МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ

КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
спеціальність 122«Комп'ютерні науки»

на тему

«Розроблення інтелектуальної гри-вікторини за допомогою Unity 3d»

Студента групи 601–ТН Вітра В'ячеслава Васильовича

Керівник роботи
кандидат технічних наук,
доцент Деркач Т.М.

Завідувач кафедри
кандидат технічних наук,
доцент Головко Г.В.

Полтава 2021

РЕФЕРАТ

Кваліфікаційна робота магістра: 82 с., 47 рисунків, 1 додаток, 29 джерел.

Об’єкт дослідження:готовий продукт «гравікторина» .

Мета роботи:розробка інтелектуальної гривікторини на платформі Unity, з використанням мови програмування C#. Кінцевий продукт повинен бути конкуренто спроможний на цільовому ринку при мінімальних затратах ресурсів та повністю відповідати тематиці завдання.

Методи:проектування та розробка зовнішнього вигляду, створення користувацького інтерфейсу, написання скриптів для автоматизації процесів, створення внутрішньої бази даних, створення моделі взаємодії з користувачем.

Ключові слова: гравікторина, Unity, скрипт, модель, C#, база даних.

REFERAT

Masters qualification work: 82 pp., 47 drawings, 1 appendices, 29 sources.

Object of research: the finished product "gamequiz".

Purpose: to develop intelectual quiz game product on the Unity platform, using the C # programming language. The final product must be competitive in the target market with minimal resource costs and fully meet the theme of the task.

Methods: design and development of appearance, creation of the user interface, writing of scripts for automation of processes, creation of an internal database, creation of model of interaction with the user.

Tags: quiz game, Unity, script, model, C#, database.

ЗМІСТ

ПЕРЕЛІКУМОВНИХПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

JVM –Java Virtual Machine

HTTP –HyperText Transfer Protocol

SQL –Structured Query Language

GUI –Graphical user interface

ER–діаграма –Entity-relationshipдіаграма

IDE –Integrated development environmen

MVS –Microsoft Visual Studio

JDK –Java Development Kit

HTML –HyperText Markup Language

CSS –Cascading Style Sheets

JSP –JavaServer Pages

GPL –General Public License

GIF –Graphics Interchange Forma

SDK– Software development kit

ПЗ –програмне забезпечення

ОС –операційна система

БД –база даних

ПС –програмне середовище

ПК –Персональний комп’ютер

ВСТУП

На даний час відбуваєтьсястрімке зростання популярності смартфонів. Число користувачів мобільних пристроїв, а також кількість завантажень мобільних додатків зростає з кожним днем. Це зумовлено високою функціональністю, швидкістю та зручністю «кишенькової» техніки, оскільки за допомогою неї можна знайти потрібну інформацію, перебуваючи в будь-якому місці: люди використовують смартфони тоді, коли доступ до комп'ютера відсутній або в ньому немає потреби.

Мобільні програми роблять життя користувачів простіше, адже невеликий пристрій, який завжди знаходиться під рукою, значно скорочує час для виходу в Інтернет.

З розвитком технологій, що лежать в основі смартфонів, індустрія мобільних додатків також постійно розвивається. Розробка мобільних додатків, що здійснюється фахівцями у цій галузі, призначена для конкретної мети.

Одні програми дозволяють підключатися до мережі всюди, другі вказують найближчімаршрути, треті допомагають знайти магазин чи необхідні товари. Є програмне забезпечення, за допомогою якого можна замовити їжу додому. Ці інструменти стали основою для обміну даними та інформацією, що дозволяє заощадити для кожного цінний час та ресурси.

Усі основні додатки діляться між тими, які необхідні тимчасово, і тими, що використовуються виключно з робочою метою.

Перша група включає ігри та інші розважальні програми, програми для відтворення відео і звуку, засоби зв'язку і багато інших.

Друга група включаєвиконання конкретної практичної завдання. Зокрема, деякі утиліти можуть контролювати розвиток бізнес-процесів та подавати аналітичні звіти.

Найбільш поширене створення мобільних додатків першого типу . За останній рік показник завантаження мобільних ігор зріс у рази і ці дані

постійно зростають. Така статистика дозволяє зробити висновок, що розробка мобільних ігор актуальна і доцільна. Головне грамотно оцінити, для кого та навіщо створюється софт, тоді корисна технологія отримає гідне визнання з боку користувачів. Цим і визначається актуальність обраної теми.

Предметом дослідження є розробка мобільних ігор.

Об'єкт дослідження: розробка інтелектуальноїмобільної гри вікторини «QuizGame».

Метою дипломної роботи є створення мобільної гри для Android користувачів за допомогою платформи Unity 3D.

Гіпотезою дослідження є припущення про те, що мобільна гра може бути корисною, цікавою, пізнавальною, якщо відповідатиме таким вимогам:

- Інтуїтивно зрозумілий інтерфейс;
- низькі системні вимоги;
- можливість використовувати без доступу до інтернету;
- цікавий матеріал.

Відповідно до мети роботи, предмету, об'єкту дослідження та гіпотезі наводяться такі завдання:

- вивчення теоретичної частини обраної теми;
- поглиблене вивчення програмних засобів у створенні програми;
- створення мобільної гри вікторини.

Етапи дослідження:

- аналіз та підбір матеріалів;
- порівняння та вибір середовища для створення гри;
- розробка структури гри;
- покроковий процес розробки мобільної гри;
- тестування ігрової програми;
- робота з оформлення дипломної роботи.

Наукова значущість дослідження полягає в тому, що досліджено засоби розробки мобільної гри, вивчено та виявлено їх особливості.

Практична значущість дослідження полягає в тому, що створену гру можна використовувати з метою перевірки своїх загальних знань а також можливістю дізнатися щось нове.

В результаті роботи потрібно розробити мобільний додаток, що задовольняє усім вимогам, удосконалити існуючі знаннята отримати нові знання з розробки програмних продуктів для мобільних систем.

Структура дипломної роботи складається із вступу, трьох розділів, висновків, списку використаної літератури.

РОЗДІЛ 1.
АНАЛІТИЧНИЙ ОГЛЯД

1.1. Опис предметної області

У мультимедійній сфері не останнє значення мають такі явища, як мультимедійні комп’ютерні ігри. Мультимедійні ігри – це ігри, в яких користувачі взаємодіють із віртуальним середовищем, створеним комп’ютером. На даний момент всі комп’ютерні ігри підпадають під цю категорію. У мультимедійні ігри може грати один користувач на локальному комп’ютері або приставці, а також можна грати з іншими гравцями через локальну мережу або глобальну мережу. Перша комп’ютерна гра Star Wars була випущена в 1962 році. Його місія – відбитись від атак астероїдів і ворожих кораблів. Із збільшенням у 1970–1980 роках більш потужних комп‘ютерів електронних ігор також стало більше: пригодницькі, симулятори,головоломки, стратегії та«екшн». Стає дедалі популярнішим такий вид комп‘ютерної гри як онлайн, де персонажами керує не логіка комп‘ютера, а гравці, які за допомогою Інтернету одночасно грають або спостерігають за грою. Таких гравців може бути сотні, тисячі і більше. Левова частка ігор це симуляція різних видів спорту, наприклад, футбол або гольф, також доволі популярні симулятори керування транспортними засобами. Багато з них були схвалені громадськістю, оскільки вони дуже цікаві та корисні для навчання.

Популярність таких ігор пояснюється вмінням спілкуватися з оточуючими. Гравці спілкуються один з одним і відчувають себе частиною глобальної сім'ї. В Україні кількість людей, які купують комп’ютерні ігри з кожним роком зростає. Якщо це просто розвага для гравців, то це дуже прибутковий бізнес для розробників, виробників і дистриб’юторів. Світовий ринок комп’ютерних ігор оцінюється в сотні мільярдів доларів.

Зараз мобільні ігри дуже популярні. Це пов’язано з тим, що останніми роками зріс попит на мобільні пристрої, а це пов’язано з наявністю всіх можливостей, у тому числі доступу до Інтернету, не лише через настільні комп’ютери, а й через планшети, смартфони та звичайні телефони. Не всім для роботи потрібен потужний комп’ютер, але мобільний пристрій, який займає мало місця, завжди носите з собою і може робити все – це інша справа. Природно, продовжують з'являтися нові програми для Android, IOS і Windows, що робить процес експлуатації пристрою більш комфортним і простим [8].

У цій дипломній роботі прикладом є ігрова вікторина, коли гравець за допомогою розуму обирає правильну відповідь та граючи дізнається щось нове і корисне для себе.

1.2. Огляд платформ для створення додатків

1.2.1 Мова програмування Java та IDE Eclipse. Симбіоз Eclipse і популярної мови програмування Java є досить популярною платформою для створення комп’ютерних і мобільних ігор. Java відіграє важливу роль у галузі та є однією з найкращих мов програмування ігор, не зважаючи на те. Він використовує ті ж принципи ООП, що і C++, але надає більш широкий спектр систем для використання. Java-код зазвичай виконується на віртуальній машині Java (JVM) і перетворюється в байт-код загального призначення, який можна виконувати в будь-якій системі. Таким чином, Java є однією з небагатьох мов програмування ігор, яка дозволяє розробникам розробляти ігри для будь-якої системи. Через збільшення кількості сторонніх модулів із відкритим вихідним кодом він також є однією з основних мов, що використовуються для розробки ігор для Android.

Eclipse – це безкоштовне модульне інтегроване середовище розробки програмного забезпечення. Розроблений і підтримуваний Eclipse Foundationвключає в себе такі проекти, як платформа Eclipse, набір

інструментів Java-програміста, система контролю версій (Github) і конструктор графічного інтерфейсу. В основному він написаний на мові Java і може використовуватися для розробки різнопланових додатків на Java, а також використовувати різні плагіни для розробки додатків на інших мовах програмування [10].

1.2.2 Мова програмування C++та IDE Microsoft Visual Studio. Також для написання програм та ігор широко використовується мова програмування C++ та IDE Microsoft Visual Studio, який чудово підігнаний під безліч проектів на цій мові.

Не дивно, що майже в кожному прочитаному посібнику C++ є найкращою мовою програмування для ігор. Піонер сучасних мов ігрового програмування, C++ додає концепцію об'єктно–орієнтованого програмування (ООП) до свого попередника C. Можливість керування дуже системними компонентами низького рівня сприяла дуже швидкому виконанню програм C++, дуже потрібному елементу в ігровому програмуванні.

Більшість ігор високого класу, так чи інакше залежить від кодів C++. Популярні ігрові консолі, такі як Xbox і PlayStation, активно використовують цю мову програмування ігор. C++ це загальновідома мова, якщо передбачається розробка футуристичних ігор.

Microsoft Visual Studio– це набір продуктів від Microsoft, який включає вбудоване сховище проміжного програмного забезпечення та інструментальні функції.

Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом користувача, включаючи Windows Forms, а також веб-сайти, веб-застосунки, веб-сервіси, як на будь-якій окремій платформі з керованим кодом, так і в рідному середовищі.

1.2.3 Мова програмування JavaScript. Не слід обминати увагою і таку мову програмування як «універсальний» JavaScript, ця мова не перебірлива до програмного середовища, а тому дуже зручна для написання програмістом в тому середовищі, яке для нього зручніше та налаштоване конкретно під його навички. Хоча він і не був призначений для розробки масштабних ігор, JavaScript перетворює конвенцію з плином днів. Це одна з найбільш використовуваних мов Інтернету і досить легко інтегрується з будь-якими веб-додатками. По мірі того, як ми все більше і більше просуваємося в напрямку веб-індустрії, онлайн-ігри з кожним днем стають все більш звичними [25].

JavaScript, безумовно, найкраща мова програмування для ігор, коли справа стосується створення інтерактивних онлайнігор. Можливість легко інтегрувати коди JavaScript із звичайними веб-технологіями, такими як HTML та CSS, також швидко сприяє збільшенню кількості мобільних ігор на платформі.

1.2.4 Мова програмування C# та кросплатформаUnity3D. C# одна з найкращих мов програмування відеоігор.Глибоке знання C# є елементарним для кожного ігрового програміста. Багатьом розробникам часто подобається вивчати C# над іншими мовами програмування ігор через високий рівень зручності, який він пропонує. Мова на базі Microsoft підтримує Unity3D, один з найкращих ігрових двигунів, який зараз використовується в промисловості. C# дає розробникам можливість будувати ігри будь-якого типу, для будь-якої архітектури без зайвих клопотів. Мову також набагато простіше вивчити, ніж C++.

Отже, C# одна з найкращих мов програмування для ігор, та повністю підходить для створення інтелектуальної гри вікторини.

Unity – це кросплатформенне середовище розробки комп’ютерних ігор, розроблене американською компанією Unity Technologies. Unity дозволяє

створювати програми, які працюють на більш ніж 25 різних платформах, включаючи персональні комп'ютери, ігрові консолі, мобільні пристрої, Інтернет-додатки тощо. Unity була випущена в 2005 році і з тих пір постійно розвивається.

Основними перевагами Unity є наявність візуального середовища розробки, кросплатформенна підтримка та модульна компонентна система. До недоліків можна віднести складність використання багатокомпонентних схем і складність підключення до зовнішніх бібліотек. На Unity написано візуалізації тисяч ігор, програм і математичних моделей, що охоплюють багато платформ і типів. При цьому Unity використовують великі розробники та незалежні студії [11].

1.2.5 Конструктори додатків. Також на просторах інтернету існують конструктори, так звані сайти чи програми за допомогою яких можна обійтися без власноручного кодування, а лише створивши власний дизайн за шаблоном, який вам пропонується.

Конструктор мобільних додатків це онлайн-редактор на веб-платформі, за допомогою якого користувачі можуть візуально бачити усі функції та самостійно створювати, тестувати, оновлювати і просувати додатки для мобільних пристроїв. У більшості випадків не вимагає для роботи спеціальних знань і навичок програмування, адже режим роботи у таких конструкторах максимально простий та інтуїтивно зрозумілий.

Розробка мобільного додатку на такому конструкторі складається з декількох етапів: створення вкладок, завантаження контенту (текст, фото, відео), визначення вигляду, підключення додаткових функцій (якщо такі потрібні), попередній перегляд, публікація в магазинах додатків (App Store, Google Play, Windows Phone Store, і т.д.).

Крім системи розробки конструктори можуть пропонувати користувачам техпідтримку, хостинг, аналітику, просування готових додатків. Такі сервіси для створення мобільних додатків можна

використовувати для власних потреб, або розробляти програми на замовлення під своїм ім'ям – деякі платформи працюють з користувачами за принципом White label. Ця послуга особливо актуальна для самостійних розробників та невеликих компаній і підприємств з огляду на доступну вартість розробки при високій якості готового продукту.

Конструктори мобільних додатків для малого бізнесу пропонують своїм клієнтам можливість включити в розроблене на замовлення додаток додаткові маркетингові функції: програми лояльності, push-повідомлення, геотаргетинг, отримання та відправлення повідомлень, корзину для товарів, дзвінок в один клік, інтеграція з соцмережами, розміщення інформації про меню і т.д.

1.3. Огляд існуючих програм в сфері вікторин

Вікторина – це гра, яка наголошує на когнітивному розвитку людини, перевіряє його пам'ять, логіку та мислення. Існує безлічрізновидів цієї гри, від настільних та письмових ігор, до цілихтелепередач та, з появою інтернету, онлайн-вікторин.

У сфері ігор-вікторин саме для ПК ця ніша відкрито не популярна та давно забута користувачами, останні оновлення вони отримували щонайменше рік тому, але навіть оновлення уже не повертають юзерів до користування та не мають колишньої зацікавленості, це пов’язано з тим, що зараз розробники комп’ютерних ігор націлені на більш прогресивні та складніші програми, що заохочують користувачів геймплеєм, здатністю взаємодіяти з зовнішнім світом та проходження сюжетних ігор, шутерів, аркад, онлайн ігор з іншими користувачами.

Зробивши запит у вільному для розробників та користувачів онлайн-магазині Steam, відразу видно наскільки ігри даного жанру користуються популярністю.

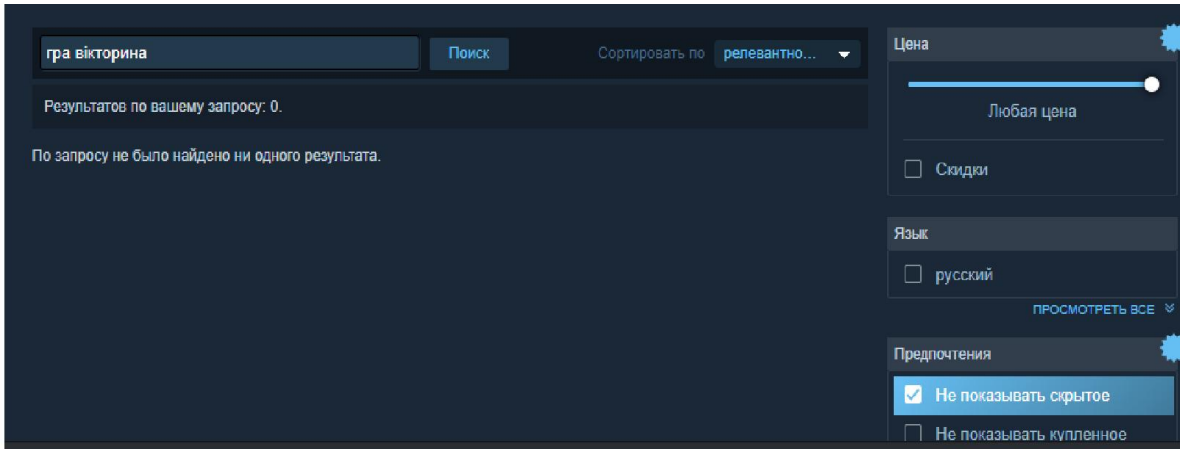

Рисунок 1.1 – Відображення запиту «гра вікторина» українською мовою

Як видно із рис. 1.1 саме українською мовою у магазині немає жодної гри. Зробимо такий самий запит, але уже російською мовою.

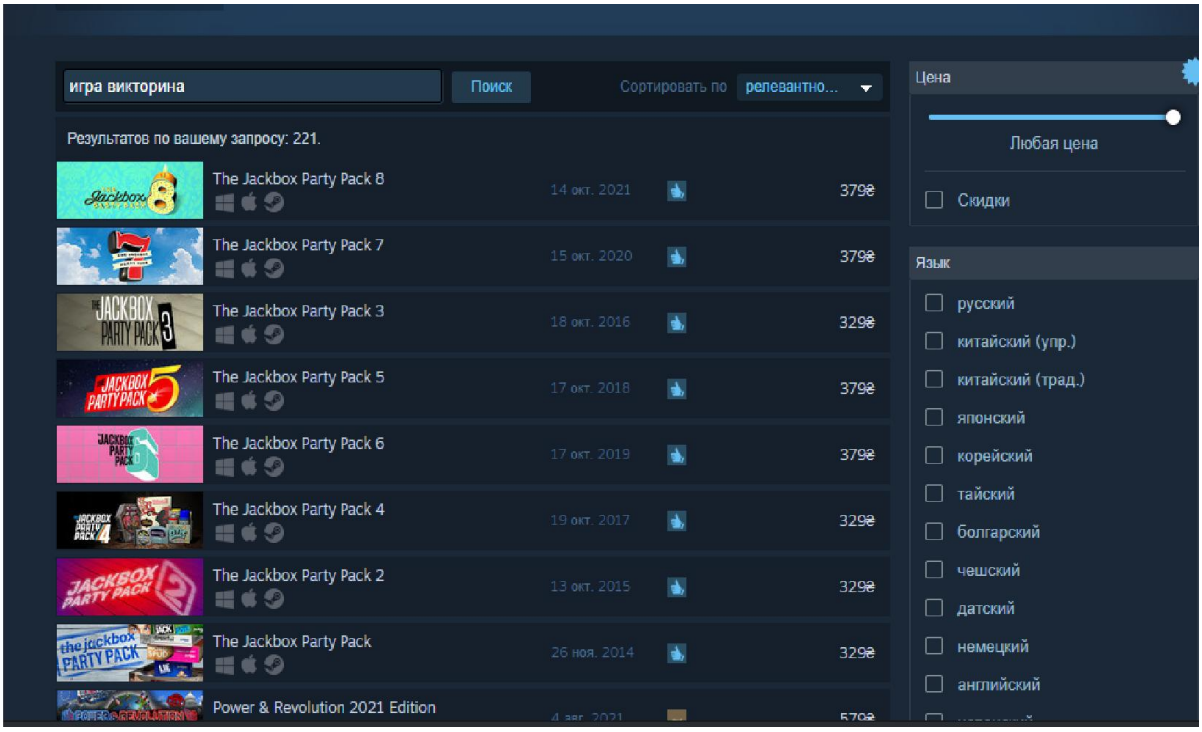

Рисунок 1.2 – Відображення запиту «гра вікторина» російською мовою

Як видно із рис. 1.2 такі ігри є в наявності, але вони вже давно не оновлювалися, і не мають популярності у магазині.

