

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 122 «Комп'ютерні науки»
(освітня програма «Комп'ютерна графіка та розробка ігор»)
на тему:

«Розробка основних механік, музики та анімацій для
логічної гри «What are the ghosts talking about?»»

Виконав студент групи КН-19-1
РЯБЧУК Олександр Вікторович

Керівник роботи:
МАРЧУК Дмитро Костянтинович

Рецензент:
КУЗЬМЕНКО Олександр Вікторович

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ЗАТВЕРДЖУЮ

Завідувач кафедри комп'ютерних наук

Марина ГРАФ

«05» січня 2023 р.

ЗАВДАННЯ

на кваліфікаційну роботу освітнього ступеня «бакалавр»
за спеціальністю 122 «Комп'ютерні науки»
(освітня програма «Комп'ютерна графіка та розробка ігор»)

Здобувач вищої освіти: РЯБЧУК Олександр Вікторович

Керівник роботи: МАРЧУК Дмитро Костянтинович

Тема роботи: Розробка основних механік, музики та анімацій для логічної гри «What are the ghosts talking about?»

Затверджена Наказом університету від «05» січня 2023 р. № 10/с

Термін здачі студентом закінченої роботи: «10» червня 2023 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження): використання Adobe After Effects, Adobe Photoshop, Spine, Fruity Loops Studio, а також рушія Unity для створення анімації, візуальних ефектів та музичного супроводу для 2D логічної гри.

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Дата	
		Завдання видав	Завдання прийняв
1	Дмитро МАРЧУК	01.09.2022	01.11.2022
2	Дмитро МАРЧУК	01.11.2022	01.04.2023
3	Дмитро МАРЧУК	01.04.2023	01.05.2023

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Пошук, огляд та аналіз аналогічних розробок. Формулювання завдання. Опрацювання літературних джерел	25.09.22 – 10.11.22	Виконав
2	Проектування структури за допомогою діаграм UML	11.11.22 – 20.11.22	Виконав
3	Написання музичних лупів, створення анімацій і візуальних ефектів	21.11.22 – 20.03.23	Виконав
4	Тестування та оформлення пояснювальної записки	21.03.23 – 25.04.23	Виконав
5	Попередній захист	10.06.2023	Виконав

Здобувач вищої освіти Олександр РЯБЧУК
Керівник Дмитро МАРЧУК

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра на тему " Розробка основних механік, музики та анімацій для логічної гри "What are the ghosts talking about?" складається з пояснювальної записки, графічних матеріалів та програмного коду. Пояснювальна записка містить 62 сторінки, 48 ілюстрації та 3 таблиці.

У роботі описано процес написання музики, створення анімацій та візуальних ефектів, включаючи вибір кольорової палітри та створення елементів графіки, таких як персонажі, об'єкти та фони. Також проаналізовано різні методи створення анімацій та їх переваги та недоліки.

У роботі представлено опис створення анімацій, музики та основних механік включаючи підбір предметів, крафтинг а також пересування персонажу.

Крім того, у роботі наведено результати тестування створених анімацій та основних механік для визначення недоліків та їх виправлення.

КЛЮЧОВІ СЛОВА: АНІМАЦІЯ, ОСНОВНІ МЕХАНІКИ, МУЗИЧНИЙ СУПРОВІД, ЛОГІЧНА ГРА, ADOBE AFTER EFFECTS, SPINE, FRUITY LOOPS STUDIO.

					КН.КР.Б – 122 – 23 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Розробка основних механік, музики та анімацій для логічної гри «What are the ghosts talking about?» Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		О.В. РЯБЧУК					4	62
Керівник		Д.К. МАРЧУК				Житомирська політехніка, група КН-19-1		
Реценз.		О.В. КУЗЬМЕНКО						
Н. Контр.								
Затверд.		М.С. ГРАФ						

ABSTRACT

The bachelor's thesis entitled "Development of Core Mechanics, Music, and Animations for the Puzzle Game 'What are the ghosts talking about?'" consists of an explanatory report, graphical materials, and source code. The explanatory report comprises 62 pages, 48 illustrations, and 3 tables.

The thesis describes the process of composing music, creating animations, and visual effects, including the selection of color palettes and the development of graphical elements such as characters, objects, and backgrounds. Various animation techniques are analyzed, along with their advantages and disadvantages.

The work presents a detailed overview of the creation of animations, music, and core mechanics, including item selection, crafting, and character movement. Additionally, the thesis provides the results of testing conducted on the animations and core mechanics to identify any flaws and propose their remedies.

KEYWORDS: ANIMATION, CORE MECHANICS, MUSIC ACCOMPOTIMENT, PUZZLE GAME, ADOBE AFTER EFFECTS, SPINE, FRUITY LOOPS STUDIO.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ	9
1.1. Постановка задачі.....	9
1.2. Аналіз аналогів програмного продукту	9
1.3. Обґрунтування вибору інструментальних засобів для створення музичних та анімаційних елементів.....	12
1.4. Дизайн-документ	14
Висновки до першого розділу.....	18
РОЗДІЛ 2. РОЗРОБКА ОСНОВНИХ МЕХАНІК, СТВОРЕННЯ АНІМАЦІЙ ТА ВІЗУАЛЬНИХ ЕФЕКТІВ, НАПИСАННЯ МУЗИЧНОГО СУПРОВОДУ	19
2.1. Розробка основних механік за допомогою діаграм UML	19
2.2. Етапи реалізації основних механік	25
2.3. Анімація персонажів та об'єктів на локаціях	30
2.4. Додавання візуальних ефектів	35
2.5. Створення музичного супроводу	36
Висновки до другого розділу	37
РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ	38
3.1. Інтерфейс та порядок роботи з програмним додатком.....	38
3.2. Тестування роботи програмного забезпечення	48
Висновки до третього розділу.....	51
ВИСНОВКИ.....	52
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	53
ДОДАТКИ.....	56
Додаток А.....	57

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

UML (Unified Modeling Language) - уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування

UI – User Interface – інтерфейс, елементи, з якими взаємодіє користувач

NPC - Non-Player Character – персонажі, якими гравець не управляє, та з якими зазвичай можна взаємодіяти

ПЗ – програмне забезпечення

PC – Personal Computer (Персональний комп'ютер)

SRP - Unity's Scriptable Render Pipeline

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

В 21 столітті актуальність логічних ігор значно зростає. У нашому сучасному цифровому світі, де технології постійно розвиваються та впроваджуються в усі сфери життя, логічні ігри стають цінним ресурсом для розвитку критичного мислення, проблемного мислення та креативного розв'язання завдань.

Однією з причин актуальності логічних ігор є їх потужний вплив на розвиток когнітивних навичок. Вони сприяють розвитку логічного мислення, уяви, аналітичних та критичних навичок. Гравець повинен уявити ситуацію, зосередитися на деталях та знайти оптимальне рішення, що вимагає абстрактного мислення та творчого підходу.

Крім того, логічні ігри сприяють тренуванню пам'яті та концентрації. Гравець повинен запам'ятати різноманітні елементи гри, розв'язувати завдання в обмежені строки, а також зосереджуватися на розв'язанні проблеми в умовах динамічного геймплею.

Також важливо звернути увагу на музикальний супровід, візуальні ефекти та анімаційну складову логічних ігор, які повинні доповнювати картину та сконцентровувати увагу на рішенні логічних задач. Тому написання саундтреків, а також анімаційного фону з додаванням візуальних ефектів для логічної гри є важливим завданням, що потребує професійного підходу та знань у галузі саунд дизайну, монтажу та анімації.

Крім навчальної цінності, логічні ігри також забезпечують задоволення та розвагу. Вони дозволяють гравцеві насолоджуватися інтелектуальним викликом та досягненням успіху в складних завданнях. Це також стимулює суспільну взаємодію, оскільки деякі логічні ігри мають мультиплеєрний режим, де гравці можуть співпрацювати або змагатися між собою.

Усі ці фактори роблять логічні ігри незамінними для розвитку критичного мислення, вирішення проблем та забави.

Метою кваліфікаційної роботи є розробка основних механік, музики та анімацій для логічної гри «What are the ghosts talking about?».

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

використовуючи такий інструментарій: Adobe After Effects, Spine та Fruity Loops Studio, а також розробка основних механік за допомогою рушія Unity.

Об'єктом дослідження технології розробки механік, анімацій, проектування діаграм UML.

Предметом дослідження є розробка основних механік, музики та анімацій для логічної гри "What are the ghosts talking about ?", використання рушія Unity, а також таких програм як: Adobe After Effects, Spine –створення анімації та візуальних ефектів, Fruity Loops Studio – музичного супроводу.

В рамках теми кваліфікаційної роботи бакалавра було опубліковано тези:

Марчук Д.К., Рябчук О.В. Генерація ігрових світів в Unity. Тези доповідей IV Всеукраїнської науково-технічної конференції Комп'ютерні технології: інновації, проблеми, рішення, 18-20 листопада 2021 року. Житомир: «Житомирська політехніка», 2021. С.56-57.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

1.1. Постановка задачі

Метою кваліфікаційної роботи бакалавра є розробка основних механік, музики та анімацій для логічної гри "What are the ghosts talking about ?", що буде додавати атмосферності до геймплею та сприяти позитивному враженню від гри.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

1. Спроекувати основні механіки гри за допомогою діаграм UML.
 - 1.1. Створити діаграму варіантів використання.
 - 1.2. Створити діаграму активності.
 - 1.3. Створити діаграму класів.
2. Розробити код основних механік.
3. Створити відповідні анімації та візуальні ефекти.
 - 3.1. Анімація головного персонажа.
 - 3.2. Анімація другорядних персонажів.
 - 3.3. Анімація локацій.
 - 3.4. Монтаж та анімація катсцен.
 - 3.5. Анімація панелей меню та налаштувань.
4. Створити музичний супровід для відповідних локацій та катсцен.

1.2. Аналіз аналогів програмного продукту

Для аналізу було обрано ігри, спираючись, насамперед, на їх анімацію та музикальний супровід. Ось декілька з них:

«Beyond the Wall»(рис.1.1) - це гра класична пригода у жанрі adventure з чарівною атмосферою. Ви стаєте членом команди дослідників, які відправляються на експедицію, щоб розкрити таємниці та збирати ресурси. Ви вступаєте в бій зі злочинними угрупованнями, дикацтвами та іншими небезпечними істотами, які прагнуть влади та контролювати цей світ.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

Переваги:

- ідентичність гри: унікальний музичний стиль надають грі своєрідну особливість;
- покращений геймплей: анімаційні рухи і ефекти впливають на геймплей, роблячи його більш динамічним, цікавим та захоплюючим;
- емоційний вплив: музика викликає різні емоції у гравців, підсилюючи їх занурення в гру та підвищує інтенсивність геймплею.

Недоліки:

- недостатня якість звуку: музика має низьку якість запису і відтворення, це зменшує задоволення від гри;
- нестабільність анімації: проблеми зі швидкістю та рівномірністю анімаційних переходів здаються незручними і відволікають увагу гравців.



Рисунок 1.1 - Beyond the

Wall[[https://store.steampowered.com/app/2290010/Beyond the Wall/](https://store.steampowered.com/app/2290010/Beyond_the_Wall/)]

«Lost in Play»(рис.1.2) - це гра в жанрі головоломки. Гравець опиняється в ролі головного героя, який втрачений у віртуальному світі та повинен знайти шлях назад до реальності.

Переваги:

- візуальна привабливість: Добре виконана анімація надає грі привабливий вигляд та візуальну привабливість, привертаючи увагу гравців і створюючи естетичне задоволення;

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						10
Змн.	Арк.	№ докум.	Підпис	Дата		

– емоційний вплив: Музика викликає різні емоції у гравців, змушуючи їх переживати гру більш інтенсивно та залучено.

Недоліки:

– монотонність: музика повторюється без варіацій, це викликає втому та нудьгу у гравців.



Рисунок 1.2 - Lost in

Play[https://store.steampowered.com/app/1328840/Lost_in_Play/]

«There Is No Game: Wrong Dimension» - рис.1.3 - це ігра-головоломка, яка використовує метафори та гумор, щоб перетворити традиційні ігрові концепції.

Переваги:

– творчий підхід: Гра використовує унікальні техніки та стилі анімації, що надає їй особливий вигляд і відчуття оригінальності. Кожен рівень має свою власну анімаційну обробку, що робить гру барвистою та різноманітною;

– атмосфера та настрій: Музика відтворює різноманітні настрої в грі, допомагаючи створити відповідну атмосферу для кожного рівня або ситуації. Вона підсилює емоційні моменти та додає глибину до геймплею;

– інтерактивність: Анімація взаємодіє з гравцем та його рухами, що додає відчуття взаємодії та реалізму. Гравець може впливати на об'єкти в грі та спостерігати, як вони реагують згідно з анімаційними ефектами.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		11

Недоліки:

- ступінь відповідності: Деякі музичні композиції можуть не завжди повністю відповідати дії або настрою в грі, що може порушити цілісність та іммерсію геймплею;
- візуальні розбіжності: Інколи можуть виникати незбіги між анімацією та рештою графіки, що може створювати деяку дисгармонію в загальному вигляді гри.



Рисунок 1.3 - There Is No Game: Wrong

Dimension[https://store.steampowered.com/app/1240210/There_Is_No_Game_Wrong_Dimension/]

Отже, проаналізувавши вище зазначені ігри було вирішено створити не нав'язливі та прості анімації, відповідний до місцевості та сетингу музикальний супровід в грі «What are the ghosts talking about?», які будуть максимально підходити під стиль гри та допомагати користувачам сконцентруватися і спонукати насолоджуватися ігровою атмосферою.

