal of Medicine: A Journal of Translational and Personalized Medicine, 76(4), 330-343.

17. Wayne, D. B., Butter, J., Siddall, V. J., Fudala, M. J., Wade, L. D., Feinglass, J., ... & McGaghie, W. C. (2006). Mastery learning of advanced cardiac life support skills by internal medicine residents using simulation technology and deliberate practice. Journal of General Internal Medicine, 21(3), 251-256.

18. Arthur, W., Bennett, W., Stanush, P. L., & McNelly, T. L. (1998). Factors that influence skill decay and retention: A quantitative review and analysis. Human Performance, 11(1), 57-101.

19. Sawyer, T., White, M., Zaveri, P., Chang, T., Ades, A., French, H., ... & Anderson, J. M. (2015). Learn, see, practice, prove, do, maintain: an evidence-based pedagogical framework for procedural skill training in medicine. Academic Medicine, 90(8), 1025-1033.

20. Maran, N. J., & Glavin, R. J. (2003). Low- to high-fidelity simulation – a continuum of medical education?. Medical Education, 37(s1), 22-28.

21. Nishisaki, A., Keren, R., & Nadkarni, V. (2007). Does simulation improve patient safety? Self-efficacy, competence, operational performance, and patient safety. Anesthesiology Clinics, 25(2), 225-236.

22. Ericsson, K. A. (2004). Deliberate practice and the acquisition and maintenance of expert performance in medicine and related domains. Academic Medicine, 79(10), S70-S81.

23. Huang, G. C., Newman, L. R., & Schwartzstein, R. M. (2014). Procedural skills education and assessment: a survey of gastrointestinal fellowship program directors. Gastrointestinal Endoscopy, 79(6), 960-966.

24. Lee, J., Lee, H., Kim, S., Choi, M., Lee, J., & Kim, Y. (2020). Effects of high-fidelity patient simulation-led clinical reasoning course: Focused on the clinical reasoning, self-reflection and learning satisfaction. Nurse Education Today, 84, 104246.

25. Gordon, J. A., Oriol, N. E., & Cooper, J. B. (2004). Bringing good teaching cases “to life”: a simulator-based medical education service. Academic Medicine, 79(1), 23-27.

26. Nestel, D., Groom, J., Eikeland-Husebø, S., & O’Donnell, J. M. (2011). Simulation for learning and teaching procedural skills: the state of the science. Simulation in Healthcare, 6(S), S10-S13.

27. McGaghie, W. C., Issenberg, S. B., Cohen, E. R., Barsuk, J. H., & Wayne, D. B. (2011). Does simulation-based medical education with deliberate practice yield better results than traditional clinical education? A meta-analytic comparative review of the evidence. Academic Medicine, 86(6), 706-711.

28. Reznick, R. K., & MacRae, H. (2006). Teaching surgical skills—changes in the wind. New England Journal of Medicine, 355(25), 2664-2669.

29. Dunn, W. F. (2011). Simulators in critical care and beyond. Society of Critical Care Medicine.

30. McDougall, E. M. (2007). Simulation in education for health care professionals. World Journal of Urology, 25(3), 283-289.

31. Kneebone, R., Nestel, D., Vincent, C., & Darzi, A. (2007). Complexity, risk and simulation in learning procedural skills. Medical Education, 41(8), 808-814.

32. Scalese, R. J., Obeso, V. T., & Issenberg, S. B. (2008). Simulation technology for skills training and competency assessment in medical education. Journal of General Internal Medicine, 23(1), 46-49.

33. Kneebone, R. L., Scott, W., Darzi, A., & Horrocks, M. (2004). Simulation and clinical practice: strengthening the relationship. Medical Education, 38(10), 1095-1102.

34. Kapadia, M. R., DaRosa, D. A., MacRae, H. M., & Dunnington, G. L. (2007). Current assessment and future directions of surgical skills training in North America. The American Journal of Surgery, 194(1), 161-167.

35. Small, S. D., & Wuerz, R. C. (1997). Simulation: A way to teach and assess technical skills and crisis management. Annals of Emergency Medicine, 30(6), 687-689.

36. Stenfors-Hayes, T., Hult, H., & Dahlgren, L. O. (2011). What does it mean to be a good teacher and clinical supervisor in medical education? Advances in Health Sciences Education, 16(2), 197-210.

