

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ

к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

**на тему: «Системний аналіз та оцінка впливу штучного інтелекту на
розробку ігор»**

Виконав: студент групи 6145 м

_____ ***Маліцький Р.П.***
(підпис)

Керівник роботи:

старший викладач
(посада, науковий ступінь вчене звання)

_____ ***Павленко А.Ю.***
(підпис)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

Спеціальність 124 "Системний аналіз"

Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

д.ф-м.н, проф. Ушкац М.В.

" ____ " _____ 2023 р.

***ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти "магістр"***

Студенту Маліцькому Роману Петровичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Системний аналіз та оцінка впливу штучного інтелекту на розробку ігор _____

Керівник роботи Павленко Альона Юріївна

Затверджені наказом ректора № 1098-уч. від "18" вересня 2023 року.

2. Термін подання роботи: 15 грудня 2023 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування

4. Перелік питань, що належать до розробки (найменування розділів)

Аналіз та порівняння впливу штучного інтелекту на розробку ігор. Розробка проекту та концепції інформаційної системи «Geometry Astro Survival». Реалізація проекту автоматизованої системи. Розрахунок економічної ефективності впровадження мобільної гри «Geometry Astro Survival». Охорона праці. Охорона навколишнього середовища.

5. Перелік презентаційних матеріалів _____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Павленко А.Ю.	05.09.2022	22.12.2022
2	Павленко А.Ю.	23.12.2022	15.06.2023
3	Павленко А.Ю.	15.06.2023	25.09.2023
4	Павленко А.Ю.	25.09.2023	14.11.2023
5	Павленко А.Ю.	25.09.2023	14.11.2023
6	Павленко А.Ю.	25.09.2023	14.11.2023

7. Дата видачі завдання 05 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Дослідження та аналіз штучних інтелектів	22.12.2022	
2	Розробка концепції	15.03.2023	
3	Розділ з розробки проектних рішень по системі (та її частинах)	15.06.2023	
4	Реалізація проекту та розробка робочої документації	25.09.2023	
5	Обчислення економічної ефективності мобільної гри	14.11.2023	
6	Вирішення питань охорони праці та навколишнього середовища	14.11.2023	
7	Оформлення пояснювальної записки	14.11.2023	
8	Подання магістерської роботи на перевірку на плагіат	16.11.2023	
9	Подання магістерської роботи рецензенту	20.11.2023	
10	Подання магістерської роботи на державний захист	10.12.2023	

Студент

(підпис)

Маліцький Р.П.

(ПІБ)

Керівник роботи

(підпис)

Павленко А.Ю.

(ПІБ)

АНОТАЦІЯ

Темою роботи є «Системний аналіз та оцінка впливу штучного інтелекту на розробку ігор».

Магістерська робота виконується для аналізу та дослідження штучних інтелектів. Об'єктом дослідження є вплив штучного інтелекту на розробку ігор.

Мета дослідження - системний аналіз та оцінка впливу штучного інтелекту на прикладі мобільної гри «Geometry Astro Survival» у жанрі Casual, Action.

Актуальність роботи полягає в тому, що, штучний інтелект може значно покращити геймплей та користувацький досвід. Використання штучного інтелекту дозволяє створити реалістичних ігрових персонажів, розумних ворогів та непередбачувані ситуації, що збільшує цікавість та виклик гри.

Наукова новизна цього дослідження полягає в тому, що воно ретельно розглядає можливості, які штучний інтелект може принести у сферу розробки ігор, а також у вирішенні питань етики, технічних аспектів і творчих викликів, пов'язаних з його використанням у даній галузі.

Крім того, штучний інтелект може допомогти вирішувати проблеми з балансуванням гри, адаптивним рівнем складності та автоматичним підлаштуванням до стилю гри кожного користувача. Він також може покращити систему управління та інтерактивність гри, забезпечуючи більш глибокий та персоналізований досвід для гравця.

Загалом, аналіз та дослідження штучного інтелекту є важливим етапом при розробці мобільної гри, оскільки вони дозволяють створити більш цікавий, реалістичний та залучаючий геймплей для користувачів.

У роботі було проведено аналіз та дослідження штучного інтелекту в контексті розробки мобільної гри. Було виконано дослідження сучасних тенденцій та методик використання штучного інтелекту в галузі ігрової розробки. Було визначено, який підхід до штучного інтелекту буде найбільш ефективним для розробки мобільної гри. Далі було розроблено алгоритми та моделі, які використовують штучний інтелект для покращення геймплею, створення розумних ворогів та непередбачуваних ситуацій. Також було проведено аналіз засобів розробки для мобільних платформ та враховано всі вимоги та обмеження обраної

інформаційної системи. Результатом роботи став покращений інтелект ворогів, новий дизайн гри та непередбачені події під час гри.

Робота викладена на 84 сторінках друкованого тексту, містить 6 розділа, 25 рисунків, 4 таблиць та 3 додатки, технічне завдання, текст програми та інструкцію з використання.

Робота виконана українською мовою.

DESCRIPTION

The theme of the work is "System Analysis and Evaluation of the Impact of Artificial Intelligence on Game Development".

The master's thesis is being conducted to analyze and research artificial intelligences. The object of the study is the development of a game using artificial intelligence and the analysis of the coefficient of useful action.

The aim of the work is to conduct a system analysis and evaluate the impact of artificial intelligence using the mobile game "Geometry Astro Survival" in the Casual Action genre as an example.

The relevance of the work lies in the fact that artificial intelligence can significantly enhance gameplay and user experience. The use of artificial intelligence allows for the creation of realistic game characters, intelligent enemies, and unpredictable situations, increasing the interest and challenge of the game.

The scientific novelty of this research lies in its thorough examination of the possibilities that artificial intelligence can bring to the field of game development, as well as in addressing ethical issues, technical aspects, and creative challenges associated with its use in this industry.

Furthermore, artificial intelligence can help solve problems with game balancing, adaptive difficulty levels, and automatic adjustment to each user's playing style. It can also improve the game's control system and interactivity, providing a deeper and more personalized experience for the player.

Overall, the analysis and research of artificial intelligence are crucial stages in mobile game development, as they allow for the creation of a more engaging, realistic, and immersive gameplay for users.

The work involved analyzing and researching artificial intelligence in the context of mobile game development. Research was conducted on current trends and methodologies for using artificial intelligence in the gaming industry. The most effective approach to artificial intelligence for mobile game development was determined. Algorithms and models were then developed that utilize artificial intelligence to enhance gameplay, create intelligent enemies, and introduce unpredictable situations. Analysis of development tools for mobile platforms was also conducted, taking into account all

requirements and limitations of the chosen information system. The result of the work includes improved enemy intelligence, a new game design, and unexpected events during gameplay.

The work is presented on 84 pages of printed text, contains 6 sections, 25 figures, 4 tables and 3 appendices, which contain a terms of reference, program text and instructions for use.

The work is done in Ukrainian.

ЗМІСТ

СЛОВНИК.....	10
ВСТУП.....	11
РОЗДІЛ I АНАЛІЗ ТА ПОРІВНЯННЯ ВПЛИВУ ШТУЧНОГО ІНТЕЛЕКТУ НА РОЗРОБКУ ІГОР.....	12
1.1 Аналіз штучних інтелектів	12
1.1.1 Типи штучних інтелектів.....	13
1.2 Аналіз впливу штучного інтелекту на розробку ігор	15
1.2.1 Аналіз штучних інтелектів за сферою діяльності при розробці ігор	16
1.3 Аналіз заміни професій у розробці ігор на штучний інтелект	21
РОЗДІЛ II РОЗРОБКА ПРОЕКТУ ТА КОНЦЕПЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ «GEOMETRY ASTRO SURVIVAL»	23
2.1 Опис об'єкту та аналіз предметної області	23
2.1.1 Аналіз предметної області	23
2.1.2 Аналіз існуючих рішень.....	24
2.1.3 Аналіз засобів розробки для мобільних платформ	27
2.2 Вимоги до інформаційної області	29
2.3 Розробка концепції інформаційної системи	32
2.4 Загальносистемні рішення.....	34
2.4.1 Діаграма варіантів використання.....	34
2.4.2 Специфікація варіантів використання.....	36
2.5 Рішення з інформаційного забезпечення.....	39
2.5.1 Діаграма класів.....	39
2.5.2 Діаграма переходів станів.....	40
2.5.3 Діаграма діяльності	43
2.5.4 Діаграма компонентів.....	44
2.5.5 Діаграма розгортання	45
2.6 Рішення з програмного забезпечення	45
РОЗДІЛ III РЕАЛІЗАЦІЯ ПРОЕКТУ МОБІЛЬНОЇ ГРИ «GEOMETRY ASTRO SURVIVAL».....	47

РОЗДІЛ IV РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ МОБІЛЬНОЇ ГРИ «GEOMETRY ASTRO SURVIVAL».....	54
4.1 Вступ.....	54
4.2 Розрахунок витрат на створення та експлуатацію мобільної гри «GEOMETRY ASTRO SURVIVAL»	54
4.3 Економічна ефективність розробки і впровадження «GEOMETRY ASTRO SURVIVAL»	56
РОЗДІЛ V ОХОРОНА ПРАЦІ.....	58
5.1 Вступ.....	58
5.2 Аналіз шкідливих та небезпечних факторів.....	58
5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій	60
5.4 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів при роботі з персональним комп'ютером ..	61
5.4.1 Заходи щодо зменшення впливу	61
5.4.2 Техніка безпеки при роботі на комп'ютері	62
РОЗДІЛ VI ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА.....	65
6.1 Вступ.....	65
6.2 Забруднення навколишнього середовища	66
6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища.....	66
ВИСНОВКИ	68
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	69
ДОДАТОК А – Технічне завдання	71
ДОДАТОК Б – Текст програми.....	74
ДОДАТОК В – Інструкція з використання	80

СЛОВНИК

- Indie¹ – це категорія розробників які створили свій проект самостійно.
- Нодам² – це такі об'єкти, як події, виклики функцій, операції керування потоком, змінні тощо.
- Pool³ - це спосіб оптимізації ваших проектів і зниження навантаження на центральний процесор, коли доводиться швидко створювати та знищувати нові об'єкти. Це хороша практика та модель проектування, яку слід пам'ятати, щоб допомогти розвантажити процесорну потужність центрального процесору для обробки більш важливих завдань і не засипатися повторюваними викликами створення та знищення.
- Skill⁴ – здібність які надає гравцю могутності.
- Pop-up⁵ - це область відображення графічного інтерфейсу користувача (GUI), зазвичай невелике вікно, яке раптово з'являється («спливає») на передньому плані візуального інтерфейсу.
- Particles⁶ - це надійна система ефектів частинок, де можна імітувати рухомі рідини, дим, хмари, полум'я, магичні заклинання та цілу низку інших ефектів.
- VFX⁷ – візуальні ефекти.
- Prefab⁸ – це контейнер для усіх закріплених у ньому об'єктах, скриптів, анімацій, ефектів та ін.
- UI⁹ – інтерфейс програми

ВСТУП

Штучний інтелект на сьогоднішній день усюди, в телефоні, комп'ютері, інтернеті і незважаючи на їх кількість – кожен з них унікальний оскільки навчаються хоч і в більшості на однакових даних, але вихідний результат завжди різний.

У компаніях котрі розробляють мобільні ігри пріоритетом є швидкість оскільки популярність конкретного жанру швидко спливає і тому потрібно швидко створювати нову гру, в новому стилі та жанру аби якомога більше охопити аудиторії та заробити на цьому ось чому штучний інтелект це спасіння для розробника оскільки один штучний інтелект проаналізує, напише або оптимізує код, інший намалює графіку, третій напише музику програмісту залишиться лише усе поєднати у гру та поширити її в інтернеті.

Таким чином компанія менше втрачає грошей, програмісти розвиваються створюючи ігри, та аналізуючи код від штучного інтелекту і, як на мене найголовніше – вчать програмувати ігри різних жанрів і тим самим розвиваються своє мислення та креативність.

У проекті «ASTRO SURVIVAL» були використано декілька штучних інтелектів аби проаналізувати їх ефективність та легкість роботи з ними.

РОЗДІЛ І АНАЛІЗ ТА ПОРІВНЯННЯ ВПЛИВУ ШТУЧНОГО ІНТЕЛЕКТУ НА РОЗРОБКУ ІГОР

1.1 Аналіз штучних інтелектів

Штучні інтелекти(далі ШІ) в тепершній час стають все більш популярними та звичнішим для нас інструментом котрий допоможе як прискорити роботу як частково, або повноцінно замінити людину.

Першим ШІ можна назвати руку-помічника на заводах, де замість людей деталі зварюють роботи котрі оцінюють два елементи, фіксують їх в конкретній позиції та виконують свою роботу, котру, можна назвати ідеальною, без перебільшення оскільки людині треба великий досвід роботи аби все чітко та без помилок виконати, а роботу зі штучним інтелектом лише набір інструкцій, а далі він сам буде робити роботу, лише змащувати час від часу його механізм.

ШІ замінюють людей все більше і більше від заводів до офісів, від зварювальника до програміста і оскільки програмісти самі пишуть ШІ, ми, можна сказати по трохи принижуємо себе роблячи пропаганду ШІ котрий робить краще та швидше ніж людина.

Виходячи з цього можна виділити наступні підходи по створенню ШІ:

- низхідний або семіотичний
- висхідний або біологічний

Де перший, низхідний - використовується для модуляції людини, її поведінки, емоцій, мови, творчості та ін. А інший, висхідний – вчиться на штучних нейронних мережах та еволюціонує для модуляції поведінки менших «неінтелектуальних» елементів.

Хоч це і відносно молода галузь дослідження котра започаткована у 1956 році, але вже, в деяких галузях показує досить шокуючі результати як-от водіння автомобілей, розпізнавання мови або її імітація.

Штучний інтелект зараз усюди, тим самим спрощує людям життя, але й лякає їх. «Чи можуть машини мислити?» таке питання від Алана Тьюринга змінило історію, та дало поштовх до розробки ШІ.

1.1.1 Типи штучних інтелектів

Так, за типами ШІ можна поділити на:

- ШІ на основі правил, виходячи з назви цей ШІ створений за допомогою правил, або, легше кажучи - операторів if-then, які допомагають йому робити висновки виходячи з цих правил.
- машинне навчання, це ШІ котрий вчиться на вхідних даних котрі йому дають, або котрі він генерує сам відносно похибки та покращує результат з часом. У свою чергу даний ШІ поділяється ще на три підвиди:
 - кероване навчання, ШІ котрому вхідні дані дає людина, після чого він їх оброблює та надсилає результат, але не будь-який, а бажаний людиною. Якщо ж результат не відповідає очікуванням ШІ знову почне їх оброблювати й з кожною ітерацією буде покращувати себе: свої алгоритми, функції та мислення в цілому.
 - некероване навчання, на відміну від керованого навчання в цьому випадку ШІ намагається сам імітувати дані які йому дають та використовує похибку на виході аби покращити себе.
 - навчання з підкріплення, в цьому випадку ШІ навчається на компромісах де пари входів/виходів ніколи не представляються, а на недостатньо правильні дії закриваються очі. Але, це не означає, що даний ШІ сам по собі й всі про нього забули, ні, навпаки, при його навчанні намагаються знайти баланс між дослідженням та використанням. Аби інтелект не був розвинений лише в одному напрямленні.
- обробка природної мови, я вже описував ситуацію з імітацією людської мови, якраз для цього створений ШІ для обробки природної мови. Якщо говорити більш детально – він використовується для взаємодії людини та комп'ютера. Так, завдяки цьому ШІ створюються чат боти, голосові помічники тощо. Але, вже існують ШІ котрі можуть максимально наблизитись до людської мови і буде надто складно зрозуміти, де

людина говорить, а де робот. І цей прогрес призводить до телефонних шахраїв та взламу голосових замків, в цілому усі ІІІ використовуються як для добра, так і для зла.

- робототехніка, на заводах, де все більше і більше людина замінюється роботами котрі виконують роботу людей. Але не тільки в заводах використовуються робототехніка, так наприклад дрони доставники, або робот пес котрий може щось тягнути або іншим способом допомагати людині. Взагалі найпростіші роботів створюють школяри на заняттях де плата створена завдаки Arduino, а код написаний на Python і ці роботи можуть як переставляти кубики так і розрізняти людей.
- експертні системи, цей ІІІ виступає у ролі «експерта» у певних сферах, він може або допомогти, або цілком замінити людину та надавати свої поради у дизайні, бізнесі, маркетингу, медицині, та, навіть праві. Дивно, що люди пишуть права для людей, а потім створюють ІІІ котрий ці права диктує людям. І нехай люди все ж таки корегують його поради, але хто знає, можливо через кілька років свої правила та права почне диктувати робот і ми не зможемо від них відмовитись.
- сильний штучний інтелект, це спроба створити ІІІ котрий імітує людину, як за поведінкою, так навчанням та мисленням. Найкраща новина в тому, що поки це більше концепція ніж реалізація, оскільки потрібно вирішити велику купу проблем, з якими, як не дивно допомагають інші види ІІІ.

Виходячи з усього вище сказаного можна зробити висновок, що ІІІ різних видів допомагають як людям так і собі, але тим сами вони і руйнують нас, людей. Якщо молоде покоління котре цікавиться темою ІІІ, ще може зрозуміти усю користь та негатив від ІІІ, то людей літнього віку ця тема – лякає й ніяк інакше. ІІІ усюди, починаючи від систем допомоги людям та сфери бізнесу завершуючи війною. Так ракети з тепловізором, системи ППО, літаки з автопілотами та багато іншого, що використовується задля перемоги та ефективного знищення ворога.

Але не варто забувати, що при розробці ІІІ завжди повинен бути рубильник, котрий вимкне систему у випадку її виходження за рамки інструкцій

та правил котрі дала людина. Оскільки без рубильника це призведе до плачевних наслідків.

1.2 Аналіз впливу штучного інтелекту на розробку ігор

Для того щоб зробити гру треба як мінімум один програміст та час, від декількох годин до декількох років. А для того, щоб прискорити цей процес потрібні ІШ, котрі розділяються на свої сфери розробки.

Так ІШ котрі пишуть код, малюють спрайти, картини, пишуть сценарій, або цілі історії на основі котрих створюються світи, пишеться музика, створюються анімації та багато інших – допомагають розробникам прискорити, покращити та в цілому підвищити якість гри та швидкість її розробки. Але навіть для цього, потрібен досвід їх використання та навички написання коду у розробника.

Оскільки ІШ це машина з набором інструкцій та правил, вона вчиться, але вона не людина і вона робить помилки. Більше того більшість ІШ мають свій ліміт у відповідях тому не можна запросити в ІШ величезну програму або повну гру – вона цього не зробить бо так запрограмована. Навпаки, вона буде надавати невеличкі шматки котрі розробник повинен правильно з'єднати аби усе працювало.

Виходячи з цього можна зробити висновок, що для правильного використання ІШ необхідно:

1. правильно поставити питання
2. мати навички доробки вихідного результату ІШ
3. мати точне уявлення, для чого конкретний ІШ призначений

Ось ці три пункта є запорукою успіху при використанні ІШ. Оскільки творець повинен чітко бачити свою мету та вміти використовувати будь-який механізм для прискорення своїх праці без втрати якості у своїй роботі та вміти виправляти недоліки того механізму котрий він використовує.

1.2.1 Аналіз штучних інтелектів за сферою діяльності при розробці ігор

При розробці ігор можна виділити наступні ШІ:

- для написання коду
- для малювання спрайтів
- для створення анімацій

Виходячи з цього списку в кожній сфері можна виділи своїх фаворитів серед ШІ для виконання поставлених цілей:

- Серед великої кількості ШІ для написання коду виділяються двоє кандидатів:
 - Chat GPT, ШІ котра є чат-ботом та генерує відповідь на основі запиту від користувача. Взагалом даний ШІ може замінити декілька різних ШІ, оскільки він може і написати код, і історію, і перевести якийсь текст та ін. На даний момент безкоштовно можна використовувати його версію 3.5, але є й платна версія 4.0 котра хвалиться більше детальними та вірними відповідями на запит користувача. Та й в цілому платна версія здатна на більші речі ніж безкоштовна.