1.3. Обґрунтування вибору інструментальних засобів для створення музичних та анімаційних елементів

При створенні анімаційних, візуальних ефектів та музичних елементів, завжди важливо обрати правильний інструментарій для досягнення бажаної

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		12

мети. В даному розділі будуть обгрунтовані вибрані для роботи з анімацією інструментальні засоби - Adobe After Effects, Spine. Для роботи з музичною частиною - Fruity Loops Studio.

Adobe After Effects(рис.1.4) - є професійною програмою для створення та редагування відео ефектів, анімації та композитінгу. Це потужне програмне забезпечення, яке використовується в галузі відеопродакшну, пост-продакшну, анімації та спеціальних ефектів.

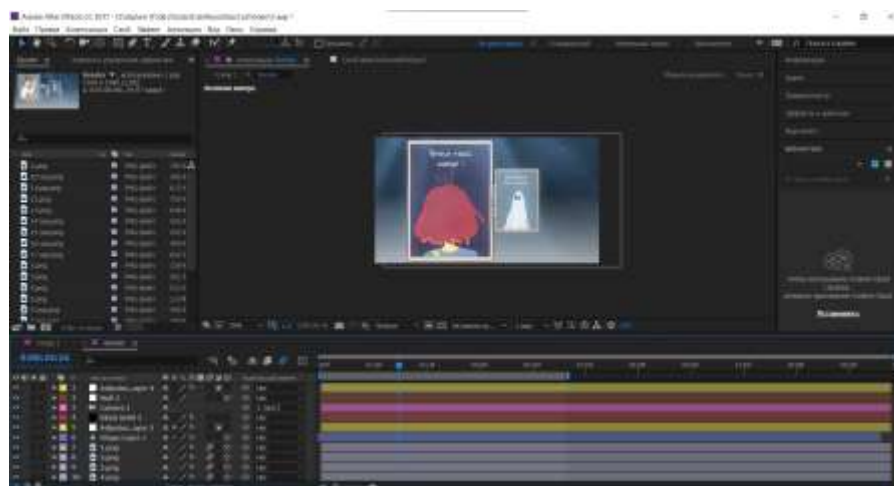


Рисунок 1.4 - Adobe After Effects

Spine (рис.1.5) є програмним інструментом для розробки 2D анімації з векторних зображень. Це потужне рішення, спрямоване на полегшення процесу створення анімаційних персонажів та їх рухів. Spine надає розширені функції, що дозволяють аніматорам та розробникам створювати живу, реалістичну та плавну анімацію для різних видів проектів, включаючи відеоігри, мультфільми, веб-сайти та інтерактивні додатки.



Рисунок 1.5 – Spine

Fruity Loops Studio (рис.1.6) є комплексною програмою для музичного створення і продюсування. Вона використовується професіоналами і новачками в галузі музики для створення різних жанрів, від електронної та хіп-хоп музики до популярних та фільмових саундтреків. FL Studio пропонує широкий набір інструментів, ефектів і можливостей для творчого процесу.



Рисунок 1.6 – Fruity Loops Studio

Отже, для створення музичних та анімаційних елементів було вирішено використовувати такі програми: Adobe After Effects, Spine, Fruity Loops Studio, оскільки вони є еталоном для створення музики та анімації, широко використовуються професіоналами.

1.4. Дизайн-документ

Дизайн-документ - це детальний план або специфікація, що описує дизайн або архітектуру певного продукту, проекту або системи. Це документ, який зазвичай створюється перед розробкою або реалізацією проекту, і він використовується для передачі інформації команді розробників, дизайнерам, менеджерам і будь-яким зацікавленим сторонам.

Вступ

Головний персонаж гри отримує запрошення на тематичну вечірку до Хелловіну. Прийшовши на місце проведення вечірки, головна героїня потрапляє в пастку зачарованого будинку. Потрапивши до однієї з кімнат, вона знайомиться з привидами, власником будинку, який потребує її допомоги.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		14

Жанр і аудиторія

- демографія: гра орієнтована на широку аудиторію, мінімальний вік гравця – 7 років;
- жанр: головоломка;
- психотип - мислителі. Концентруються на вирішенні завдань, головоломок та прийнятті викликів, що висуваються грою;
- гравці даної гри – мідкорні.

Основні особливості гри

Ключові особливості гри (USP):

- унікальний візуальний стиль, графіка, яка дозволить гравцям відволіктись від проблем повсякденності;
- харизматичний привид-помічник;
- цікавий та інтригуючий сюжет;
- різноманітність головоломок.

Час, за який можна пройти один рівень гри залежить від складності головоломки.

Опис гри

Основна задача гравця – проходження ігрового сюжету, розгадуючи голово-ломки різних рівнів складності. Ігровий процес полягає розгадуванні багаторівне-вих головоломок.

Платформа

Гра буде розроблятися тільки під PC-платформу. Системні вимоги для PC наведено у таблиці 1.1.

Таблиця 1.1

Системні вимоги для PC

Вимоги	Мінімальні	Рекомендовані
Операційна система	Microsoft® Windows® 7 або новіше	
Процесор	3 тактовою частотою 1,5 ГГц або аналогічна або краще	
Оперативна пам'ять	4Гб	8Гб

Жорсткий диск	ЗГб вільного місця	5Гб
DirectX	DirectX® 9.0c	
Управління	Мишка, клавіатура	

Суть ігрового процесу

Суть ігрового процесу полягає в управлінні головним героєм. Перонажем можна виконувати маніпуляції для проходження гри: підбір предметів, пересування, поєднання предметів. Також можна спілкуватися з привидами та іншими створіннями, що існують в грі.

Також, якщо виникають складності в проходженні гри можна використати підказку, яка полегшить рішення головоломки.

Хід гри та сюжет

Події гри відбуваються в різноманітних кімнатах будинку.

Початок – катсцена (головна героїня отримує запрошення та приходить на вечірку).

Дівчина-художниця (далі Фріда) отримує дивне запрошення на вечірку, але коли вона потрапляє до будинку, то помічає, що вона в пастці. Усі двері та вікна зачинені. На даному етапі відбувається знайомство Фріди та привида-помічника, що з'являється у кімнаті. Дієго, так звати привида, розповідає про правила гри та вказує на першу головоломку.

Але чому привид вирішує допомагати дівчині? Відповідь дуже проста – даний персонаж – художник в минулому і привид тепер, намагається розшукати клаптики картини, на якій була зображена його покійна дружина. Злі духи, які потрапили до будинку з його смертю, розкидали шматки картини по різних кімнатах та Дієго не має змоги зібрати їх до купи. Ця картина була і є для нього дуже цінною, тому він звертається за допомогою саме до Фріди, майбутньої художниці. Він вважає, що тільки вона здатна йому допомогти.

Кінець – фінальна катсцена.

В грі наявний успішний фінал.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

Фізична модель

Основні фізичні закони: усе підлягає гравітації.

Гра буде використовувати вбудовані в Unity фізичні рушії.

Наприклад, для персонажів буде застосовано Animation Controller, що дозволить “оживити” їх.

Персонаж гравця

Основним персонажем гравця є Фріда, яка виглядає як звичайна людина. Має інвентар у вигляді сумки художника.

Доступні команди в грі наведені у таблиці 1.2.

Таблиця 1.2

Доступні команди

Команда	Параметри (ціль)
Йти в точку	Точка на місцевості
Залишатись на місці	-
Підбір предметів	Предмети на локації

Персонаж Фріди може знаходитись в наступних станах:

- нерухомий;
- рухається.

Персонажі гри

Другим головним персонажем гри є Дієго – привид-помічник, з яким, здебільшого, буде контактувати головна героїня. Власне в грі, привид з’являється ще під час початкової катсцени, де гравець знайомиться з ним та його історією. Після того, як Фріда виходить з будинку – Дієго стає вільним і також покидає будинок.

Загальна характеристика: помічник.

Дієго може лише говорити.

Другорядними персонажами є: скелет, домовичок.

Зі скелетом можна пограти в гру, а ось домовичка потрібно нагодувати,

щоб мати змогу перейти до іношої кімнати.

Музичний супровід

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						17
Змн.	Арк.	№ докум.	Підпис	Дата		

У грі майже постійно на фоні буде грати музика, переважно спокійна, яка створює атмосферу загадковості та фантастичності.

Передбачаються музичні теми, обмежені виконанням місій і складаються з фрагментів. Фрагменти зациклені і можуть об'єднуватися для створення безперервного не одноманітного музичного фону. Середня тривалість кожного фрагмента - 1 хвилина.

Анімаційні вставки

У грі присутні такі анімаційні вставки:

- початкова катсцена;
- фінальна катсцена.

Висновки до першого розділу

Були визначені основні цілі та завдання проекту. Чітко сформульовані цілі дозволяють уточнити напрямок роботи і забезпечити його успішне виконання.

Були досліджені існуючі аналоги, які мають схожі функціональні можливості. Цей аналіз дозволив отримати інформацію про сильні та слабкі сторони існуючих рішень та визначити недоліки, які можна врахувати при розробці програмного продукту.

Було проведено аналіз доступних інструментів та технологій для створення музичних та анімаційних елементів. Були враховані функціональні вимоги проекту, а також особливості та можливості кожного з інструментів. На основі цього аналізу був обрано найбільш оптимальні засоби, які задовольняють потреби проекту.

Також було розроблено дизайн-документ, який допоможе при розробці даної гри, адже містить детальний план та специфікацію, що описує дизайн та вимоги до даного проекту.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. РОЗРОБКА ОСНОВНИХ МЕХАНІК, СТВОРЕННЯ АНІМАЦІЙ ТА ВІЗУАЛЬНИХ ЕФЕКТІВ, НАПИСАННЯ МУЗИЧНОГО СУПРОВОДУ

2.1. Розробка основних механік за допомогою діаграм UML

Діаграма діяльності – ще одна важлива діаграма UML, що описує динамічні аспекти системи.

Діаграма дій - це, по суті, блок-схема, що представляє потік від однієї дії до іншої. Діяльність може бути описана як робота системи.

На рисунку 2.1 зображена діаграма активності при виконанні користувачем ходу. Виконання ходу - це дія користувача. Початкова дія користувача – “Клік” або “Переміщення по локації” . Якщо спочатку користувач обирає “Переміщення по локації” то наступною дією буде “Клік”, так як від цього залежить визначення наявності предмета та відкриття головоломки. Якщо після “Кліку” немає відгука оточення, то далі буде слідувати “Дія користувача” і так поки відгук оточення не стане наявним. Якщо ж після “Кліку” наявний відгук оточення, далі йде визначення наявності предмета. Якщо предмет наявний, відбувається підняття предмета. Наступним кроком є визначення наявності місця в інвентарі. Місце наявне – оновлюється інвентар і відкривається інтерфейс головоломки. Якщо ж предмет не наявний відбувається відкриття головоломки, так само і при відсутності місця в інвентарі. В інтерфейсі головоломки гравець визначає чи потрібна йому підказка в вирішенні, якщо не потрібна – гравець вирішує головоломку своїми силами. Якщо підказка потрібна то: визначається наявна кількість підказок, так як кількість підказок вичерпна. Якщо кількість підказок більше нуля – підказка доступна, далі запускається діалог з оновленням фрази помічника, після оновлення фрази діалог активується і гравець повертається до розв’язання головоломки. Якщо кількість підказок менше нуля – гравець повертається до розв’язання головоломки самостійно. Після розв’язання головоломки рівень закінчується.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		19