37. Bond, W. F., & Spillane, L. (2002). The use of simulation for emergency medicine resident assessment. Academic Emergency Medicine, 9(11), 1295-1299.

38. Ziv, A., Small, S. D., & Wolpe, P. R. (2000). Patient safety and simulation-based medical education. Medical Teacher, 22(5), 489-495.

39. Issenberg, S. B., McGaghie, W. C., Hart, I. R., Mayer, J. W., Felner, J. M., Petrusa, E. R., ... & Ewy, G. A. (1999). Simulation technology for health care professional skills training and assessment. JAMA, 282(9), 861-866.

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Створення симулятору для тренування медичних процедур мовою C# та з використанням Unity

Виконав студент 4 курсу
Групи ПД-43
Артеменко Микола Анатолійович
Керівник роботи
К.е.н., доцент кафедри ІПЗ Аверічев Ігор Миколайович

Київ – 2024

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи –** поліпшення навчання студентів-медиків напрямку «загальна медицина» з використанням інтерактивного симулятора для тренування медичних процедур.
- **Об'єкт дослідження -** процес підготовки студентів-медиків напрямку «загальна медицина» до виконання медичних процедур.
- **Предмет дослідження -** інтерактивний симулятор медичних процедур, розроблений за допомогою C# та Unity.

2

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати існуючі симулятори медичних процедур для загальної медицини.
2. Визначити вимоги до розроблюваного симулятора.
3. Проаналізувати інструменти для реалізації симулятора на C# з використанням Unity.
4. Реалізувати основні функції симулятора на C# з використанням Unity.
5. Провести тестування симулятора на C# з використанням Unity.

3

АНАЛІЗ АНАЛОГІВ

Симулятор	TraumaMan	Vimedix	SimMan 3G	Nursing Anne	CureAndLearn
Підтримка навчальних матеріалів	Включає текстові матеріали та відеоуроки	Інтерактивні інструкції, відеоуроки, аналітика	Навчальні матеріали, відеоуроки	Текстові матеріали, інструкції	Інтерактивні інструкції, тестування
Зворотній зв'язок	Реалістична реакція	Аналіз дій користувача	Покрокові інструкції	Тактильний зворотній зв'язок	Інтерактивний зворотній зв'язок
Використання інтерактивних технологій	Обмежені інтерактивні можливості	Використання VR технологій	Інтерактивні сценарії та симуляції	Мінімальне використання інтерактивних технологій	Використання тривимірних моделей
Реалістичність	Висока	Висока	Висока	Середня	Висока
Платформа	Фізичний манекен	ПК, VR	Фізичний манекен	Фізичний манекен	ПК
Набір сценаріїв	Травми, екстрена допомога	Ультразвукова діагностика	Реанімація, кардіологія	Основні медсестринські процедури	Загальні медичні процедури

4

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги:

1. Вибір і запуск різних медичних процедур для тренування.
2. Вибір рівня складності медичних процедур.
3. Надання зворотнього зв’язка щодо правильності виконання процедур.

Нефункціональні вимоги:

1. Вибір медичної процедури із списку (вакцинація, місцева анестезія, перев’язка тощо)
2. Використання тривимірних моделей.
3. Інтерактивні підказки та навчальні матеріали.
4. Українська локалізація