Що стосується ігор–вікторин на мобільних платформах, то тут ситуація набагато краще. Хоч ця тема не є самою популярною, але свою гілку в цьому

великому дереві мобільного геймінгу має. Проте саме українську мову підтримують лише 20% всіх ігор цього жанру. Тепер зробимо запит у онлайн-магазині мобільних додатків «PlayМаркет».

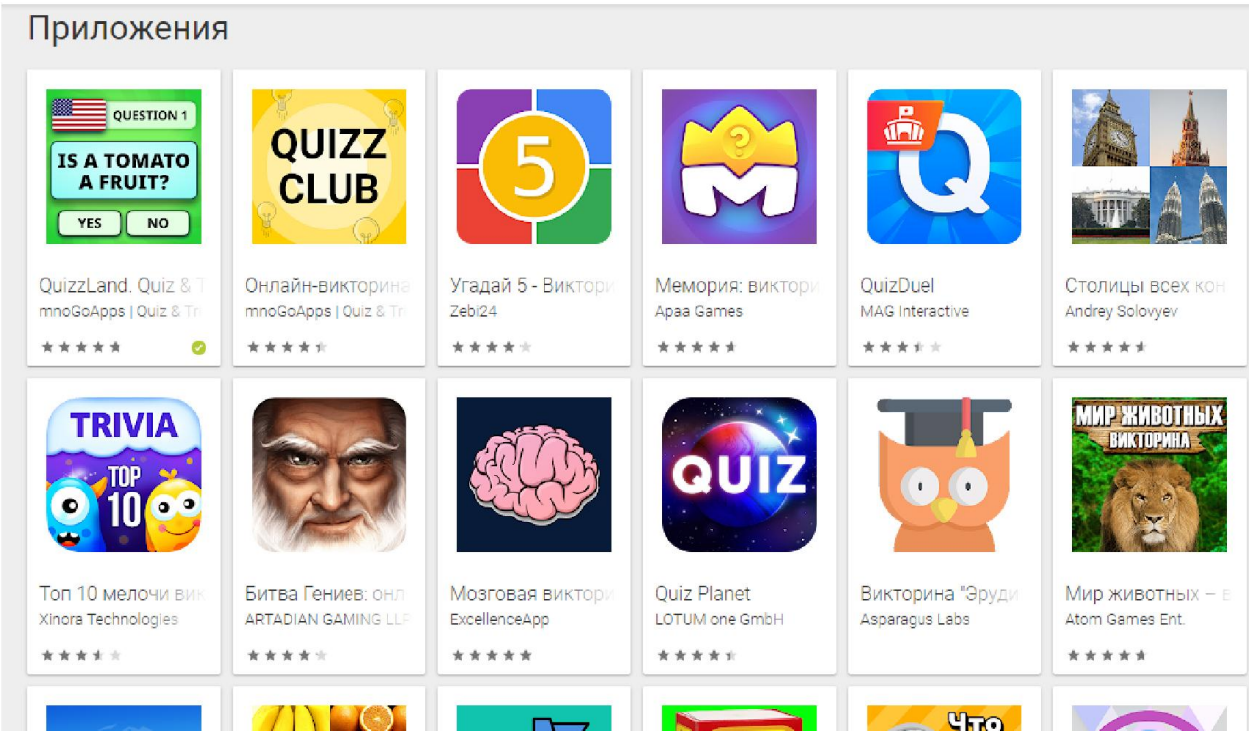

Рисунок 1.3 – Відображення запиту «гра вікторина» у магазині «PlayМаркет»

1.4. Огляд перспективи ринку в сфері вікторин

Отже, перспективи у розробці гри-вікторини на ПК дуже примарні, тому немає ніякого сенсу витрачати ресурси та час, Зараз ПК потребує більш просунутих продуктів, які дозволяють взаємодіяти з іншими користувачами та знаходити щось новеу створеному зовнішньому світі. Звичайно додаткові завдання, які в іграх тепер є типовими рисами більшості гігантів у сфері комп’ютерних ігор, таких як GTA V, багато вбудованих додаткових міні-ігор і дій персонажів, які не пов'язані з основним сюжетом, але доступні для розваги користувачів, які купують такі продукти. Тому планувати розробку для версії для ПК звісно не має сенсу.

Тепер оглядаючи ринок мобільних ігор, можна зробити висновок, що в цій ніші існування нашого додатку має місце бути, тим паче що саме українською мовою таких додатків обмежена кількість.

Мобільний додаток – це спеціально розроблене під функціональні можливості гаджетів програмне забезпечення. Призначення ПЗ може бути найрізноманітнішим: послуги, магазини, розваги, онлайн-помічники та інше. Ці програми завантажуються та встановлюються самим користувачем через мобільні маркетплейси. Найбільші майданчики – AppStore,PlayМаркет. Технічно всі програми створюються під конкретну платформу мобільного гаджета. Найбільш поширені операційні системи – iOS, Android, Windows Phone.

Лідируючу позицію серед операційних систем для телефонів займає Android– операційна система для смартфонів і планшетів, а в останній час для фотоапаратів, телевізорів та інших гаджетів, створена компанією Googlena базі ядраUnix, як безкоштовна операційна система з відкритим вихідним кодом.

Тому проект буде створено саме для пристроїв наAndroid, він буде безкоштовний та розміщений на платформі Play маркет, особливістю цього онлайн-магазину є те, що всі ігри які доступні для завантаження створені під ОС Android. Загальна цінова політика мобільних вікторин в середньому коливається від 40–50грн, але також розробники пропонують багато безкоштовних ігор, оскільки, якщо ти самостійний розробник, а не працівник фірми, потрібно створювати якісні та безкоштовні продукти задля приросту аудиторії та рекламування себе як розробника.

1.5 Функціональні вимоги до проекту

Для того, щоб визначити функціонал який обов'язково повинен бути присутнім у грі потрібно чітко окреслити технічне завдання та функціональні вимоги до проекту.

Технічне завдання – це документ, який повністю описує цілі, завдання та межі проекту, функціональні та нефункціональні вимоги, екрани майбутньої програми та інтеграцію із сервісами.

Технічне завдання розробляється спільними силами аналітика та команди розробників. Такий підхід гарантує якість і повноту документа, що розробляється, завдяки участі компетентних технічних фахівців.

Крім того, у технічному завданні формуються всі умови здачі проекту, що є важливим розділом для замовника. Це означає, що на підставі затвердженого та погодженого технічного завдання здійснюватиметься приймання готового мобільного додатка.

Для того, щоб визначити функціонал, розглянемо функції які будуть потрібні для гри та структуризуємо їх за допомогою таблиць та діаграм.

Таблиця 1.1 – Функції адміністратора

Функція	**Виконавець**
РЕДАГУВАННЯ. Створення та редагування питань і відповідей.	Адміністратор
РЕДАГУВАННЯ. Створення та редагування категорій	Адміністратор
РЕДАГУВАННЯ. Створення та редагування додаткових налаштувань	Адміністратор

Після того як функції адміністратора визначено та занесено до таблиці, визначаємо функції користувача та обираємо дозволи які будуть йому потрібні у користуванні грою.

Таблиця 1.2 – Функції користувача

Функція	Виконавець
ПЕРЕГЛЯД. Перегляд інформації про гру	Користувач
ПЕРЕГЛЯД. Перегляд властивостей гри	Користувач
ПЕРЕГЛЯД. Перегляд налаштувань гри	Користувач
РЕДАГУВАННЯ. Увімкнення та вимкнення звуку та музики у грі	Користувач
РЕДАГУВАННЯ. Вибір відповіді на представлене грою запитання.	Користувач

Гра-вікторина буде складатися із трьох частин (в Unity сцени), на яких і буде відображено весь процес гри та взаємодії з користувачем. Перша частина це головне меню гри, тут дуже просте відображення, це кнопка грати і назва гри:

- грати – при натисненні на нього виконується перехід до іншої сцени де запропоновано вибір категорії;

- назва гри – не клікабельний елемент, який просто відображає заданий розробником текст;

Друга частина теж складається з декількох елементів, це категорії з питаннями та дві стрілки для гортання категорій:

- назва категорії – при натисненні на назву конкретної категорії виконується перехід до іншої сцени та починається процес гри;

- стрілки вгорі та знизу для можливості гортати категорії;

Третя сцена це безпосередньо сам процес гри, де міститься питання та яке потрібно обрати відповідь, три варіанти відповіді і таймер часу:

- питання – не клікабельний елемент, який відображає запитання, на яке користувачеві потрібно обрати правильну відповідь;

- правильна відповідь –при натисненні на нього, з’явиться відображення тексту, що ваша відповідь вірна і запропоновано перехід до наступного запитання;

- неправильна відповідь – неправильних відповідей буде дві, при натисненні на якусь із них з’явиться відображення тексту, що ваша відповідь невірна здійснено перехід до головного меню;

- таймер – не клікабельний елемент, який вказує скільки часу залишилося на відповідь, по закінченню таймера виводиться повідомлення, що час сплив та відбувається перехід до головного меню.

Після того як було розглянуто технічне завдання та визначено відповідний функціонал для даного проекту, потрібно візуалізувати дії користувача та адміністратора (розробника). На цьому етапі потрібно побудувати Use-caseдіаграми.

Use-case – діаграма, що описує, який функціонал програмної системи, що розробляється, доступний кожній групі користувачів.

Основні елементи діаграми - учасник (actor) та прецедент (варіант).

Учасник (actor) – це безліч логічно пов'язаних ролей, що виконуються при взаємодії з прецедентами чи сутностями (система, підсистема чи клас). Учасником може бути людина або інша система, підсистема або клас, які є чимось поза сутністю. Графічно учасник зображується "чоловічком".

Прецедент (use case) – опис множин послідовних подій (включаючи варіанти), що виконуються системою, які призводять до результату, що спостерігається учасником. Прецедент представляє поведінку сутності, описуючи взаємодію між учасниками та системою. Прецедент не показує, як досягається деякий результат, а тільки що саме виконується. Прецеденти позначаються дуже простим чином – у вигляді еліпса, усередині якого вказано його назву.

На діаграмі варіантів використання можна відобразити наступніелементи :

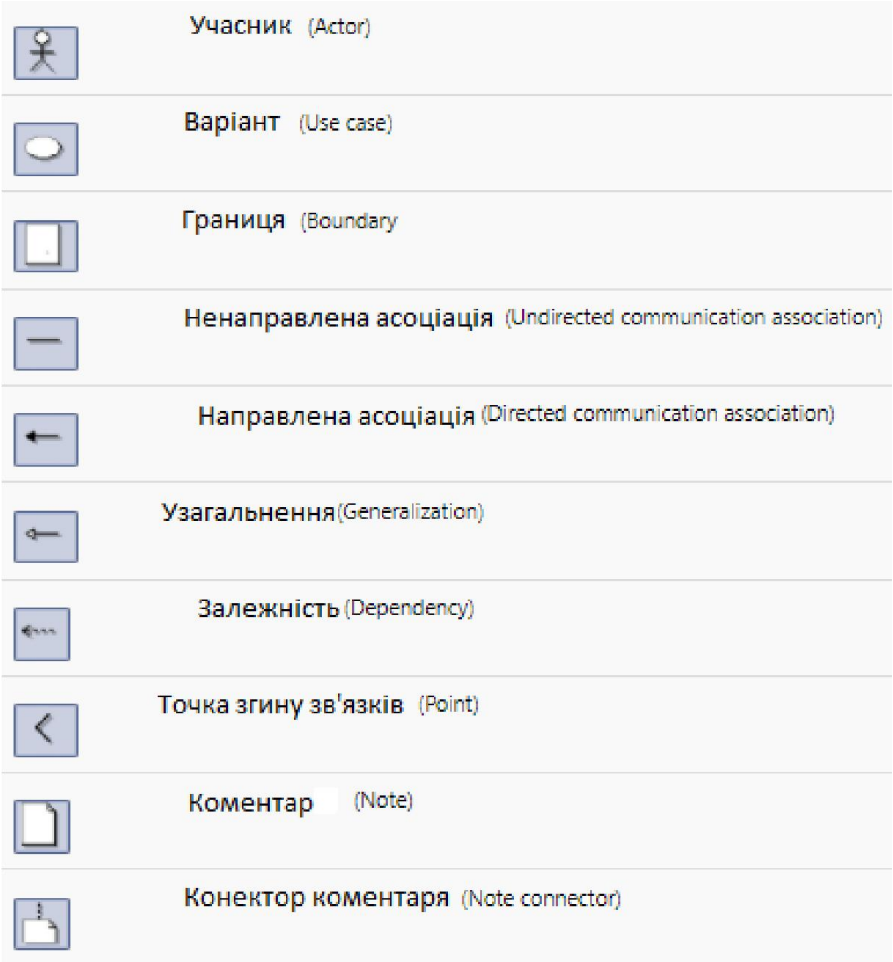

Рисунок 1.4 – Основні елементи діаграми варіантів використання

Також для розробки проекту, потрібно створити на наповнити базу даних, вона буде реалізована в Unity, оскільки це буде зручніше у подальшому її використанні, без зовнішніх підключень БД.

База даних – сукупність даних, організованих згідно з концепцією, яка описує характеристику цих даних і взаємозв'язки між їх елементами. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти. Дані у базі організовують відповідно до моделі

організації даних. Отже , сучасна база даних, крім зазначених даних, містить їх опис а також може містити засоби для їх обробки.

Існують різні типи БД:

- БД як плоскі таблиці, у яких вся взаємозалежна інформація зберігається у одній. Такі таблиці можуть бути створені за допомогою програми Microsoft Excel, редактора Microsoft Word або іншого текстового редактора. БД цього типу часто містять надлишкову інформацію, що повторюється, що робить їх малоефективними при роботі з великими обсягами даних.

- Реляційні БД складаються з декількох різних таблиць, кожна з яких містить інформацію про однотипні об'єкти, процеси, події. Рядки таблиці називаються записами, а стовпці – полями. Кожен запис містить дані про один об'єкт (процес, подію). Кожне поле визначає одну характеристику об'єкта (процесу, події) і має певний тип даних (текст, число, дата). Між таблицями встановлюються зв'язки з полів, що збігаються.

Використання пов'язаних таблиць дозволяє уникнути непотрібного дублювання даних, що зменшує обсяг пам'яті, необхідний їх зберігання, а також збільшує швидкість і точність обробки інформації. Для зберігання даних типу текст необхідна пам'ять у байтах, рівна кількості символів, що становлять цей текст, в той час як для зберігання числових даних потрібно лише два або чотири байти.

Також слід розуміти як саме БД буде реалізована у проекті, та розглянути можливі варіанти.

Системи управління базами даних (СУБД) - це програма, з допомогою якої реалізується централізоване управління даними, які у базі, доступом до них, та підтримкою в актуальному стані.

СУБД можна класифікувати за способом встановлення зв'язків між даними, характеру виконуваних ними функцій, сфері застосування, кількості

підтримуваних моделей даних, характеру мови спілкування з базою даних та іншим параметрам.

Класифікація СУБД:

- за функціями СУБД, що виконуються, поділяються на операційні та інформаційні;

- у сфері застосування СУБД поділяються на універсальні та проблемно-орієнтовані;

- за мовою спілкування СУБД, що використовується, поділяється на закриту, незалежну мову спілкування між користувачем і базою даних, і відкриту, в яких для спілкування з БД використовується мова програмування, розширена операторами мови маніпулювання даними;

- за кількістю підтримуваних рівнів моделей даних СУБД поділяються на одно-, дво-, трирівневі системи;

- за способом встановлення зв'язків між даними розрізняють реляційні, ієрархічні та мережеві БД;

- за способом організації зберігання даних та виконання функцій обробки БД поділяються на централізовані та розподілені.

Найбільшого поширення набули СУБД Microsoft Access, MySQL, PostgreSQL, SQLServer, Oracle, Sybase, Interbase, Firebird.

Етапами роботи в СУБД є:

- Створення структури БД, тобто визначення переліку полів, з яких складається кожен запис таблиці, типів та розмірів полів (числовий, текстовий, логічний тощо), визначення ключових полів для забезпечення необхідних зв'язків між даними та таблицями;

- введення та редагування даних у таблицях БД за допомогою стандартної форми, що представляється за замовчуванням, у вигляді таблиці та за допомогою екранних форм, спеціально створюваних користувачем;

- обробка даних, що містяться в таблицях, на основі запитів та на основі програми;

- виведення інформації з ЕОМ з використанням звітів та без використання звітів.

Уся інформація, що міститься всередині БД, це різні об’єкти, кожен з яких строго або не строго типізований.

До складу бази даних можуть входити такі об'єкти як: таблиці, форми, звіти, запити, макроси, модулі, тощо. Таблиці це один із основних та обов'язкових об'єктів БД. Поля таблиці можуть містити такі типи даних:

- Текстовий, це тип даних за замовчуванням. Текстове поле може містити символьні або числові дані, які не потребують обчислень;
- МЕМО, призначене для введення тесту обсягом трохи більше 65535 символів;
- Числовий, призначений для зберігання чисел, які у розрахунках. Має багато підтипів за розмірами, від яких залежить точність обчислень;
- Дата/Час, зберігаються у спеціальному фіксованому числовому форматі;
- Грошовий, призначено для зберігання даних про грошові одиниці. Лічильник містить ціле число, яке автоматично збільшується на одиницю для кожного нового запису;
- Логічний, може містити лише два значення: 1/0;
- Гіперпосилання, дозволяє вставляти у полі гіперпосилання;
- Майстер підстановок, будує для поля список значень з полів з іншої таблиці.

Запити дозволяють переглянути та редагувати, аналіз та дані що змінилися в одній чи кількох таблицях. Загалом є кілька видів запитів.

Запит на вибірку – дозволяє вибрати відповідні критерії запису з взаємопов'язаних таблиць, зробити сортування цих записів, виконати обчислення над якоюсь групою записів. Результатом виконання запиту вибірку є таблиці, які існують до закриття запиту. За підсумками запиту на вибірку будуються запити інших видів.

Запит створення таблиць – використовує як базового запит на вибірку, але результат запиту зберігається у новій таблиці і після закриття запиту.

Запити на оновлення, додавання та видалення записів – призначені для зміни даних у таблицях.

Наприклад БД Access має потужні інструментальні засоби, що дозволяють створювати складні запити, не вдаючись до програмування. До таких засобів відносяться Майстер запитів та Конструктор запитів. Найпростіше створити новий запит за допомогою Майстра запитів, він розбиває весь процес створення запиту на ряд елементарних кроків, на кожному з яких користувачеві пропонується вибрати один з можливих варіантів. За допомогою Конструктора запитів можна не лише створювати нові запити, а й змінювати існуючі.

Форми. формами є діалогові вікна що містять елементи управління – текстові поля, кнопки, прапорці, перемикачі, списки, написи, рамки. Кожен із цих елементів має набір властивостей, змінюючи які можна як змінювати зовнішній вигляд форми (розташування, колір, розмір та інших.), так і визначати їхню реакцію на якусь подію. Використання форм дозволяє:

- автоматизувати введення даних;
- керувати ходом виконання програми;
- встановлювати різні права доступу для різних користувачів під час роботи з однією і тією ж БД;
- виконувати первинну перевірку даних, що вводяться.

Джерелом даних для форм є таблиці та запити. Зазвичай на формі відображається один запис таблиці-джерела (режим форми), хоча можна вивести на форму і відразу всі записи (режим таблиці).

Маючи на руках прототип та технічне завдання, яке повністю описує структуру програми, опис екранів та інтеграції, можна приступати до етапу розробки дизайну та кодування мобільного додатка.