Якщо порівнювати Chat GPT наприклад з асистентом гугл котрий по запиту користувача знайде інформацію на конкретній сторінці серед купи інших то, даний ШІ створить для вас відповідь на ваш запит базуючих на тих даних котрі в ньому збережені. Це буде в більшості своїй унікальна інформація, але не сто відсоткова. Оскільки цей ШІ використовує нейронну сітку котру навчали на великій кількості інформації, а саме приблизно 570 гігабайт тексту збережено в ньому, це більше 500 трільйонів слів які даний ШІ здатен розуміти та використовувати базуючись на архітектурі «трансформер» використовуючи генеративну модель даних.

Інтерфейс Chat GPT представлено на рисунку 1.1

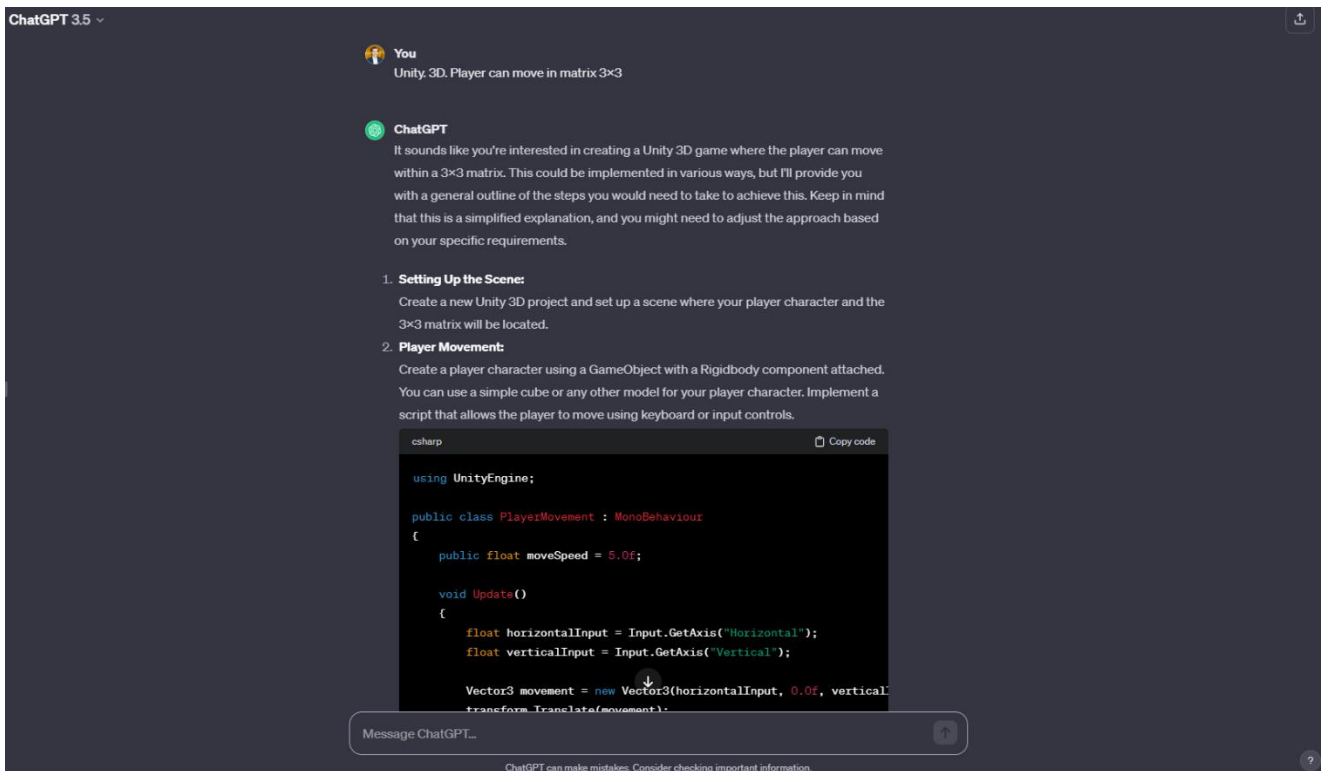
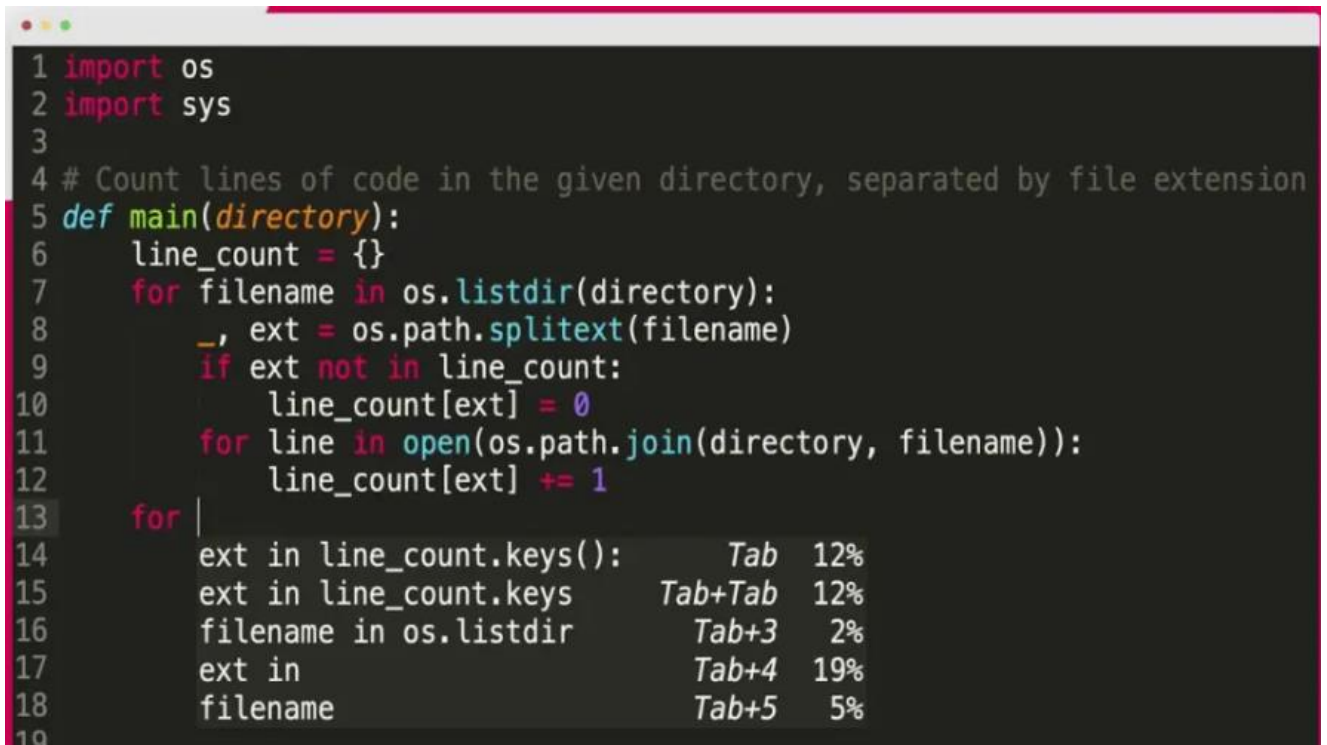


Рис 1.1 – скріншот інтерфейсу Chat GPT

- Tabnine, ІІІ для написання коду, але він більш вузько спеціалізований ніж Chat GPT, оскільки він може працювати лише з кодом. Він як і Chat GPT розуміє запит користувача, але відповідь у нього лише кодом різних мов програмування. Він здатен вивчити увесь код проекту, оцінити його, та написати новий або виправити існуючий прямо в середовищі розробки.

Даний ІІІ розповсюджується як плагін до середовища розробки, що набагато зручніше ніж Chat GPT. Більше того, він використовує навчання з підкріпленням, що дозволяє навчатись у режимі реального часу не відправляючи ніякі дані на сервер, а оброблюючи усе локально, всередині середовища розробки.



```
1 import os
2 import sys
3
4 # Count lines of code in the given directory, separated by file extension
5 def main(directory):
6     line_count = {}
7     for filename in os.listdir(directory):
8         _, ext = os.path.splitext(filename)
9         if ext not in line_count:
10             line_count[ext] = 0
11         for line in open(os.path.join(directory, filename)):
12             line_count[ext] += 1
13     for |
14         ext in line_count.keys():           Tab 12%
15         ext in line_count.keys           Tab+Tab 12%
16         filename in os.listdir           Tab+3 2%
17         ext in                           Tab+4 19%
18         filename                         Tab+5 5%
19
```

Рис. 1.2 – скріншот інтерфейсу Tebnine

- Для створення ж спрайтів чудово підходить PlaygroundAI. Оскільки він безкоштовний і може зчитувати запит користувача як змістовне речення, так і картинку. Наприклад користувач хоче створити щось схоже на те, що в нього є, він завантажує до ІІІ свої спрайти та пише запит, а ІІІ генерує йому результат. Це єдина ІІІ котра здатна генерувати спрайти для ігор у потрібному форматі, в необхідному розширенні та, якщо потрібно з анімацією. ІІІ вчиться у реальному часі завдяки запитам користувачів через це її використовуються повсякденно для різних задач, не тільки для створення ігор, але і для скетчів або референсів. Але це не означає, що інші ІІІ для створення спрайтів погані або їх не можна використовувати, єдине що перешкоджає цьому – їх ціна, вони платні і їх використовувати розробникам-новачкам не вийде, але звісно, вони можуть бути кращими за PlaygroundAI.

Інтерфейс PlaygroundAI представлено на рисунку 1.3

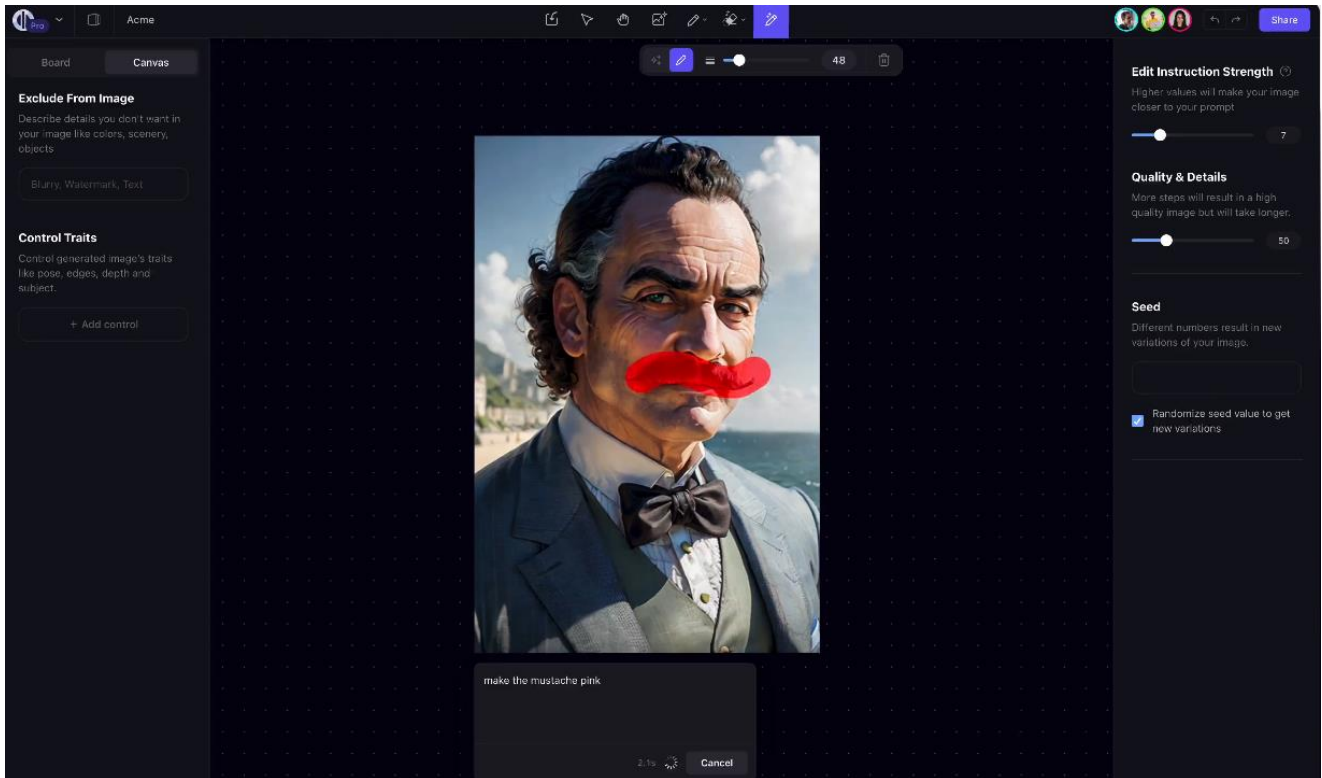


Рис 1.3 – скріншот інтерфейсу PlaygroundAI

- Для генерування анімацій не все так просто оскільки їх можна створювати, у випадку якщо це 2D картинка – в тому ж ШІ в котрому вона була згенеровано.

Для створення 2D анімацій, виділяються двоє:

- Midjourney, коштує грошей, хоча і має безкоштовний період. Генерує картинки за запитом, але більш футурестичні, які для ігор не сильно підходять. Користується великим спросом через свою легкість у роботі, та більш професійним результатом. Наприклад для генерування дизайну кредитки, візитки або білборду. Також генерує результат за запитом користувача, або на основі існуючих картинок. Але нажаль працює через сторонню програму Discord, тому будь-які проблеми з нею – впливають на роботу ШІ. Більше того, даний ШІ потребує немалих навичок при написанні запитів, оскільки ШІ генерує все в точності як користувач запросить, тому не кожен зможе створити через неї щось гарне та професійне.

Інтерфейс Midjourney представлено на рисунку 1.4

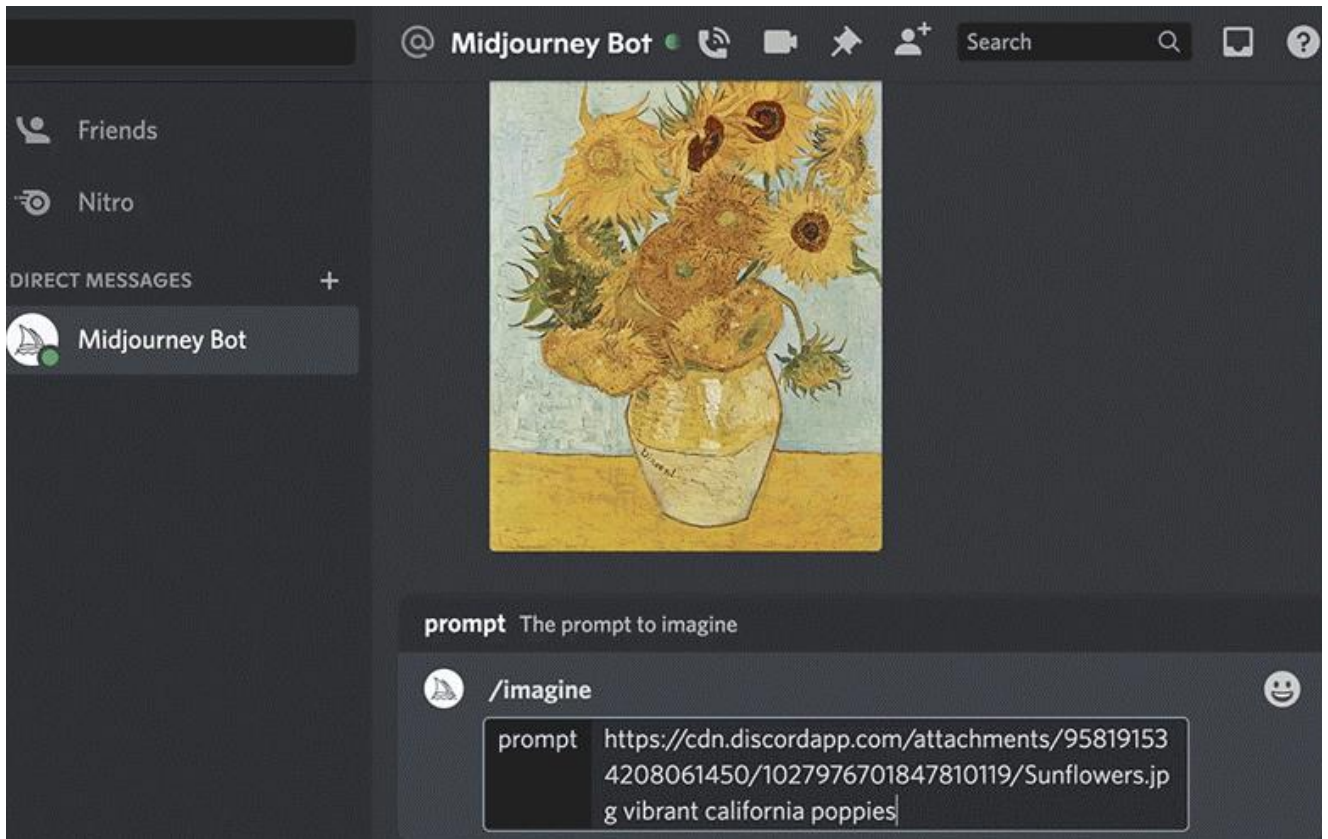


Рис 1.4 – скріншот інтерфейсу Midjourney

- DALL-E, досить складний ШІ але при цьому отримав свою популярність оскільки здатен генерувати якісні образи людей на відміну від інших ШІ, але при цьому потребує у використанні технічних та професійних знань. Найбільшим плюсом цього ШІ є можливість використовувати результат у комерційних цілях на відміну від інших ШІ, де це заборонено і потребує редагування користувачем. Більше того він не потребує сторонньої програми, а працює прямо з браузера, і, як і інші ШІ генерує результат за запитом, або використовуючи інші згенеровані картинки.

Інтерфейс DALL-E представлено на рисунку 1.5

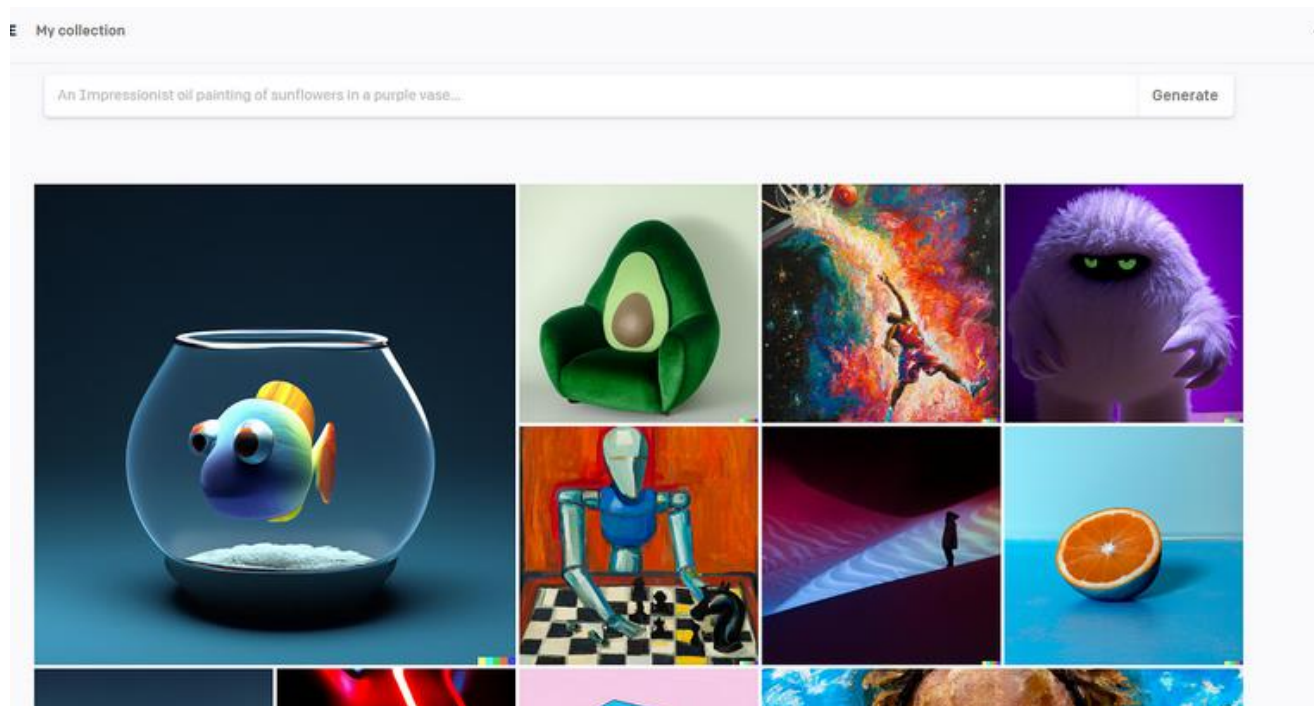


Рис. 1.5 – скріншот інтерфейсу DALL-E

Виходячи з цього, можна зробити висновки, що кожен з ІІІ, котрі підходять для створення ігор потребують знань, як теоретичних так і технічних тому новачку буде складно ними користуватись. Але це не означає, що не потрібно намагатись, бо з кожною ітерацією ІІІ стає розумнішим, а результат все краще та краще що зменшує необхідність у технічних знаннях користувача.

