**ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**
**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**
**КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ**

# КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Ігрова платформа з елементами тайм-менеджменту для людей з особливими потребами»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

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

_____ _____
*(підпис)* *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувачка вищої освіти гр. ІСД-41
Анастасія СИЧ
Ім'я, ПРІЗВИЩЕ

Керівник:
*науковий ступінь, вчене звання*
Валентина ДАНИЛЬЧЕНКО, PhD
Ім'я, ПРІЗВИЩЕ

Рецензент:
*науковий ступінь, вчене звання*
Ім'я, ПРІЗВИЩЕ

**Київ 2025**

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

**Навчально-науковий інститут Інформаційних технологій**

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітньо-професійна програма Інформаційні системи та технології

**ЗАТВЕРДЖУЮ**
Завідувач кафедрою ІСТ
_____Каміла СТОРЧАК
«____» _____2025 р.

# ЗАВДАННЯ
# НА КВАЛІФІКАЦІЙНУ РОБОТУ

Сич Анастасія Олександрівна

*(прізвище, ім'я, по батькові здобувача)*

1. Тема кваліфікаційної роботи: Ігрова платформа з елементами тайм-менеджменту для людей з особливими потребами

керівник кваліфікаційної роботи Валентина ДАНИЛЬЧЕНКО, PhD,

*(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)*

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025р. № 56

2. Строк подання кваліфікаційної роботи «30» травня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, офіційна документація програмних засобів, загальні принципи дизайнк користувацького інтерфейсу, медична література.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз проблематики тайм-менеджменту людей із РАС та СДУГ
2. Проєктування архітектури та програмна розробка додатку для планування
3. Тестування розробленого рішення, збір та інтерпретація даних про ефективність роботи

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання «24» лютого 2025 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Підбір науково-технічної літератури | 01.03.2025 р. | |
| 2 | Дослідження проблематики планування часу людей із РАС та СДУГ | 15.03.2025 р. | |
| 3 | Постановка технічного завдання, функціональних та нефункціональних вимог до планувальника | 17.03.2025 р. | |
| 4 | Проєктування архітектури та вибір технологічного стеку додатку | 4.04.2025 р. | |
| 5 | Розробка додатку, тестування та виправлення помилок у роботі онлайн-магазину | 16.05.2025 р. | |
| 6 | Розробка демонстраційних матеріалів | 26.05.2025 р. | |
| 7 | Попередній захист дипломної роботи | 27.05.2025 р. | |
| 8 | Подання роботи в деканат | 30.05.2025 р. | |

Здобувач(ка) вищої освіти _____ Анастасія СИЧ
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Валентина ДАНИЛЬЧЕНКО
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

# РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 73 стор., 61 рис., 4 табл., 20 джерел.

*Мета роботи* – Розробити концепцію та прототип мобільної гри-симулятора, яка поєднує елементи догляду за віртуальним вихованцем з механіками тайм-менеджменту, для підтримки користувачів з СДУГ та РАС в організації їхнього часу та підвищенні мотивації до виконання щоденних завдань.

*Об'єкт дослідження* – Процеси управління часом та планування діяльності користувачами з СДУГ та РАС.

*Предмет дослідження* – Функціональні та інтерфейсні особливості мобільного додатку, що поєднує елементи гри-симулятора та інструменти тайм-менеджменту, адаптовані для потреб користувачів з СДУГ та РАС.

У роботі проведено аналіз психологічних та когнітивних особливостей людей з СДУГ та РАС, що впливають на їхні навички тайм-менеджменту. Досліджено ефективність існуючих програмних рішень для управління часом для цієї цільової аудиторії. Сформульовано вимоги до інклюзивного додатку для планування часу, що враховує потреби користувачів з когнітивними розладами. Описано концепцію та основні механіки розробленої гри-симулятора. Представлено результати тестування гри на вибірці користувачів з цільової аудиторії та оцінку їхнього користувацького досвіду.

# ABSTRACT

Textual part of the qualification work for obtaining a bachelor's degree: 73 pages, 61 figs., 4 tables, 20 sources.

*The purpose of the work is* to develop a concept and prototype of a mobile simulation game that combines elements of virtual pet care with time management mechanics to support users with ADHD and ASD in organizing their time and increasing motivation to complete daily tasks.

*Object of the research* – Time management processes and activity planning by users with ADHD and ASD.

*Subject of the research* – Functional and interface features of a mobile application that combines elements of a simulation game and time management tools adapted to the needs of users with ADHD and ASD.

The work analyzes the psychological and cognitive characteristics of people with ADHD and ASD that affect their time management skills. The effectiveness of existing software solutions for time management for this target audience is studied. Requirements for an inclusive time planning application that takes into account the needs of users with cognitive disorders are formulated. The concept and main mechanics of the developed simulation game are described. The results of testing the game on a sample of users from the target audience and an assessment of their user experience are presented.

**ЗМІСТ**

# ВСТУП

Актуальність теми: у сучасному світі спостерігається тенденція до збільшення кількості діагностованих випадків когнітивних розладів, зокрема синдрому дефіциту уваги з гіперактивністю (СДУГ) та розладів аутичного спектру (РАС). Це зумовлює необхідність розробки нових підходів та інструментів, які б сприяли їхній ефективній адаптації до вимог суспільства. Однією з ключових проблем, з якою стикаються люди з цими розладами, є труднощі в управлінні часом, організації та плануванні своєї діяльності.