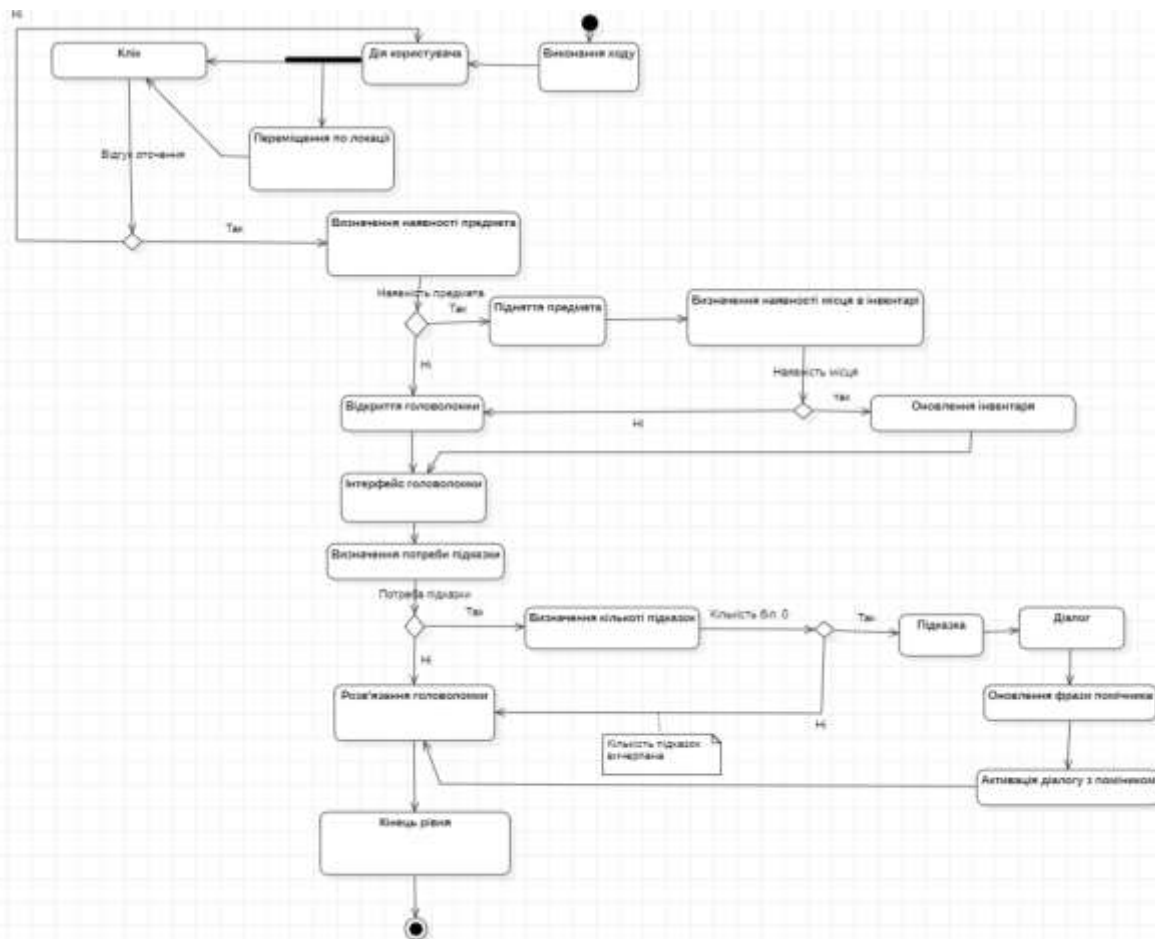


Рисунок 2.1 – Діаграма активності при виборі пункту «Виконання ходу»

На рисунку 2.2 зображена діаграма активності при виборі користувачем пункту «Налаштування». Якщо гравець вирішує змінити налаштування - відбувається збереження. В системі програми здійснюється пошук файлів збереження. Відбувається збереження, після збереження налаштувань вони відобразяться на сцені.



Рисунок 2.2 – Діаграма активності при виборі пункту «Налаштування»

На рисунку 2.3 зображена діаграма активності при виборі користувачем пункту «Вихід в меню». При виборі цього пункту відбувається закриття ігрової сцени, а далі відображається сцена меню.



Рисунок 2.3 – Діаграма активності при виборі пункту «Вихід в меню»

Діаграма класів – це UML-діаграма, яка описує систему, візуалізуючи різні типи об'єктів усередині системи та види статичних зв'язків, що існують між ними. Вона також ілюструє операції та атрибути класів.

Зазвичай вони використовуються для вивчення концепцій галузі, розуміння вимог до програмного забезпечення та опису детальних проектів.

Спираючись на дану інформацію було побудовано діаграму класів, що зображена на рисунку 2.4.

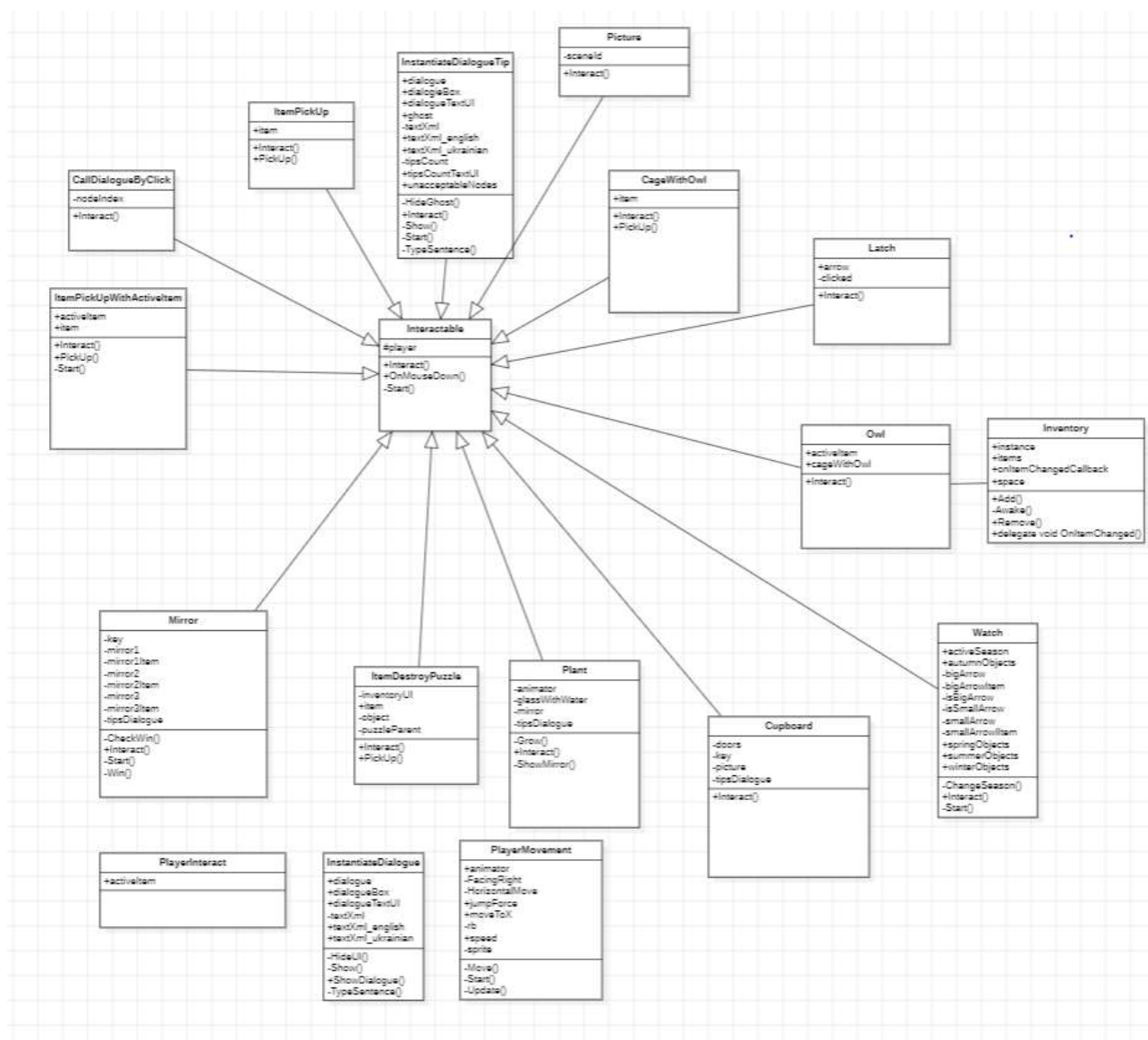


Рисунок 2.4 - Діаграма класів

Опис основних класів, а також їх полів та методів:

- Клас Interactable – абстрактний клас, який має шаблон для виконання методу після натиску на об’єкт, задає основу для методу interact/
- Клас ItemPickUp – клас, відповідає за підбір речі зі сцени в інвентар персонажа
- Клас itemDestroyPuzzle – клас, який відповідає за предмет зі сцени, при цьому закриває головоломку.
- Клас PlayerMovement – клас, що відповідає за переміщення

персонажа

- Клас `PlayerInteract` – клас, що зберігає активний предмет в інвентарі, який потрібний для підбору іншого предмета або для взаємодії.
- Клас `CallDialogueByClick` – клас, що відповідає за об'єкт, при кліку на який визивається діалог.
- Клас `InstantiateDialogue` - клас для відображення потрібної фрази в діалозі.

Варто зазначити, що у проектуванні було використано патерн. Шаблони проектування — це оптимізовані рішення розв'язання проблем програмування, що допускають багаторазове використання, з якими ми стикаємося щодня. Шаблон проектування – це не клас чи бібліотека, які ми можемо просто вставити до нашої системи. Він набагато більше. Це деякий шаблон, який має бути реалізований у належній ситуації.

В іграх, зазвичай, патерни використовуються для створення ворогів з дещо різною поведінкою, але однаковою основою. Так як дана гра не має нічого з вищевказаного, а головоломки не можна назвати однотипними, в них не можна виділити певну основну ідею, яку можна було б спростити за допомогою патерна, було вирішено використати шаблонний метод у класі `Interactable`.

Шаблонний метод — поведінковий патерн проектування, який визначає загальну структуру алгоритму в базовому класі, дозволяючи підкласам перевизначати певні кроки цього алгоритму без зміни його загальної структури.

Для початку певний клас треба зробити абстрактними. З цієї причини усі підкласи повинні будуть реалізувати кожен з кроків по-своєму. Таким чином, є основний метод `Interact()`, який викликається по кліку мишею на об'єкт з цим класом, але сам клас `Interactable` не можна використовувати сам по собі, його можуть використовувати лише класи, що його наслідують, і в них даний метод може бути перевизначений.

З діаграми класів можна виконати кодогенерацію, після чого отримаємо файли класів спроектованої гри. Кодогенерація – це процес автоматичного перетворення діаграм UML у код програми.

Для генерації коду програми в StarUML потрібно завантажити розширення для потрібної мови програмування, в нашому випадку – C#.

Для цього можна перейти у StarUML – Tools – Extension Manager, і встановити розширення C#. Далі треба обрати Tools – C# – Generate code (рис.2.5), після чого обрати модель, та обрати папку, до якої збережуться згенеровані класи (рис.2.6).

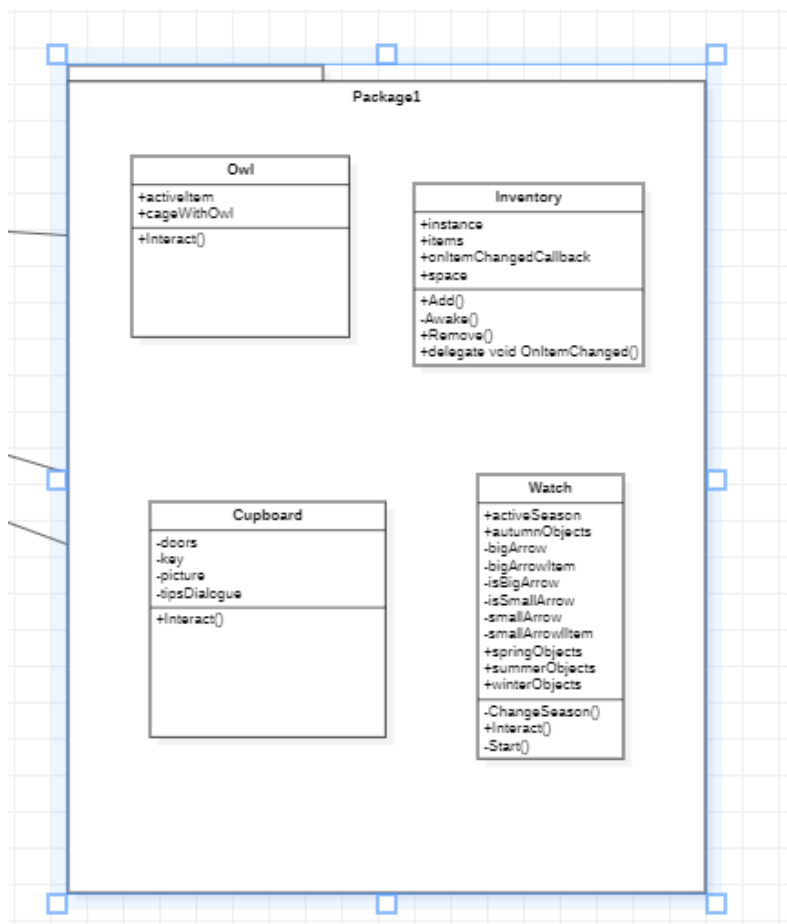


Рисунок 2.5 – Генерація коду

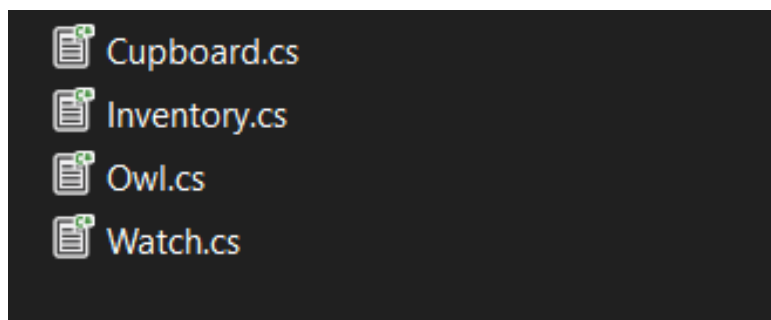
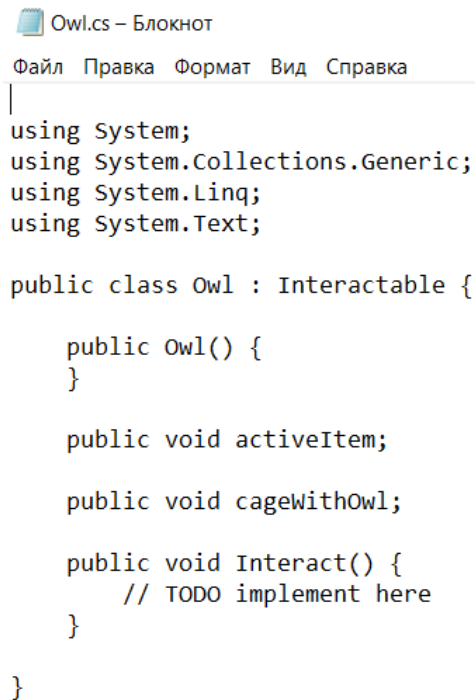


Рисунок 2.6 – Згенеровані класи

Згенеровані класи мають лише заготовки методів та декілька полів, приклад можна побачити на рис.2.7.



```
Owl.cs – Блокнот
Файл  Правка  Формат  Вид  Справка
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

public class Owl : Interactable {

    public Owl() {
    }

    public void activeItem;

    public void cageWithOwl;

    public void Interact() {
        // TODO implement here
    }

}
```

Рисунок 2.7 – Згенерований клас Owl

2.2. Етапи реалізації основних механік

Для початку було розроблено клас ItemPickUp, який потрібен саме для того, щоб додавати відповідний предмет до інвентарю. Найголовніший метод, який містить даний клас має назву PickUp:

```
public void PickUp()
{
    Debug.Log("Picking up item" + item.name);
    bool wasPickedUp = Inventory.instance.Add(item); //add to inventory
    if(wasPickedUp)
        Destroy(gameObject);
}
```

Даний метод додає предмет до інвентарю та виводить відповідне повідомлення в консолі для перевірки того чи дійсно предмет додався до інвентарю. Також, якщо відповідний предмет було підібрано – він видаляється з головної сцени.