Так чи інакше ІІІ допомагають розробникам створювати ігри, але не без проблем, бо це машина, котра не наділена навичками людини, можливо вона і створює щось краще, але цей результат не виправдовує очікувань розробника, або взагалі не працює у випадку кода.

1.3 Аналіз заміни професій у розробці ігор на штучний інтелект

На сьогоднішній день все більше та більше людей залишаються безробітними, тому що їхню професію частково, або повністю замінила машина із ІІІ. Через це піднімаються мітинги, протести і можливо будуть війни, але я вважаю, що окрім ІІІ розвиватись потрібно й людям.

Людський мозок здатен оброблювати велику кількість даних та операцій одночасно це живий комп'ютер, але навідміну від реального комп'ютера його запчастини не можна замінити, та/або вдосконалити аби збільшити обчислювальну міць. Через це, як вважає Ілон Маск ШІ здатен перевершити людський мозок за долі секунди в декілька тисяч мільйонів разів, оскільки він розвивається експоненціально на відміну від людського мозку котрий розвивається багато тисяч років.

Більше того, така здатність ШІ у розвитку не повинна залишатись без нагляду й саме тому за запитом ООН було створено орган високого рівня з питань консультації розвитку ШІ аби контролювати розвиток ШІ у світі й не дати йому вийти за межі здорового глузду.

ШІ потрібно створювати з блокуванням доступу до інтернету аби він працював локально та виконував те, що користувач скаже. Більше того, не можна в одному ШІ поєднувати функції декількох оскільки сильно навчаний ШІ здатен на справді, великі речі котрі людина не може собі уявити.

РОЗДІЛ II РОЗРОБКА ПРОЕКТУ ТА КОНЦЕПЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ «GEOMETRY ASTRO SURVIVAL»

2.1 Опис об'єкту та аналіз предметної області

2.1.1 Аналіз предметної області

Розробники створюють все більше ігор, змішуючи різні жанри та ігрові механіки, щоб захопити увагу користувачів і залучити їх до гри на якомога більш тривалий час. Вони також намагаються стимулювати користувачів витратити реальні гроші на покупку ігрової валюти, доповнень та іншого контенту, який доступний в грі.

Багато користувачів проводять значну кількість часу, граючи в ігри, а для деяких це стає способом заробітку або отримання популярності на стрімінгових сервісах або у таблицях лідерів. Проте, це може мати й негативні наслідки, такі як погіршення зору та залежність від комп'ютера або смартфона.

Незважаючи на це, ігри також сприяють розвитку корисних навичок, таких як уважність, пам'ять, усидливість, швидкість реакції, спрага до знань та робота у команді. Це залежить від кожної особи, і гра може приносити задоволення, заробіток, навички та знання, але також може мати й негативний вплив.

Отже, підсумовуючи всі ці фактори, можна зробити висновок, що від людини залежить, які наслідки приносить гра. Вона може приносити задоволення, заробіток, знання, кохання та корисні навички. З іншого боку, гра може призвести до болю, витрат, погіршення відносин з рідними та/або друзями, втрати базових навичок та інших негативних наслідків.

В кінцевому рахунку, гра є лише інструментом у руках людини, і вона сама вирішує, як ним користуватися.

Тип ігри залежить від розробника та його творчості. Існує багато жанрів ігор, таких як:

- RPG (рольові ігри), де учасники приймають конкретні ролі та створюють історію або діють у вигаданих ситуаціях.
- ACTION (екшн), який вимагає фізичних зусиль, таких як швидкість реакції та прийняття тактичних рішень.
- ROGULIKE, піджанр рольових ігор з випадковим створенням рівнів та перманентною смертю персонажа.
- SIMULATOR (симулятори), які точно відтворюють фізичні закони та властивості реального світу.
- SHOOTER (стрілялки), де головне завдання полягає в стрільбі по цілях.
- PLATFORMER (платформери), де гравець стрибає по платформах та подолає перешкоди.
- TOP DOWN, де камера розташована зверху і дивиться на ігрове поле.
- 2D, де графіка двовимірна.
- 3D, де графіка тривимірна.

Це лише декілька прикладів жанрів ігор, які можуть бути створені розробниками залежно від їхньої фантазії.

Ми зупинимося на комбінації декількох жанрів, а саме: Top Down 2D Rogulike Shooter.

2.1.2 Аналіз існуючих рішень

Play Market, App Store та інші платформи є цифровими магазинами, де можна завантажити різноманітний контент, такий як ігри, фільми, музику, книги та багато іншого. Цей контент може бути як безкоштовним, так і платним.

Зосередимося на пошуку ігор, оскільки вони є дуже популярними серед користувачів мобільних пристроїв. Фактично, майже всі грають у мобільні ігри, тому знайти їх нескладно. І хоча деякі люди можуть стверджувати, що не грають в ігри, це не зовсім правда.

Зараз нас цікавлять ігри з жанрів 2D, rogulike, shooter та top down. Ось кілька прикладів таких ігор:

- Pixel Dungeon[6] - це 2D гра з видом зверху, в якій гравець повинен пройти якомога глибше в підземелля. На його шляху він зустріне загадки, потужних ворогів та артефакти. Розробник відмовився від системи прокачки здібностей гравця і замість цього впровадив артефакти, які надають персонажу різні здібності. Однак, варто зазначити, що гра має обмежену ігрову зону та невелику кількість ворогів. Геймплей протікає повільно та розмірено, дозволяючи гравцю ретельно розглядати свої дії та стратегію.

Скріншот ігрового процесу Pixel Dungeon представлено на рисунку 2.1

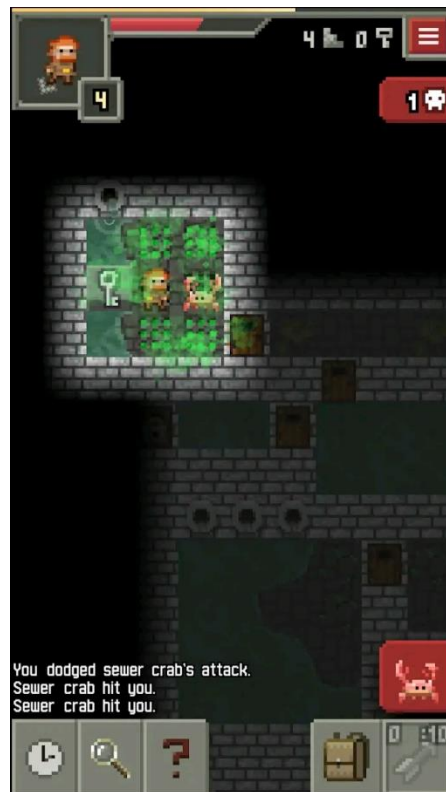


Рис. 2.1 – Скріншот ігрового процесу Pixel Dungeon

- Caves: Roguelike[2] - ця гра дуже схожа на Pixel Dungeon, але вона має систему прокачки рівня гравця та можливість вибору здібностей. Вона також має вид зверху та 2D графіку. Проте, вона відрізняється тим, що є більш динамічною, іноді не даючи гравцю часу на відпочинок.

Скріншот ігрового процесу Caves: rogulike представлено на рисунку 2.2



Рис. 2.2 – Скріншот ігрового процесу Caves: rogulike

- Apple Knight: Action Platformer[1] - ця гра виділяється своєю графікою, оскільки розробник витратив багато часу на створення ігрового світу, щоб зробити гру незабутньою для гравця. Хоча щодо жанру, це гра 2D, rogulike та platformer, в якій відсутні зброя та здібності. Ігрова зона поділена на рівні, які гравець повинен пройти по черзі, доки не досягне кінця гри.

Скріншот ігрового процесу Apple knight: action platformer представлено на рисунку 2.3



Рис. 2.3 - Скріншот ігрового процесу Apple knight: action platformer

- Space Horizon[7] - це шутер з видом зверху, в якому гравцеві потрібно досліджувати оточуюче середовище та збирати різні предмети, щоб покращити зброю та захисний костюм. Здібності гравця залежать від типу зброї, яку він використовує, і немає особливих навичок.

Скріншот ігрового процесу Space horizon представлено на рисунку 2.4



Рис. 2.4 – Скріншот ігрового процесу Space horizon

2.1.3 Аналіз засобів розробки для мобільних платформ

З метою розробки мобільних додатків все більш популярними стають інструменти, що дозволяють це зробити. Один з таких інструментів

- Unity[9], який є мультиплатформною програмою для розробки ігор, включаючи мобільні, з використанням мови програмування C#. Unity користується великою популярністю завдяки своїм перевагам:

- Вона умовно безкоштовна.
- Її легко освоїти.
- Вона має низький поріг входження.
- Має зручний інтерфейс.
- Unity має велику базу бібліотек та доповнень, створених користувачами.

Однак, є й деякі недоліки:

- Unity займає багато місця на комп'ютері.
 - Іноді вона може працювати повільно.
- Godot[8] - це відкритий мультиплатформний двигун для розробки ігор у 2D та 3D. Його загальна архітектура базується на концепції дерева зі спадкових "сцен". Кожен елемент сцени може стати самостійною сценою, що дозволяє легко змінювати архітектуру проекту та розширювати його елементи у будь-якому напрямку. Godot використовує мову програмування GDScript.

Переваги Godot:

- Він має відкритий код.
- Має зручну систему додатків.

- Підтримує мультиплатформність.

Недоліки Godot:

- Має високий поріг входження.
- Мала база застосунків, створених користувачами.
- Не може обробляти 2D та 3D одночасно.

- Unreal Engine[12] - це ігровий двигун, який має велику кількість вже вбудованих інструментів та ще більшу базу бібліотек та доповнень від користувачів. Він переважно використовується для 3D проектів, але, за необхідності, можна створити й 2D. Unreal Engine використовує мову програмування C/C++.

Переваги Unreal Engine:

- Має зручні інструменти.
- Має високу оптимізацію.
- Має найбільшу базу бібліотек та доповнень від користувачів.
- Має великий інструментарій для створення будь-яких 3D проектів.
- Unreal Engine користується популярністю серед великих компаній, що займаються розробкою ігор.

Недоліки Unreal Engine:

- Вимагає потужного комп'ютера.
- Має високий поріг входження.
- Має складний інтерфейс.
- Займає багато місця на комп'ютері.

- Game Maker Studio[5] - це платний, але легкий у освоєнні двигун для створення 2D проектів. Він користується популярністю серед indie розробників і має вбудований графічний редактор та зручний анімаційний редактор. Більшість розробок в Game Maker Studio використовують графічне програмування за допомогою так званих нод2.

Переваги Game Maker Studio:

- Має зручний інтерфейс.

- Має велику кількість вбудованих додатків.
- Має низький рівень входження.

Недоліки Game Maker Studio:

- Вимагає покупки ліцензії.

2.2 Вимоги до інформаційної системи

У виборі ігор є великий вибір, але іноді не вистачає того, що містить усе, що потрібно. З цією метою була розпочата розробка проекту, який поєднує жанри ігор, які раніше не були об'єднані. Хоча світ мобільних ігор безмежний, можливо, такої гри ще не було створено.

Тому було прийняте рішення створити проект, який об'єднує складні жанри та вимагає балансування. Цей проект передбачає велику кількість ворогів, велике ігрове поле, але все ж таки обмежене. Його особливості включають велику кількість зброї та здібностей, можливість зміцнювати ігрового персонажа та змагатися з друзями за перше місце у таблиці лідерів. Гравцям надається можливість змінювати свою тактику та використовувати різні здібності та зброю для досягнення кращого результату. Однак, незалежно від комбінації, все залежить від гравця - його реакції та можливості дивитися на декілька кроків вперед.

Тому для пересування гравця ми вибираємо джойстик, оскільки він відповідає всім критеріям - легкий, плавний і зручний[3].

Вороги повинні бути збалансованими, оскільки ніхто не захоче грати у гру, де навіть найслабші вороги можуть легко вбити персонажа. Перші вороги повинні бути слабкими і трохи повільнішими за гравця, щоб дати гравцю певну перевагу на початку гри.

З ворогів обов'язково має бути шанс випадання певних предметів: кристалу досвіду (experience crystal), аптечки (medkit) та монет (coin). Без цих предметів гра не матиме того самого ефекту, який задумав розробник. Кристал досвіду збільшує відсоток підвищення рівня персонажа, наприклад, на 10%. Для підняття рівня гравця потрібно зібрати 10 таких кристалів. Аптечка повністю відновлює здоров'я

гравця. Монети дають гравцю, наприклад, 5 монет, за які він може покращити свої базові характеристики у меню "Покращення" (Upgrades).

Система випадіння предметів включає математичну операцію для обчислення відсотка випадіння предмету після смерті ворога. Кожному предмету присвоєний програмно заданий відсоток, який може бути змінений за потреби. Наприклад, після знищення базового ворога запускається функція, яка обчислює відсоток випадіння предмету. Якщо цей відсоток співпадає з відсотком одного з предметів, то ворог залишає цей предмет на місці своєї смерті.

Здоров'я гравця повинно бути відображене, наприклад, у верхньому лівому куті, щоб гравці могли контролювати своє здоров'я і, при необхідності, уникати ворогів або шукати аптечки, які випали з ворогів[3].

Система ворогів передбачає появу ворогів навколо гравця. Після появи, вороги рухаються за заданим шаблоном, який полягає у наближенні до гравця з метою його знищення[3].

Якщо ворог з'явився на ігровому полі і не може догнати гравця, а з кожною секундою віддаляється від нього, такий ворог переміщується до пула (pool) всіх ворогів. Таким чином, ворог стає неактивним і не займає ігрове середовище та ресурси пристрою. Коли система потребує показати ворога, вона бере одного з тих, хто знаходиться у пулі об'єктів[3].

Для підвищення складності гравця можна додати зграї ворогів, які з'являтимуться у формі заповненої геометричної фігури, наприклад, кола. Після появи, зграя отримує напрямок руху, який визначається різницею між точкою появи і поточною позицією ворогів. Це створює несподіванку для гравця, змушуючи його активувати здібності або швидко тікати, часто ризикуючи. Зграя, так само як і інші вороги, переміщується до пулу об'єктів після віддалення від гравця, звідки вона буде взята наступного разу. Якщо у зграї недостатньо ворогів, вони будуть створені заново.

Тепер настав час для системи підняття рівня гравцем. В цій системі гравець матиме можливість згенерувати три випадкові здібності та регенерувати їх. Після генерації здібностей, вони будуть відображені у спеціальному меню, яке з'явиться після підняття рівня гравцем. Для кожної здібності буде представлено картинку,

короткий опис її функціональності (наприклад, пасивна здібність "Критичний удар" - збільшує шанс подвоєного пошкодження ворога) та позначка, яка вказуватиме, чи обрана ця здібність гравцем вперше.

Усі обрані здібності будуть відображені біля здоров'я персонажа у верхньому лівому куті екрана. Кількість доступних здібностей буде обмежена, щоб гравцям потрібно було розмірковувати над комбінацією здібностей, щоб протриматися у грі якомога довше.

Якщо гравець вибрав активну здібність, підтвердив свій вибір та здібність виявилася активною, то вікно з описом роботи здібності повинно з'явитися на екрані. Наприклад, якщо гравець вибрав здібність Dash, то після натискання на кнопку, яка з'явиться автоматично знизу-зліва, гравець швидко переміститься наперед на коротку відстань, уникаючи зустрічі з ворогами. Якщо в напрямку, в якому гравець активує Dash, є вороги, то вони будуть ігноровані, окрім тих, які знаходяться у точці, де гравець зупиниться. Після активації здібності вона буде на перезарядці. Для кожної активної здібності передбачені обмеження, щоб уникнути overpowered skill.

Щоб гравець мав бажання грати знову і знову, можна додати складностей, наприклад, боса. Бос - це дуже сильний ворог з багатою кількістю здоров'я та броні, але дуже повільний. Бос має особливі здібності та може стріляти по гравцю та створювати собі приспівників. У кожного боса є свої навички та здібності, але їх вигляд та спосіб пересування завжди будуть унікальними.

Система досягнень - це проста система, яка зчитує дані гри, такі як кількість знищених ворогів або час, проведений гравцем у грі. В кінці гри гравець отримає повідомлення про досягнення.

Дизайн гри повинен бути простим та естетичним, щоб користувачі могли насолоджуватися грою. Анімації можуть додати грі динаміки та зробити її більш привабливою для гравців. Система частинок може додати додаткової динаміки завдяки VFX ефектам.

Музика та звуки повинні бути динамічними та швидкими, щоб створити адреналін у гравців та збудити бажання знищити все, що бачать.

Звіт про гру буде представлений у двох екземплярах: один буде відображатися у вікні лідерів, а другий - у вікні кінця гри. У вікні кінця гри буде детальна інформація про кінцеві характеристики гравця, його досягнення у кількості знищених ворогів, піднятих монет, заробленого рівня та кількості часу, протягом якого гравець протримався живим. Ця інформація буде збережена на пристрої гравця та буде доступна для перегляду у локальній таблиці лідерів.

Після проведення аналізу були сформульовані наступні функціональні вимоги:

- Створення таблиці лідерів.
- Перегляд таблиці лідерів.
- Зручний та привабливий інтерфейс.
- Інтуїтивне керування гравцем.
- Підвищення рівня гравця.
- Отримання гравцем здібностей.

Для реалізації цих функціональних вимог необхідні наступні програмні засоби:

- Операційна система.
- Мова програмування для розробки.
- Система управління базою даних.

Технічне завдання на розробку гри наведено у додатку Б.

2.3 Розробка концепції

На основі сформульованих вимог були розроблені концепції та моделі інформаційної системи, які відповідають функціональним вимогам (рисунок 2.5)

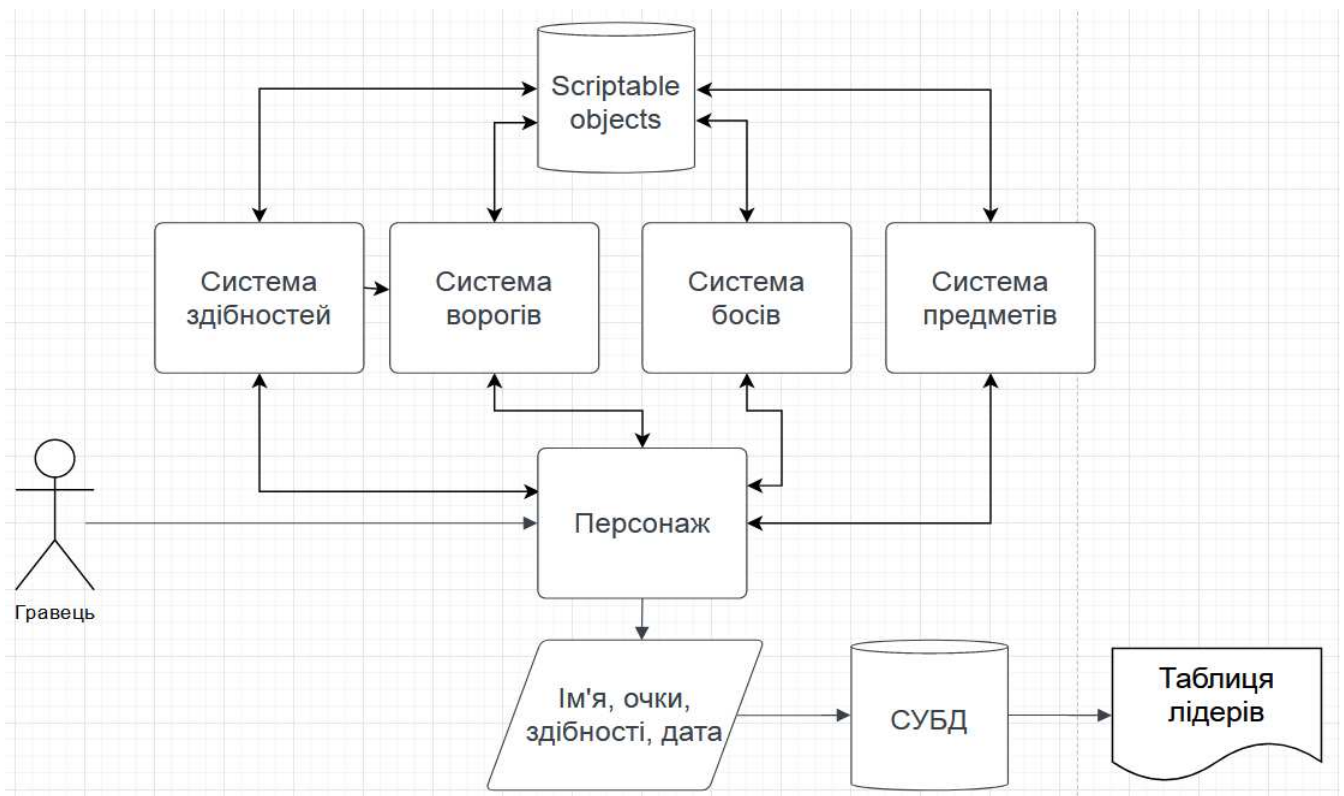


Рис. 2.5 – Схема запропонованої концепції інформаційної системи «Geometry Astro Survival»

Концепція гри передбачає наявність гравця та його персонажа. Гравець керує персонажем, який знищує ворогів і стає сильнішим, підвищуючи свій рівень та отримуючи нові здібності. Гра теоретично не має кінця, оскільки гравець не обмежений часом, а лише ігровим полем. Проте, розумне використання здібностей дозволяє персонажу гравця жити безкінечно, набираючи очки та піднімаючись у таблиці лідерів.