Традиційні методи тайм-менеджменту та відповідні програмні рішення часто виявляються неефективними для цієї категорії користувачів. Це пов'язано з тим, що вони не враховують специфічні когнітивні особливості людей з СДУГ та РАС, такі як проблеми з концентрацією уваги, імпульсивність, труднощі з плануванням та схильність до сенсорного перевантаження.

У зв'язку з цим, виникає потреба у розробці інклюзивних підходів до підтримки навичок тайм-менеджменту, які б базувалися на принципах персоналізації, гейміфікації та візуальної підтримки. Застосування ігрових механік та візуальних стимулів може сприяти підвищенню мотивації та залученості користувачів з когнітивними розладами до процесу планування та виконання завдань.

Метою роботи було розробити концепцію та прототип мобільної гри-симулятора, яка поєднує елементи догляду за віртуальним вихованцем з механіками тайм-менеджменту, для підтримки користувачів з СДУГ та РАС в організації їхнього часу та підвищенні мотивації до виконання щоденних завдань.

Об’єктом дослідження було визначено процеси управління часом та планування діяльності користувачами з СДУГ та РАС.

Предметом дослідження стала можливість використання гри-симулятора догляду за віртуальним вихованцем з елементами тайм-менеджменту для підтримки користувачів з СДУГ та РАС. Розглядаються особливості розробки

інтерфейсу та ігрових механік, спрямованих на врахування когнітивних потреб цільової аудиторії та функціональні та інтерфейсні особливості мобільного додатку, що поєднує елементи гри-симулятора та інструменти тайм-менеджменту, адаптовані для потреб користувачів з СДУГ та РАС.

У даній дипломній роботі було поставлено завдання дослідити можливість використання гри-симулятора догляду за віртуальним вихованцем з елементами тайм-менеджменту для підтримки користувачів із СДУГ та РАС, а також розробити та протестувати таку гру. Для цього проводився аналіз проблематики побудови ефективного тайм-менеджменту для людей з аутизмом та синдромом дефіциту уваги і гіперактивності, включаючи їхні психологічні та когнітивні особливості, що впливають на тайм-менеджмент. Також було досліджено ефективність існуючих програмних рішень для тайм-менеджменту людей з особливими потребами та сформульовано вимоги до інтерфейсу та функціональних особливостей інклюзивного додатку для планування часу.

Методи дослідження включали розробку користувацького інтерфейсу додатку з урахуванням індивідуальних потреб основної групи користувачів, розробку ігрових механік, необхідних для правильної організації планування часу та менеджменту, а також продумку візуального стилю гри та необхідної кількості асетів. Значна частина роботи була присвячена технічній реалізації гри, що охоплювала проєктування архітектури та структури коду, реалізацію базових механік догляду за віртуальним улюбленцем, а також інтеграцію планувальника та системи мотивації. Завершальним етапом було проведення тестування гри на вибірці користувачів з цільової аудиторії для оцінки користувацького досвіду та рівня залученості, що дозволило визначити потенційні покращення та оновлення гри.

У роботі проведено аналіз психологічних та когнітивних особливостей людей з СДУГ та РАС, що впливають на їхні навички тайм-менеджменту. Досліджено ефективність існуючих програмних рішень для управління часом для цієї цільової аудиторії. Сформульовано вимоги до інклюзивного додатку для планування часу, що враховує потреби користувачів з когнітивними розладами.

Описано концепцію та основні механіки розробленої гри-симулятора. Представлено результати тестування гри на вибірці користувачів з цільової аудиторії та оцінку їхнього користувацького досвіду.

Апробація дослідження здійснювалась у формі участі в II всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу» з поданням тез на тему "Використання ігор в терапії нейровідмінних осіб: аналіз та розробка рекомендацій".

# РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ ПОБУДОВИ ЕФЕКТИВНОГО ТАЙМ-МЕНЕДЖМЕНТУ ДЛЯ ЛЮДЕЙ З АУТИЗМОМ ТА СИНДРОМОМ ДЕФІЦИТУ УВАГИ І ГІПЕРАКТИВНОСТІ

## 1.1 Психологічні та когнітивні особливості людей з аутизмом та СДУГ, що впливають на тайм-менеджмент

В останні десятиліття кількість діагностованих випадків когнітивних розладів, таких як синдром дефіциту уваги з гіперактивністю (СДУГ) та розлад аутичного спектру (РАС), зростає (рис. 1.1). Це пов'язано не тільки зі збільшенням усвідомленості суспільства та вдосконаленням методів діагностики, але й із визнанням того, що раніше такі розлади часто залишалися непоміченими або неправильно діагностованими. Тепер, коли знання про ці стани поглибились, стає зрозумілим, що ці люди потребують індивідуального підходу та підтримки для того, щоб повноцінно інтегруватися в суспільство і реалізувати свій потенціал.