Наступним було реалізовано клас Item, який відповідає за предмет, що вже знаходиться в інвентарі. Метод Use відповідає за використання предмету:

```
public virtual void Use()
{
    // Use the item
    Debug.Log("Using " + name);
    //craft new item if it is possible
    if(PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem ==
itemComponent && itemComponent != null)
    {
        Craft(this, itemComponent, itemResult);
    }
    //else just make this item active
    else
        PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem = this;
    InventoryUI.instance.UpdateUI();
}
```

Якщо ж два предмети можна поєднати, то викликається інший метод під назвою Craft:

```
public void Craft(Item itemComponent1, Item itemComponent2, Item itemResult)
{
    Inventory.instance.Remove(itemComponent1);
    Inventory.instance.Remove(itemComponent2);
    Inventory.instance.Add(itemResult);
    PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem = itemResult;
}
```

Якщо ж предмети не поєднуються, то іконка інвентарю стає активною та користувач матиме змогу використати даний предмет.

Було реалізовано клас DragAndDrop, який надає базовий функціонал для перетягування об'єктів він містить такі методи:

1. Start(): Цей метод викликається при старті гри і ініціалізує початкові значення. В даному випадку, він отримує посилання на компонент Rigidbody2D, який використовується для переміщення об'єкта.

```
protected virtual void Start()
```

```
{
    rb = GetComponent<Rigidbody2D>();
}
```

2. **GetMouseWorldPosition():** Цей метод повертає позицію курсора миші у світових координатах. Він використовує функцію **Camera.main.ScreenToWorldPoint()**, щоб перетворити позицію курсора миші з екранних координат у світові координати.

```
protected Vector3 GetMouseWorldPosition()
{
    return Camera.main.ScreenToWorldPoint(Input.mousePosition);
}
```

3. **OnMouseDown():** Цей метод викликається, коли натиснута кнопка миші на об'єкті. Він обчислює зсув між позицією об'єкта і позицією курсора миші у світових координатах і зберігає його у змінній **mousePositionOffset**. Це дозволяє коректно переміщувати об'єкт разом з курсором миші.

```
protected virtual void OnMouseDown()
{
    mousePositionOffset = transform.position - GetMouseWorldPosition();
}
```

4. **OnMouseDrag():** Цей метод викликається, коли об'єкт перетягують з натиснутою кнопкою миші. Він обчислює нову позицію об'єкта на основі позиції курсора миші та зсуву **mousePositionOffset** і застосовує метод **Rigidbody2D.MovePosition()**, щоб перемістити об'єкт у встановлену позицію.

```
protected virtual void OnMouseDrag()
{
    Vector2 position = GetMouseWorldPosition() + mousePositionOffset;
    rb.MovePosition(position);
}
```

Також було реалізовано клас **ItemPickUpWithActiveItem**, який успадковує клас **Interactable** і розширює його функціонал для забирання предметів з використанням активного предмета гравця. Він використовує інвентар гравця (об'єкт **Inventory.instance**) для додавання та видалення предметів з інвентаря. Він містить такі методи:

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		27

1. **Start():** Цей метод викликається при старті гри і ініціалізує початкові значення. В даному випадку, він отримує посилання на головного гравця (player) за допомогою `PlayerManager.instance.player.transform`.

```
private void Start()
{
    player = PlayerManager.instance.player.transform;
}
```

2. **Interact():** Цей метод викликається при взаємодії з об'єктом, зазвичай викликається гравцем. Він перевіряє, чи активний предмет гравця (`player.GetComponent<PlayerInteract>().activeItem`) відповідає `activeItem` цього об'єкту. Якщо так, то викликається метод `PickUp()` та видаляється активний предмет з інвентаря гравця (`Inventory.instance.Remove(activeItem)`).

```
public override void Interact()
{
    if (player.GetComponent<PlayerInteract>().activeItem == activeItem)
    {
        PickUp();
        Inventory.instance.Remove(activeItem);
    }
}
```

3. **PickUp():** Цей метод викликається для забирання предмета. Він додає предмет до інвентаря гравця за допомогою `Inventory.instance.Add(item)`. Якщо предмет було успішно додано, то об'єкт, до якого він прикріплений, знищується за допомогою `Destroy(gameObject)`.

```
public void PickUp()
{
    Debug.Log("Picking up item" + item.name);
    bool wasPickedUp = Inventory.instance.Add(item); //add to inventory
    if (wasPickedUp)
        Destroy(gameObject);
}
```

Для створення механіки руху гравця було реалізовано клас `PlayerMovement`, в якому наявні такі методи:

1. Start(): Цей метод викликається при старті гри і ініціалізує початкові значення. В даному випадку, він отримує посилання на компонент SpriteRenderer, який використовується для зміни напрямку спрайту гравця:

```
private void Start()
{
    //rb = GetComponent<Rigidbody2D>();
    sprite = GetComponentInChildren<SpriteRenderer>();
}
```

2. Update(): Цей метод викликається на кожній кадрі гри і відповідає за оновлення руху гравця. Він перевіряє, чи натиснута кнопка "Horizontal" (горизонтальний рух) і викликає метод Move(). Встановлюється значення HorizontalMove на основі вхідного значення осі "Horizontal" помноженого на швидкість. Також оновлюється параметр "HorizontalMove" в аніматорі, щоб відтворити відповідну анімацію залежно від значення швидкості.

```
private void Update()
{
    //movement
    if (Input.GetButton("Horizontal"))
        Move();
    HorizontalMove = Input.GetAxisRaw("Horizontal") * speed;
    animator.SetFloat("HorizontalMove", Mathf.Abs(HorizontalMove));
}
```

3. Move(): Цей метод відповідає за рух гравця. Він отримує вхідне значення осі "Horizontal" і вираховує вектор напрямку руху гравця. Потім застосовується метод Vector3.MoveTowards(), щоб плавно перемістити гравця в напрямку руху зі швидкістю, визначеною змінною speed. Крім того, метод також змінює орієнтацію спрайту гравця, щоб він обертався у відповідному напрямку руху.

```
private void Move()
{
    Vector3 dir = transform.right * Input.GetAxis("Horizontal");
    transform.position = Vector3.MoveTowards(transform.position, transform.position + dir,
    speed * Time.deltaTime);
    //flip sprite to face movement direction
}
```

```

        sprite.flipX = dir.x < 0.0f;
    }

```

Також було реалізовано клас ShowObjectWithItemClick, який успадковує клас Interactable і розширює його функціонал для показу певного об'єкту після кліку на певний предмет гравцем. Він використовує інвентар гравця (об'єкт Inventory.instance) для видалення предмета з інвентаря. Він містить такі метод:

Interact(): Цей метод викликається при взаємодії з об'єктом, зазвичай викликається гравцем. Він спочатку викликає базовий метод Interact() з батьківського класу Interactable. Далі перевіряє, чи активний предмет гравця (PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem) відповідає item цього об'єкту. Якщо так, то встановлюється активність (SetActive(true)) об'єкту, який потрібно показати (objectToShow), та видаляється активний предмет з інвентаря гравця (Inventory.instance.Remove(item)).

```

public override void Interact()
{
    base.Interact();
    if (PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem == item)
    {
        objectToShow.SetActive(true);
        Inventory.instance.Remove(item);
    }
}

```

2.3. Анімація персонажів та об'єктів на локаціях

В грі присутні такі анімації персонажів як: анімація головної героїні, скелета, круп'є, домовичка та чортиків. Була використана скелетна та покадрова анімація. Скелетна анімація була створена в програмі Spine, а покадрова – в Adobe After Effects.

Скелетна анімація - анімація, яка є методом анімації персонажів та об'єктів, в якому рух та деформацію відтворюють за допомогою скелетної структури. Вона базується на розбитті моделі на окремі кістки (бони) та

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		30

встановленні зв'язків між ними. Кожна кістка має свою позицію та обмеження на рух, які визначаються розробником анімації.

Спочатку було розроблено анімацію головної героїні, а саме – стан спокою(рис.2.8) та стан руху(рис.2.9).



Рисунок 2.8 – Скелетна анімація спокою головної героїні



					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		31

Рисунок 2.9 – Скелетна анімація руху головної героїні

Наступною була реалізована анімація скелета, а саме – стан спокою(рис.2.10).



Рисунок 2.10 – Скелетна анімація спокою скелета

Також було реалізовано анімацію додаткових персонажів, а саме – чортика №1(рис.2.11), чортика №2(рис.2.12), чортика №3(рис.2.13) та домовичка(рис.2.14) в різних станах.