Висновки до розділу 1

Отже, у цьому розділі виконано аналітичний аналіз, розглянуто предметну область проекту, розглянуто можливі платформи та мови програмування для подальшої реалізації проекту, проаналізовано уже існуючі додатки в сфері ігор вікторин, як для ПК, так і для смартфонів, обрано подальший шлях створення та просування додатку саме в мобільному геймінгу, конкретно на ОС Android. Описано майбутній функціонал додатку для розробника і користувача та відповідно створено Use-case діаграми.

Оскільки ігрова індустрія з кожним днем все більше і більше розвивається, а галузь вимагає експертних розробників з глибоким досвідом у сфері програмування ігор, то почати свою кар’єру із виконання тематичного дипломного проекту, це те що зараз потрібно розробникупочатківцю. Цей проект чудово підходить, як з точки зорудосвіду та перевірки уже наявних знань, так і для майбутнього резюме.

Також розробка гривікторини, це можливість внести щось нове до зацікавлення користувачами ігор та додатків, у яких треба думати та розвиватися, бо в нинішній час кожна друга гра це звичайний «клікер» або «стрілялка». Плюсом буде те, що конкуренція на цьому ринку не занадто висока, і якщо виділятися і бути унікальним, є шанс бути поміченим.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ ТА ВИБІР ПЗ

2.1 Вибір платформи

Для створення даного проектупотрібно мати ПЗ із можливістю створення та редагування графічного інтерфейсу користувача, зручний компілятор для написання та перевірки коду який буде легко взаємодіяти із ПЗ та інтегруватися в гру.

В наш час існує безліч як платних, так ібезкоштовних інструментів розробки ігор, починаючи від простих бібліотек для відомих мов програмування, закінчуючи великими редакторами з великим функціоналом.

Для створення якісної гри необхідно використати якісні інструменти. Тому було прийнято рішення використати готовий інструмент для розробки ігор та додатків.Оскільки середовища для розробки додатків стали дуже популярними, кількість їх на ринку прибуває з кожним днем. Виконаємо розбірнайпопулярніших ігрових платформ, які встигли зарекомендуватисебе на ринку та підходять під поставлені завдання.

Unity3d – це середовище розробки додатків та ігор, що працюєіз операційними системами Windows та MacOS. Unity дозволяє розробляти як двовимірні, так і тривимірні програми та ігри. Середовище дозволяє компілювати проект під безліч пристроїв. У цей список потрапляють пристрої з ОС для ПК та MAC: Windows,MacOS, Linux, з ОСсмартфонів: Android, iOS, Windows Phone, і навіть під ігрові консолі:PlayStation, Wii, Xbox. Крім іншого, проект може бути запущений іу браузері, але тільки за допомогою спеціального плагіна від компанії розробників Unity – UWP (unity web player).

Середовище має поширений серед редакторів Drag&Dropінтерфейс, який складається з різнихвікон, які, у свою чергу, дуже просто налаштовуються по висоті,ширини та розмірів. Ігровий двигун Unity3D

повністю інтегрований всередовище розробки Unity.Тісна інтеграція дозволяє отримати всі операції, які гра може виконувати в редакторі. Unity також надає API для доступу до інструментів введення та налаштування Android та iOS.

З переваг можна виділити:

- мультиплатформенність;
- високафункціональність;
- велике ком'юніті;
- зручність у роботі та компіляції;
- велика кількість бібліотек, що розширюють функціонал.

Недоліки:

- закритий вихідний код;
- умовно-безкоштовний.

Unity підтримує дві сценарні мови, це C# (модифікований) та JavaScript, що дозволяє широкому колу програмістів користуватися платформою.

UnrealEngine – це інструмент розробки ігор від американської компанії EpicGames, написаний мовою C++, і запускається усіма популярними ОС для ПК і Mac: Windows, Linux, MacOS.

Мультиплатформенність середовища дозволяє створювати ігри для більшості операційних систем та платформ, до цього списку потрапляють: Windows, Mac, Xbox, GameCube, Wii, PlayStation, iOS, Android та NintendoSwitch.

UnrealEngine має низку спрощених методів портування, використовуючи модульну систему залежних компонентів. Підтримує лише одну мову програмування, C++, і має низку переваг та недоліків.

Переваги:

- велике ком'юніті;
- підтримка графічного реактора для створення алгоритмів;
- висока функціональність;

- мультиплатформенність.

Недоліки:

- умовно-безкоштовний;
- високі системні вимоги;
- непростий у освоєнні.

Cocos2d – це крос-платформний фреймворк, що дозволяє створювати ігри та інтерактивні програми. Найчастіше його використовують длястворення саме мобільних ігор. Сам фреймворк має безліч відгалужень, таких як Cocos2d-html, Cocos2d-ObjC, Cocos2d-x та Cocos2dXNA.

Охоплення платформ даного інструменту вражає, він дозволяє компілювати додатки під iOS, Android, WindowsPhone, Windows, Linux,MacOSтакож браузери з підтримкою HTML 5, Xbox360 та інші платформи, що підтримують Python.

Під кожну конкретну платформу може знадобитися конкретне відгалуження даного фреймворку. Основною відмінністю використання різних відгалужень є мови програмування.Так Cocos2d охоплює: C++, Lua, Javascript, Java, Objective-C, C#, Python.

Переваги:

- відкритий код продукту;
- великий вибір платформ;
- зручність у роботі.

Недоліки:

- невелике ком'юніті;
- рідкісні оновлення;
- вузький вибір мов для певних платформ;
- невелика кількість навчальних матеріалів.

Були проаналізовані три найбільш популярні та підходящірозробки конкретно цієї гри середовища розробки.Безумовно всі перераховані вище платформи хороші і підходятьдля розробки кросплатформенного додатка, але тільки Unity підтримує мову програмування C# з можливістю компіляції під

Android з мінімальними часовими затратами, що є привабливим через просте використання як у розробці, так і вкомпіляції під обрану платформу, тому мною ухвалено рішеннявикористовувати середовище розробки Unity для розробкимобільної гри.

Мовою програмування скриптів для об'єктів було обрано C#.C# – об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET.

Перша перевага, на мій погляд, – сама мова C#. Ця високорівнева мова дозволяє програмісту легко увійти в розробку гри. Цеважливий момент, тому що на відміну від інших двигунів, де використовується мова C++, у C# є багато елементів і прийомів, які вже реалізовані, і програмісту потрібно лише скористатися ними.

Синтаксис C#близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато що від своїх попередників – мов C++, ObjectPascal, Smalltalk – C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад множинне спадкування класів[29].

В таблиці 2.1 представлені платформи та їх порівняння.

Таблиця 2.1 Огляд платформ для створення ігор

Платформа	Мова програмування	Ціна	Мультиплатформенність	Відкритий код	Зручність використання	Масштаб проектів що створюються
Unity	C#, Javascript	Free, 1500$ pro	+	-	5/5	Великі, середні, малі
UE4	C++	Free, 3000$ pro	+	-	4/5	Великі, Середні
Cocos 2	C++, Javascript,	Free,	+	+	3/5	Великі, середні, малі

D-x	Lua					

Після установки даного ПЗ, потрібно його легко сприймати, для цього тут передбачений інтуїтивно зрозумілий інтерфейс та додаткові пояснювані матеріали.

Під поняттям «інтерфейс» прийнято розуміти набір інструментів, що використовуються при взаємодії двох систем.

У перекладі з англійської слово interface буквально означає місце дотику, а під системами, між якими здійснюється така взаємодія, можуть матися на увазі різні об'єкти. Наприклад, це може бути взаємодія між обладнанням і людиною, різними видами обладнання, але найчастіше під інтерфейсом мають на увазі систему взаємодії програми з людиною для обміну даними та отримання потрібної інформації.

Останній варіант інтерфейсу часто називають користувальницьким. По виду взаємодії з обчислювальною технікою його поділяють такі види:

- Інтерфейс у вигляді командного рядка. Один з перших типівінтерфейсів, який відрізняється високою трудомісткістю і використовується для управління комп'ютерами та системами все рідше. Тим не менш, через свою просту і надійність міцно займає свою нішу в професійному середовищі. Передбачає введення команд підтримуваною машинною мовою в командний рядок із клавіатури.

- Графічний інтерфейс – найпоширеніший на сьогоднішній день спосіб взаємодії користувача з операційними системами і прикладним програмним забезпеченням. Відрізняється інтуїтивністю, простотою сприйняття та введення інформації. Для взаємодії з таким інтерфейсом часто достатньо звичайної комп'ютерної миші.

- Жестовий інтерфейс можна назвати продовженням розвитку графічного інтерфейсу. У ньому керування елементами взаємодії з

операційною системою та програмами використовуються сенсорні екрани, стилуси, графічні планшети та інші передові технології.

На рис. 2.1 зображено інтерфейс платформи Unity 3d:

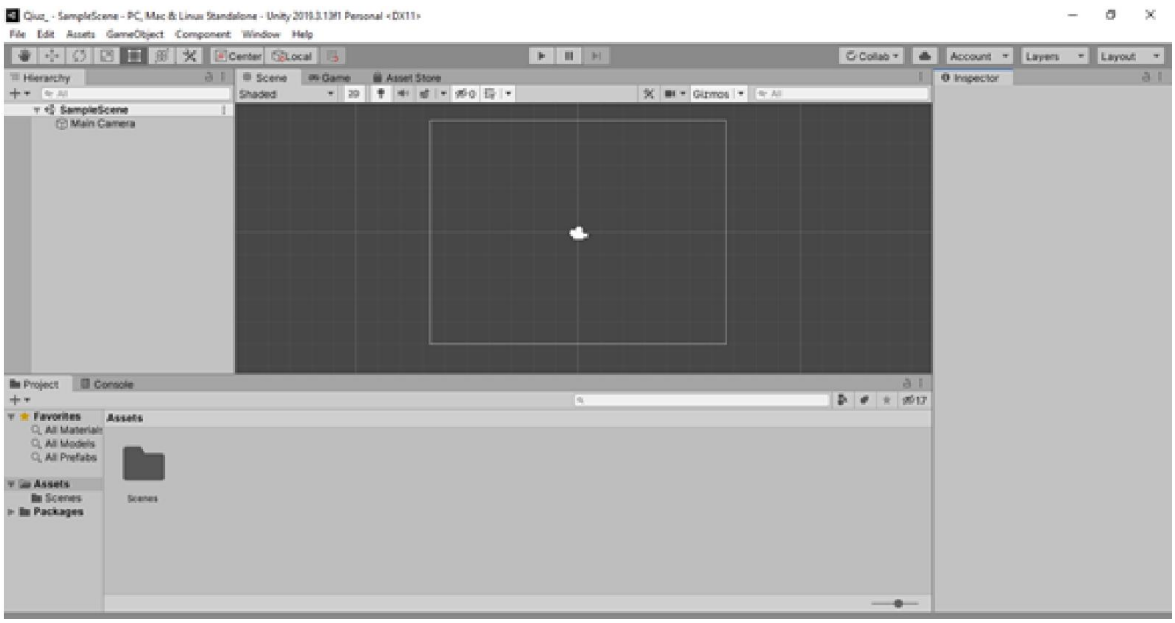

Рисунок 2.1 – Інтерфейс платформи Unity

На рис. 2.2 зображено інтерфейс IDEMicrosoftVisualStudio, програмне середовище де буде проводитися написання скриптів.

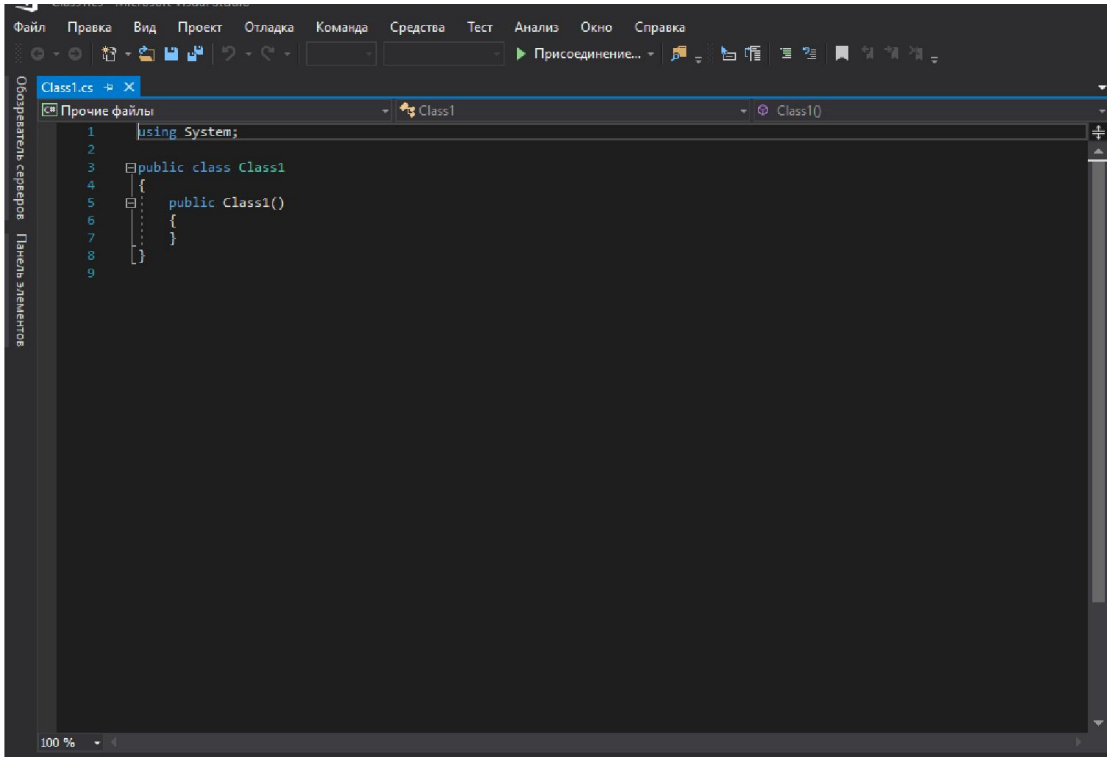

Рисунок 2.2– Інтерфейс IDE Microsoft Visual Studio

Після того, як було обрано платформу для розробки Unityта мовою програмування для написання скриптів C#, потрібно переходити до проектування та відповідно розробки самого проекту.

2.2 Проектування дизайну

Ми всі знаємо, що красивий дизайн продає. Це вірно і для створення мобільних додатків. Звісноважливо, щоб користувачі вибрали програму серед сотень інших, потрібно якнайшвидше закохати їх у свій продукт. Як показують дослідження, користувачеві необхідно всього 7 секунд, щоб вирішити, подобається йому додаток чи ні.

Щоб створити дизайн програми, потрібно тримати в голові потреби користувачів і робити усе, щоб допомогти користувачам “дійти до кінця”. Використання програми не повинно бути схожим на IQ тест.

Користувальницький інтерфейс це всі елементи, які бачить користувач, та з якими взаємодіє. Наприклад:

- кнопки;
- текстові поля;
- чекбокси;
- слайдери;
- рядки пошуку;
- теги;
- іконки.

Дизайн додатку сфокусований на тому, щоб зробити взаємодію користувача з продуктом приємним та максимально зрозумілим. Для цього на початковому етапі використовують варфрейми.

Варфрейм це образ дизайну низької точності. У ньому повинно чітко показуватися:

- Основну групу контенту (Що?)
- Структуру інформації (Де?)
- Опис та базову візуалізацію взаємодії між інтерфейсом та користувачем (Як?)

Ці «каркаси» не є безглуздим набором блоків, хоча він виглядає приблизно так. Розглядати його потрібно як скелет дизайну та створювати таким чином, щоб вони зображали кожну деталь фінального продукту.

Вони показують, як усі екрани будуть з'єднані та які елементи вони відображатимуть: від кнопок та спливаючих вікон до візуалізації та тексту. Але поки що без контенту фотографії, відео, кольори та шрифти будуть додані пізніше, адже на цьому етапі вони відповідають за демонстрацію логіки [20].

«Зображення» – це ключовий термін, який допоможе знайти необхідну точність, зручний темп. Звісно тут немає можливості показати занадто багато деталей, але, з іншого боку, створюється точний образ фінального дизайн-продукту, який не втрачає жодної його важливої деталі.

Візуалізація має бути побудована за правилами естетики, але дуже спрощена. Чорний, сірий та білий – це типові кольори, які знадобляться, за бажанням можна додати більш яскраві кольори, та вони повинні бути однотонні.Для представлення «каркасів» у грі вікторини я використовував чорні рамки та текст, кнопки виділив синіми кольорами, ігрове поле також окреслив голубою зоною для простішого сприйняття, коли відповідь правильна, користувач побачить зелену «галочку», що і представлено у варфреймах, і відповідно червоний хрестик, коли відповідь невірна, або час на виділене питання вичерпано – це теж вважається програшом.

На рисунку 2.3 зображено варфрейм головної сторінки гривікторини:

Рис 2.3 – Варфрейм головного меню

Головне меню має лише кнопку «Грати» і власне саму назву нашою гри, що значно спрощує дії користувача. При аналізі подібних ігор, було виявлено що цього буде цілком достатньо, при публікуванні гри та перегляду відгуків, уже працюючих проектів у даній сфері, зрозуміло що даний тип меню максимально простий та мінімалістичний для легкого сприйняття різними віковими категоріями користувачів.

На рис. 2.4 зображено варфрейм, де відображається список категорій можливих для гри.

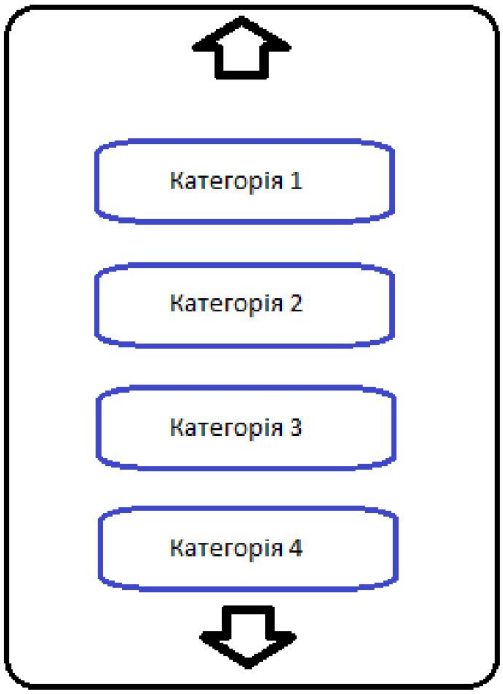

Рис 2.4 – Варфрейм категорій гри

На другій сцені гри відбувається вибір категорій користувачем, тут відповідно зображено список категорій доступних для гри, та стрілки вгорі і внизу для зручності гортання.

На третій сцені в якій відбувається сам процес гри, зображено питання на яке користувач повинен дати відповідь, та декілька варіантів відповіді, із який правильний лише один, також внизу екрана розташовано таймер, що зображено на рис. 2.5

Рисунок 2.5 – Варфрейм процесу гри

При натисненні правильної відповіді з’являється сторінка, яка повідомляє гравця про успіх та відбувається перехід до наступного запитання, що зображено на рис. 2.6.

Рисунок 2.6 – Варфреймдемонстрації правильної відповіді

При натисненні користувача на неправильну відповідь з’являється сторінка, яка повідомляє гравця про невдачу, та відбувається перехід до

списку категорій для того щоб користувач міг обрати іншу категорію, або спочатку почати відповідати на запитання тієї категорії де була допущена помилка, що зображено на рис. 2.7.

Рисунок 2.7 – Варфреймдемонстрації неправильної відповіді

Також при закінченні часу на таймері відбувається сповіщення користувача про те що час який даний на питання вичерпано і відбувається перехід до списку категорій.

Рисунок 2.8 – Варфрейм демонстрації варіанту, коли час вийшов

На загальному макеті проекту зображено варфрейми усіх сцен та можливих варіантів подій, також помічено вказівниками (стрілками) яку сцену бачить користувач.

- перша сцена при запуску гри це назва гри та кнопка грати;
- друга, це вибір категорії які розбиті на тематичні питання;
- третя, де безпосередньо розпочинається сам процес гри.

Також після вибору відповіді, можливі два варіанта подій, які також описані, це правильна відповідь і перехід до наступного питання, та неправильна і повернення до вибору категорії. Можливий ще один варіант, якщо вийшов час – теж повернення до вибору категорій.