Рис. 1.1 Поширеність аутизму в Каліфорнії за роком народження. [1]

Люди з когнітивними порушеннями, зокрема із розладами нейророзвитку, такими як синдром дефіциту уваги з гіперактивністю(СДУГ) і аутизм, часто стикаються з труднощами в управлінні часом, організації та плануванні. Компетентність у цих сферах є критично важливою для повсякденного життя, адже вона включає управління домашніми справами, організацію сімейного життя та підтримання стабільності у роботі.

Виконання таких завдань потребує розвинених організаційних навичок, гнучкості розпорядку та ефективного використання часу - усіх компонентів, які залежать від виконавчих функцій вищого рівня, як-от самосвідомість, розподіл часу та послідовне структурування дій. Ці вищі когнітивні функції, в свою чергу, опираються на основні когнітивні здібності, такі як зорове сприйняття, просторові відносини і пам’ять.

Завдяки сучасним дослідженням ми знаємо, що багато труднощів людей з такими розладами пов'язані із порушеннями виконавчих функцій, включаючи організацію часу, планування, та гнучкість у зміні діяльності. Традиційний підхід до підтримки цих людей полягав у спробах адаптації до суспільних «норм», що часто призводило до стресу та обмеження їхніх можливостей. Сучасний підхід, однак, передбачає розробку інструментів, які допоможуть не нав'язувати традиційні форми поведінки, а натомість підлаштовувати середовище під особливості таких людей, щоб допомогти їм ефективно структурувати свій час і діяльність.

Важливість управління часом і структурування повсякденних дій зростає в разі наявності когнітивних порушень, коли такі функції, як відчуття часу, послідовність завдань та управління розпорядком, є менш розвиненими, ніж у загальній популяції. Виконавчі функції відіграють ключову роль в управлінні часом, організації та виконанні завдань. Вони включають в себе процеси, такі як когнітивна гнучкість, здатність контролювати імпульси, а також планування і прийняття рішень. Люди із синдромом дефіциту уваги і гіперактивності (СДУГ) стикаються зі значними перешкодами в цих аспектах через порушення в роботі префронтальної кори мозку, яка відповідає за виконавчі функції. Дефіцит дофаміну

та інших нейротрансмітерів може призводити до труднощів у концентрації, регулюванні емоцій і розподілі часу. Прикладом є ситуація, коли студент із СДУГ намагається виконати домашнє завдання. Він може відволікатися на зовнішні подразники, часто переключатися між завданнями і не завершувати їх. Це призводить до хронічного відчуття невдачі та зниження самооцінки.

Згідно з Міжнародною класифікацією функціонування, інвалідності та здоров'я (ICF), управління часом є однією з цих самих виконавчих функцій, що включає впорядкування подій у хронологічному порядку та виділення часу на діяльність, необхідну для досягнення конкретної мети. У людей із СДУГ та аутизмом часто спостерігаються порушення цих функцій, що ускладнює їх здатність ефективно організовувати повсякденні дії, такі як планування завдань та вчасний вихід з дому. Крім того, різні дослідження демонструють, що ці труднощі посилюються внаслідок проблем з емоційною регуляцією, які також є типовими для осіб із СДУГ, що ускладнює контроль над власним розпорядком дня та емоційними реакціями на неочікувані зміни в ньому.

Журнал "Cogent Psychology" в 2019 році провів дослідження, щоб відслідкувати відмінності в навичках управління часом і загальної самоефективності серед осіб з когнітивними вадами та без них. Згідно з результатами, між двома групами спостерігалися значні відмінності ($p < 0,001$) за всіма трьома підшкалами самооцінки навичок управління часом і загальної самоефективності. Учасники з когнітивними порушеннями отримали нижчі оцінки за всіма показниками результатів порівняно з учасниками без когнітивних порушень (рис. 1.2). [2]

**Table 5. Differences in Time Management Skills and General Self-efficacy among participants with and without cognitive disability**

| | Participants without cognitive disability n = 147 | Participants with cognitive disability n = 86 | |
|---|---|---|---|
| **Variable** | **Mean (Sd)** | **Mean (Sd)** | **t-value** |
| ATMS-s subscales (possible range 30–120) | | | |
| Time management | 56.59 (10.89) | 40.49 (10.62) | 11.037** |
| Organization & planning | 56,32 (6.69) | 52.67 (6.19) | 4.073** |
| Regulation of emotions | 50.12 (9.64) | 43.81 (12.18) | 4.354** |
| S-GSE | | | |
| (possible range 10–40) | 30.35 (4.54) | 24.97 (5.34) | 8.311** |

**p ≤ .001

Рис. 1.2 Результати дослідження відмінності в навичках управління часом [2]

Внаслідок цього і подібних досліджень ми можемо наглядно побачити наскільки людям з когнітивними порушеннями важче контролювати всі аспекти свого життя.

Вивчення когнітивних та психологічних особливостей людей з розладами нейророзвитку дедалі більше виявляє перешкоди, які виникають у цієї групи людей в організації добового розпорядку. Основні проблеми пов’язані з дисфункціями виконавчих функцій, що визначають здатність до планування та послідовної реалізації завдань.