5

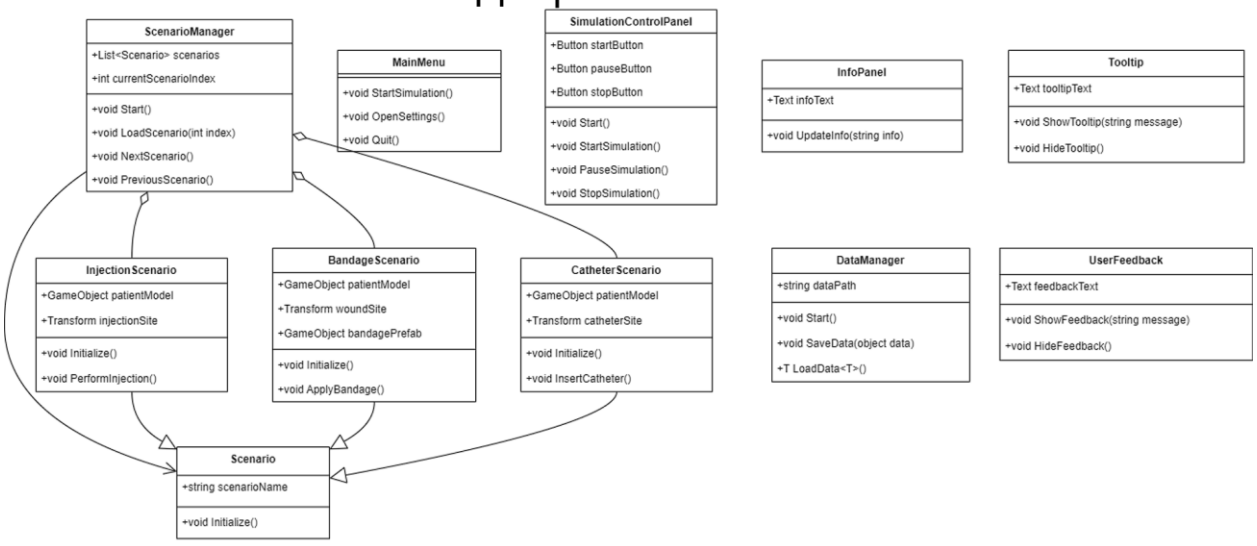
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