Гра включає декілька систем:

- Система здібностей відповідає за всі доступні здібності гравця та їх функціонал під час гри.
- Система ворогів відповідає за рух ворогів, їх поведінку при зіткненні з гравцем та способи їх знищення.
- Система предметів відповідає за предмети, які можуть випасти з ворогів, та їх функціонал після підняття гравцем.
- Система босів відповідає за поведінку босів, їх рух, логіку завдання пошкоджень гравцеві та їх поведінку при різних рівнях здоров'я.

Кожен ворог та бос оцінюється певною кількістю очок, яка може відрізнятися в залежності від типу ворога чи боса:

- базовий (1)
- живучий (2)
- швидкий (3)
- літаюча міна (2)
- троянське коло (спочатку 2, потім по 1)
- пила (3)
- боси (кількість очків, яку має кожен бос, залежить від тривалості гри гравцем. Чим довше гравець грає, тим більше очків мають боси)

Хоча кількість ворогів є обмеженою, гравцю не варто постійно від них тікати, оскільки вороги, які опинилися занадто далеко від гравця, автоматично переносяться у "pool" об'єктів, звідки вони знову з'являються поруч з гравцем.

Уся інформація про здібності гравця, ворогів, босів та предметів знаходиться у Scriptable objects, що дозволяє зменшити навантаження на базу даних.

На основі цієї концепції інформаційної системи буде розроблений технічний проект.

2.4 Загальносистемні рішення

2.4.1 Діаграма варіантів використання

У грі, яку використовують десятки, сотні, тисячі користувачів, кожен грає по-своєму. Хоча розробник задумав гру з однією ціллю - грати, не всі користувачі просто грають. Багато людей шукають натхнення та способи використання ігрової механіки або просто копіюють проекти, змінюючи лише графіку та/або музику.

Незважаючи на ці фактори, всі завантажують ігри та хоча б раз запускають їх. Розробники стараються створити зрозумілі відносини між гравцем, ворогами, об'єктами та ігровим світом. Професійним рівнем опису цих відносин є "Діаграма варіантів використання".

Діаграма варіантів використання є концептуальним зображенням системи під час її проектування та розробки. Вона містить акторів, варіанти використання та зв'язки між ними.

Діаграма варіантів використання представлена на рисунку 2.6

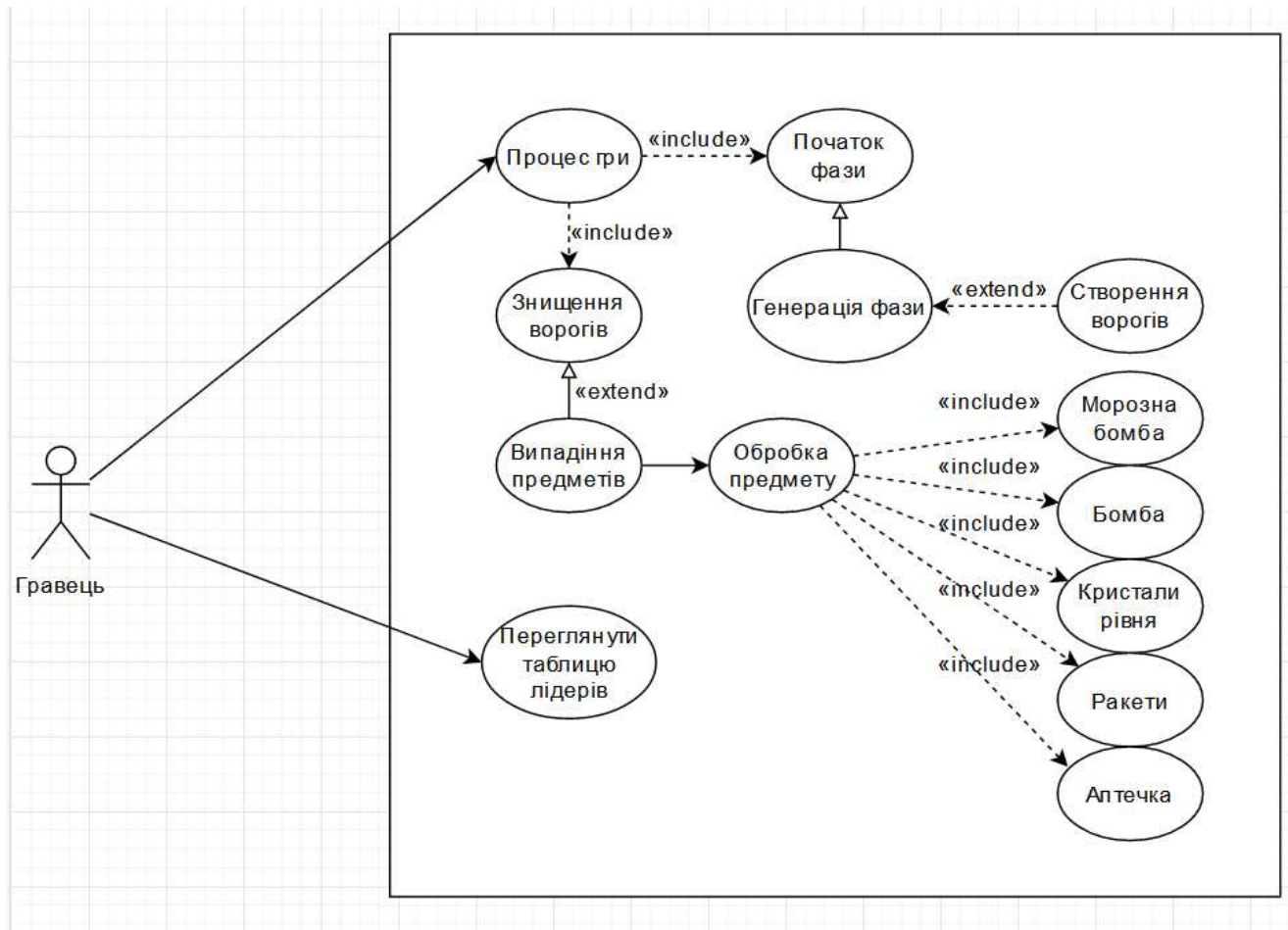


Рис. 2.6 – діаграма варіантів використання

Після початку гри гравець виходить у головне меню. Після натискання кнопки (Старт)"Start" гравець переходить до гри. Персонаж з'являється на ігровому полі, і разом з гравцем розпочинається фаза, під час якої вороги з'являються навколо гравця за межами камери. Гравець повинен їх знищувати, і після їх усунення може випасти предмет, який гравець може підняти, щоб стати сильнішим. Коли кількість знищених ворогів досягає певного рівня або заданої кількості ворогів, наступає нова фаза.

2.4.2 Специфікація варіантів використання

Стандартна структура опису варіанта використання включає наступні складові:

- Назва;
- Ідентифікатор;
- Короткий огляд;
- Учасники (головні та допоміжні);
- Попередні умови;
- Основний сценарій;
- Альтернативні сценарії;
- Постсценарій;
- Особливі вимоги.

Враховуючи всі компоненти структури специфікації, можна створити наступні специфікації варіантів використання (ВВ):

А) Специфікація ВВ "Відображення даних таблиці лідерів":

Ім'я: Відображення даних таблиці лідерів

ID: 0

Короткий опис: система виведе на екран таблицю лідерів.

Основний учасник: гравець

Допоміжні учасники: відсутні

Передумови:

- Потрібне підключення до Інтернету

Основний потік:

- ВВ починається, коли гравець натискає кнопку "Leaderboard"
- Система виводить локальну таблицю лідерів гравцеві

Постумови:

- Гравець натискає кнопку "Global"
- Система завантажує глобальну таблицю лідерів із бази даних

Альтернативні потоки:

- Гравець повертається в головне меню

Б) Специфікація ВВ "Змагальні цілі":

Ім'я: Змагальні цілі

ID: 1

Короткий опис: система дозволить гравцям конкурувати між собою.

Основний учасник: гравець

Допоміжні учасники: інші гравці

Передумови:

- Необхідне підключення до Інтернету

Основний потік:

- ВВ розпочнеться, коли гравець програє і введе свій нікнейм у відповідне поле.
- Натисканням кнопки "Back to menu" ініціює ВВ з ID - 0.

Постумови:

- Гравець почне нову гру без перевірки свого місця у рейтингу.

Альтернативні потоки:

- Гравець не зможе конкурувати з іншими, якщо не введе свій нікнейм.

В) Специфікація ВВ "Економія часу":

Ім'я: Економія часу

ID: 2

Короткий опис: гравець може зменшити тривалість гри, щоб відпочити від домашніх обов'язків або роботи.

Основний учасник: гравець

Допоміжні учасники: відсутні

Передумови: відсутні

Основний потік:

- ВВ почнеться, коли гравець запустить систему.

Постумови:

- Гравець завершить гру та вимкне систему.

Альтернативні потоки: відсутні

Г) Специфікація ВВ "Грати в гру":

Ім'я: Грати в гру

ID: 3

Короткий опис: ознайомлення з грою та отримання ігрового досвіду.

Основний учасник: гравець

Допоміжні учасники: ігровий персонаж, вороги, ігрові об'єкти

Основний потік:

- ВВ почнеться, коли гравець натисне кнопку "Start".
- Система розпочне виконання своїх функцій.

Постумови:

- Гравець керує персонажем, знищує ворогів, піднімає предмети та розвиває ігрового персонажа.

Альтернативні потоки:

- Гравець повертається в головне меню.

Д) Специфікація ВВ "Максимальна розвиненість персонажа":

Ім'я: Максимальна розвиненість персонажа

ID: 4

Короткий опис: досягнення найвищого рівня розвитку гравцем.

Основний учасник: гравець

Допоміжні учасники: ігровий персонаж, вороги, предмети

Основний потік:

- ВВ почнеться, коли активується ВВ з ID - 3.
- Гравець, керуючи ігровим персонажем, збирає кристали рівня, що збільшує його рівень та підвищує його навички.

Постумови:

- Гравець досягне максимального рівня у всіх своїх навичках.

Альтернативні потоки:

- Гравець програє.

Для правильного функціонування будь-якого з варіантів використання, гравець повинен переконатись, що його пристрій має достатньо заряду. Це необхідно для того, щоб пристрій не вимкнувся під час виконання конкретного варіанту використання та дозволив гравцеві насолоджуватися ним на повну катушку.

2.5 Рішення з інформаційного забезпечення

2.5.1 Діаграма класів

Діаграма класів становить статичне відображення структури моделі, де представлені статичні (декларативні) компоненти, такі як класи, типи даних, їх вміст і взаємозв'язки між ними.

Діаграма класів представлена на рисунку 2.7

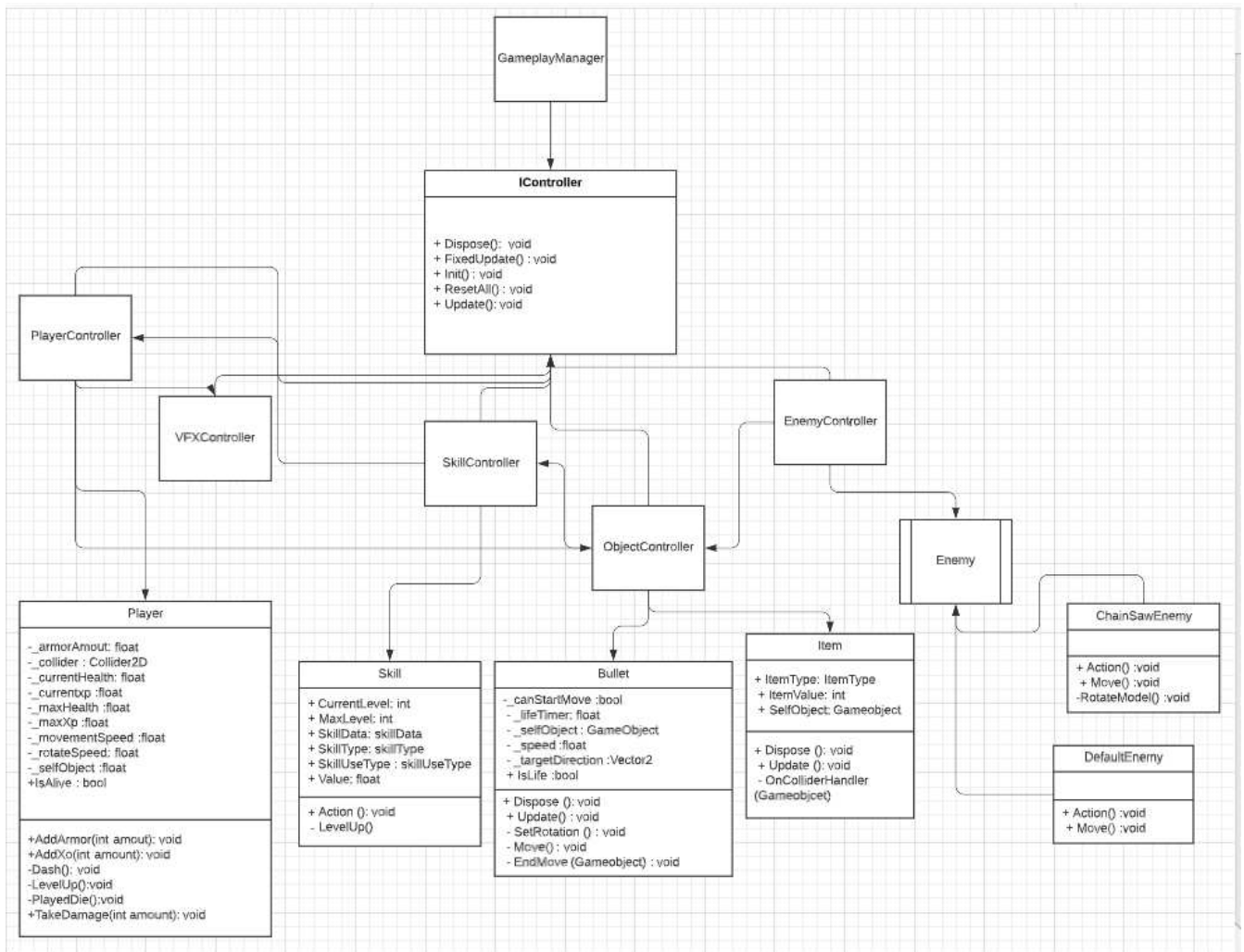


Рис. 2.7 – діаграма класів

2.5.2 Діаграма переходів станів

Діаграма переходів станів – це графічне зображення, що входить в стандарт мови UML, і призначене для відображення поведінки розроблюваної програмної

системи під впливом управлінських впливів ззовні. Застосовуючи діаграму переходів станів, можна моделювати подальший хід функціонування системи на підставі її попереднього та поточного стану.

Діаграму переходів стану поразки представлено на рисунку 2.8

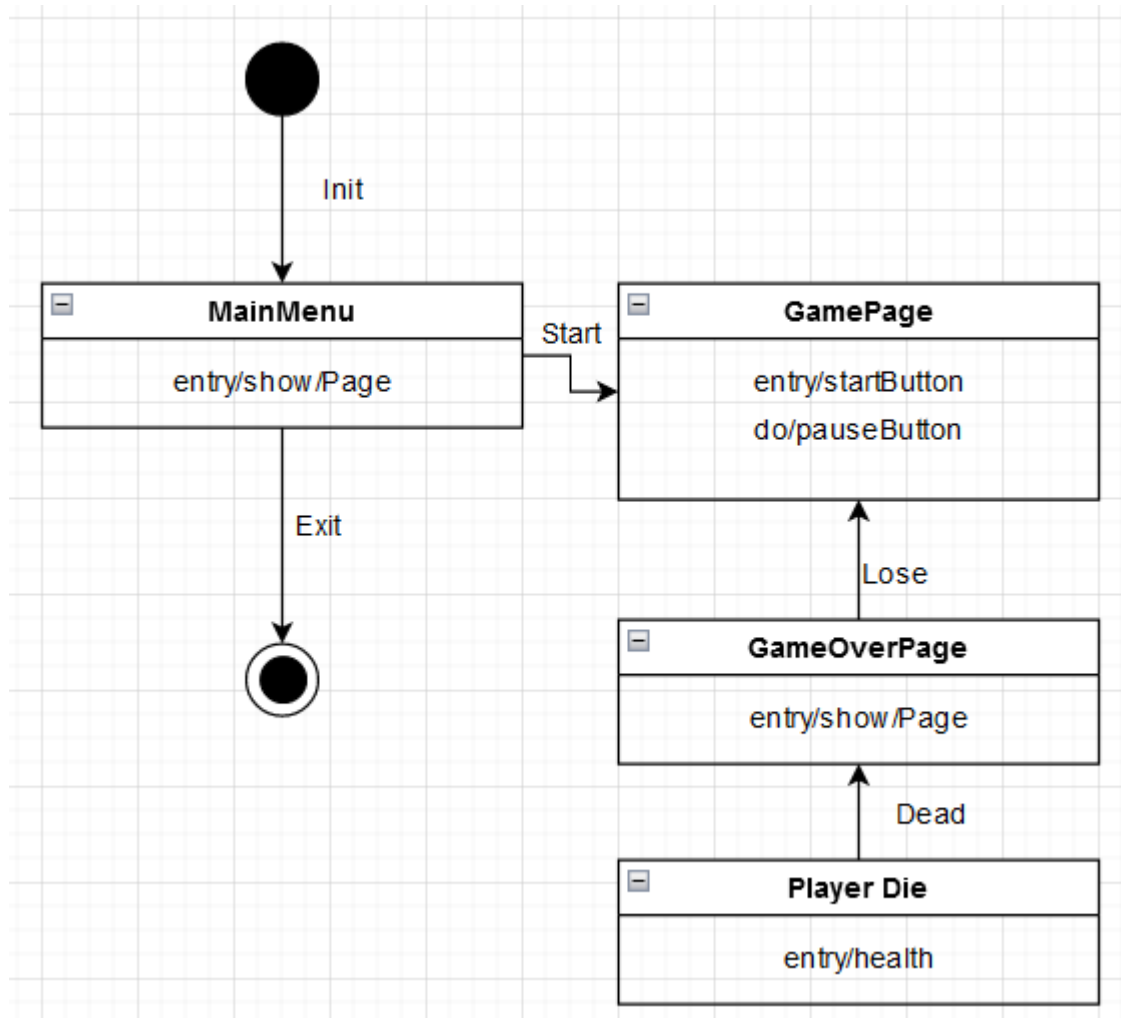


Рис. 2.8 – діаграма переходів стану поразки

На діаграмі переходів станів невдачі відображено хід гри, де після загибелі героя відображається виринаюче вікно із статистикою по матчу, і гравець може перейти до головного меню.