У людей з аутизмом особливості зорового сприйняття та порушення сприйняття часу часто призводять до труднощів у усвідомленні та плануванні діяльності. Для осіб за аутичним спектром характерною є необхідність у чіткому розпорядку та стабільності, адже зміни в повсякденному житті часто викликають негативні реакції, що можуть посилюватись на тлі труднощів з емоційною регуляцією. Вони можуть витрачати багато часу на те, щоб адаптуватися до нових умов, а будь-яке порушення звичного ритму чи неочікувані ситуації можуть призводити до сильного стресу. Це, у свою чергу, може впливати на здатність завершувати завдання або долучатися до діяльності, яка передбачає гнучкість і швидкі зміни. Наприклад, навіть невелика зміна в розкладі, як-от перенесення

зустрічі на інший час, може викликати тривалий період дезорієнтації й підвищену тривожність.

В свою чергу, люди з СДУГ також можуть стикатися з труднощами в обробці часу, що проявляється у сприйнятті часу як розмитого й не конкретного ресурсу. Часто для них існують лише два часові стани: "зараз" і "не зараз". Це ускладнює планування та пріоритетизацію завдань, адже вони можуть недооцінювати час, необхідний для виконання задачі, або ж, навпаки, відчувати, що завдання забере "вічність", і тому уникати його. Замість цього вони можуть концентруватися на діяльності, яка дає миттєву винагороду, як-от перегляд відео чи прокручування стрічки новин, відкладаючи важливі, але менш стимулюючі завдання.

Проблема ускладнюється тим, що підвищена імпульсивність часто змушує їх братися за щось нове, не завершивши попереднє. Наприклад, уявімо людину, яка почала працювати над презентацією для роботи: раптово вона згадує, що потрібно відповісти на повідомлення, потім відкриває браузер для пошуку інформації, але натомість захоплюється читанням новин або статей. У результаті основне завдання залишається незавершеним, а час спливає.

Крім того, їхня підвищена схильність до прокрастинації пов'язана з тим, що мозок людей із СДУГ має складнощі у створенні достатньої внутрішньої мотивації, особливо якщо завдання є рутинним або не викликає миттєвого інтересу. Навіть базові методи тайм-менеджменту, такі як розподіл задач за пріоритетами, використання чек-листів чи застосунків для організації часу, можуть виявитися неефективними. Це відбувається через складність слідування структурі або постійне бажання переключатися між завданнями, які видаються більш цікавими в даний момент.

Додатковою проблемою є труднощі з емоційною регуляцією та довгостроковим мисленням. Люди з СДУГ часто важко сприймають віддалені наслідки своїх дій і вважають їх менш важливими, ніж негайне задоволення. Наприклад, навіть заплановані перерви на відпочинок чи релакс можуть перетворитися на багатогодинне залипання в соціальних мережах, серіалах чи

відеоіграх. Це часто призводить до відчуття провини та стресу, адже час було втрачено, а необхідні завдання залишилися незавершеними.

Однак ще одним неприємним наслідком двох вищеперелічених станів є розвиток у таких людей і інших когнітивних та психологічних станів. Так, на управління часом значний вплив можуть мати інші фактори/стани, які ускладнюють структурування повсякденних завдань, планування та досягнення поставлених цілей. До таких станів належать тривожні розлади, депресія та обсесивно-компульсивний розлад (ОКР). Кожен із цих станів створює унікальні перешкоди для організації часу, які варто враховувати при розробці інклюзивних рішень.

Тривожні розлади часто супроводжуються хронічним страхом невдачі або помилок, що може спричинити уникнення завдань. Наприклад, людина з тривожним розладом може відкладати виконання важливих завдань через надмірний страх, що її дії будуть недостатньо правильними або оціненими. Цей страх, у свою чергу, формує порочне коло: затримки призводять до зростання обсягу роботи, що лише підсилює тривогу. Для таких користувачів особливо важливі інструменти, які допомагають зменшувати рівень тривожності, наприклад, поділ великих завдань на маленькі кроки, функція позитивних нагадувань або впровадження елементів релаксації, як-от дихальні вправи.

Депресія створює серйозні виклики через значне зниження рівня енергії та мотивації. Люди, які переживають депресивні стани, часто відчувають, що навіть базові дії, такі як встати з ліжка чи приготувати їжу, є непереборно складними. Відсутність відчуття досягнення чи прогресу підсилює апатію, і це може спричиняти повну відмову від планування чи виконання задач. У таких випадках додатки для тайм-менеджменту мають зосереджуватися на мотивації та створенні почуття досягнення. Наприклад, інтерактивні візуалізації прогресу (гейміфікація) можуть допомогти користувачам побачити результат навіть найменших кроків, що сприятиме формуванню позитивного зворотного зв’язку.

Обсесивно-компульсивний розлад (ОКР), у свою чергу, вносить специфічні труднощі у вигляді нав'язливих думок і дій. Наприклад, людина з ОКР може

витрачати надто багато часу на перевірку дрібниць (чи вимкнена плита, чи правильно виконане завдання), через що не залишається часу на інші справи. Для таких користувачів важливо, щоб додаток мав функцію нагадувань і чіткої структури, яка б дозволила мінімізувати повторювані дії. Також корисними можуть бути функції, які заохочують переключення уваги після виконання завдання, зменшуючи ризик зациклення.