6

Діаграми варінтів використання симулятору

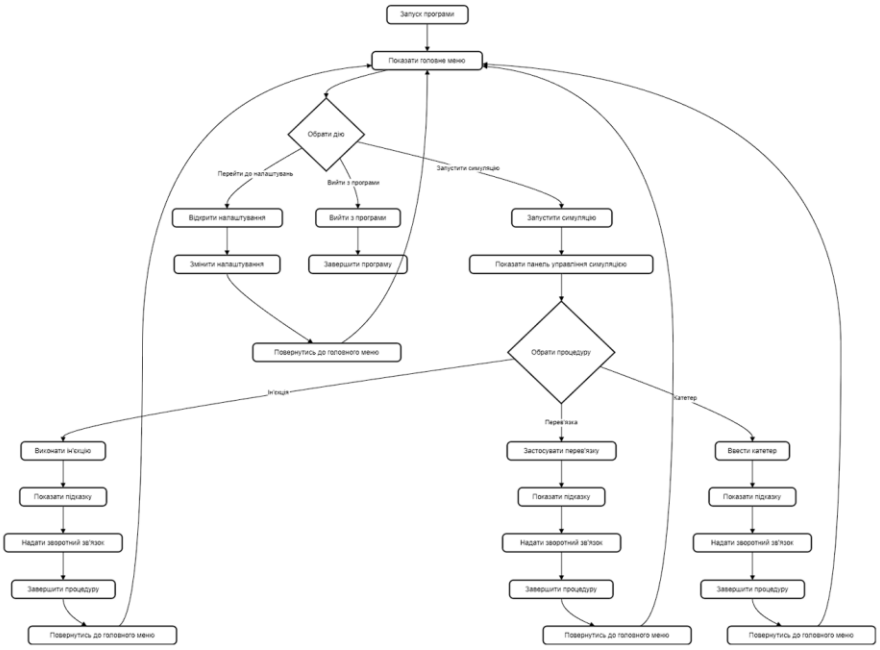
7

Діаграма класів



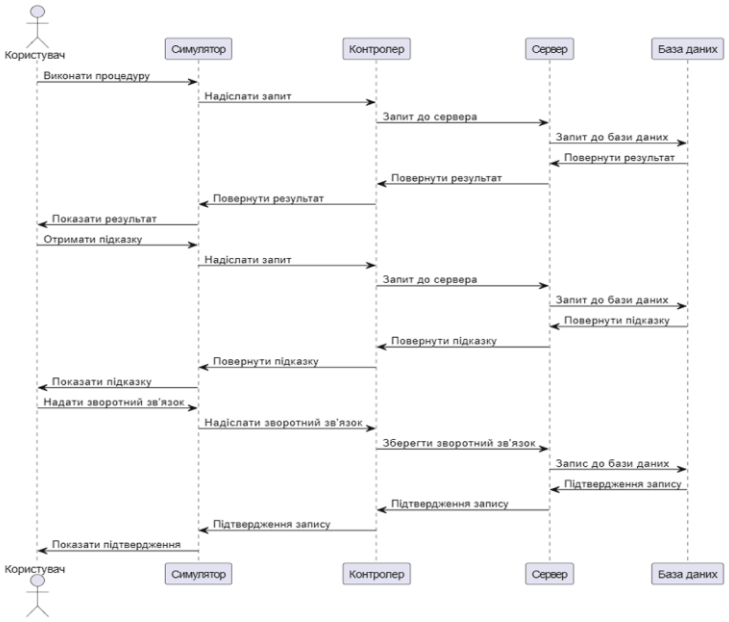
8

Діаграма діяльності



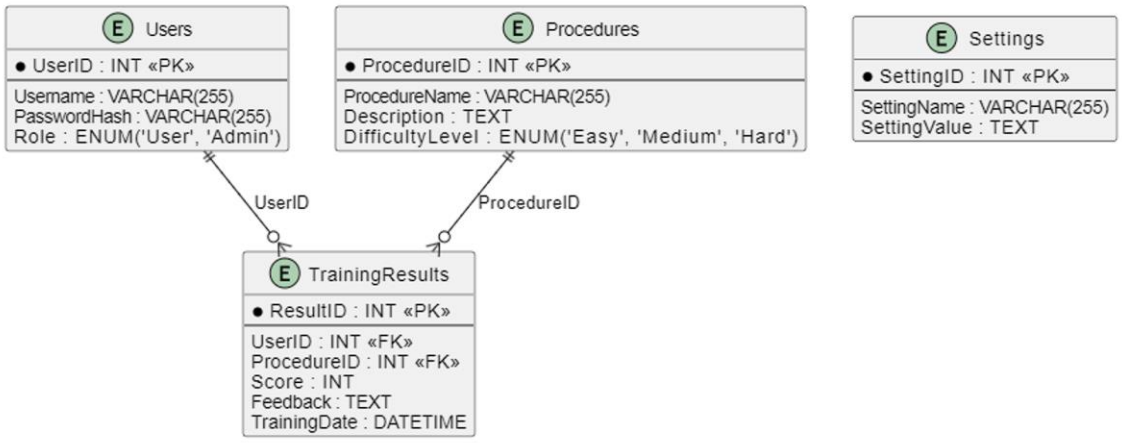
9

Діаграма послідовності



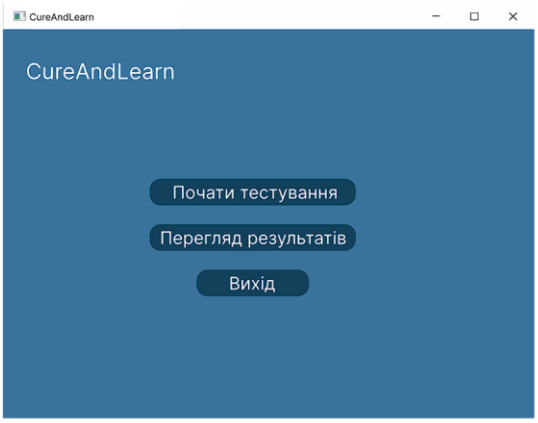
10

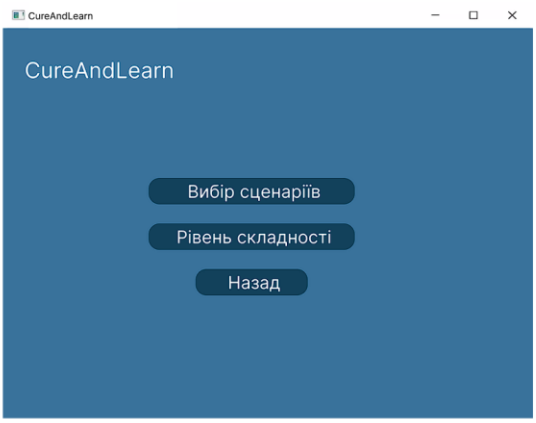
ER-діаграма



11

Екрані форми

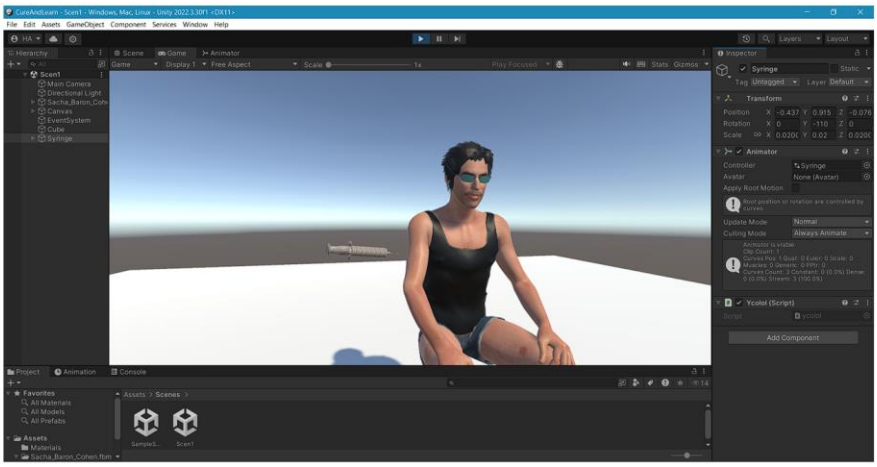




Головне меню

12

Екрані форми



Сцена № 1 – «Ін’єкція»

13

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Артеменко М.А., Аверічев І.М. Створення симулятору для тренування медичних процедур. Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT», 18 квітня 2024 р., Київ, Державний університет інформаційно-комунікаційних технологій. .Збірник тез К.: ДУІКТ, 2024. С. 94-95

Артеменко М.А., Аверічев І.М. Створення тренажера для навчання медичним процедурам використовуючи «Unity» та «C#» // Всеукраїнська Науково-технічна конференція «Застосування програмного забезпечення в ІКТ», 24 квітня 2024 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез К.: ДУІКТ,2024. С. 24-25

14

ВИСНОВКИ

1. Проаналізовано предметну галузь симуляторів медичних процедур для загальної медицини.
2. Проведено аналіз впливу використання симулятора для тренування медичних процедур.
3. Визначено вимоги до розроблюваного симулятора.
4. Проаналізовано інструменти для реалізації симулятора на C# з використанням Unity.
5. Релізовані основні функції симулятора на C# з використанням Unity.
6. Проведено тестування симулятора на C# з використанням Unity.

ДОДАТОК Б. ЛІСТИНГИ ОСНОВНИХ МОДУЛІВ

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

public class ScenarioManager :
MonoBehaviour
{
    public List<Scenario> scenarios;
    private int currentScenarioIndex = 0;

    public void Start()
    {
        LoadScenario(currentScenarioIndex);