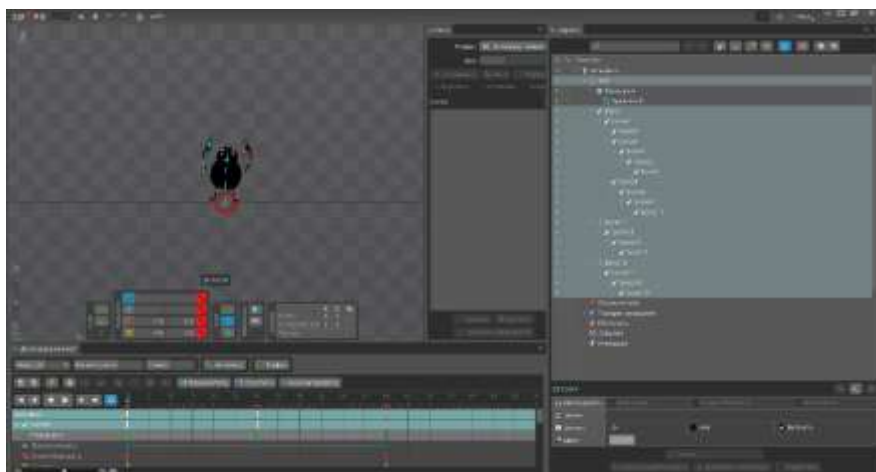


Рисунок 2.11 – Скелетна анімація чортика №1

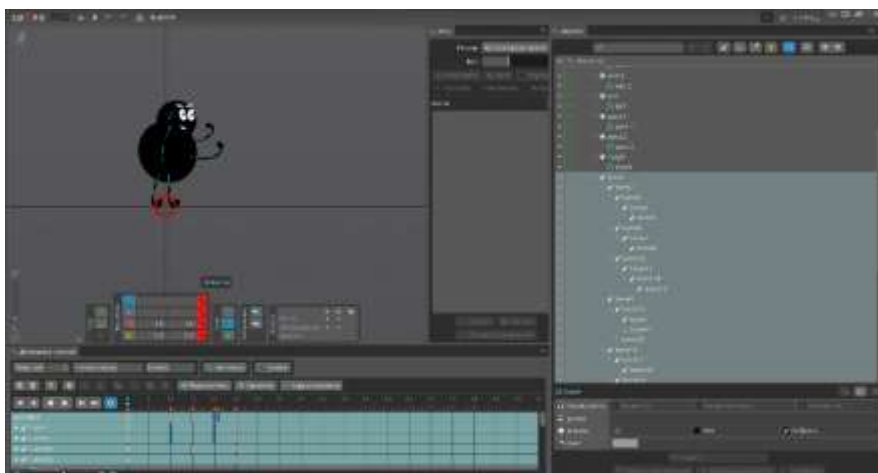


Рисунок 2.12 – Скелетна анімація чортика №2



Рисунок 2.13 – Скелетна анімація чортика №3

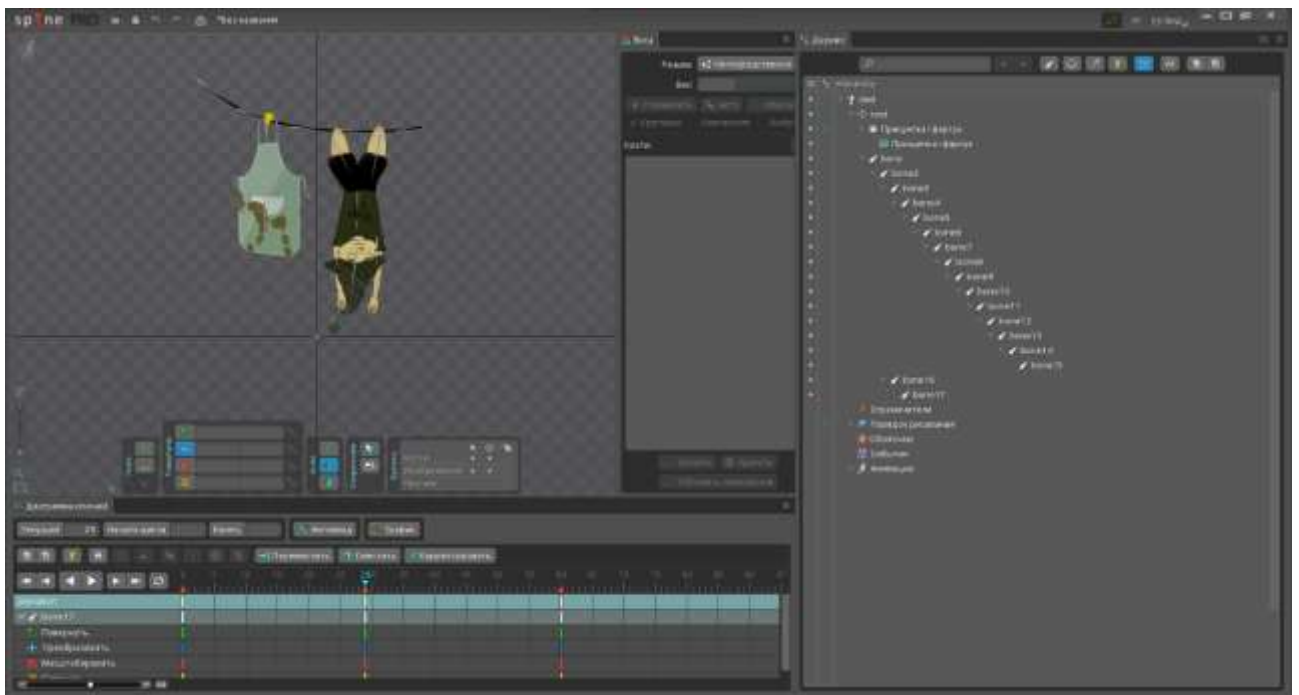


Рисунок 2.14 – Скелетна анімація домовичка

Також було реалізовано покадрову анімацію для круп'є(рис.2.15). Покадрова анімація є одним з методів створення анімації, де кожний кадр анімації створюється окремо та послідовно відтворюється зі зміною кадру з одного на інший. В цьому типі анімації кожен кадр представляє собою окремий зображений кадр, який утворює рух, коли його послідовно відтворюють з певною швидкістю.

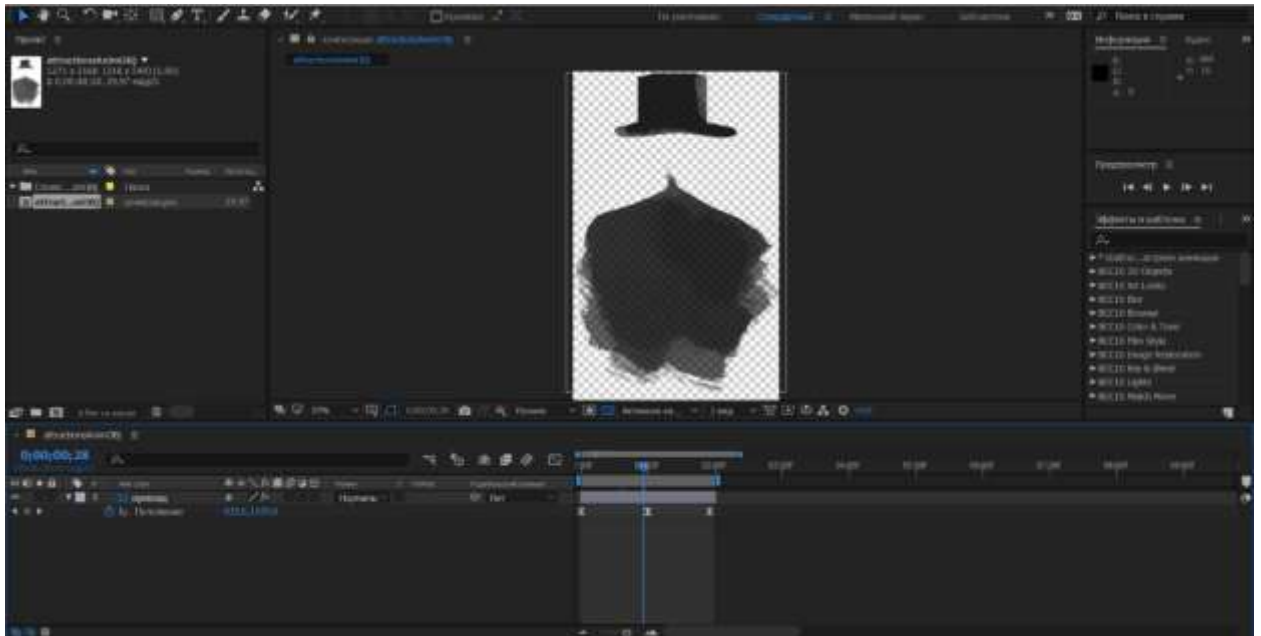


Рисунок 2.15 – Покадрова анімація круп'є

Також було реалізовано анімацію предметів на локаціях. Наприклад, анімація вогню свічки(рис.2.16), яка є покадровою.

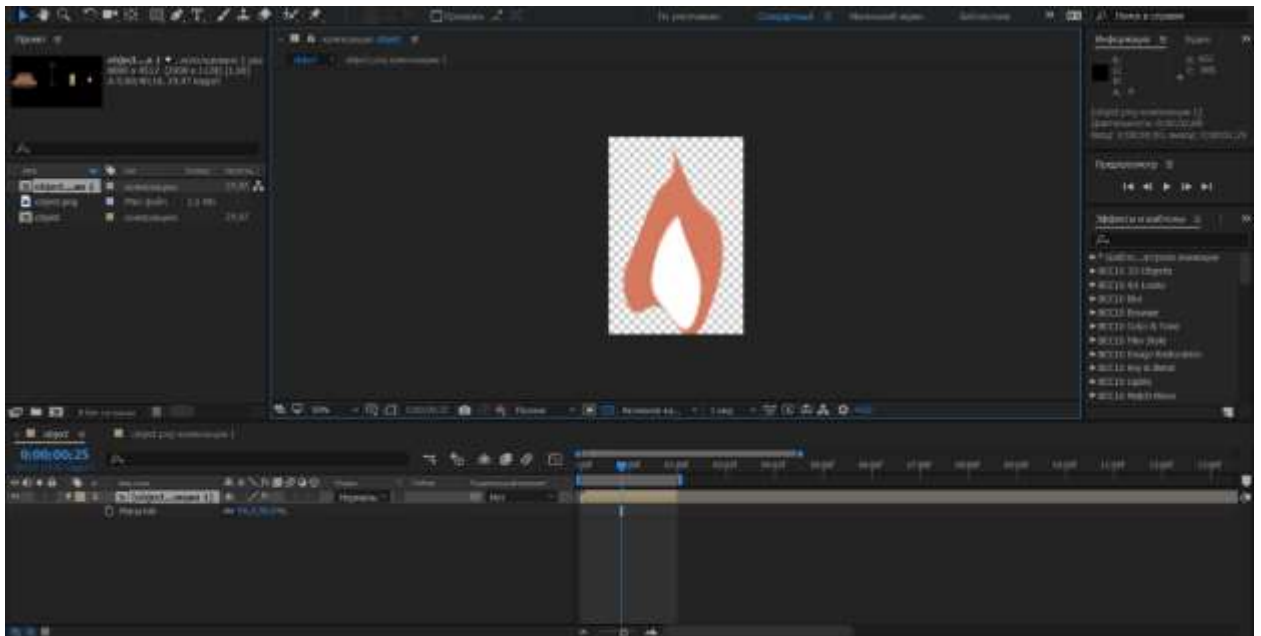


Рисунок 2.16 – Покадрова анімація вогню свічки

Було використано і сам рушій Unity як середовище для анімації об'єктів. Так, наприклад, було анімовано підсвітку предметів(рис.2.17). Unity надає потужні засоби для створення та управління анімаціями різних об'єктів, включаючи персонажів, об'єкти оточення, інтерфейсні елементи та інше. Для створення анімацій в Unity можна використовувати різні підходи, такі як ключові кадри, рухові криві та механіка фізики.



Рисунок 2.17 – Анімація підсвітки гірлянди

2.4. Додавання візуальних ефектів

Було реалізовано світлові ефекти за допомогою рушія Unity, а саме – використано плагін Universal RP. Це одна з графічних конвеєрних технологій, яку надає Unity. Вона є частиною SRP і надає широкі можливості для реалізації візуальних ефектів та оптимізації графічного рендерингу в проектах Unity.

Для початку було додано джерела світла та налаштовано відповідно до локації(рис.2.18).



Рисунок 2.18 – Додавання ефекту світла до локації

2.5. Створення музичного супроводу

Для початку було створено головну мелодію, яка відображається при запуску гри (на головному та інших панелях меню). Дана мелодія(рис.2.19) повністю передає атмосферу осені та Хелловіну, а також буде запам'ятовуватись користувач та асоціюватись саме з грою «What are the ghost talking about?».



Рисунок 2.19 – Створення музичного супроводу для панелей меню

Для кожної локації було створено окремий музичний супровід, який повністю передає атмосферу кімнати та є не нав'язливим.

Наприклад, для локації «Кухня» було створено мелодію(рис.2.20) за допомогою басової лінії та зацикленого лупу. Додатково використовувався патерн хейхетів. Для атмосферності було додано ефекти кімнати, ділею та панорами.



Рисунок 2.20 – Створення музичного супроводу для локації «Кухня»

					КН.КР.Б – 122 – 23 – ПЗ	Арк. 36
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до другого розділу

Було використано діаграми UML для аналізу та проектування основних механік гри. Це дозволило зрозуміти структуру гри, визначити взаємозв'язки між класами та функціональність різних компонентів.

Було описано послідовність кроків, необхідних для реалізації основних механік гри. Це включало створення необхідних класів, реалізацію логіки та взаємодії між компонентами, а також тестування та налагодження.

Було використано спеціалізовані інструменти, такі як Spine або Unity's Animation Editor, Adobe After Effects для створення покадрової та скелетної анімації та реалізації різних рухів та дій об'єктів.

Розглянуто додавання різноманітних візуальних ефектів для покращення графічного враження гри. Це включало створення та налаштування шейдерів, використання технік для створення ефектів світла та інших аспектів гри.

Було розглянуто створення музичних композицій для різних подій та ситуацій у грі з використанням спеціалізованих звукових редакторів та інструментів.

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
						37
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ

3.1. Інтерфейс та порядок роботи з програмним додатком

Представлено дії гравця на 3 локаціях («Кімната зі світлом», «Кімната з телескопом», «Музична студія»).

Опинившись на локації під назвою «Кімната зі світлом»(рис.3.1) користувач може взаємодіяти з такими предметами як: лампа, свічка, коробка, банка та ліхтарики на стіні.

Під час взаємодії користувача з коробкою на користувача очікує головоломка з світлячком(рис.3.2), під час якої необхідно обмежити перебування світлячка темно-синіми кільцями так, щоб він протاپив до пастки.



Рисунок 3.1 – Локація «Кімната зі світлом»

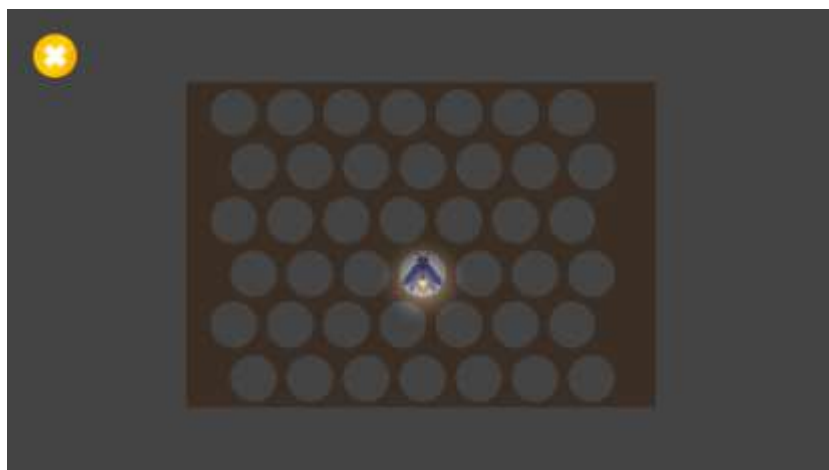


Рисунок 3.2 – Головоломка зі світлячком

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		38

Після того як головоломка зі світлячком буде розгадана, в інвентарі користувача з'явиться дана комаха, яку потрібно помістити до банки, що знаходиться на шафі(рис.3.3).

Також можна запалити свічку, скориставшись сірниками з попередньої локації(рис. 3.4). Таким чином гравець освічує усю кімнату та йому легше орієнтуватись в просторі.