У контексті розробки інклюзивного додатку для планування часу, важливо враховувати ці особливості та адаптувати функціонал під різні потреби. Наприклад, персоналізовані нагадування, елементи мотивації, інтерактивні візуалізації прогресу та функції емоційної підтримки можуть значно підвищити ефективність інструменту для користувачів із цими розладами. Завдяки індивідуалізованому підходу, люди з тривожними розладами, депресією або ОКР зможуть краще структурувати свій час, зменшити рівень стресу та відчути більше контролю над власним життям.

Усе це створює ситуацію, коли класичні методи організації часу не враховують особливості мислення таких людей. Замість того щоб допомогти, вони можуть лише посилювати стрес через постійне відчуття "неорганізованості" чи "не виконаного плану".

Тому для роботи з часом людям із СДУГ та іншими розладами нейророзвитку потрібні адаптовані стратегії, що враховують їхні особливості, зокрема використання візуальних нагадувань, коротших часових рамок для виконання завдань, а також впровадження винагород за виконані задачі для стимулювання мотивації.

Дуже багато ресурсів та медичних статей зараз освітлюють тему СДУГ та подібних розладів нейророзвитку у населення. Нижче наведено загальний графік (рис. 1.3), який було складено за допомогою близько 20 статей від різних європейських та американських лікарів. Згідно даним, найвищий рівень поширеності синдрому дефіциту уваги та гіперактивності (СДУГ) спостерігається серед дітей віком 6–12 років (9.4%) і підлітків 13–17 років (8.8%).

Ці цифри свідчать про те, що найбільш критичними є шкільні та підліткові роки, коли дитина активно взаємодіє із соціумом, здобуває освіту та формує життєво важливі навички. Неспроможність забезпечити адаптовані умови для таких дітей може призвести до тривалих труднощів у навчанні, соціальній інтеграції та емоційному благополуччі. Водночас навіть у старшому віці, коли рівень поширеності дещо знижується, відсутність підтримки та розуміння може негативно впливати на кар'єру, особисті стосунки та самореалізацію. Це доводить необхідність створення інтегрованих підходів, які враховують потреби людей із СДУГ на всіх етапах життя, оскільки своєчасна адаптація середовища та методів взаємодії здатна суттєво покращити якість життя та потенціал цих осіб.

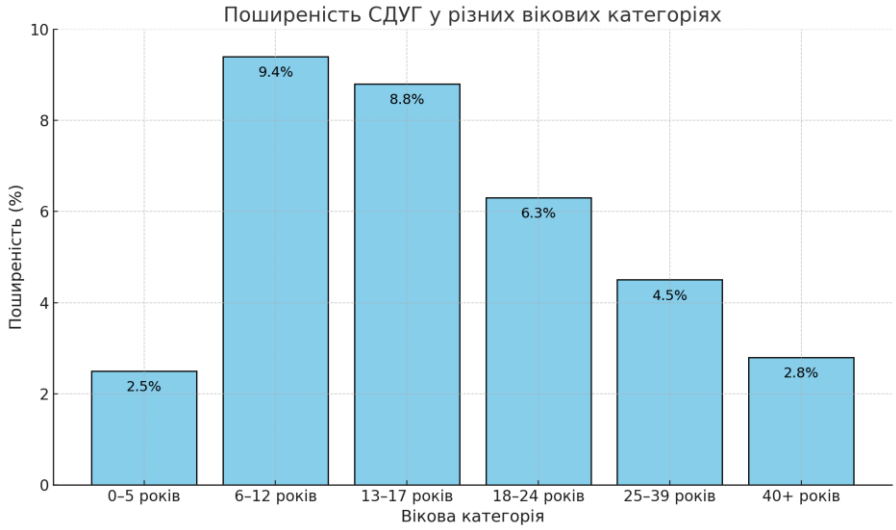



Рис. 1.3 Графік поширення СДУГ у різних вікових категоріях

В світлі вищезгаданого виникає потреба у створенні інклюзивних рішень для планування часу, які здатні підлаштуватись під індивідуальні потреби та особливості. Наприклад, це можуть бути додатки, що пропонують персоналізовані нагадування, візуальні підказки та інтерактивні модулі для планування справ. Такі підходи не лише підвищують самоефективність, але і дають людям кращу візуалізацію їх планів і зайнятості.

Виходячи з усього вищесказаного можна зібрати наступну проблематику: у світі зростає кількість діагностованих випадків когнітивних розладів, таких як СДУГ та РАС, що зумовлює потребу в індивідуальній підтримці, адже люди з цими

порушеннями часто стикаються з труднощами в управлінні часом, організації та плануванні через порушення виконавчих функцій, і традиційні методи підтримки можуть бути неефективними, не враховуючи специфіки їхнього мислення. Зокрема, люди з аутизмом потребують чіткого розпорядку, а зміни можуть викликати стрес, тоді як люди з СДУГ мають проблеми зі сприйняттям часу, імпульсивністю, прокрастинацією та мотивацією, до того ж інші когнітивні стани, такі як тривожні розлади, депресія та ОКР, також ускладнюють управління часом. Тому існує потреба в інклюзивних рішеннях для планування часу, які б адаптувалися до індивідуальних потреб, використовуючи візуальні підказки, короткі часові рамки та заохочення для підвищення ефективності, враховуючи високий рівень поширеності СДУГ серед дітей та підлітків, що підкреслює необхідність своєчасної підтримки, де персоналізовані додатки можуть допомогти людям з когнітивними розладами краще контролювати свій час та діяльність.