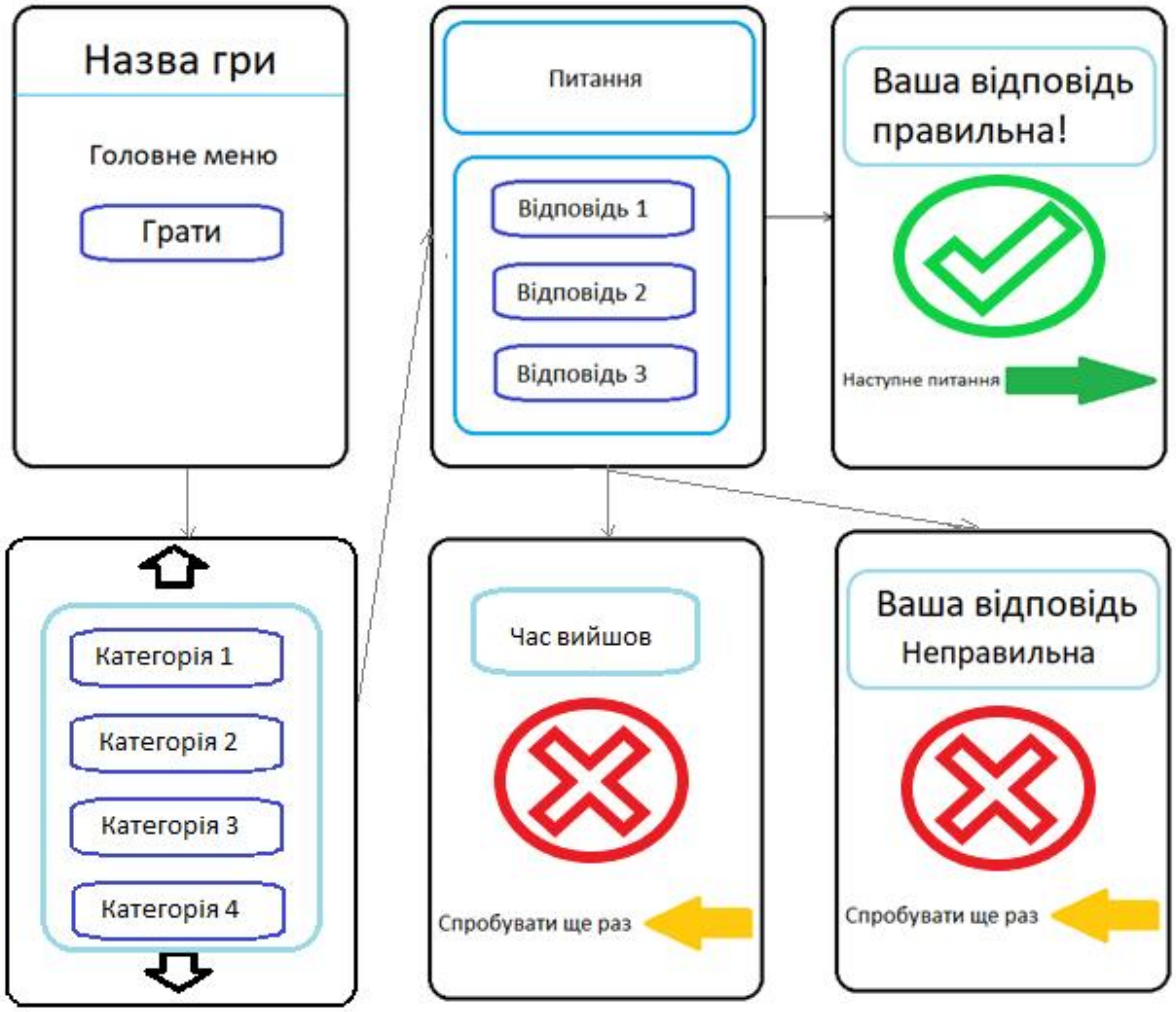

Рис 2.9 – Загальний макет проекту

Висновки до розділу 2

Були проаналізовані три найбільш популярні та підходящі розробки конкретно цієї гри середовища розробки, проте тільки Unity підтримує мову програмування C# з можливістю компіляції під Android з мінімальними часовими затратами.

Вагомим плюсом є те, щовона має весь необхідний функціонал і особливості системи, такі як створення окремого редактора для функцій користувача у своєму внутрішньому інтерфейсі, можливість редагування гри за допомогою візуальних сцен, а не лише скриптингу, можливість конвертування з підтримкою багатьох ОС . Тому мною ухвалено рішення використовувати середовище розробки Unity для розробки мобільної гри.

Для написання ігрових скриптів було обрано мову програмування C#, ця мова має усі необхідні інструменти і чудово взаємодіє разом із Unity, має зрозумілі методи і класи та не випадково утримує лідерство серед розробників саме для Android. У C# є багато елементів і прийомів, які вже реалізовані, і програмісту потрібно лише скористатися ними.

Скрипти на мові програмування C# будуть написані та реалізовані у MicrosoftVisualStudio.

Дизайн було спроектовано інтуїтивно зрозумілим та максимально простим для зручності гравця, у створену внутрішню базу даних буде додано питання різної складності і тематики та поділено їх на категорії.

Було створено та проілюстровано варфрейми усіх сцен гри та детально описано їх використання, також було описано зв’язки і переходи між сценами та можливі варіанти розвитку взаємодії з користувачем.

РОЗДІЛ 3.
РОЗРОБКА ФІНАЛЬНОГО ПРОДУКТУ

3.1 Файлова структура проекту

Жоден сучасний електронний накопичувач не обходиться без файлової системи. Це свого роду операційна система жорсткого диска, що дозволяє йому управляти потоками даних та зберігати їх у необхідних форматах. Багато користувачів також називають її файловою структурою. Необхідно роз’яснити, що таке файлова система та файлова структура, у чому їх відмінності.

Файлова система– це набір встановлених програм, передбачений на будь-якому сучасному носії інформації. У ній може зберігатися велика кількість елементів, але це не всі її функції, оскільки вона визначає спосіб їх зберігання.

Файлова структура – це те саме, що й файлова система, але з урахуванням того що в першій всі описи виконані логічно, а в другій – фізично, а саме з точки зору апаратних засобів та програмування. Це функціональна частина операційної системи, яка відповідає за те, щоб файли, папки та будь-які інші дані зберігалися надійно та у певній послідовності.

Видів логічних структур є кілька. Призначені для різних архітектур та різної складності. Серед них:

Однорівнева (лінійна) – це звичайна послідовність імен файлів. Часто вона використовується для невеликих за обсягом структур, на яких може розташовуватись дуже мала кількість елементів. Доступ до інформації на таких носіях здійснюється через просте звернення на ім'я, оскільки жодних вкладених папок, де могли знаходитися дані немає.

Багаторівнева (ієрархічна) – багаторівнева послідовність папок та файлів. Використовується на різноманітних сховищах, де зберігаються десятки, сотні або тисячі елементів. Вона має деревоподібну побудову.

Початковий або кореневий каталог містить вкладені каталоги першого рівня, які можуть містити папки другого рівня і так далі. У каталозі будь-якого рівня передбачено збереження файлів.

У розробленому проекті міститься ряд каталогів, які зберігаютьв собі:

- скрипти;
- сцени;
- зображення інтерфейсу користувача та іконки;
- зображення для запитань;
- анімації;
- шрифти.

У директорії Scripts містяться скрипти з описом усіх класівта взаємодій, також тут розташовано підпапку Editor, в якій лежить скрипт, за допомогою якого було створено розширений інтерфейс в середині Unity.

Цей інтерфейс можна відкрити та додати свої зміни, знайшовши у головній директорії QuizGame, параметр Config.

У папці Prefabs містяться шаблони кнопок відповідей та область питання, які використовуються у грі.

У папці Animationsвідповідно розташовано анімації гри.

ПапкиFonts та Presetsмістять в собі шрифти і кольорову палітру, які використовуються у грі.

У директорії Sprites містяться елементи всьогоінтерфейсу користувачата містяться картинки до питання, на які належить відповісти користувачеві.

Файлова структура зображена на рис. 3.1

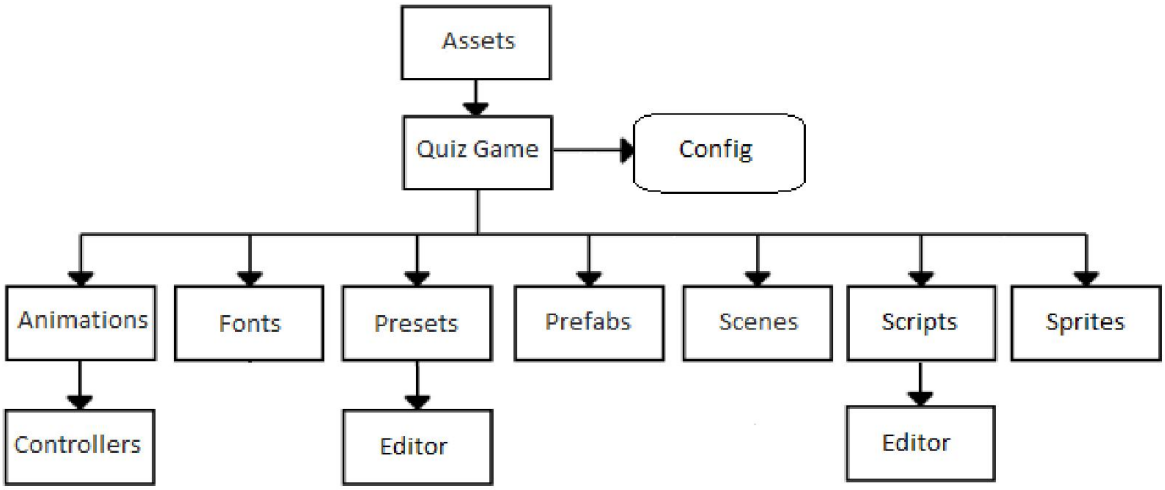

Рисунок 3.1 – Файлова структура гри вікторини

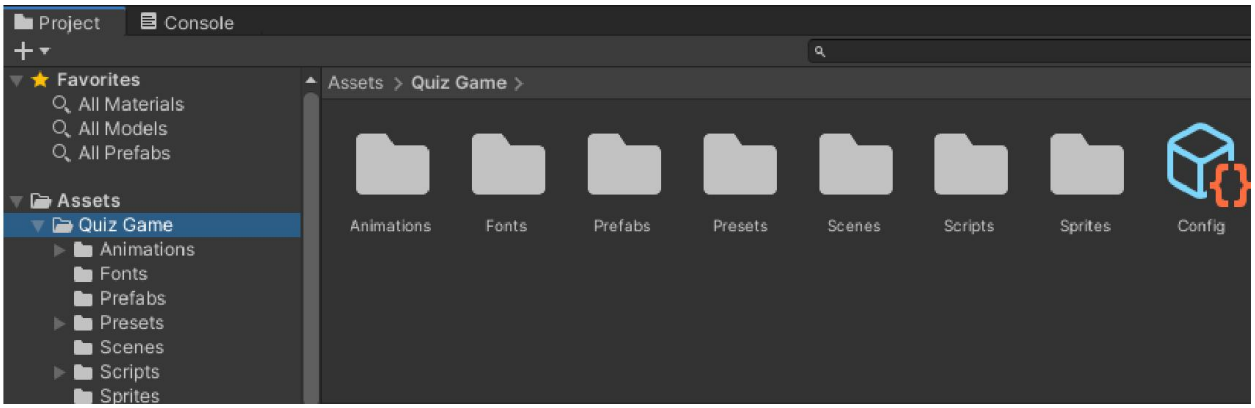

Рисунок 3.2 – Файлова структура в середовищі Unity

3.2Розширення інтерфейсу Unity для проекту

Редактор Unity знаменитий не тільки тим, що в ньому нескладно навчитися робити ігри використовуючи вже наявні інструменти, а й також можливістю доповнювати сам редактор, покращуючи його інструменти або створюючи нові конкретно під проект.

Найчастіше розширюють редактор саме з метою створення нових інструментів для зручності роботи в ньому. Перед початком роботи над розширенням редактора, необхідно засвоїти одне головне правило:

Робота в редакторі і робота в грі це два різних процеси.

Процес, коли запускаємо гру після складання проекту, зазвичай називають Runtime. Під час цього процесу редактор і його інструменти недоступні. Всі елементи, створені в процесі розширення редактора, також будуть недоступні в режимі Runtime, тому необхідно заздалегідь розділяти області, де будемо працювати з редактором, а де з об'єктами гри.

Всі області редактора або вікна, які бачимо при запуску Unity, мають свою певну роль. У вікні Hierarchy знаходяться всі об'єкти сцени, у вікні Project – всі файли проекту, а у вікні Inspector – налаштування і властивості об'єктів. Також можна додавати нові вікна якщо необхідно, наприклад – Scene, Game або Profiler [7].

На рис. 3.3 зображено налаштування і файл конфігурації QuizManager.

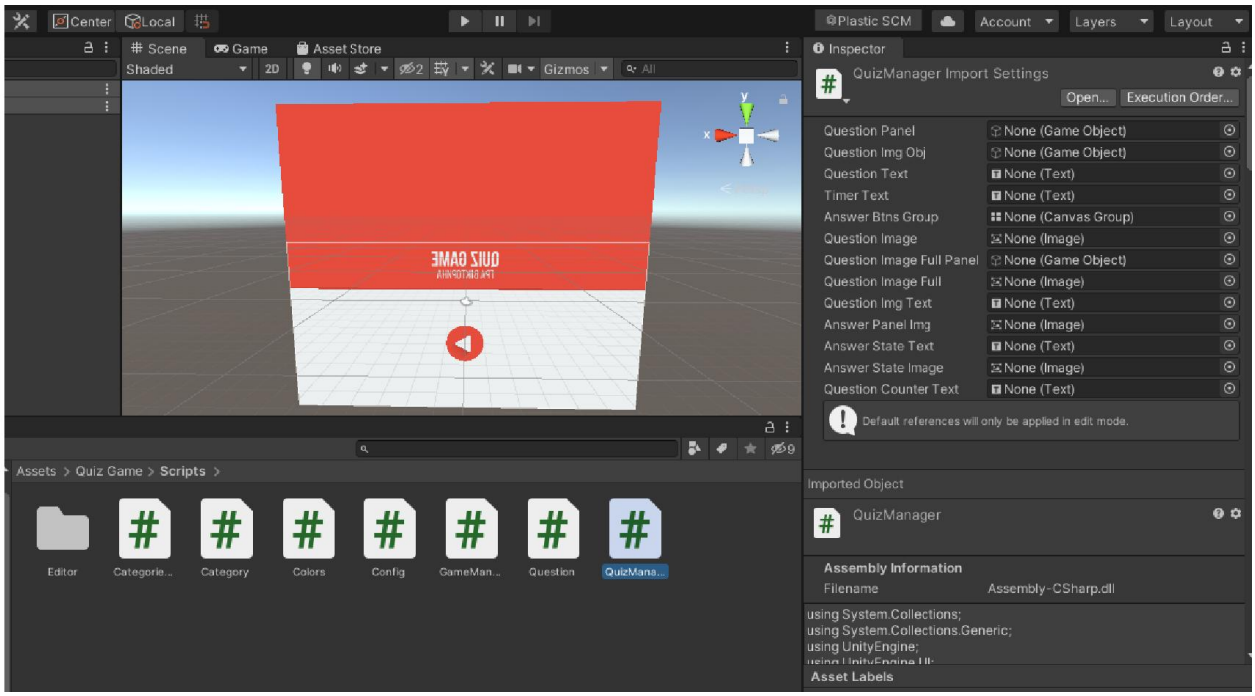

Рис 3.3 – Налаштування і файл конфігурації

Налаштування кожного компонента в інспекторі відображаються за допомогою стандартного редактора, до якого необхідно отримати доступ і переналаштувати його. У нас вже доданий компонент QuizManager. Щоб змінити його редактор, необхідно створити спеціальний скрипт, керуючий його відображенням в інспекторі. Перед тим як створити скрипт редактора, потрібно згадати головне правило поділу процесів редактора і Runtime'a, для цього досить використовувати одну зі спеціальних папок редактора – Editor.

Ця спеціальна папка Editor, як і інші спеціальні папки по типу Resource або StreamingAssets виконує окремі функції в редакторі Unity. У папках Editor повинні знаходиться тільки ті об'єкти, які використовуються тільки в редакторі. Там можуть знаходиться скрипти для розширення, зображення, та інші файли.

Завдання цієї папки в тому, щоб ці файли не потрапили в кінцеву збірку гри і тим самим не засмічували її об'єктами, які ніколи не використовуються в самій грі. Тому створюємо в проекті спеціальну папку Editor і новий скрипт в ній, для розширення компонента Config. За правилом гарного тону серед розробників, прийнято називати скрипти і розширення також, як і назви їх компонентів, тільки додаючи в кінці * Editor. У нашому випадку назвемо цей скрипт ConfigEditor.

У даному скрипті було розроблено загальний вигляд та назви доданих компонентів для розширення інтерфейсу, їх особливості, методи реалізації та головне готовий загальний вигляд розширення, додаткові функції його використання, застереження і поради для зручного користування.

На рис. 3.4 зображено розташування та властивості створеного скрипту ConfigEditor.

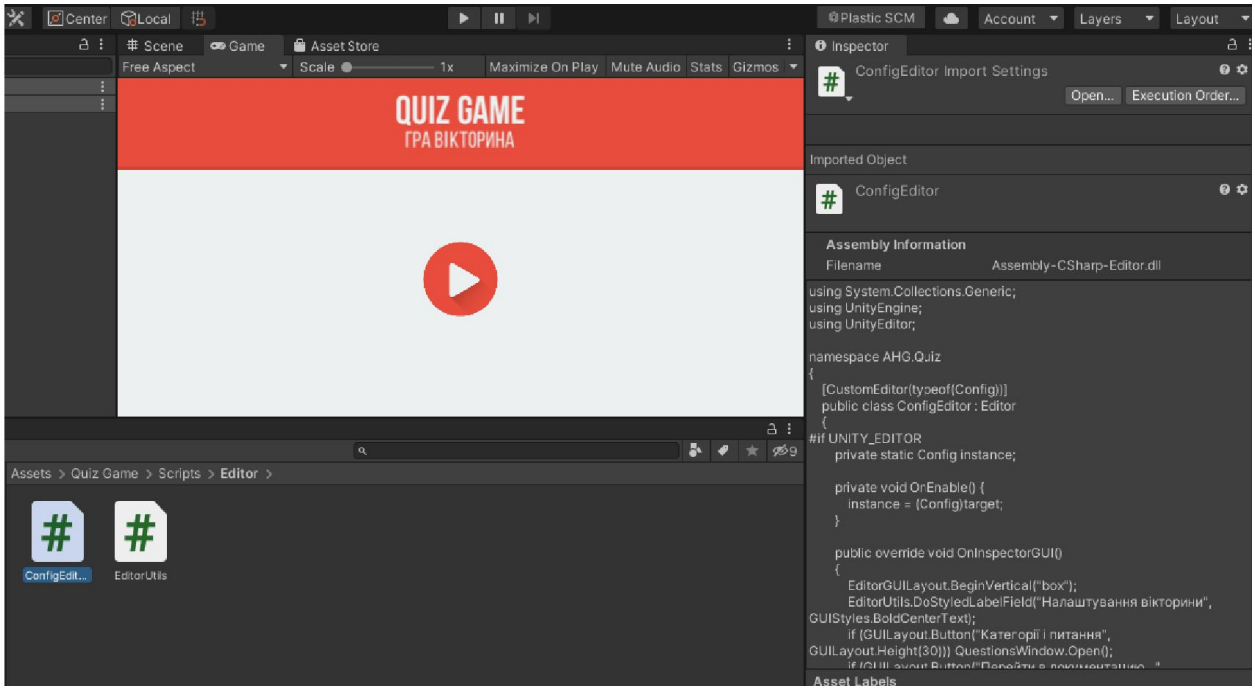

Рисунок 3.4 – Властивості скрипту ConfigEditor

На рис. 3.5 зображено як буде виглядати створений розширений інтерфейс у середовищі Unity, на перший погляд це лише одна кнопка, але за нею і знаходиться основна частина, де розміщено всі функції, наприклад додавання і віднімання можливих категорій, запитань і відповідей.