Діаграму переходів стану активності гравця представлено на рисунку 2.9

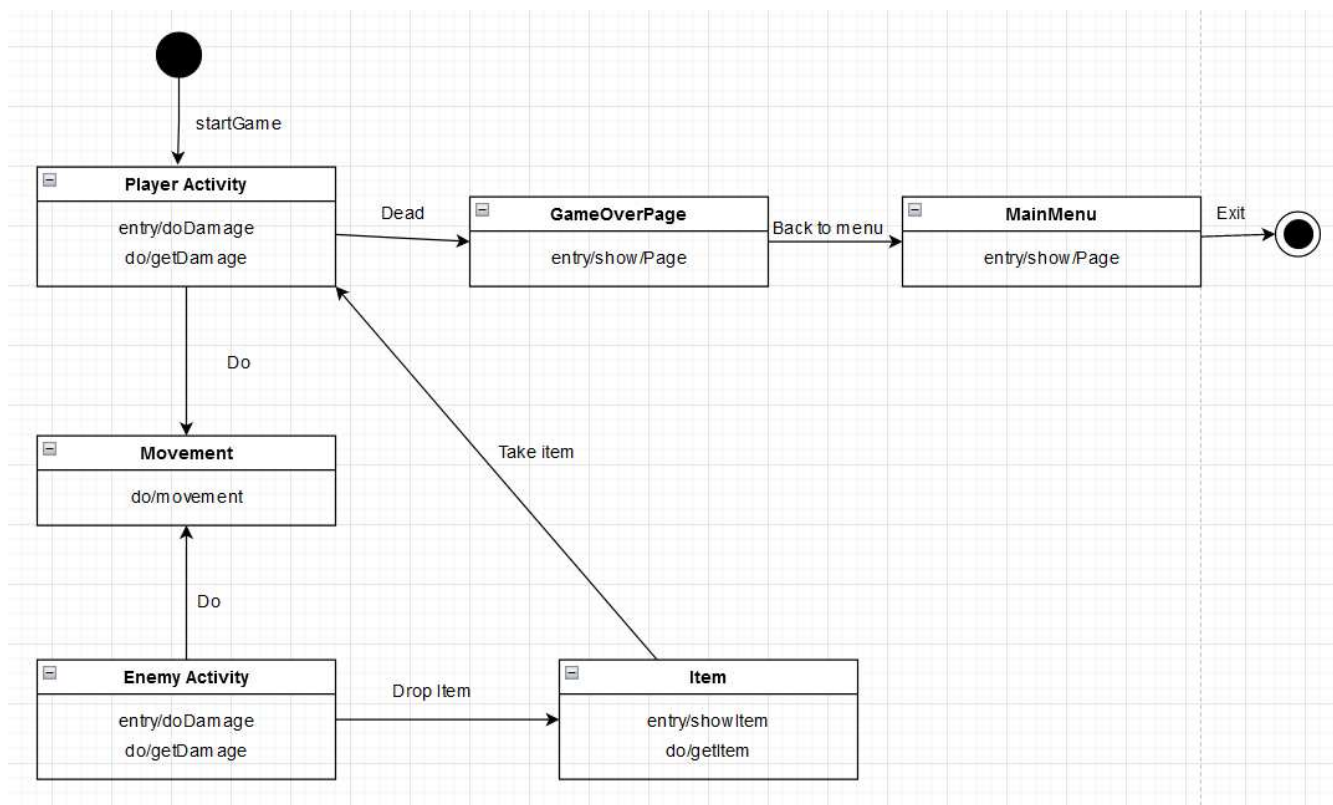


Рис. 2.9 – діаграма переходів станів активності гравця

На діаграмі переходів станів відображено активність гравця. Кожен персонаж у грі має свій власний метод для обробки кожної дії. Після загибелі основного героя відбувається поразка, і гра завершується.

Діаграма переходів станів процесу гри представлена на рисунку 2.10

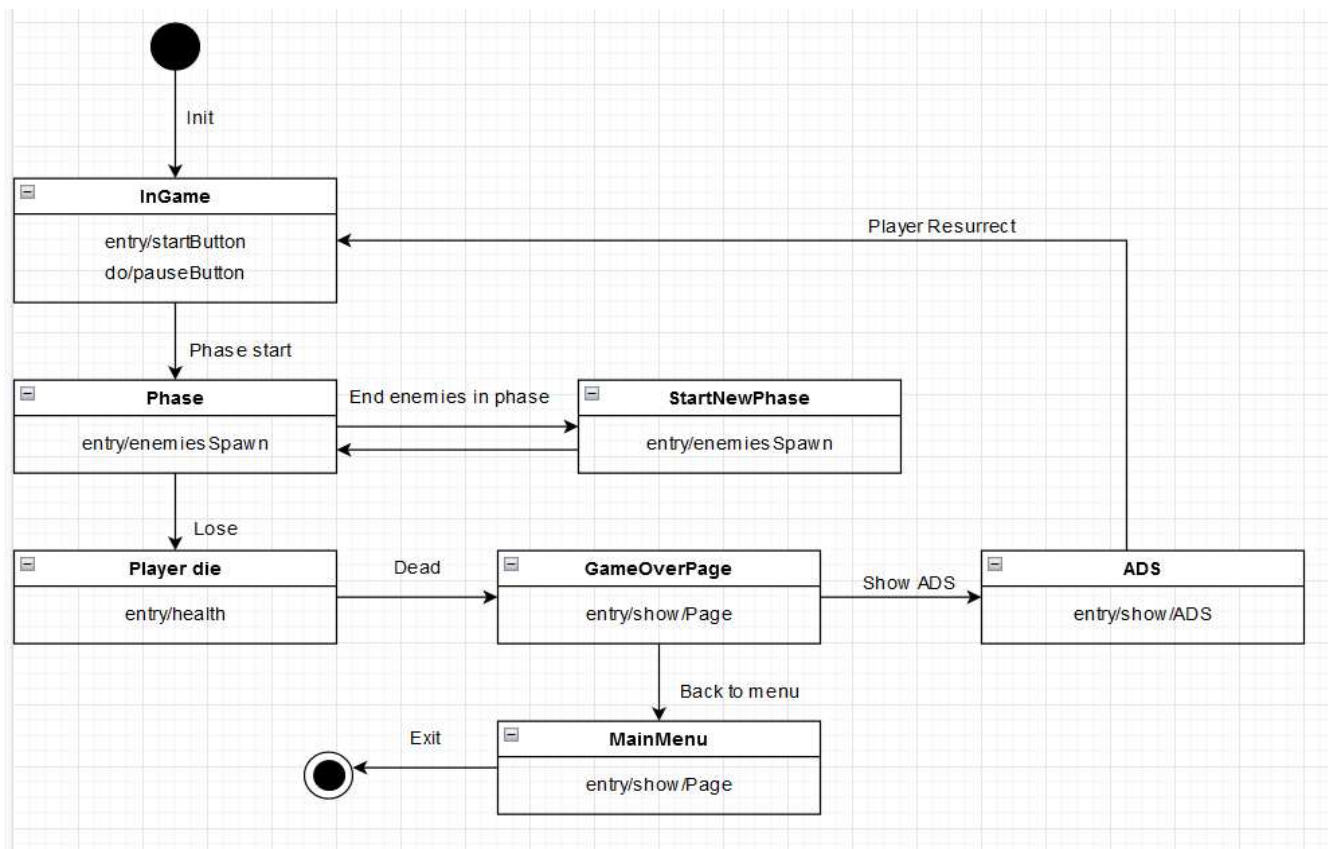


Рис. 2.10 – діаграма переходів станів процесу гри

На діаграмі переходів станів геймплейного процесу показано хід гри. Після запуску ініціюється процес генерації фаз, в результаті якого на сцені з'являються вороги та падають предмети. Після поразки гра завершується, але є можливість перегляду реклам і продовження гри.

2.5.3 Діаграма діяльності

Діаграма діяльності представлена на рисунку 2.11

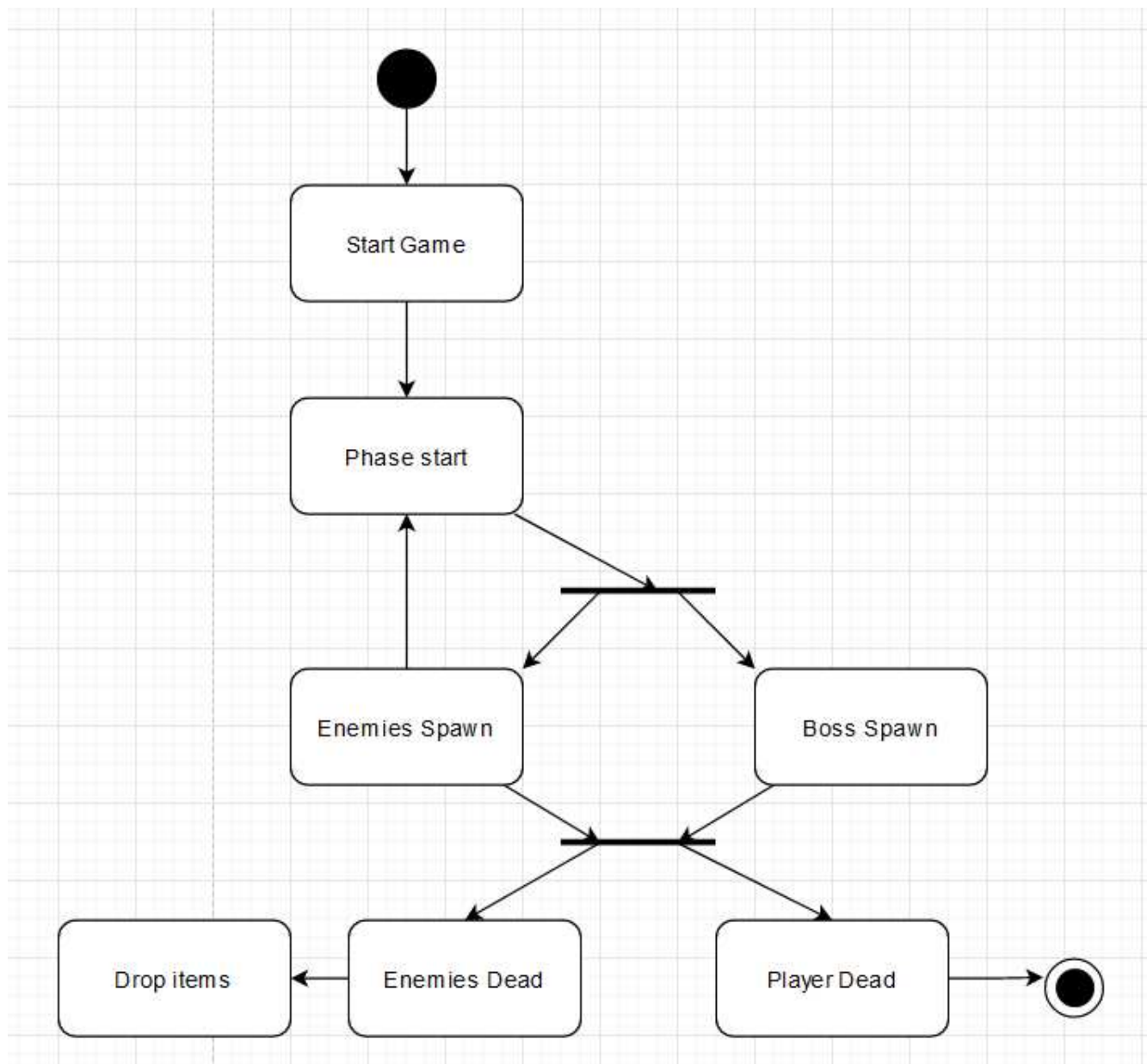


Рис. 2.11 – діаграма діяльності

Діаграма діяльності - це графічне відображення, що входить в стандарт мови UML. На ній демонструються дії та конкретна специфікація виконуваної поведінки. Вона представлена у вигляді координованого послідовного та паралельного виконання вкладених елементів, таких як дії та вкладені види діяльності. Ці елементи з'єднані стрілками, які передають потоки від виходів одного вузла до входів іншого. Загалом діаграми діяльності складаються з обмеженої кількості фігур.

2.5.4 Діаграма компонентів

Діаграма компонентів (Component diagram) відображає взаємозв'язки між модулями, які можуть бути логічними або фізичними, утворюючи систему, яку ми моделюємо.

Діаграма компонентів представлена на рисунку 2.12

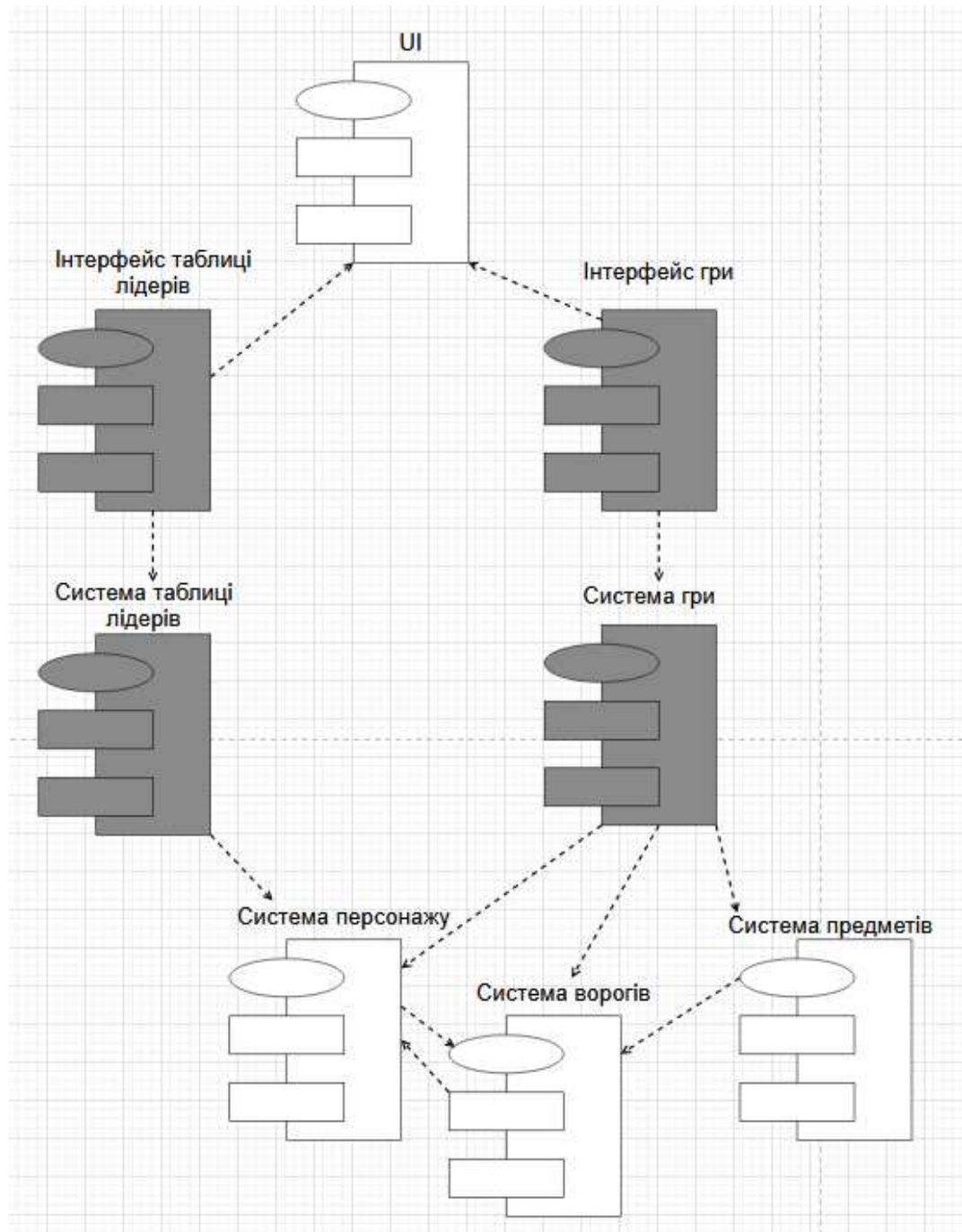


Рис. 2.12 – діаграма компонентів.

2.5.5 Діаграма розгортання

Діаграма розгортання у UML відображає обчислювальні вузли, компоненти та об'єкти, що виконуються на цих вузлах під час роботи програми.

Діаграма розвертання представлена на рисунку 2.13

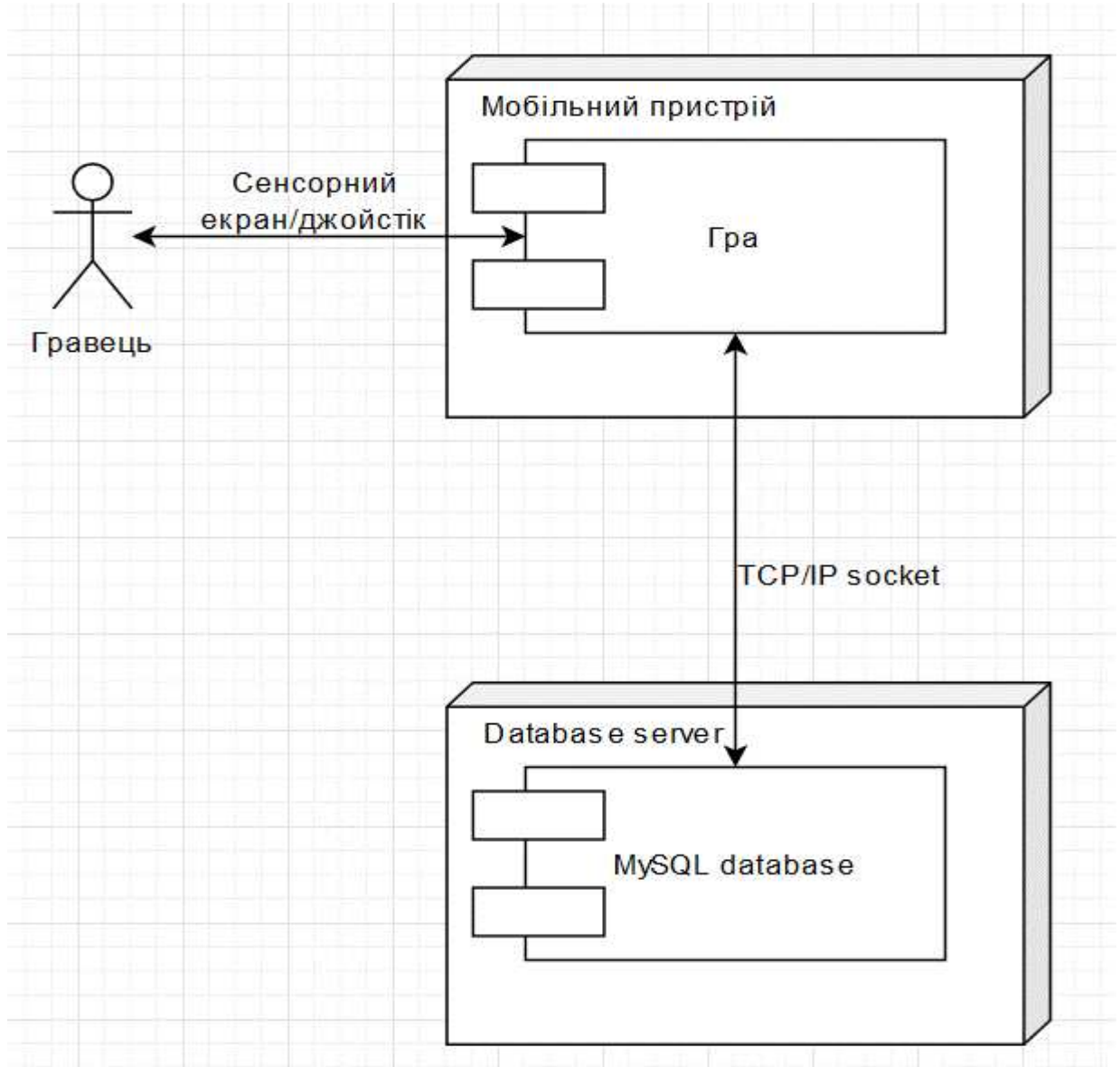


Рис. 2.13 - діаграма розвертання

2.6 Рішення з програмного забезпечення

Враховуючи переваги та недоліки різних програм для розробки ігор, можна зробити висновок, що Unity є оптимальним інструментом для створення мобільних ігор. Це пояснюється його великою бібліотекою, яка містить різноманітні матеріали

для гри, такі як графіка, допоміжні інструменти та музика. Unity дозволяє розробляти ігри навіть без глибоких знань у створенні 3D та 2D графіки.