## 1.2 Дослідження ефективності існуючих програмних рішень для тайм-менеджменту людей з особливими потребами

У сучасному світі цифрових технологій програмні рішення для управління часом стали невід’ємною частиною повсякденного життя для багатьох людей. Вони забезпечують можливість ефективніше організовувати свої завдання, встановлювати пріоритети, нагадувати про важливі події, а також досягати поставлених цілей у визначений термін. Здавалося б, такі інструменти мали б бути універсально доступними та корисними для всіх, незалежно від їхніх індивідуальних особливостей. Проте для людей із розладами аутичного спектру (РАС) та синдромом дефіциту уваги і гіперактивності (СДУГ) використання стандартних програм стає серйозним викликом. Це пов’язано з тим, що більшість із цих інструментів були розроблені без урахування специфічних потреб цієї категорії користувачів, що призводить до численних труднощів під час їх використання.

Загалом, програми для тайм-менеджменту можна поділити на дві великі категорії. Першу категорію становлять універсальні рішення, які створені для широкої аудиторії і використовуються мільйонами людей по всьому світу. Вони пропонують багатий набір функцій, зокрема можливість створення календарів, встановлення нагадувань, організації проєктів і моніторингу виконання завдань. Відомими представниками таких рішень є Google Calendar, Microsoft To-Do, Notion, Trello та інші. Проте їхній багатофункціональний характер часто стає бар'єром для людей із РАС та СДУГ. Річ у тому, що ці програми зазвичай містять складні інтерфейси, які перевантажені текстовою та візуальною інформацією яка не є необхідною. Для людей із розладами аутичного спектру така кількість деталей може викликати сенсорне перевантаження, ускладнюючи навігацію додатком. Крім того, відсутність простих і зрозумілих візуальних підказок робить такі програми малоефективними для людей, які потребують структурованого підходу до організації часу.

До другої категорії належать спеціалізовані додатки, які створені з урахуванням потреб користувачів із когнітивними порушеннями. Прикладом таких рішень є додаток Focus@Will, який допомагає людям із СДУГ зберігати концентрацію завдяки адаптивній музиці для підвищення продуктивності. Ще одним прикладом є Habitica — гейміфікована платформа для організації завдань, яка перетворює повсякденні задачі на квести з винагородами. Також до таких рішень належить Toggl Track, що забезпечує легкий у використанні тайм-трекер із мінімалістичним інтерфейсом. Попри їхню спрямованість на специфічні потреби, ці додатки мають різний ступінь ефективності, який залежить від таких факторів, як простота використання, адаптивність і доступність персоналізації.

Для оцінки ефективності програм для тайм-менеджменту людей із СДУГ було обрано три популярні додатки: Focus@Will, Habitica і Toggl Track.

Для порівняння і аналізу вищевказаних трьох додатків були виділені наступні чотири критерії: простота використання додатку відповідала за інтуїтивність інтерфейсу та зручність навігації всередині самого додатку, адаптивність відповідала за можливість налаштування додатку під індивідуальні потреби

користувача, мотиваційний аспект - відповідав за наявність в додатку елементів гейміфікації чи стимулювання користувача по ходу виконання завдань, і доступність відповідала за цінову категорію додатку і його функцій. (таб. 1.1)

Таблиця 1.1

Порівняння трьох спеціалізованих додатків

| **Показник** | **Focus@Will** | **Habitica** | **Toggl Track** |
|---|---|---|---|
| **Простота використання** | 8/10 | 7/10 | 9/10 |
| **Адаптивність** | 6/10 | 9/10 | 7/10 |
| **Мотиваційний аспект** | 5/10 | 10/10 | 6/10 |
| **Доступність** | Платний (від $3/міс) | Безкоштовний (платні функції) | Безкоштовний (платні функції) |

За результатами проведеного дослідження можна зробити висновок, що кожний з цих додатків виконує тільки певні функції. Так додаток “Focus@Will” відмінно підходить для покращення концентрації завдяки унікальній функції адаптивної музики, але обмежена можливостями персоналізації. Додаток “Habitica” виявився найбільш ефективним для людей із СДУГ завдяки гейміфікації, що перетворює задачі на гру. Однак, на жаль, інтерфейс може бути складним для новачків. Ну і на завершення додаток “Toggl Track” здивував найпростішим і самим зрозумілим інтерфейсом, однак аспекти мотиваційного заохочення виявились дуже обмеженими. (рис. 1.4)