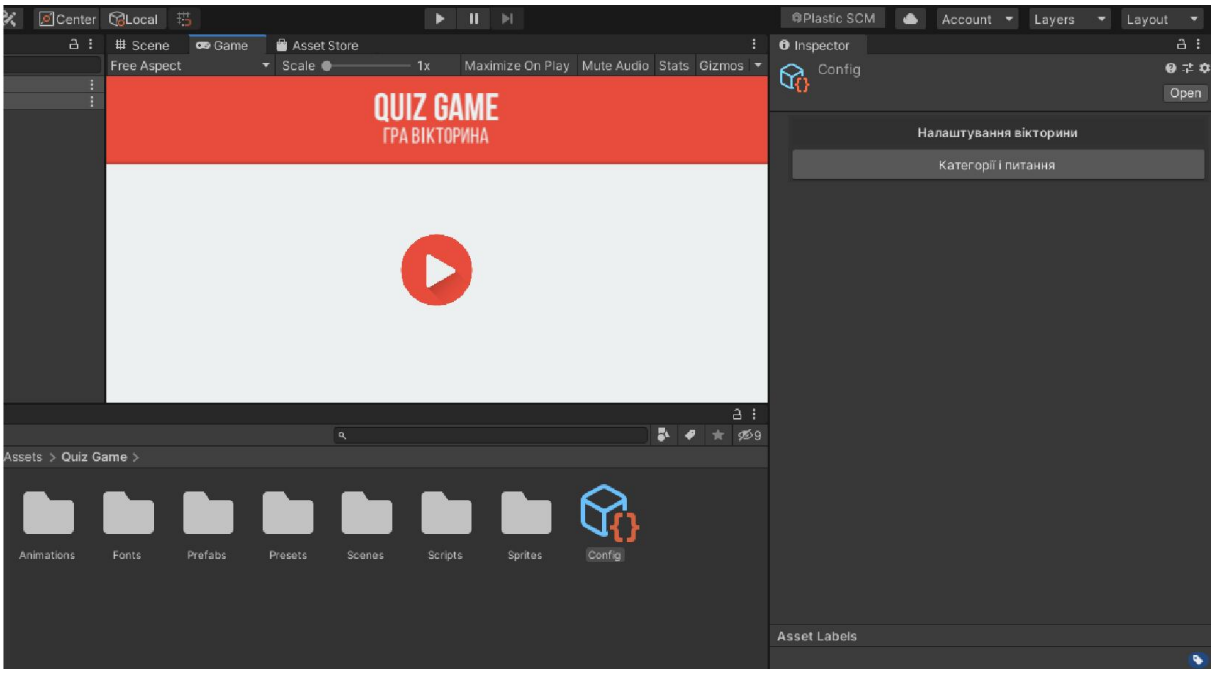

Рисунок 3.5 – Перша частина розширеного інтерфейсу

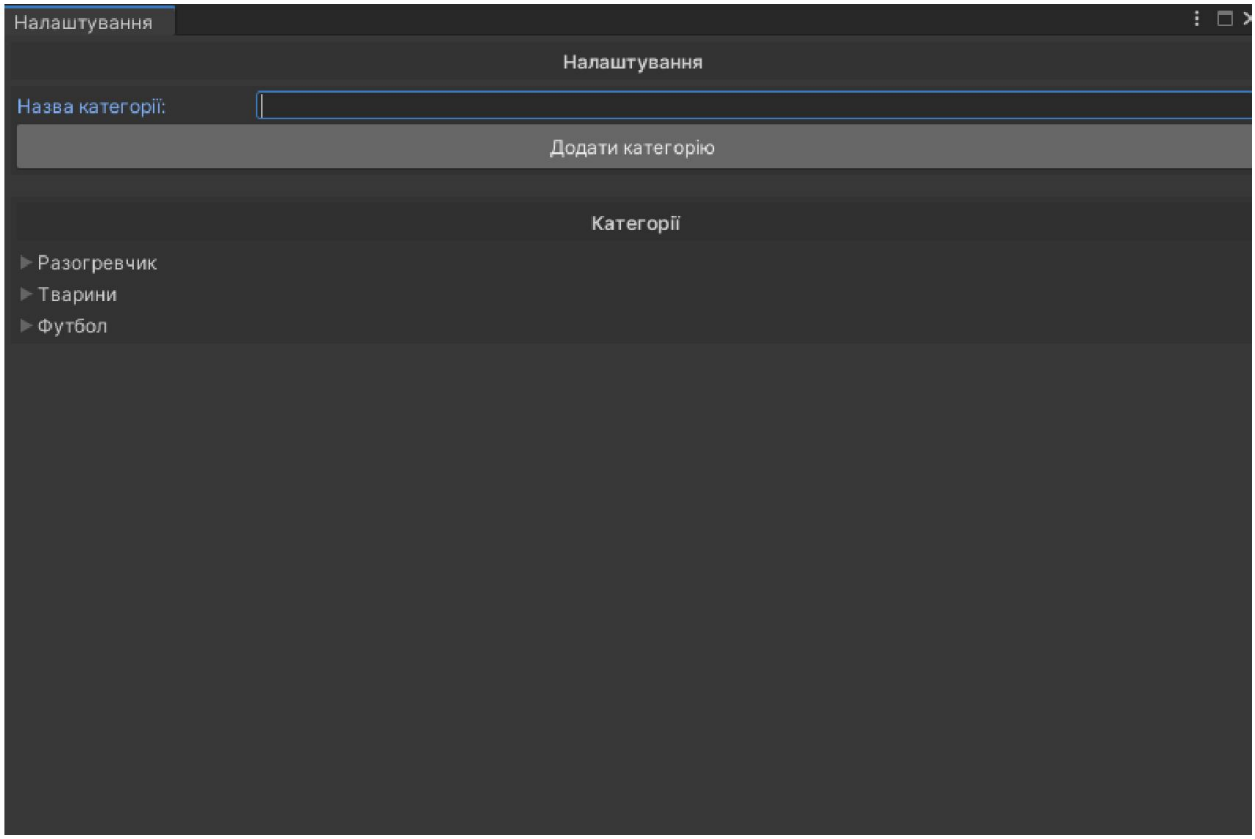

Рисунок 3.6 – Меню додавання категорій

Після того як категорія була створена, потрібно заповнити її відповідними до тематики категорії питаннями та відповідями.

Рисунок 3.7 – Додавання питань у створену категорію

Поле для додавання питання і трьох відповідей, з яких лише одна є вірною, зображено на рис. 3.8

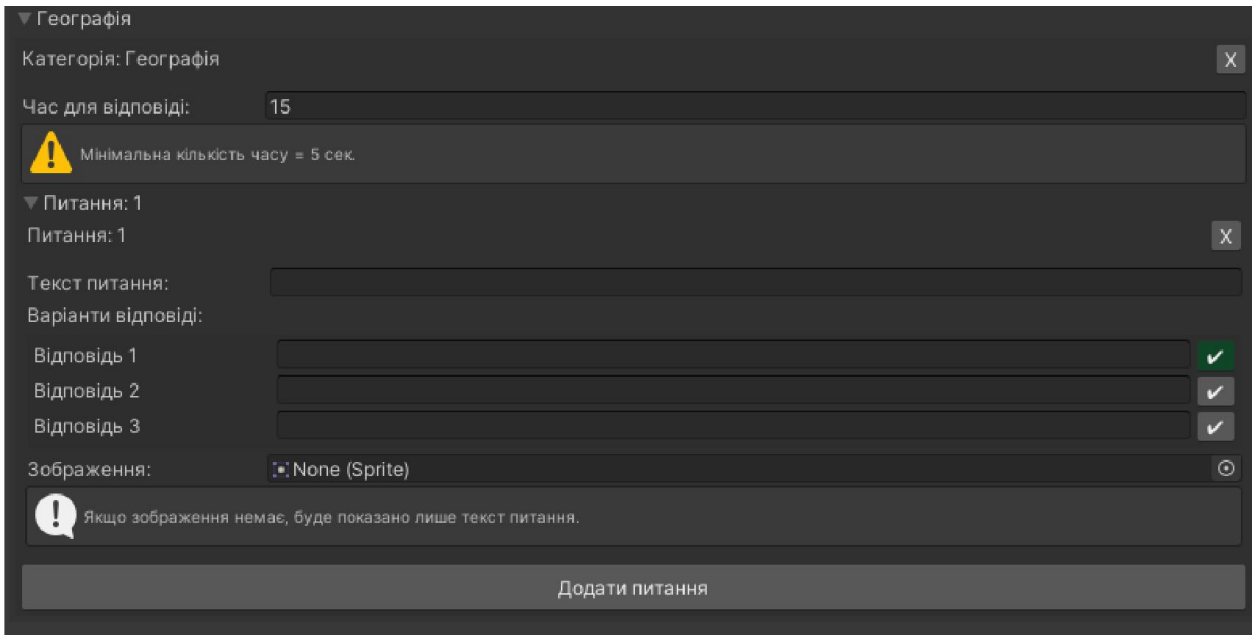

Рисунок 3.8 – Порожні поля для заповнення питанням і відповідями

При заповненні полів потрібними даними, питання зберігається, та відбувається перехід до створення наступного питання, в кожну категорію можливо додавати безліч запитань.

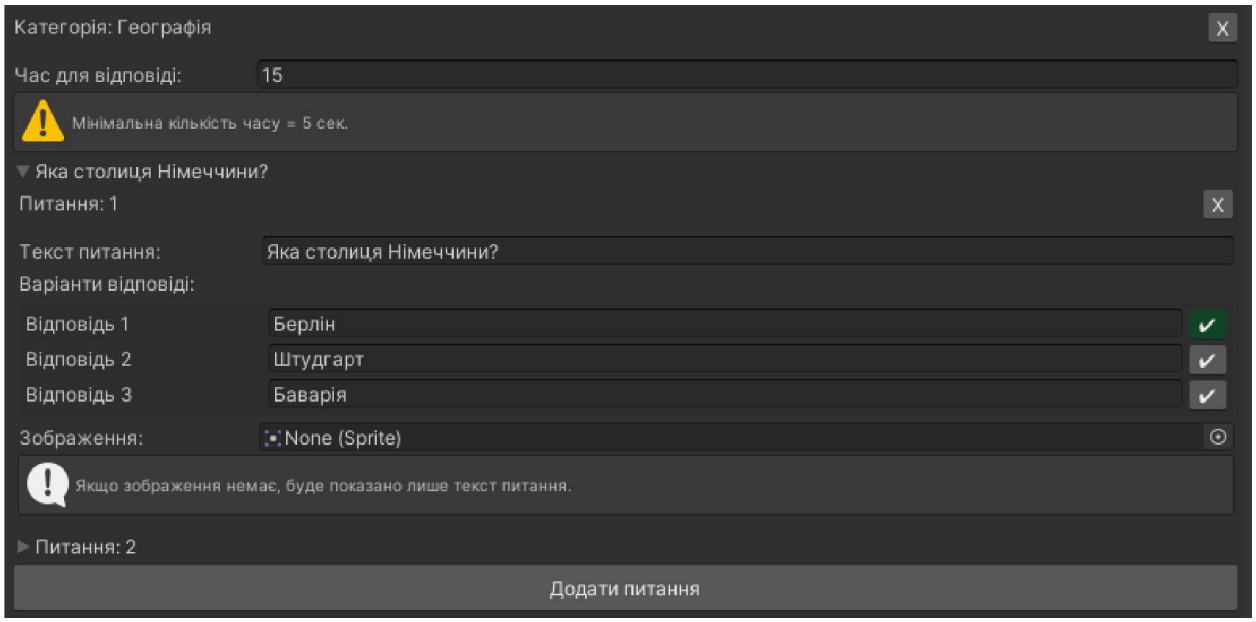

Рисунок 3.9 – Заповнені даними поля і збереження питання

3.3 Створення та введення в гру анімацій

Анімація - це динамічне середовище, в якому зображення або об'єкти маніпулюють для відображення у вигляді рухомих зображень. У традиційній анімації зображення малюється вручну на прозорій целулоїдній плівці і має бути знято та відображене на плівці.

Сьогодні більшість анімацій створюється за допомогою комп'ютерної графіки (CGI). Комп'ютерна анімація може бути дуже детальною 3D-анімацією, і з міркувань стилю можна використовувати двовимірну комп'ютерну анімацію з низькою пропускною здатністю або в реальному часі.

Інші поширені прийоми анімації використовують методи, які припиняють переміщення до дво- та тривимірних об'єктів, наприклад, ляльок або фігурок з глини/пластиліну. Техніка стоп-моушн, яка використовує живих акторів як інсценування, називається пікселацією.

Зазвичай анімаційні ефекти досягаються шляхом швидкої зміни безперервних зображень з невеликою відмінністю один від одного.

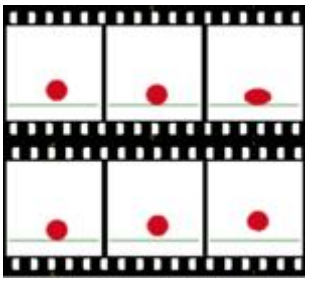

Рисунок 3.10 – Приклад створення елементарної анімації

Припустимо, що ця ілюзія, як і загальний кінофільм, спирається на феномен фі (зорова ілюзія, яка представляє серію таких зорових ілюзій)і бета-моушн (серія статичних зображень на екрані створює ілюзію плавності поточної сцени), але точна причина поки невідома. Засоби моделювання

механічної анімації, засновані на швидкому відображенні безперервних зображень.

Телебачення та відео – це популярні електронні засоби масової інформації, спочатку аналогові, але тепер цифрові. Для відображення на комп’ютерах розроблено такі технології, як GIF та Flash-анімація.

Окрім короткометражних, художніх фільмів, анімацій gif та інших засобів масової інформації, призначених для відображення рухомих зображень, анімація також широко використовується у відеоіграх, графіці та спецефектах.

Використовуються і прості механізми для фізичного переміщення різних частин зображення, наприклад рухоме зображення в «Чарівному ліхтарі», яке також можна розглядати як анімацію. Механічна анімація реального обладнання роботів називається електронною анімацією [28].

Нижче наведено основні типи анімації, які можна використовувати для створення цифрових персонажів для телешоу, комерційних програм, логотипів компаній, фільмів, відео чи ігор.

- Традиційна анімація;
- 2D векторні анімації;
- 3D комп'ютерної анімації;
- Графік руху;
- Стоп моушен.

Традиційна анімація. У минулому деякі зображення відображалися у вигляді кадрів, які швидко повинні змінюватися, намальованих на целулоїдній плівці кольоровими маркерами. Цей тип комп’ютерної анімації називають традиційною анімацією. Використовується, в основному для попередніх ескізів майбутніх персонажів.

Такий процес у більшостівипадків буде дуже дорогим і трудомістким, оскільки аніматор повинен створити набір таких кадрів на основній частоті 24 к/с щоб вони були різні. Цей метод використовується в загальному для персональних комп’ютерів, а також планшетів, які використовують

спеціальні комп’ютерні програми, за допомогою яких створюється анімація в стилі давніх мультфільмів Disney.

Рис 3.11 – Приклад традиційної анімації

2DВекторні анімації. Стиль анімації є найбільш поширеним. Його каркас створюється на майже рівній поверхні, та можливе опрацювання слоїв. Крім того, векторна анімація також використовує одні із видів традиційної. По суті, це те саме, за виключенням, що кадри обробляються, називаються малюванням та живописом. При цьому аніматор накладає на папір прозорі целулоїдні аркуші для малювання героїв мультфільмів, а потім перемальовує на плівку.

Нарешті, кадрирізних персонажів створені авторомнакладаються.За допомогоюмайже прозорої плівки це дозволяє завершити створення композиції, що складається з різних персонажів та елементів.

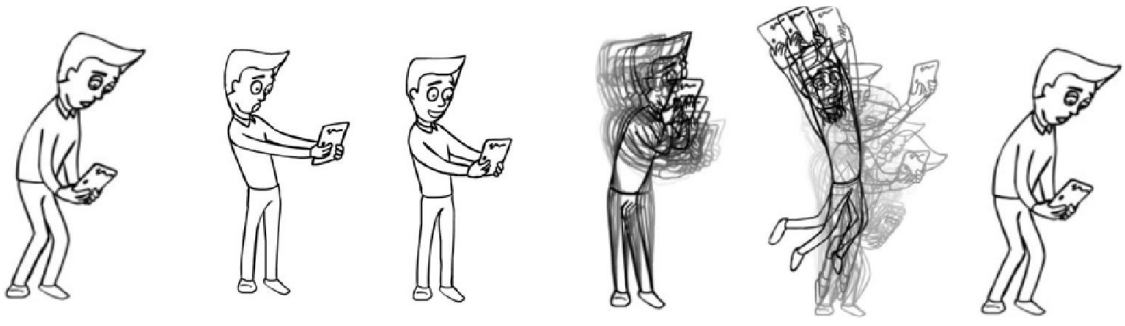

Рисунок 3.12 – Приклад векторної анімації

3D комп’ютерна анімація. 3D-анімація майже на сто відсотків відрізняється від інших видів мультиплікацій в комп'ютерній графіці.

Незважаючи на те, що йде використання тих самих принципів руху і композицій, технічні методи, які використовуються для вирішення різних завдань, досить різні.

Наприклад у 3D-анімації аніматор не обов'язково повинен бути спеціалістом-графіком. Це не стільки малювання, скільки гра ляльками. Це звикли називати комп'ютерною графікою (CGI). Це відбувається, коли комп'ютерний аніматор створює серію зображень і об'єднує їх, щоб утворити анімацію та використовує комп'ютерну графіку, щоб завершити поєднання динамічних і статичних зображень. Персонаж, створений у 3D, відображається на моніторі в цифровому вигляді, а потім об'єднується з рамкою, яка дозволяє створювати різні анімації для кожної моделі.

Анімації створюються шляхом формування моделей в різних ключових епізодах, щоб потім комп'ютер виконав їх «відтворення», інтерпретуючи анімацію шляхом додавання проміжних (дублюючих) кадрів між ключовими (основними) кадрами. Крім того, для обробки кривих, що представляють окремі частини об'єкта в різний час, потрібно багато часу.

3D-анімація повинна враховувати всіх персонажів, навіть тих, які закриті та невидимі увизначений період часу. Головна відмінність цих типів анімації виражається в тому, що в традиційній анімації та 2D анімації автори використовують окремі кадри для роботи, тоді як у 3D-анімації завжди є потік, що не переривається.

Якщо такий потік зупиняється, це вважається помилкою. Навіть якщо персонаж залишиться вдома,безперервні кадри будуть працювати, створюючи ілюзію реальності.

Рисунок 3.13 – Приклад 3D-анімації

Графік руху. Технологія використовується для створення рекламних відеороликів, анімованих логотипів, вступних назв фільмів і показу реклами. Це робиться за допомогою рухомого графічного тексту та елементів, простішими словами динамічною графікою. Це процес, який використовує «відтворення» анімованих кадрів для створення плавного руху між кадрами. Програма візуалізації кадрів підтримує сценарії, які автоматично змінюють анімацію для створення кількох ефектів. 3D композитинг створюється за допомогою відносно рухомих плоских елементів, створюючи ілюзію об’єму. В основному вони супроводжуються музичними чи звуковими ефектами. Ці об’єкти часто використовують в різноманітних мультимедійних сценах і проектах.

Рисунок 3.14 – Приклад анімації графіку руху

Стоп-моушн. Анімація стоп-моушн – це форма комп'ютерної анімації, яка більше схожа на звичайну традиційну анімацію. Все, що потрібно зробити, це сфотографувати об’єкт, і коли ви перемістите об’єкт на відносно невелику відстань, ви зробите ще один знімок. Цей процес буде повторюватися багато разів, і коли зображення будуть скопійовані один в одного, буде створюватися враження руху. Існує багато форм анімації стоп-моушн, включаючи вирізання паперу, глиняну анімацію та ляльки.

Але цей процес відрізняється тим, що для створення анімації потрібно знімати об’єкти, які повинні будуть рухатися кілька разів поспіль.

Рисунок 3.15 – Приклад стоп-моушн анімації

У цьому проекті необхідно використовувати анімацію, оскільки вона доповнює наше життя та додає рухливості, показуючи користувачеві, що гра взаємодіє з ним і реагує на його дії.