Також важливо відзначити особливість Unity, якою є ScriptableObject. Це контейнер даних, який можна використовувати для збереження великих об'ємів інформації, незалежно від екземпляру класу. Використання ScriptableObject може сприяти ефективному управлінню пам'яттю у проєкті, особливо при наявності Prefab⁸, який зберігає незмінні дані в прикріплених скриптах.

C#[16](C Sharp) - це об'єктно-орієнтована мова програмування, розроблена інженерами компанії Microsoft у 1998 році. Створена для використання у платформі .NET, C# належить до сім'ї мов C і має подібний синтаксис. Мова підтримує статеву типізацію, поліморфізм, перегрузку операторів, делегати, атрибути, івенти та мову запитів LINQ.

У порівнянні з C++, C# відзначається кращою керованою пам'яттю, яка уникає витоку та інших проблем. Вибір мови упав на C#, оскільки:

- Вона є об'єктно-орієнтованою, простою і потужною мовою програмування, що дозволяє розробникам створювати багатofункціональні програми.
- C# відноситься до мов компільованого типу, маючи всі переваги цього підходу.
- Мова поєднує найкращі ідеї з сучасних мов програмування, таких як Java, C++, Visual Basic тощо.
- Завдяки різноманітній синтаксичній конструкції та можливості працювати з платформою .Net, C# дозволяє швидше, ніж будь-яка інша мова, розробляти програмні рішення.
- C# відрізняється надійністю та елегантністю.

Для реалізації таблиці лідерів потрібно створити базу даних, де будуть зберігатися усі рекорди гравців. Обрано phpMyAdmin як Систему Управління Базами Даних (СУБД) через її доступність, простий та інтуїтивний інтерфейс, великий функціонал, високу швидкість і зручність у маніпуляції даними.

РОЗДІЛ III РЕАЛІЗАЦІЯ ПРОЕКТУ АВТОМАТИЗОВАНОЇ СИСТЕМИ

Було розроблено технічне завдання для створення та описано в Додатку Б. Вихідний код програми подано в Додатку В.

В інформаційній системі реалізовано наступні функціональності:

1. Запит до СУБД та отримання таблиці лідерів, яке відбувається через спеціальне вікно, де таблиця відображена (рисунок 3.3).
2. Пересування гравця, яке виконується за допомогою джойстика (рисунок 3.2).
3. Знищення ворогів, що відбувається завдяки автоматичній зброї гравця, яка вистрілює у напрямку руху гравця, коли перед ним з'являється ворог.
4. Вибивання предметів, що реалізовано за допомогою псевдовипадковості, яка обчислюється після смерті ворога.
5. Збільшення рівня гравця та отримання могутніх здібностей, що відбувається через підняття рівня гравцем кристалів досвіду. Після досягнення 100% гравець може вибрати одну з трьох здібностей.
6. Можливість перегляду реклами для відродження персонажу, що використовує Unity Ads.
7. Обмеження ігрової зони: на кінцях ігрової зони присутній пустотний простір, у якому неможливо знаходитись. Якщо гравець не покине цю зону протягом 5 секунд, його повернуть до центру ігрової зони.

Графічний інтерфейс (UI) є важливою частиною будь-якої гри, оскільки впливає на перше враження гравця. Дизайн інтерфейсу має бути зручним, враховуючи фактори, такі як розмір кнопок для забезпечення комфортного використання гравцем. Кожна сторінка та спливаюче вікно інтерфейсу повинні мати велику назву для полегшення орієнтації гравця.

У проекті буде реалізовано три ігрові сторінки та чотири спливаючих вікна.

Інтерфейс сторінки головно меню представлено на рисунку 3.1[4]

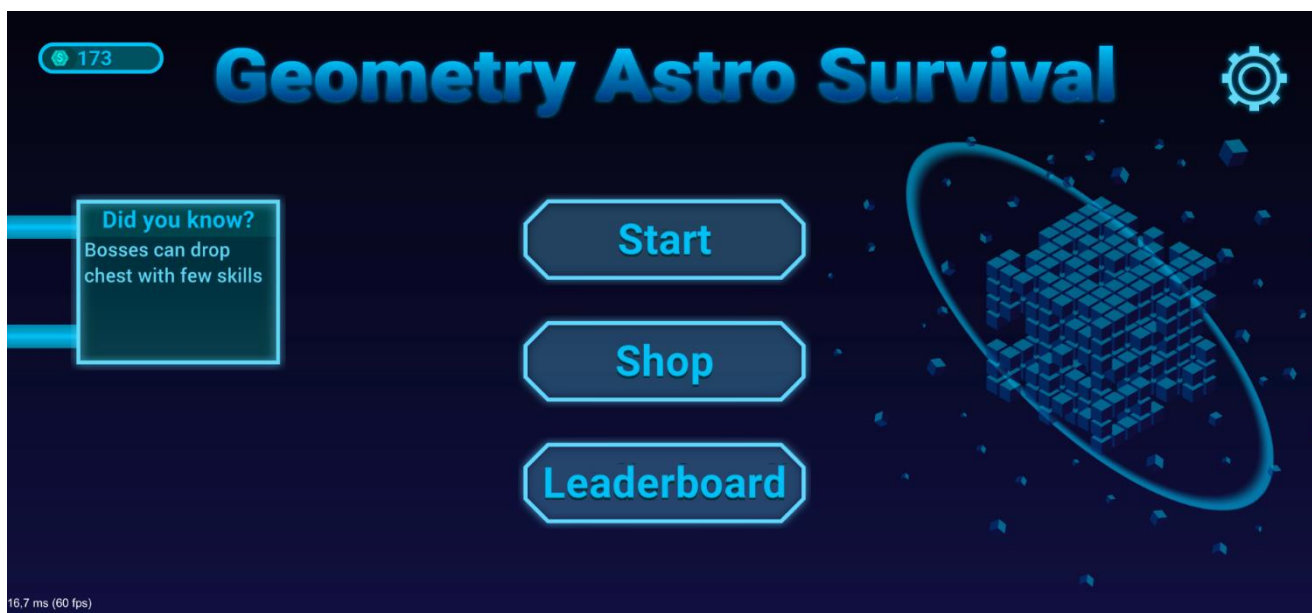


Рис. 3.1 – інтерфейс сторінки головного меню

Назва гри вражає своєю величезністю та привабливістю. Кнопки мають достатній розмір для комфортного використання, а шрифт обраний таким чином, щоб забезпечити легкість читання. Розташування кнопок оптимізоване для зручності натискання:

- "Start" веде до ігрового вікна.
- "Upgrades" перенаправляє до вікна прокачки гравця за допомогою грошей в грі (ця функція на даний момент не реалізована).
- "Leaderboard" відкриває вікно таблиці лідерів.

Інтерфейс ігрового вікна представлено на рисунку 3.2[4]

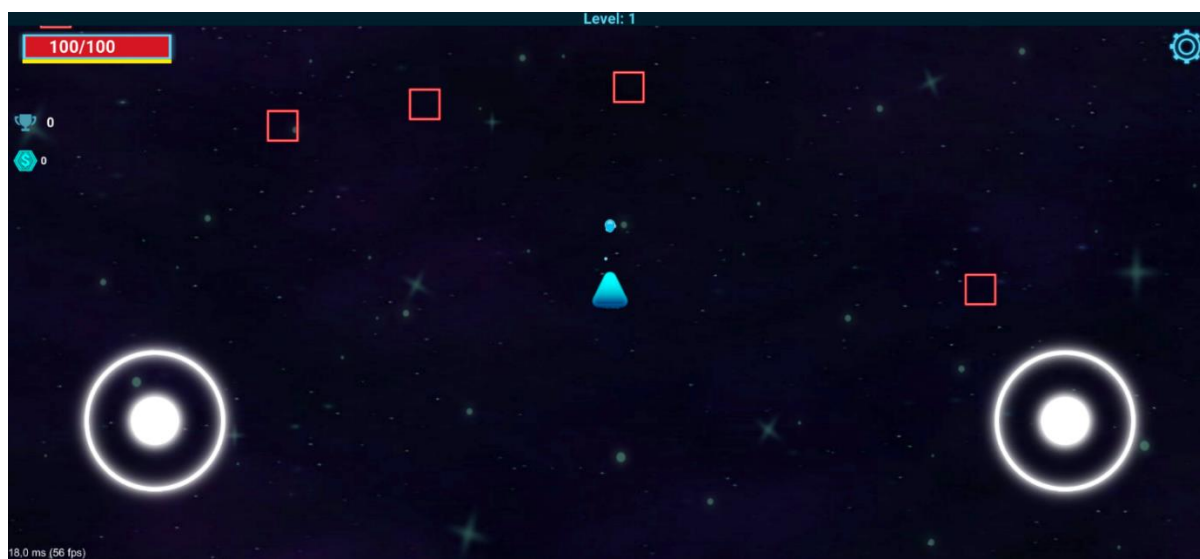


Рис. 3.2 – інтерфейс сторінки ігрового вікна

Управління персонажем реалізоване за допомогою великого джойстика. Кнопка паузи розташована у верхньому правому куті для можливості призупинення гри на необмежений період. Графічне представлення реалізовано для наступних елементів інтерфейсу:

- Здоров'я гравця відображається червоною смугою у верхньому лівому куті.
- Очки гравця відображаються у вигляді кубка зліва на екрані.
- Прогрес рівня гравця представлений смугою, яка заповнюється зверху і розтягнута на усю ширину екрану.

Текстове відображення містить наступні елементи інтерфейсу:

- Очки гравця.
- Прогрес рівня гравця (присутній або повинен бути).
- Здоров'я гравця.

Інтерфейс сторінки лідерів представлено на рисунку 3.3



Рис. 3.3 – інтерфейс сторінки лідерів

По-перше, гравці у таблиці представлені у вигляді списку, при цьому кожен об'єкт у списку має наступні текстові поля:

- Номер.
- Нікнейм гравця.
- Кількість очків.
- Дата занесення рекорда до таблиці.

Додатково у таблиці присутня кнопка "Повернутися у головне меню", щоб гравець міг зручно переміщатись між різними сторінками. Щоб відокремити локальну і глобальну таблиці лідерів, було введено дві кнопки, кожна з яких активує відповідну таблицю. Це дозволяє уникнути надмірного завантаження інтерфейсу та забезпечує зручне читання інформації.

Спливаюче вікно програшу представлено на рисунку 3.4[4]

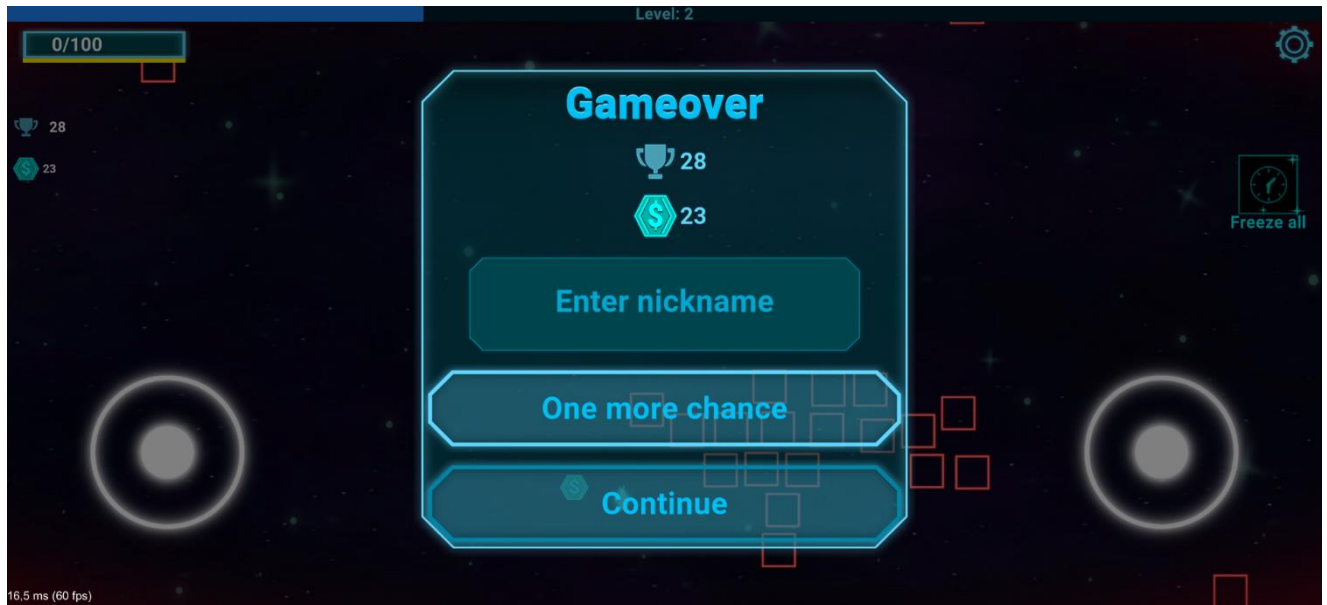


Рис. 3.4 – інтерфейс спливаючого вікна програшу

Має кнопку "Повернутися у головне меню" та кнопку для перегляду реклами, яка дозволяє відновити гравця з повним здоров'ям. На екрані відображено кількість набраних гравцем очок. Також присутнє поле для введення нікнейму гравця, щоб він міг потрапити до таблиці лідерів. Якщо гравець не введе нікнейм, його інформація не відобразиться в таблиці лідерів.

Спливаюче вікно паузи представлено на рисунку 3.5[4]



Рис. 3.5 – інтерфейс спливаючого вікна паузи

Гра припиняється, надаючи гравцеві можливість відпочити. Кількість отриманих очок гравцем відображена у текстовому форматі. У грі доступні дві кнопки: одна для повернення до гри, інша для переходу у головне меню.

Інтерфейс спливаючого вікна підняття рівня представлено на рисунку 3.6[4]

Рис. 3.6 – інтерфейс спливаючого вікна підняття рівня



Гра призупиняється, надаючи гравцеві час для обдумування та вибору однієї з трьох доступних здібностей, які представлені відповідними кнопками. У текстовому форматі надано:

- Назву здібності,

- Короткий її опис,
- Позначку "нової здібності" (яка відображається лише для здібностей, які гравець ще не обрав).

Після вибору здібності кнопка підтвердження стає інтерактивною та виконує подвійну функцію кнопки продовження гри, яку можна натискати. Кнопка перегляду реклами для зміни здібностей стає інтерактивною лише після натискання гравцем (перегляд реклами можливий лише один раз).

Інтерфейс спливаючого вікна відкриття скриньки представлений на рисунку 3.7[4]

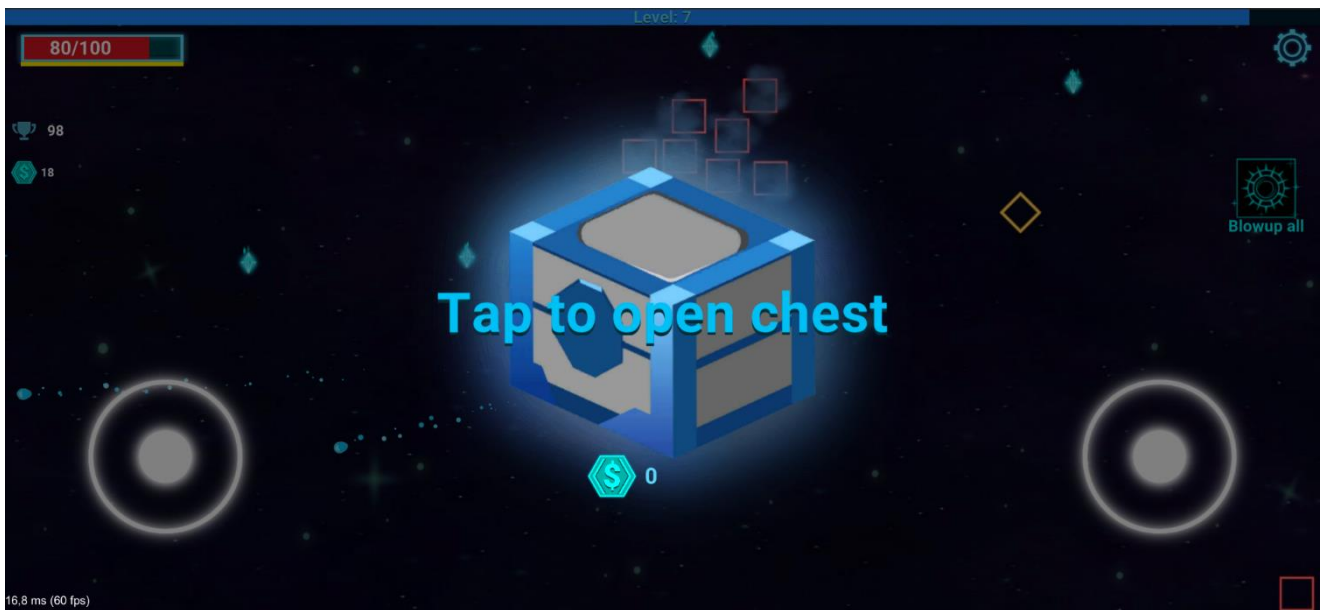


Рис. 3.7 – інтерфейс спливаючого вікна відкриття скриньки

Спливаюче вікно вибухає на екрані негайно після того, як гравець піднімає скриньку, і в цей момент гра призупиняється.

Інтерфейс спливаючого вікна відкритої скриньки представлений на рисунку 3.8[4]



Рис. 3.8 – інтерфейс спливаючого вікна відкритої скриньки

Розгорнута скринька презентує випадково отримані здібності, а також включає єдину кнопку для продовження гри, доступну за її натискання. Завдяки інтуїтивно зрозумілому інтерфейсу, користувач будь-якого рівня зможе легко орієнтуватися в грі.

Інструкція користувача представлена у додатку Г.

РОЗДІЛ IV РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ МОБІЛЬНОЇ ГРИ «GEOMETRY ASTRO SURVIVAL»

4.1 Вступ

Кожна створена та опублікована гра може принести її розробнику гроші. Їх кількість залежить від багатьох факторів, таких як:

- реклама
- цікавість гри
- різноманітність геймплею
- зацікавленість розробника у подальшій розробці гри
- спілкування з аудиторією

Так усі ці фактори впливають на прибуток і чим активніше веде себе розробник тим більше він заробить.

Після публікації головна ціль розробника заробити стільки, скільки він витратив на розробку гри й витрати можуть бути найрізноманітнішими, наприклад:

- найм художника для спрайтів
- найм аніматора
- витрати на повсякденні речі
- та інші

Усі ці фактори впливають на першу ціль розробника і чим більше розробник розроблює гру тим більше він хоче заробити оскільки він витрачає власний час та сили, а його гра це інформаційне відтворення його знань.

4.2 Розрахунок витрат на створення та експлуатацію «Geometry Astro Survival»

Розрахунок витрат на створення та експлуатацію мобільної гри залежить від багатьох факторів, таких як обсяг роботи, складність гри, обрані технології, ринкові стратегії та інші.

Взагалом можна навести наступні приклади витрати:

1) Розробка:

- Зарплата розробників: Витрати на оплату праці програмістів, дизайнерів, тестувальників тощо.
- Обладнання та програмне забезпечення: Купівля апаратного забезпечення та ліцензій на розробницьке програмне забезпечення.
- Маркетинг та реклама: Витрати на просування гри, створення рекламних матеріалів, участь в конференціях тощо.
- Ліцензії та авторські права: Вартість ліцензій на використання деяких технологій чи брендів, а також витрати на авторські права для гри.

2) Тестування та якість:

- Тестувальники: Оплата роботи тестувальників для перевірки функціональності та виявлення помилок.
- Засоби тестування: Покупка або використання засобів для автоматизованого та ручного тестування.

3) Інфраструктура:

- Хостинг: Вартість хостингу для збереження та надання доступу до гри.
- Сервери та бази даних: Купівля або оренда серверів та інфраструктури для збереження даних гри.