Рисунок 3.3 – Світлячок в банці



Рисунок 3.4 – Запалення свічки

Якщо ж користувач натисне на ліхтарики, що вісять на стіні – перед ним постане нова головоломка(рис.3.5), в якій потрібно з'єднати кола з однаковим кольором.

При натисканні на велику лампу відкриється новий інтерфейс, в якому необхідно ввімкнути штепсель від лампи в розетку(рис.3.6).

Після виконання усіх цих дій з'явиться шматок картини та гравець зможе перейти до наступної локації, а також уся кімната буде повністю освічена(рис.3.7).

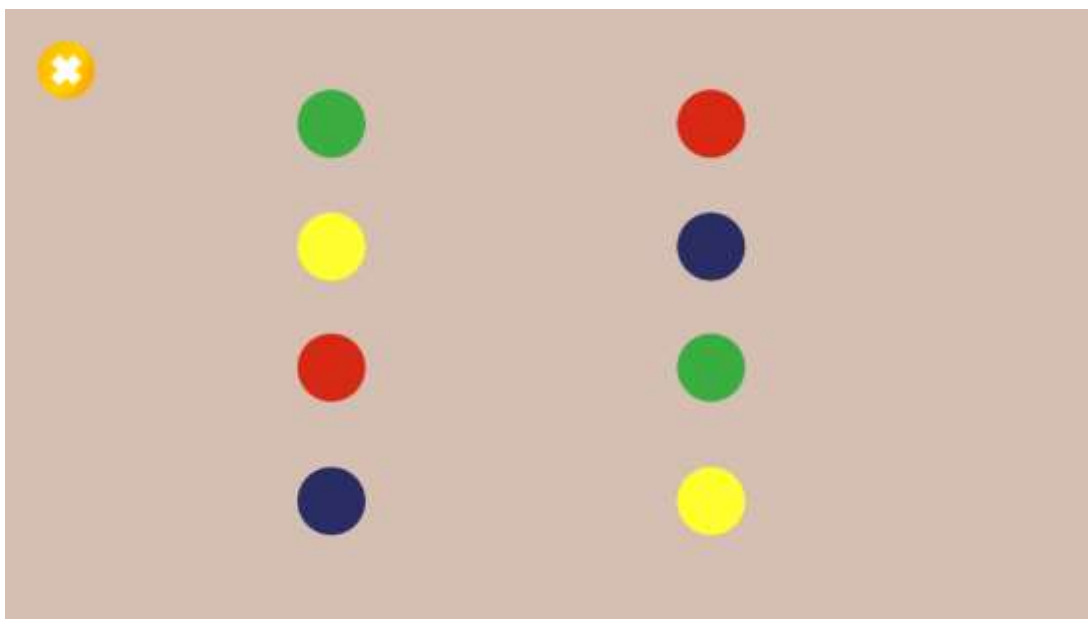


Рисунок 3.5 – Головоломка з кольорами

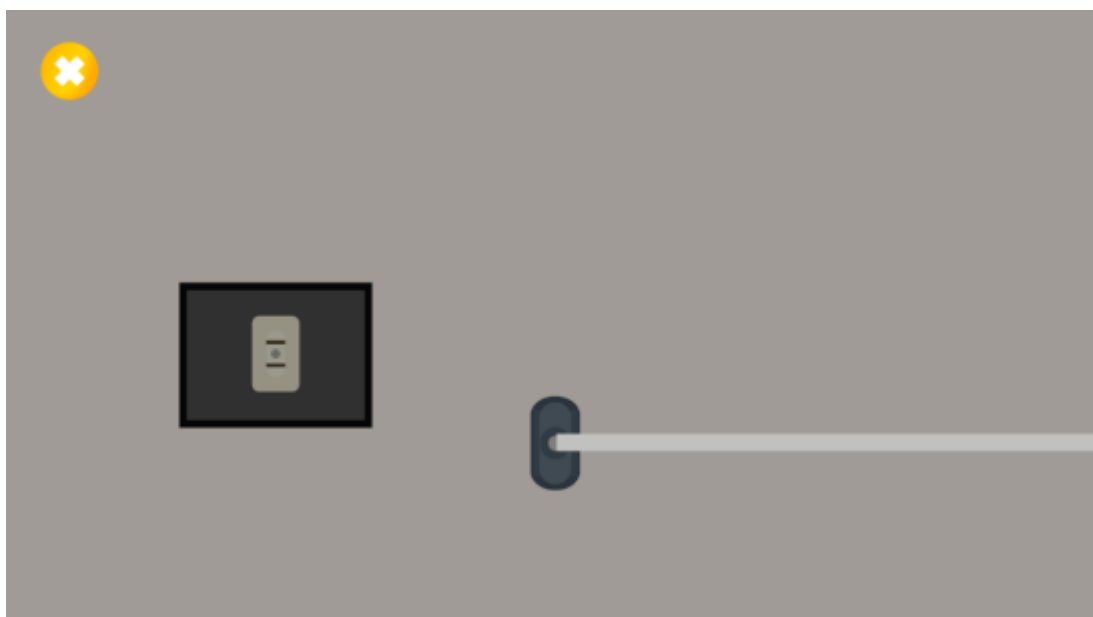


Рисунок 3.6 – Розетка і штепсель



Рисунок 3.7 – Вигляд кімнати після проходження усіх головоломок

Також під час проходження даної локації гравець може скористатись підказками від привида(рис.3.8), а також повзаємодіяти з іншими предметами(рис.3.9).



Рисунок 3.8 – Отримання підказки від привида



Рисунок 3.9 – Взаємодія з фурнітурою

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		41

Наступною локацією є «Кімната з телескопом»(рис.3.10). На даній локації користувач може взаємодіяти з такими предметами як: телескоп, книги, плакати, сейф, настільні ігри.



Рисунок 3.10 – Локація «Кімната з телескопом»

Під час взаємодії користувача з телескопом з'являється головоломка з зірками(рис.3.11), в якій необхідно об'єднати зірки за номерами.

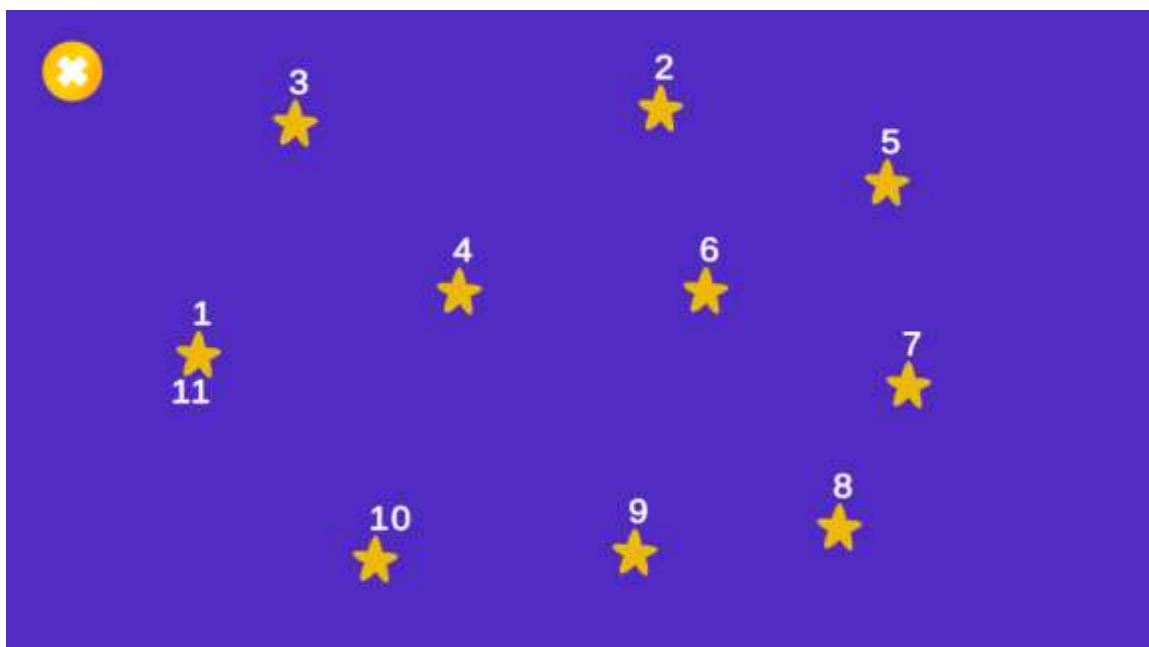


Рисунок 3.11 – Гооловоломка з зірками

Якщо ж користувач натисне на книги, що знаходяться на полиці – відкриється головоломка(рис.3.12), в якій необхідно виставити книги так, щоб лінія на них була неперервною.



Рисунок 3.12 – Головоломка з книгами

При натисканні на плакат, що вісить на стіні, на користувача очікує sudoku(рис.3.13). Цифри необхідно вводити з клавіатури.

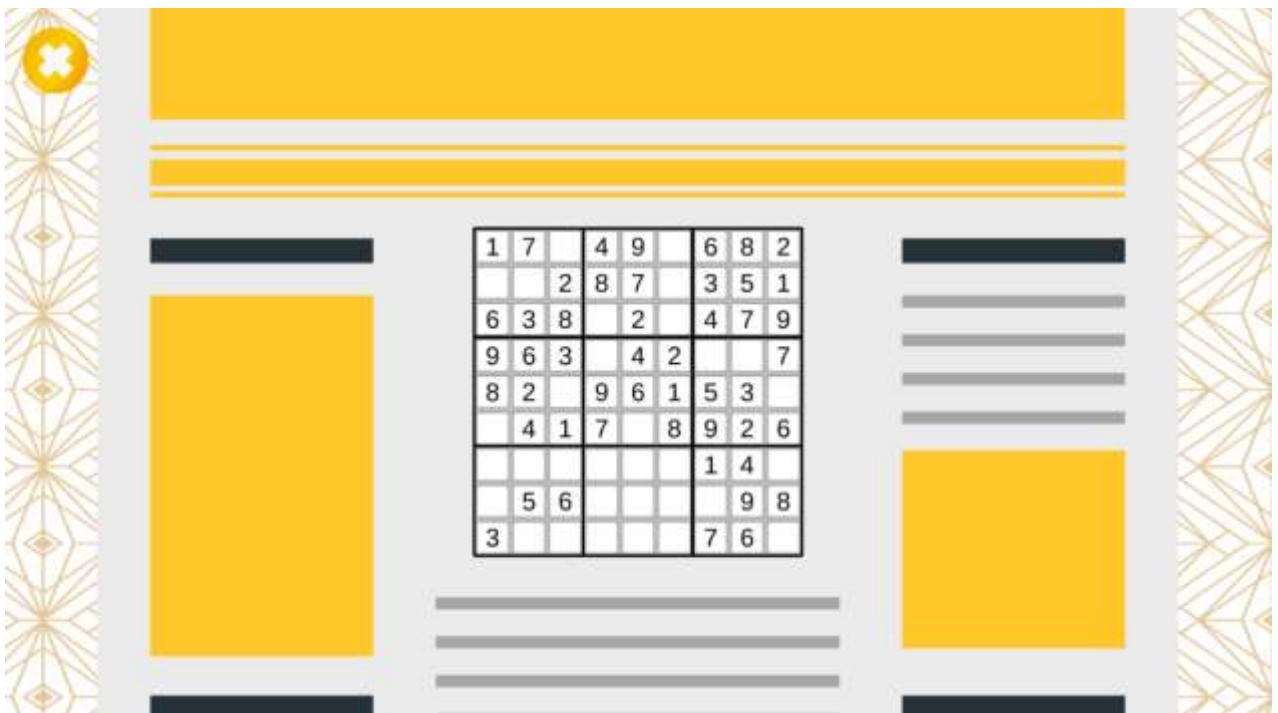


Рисунок 3.13 – Судоку

Також наявна головоломка з настільними іграми, а саме - пазлами(рис.3.14). Усі головоломки на локації підсвічуються, тому її буде легко знайти.

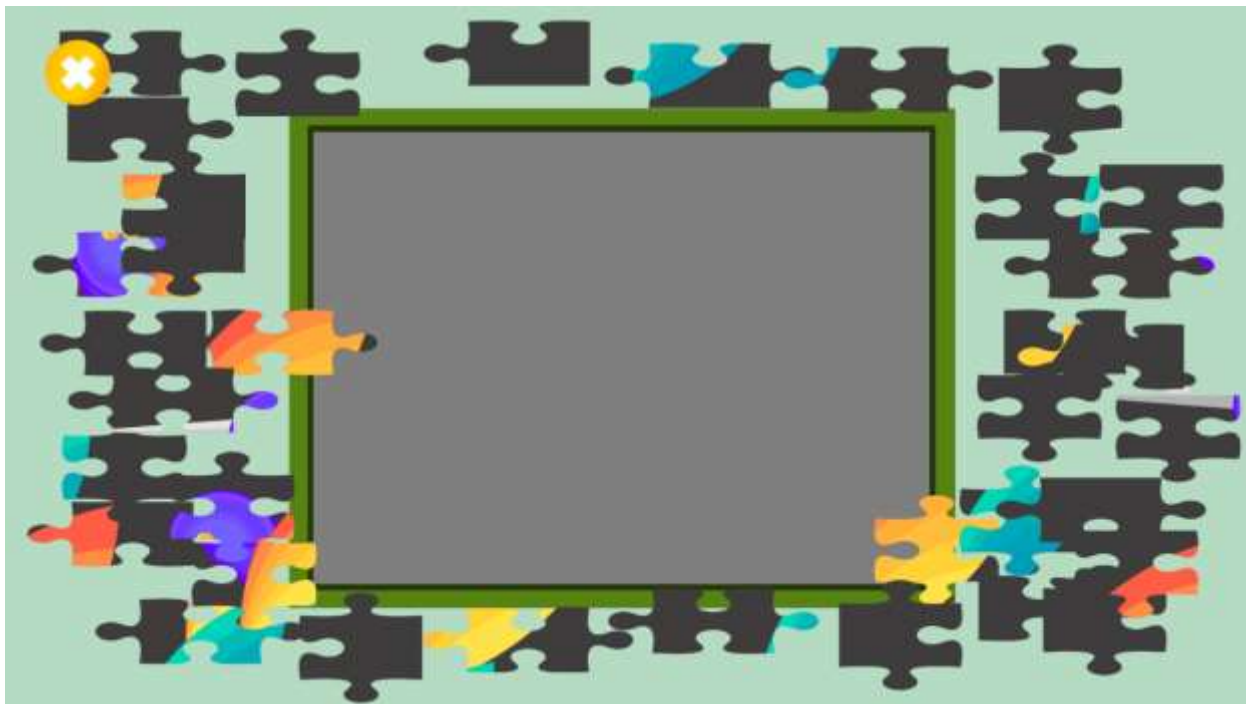


Рисунок 3.14 – Пазли

Після успішного проходження кожної головоломки з'явиться число та його позиція в сейфі(рис.3.15). Це робиться для того, щоб отримати шматок картини.