На рисунках наведених нижче буде представлено приклади створення анімації в Unity, їх елементи, стани об’єктів при яких спрацьовує анімація та позиціонування.

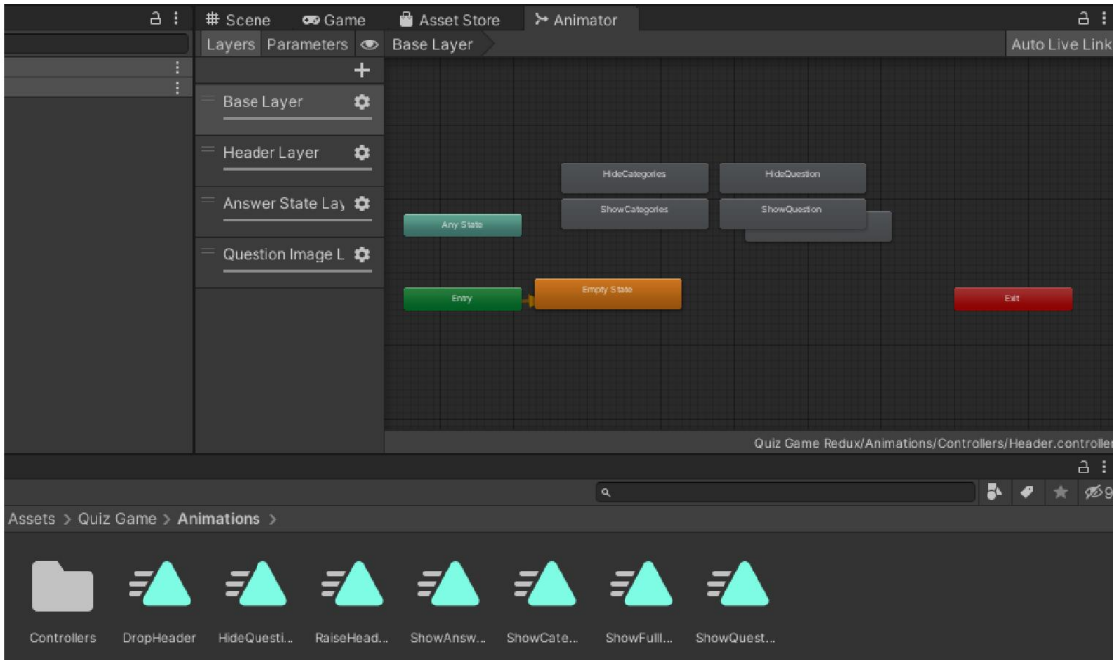

Рисунок 3.16– Елементи з яких складається анімація

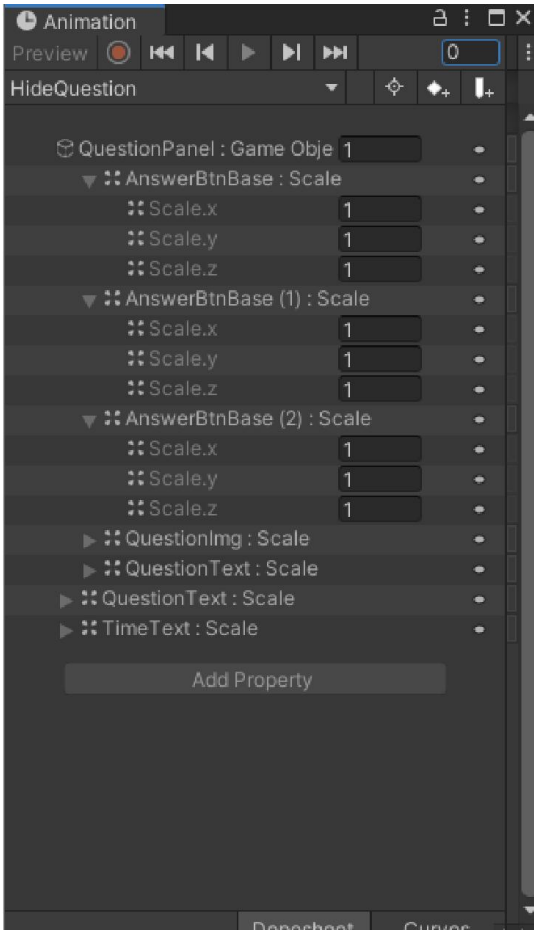

Рисунок 3.17 – Позиціонування анімацій

3.4 Інтегрована БД в Unity.

Оскільки в нашій грі буде використано платформу Unity 3D, створювати окрему базу даних не потрібно, усі інструменти уже є в наявності та повністю готові до використання, у самому інтерфейсі з'являється вікно для заповнення інформацією, в нашому випадку питання до гри, та вікна для заповнення відповідей, одна з яких (з нульовим індексом) буде правильна, інші ні.

Всі питання та відповіді автоматично генеруються у CS–скрипт та зберігаються там уже в закодованому вигляді.

Як було описано раніше, окрема, зовнішня БД для нашого проекту не буде потрібна, оскільки за допомогою методу «OnInspectorGUI» було розширено внутрішнє програмне забезпечення, та додано окремі, потрібні

конкретно під наш проект, елементи, скористувавшись ними було створено БД, та зображено на рис. 3.18 та 3.19.

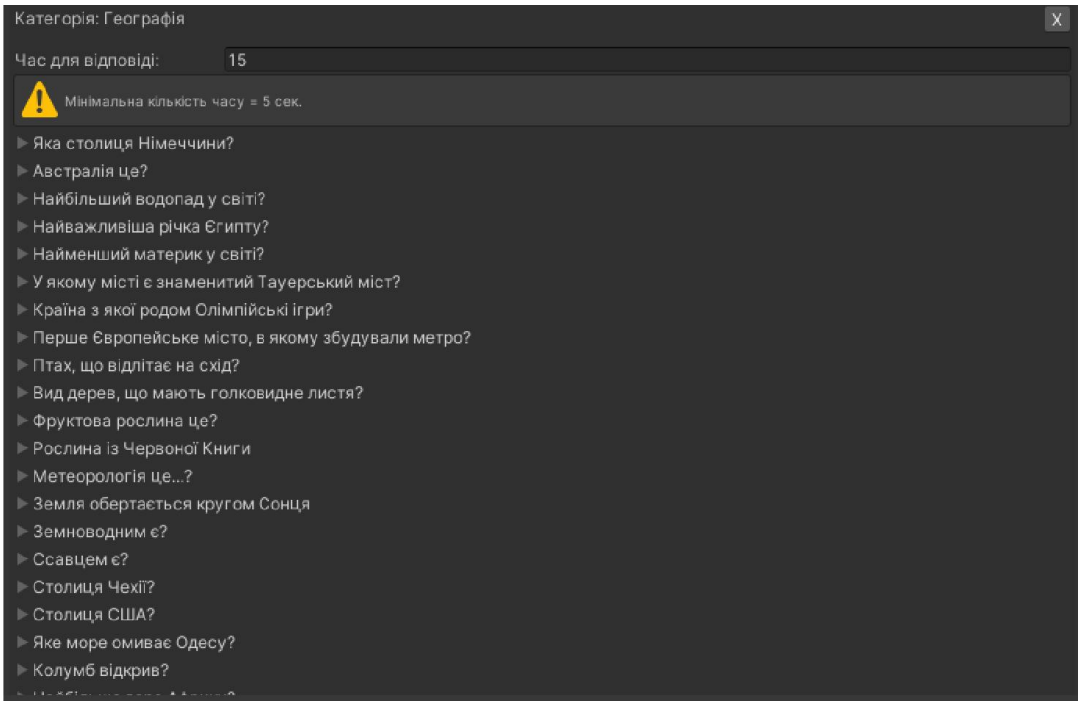

Рисунок 3.18 – Питання у БД Категорії «Географія»

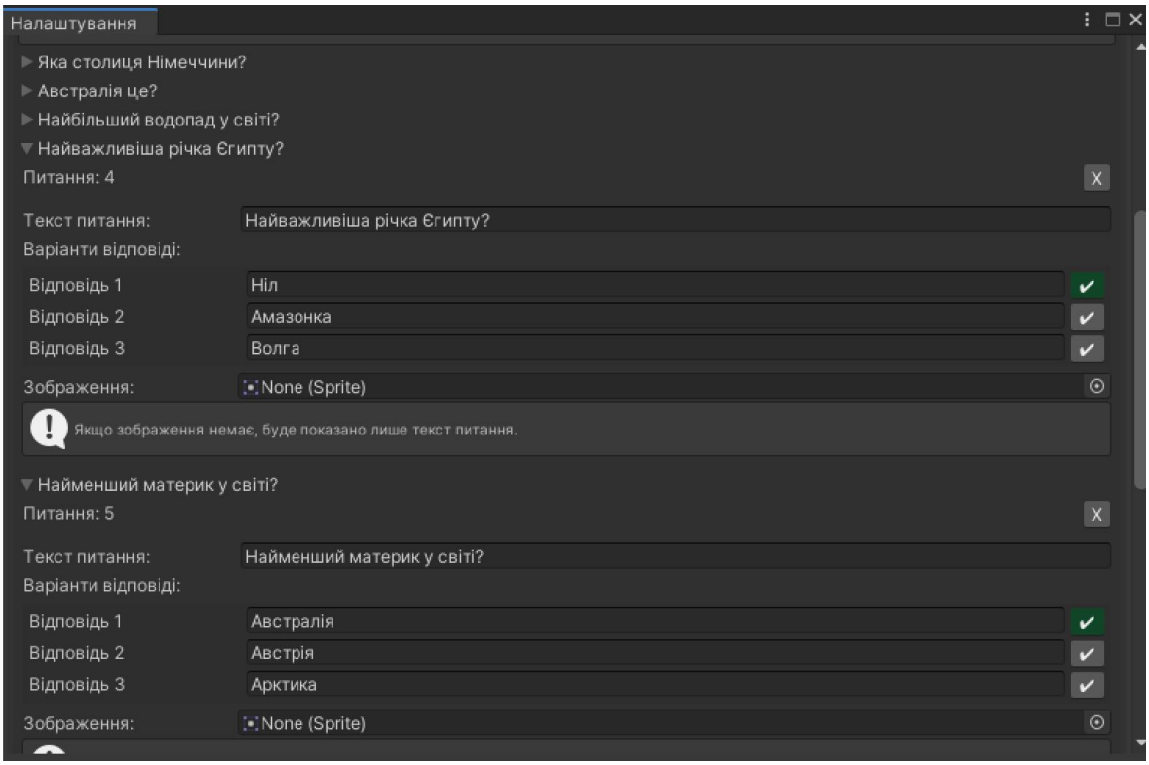

Рисунок 3.19 – Варіанти відповідей у БД Категорії «Географія»

3.5. Інтегрування додатку на ОС Android

Для вдалого переформатування проекту у середовище розробки потрібного для нашого проекту, нам знадобиться ще ряд деяких інструментів, і для зручності розробника Unity пропонує всі ці інструменти встановити та налаштувати, для цього лише потрібно завантажити необхідний функціонал, що працює досить інтуїтивно та за допомогою підказок від платформи.

Для початку нам потрібно завантажити Android SDK та JavaJDK, зазвичай вони завантажуються разом із Unity, проте якщо цього не сталося, їх потрібно завантажити, встановити та підв’язати вручну.

SDK (Software development kit) – це комплект для розробки програмного забезпечення, який допомагає розробникам створювати додатки для конкретних платформ. Це можуть бути комп'ютери, ігрові приставки або мобільні пристрої.

Якщо точніше, мобільні SDK дають розробникам додатків все необхідне для того, щоб легко створювати всілякі високопродуктивні додатки для смартфонів і планшетів, які можна опублікувати на маркетплейсах Google Play і App Store.

Сьогодні індустрія зміцнилася в думці, що SDK це бібліотека, вбудована в додаток, а API це хмарні сервіси, які працюють спільно з SDK або додатком.

Ви створюєте що-небудь за допомогою SDK. Ви використовуєте або споживаєте що-небудь за допомогою API. Ви використовуєте SDK для доступу до функціоналу бібліотеки, а API для управління ним. Далеко не у кожного розробника в арсеналі є всебічні знання в програмуванні і великі навички розробки програмного забезпечення. І навіть ті, хто всім цим володіє, вважають за краще економити значну кількість часу, використовуючи інструменти програмування з мобільного SDK.

Інтегруючи якісний і добре написаний код з різними допоміжними матеріалами, ви можете додати нові особливості в свій мобільний додаток, щоб воно напевно було надійним і не поступалося додаткам ваших конкурентів. До того ж, мобільний SDK допомагає вашому додатку стабільніше і краще працювати.Всім хочеться, щоб SDK був не тільки високої якості, але і захищав інформацію кінцевих користувачів.

Дуже важливо шукати мобільний SDK, для якого важливий дозвіл користувача про використання персональних даних, хоча зазвичай використовуються кілька бібліотек, інтегрованих з додатком. Високоякісний мобільний SDK, який забезпечує безпеку вашим користувачам, допоможе поліпшити враження від роботи з додатком, його надійність і показники утримання клієнтів.

Java Development Kit (JDK) – це один з трьох основних технологічних пакетів, які використовуються в програмуванні на Java, поряд з JVM (Java Virtual Machine) і JRE (Java Runtime Environment). Важливо розрізняти ці три технології, а також зрозуміти, як вони пов’язані:

- JVM , це компонент платформи Java, який виконує програми.
- JRE, це дискова частина Java, яка створює JVM.
- JDK, дозволяє розробникам створювати програми на Java, які можуть виконуватися і запускатися JVM і JRE.

Розробники, які незнайомі з Java, часто плутають Java Development Kit і Java Runtime Environment. Відмінність полягає в тому, що JDK – це пакет інструментів для розробки програмного забезпечення на основі Java, тоді як JRE – це пакет інструментів для запуску коду Java.

Технічне визначення: JDK є реалізацією специфікації платформи Java, включаючи бібліотеки компілятора та класів.

Повсякденне визначення: JDK – це програмний пакет, який ви завантажуєте для створення програм на основі Java.

На додаток до JRE, що є середовищем, що використовується для запуску програм Java, кожен JDK містить компілятор Java.

Компілятор – це програма, здатна приймати необроблені файли .java, які є звичайним текстом, і перетворювати їх у виконувані файли .class.

JRE можна використовувати як окремий компонент для простого запуску програм Java, але він також є частиною JDK. JDK вимагає JRE, оскільки запуск програм на Java є частиною їх розробки.

Для того щоб створити готовий apk. файл потрібно в системних налаштуваннях Unity обрати платформу та налаштувати основні та додаткові функції.

Спочатку відкриємо Build Settings, та оберемо платформою під яку ми саме створювали наш проект – ОС Android.

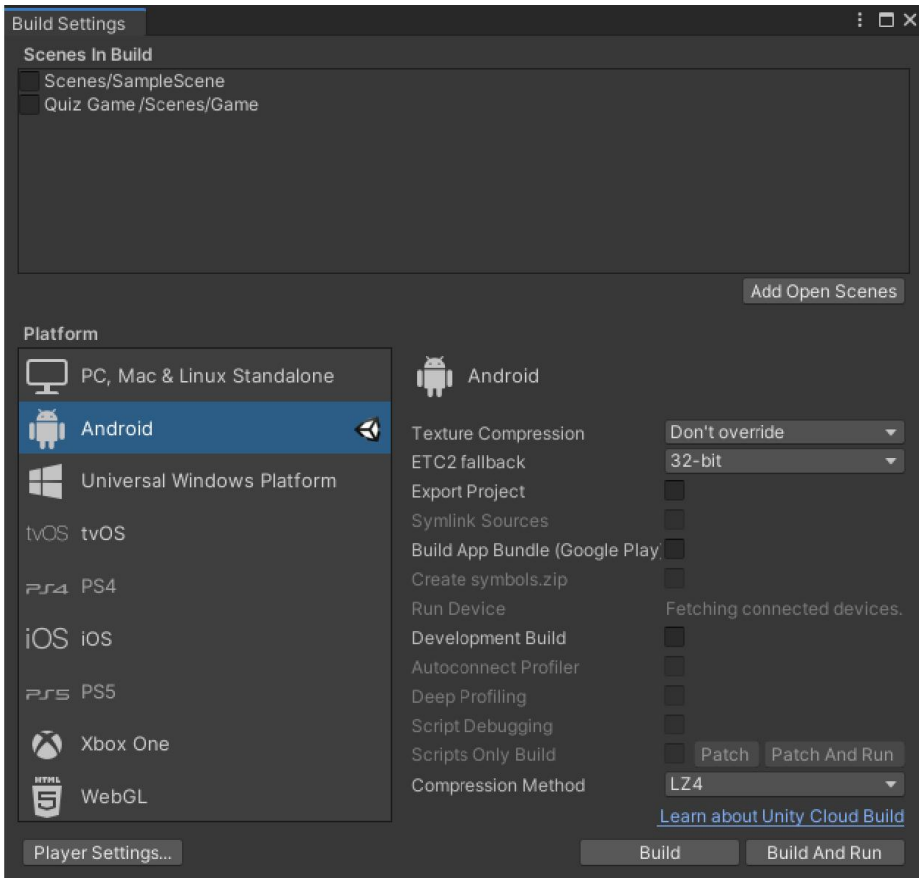

Рисунок 3.20 – Вибір платформи для збірки проекту

Виділити сцени які будуть використовуватися у кінцевій збірці проекту:

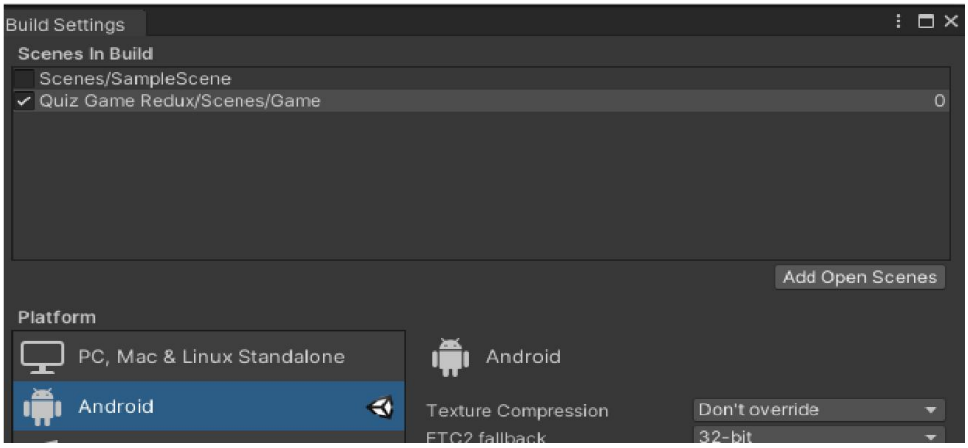

Рисунок 3.21 – Вибір потрібної сцени

Перевірити та відредагувати дляконкретного проекту налаштування у вікні Project Settings, у розділі Player:

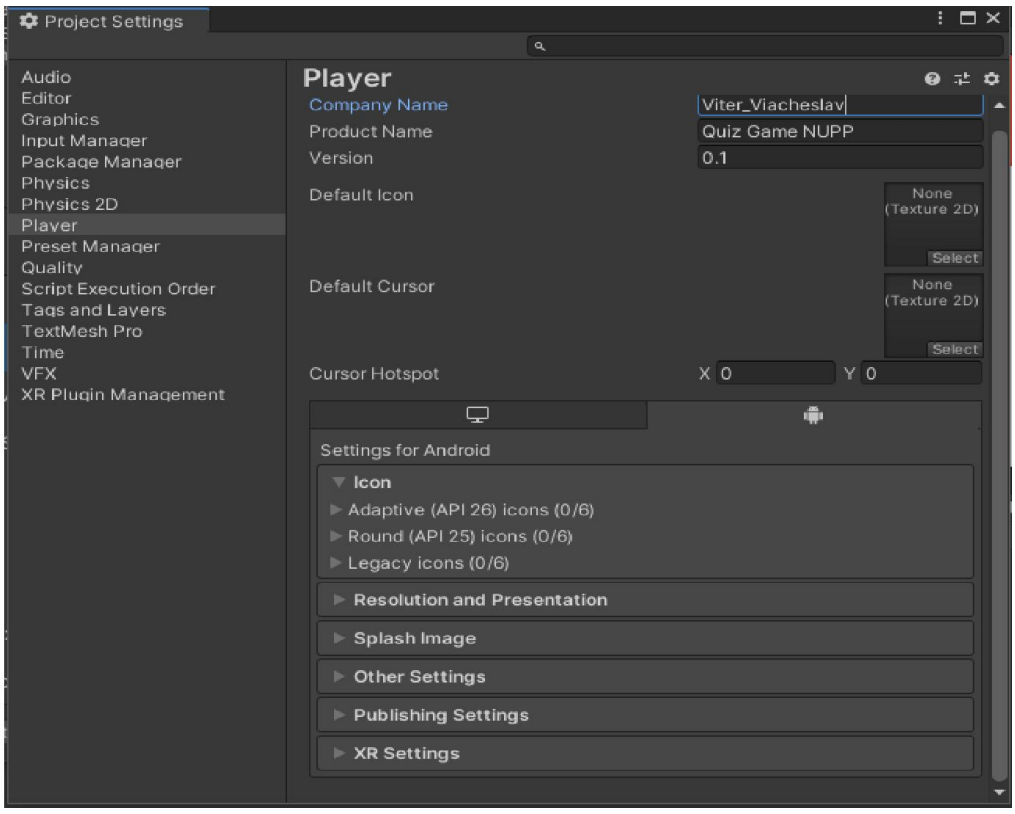

Рисунок 3.22 – Редагування даних у розділі «Player»

Заповняти дані поля потрібно спеціальним чином, щоб збірка була стабільна та коректна:

- CompanyName (писати англійською і краще без розділових знаків і пробілів);
- ProductName (аналогічно - англійською та без спеціальних символів та пробілів);
- Version (можна залишити значення за замовчуванням, але якщо програма збирається повторно, то значення треба змінювати на більше, тоді при встановленні нової версії програма оновиться)

Встановити зображення для іконки програми можна додавши його вDefaultIcon. Якщо необхідно, у розділі ResolutionandPresentation можна зафіксувати орієнтацію програми: горизонтальну (Landscape) або вертикальну (Portrait).