4) Підтримка та оновлення:

- Клієнтська підтримка: Витрати на обслуговування клієнтів та вирішення проблем.
- Оновлення та розширення: Розробка нових рівнів, персонажів, оновлень для гри.

5) Оплата магазинам додатків:

- Комісія магазинів: Витрати на комісії, які вимагають магазини додатків (наприклад, App Store, Google Play).

6) Фінансові витрати:

- Оплата платіжних систем: Витрати на оплату платіжних систем для обробки транзакцій.

7) Податки та інші обов'язкові витрати:

- Податки: Витрати на обов'язкові податки та збори.

Точно вирахувати витрати гри неможливо доки гра не буде завершена та випущена в магазині додатків оскільки купа факторів впливають на ціну розробки. Більше того якщо гра буде погана то окрім витрат розробник нічого не отримає. А коли гра випущена та приносить велику кількість грошей на її підтримку все одно виділяються кошти.

4.3 Економічна ефективність розробки і впровадження «Geometry Astro Survival»

На економічну ефективність розробки впливають численні фактори більшість з яких не можна цілком визначити на початку та під час розробки.

Ось декілька прикладів цих факторів:

1) Маркетинговий потенціал:

- Цільова аудиторія: Розуміння та привертання цільової аудиторії гри є ключовим. Якщо гра відповідає потребам та інтересам користувачів, це сприятиме успіхові на ринку.
- Маркетингові стратегії: Якісна реклама та маркетинг можуть визначити комерційний успіх гри. Важливо враховувати витрати на рекламу та їх віддачу.

2) Модель монетизації:

- Безкоштовний чи платний контент: Вибір правильної моделі монетизації впливає на прибутковість гри. Безкоштовний зазвичай використовується з інтегрованими покупками та рекламою, тоді як платний контент може забезпечити швидкий прибуток від продажу гри.

3) Витрати та бюджет:

- Розумне управління витратами: Ефективне управління бюджетом розробки та маркетингу є важливим аспектом економічної ефективності. Великі витрати не завжди гарантують великі прибутки.

4) Залучення гравців:

- Сприятливий ігровий досвід: Якість гри та задоволення користувачів від ігрового досвіду можуть сприяти утриманню гравців та залученню нових.
- Постійні оновлення та підтримка: Активне оновлення гри та надання підтримки можуть покращити ретенцію гравців.

5) Конкуренція на ринку:

- Конкурентна аналітика: Аналіз конкуренції може визначити те, наскільки ваша гра буде вигідно відрізнятися від інших на ринку.

6) Множник соціальних мереж та відгуки:

- Вплив соціальних мереж: Швидке розповсюдження через соціальні мережі може суттєво збільшити кількість завантажень гри.
- Відгуки та рейтинги: Позитивні відгуки та високі рейтинги у магазинах додатків можуть підвищити привабливість гри для нових гравців.

7) Довгострокова стратегія:

- Плани на майбутнє: Розробка довгострокової стратегії, включаючи плани на розширення та підтримку гри, може покращити економічну ефективність проекту на тривалий термін.

Від цих факторів залежить початкова ефективність розробки гри. Однак про гру можна на початку розробки сказати, що вона буде погана і потрібно перероблювати і цей фактор не залежить від програміста настільки, наскільки від дизайнера гри, тому що він говорить, що це буде за гра та як в неї грати й якщо придумати ідею гри недостатньо цікавою то вже на стадії розробки прототипу можна зрозуміти, що потрібно щось змінювати або взагалі зупиняти розробку.

Таким чином уся команда та компанія котра відповідає за розробку тим чи іншим чином впливає на її успіх. Кожен гвинтик важливий.

РОЗДІЛ V ОХОРОНА ПРАЦІ

5.1 Вступ

Усі ми сидимо за комп'ютерами, для когось це розважання, а для іншого робота. Але як із будь-яким видом діяльності – сидіння за комп'ютером має свої наслідки якісь просто неприємні, а інші довгострокові та болючі. Але більшість з них можна уникнути фізичними вправами та прогулянкою на свіжому повітрі, інші потребують втручання лікарів.

Метою цього розділу є проведення аналізу умов праці, технологічних процесів, обладнання та апаратури на робочому місці програміста. На підставі такого аналізу визначаються потенційно небезпечні зони, порівняння умов праці з нормативними вимогами, можливі аварійні ситуації, а також розробляються заходи для усунення чи обмеження можливих наслідків.

5.2 Аналіз шкідливих та небезпечних факторів

Робоче місце програміста може бути піддане ряду шкідливих та небезпечних факторів, які впливають на здоров'я та комфорт праці.

Ось декілька аспектів котрі при неправильному використанні або розташуванню можуть бути шкідливими та небезпечними:

1) Ергономіка робочого місця:

- Меблі та обладнання: Неправильно підібрані чи неекономічні меблі можуть призводити до неправильної постави та болей у спині.
- Монітор та клавіатура: Некоректне розташування монітора чи некомфортна клавіатура можуть викликати напруження в очах та руках.

2) Зорове навантаження:

- Блиск монітора: Сяйво та блиск монітора можуть спричиняти втомленість та дискомфорт для очей.

- Подовжений період роботи перед екраном: Довгий час перед монітором може викликати синдром сухого ока та інші проблеми з ознаками.

3) Фізичні та емоційні навантаження:

- Монотонна робота: Постійне програмування може викликати фізичне та емоційне виснаження.
- Стрес та тиск термінів: Велике завдання та обмежений час для його виконання можуть призводити до стресу.

4) Сидячий спосіб життя:

- Безперервна сидяча робота: Довготривала сидяча поза може призводити до проблем із спиною та обмежити фізичну активність.

5) Технічні аспекти:

- Використання застарілої або неякісної техніки: Застаріла та неякісна техніка може призводити до перебоїв у роботі та непередбачуваних ситуацій.

6) Монотонність та рутинність:

- Можливість виникнення втоми від рутинної роботи: Довготривала робота над одноманітними завданнями може викликати втомленість та зниження продуктивності.

Для уникнення небезпечних факторів на робочому місці програміста слід приділяти увагу організації робочого простору. Використовуйте ергономічні меблі та обладнання, щоб забезпечити правильну поставу і зменшити навантаження на тіло. Правильно розташовуйте монітор, клавіатуру та мишу для зменшення напруження.

Важливо забезпечувати регулярні перерви для відпочинку очей та тіла, а також виконувати короткі фізичні вправи. Змінюйте позицію за робочим столом, використовуючи стіл-стійку або столічку для більшої варіативності. Розподіліть робочий день так, щоб включати різноманітні завдання та уникати монотонності.

Окрему увагу слід приділяти освітленню. Використовуйте природне світло та додаткове освітлення для зменшення втоми та поліпшення робочих умов.

Регулярно оновлюйте апаратне та програмне забезпечення для забезпечення безпеки та виявлення можливих загроз у кіберпросторі.

Додатково, для підтримки психічного здоров'я, використовуйте методи релаксації, такі як медитація та глибоке дихання. Плануйте робочий день, включаючи перерви та конкретні завдання, для ефективного управління часом та уникнення стресу. Шукайте підтримку в колег, друзів та родини, якщо виникає стрес чи напруга. Такий підхід допоможе створити безпечне та комфортне робоче середовище для програміста, сприяючи його здоров'ю та продуктивності.

5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій

Робота за комп'ютером супроводжується тривалими зоровими навантаженнями, що може негативно впливати на здоров'я очей. Забезпечення правильного освітлення робочого місця має позитивний вплив на психофізіологічний стан людини та сприяє підвищенню ефективності та працездатності. Досягнення оптимальних умов роботи включає в себе природне освітлення в світлий час доби та адекватне штучне освітлення в темний період.

Згідно з вимогами ДБН В.2.5-28-2006 "Природне і штучне освітлення", робочі зони повинні відповідати нормативам, затвердженим Міністерством будівництва, архітектури та житлово-комунального господарства України. Також, робочі зони, де може виникнути небезпека при відключенні штучного освітлення, мають бути обладнані аварійним освітленням.

Для забезпечення необхідних умов для зорового комфорту, система освітлення повинна враховувати рівномірність освітлення, оптимальну яскравість, відсутність відблисків, правильний контраст, правильну кольорову гамму, відсутність стробоскопічного ефекту та пульсації світла.

Робоче місце для розробки дипломної роботи освітлюється природним світлом через вікно та штучним світлом в темний період. Урахуванням різноманітних факторів, таких як переадаптація ока та відповідність рівнів освітленості, можна забезпечити оптимальні умови для здоров'я та продуктивності.

5.4 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів при роботі з персональним комп'ютером

5.4.1 Заходи щодо зменшення впливу

Для зменшення впливу шкідливих та небезпечних факторів при роботі з персональним комп'ютером рекомендується впроваджувати наступні заходи:

1) Ергономічне робоче місце:

- Правильне розташування обладнання: Забезпечте оптимальне розташування монітора, клавіатури та миші для зручності та зменшення навантаження на тіло.
- Стіл та стілець: Вибирайте ергономічний стіл та стілець, щоб підтримувати правильну поставу та зменшувати стрес на спині.

2) Освітлення:

- Природне та штучне освітлення: Максимально використовуйте природне світло та налаштовуйте штучне освітлення згідно з нормативами для забезпечення оптимальних умов.

3) Очі:

- Регулярні перерви для очей: Встановлюйте періодичні перерви для відпочинку очей, фокусуючись на віддалених об'єктах, щоб зменшити напругу.
- Антиблікові фільтри: Використовуйте антиблікові фільтри для моніторів, щоб знизити вплив блиску та сяйва.

4) Фізична активність:

- Регулярні вправи: Виконуйте короткі фізичні вправи та розтяжки, щоб підтримувати кровообіг та зняти напругу.

5) Разнообразність задач:

- Різноманітність завдань: Варіюйте види роботи та включайте короткі перерви, щоб уникнути монотонності та підтримувати високий рівень уваги.

6) Коректне використання техніки:

- Оновлення програмного забезпечення: Регулярно оновлюйте оперативну систему та програмне забезпечення для забезпечення безпеки та ефективної роботи.

7) Оптимізація робочого часу:

- Ефективне управління часом: Плануйте робочий день, враховуючи перерви та відпочинок, для збереження енергії та продуктивності.

8) Системи безпеки та конфіденційності:

- Оновлення антивірусного програмного забезпечення: Забезпечте актуальність та ефективність антивірусного захисту для запобігання кіберзагрозам та захисту конфіденційності.

9) Підтримка психічного здоров'я:

- Психологічна підтримка: Забезпечте доступ до психологічної підтримки для працівників, особливо в умовах стресу або великої робочої навантаженості.

При виконанні усіх цих рекомендацій вдасться уникнути більшості шкідливих факторів, а виконання їх на регулярній основі збільшить фізичну силу та психічне здоров'є програміста. Особливо враховуючи стрес котрий майже кожного дня накоплюється можна точно сказати, що рано чи пізно програміст вигорить. Саме тому важливо слідкувати за своїм робочим місцем та процесом роботи, аби не повернутись до старих, шкідливих звичок.

5.4.2 Техніка безпеки при роботі на комп'ютері

Робота на комп'ютері може бути комфортною та продуктивною, якщо ви дотримуетесь правил техніки безпеки. Ось деякі поради та інструкції щодо безпеки при роботі на комп'ютері:

1) Ергономіка робочого місця:

- Регулюйте висоту стільця та столу так, щоб ваші лікті були на рівні столу, а коліна утворювали прямий кут.
- Монітор повинен бути на рівні вашого ока, і ви повинні дивитися прямо на екран, щоб уникнути надмірної напруги для очей та шиї.

2) Перерви та розтяжки:

- Робіть короткі перерви кожні 20-30 хвилин для відпочинку очей і фізичних вправ.
- Виконуйте розтяжки для ший, рук та спини, щоб підтримувати гнучкість та зменшувати напругу м'язів.

3) Освітлення:

- Забезпечте природне та/або добре регульоване штучне освітлення для уникнення надмірного блиску на екрані та забезпечення оптимальних умов для очей.

4) Коректне використання комп'ютера:

- Використовуйте антиблікові екрани для монітора, щоб зменшити відблиски і саяво.
- Не розташовуйте монітор або екран так, щоб відблиски від вікна або освітлювальних приладів потрапляли на екран.

5) Правильне використання клавіатури та миші:

- Підтримуйте природню позу для рук при використанні клавіатури та миші, щоб уникнути травм та дискомфорту.
- Використовуйте мишу з оптичним сенсором, щоб уникнути дротових мишей, які можуть створювати зайвий непорядок на робочому столі.

6) Зберігання кабелів:

- Закріплюйте кабелі так, щоб вони не перешкоджали рухам та не створювали небезпеки спотикання.

7) Комфортне та безпечне робоче середовище:

- Вибирайте зручні меблі та організовуйте простір так, щоб уникнути загроз для безпеки та комфорту.

У змісті вищевказаних порад та інструкцій щодо техніки безпеки при роботі на комп'ютері виявляється важливість створення ергономічного та безпечного робочого середовища. Дотримання цих принципів сприяє покращенню комфорту, збереженню здоров'я та підвищенню продуктивності працівників, які витрачають значний час за комп'ютером.

Остаточно, дотримання рекомендацій щодо техніки безпеки при роботі на комп'ютері не лише сприяє збереженню здоров'я та комфорту працівників, але й забезпечує стійку та ефективну роботу в сучасному інформаційному середовищі.

РОЗДІЛ VI ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

Охорона навколишнього середовища для програміста визначається не лише фізичними, а й ергономічними та культурними чинниками. Загалом можна визнати, що сучасні умови роботи програміста мають свої позитивні та викликові аспекти з точки зору охорони навколишнього середовища.

Позитивні аспекти:

- Ергономіка робочого місця: Сучасні стандарти дозволяють створювати комфортне та ергономічне робоче середовище для програмістів, що сприяє збереженню фізичного та психічного здоров'я.
- Використання технічних засобів: Програмісти мають доступ до високоякісної техніки та програмного забезпечення, яке полегшує роботу та покращує їх продуктивність.
- Дистанційна робота: Можливість виконання роботи віддалено дозволяє програмістам створювати комфортне робоче середовище відповідно до своїх власних вподобань та потреб.

Викликові аспекти:

- Вплив на здоров'я: Довготривала робота за комп'ютером може впливати на зір, м'язи та загальний фізичний стан.
- Психологічний стрес: Високий темп роботи, термінові проекти та велика відповідальність можуть викликати стрес та втому.
- Безпека даних: Захист від кіберзагроз та конфіденційність інформації вимагають постійного вдосконалення та уваги.

Загалом, оцінка охорони навколишнього середовища для програмістів є багатогранною. Важливо підтримувати баланс між продуктивністю та збереженням здоров'я, вдосконалюючи технічне та організаційне середовище роботи.

6.2 Забруднення навколишнього середовища

Робота програміста не є прямим джерелом значущого забруднення навколишнього середовища у фізичному розумінні, адже вона не пов'язана з великими викидами шкідливих речовин або виробництвом небезпечних відходів, як, наприклад, у виробничих галузях. Проте, існують інші аспекти, які можуть впливати на екологію:

- **Енергоспоживання:** Робочі станції та сервери, використовувані програмістами, вимагають електроенергії. Залежно від джерела цієї енергії, може бути вплив на рівень викидів CO₂ та інші шкідливі речовини, якщо енергія генерується з використанням вугільного палива.
- **Електронні відходи:** Використання та оновлення обладнання, такого як комп'ютери, монітори, та інші електронні пристрої, може призводити до утворення електронних відходів, які, якщо неправильно утилізовані, можуть негативно впливати на довкілля.
- **Робочі поїздки:** Якщо програмісти використовують автомобілі для робочих поїздок, це може призводити до викидів вуглеводнів та інших забруднюючих речовин.
- **Використання ресурсів:** Виробництво комп'ютерів та інших технічних засобів потребує видобутку рідкісних металів і інших природних ресурсів, що може мати вплив на екосистему та природні резерви.

Враховуючи ці фактори можна зробити висновок, що програміст не є прямим джерелом забруднення навколишнього середовища, якщо тільки говорити про роботу програмістом, але беручи за увагу людину взагалом то ми забруднюємо навколишнє середовище викидами небезпечних речовин в атмосферу, воду та землю.

6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища

Зменшення забруднення навколишнього середовища в контексті роботи програмістів може включати різні заходи, спрямовані на мінімізацію негативного впливу на екосистему та зменшення викидів забруднюючих речовин. Ось деякі можливі заходи:

- 1) Використання енергоефективного обладнання:
 - Підтримка та просування використання комп'ютерів, серверів та іншого обладнання з низьким енергоспоживанням.
- 2) Відповідальне утилізація електронних відходів:
 - Забезпечення правильної утилізації старого обладнання через організації, що займаються переробкою електроніки, для уникнення негативного впливу електронних відходів на довкілля.
- 3) Промоція віддаленої роботи:
 - Сприяння можливостям віддаленої роботи, яка може допомогти зменшити кількість робочих поїздок та відповідно викидів шкідливих речовин.
- 4) Використання відновлюваних джерел енергії:
 - Підтримка енергосистем, які використовують відновлювані джерела енергії, такі як сонячна чи вітрова енергія.
- 5) Працевлаштування в екологічно чистих офісах:
 - Створення офісних просторів, які використовують екологічно чисті матеріали та технології, з максимальним енергозбереженням.
- 6) Ініціативи щодо зменшення споживання ресурсів:
 - Популяризація культури використання та вибору екологічно чистих товарів та послуг серед працівників.
- 7) Участь у еко-проектах та програмах:
 - Залучення до екологічних ініціатив, які спрямовані на зменшення впливу технологій на природу.

Ці заходи можуть допомогти програмістам та ІТ-компаніям зменшити свій екологічний слід і сприяти створенню більш екологічно чистого робочого середовища.

ВИСНОВОК

У магістерській роботі розроблено інформаційну систему мобільної гри «Geometry Astro Survival» з використанням штучного інтелекту. У початкових етапах розробки було проведено аналіз предметної області, вивчено процес та структуру створення ігор, а також виконано дослідження об'єкта, здійснено аналіз інформаційних процесів і сформульовано вимоги. Результатом цього процесу стали концепція мобільної гри та технічний проект, які були розроблені з використанням сучасних програмних та технічних засобів, таких як об'єктно-орієнтована мова програмування C#, ігровий двигун Unity і система управління базами даних phpMyAdmin, з орієнтацією на використання штучного інтелекту.

Під час реалізації проекту мобільної гри "Geometry Astro Survival" було впроваджено ряд функцій, таких як запит до системи управління базами даних для отримання таблиці лідерів, керування рухом гравця, боротьба з ворогами, взаємодія з предметами, підняття рівня гравця та отримання могутніх здібностей. Крім того, гравці можуть переглядати рекламу для відродження персонажа та обмежувати ігрову зону. Це дозволило створити захоплюючу, динамічну гру, яка є не тільки цікавою, але й наповненою несподіванками завдяки використанню штучного інтелекту.

СПИСОК ВИКОРИСТОВАНОЇ ЛІТЕРАТУРИ

1. Apple knight: Action Platformer. URL: <https://play.google.com/store/apps/details?id=online.limitless.appleknight.free> (дата звернення 12.03.2022).
2. Caves (rogulike) . URL: <https://play.google.com/store/apps/details?id=thirty.six.dev.underworld> (дата звернення 12.03.2022).
3. Developing 2D Games with Unity: Independent Game Programming with C# Jared Halpern New York, NY, USA (дата звернення 18.03.2022).
4. Game design basics: How to start creating video games. URL: <https://www.cgspectrum.com/blog/game-design-basics-how-to-start-building-video-games> (дата звернення 10.04.2022). – Текст: електронний.
5. Game maker Studio — огляд програми. URL: <https://public-pc.com/game-maker-studio-obzor-programmyi/> (дата звернення 12.03.2022). – Текст: електронний.
6. Pixel dungeoun. URL: <https://play.google.com/store/apps/details?id=com.watabou.pixeldungeon> (дата звернення 12.03.2022).
7. Space Horizon — Космічне виживання, вид згори. URL: <https://play.google.com/store/apps/details?id=com.IndieGamesDev.SpaceHorizonSurvival> (дата звернення 12.03.2022).
8. Годот краще, ніж Unity для розробки гри? Які плюси та мінуси? URL: <https://otvetus.com/godot-luchshe-chem-unity-dlya-razrabotki-igri-kakovi-plyusi-i-minusi-31412> (дата звернення 12.03.2022). – Текст: електронний.
9. Двигун Unity — особливості, переваги та недоліки. URL: <https://cubiq.ru/dvizhok-unity/> (дата звернення 12.03.2022). – Текст: електронний.
10. Дослідження впливу шуму на працездатність. URL: <https://library.if.ua/book/86/6047.html> (дата звернення: 09.06.2022). - Текст: електронний.