    }

    public void LoadScenario(int index)
    {
        if (index >= 0 && index <
scenarios.Count)
        {

scenarios[currentScenarioIndex].gameObject.
SetActive(false);
            currentScenarioIndex = index;

scenarios[currentScenarioIndex].Initialize();

scenarios[currentScenarioIndex].gameObject.
SetActive(true);
        }
    }

    public void NextScenario()
    {
        LoadScenario((currentScenarioIndex +
1) % scenarios.Count);
    }

    public void PreviousScenario()
    {
        LoadScenario((currentScenarioIndex - 1
+ scenarios.Count) % scenarios.Count);
    }
}
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour
{
    public void StartSimulation()
    {

SceneManager.LoadScene("SimulationScene"
);
    }

    public void OpenSettings()
    {

SceneManager.LoadScene("SettingsScene");
    }

    public void Quit()
    {
        Application.Quit();
    }
}
using UnityEngine;
using UnityEngine.UI;

public class SimulationControlPanel :
MonoBehaviour
{
    public Button startButton;
    public Button pauseButton;
    public Button stopButton;

    public void Start()
    {

startButton.onClick.AddListener(StartSimulat
ion);

pauseButton.onClick.AddListener(PauseSimu
lation);

stopButton.onClick.AddListener(StopSimulati
on);
    }

    public void StartSimulation()
    {
        // Code to start simulation
        Debug.Log("Simulation started");
    }

    public void PauseSimulation()
    {
        // Code to pause simulation
        Debug.Log("Simulation paused");
    }

    public void StopSimulation()
    {
        // Code to stop simulation
        Debug.Log("Simulation stopped");
    }
}
using UnityEngine;
using UnityEngine.UI;

public class InfoPanel : MonoBehaviour
{
```

```
    public Text infoText;

    public void UpdateInfo(string info)
    {
        infoText.text = info;
    }
}
```