У Unity доступні додаткові параметри візуалізації для ігор: меню Edit, далі Project Settings далі Quality. У цьому вікні налаштовуються параметри рендеринга, іншими словами якості картинки гри, які можна налаштувати індивідуально.

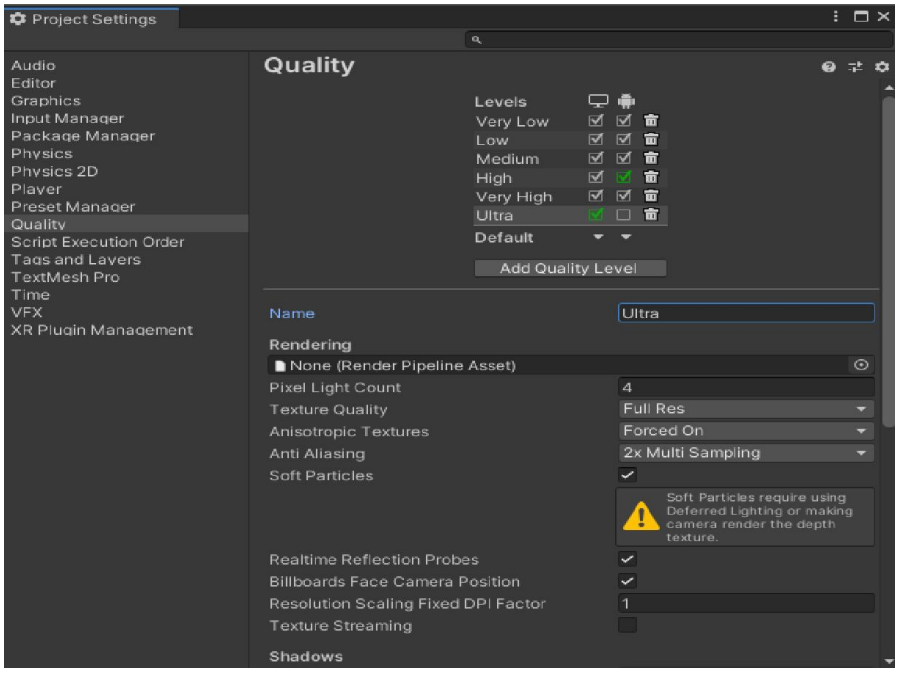

Рисунок 3.23 – Встановлення якості гри

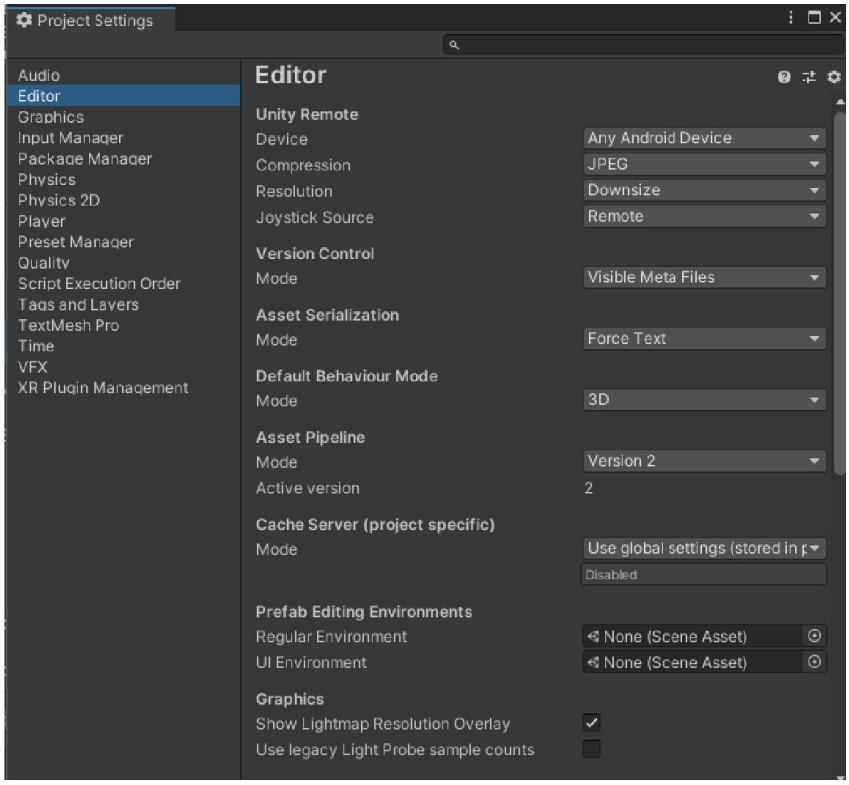

Рисунок 3.24 – Налаштування редактора гри

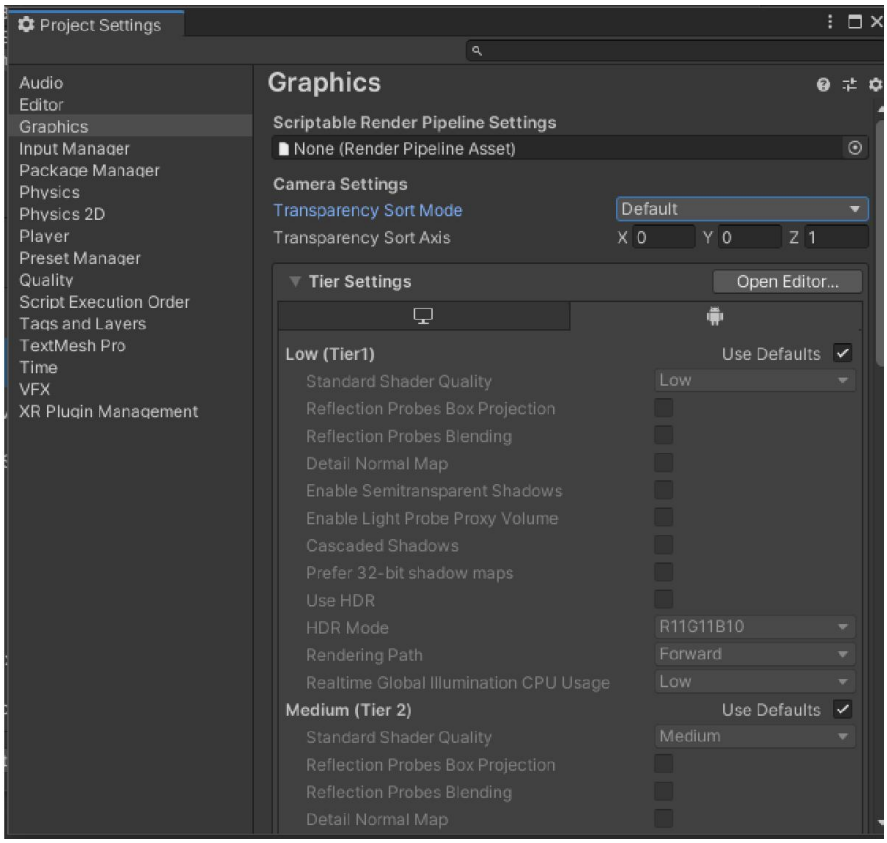

Рисунок 3.25 – Налаштування графіки(1)

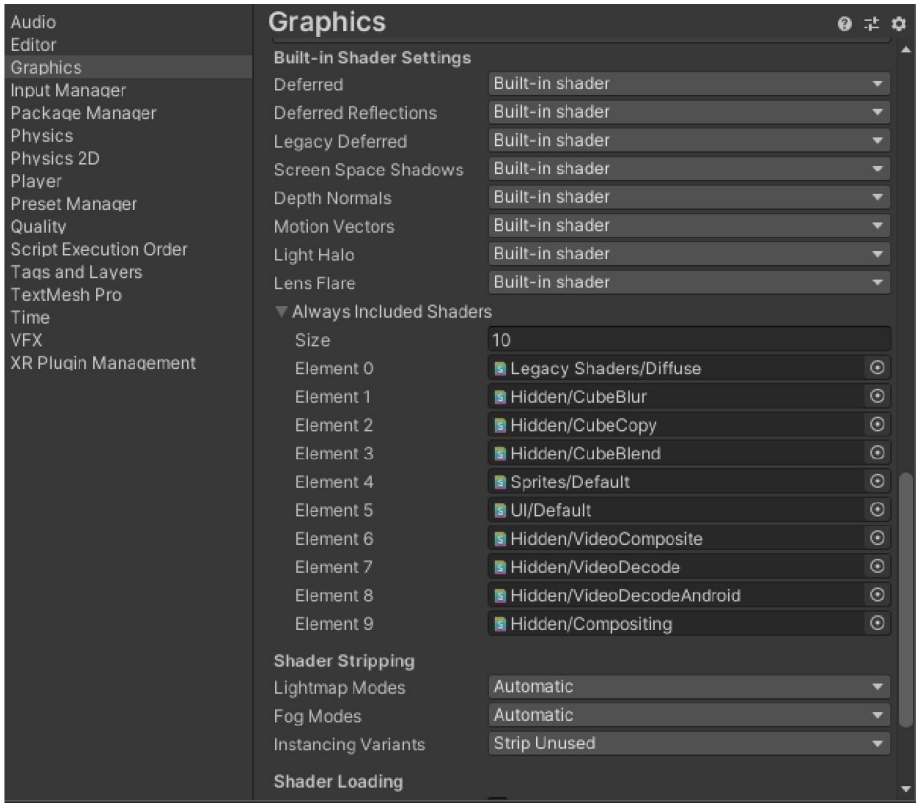

Рисунок 3.26 – Налаштування графіки(2)

Для створення готового apk. файлубуло системно налаштуванно Unity, обрано платформу та налаштовано додаткові функції.

Готовий згенерований apk-файл зберігається у папці із компонентами гри, для його запуску, потрібно щоб на ПК був присутній емулятор Android–додатків, або просто скачати чи перенести файл на телефон, що має ОС Android та встановити додаток на телефон.

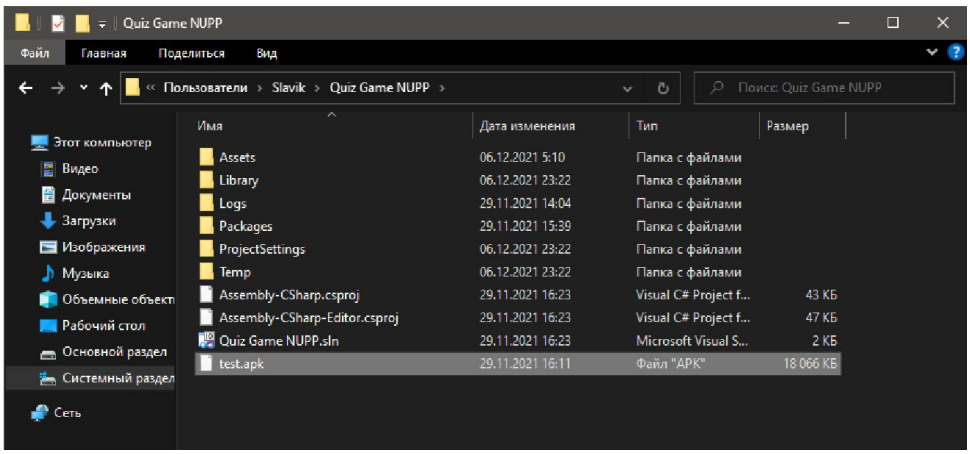

Рисунок 3.27 – Згенерований apk-файл у папці з грою

3.6 Опис геймплею гри

Запустивши додаток на мобільному пристрої в першу чергу завантажується головне меню гри та вмикається фонова музика, гучність якої регулюється кнопками пристрою. Також на головному меню присутній спрайт Play, який зображений іконкою, при натисненні на яку відбудеться перехід на сцену, де гравцю пропонується обрати категорію зі списку.

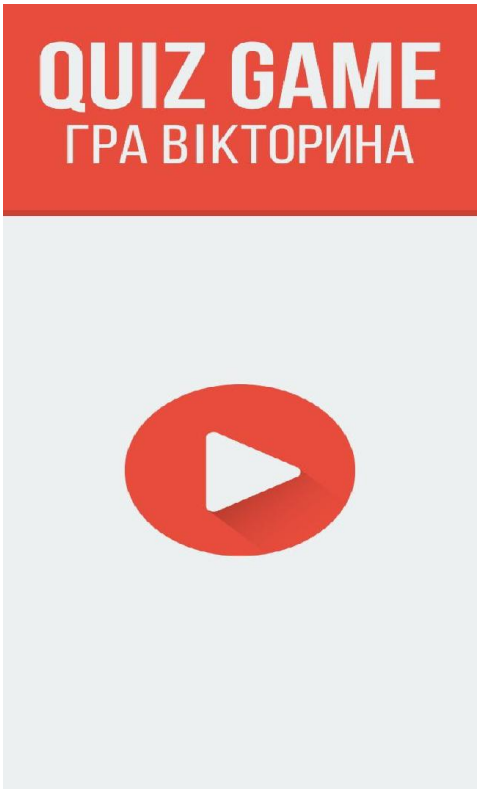

Рисунок 3.28 – Головне меню гри

При натисненні на кнопку «Play»відбувається перехід до сцени, де пропонується вибір категорій. Користувачу для гри пропонується наступний перелік можливих категорій: Тварини, Футбол, Географія, Історія, Спорт, Кінофільми, Музика, Книги, Мультфільми, Авто, Космос, Відомі логотипи. Кожна категорія має від 20 до 25 питань.

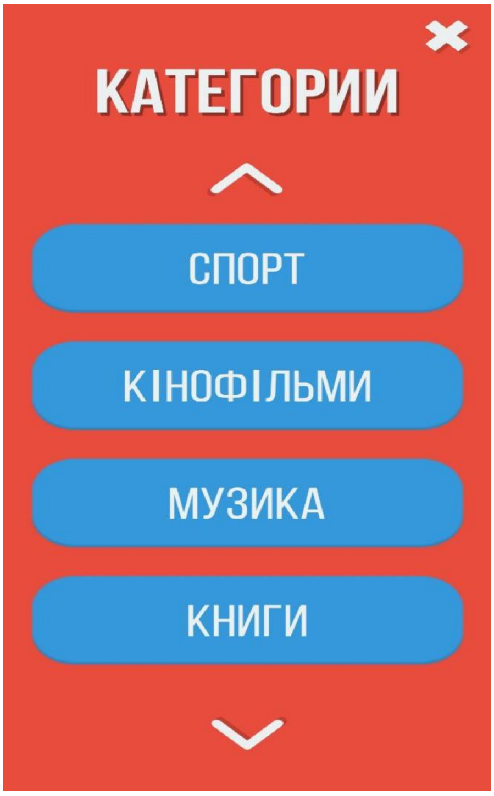

Рисунок 3.29 – Список категорій (1)

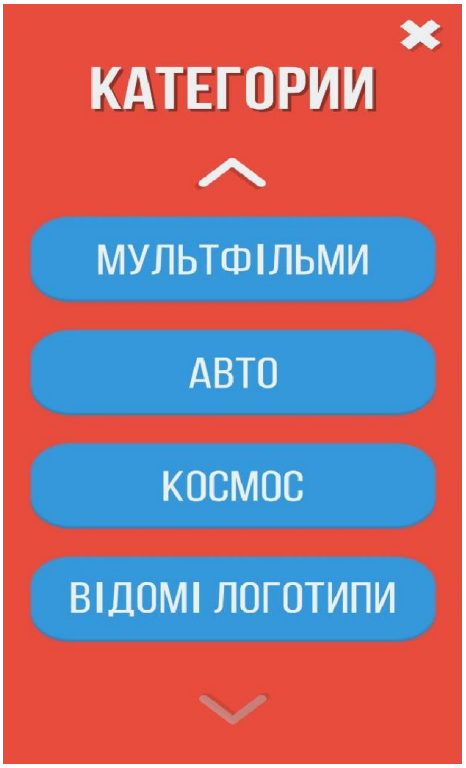

Рисунок 3.30 – Список категорій (2)

При обрання категорії відкривається вікно із запитанням та варіантами відповідей, також автоматично запускається таймер, який обмежує час гравця на роздуми, він розташований у нижній частині екрану, якщо гравець не дав відповідь до закінчення таймеру йому автоматично зараховується програш та відбувається перехід до головного меню.

Рисунок 3.31 – Процес гри (Варіант питання з картинкою)

При наданні користувачем правильної відповіді, з’являється вікно, що ілюструє успіх гравця та відбувається перехід до наступного питання.

При наданні користувачем неправильної відповіді, з’являється вікно що сповіщає про невдачу та відбувається перехід до головного меню гри.

Рисунок 3.32 – Надання правильної відповіді

Рисунок 3.33 – Надання неправильної відповіді

Також можливий варіант, коли гравець не обирає ніяку відповідь, тоді спливає час виокремлений на надання відповіді, що автоматично сприймається, як програш.

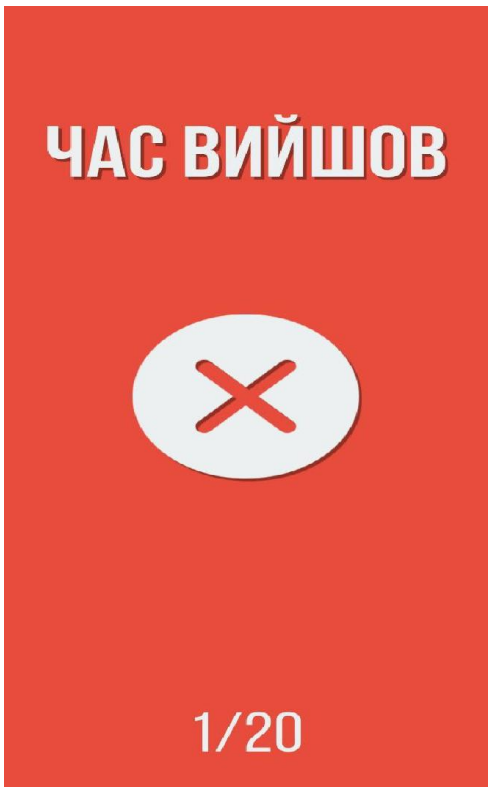

Рисунок 3.34 – Ненадання відповіді у відведений час

Основна задумка гри вікторини полягає у наданні користувачем правильних відповідей за допомогою розуму. Це чудовий спосіб перевірки своїх знань у різних галузях та можливість дізнатися щось нове.