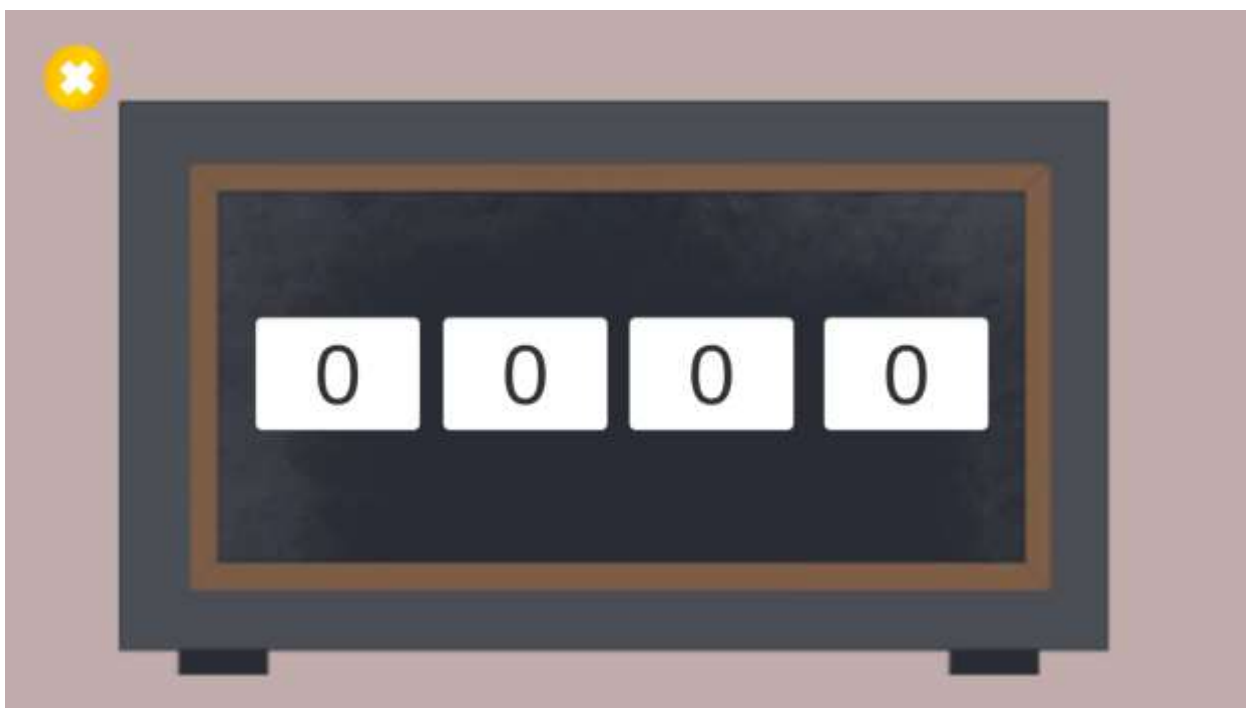


Рисунок 3.15 – Сейф

Також на даній локації можна взаємодіяти і з іншими предметами(рис.3.16).



Рисунок 3.16 – Взаємодія з книжною полицею

Наступною локацією є «Музична студія»(рис.3.17). На даній локації можна взаємодіяти з такими предметами як: календар на стіні, музична скринька, музичний програвач, комп'ютер.



Рисунок 3.17– Локація «Музична студія»

Під час взаємодії з календарем(рис.3.18), користувач отримає необхідну інформацію для проходження головоломки з балериною(рис.3.19).

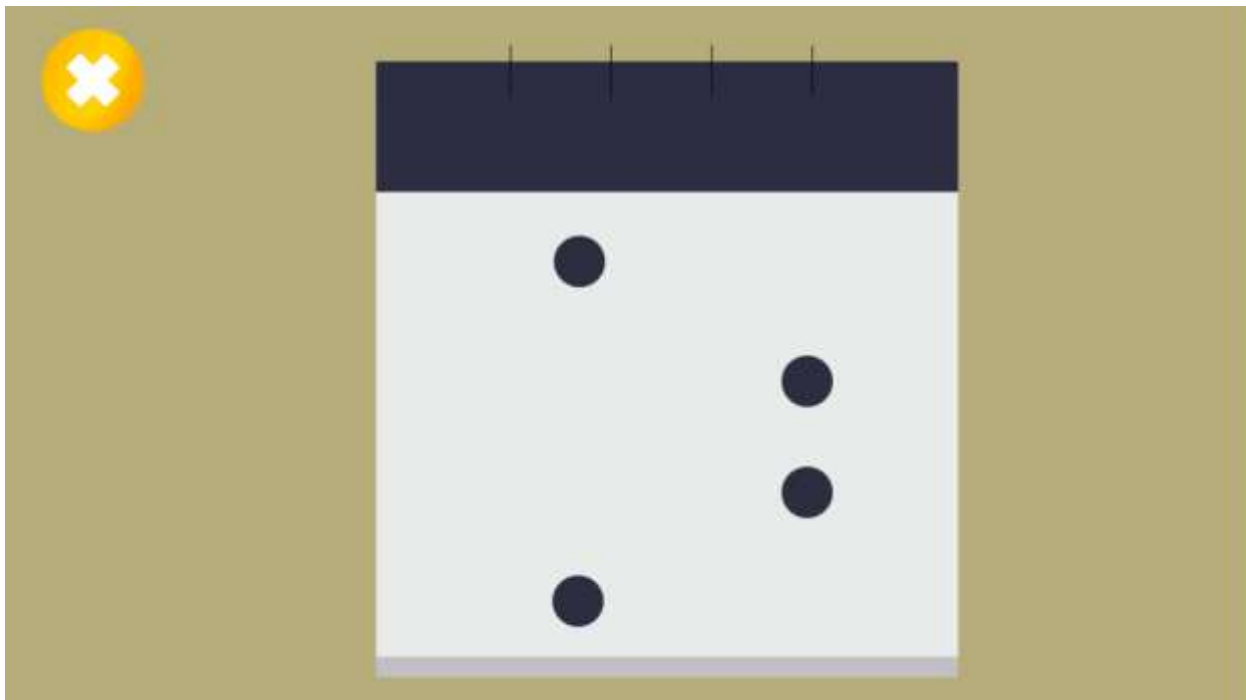


Рисунок 3.18– Взаємодія з календарем



Рисунок 3.19– Головоломка з балериною

Під час взаємодії користувача з музичним програвачем відкривається нова головоломка(рис.3.20), в якій необхідно витягнути касету, пересуваючи кубики.

Якщо ж користувач настикає на комп'ютер, то вмикається спектрум(рис.3.21) і починає відтворюватись музика.

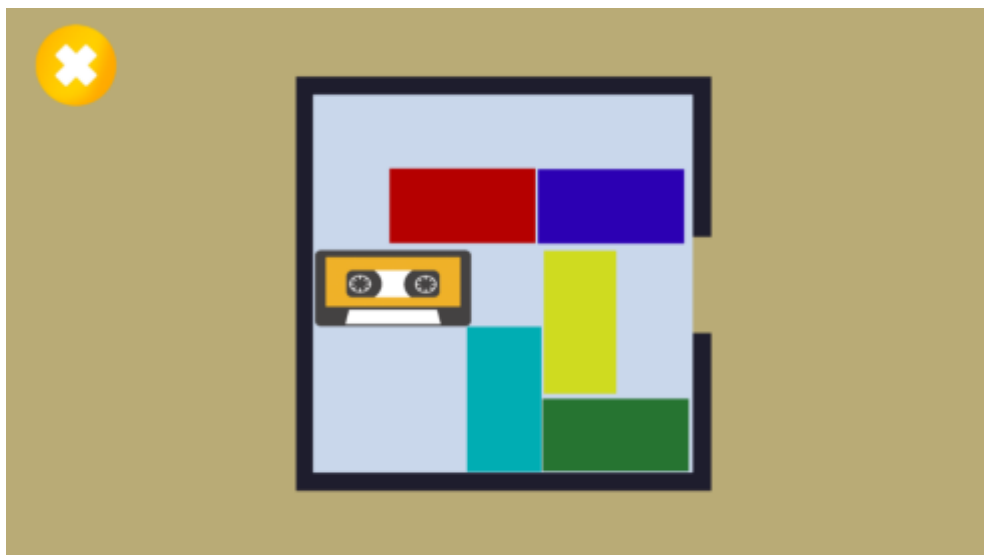


Рисунок 3.20– Головоломка з музичним програвачем



Рисунок 3.21– Спекtrum

Також наявна взаємодія і з іншими предметами(рис.3.22).



Рисунок 3.22– Взаємодія з музичними пластинками

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		47

3.2. Тестування роботи програмного забезпечення

Проведено тестування трьох локацій – «Кімната зі світлом», «Кімната з телескопом», «Музична студія» для виявлення та усунення недоліків. Представлено пункти, які було протестовано – таблиця 3.1.

Таблиця 3.1

Локації	Вхідні дані	Очікувані дані	Спостережені дані
Локація «Кімната зі світлом»	Натиск кнопки «Esc»	Відкривається меню паузи	Як і очікувалося
	Натиск на кнопку підказок	З'являється привид, який каже підказку	Як і очікувалося
	Клік на предмет, який можна підібрати	Предмет додається до інвентаря, інтерфейс оновлюється	Як і очікувалося
	Клік на предмет, який відкриває головоломку	Відкривається інтерфейс головоломки	Як і очікувалося
	Клік на предмет, який викликає фразу головної героїні	З'являється інтерфейс діалогу, де пишеться фраза, і після деякого часу зникає	Як і очікувалося
Локація «Кімната зі світлом», головоломка «Велика лампа»	Клік на кнопку з хрестиком	Головоломка закривається	Як і очікувалося
	З'єднання штепселю з розеткою	В лампі вмикається світло	Як і очікувалося
Локація «Кімната зі світлом», головоломка «Ліхтарики на стіні»	Клік на кнопку з хрестиком	Головоломка закривається	Як і очікувалося
	З'єднання кіл відповідно до кольорів	Після з'єднання кіл в ліхтариках вмикається світло	Як і очікувалося
Локація «Кімната зі світлом»	Клік на кнопку з хрестиком	Головоломка закривається	Як і очікувалося

світлом», головоломка «Спіймай світлячка»	Клік на відповідне коло для обмеження радіусу пересування світлячка	Після успішного проходження головоломки, світлячок з'являється в інвентарі	Як і очікувалося
Локація «Кімната з телекоспом»	Натиск кнопки «Esc»	Відкривається меню паузи	Як і очікувалося
	Натиск на кнопку підказок	З'являється привид, який каже підказку	Як і очікувалося
	Клік на предмет, який можна підібрати	Предмет додається до інвентаря, інтерфейс оновлюється	Як і очікувалося
	Клік на предмет, який відкриває головоломку	Відкривається інтерфейс головоломки	Як і очікувалося
	Клік на предмет, який викликає фразу головної героїні	З'являється інтерфейс діалогу, де пишеться фраза, і після деякого часу зникає	Як і очікувалося
Локація «Кімната з телекоспом», головоломка з книгами	Сортування книг в правильному порядку	Книги успішно відсортовані та отримано правильну позицію і число шифру	Як і очікувалося
Локація «Кімната з телекоспом», головоломка з зірками	Утримуючи клік лівою кнопкою миші зірки послідовно з'єднуються	Зірки послідовно з'єднані, отримано правильну позицію і число шифру	Як і очікувалося
Локація «Кімната з телекоспом», головоломка з пазлами	Пересування пазлів відповідно до їх позицій	Пазли складаються в єдну картину, отримано правильну позицію і число шифру	Як і очікувалося
Локація «Кімната з телекоспом», головоломка «Судоку»,	Введення цифр в відповідні комірки	Цифри знаходяться на правильних позиціях, отримано правильну позицію і число шифру	Як і очікувалося

Локація «Музична студія»	Натиск кнопки «Esc»	Відкривається меню паузи	Як і очікувалося
	Натиск на кнопку підказок	З'являється привид, який каже підказку	Як і очікувалося
	Клік на предмет, який можна підібрати	Предмет додається до інвентаря, інтерфейс оновлюється	Як і очікувалося
	Клік на предмет, який відкриває головоломку	Відкривається інтерфейс головоломки	Як і очікувалося
	Клік на предмет, який викликає фразу головної героїні	З'являється інтерфейс діалогу, де пишеться фраза, і після деякого часу зникає	Як і очікувалося
	Натиск кнопки «Esc»	Відкривається меню паузи	Як і очікувалося
Локація «Музична студія», головоломка з музичною скринькою	По кліку зафарбовуються відповідні кільця	Кільця зафарбовані в правильному порядку, з'являється нота	Як і очікувалося
Локація «Музична студія», головоломка з музичним програвачем	Переміщення кубиків в відповідні позиції для створення маршруту касети	Маршрут створено, головоломка успішно розгадана	Як і очікувалося
Локація «Музична студія», взаємодія з патифоном	Переміщення голки до патифону	Отримання ноти	Як і очікувалося
Локація «Музична студія», взаємодія з календарем	Клік на календар	Відображення правильної послідовності шифру для головомки з музичною скринькою	Як і очікувалося

Локація «Музична студія», взаємодія з комп'ютером	Клік на ком'ютер	Відображення спектру та відтворення аудіо	Як і очікувалося
Локація «Музична студія», взаємодія з касетним програвачем	Переміщення касети до програвача	Отримання ноти	Як і очікувалося
Локація «Музична студія», взаємодія з картинами на стіні	Переміщення нот до відповідних картини	Отримання частинки картини	Як і очікувалося

При тестуванні додатку було виявлено низку проблем: некоректно відображались джерела світла, анімація фону відображалась поверх персонажа, елементи інвентарю зникали, якщо їхня кількість була зavelika. Це було виправлено за допомогою правильного розміщення об'єктів в просторі та накладання слоїв.