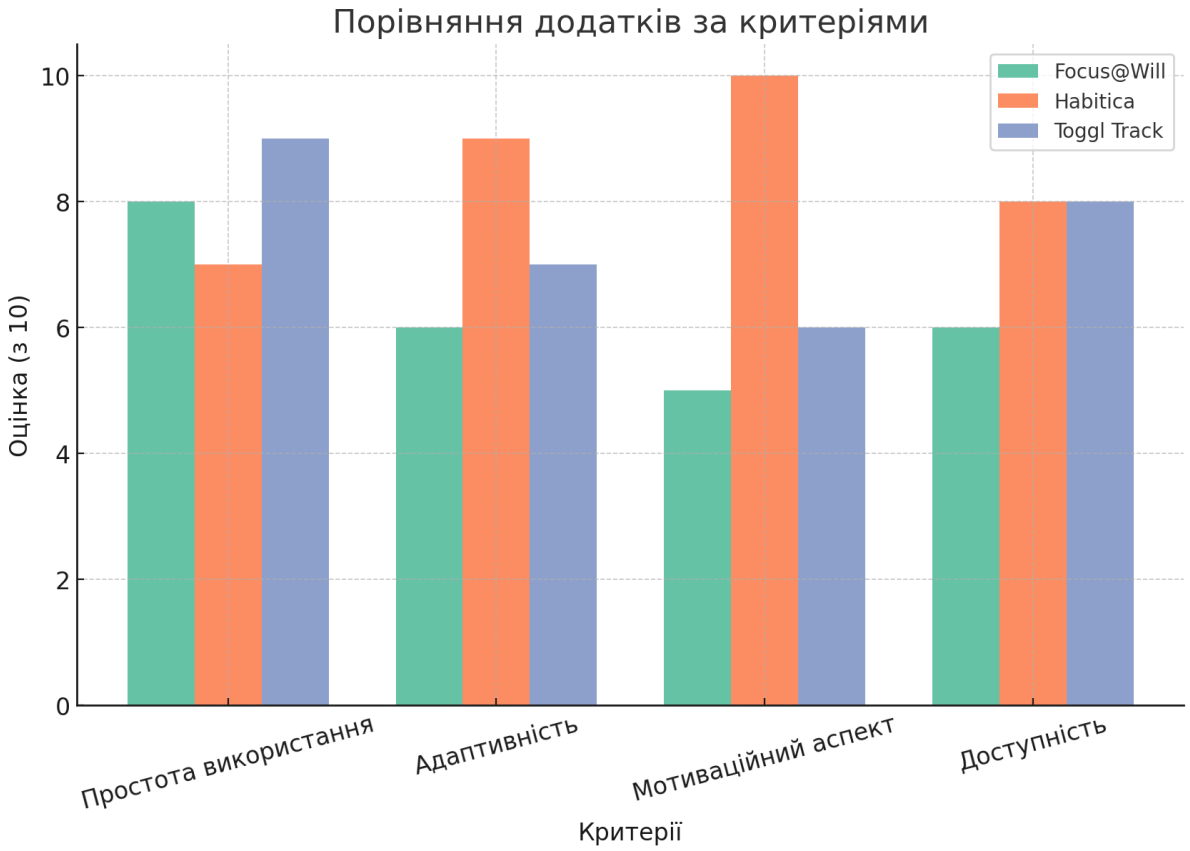



Рис. 1.4 Діаграма порівняння оцінок за всіма критеріями

Також окремим аспектом як в дослідженні, так і при виборі додатку завжди стоїть ціна. Деякі користувачі можуть і готові платити десятки доларів аби додаток повністю покривав всі їх потреби, в той час як інші користувачі шукають безкоштовні альтернативи, навіть якщо доведеться завантажувати декілька безкоштовних додатків замість одного платного. Тому для кращого аналізу справді корисного функціоналу трьох додатків було протестовано який відсоток з заявленого функціоналу є справді безкоштовним. (рис. 1.5)


Рис. 1.5 Співвідношення безкоштовних і платних функцій у кожному додатку

Результати дослідження свідчать, що існуючі програмні рішення для тайм-менеджменту можуть бути корисними для людей із СДУГ, але їх ефективність залежить від специфічних потреб користувача. Habitica вирізняється мотиваційними елементами та високою адаптивністю, тоді як Toggl Track забезпечує простоту використання. У майбутньому слід орієнтуватися на створення додатків із поєднанням цих сильних сторін: простоти, гейміфікації та персоналізації.

Дослідження ефективності трьох популярних додатків для людей із СДУГ виявило, що Focus@Will добре підходить для концентрації, але має обмежену персоналізацію, Habitica ефективна завдяки гейміфікації, але її інтерфейс може бути складним, а Toggl Track має простий інтерфейс, але обмежені мотиваційні аспекти, при цьому вартість також є важливим фактором при виборі додатку. Результати свідчать, що існуючі рішення можуть бути корисними, але їхня ефективність залежить від індивідуальних потреб, і в майбутньому слід орієнтуватися на створення додатків із поєднанням простоти, гейміфікації та персоналізації. Основними недоліками існуючих програм є складні інтерфейси, сенсорне перевантаження, відсутність візуальної структури, обмежена персоналізація, нерівномірність функціоналу та ціновий бар'єр.