```
using UnityEngine;
using UnityEngine.UI;

public class Tooltip : MonoBehaviour
{
    public Text tooltipText;

    public void ShowTooltip(string message)
    {
        tooltipText.text = message;
        tooltipText.gameObject.SetActive(true);
    }

    public void HideTooltip()
    {
        tooltipText.gameObject.SetActive(false);
    }
}
```

```
using UnityEngine;
using UnityEngine.UI;

public class UserFeedback : MonoBehaviour
{
    public Text feedbackText;

    public void ShowFeedback(string message)
    {
        feedbackText.text = message;

feedbackText.gameObject.SetActive(true);
    }

    public void HideFeedback()
    {

feedbackText.gameObject.SetActive(false);
    }
}
```

```
CREATE TABLE Users (
    UserID INT AUTO_INCREMENT
PRIMARY KEY,
    Username VARCHAR(255) NOT NULL,
    PasswordHash VARCHAR(255) NOT
NULL,
    Role ENUM('User', 'Admin') NOT NULL
);

CREATE TABLE Procedures (
    ProcedureID INT AUTO_INCREMENT
PRIMARY KEY,
    ProcedureName VARCHAR(255) NOT
NULL,
    Description TEXT,
    DifficultyLevel ENUM('Easy', 'Medium',
'Hard') NOT NULL
);
```

```
CREATE TABLE TrainingResults (
    ResultID INT AUTO_INCREMENT
PRIMARY KEY,
    UserID INT,
    ProcedureID INT,
    Score INT,
    Feedback TEXT,
    TrainingDate DATETIME DEFAULT
CURRENT_TIMESTAMP,
    FOREIGN KEY (UserID) REFERENCES
Users(UserID),
    FOREIGN KEY (ProcedureID)
REFERENCES Procedures(ProcedureID)
);

CREATE TABLE Settings (
    SettingID INT AUTO_INCREMENT
PRIMARY KEY,
    SettingName VARCHAR(255) NOT
NULL,
    SettingValue TEXT
);
```

```
using System;
using MySql.Data.MySqlClient;
using UnityEngine;

public class DatabaseManager :
MonoBehaviour
{
    private MySqlConnection connection;

    private void Start()
    {
        string connectionString =
"Server=yourserver;Database=yourdatabase;U
ser
ID=yourusername;Password=yourpassword;P
ooling=false";
        connection = new
MySqlConnection(connectionString);

        try
        {
            connection.Open();
            Debug.Log("Database connection
established.");
        }
        catch (Exception ex)
        {
            Debug.LogError("Error connecting to
database: " + ex.Message);
        }
    }

    private void OnDestroy()
    {
        if (connection != null)
        {
            connection.Close();
        }
    }
```

```
    public void ExecuteQuery(string query)
    {
        MySqlCommand cmd = new
MySqlCommand(query, connection);
        try
        {
            cmd.ExecuteNonQuery();
            Debug.Log("Query executed
successfully.");
        }
        catch (Exception ex)
        {
            Debug.LogError("Error executing
query: " + ex.Message);
        }
    }
}
    public void AddUser(string username, string
passwordHash, string role)
    {
        string query = $"INSERT INTO Users
(Username, PasswordHash, Role) VALUES
('{username}', '{passwordHash}', '{role}')";
        ExecuteQuery(query);
}
    public void GetUsers()
    {
        string query = "SELECT * FROM Users";
        MySqlCommand cmd = new
MySqlCommand(query, connection);

        try
        {
            MySqlDataReader reader =
cmd.ExecuteReader();
            while (reader.Read())
            {
                Debug.Log("User ID: " +
reader["UserID"] + ", Username: " +
reader["Username"]);
            }
            reader.Close();
        }
        catch (Exception ex)
        {
            Debug.LogError("Error reading from
database: " + ex.Message);
        }
}
    using UnityEngine;
    using UnityEngine.UI;

    public class UIManager : MonoBehaviour
    {
        public InputField usernameField;
        public InputField passwordField;
        public Dropdown roleDropdown;

        public DatabaseManager dbManager;

        public void OnAddUserButtonClicked()
        {
            string username = usernameField.text;
            string passwordHash =
HashPassword(passwordField.text);
            string role =
roleDropdown.options[roleDropdown.value].t
ext;

            dbManager.AddUser(username,
passwordHash, role);
        }

        private string HashPassword(string
password)
        {
            // Реалізуйте хешування паролю
            return password; // Поверніть
хешований пароль
        }
}
```