На сьогоднішній день недостатньо просто створити гру та завантажити її до маркетплейсу, потрібно зацікавити користувача, в основному розробники домагаються цього за рахунок додавання різноманітних таблиць рекордів, змагань між гравцями, яскравого інтерфейсу, оновлення запитань для користувачів та різноманітних внутрішньо-ігрових конкурсів і розваг.

Суть даного проекту базується саме на основному принципі, тобто створити умови для залучення логіки і розуму користувачів.

3.7 Тестування та підтримка

Після завершення роботи над кодом, механікою гри, наповненням її контентом та усіх умов за яких гра може функціонувати, відбувається її доопрацювання. Коли гра ще не добудована повністю, але можливість зіграти вже з’явилась її називають альфа-версією.

Вона може містити деякі помилки, можлива відсутність деяких можливостей для користувача, проблеми зі звуком чи відсутність потрібних об'єктів. Для того щоб виявити такі проблеми потрібні тестувальники. Вони запускають гру та намагаються спробувати усі можливі функції гри .

Види тестування:

- Функціональне тестування.

Мета: визначити всі можливі поведінкові сценарії користувачів і перевірити їх в рамках продукту.

Функціональне тестування – це тестування ПЗ з метою перевірки реалізованості функціональних вимог, тобто можливостіПЗ у певних умовах вирішувати завдання, потрібні користувачам.Функціональні вимоги визначають, що саме робить ПЗ, які завдання воно вирішує.

Тестування було виконано шляхом виконання ігрового докладання за певних умов, для спостереження та оцінки аспектів роботи гри.У процесі функціонального тестування проводився аналіз програми для виявлення невідповідностей між існуючими особливостямивиконання програми та вимогами до додатка.

- Тестування продуктивності.

Мета: визначити вплив на гаджет продукту, чи є проблеми оптимізації.

Існує багато видів тестування продуктивності. Класифікація видів тестування продуктивності будується на основі того, яку мету переслідує певний вид тестування. Як правило, тестування продуктивності переслідує не

одну, а кілька цілей у зв'язку з тим, багато типів тестування в ході його проведення поєднуються з іншими цілями або повторюються кілька разів в ході циклу тестування.

Основна відмінність продуктивності тестування також полягає в тому, що воно відбувається тільки після повного функціонального тестування. Помилки функціональності не виправляються під час тестування продуктивності.Для цього виду тестування найчастіше виділяється окремий навантажувальний стенд, що повторює копію промислового стенду. У зв'язку з масовим поширенням Agile методологій тестування продуктивності також інтегрується у життєвий цикл розробки програмного забезпечення.

- Тестування безпеки

Мета: перевірити наявність проблем в коді, якими можуть скористатися нечесні гравці.

Це тип тестування програмного забезпечення, який виявляє вразливості, загрози, ризики у програмному додатку та запобігає зловмисним атакам від зловмисників. Основна ціль тестування безпеки є виявлення всіх можливих лазівок та слабких місць у програмній системі, які можуть призвести до втрати інформації, доходів, репутації. Розробник обов’язково повинен взяти до уваги та виправити можливі прогалини у коді.

- Юзабіліті

Мета: оптимізація UI / UX в рамках постійного користування та об’єктивна оцінка зручності і оглядностіта читабельності.

Юзабіліті тестування – це дослідження, яке виконується з метою визначення, чи зручний додаток для його передбачуваного застосування, тобто це метод оцінки зручності продукту у використанні,заснований на залученні користувачів як тестувальників, та підсумовування отриманих від них висновків. Юзабіліті тестування є найпоширенішим підходом при тестуванні додатків.

Після цього етапу розробники створюють рекламний трейлер, показуючи ігровий процес у ролику, та даючи потенційним користувачам уявлення про що саме буде відбуватися у грі, та що очікувати гравцям у готовому продукті[5].

Потім у відкритий доступ викладається бета-версія гри, протестувати яку можуть і потенційні користувачі гри. У цій версії відбувається пошук помилок, перевірка чи коректно об'єкти ігрового світу взаємодіють з користувачем, перевірка управління. Також можлива зміна ігрового балансу, деяких можливостей ігрових персонажів та зміни в оформленні.

Після виходу гри можливе виявлення досі невідомих помилок ПЗ, які не були виявлені тестувальниками. Так відбувається бо збій виникає за непердбачених обставин у грі, проте вважається зовсім незначним, або виникає тому, щоб встигнути до кінцевого терміну публікації гри, потрібно було прискорити розробку гри.

Такі помилки мають варіацію від незначних графічних до серйозних, останні мають можливість впливати на процес гри або навіть робити його неможливим. Зазвичай розробники розробляють деякі оновлення (патчі) для того щоб виправити помилки. Але бувають і такі помилки які виявляються корисними для користувачів, вони дозволяють гравцям обходити правила, що встановлені у грі.

Також користувачів може не влаштовувати якийсь із аспектів гри, наприклад складність. У цьому випадку розробки теж випускають оновлення із вдосконаленнями. Розробка та випуск оновлень для гри випускаються безкоштовно, проте можуть залучити нових гравців та втримати старих, що може збільшити можливі продажі, та позбавляє гру від недоліків.

Для того щоб підтримувати інтерес впродовж довгого часу для гри випускають деякі доповнення, що не лише виправляють помилки, а й додають різні можливості які можуть розвивати і доповнювати сюжет самої гри. Ці доповнення можуть бути різного типу, як платні, так і безкоштовні. В

основному вони розповсюджуються через мережу, і мають назву DLC (Downloadable content). У більшості випадків DLC є платними, хоч вони і не змінюють саму суть гри, проте надають деякі свіжі можливості для гравця

Висновки до розділу 3

Отже, у цьому розділі було створено та продемонстровано файлову структуру проекту, розглянуто можливі структури в цілому та детально описано даний проект, конкретно які файли було створено та де і яким чином ці файли зберігаються.

Було створено та описано етапи створення розширеного інтерфейсу Unity. Продемонстровано зовнішній вигляд цього інтерфейсу на рисунках, описано за допомогою яких засобів його було реалізовано та яким чином відбувається збереження цих даних.

Було дано визначення поняття база даних, та класифіковано її за типами. Також розглянуто поняття системи управління базами даних, проведено їх класифікацію і етапи роботи. Було наведено можливі для використання об’єкти у базі даних та розглянуто інші корисні інструменти у роботі з базами даних, такі як запити і форми.

Оскільки окрема база даних у даному проекті не знадобиться, було створено та використано у подальшому внутрішню базу даних у Unity, та продемонстровано їх.

Після цього, детально описано процес інтегрування розробленого додатку на операційну систему Android, та показано можливі додаткові налаштування проекту, фінальним етапом завершення цього процесу продемонстровано згенерований apk. файл.

Після того як готовий додаток уже встановлений на смартфон із ОС Android, відбувається опис процесу гри, де детально і покроково розписано які саме дії виконує користувач і відповідно які дії він бачить на своєму екрані.

ВИСНОВКИ

Уданій дипломній роботі було виконано аналітичний аналіз, розглянуто предметну область проекту, розглянуто можливі платформи та мови програмування для подальшої реалізації проекту, проаналізовано уже існуючі додатки в сфері ігор вікторин, як для ПК, так і для смартфонів, обрано подальший шлях створення та просування додатку саме в мобільному геймінгу, конкретно на ОС Android. Описано майбутній функціонал додатку для розробника і користувача та відповідно створено Use-case діаграми.

Було розглянуто на проаналізовано існуючі програми для створення мобільних додатків та обрано мову програмування на якій реалізовано проект. Ухвалено рішення використовувати середовище розробки Unity для розробки мобільної гри.Для написання ігрових скриптів було обрано мову програмування C#. Середовищем розробки скриптів було обрано MicrosoftVisualStudio. Також було створено та описано дизайн додатку.

Створено та описано розробку розширеного інтерфейсу в Unity, та створення внутрішньої бази даних. Описано етапи інтеграції гри та компілювання на операційну систему Android. Установлено цей додаток на смартфон та протестовано його можливості.

Отже, в результаті роботи було розроблено мобільний додаток, що задовольняє всім вимогам,та отримано нові знання з розробки програмних продуктів для мобільних систем.

СПИСКИ ЛІТЕРАТУРИ

1. Unity Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.unity3d.com/ru/current/ScriptReference/Editor.OnInspectorGUI.html

2. Unity Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.unity3d.com/ru/current/ScriptReference/Android.IPostGenerateGradleAndroidProject.html

3. Хабр планування оптимізації в Unity [Електронний ресурс] – Режим доступу до ресурсу:https://habr.com/ru/company/intel/blog/254353/

4. Робота с EventSystem в Unity [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/359106/

5. Dou [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/articles/games–testing–specifications/

6. Unity Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.unity3d.com/ru/current/ScriptReference/Animations.AnimatorController.html

7. Хабр Розширення функціоналу в Unity 3D [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/305544/

8. Allref Комп’ютерні ігри [Електронний ресурс] – Режим доступу до ресурсу:https://allref.com.ua/uk/skachaty/Kompyuterni_igri%7C_yih_rozvitok_ta_suchasni_realiyi

9. Голдстоун В. Unity Game Development Essentials. – UK: Packt Publishing Ltd,– 2009. – 316 с.

10. Студопедія [Електронний ресурс] – Режим доступу до ресурсу: https://studopedia.ru/19_316706_seredovishche–ta–mova–programuvannya.html

11. Hocking J. Unity in Action: Multiplatform Game Development in C# with Unity 5 1st Edition / Joe Hocking, – 2015. – 352 с.

12. Mayra F. An Introduction to Game Studies / Frans Mayra. – Thousand Oaks: SAGE Publications Ltd, – 2008. – 208 с.

13. Steven Goodwin. Polished Game Development: From First Steps to Final Release , – 2016. – 269 с.

14. Android. Програмування для професіоналів» / Б. Харді, Б. Філліпс, К. Стюарт, К. Марсикано. – 2016. – 300 с.

15. «Інтерфейс. Основи проектування взаємодії. 4-е вид.» / А.Купер, Р. Рейман, Д. Кронін, К. Носсел. – 2016. – 719 с.

16. Гриффітс Д. Head First. Програмування для Android/Д. Гриффітс. – 2016. – 704 с.

17. Learning C# Program with Unity 3D / Alex Okita. – 2014. – 690 с.

18. Unity 2D Game Development / Dave Calabrese. – 2014. – 126 с.

19. Learn Unity for 2D Game Development / Alan Thorn. – 2013. – 310 с.

20. Дизайн додатків [Електронний ресурс] – Режим доступу до ресурсу: https://vc.ru/design/163626-dizayn-mobilnyh-prilozheniy-polnyy-gayd-po-ux-ui

21. Основи двигуна Unity[Електронний ресурс] – Режим доступу до ресурсу:https://itproger.com/course/unity

22. Створення гри на Unity[Електронний ресурс] – Режим доступу до ресурсу: https://proglib.io/p/sozdaem-2d-igru-na-unity-instrukciya-dlya-novichka-2020-09-01

23. База даних та Unity[Електронний ресурс] – Режим доступу до ресурсу: https://www.cyberforum.ru/unity/thread1748834.html

24. Особливості тестування[Електронний ресурс] – Режим доступу до ресурсу: https://training.qatestlab.com/blog/technical-articles/gaming-on-pc-features/

25. Опис мови JavaScript[Електронний ресурс] – Режим доступу до ресурсу: https://htmlweb.ru/java/js.php

26. Gogotop [Електронний ресурс] – Режим доступу до ресурсу: https://gototop.net/seo–wikipedia/interfeis/

27. Етапи розробки комп’ютерних ігор [Електронний ресурс] – Режим доступу до ресурсу:https://blog.fugas.space/gamedev-stages/

28. Анімація [Електронний ресурс] – Режим доступу до ресурсу:https://vue.gov.ua/Анімація_(мультиплікація)

29. Мова програмування C# [Електронний ресурс] – Режим доступу до ресурсу:https://uk.wikipedia.org/wiki/C_Sharp

ДОДАТОК А. ЛІСТИНГ ВИХІДНИХ КОДІВ

Лістинг файлу QuizManager:

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

namespace AHG.QuizGame
{
publicclassQuizManager : MonoBehaviour
    {
#pragmawarningdisable CS0649
        [Header("Панельпроведеннявікторини")]
        [SerializeField] private GameObject questionPanel;
        [SerializeField] private GameObject questionImgObj;
        [SerializeField] private Text questionText;
        [SerializeField] private Text timerText;
        [SerializeField] private CanvasGroup answerBtnsGroup;
        [SerializeField] privateButton[] answerBtns;
        [SerializeField] private Image questionImage;
        [SerializeField] private GameObject questionImageFullPanel;
        [SerializeField] private Image questionImageFull;
        [SerializeField] private Text questionImgText;
        [Header("Панельправильностівідповіді")]
        [SerializeField] private Image answerPanelImg;
        [SerializeField] private Text answerStateText;
        [SerializeField] private Image answerStateImage;
        [SerializeField] private Text questionCounterText;
        [SerializeField] privateSprite[] answerStateIcons;
#pragmawarningrestore CS0649

privateconststring ANIM_SHOW_QUESTION = "ShowQuestion", ANIM_HIDE_QUESTION =
"HideQuestion", ANIM_SHOW_ANSWER = "ShowAnswer",
            ANIM_HIDE_ANSWER = "HideAnswer", ANIM_SHOW_FIMAGE = "ShowFullImage",
ANIM_HIDE_FIMAGE = "HideFullImage";
privatereadonlystring[] ANSWER_TYPES_TITLES = { "ВІрнавІдповІдь", "НеВІрнавІдповІдь",
"Часвийшов" };

private Category currentCategory;
private Question currentQuestion;

private Coroutine timerCoroutine;

private Queue<Question> questionsQueue = new Queue<Question>();

#region Singleton
publicstatic QuizManager Instance { get; privateset; }
privatevoidInitSingleton()
        {
if (Instance == null)
                Instance = this;
        }
#endregion

privatevoidAwake() {
InitSingleton();
}
```

```
publicvoidPlay(Category category)
        {
if (currentCategory != null) return;
            currentCategory = category;
GenerateQuestions();
        }

publicvoidToggleImageView(bool full)
        {
if (full) questionImageFull.sprite = currentQuestion.Image;
GameManager.Instance.PlayAnimation(full ? ANIM_SHOW_FIMAGE : ANIM_HIDE_FIMAGE, 3);
}

privatevoidGenerateQuestions()
        {
            List<Question> questions = new List<Question>(currentCategory.Questions);
for (int i = 0; i <currentCategory.Questions.Count; i++)
            {
int id = Random.Range(0, questions.Count);
                Question q = new Question(questions[id].Text, questions[id].Image,
newstring[3], -1);
int correctAnswerID = questions[id].CorrectAnswerID;
                List<string> answers = new List<string>(questions[id].Answers);
for (int a = 0; a <q.Answers.Length; a++)
                {
int answerID = Random.Range(0, answers.Count);
q.Answers[a] = answers[answerID];
if (q.CorrectAnswerID == -1)
                    {
if (answerID == correctAnswerID) q.CorrectAnswerID = a;
elseif (answerID < correctAnswerID) correctAnswerID--;
                    }
answers.RemoveAt(answerID);
                }
                questionsQueue.Enqueue(q);
questions.RemoveAt(id);
            }
ShowQuestion();
        }

privatevoidShowQuestion()
        {
            Time = currentCategory.AnswerTime;
            answerBtnsGroup.interactable = false;
if (questionsQueue.Count > 0)
            {
                currentQuestion = questionsQueue.Dequeue();
if (currentQuestion.Image != null) questionImage.sprite = currentQuestion.Image;
                questionImgObj.SetActive(currentQuestion.Image != null);
questionText.gameObject.SetActive(currentQuestion.Image == null);
                questionText.text = questionImgText.text = currentQuestion.Text;
for (int i = 0; i <currentQuestion.Answers.Length; i++)
                    answerBtns[i].GetComponentInChildren<Text>().text =
currentQuestion.Answers[i];
GameManager.Instance.PlayAnimation(ANIM_SHOW_QUESTION);
GameManager.Instance.WaitForSeconds(GameManager.Instance.GetAnimationLength(ANIM_SHOW_QUE
STION), () => {
                    answerBtnsGroup.interactable = true;
                    ToggleTimer(true);
});
            }
else
            {
```

```
ResetQuiz();
            }
        }

publicvoidSelectAnswer(int id) => StartCoroutine(
IESelectAnswer(Time > 0 && id == currentQuestion.CorrectAnswerID ? AnswerType.Correct :
AnswerType.Wrong));

private IEnumerator IESelectAnswer(AnswerType type)
        {
            ToggleTimer(false);
            answerBtnsGroup.interactable = false;
if (type != AnswerType.TimeOver) yieldreturnnew WaitForSeconds(1);
            answerStateImage.sprite = answerStateIcons[type == AnswerType.Correct ? 0 :
1];
GameManager.Instance.PlayAnimation(ANIM_HIDE_QUESTION);
if (questionImageFullPanel.activeSelf) ToggleImageView(false);
            answerStateText.text = ANSWER_TYPES_TITLES[(int)type];
int crntQuestionIndex = currentCategory.Questions.Count - questionsQueue.Count;
            questionCounterText.text =
$"{crntQuestionIndex}/{currentCategory.Questions.Count}";
            answerPanelImg.color = type == AnswerType.Correct ?Colors.Green :
Colors.LightRed;
GameManager.Instance.PlayAnimation(ANIM_SHOW_ANSWER, 2);
yieldreturnnewWaitForSeconds(GameManager.Instance.GetAnimationLength(ANIM_SHOW_ANSWER));
yieldreturnnewWaitForSeconds(2);
switch (type)
            {
case AnswerType.Wrong: WrongAnswer(); break;
case AnswerType.TimeOver: TimeOver(); break;
case AnswerType.Correct: CorrectAnswer(); break;
}
        }

privatevoidTimeOver()
        {
ResetQuiz();
        }

privatevoidWrongAnswer()
        {
ResetQuiz();
        }

privatevoidCorrectAnswer()
        {
GameManager.Instance.PlayAnimation(ANIM_HIDE_ANSWER, 2);
ShowQuestion();
        }

privatevoidResetQuiz()
        {
GameManager.Instance.PlayAnimation(ANIM_HIDE_ANSWER, 2);
GameManager.Instance.WaitForSeconds(GameManager.Instance.GetAnimationLength(ANIM_SHOW_ANS
WER), () => {
GameManager.Instance.ToggleHeader(false);
                questionsQueue.Clear();
                currentQuestion = null; currentCategory = null;
            });
        }

privatevoidToggleTimer(bool start)
        {
if (timerCoroutine != null)
```

```
            {
                StopCoroutine(timerCoroutine);
                timerCoroutine = null;
            }
if (start) timerCoroutine = StartCoroutine(IETimer());
        }

private IEnumerator IETimer()
        {
while (Time > 0)
            {
yieldreturnnewWaitForSeconds(1);
                Time--;
            }
StartCoroutine(IESelectAnswer(AnswerType.TimeOver));
        }

privateint _time;
privateint Time
        {
get{ return _time; }
set
            {
                _time = value;
                timerText.text = _time.ToString();
}
        }
    }
       }
```

Лістинг файлу Categories:

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

namespace AHG.QuizGame
{
    [Serializable]
publicclassCategory
    {
//Назвакатегорії

publicstring Name;

//Списокпитань

public List<Question> Questions;

///Часдлявідповідінапитання

publicint AnswerTime = 15;

publicCategory() {
Questions = new List<Question>();
        }
    }
       }
```

ЛістингфайлуQuestions:

```
using UnityEngine;
using System;

namespace AHG.QuizGame
{
    [Serializable]
publicclassQuestion
    {

        //Текстпитання

publicstringText;

// Картинка питання (не обов'язково)

publicSpriteImage;

//Списокваріантіввідповідей

publicstring[] Answers;

///Ідентифікаторвірноївідповіді

publicint CorrectAnswerID;

publicQuestion(string text, Sprite image, string[] answers, int correctAnswerID)
        {
            Text = text;
            Image = image;
            Answers = answers;
            CorrectAnswerID = correctAnswerID;
        }

publicQuestion() : this(null, null, newstring[3], 0) { }
}
       }
```