11. Дослідження мікроклімату у виробничому приміщенні. URL: <https://zakon.rada.gov.ua/rada/show/va042282-99#Text> (дата звернення: 09.06.2022). - Текст: електронний.

12. Огляд нереального двигуна: плюси, мінуси та придатність. URL: <https://www.newgenapps.com/uk/blogs/unreal-engine-review-pros-cons-and-suitability/> (дата звернення 12.03.2022). – Текст: електронний.

13. Охорона праці на підприємстві. URL: <https://te.dsp.gov.ua/ohorona-pratsi-na-pidpryyemstvi-shho-potribno-znaty/> (дата звернення: 09.06.2022). - Текст: електронний.

14. Оцінка показників електробезпеки у виробничому приміщенні. URL: <https://www.sop.com.ua/article/756-elektrobezopasnost> (дата звернення: 09.06.2022). - Текст: електронний.

15. Оцінка якості освітлення. URL: <https://buklib.net/books/35234/> (дата звернення: 09.06.2022). - Текст: електронний.

16. Чи варто з'язуватись з С#. URL: <https://habr.com/company/vdsina/blog/522934/> (дата звернення 12.03.2022). – Текст: електронний.

17. Штучний інтелект. URL: https://www.wikiwand.com/uk/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82 (дата звернення 09.12.2023). – Текст: електронний.

18. Як «оживити» картинку: ТОП-6 ШІ для створення анімацій URL: <https://beincrypto.com/kak-ozhivit-kartinku/> (дата звернення 09.12.2023). – Текст: електронний

ДОДАТОК А – Технічне завдання

Вступ

Завданням даної дипломної роботи є розробка мобільної гри. В таблиці 1 представлено загальні відомості про систему.

Таблиця 1

Загальні відомості про систему

Повна назва системи	Мобільна гра «Geometry Astro Survival»
Умовне позначення системи	Geometry Astro Survival
Розроблювач	Малицький Роман Петрович
Замовник	Кафедра інформаційних управляючих систем та технологій

1. Підстави для розробки

Geometry Astro Survival розробляється на підставі завдання для дипломного проектування за спеціальністю 124 – «Системний аналіз».

2. Призначення розробки

Задовільнити потреби геймерів

Основними потребами геймерів є:

- цікава та захоплююча гра;
- можливість здобувати могутні навички;
- змагання за перше місце у таблиці лідерів;
- випробування себе на міцність і ін.

3. Мета створення системи

Основною метою є аналіз штучних інтелектів та створення Geometry Astro Survival є комбінація декількох жанрів ігор та введення незвичайних ігрових механік аби будь-який геймер міг знайти у грі те, що йому подобається, але у інших подібних проектах це не було реалізовано разом.

4. Вимоги до програми

4.1 Вимоги до функціональних характеристик

ІС, що буде спроектована, повинна виконувати наступні завдання:

- здійснювати відображення таблиці лідерів після запиту гравця до СУБД;
- здійснювати запис рекорда гравця після його програшу;
- здійснювати відображення локальної таблиці лідерів;
- коректне функціонування наступних механік:
 - пересування гравця (виконується за допомогою джойстика);
 - поява ворогів (виконується за допомогою особливого скрипта по обробці позицій гравця та можливих точок появи ворогів);
 - знищення ворогів (обробляє тип знищення (лазер, бомба, здібність та ін.));
 - випадіння предметів (залежить від типа знищення та кількості предметів у гравця (відноситься до ракет));
 - підймання предметів (обробка зіткнення двох колайдерів – предмета та гравця);
 - підвищення рівня;
 - набуття здібностей;
 - активація здібностей;
 - обмеження ігрової зони.

Змінювати параметри гри неможливо після зібрання apk файлу, але в Unity це цілком можна зробити відкривши спеціальний ScriptableObject а саме – GameplayData.

4.2 Вимоги до надійності

Повинні використовуватись передові патерни проектування систем задля швидкості, надійності та, головне, для подальшого розширення проекту та додавання нових функцій.

4.3 Умови експлуатації

Geometry Astro Survival розроблено на мові програмування C#, у якості СУБД було використано phpMyAdmin

4.4 Вимоги до інформаційної та програмної сумісності

Мобільний пристрій користувача повинен мати систему Android з версією не нижче шостого.

Вимоги до сервера:

- Процесор з частотою 550 МГц або більш потужний;
- Для підвищення швидкодії рекомендується 2-х процесорна платформа;
- Оперативна пам'ять 1 GB, рекомендується 4Gb.

4.5 Вимоги до технічного забезпечення

Технічні вимоги до мобільних пристроїв користувачів визначаються вимогами операційної систем.

5. Вимоги до програмної документації

Усі регламентуючі документи є елементами Geometry Astro Survival, які безпосередньо входять до цього. Регламент системи описує правила її ефективного використання. В таблиці 2 наведено склад документів.

Таблиця 2

Склад документів організаційно-методичного та експлуатаційного забезпечення

№	Назва	Стадія
1	Опис організаційної структури ІПГІА	Технічне проектування
2	Керівництво користувача	Реалізація системи

6. Стадії та етапи розробки

В таблиці 3 представлено стадії та етапи по створенню Geometry Astro Survival.

Таблиця 3

Стадії та етапи робіт по створенню АРМ СВК

№	Стадія	Документи
1	Аналіз предметної області	Звіт з описом предметної області
2	Системне проектування	Технічне завдання
3	Ескізне проектування	Ескізний проект
4	Технічне проектування	Технічний проект
5	Робоче проектування	Робочий проект

ДОДАТОК Б – Текст програми

Базовий клас ворога:

```
using UnityEngine;
using TandC.RunIfYouWantToLive.Common;
using System;

namespace TandC.RunIfYouWantToLive
{
    public abstract class Enemy
    {
        public event Action<Enemy> DestroyEvent;
        public event Action<Enemy> DeathEvent;
        public event Action<GameObject, Enemy> OnCollisionEvent;
        public event Action<Enemy> OnEndEnemyLifeTime;
        public GameObject SelfObject { get => _selfObject; }
        private GameObject _selfObject;
        private Animator _animator;
        protected Rigidbody2D _rigidbody2d;
        private Enemies _enemies;
        private OnBehaviourHandler _onBehaviourHandler;
        private SpriteRenderer _enemySprite;

        protected float _lifeTimer;

        private float _currentHealth;

        public int EnemyId { get; private set; }
        public float Health { get; private set; }
        public int Damage { get; private set; }
        public int GainedExperience { get; private set; }
        public float MovementSpeed { get; set; }
        public Enumerators.EnemyType EnemyType { get; set; }
        private bool _isAlive;
        private bool _wasKilled;
        protected float _attackCooldown = 1f;
        protected float _actualCooldown;
        public Rigidbody2D Rigidbody2D;
        public bool IsCanAction { get; set; }

        public Transform EnemyTransform { get; set; }

        private bool _isInitialize;
        protected Transform _playerTransform;

        private bool _isDamaged;
        private float _blinkTimer;
        private const float _blinkTime = 0.15f;
```

```

private Color _enemyColor;
public bool IsBoss = false;

private float _frozenStartenTime = 5f;
private float _frozenTimer;
private bool _isFroze = false;

public int DropChance;

public bool IsEndLifeTime = false;

private GameObject _lastTouchedObject;

public Enemy(Transform parent, Enemies data, Transform
playerTransform, Vector2 position, bool isBoss = false, bool init =
true)
{
    if (init)
    {
        _enemies = data;

        EnemyId = _enemies.enemyId;
        MovementSpeed = _enemies.movementSpeed;
        Damage = _enemies.damage;
        _attackCooldown = 1f;
        _playerTransform = playerTransform;
        _selfObject =
MonoBehaviour.Instantiate(_enemies.enemyPrefab, parent);
        _selfObject.transform.position = position;
        _selfObject.name = $"{_enemies.type}";
        _animator =
_selfObject.GetComponent<Animator>();
        _rigidbody2d =
_selfObject.GetComponent<Rigidbody2D>();
        this.Rigidbody2D = _rigidbody2d;
        _onBehaviourHandler =
_selfObject.GetComponent<OnBehaviourHandler>();
        _onBehaviourHandler.Trigger2DEntered +=
OnBehaviorHandler;
        _onBehaviourHandler.Trigger2DStay +=
OnCollisitonStayHandler;
        GainedExperience = data.experience;
        EnemyTransform = _selfObject.transform;
        Health = data.health;
        _enemySprite =
_selfObject.GetComponent<SpriteRenderer>();
        _enemyColor = data.enemyColor;
        _enemySprite.color = _enemyColor;
        _selfObject.transform.localScale =
data.scaleFactor;

        _isAlive = true;
        _isInitialize = true;
    }
}

```



```

        IsCanAction = true;
        IsBoss = isBoss;
        EnemyType = data.type;
        _lifeTimer = data.lifeTime;
    }
    else
        _isInitialize = false;
}
public void FrozeEnemy()
{
    _rigidbody2d.constraints =
RigidbodyConstraints2D.FreezeAll;
    _frozenTimer = _frozenStartTime;
    _isFroze = true;
}

public abstract void Action();

public void AddLifeTime(float time)
{
    _lifeTimer = time;
}

public void IncreaseParam(float mulriplier)
{
    Health += mulriplier;
    Damage = (int)(Damage * mulriplier);
}

public void Update()
{
    if (IsEndLifeTime)
    {
        //Dispose();
        return;
    }
    if (!_isAlive)
        return;
    if (!_isInitialize)
        return;

    if (!IsCanAction)
    {
        _actualCooldown -= Time.deltaTime;
        if (_actualCooldown <= 0)
        {
            Action();
        }
    }

    if (_isDamaged)
    {
        _blinkTimer -= Time.deltaTime;
        if(_blinkTimer <= 0)
        {

```

```

        SetNormalColor();
    }
}
_lifeTimer -= Time.deltaTime;
if (_lifeTimer >= 0)
{
    return;
}
OnEndEnemyLifeTime?.Invoke(this);
}

public abstract void Move();

public void FixedUpdate()
{
    if (IsEndLifeTime)
        return;
    if (!_isAlive)
        return;
    if (!_isInitialize)
        return;
    if (_isFroze)
    {
        _frozenTimer -= Time.deltaTime;
        if(_frozenTimer <= 0)
        {
            _rigidbody2d.constraints =
RigidbodyConstraints2D.None | RigidbodyConstraints2D.FreezeRotation;
            _isFroze = false;
        }
        return;
    }
    Move();
}

public void TakeDamage(float damage)
{
    if (!_isAlive)
        return;
    if (!_isInitialize)
        return;
    Health -= damage;
    SetDamageColor();
    if (Health > 0)
    {
        _blinkTimer = _blinkTime;
        _isDamaged = true;
    }
    if (Health <= 0)
    {
        Dispose();
        DeathEvent?.Invoke(this);
    }
}

```

```

    }
    private void SetNormalColor()
    {
        _enemySprite.color = _enemyColor;
    }
    private void SetDamageColor()
    {
        _enemySprite.color = Color.red;
    }

    private void OnCollisitonStayHandler(GameObject
collider)
    {
        //if (collider.tag == _selfObject.tag ||
collider.tag == "Border")
        //{
            //    return;
        //}
        //if(_lastTouchedObject != null)
        //{
            //    if(_lastTouchedObject == collider)
            //    {
                Debug.LogError("Same");
                return;
            //    }
        //}
        _lastTouchedObject = collider;
        //OnCollderEvent?.Invoke(collider.gameObject,
this);

    }

    private void OnBehaviorHandler(GameObject collider)
    {
        if(collider.tag == _selfObject.tag || collider.tag
== "Border")
        {
            return;
        }
        OnCollderEvent?.Invoke(collider.gameObject, this);
    }

    public void Dispose()
    {
        if (!_isAlive)
            return;
        if (!_isInitialize)
            return;

        _isAlive = false;

        Destroy();
    }

```

```

        //public void Disapear(Action callback)
        //{
        //    callback?.Invoke();
        //}

        protected void PlayAnimation(string name, int layer =
-1, float normalizedTime = 0)
        {
            _animator.Play(name, layer, normalizedTime);
        }

        public void Destroy()
        {
            MonoBehaviour.Destroy(_selfObject);
            _isInitialize = false;
            //DestroyEvent?.Invoke(this);
        }
    }
}

```

Повний код програми представлений за посиланням:
<https://github.com/Tapochki/GeometryAstroGame>.

ДОДАТОК В – Інструкція з використання

Geometry Astro Survival - мобільна двовимірна гра з видом зверху, де гравець з'являється на обмеженому ігровому полі та має лише одну мету – протриматися якомога більше часу та заробити максимум очків аби зайняти найвище місце у таблиці лідерів.

Після запуску гри – гравця буде вітати головне меню де є п'ять кнопок:

- Start – запускає гру
- Upgrades – меню прокачки базових характеристик персонажа
- Settings – меню налаштування
- Leaderboard – таблиця лідерів
- Credits – меню інформація про розробників

В меню Upgrades гравець може збільшити свої базові характеристики за монети назавжди. В меню Settings гравець може вимкнути музику, звуки, та, якщо потрібно вимкнути додаткові ігрові ефекти. В меню Leaderboard гравець може побачити таблицю лідерів, яка є двох типів – локальна, та глобальна. Локальна таблиця покаже усі досягнення гравця з пристрою (якщо запустити гру з іншого пристрою, то цих досягнень не буде). А глобальна таблиця лідерів покаже досягнення усіх гравців які коли небудь грали. У меню Credits можна побачити інформацію про розробників – сторінку в інстаграмі, та коротку інформацію про людину.

Сторінка з досягненнями буде відображати усі поточні досягнення гравця, ці досягнення будуть прив'язані до Google акаунту користувача. Після натискання кнопки Start гравець з'явиться на ігровому полі, аби почати двигатися потрібно використовувати лівий стік. Зверху зліва є кнопка паузи на випадок, якщо потрібно зупинити гру на необмежений час у цьому випадку з'явиться меню у якому буде лише дві кнопки – повернення у головне меню, та продовжити гру, друга у свою чергу викмне паузу та гра продовжиться.

Гра розділена на фази – тобто чим більше персонаж у грі «живе» тим складніше стає гра. У грі є вороги які будуть намагатися забрати життя у гравця аби цього не сталося гравець має змогу знищувати їх за допомогою різноманітної зброї та навичок.

У гравця є базова автоматична зброя яка стріляє у напрямку в який повернений гравець. За допомогою неї потрібно знищувати різноманітних ворогів.

Види ворогів:

- базовий(зелений квадрат)
- живучий(повільний фіолетовий шестикутник)
- швидкий(швидка жовта зірка)
- літаюча міна(коло з центральним вирізом – блимає червоним у центрі і чим ближче до гравця тим швидше блимає)
- пила(коло з зубцями пили)
- чорна діра
- троянське коло(розділяється на частинки)

Усі вороги наслідуються від базового, тому мають його параметри, але для деяких ворогів ці параметри переписані.

Параметри базового ворога:

- slow speed
- weak attack
- attack speed – one attack per second
- low health
- no armor

Унікальні навички ворогів:

- літаюча міна при зіткненні з гравцем буде вибухати та наносити гравцю колосальний урон
- пила при зіткненні з гравцем буде наносити урон увесь час доки вона зіткнена з гравцем
- чорна дірка з'являється лише в кінці ігрового поля, повільно пересуваючись вона переслідує гравця, а коли зіткнеться – поверне гравця у центр поля зі зменшеним у половину здоров'ям.

- троянське коло після смерті розділяється спочатку на дві частини, де кожна частина в свою чергу розділяється ще на дві частини після смерті. Чим менша ця частина тем менше в неї здоров'я та тим менше вона завдає пошкодження гравцю.

З кожного вбитого ворога є шанс випадання одного з наступних предметів:

- small experience crystal – надає малу кількість очків підіймання рівня
- medium experience crystal – падає рідше за small experience crystal та надає більше очків підіймання рівня
- big experience crystal – випадає переважно з босса надає багато очків підіймання рівня
- medkit – повне відновлення здоров'я гравця
- coin – дає гравцю монети одного з номіналу: одна, п'ять, десять та двадцять п'ять
- rocket – предмет для здібності Rockets
- frozen mine – міна заморозки. Випадає рідко та при підійманні заморожує ворогів у радіусі дії
- bomb – при підійманні завдає багато пошкодження ворогам у радіусі дії

Коли гравець підіймає один з кристалів він збільшує свої очки підіймання рівня і коли кількість очків досягає максимуму – рівень гравця збільшується. З'являється спливаюче вікно в якому гравцю потрібно вибрати одну з трьох унікальних здібностей які вибираються випадково з усього числа здібностей, а потім підтвердити свій вибір. Кожна здібність має короткий опис того, що вона робить. Стати безсмертним неможливо тому що кожна здібність має максимальний рівень. Здібності поділені на два типи – ПАСИВНИЙ та АКТИВНИЙ – пасивні здібності збільшують одну з характеристик гравця на певне числове значення. А активна здібність активується сама або по натисканню кнопки. Для кожної активної здібності є хоча б одна ДОДАТКОВА пасивна здібність.

Типи здібностей та методи їх активації:

- Пасивні здібності:
 - Fire rate – збільшує частоту пострілів,
 - Damage – збільшує пошкодження гравця,

- Critical hit – збільшує шанс нанесення подвійного пошкодження,
 - Critical multiplier – збільшує відсоток подвійного пошкодження базове значення 100%,
 - Bullet speed – збільшує швидкість пулі,
 - Max health – збільшує максимальну кількість здоров'я гравця,
 - Speed – збільшує швидкість пересування гравця,
 - Armor – збільшує броню гравця,
 - Luck – збільшує вдачу гравця(діє на випадіння предметів та на кількість здібностей зі скриньки яка випадає з боса гри),
 - Experience – збільшує кількість отриманих очків підіймання рівня,
 - Health regen – повільна регенерація здоров'я,
 - Double shoot – подвійний постріл,
 - Bullet count – збільшує кількість випущених гравцем пуль.
- Активні здібності:
- Camouflage – маскує гравця дозволяючи йому проходити скрізь ворогів,
 - Grenade – кидає гранату у ворогів,
 - Auto drone – автоматичний дрон який обертається навколо гравця та атакує ворогів,
 - Dash – швидке переміщення гравця у напрямку куди повернений гравець,
 - Rockets – запускає ракету у напрямку куди дивиться гравець,
 - Laser – вистрілює лазером у напрям куди дивиться гравець та знищує усіх ворогів у радіусі атаки лазера,
 - Shield – активний щит дозволяє знешкодити одну атаку ворога,
 - Auto gun – автоматична зброя яка вистрілює у будь яку сторону.
- Додаткові здібності:
- Drone gamage – збільшення пошкодження дрону,
 - Drone speed – швидкість обертання дрона,
 - Drone count – кількість дронів,
 - Grenade reload speed – швидкість створення нової гранати,

- Grenade damage – збільшення пошкодження гранати,
- Rocket damage – збільшення пошкодження ракети,
- Rocket count – кількість ракет які гравець може хранити у собі,
- Laser cooldown – швидкість перезаряджання лазера,
- Laser radius – збільшення радіуса лазера.

Завдяки цьому арсеналу здібностей гравець може себе зробити дуже сильним та заробити багато очків. Однак інколи буде з'являтися босс. Це дуже міцний та сильний ворог який має свої особливі навички. Він також буде переслідувати гравця та намагатися його знищити. Після його знищення випаде скринька зі здібностями – це додатковий спосіб отримати одну, три, або п'ять здібностей за раз не підіймаючи свого рівня.

Кількість здібностей які гравець може взяти – обмежено. П'ять пасивних та активних.

Коли гравця торкається ворог його здоров'я зменшується, якщо воно паде нижче/до нуля то гра закінчена – з'явиться відповідне меню яке покаже кількість очків та копійок які заробив гравець та кнопку повернення у головне меню звідки можна почати гру знову, але вже без навичок – з чистого листа.

Усе що було прокачане у меню UPGRADES залишається назавжди.