Висновки до третього розділу

Було описано інтерфейс користувача програмного додатку та наведено послідовність кроків для його використання. Було розглянуто різні компоненти і елементи інтерфейсу, їх функціональність та взаємодію з користувачем. Порядок роботи з програмним додатком було описано з урахуванням послідовності дій та основних функцій, які можна виконувати за допомогою програмного забезпечення.

Було розглянуто процес тестування функціональності програмного додатку. Були описані різні види тестування, такі як модульне тестування, інтеграційне тестування та системне тестування. Було наведено приклади тестів, які можна провести для перевірки працездатності програмного додатку та виявлення можливих помилок та проблем.

ВИСНОВКИ

В даній кваліфікаційній роботі бакалавра було проведено аналіз проблематики, методів та засобів вирішення задачі, пов'язаної з розробкою програмного додатку – гри-головоломки «What are the ghosts talking about?». Аналіз показав, що існує необхідність у створенні механік, анімацій, візуальних ефектів та музичного супроводу для досягнення бажаного функціоналу та враження користувачів.

В результаті проведеного аналізу було розроблено основні механіки програмного додатку. Це включало в себе визначення правил, взаємодії об'єктів та логіки гри. Розроблені механіки вирішують поставлену задачу та забезпечують коректну роботу програмного додатку, а саме: правильна взаємодія користува й персонажа, персонажа і предметів на локації.

Для забезпечення цікавого та привабливого користувачам досвіду були створені анімації та візуальні ефекти, а саме: анімації головних і другорядних персонажів, предметів на локаціях та робота зі світлом. Це додало грі динаміки та візуальної привабливості.

Також був написаний музичний супровід, який доповнив загальний настрій та атмосферу програмного додатку. Музика була створена з урахуванням тематики та характеру гри, що підкреслило її унікальність та особливість.

Було проведено тестування програмного додатку, щоб перевірити його працездатність, коректність роботи та виявити та виправити можливі помилки.

Загалом, робота показала, що вивчення проблематики, використання відповідних методів та засобів, розробка механік, анімацій, візуальних ефектів та музичного супроводу, а також робота з програмним додатком та його тестування, допомогли успішно розв'язати поставлену задачу. Результатом є готовий програмний додаток, який забезпечує задану функціональність та забезпечує задоволення користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Fruity Loops Studio - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://www.image-line.com/fl-studio/>
2. Adobe After Effects - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://www.adobe.com/products/aftereffects.html>
3. Spine - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/>
4. Fruity Loops Studio Manual - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://www.image-line.com/fl-studio-learning/>
5. Getting Started: The FL Studio Interface & Main windows - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/html/basics_interface.htm
6. How to use FL Studio - Making music - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/html/basics_workflow.htm
7. Fruity Loops Studio System Settings - MIDI - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/html/envsettings_midi.htm
8. Fruity Loops Studio System Settings - Audio - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/html/envsettings_audio.htm
9. Adobe After Effects Work with compositions - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://helpx.adobe.com/ua/after-effects/how-to/create-composition-animation.html?playlist=/ua/uk/ccx/v1/collection/product/after-effects/segment/designer/explevel/beginner/applaunch/ccl-get-started-1/collection.ccx.js&ref=helpx.adobe.com>
10. Adobe After Effects Build custom transitions - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://helpx.adobe.com/ua/after-effects/how-to/create-custom-transitions.html?playlist=/ua/uk/ccx/v1/collection/product/after->

[effects/segment/designer/explevel/beginner/applaunch/ccl-get-started-1/collection.ccx.js&ref=helpx.adobe.com](https://helpx.adobe.com/ua/after-effects/how-to/animate-along-path.html?playlist=/ua/uk/ccx/v1/collection/product/after-effects/topics/animating-essentials/collection.ccx.js&ref=helpx.adobe.com)

11. Adobe After Effects Animate an object to follow along a path - [Офіційний веб-сайт]. Режим доступу до ресурсу: <https://helpx.adobe.com/ua/after-effects/how-to/animate-along-path.html?playlist=/ua/uk/ccx/v1/collection/product/after-effects/topics/animating-essentials/collection.ccx.js&ref=helpx.adobe.com>

12. Adobe After Effects Keyframe animation - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://helpx.adobe.com/ua/after-effects/how-to/keyframe-animation.html?playlistPath=/services/playlist.helpx/products:SG_AFTEREFFECTS_1_1/set-header:animating-essentials/learn-path:key-techniques/playlist:topic/uk_UA.json

13. Adobe After Effects Control animation of multiple layers with parenting - [Офіційний веб-сайт]. Режим доступу до ресурсу: https://helpx.adobe.com/ua/after-effects/how-to/animating-with-parenting.html?playlistPath=/services/playlist.helpx/products:SG_AFTEREFFECTS_1_1/set-header:animating-essentials/learn-path:key-techniques/playlist:topic/uk_UA.json

14. Техніки колірного графічного дизайну - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-getting-started>

15. Spine Getting started - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-getting-started>

16. Spine User Guide - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-user-guide>

17. Spine Animating - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-animating>

18. Spine Skeletons - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-skeletons>

19. Spine Bones - [Офіційний веб-сайт]. Режим доступу до ресурсу: <http://esotericsoftware.com/spine-bones>

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		54

20. Spine Images - [Офіційний веб-сайт]. Режим доступу до ресурсу:
<http://esotericsoftware.com/spine-images>

					КН.КР.Б – 122 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		55

ДОДАТКИ

Найбільш важливі функції та класи

Файл DragAndDrop.cs:

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

public class DragAndDrop : MonoBehaviour
{
    protected Vector3 mousePositionOffset;
    protected Rigidbody2D rb;

    protected virtual void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    protected Vector3 GetMouseWorldPosition()
    {
        return Camera.main.ScreenToWorldPoint(Input.mousePosition);
    }

    protected virtual void OnMouseDown()
    {
        mousePositionOffset = transform.position - GetMouseWorldPosition();
    }

    protected virtual void OnMouseDrag()
    {
        Vector2 position = GetMouseWorldPosition() + mousePositionOffset;
        rb.MovePosition(position);
    }
}
```

Файл Interactable.cs:

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

public abstract class Interactable : MonoBehaviour
{
    protected Transform player;

    void Start()
    {
        player = PlayerManager.instance.player.transform;
    }

    public virtual void Interact()
    {
        Debug.Log("Interact");
    }

    public void OnMouseDown()
    {

```

```

        Interact();
    }
}

```

Файл Item.cs:

```

using UnityEngine;

[CreateAssetMenu(fileName = "New item", menuName = "Inventory/Item")]
public class Item : ScriptableObject
{
    //"new" because sth already owns this name
    new public string name = "New item";
    public Sprite icon = null;
    public int numberInInventory = 0;
    public Item itemComponent; //item that is needed to craft itemResult
    public Item itemResult;    //item that is result of craft

    public virtual void Use()
    {
        // Use the item
        Debug.Log("Using " + name);

        //craft new item if it is possible

        if(PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem ==
itemComponent && itemComponent != null)
        {
            Craft(this, itemComponent, itemResult);
        }
        //else just make this item active
        else

        PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem =
this;

        InventoryUI.instance.UpdateUI();
    }

    public void Craft(Item itemComponent1, Item itemComponent2, Item
itemResult)
    {
        Inventory.instance.Remove(itemComponent1);
        Inventory.instance.Remove(itemComponent2);
        Inventory.instance.Add(itemResult);

        PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem =
itemResult;
    }
}

```

Файл ItemDestroyPuzzle.cs:

```

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

public class ItemDestroyPuzzle : Interactable
{

```

```

        [SerializeField] private GameObject inventoryUI;
        [SerializeField] private GameObject objects;

        public Item item;
        [SerializeField]
        private GameObject puzzleParent;
        public override void Interact()
        {
            base.Interact();
            if (this.isActiveAndEnabled)
                Pickup();
        }

        public void Pickup()
        {
            Debug.Log("Picking up item" + item.name);

            inventoryUI.SetActive(true);
            objects.SetActive(true);

            if (FindObjectOfType<CameraMovement>() != null)
                FindObjectOfType<CameraMovement>().AbleMovement();

            bool wasPickedUp = Inventory.instance.Add(item); //add to
inventory
            if (wasPickedUp)
                Destroy(gameObject);

            Destroy(puzzleParent);
        }
    }
}

```

Файл ItemPickUp.cs:

```

using UnityEngine;

//things that can be store in inventory
public class ItemPickUp : Interactable
{
    public Item item;

    public override void Interact()
    {
        base.Interact();
        Pickup();
    }

    public void Pickup()
    {
        Debug.Log("Picking up item" + item.name);

        bool wasPickedUp = Inventory.instance.Add(item); //add to
inventory
        if (wasPickedUp)
            Destroy(gameObject);
    }
}

```

```
}
```

Файл ItemPickUpWithActiveItem.cs:

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

public class ItemPickUpWithActiveItem : Interactable
{
    public Item item;

    public Item activeItem;

    private void Start()
    {
        player = PlayerManager.instance.player.transform;
    }
    public override void Interact()
    {
        if (player.GetComponent<PlayerInteract>().activeItem ==
activeItem)
        {
            Pickup();
            Inventory.instance.Remove(activeItem);
        }
    }

    public void Pickup()
    {
        Debug.Log("Picking up item" + item.name);

        bool wasPickedUp = Inventory.instance.Add(item); //add to
inventory
        if (wasPickedUp)
            Destroy(gameObject);
    }
}
```

Файл ItemPickUpWithoutDestroying.cs:

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

public class ItemPickUpWithoutDestroying : Interactable
{
    public Item item;

    public override void Interact()
    {
        base.Interact();
        Pickup();
    }

    public void Pickup()
    {
        Debug.Log("Picking up item" + item.name);
        Inventory.instance.Add(item);
    }
}
```

```

        gameObject.SetActive(false);
    }
}

```

Файл ShowObjectWithItemClick.cs:

```

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

public class ShowObjectWithItemClick : Interactable
{
    [SerializeField] Item item;
    [SerializeField] GameObject objectToShow;

    public override void Interact()
    {
        base.Interact();
    }
}

```

```

if (PlayerManager.instance.player.GetComponent<PlayerInteract>().activeItem ==
item)
{
    objectToShow.SetActive(true);
    Inventory.instance.Remove(item);
}
}

```

Файл PlayerMovement.cs:

```

using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    //private Rigidbody2D rb;
    private SpriteRenderer sprite;
    public Animator animator;
    private float HorizontalMove = 0f;
    //private bool FacingRight = true;

    public float speed = 5f;
    //public float jumpForce = 5f;

    //public float moveToX;

    private void Start()
    {
        //rb = GetComponent<Rigidbody2D>();
        sprite = GetComponentInChildren<SpriteRenderer>();
    }

    private void Update()
    {
        //movement
        if (Input.GetButton("Horizontal"))
            Move();
    }
}

```

```

        HorizontalMove = Input.GetAxisRaw("Horizontal") * speed;
        animator.SetFloat("HorizontalMove", Mathf.Abs(HorizontalMove));
    }

    private void Move()
    {
        Vector3 dir = transform.right * Input.GetAxis("Horizontal");

        transform.position = Vector3.MoveTowards(transform.position,
transform.position + dir, speed * Time.deltaTime);

        //flip sprite to face movement direction
        sprite.flipX = dir.x < 0.0f;
    }
}

```

Файл PlayerInteract.cs:

```

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

public class PlayerInteract : MonoBehaviour
{
    public Item activeItem;
}

```