Функціональність нового програмного забезпечення для управління часом має бути розроблена з урахуванням когнітивних особливостей користувачів із розладами аутичного спектру та синдромом дефіциту уваги і гіперактивності, забезпечуючи персоналізований досвід взаємодії. Ключовими елементами функціоналу повинні стати візуально орієнтований інтерфейс із можливістю індивідуального налаштування відображення елементів, шрифтів та кольорової гами, а також система гнучких нагадувань, що підтримують різні формати сповіщень, включаючи візуальні, звукові та вібраційні сигнали з регульованою інтенсивністю. Інтегрована система управління завданнями повинна передбачати можливість їх категоризації та пріоритезації з використанням візуальних маркерів та піктограм, а вбудовані інструменти для розбиття складних завдань на менші, керовані етапи сприятимуть зниженню когнітивного навантаження. Окрім того,

наявність опціональних елементів гейміфікації, таких як система заохочень за виконання завдань та візуалізація прогресу у формі інтерактивних діаграм або шкал, покликана підтримувати мотивацію користувачів. (рис. 1.6)

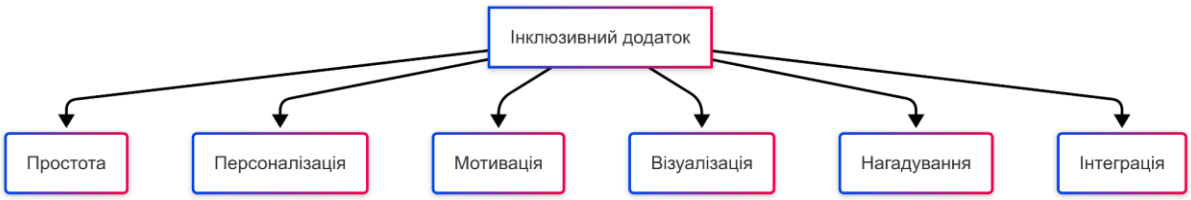



Рис. 1.6 Загальний список вимог до нового додатку

Новий інклюзивний додаток для тайм-менеджменту повинен мати простий та інтуїтивно зрозумілий інтерфейс з мінімумом тексту та чіткими візуальними елементами, високу ступінь персоналізації з налаштуванням відображення, звуків та складності функцій, елементи гейміфікації та мотивації з системою винагород та візуалізацією прогресу, чітку візуальну структуру з кольоровим кодуванням та іконками, гнучкі нагадування з можливістю налаштування та інтеграцію з іншими інструментами, такими як календар та списки завдань, що сприятиме більшій самостійності та продуктивності користувачів з РАС та СДУГ.

## 1.3 Вимоги до інтерфейсу та функціональних особливостей інклюзивного додатку для планування часу

На основі проведеного аналізу психологічних та когнітивних особливостей людей із синдромом дефіциту уваги і гіперактивності (СДУГ) та розладами аутичного спектру (РАС), а також результатів дослідження ефективності існуючих програмних рішень для тайм-менеджменту, можна визначити основні вимоги до інтерфейсу та функціональних особливостей інклюзивного додатку для планування часу. Важливо враховувати специфіку сприйняття часу, організації діяльності та мотиваційні особливості цільової аудиторії, що дозволить створити ефективний ігровий симулятор з елементами планування.

Однією з ключових особливостей такого додатку є гейміфікація процесу планування. Дослідження показують, що люди з когнітивними розладами часто мають труднощі з довгостроковою мотивацією, оскільки їхній мозок схильний надавати перевагу негайному задоволенню замість поступового досягнення цілей. Саме тому ігровий підхід дозволяє зробити виконання завдань більш привабливим (таб. 1.2). У грі-тамагочі варто впровадити систему внутрішніх нагород, яка мотивуватиме користувачів виконувати заплановані завдання. Наприклад, за своєчасне виконання завдань користувач отримуватиме віртуальну валюту або бонуси, які можна використовувати для догляду за цифровим вихованцем. Це сприятиме формуванню позитивного підкріплення та підвищенню самомотивації користувача. Другою важливою вимогою є простота та адаптивність інтерфейсу. Люди з СДУГ та аутизмом можуть мати труднощі з навігацією у складних меню, тому інтерфейс гри має бути мінімалістичним та інтуїтивно зрозумілим. Візуальна складова повинна включати зрозумілі графічні підказки, які допомагатимуть користувачам швидко орієнтуватися в грі. Крім того, необхідно передбачити можливість налаштування кольорової гами, розміру шрифтів та інших візуальних параметрів, що дозволить підлаштовувати гру під індивідуальні потреби користувачів. Використання кольорового кодування для позначення різних типів завдань допоможе швидко сприймати інформацію і зменшить когнітивне навантаження.

Таблиця 1.2

Вплив гейміфікації на мотивацію користувачів

| Елемент гри | Рівень впливу на мотивацію (за шкалою 1-10) |
|---|---|
| Віртуальна валюта | 9 |
| Кастомізація персонажа | 8 |
| Нагороди за виконані завдання | 10 |
| Анімаційні нагадування | 7 |

Ще одним важливим аспектом є впровадження механізмів підтримки концентрації уваги. Одним із рішень може стати режим "Якісний час із улюбленцем", в якому користувач виконує заплановане завдання, а його віртуальний вихованець взаємодіє з ним у цей період. Якщо користувач залишає режим до завершення таймера, його вихованець отримує штрафи у вигляді зниження настрою або втрати ресурсів. Така механіка сприятиме формуванню звички концентруватися на завданні без відволікань. Додатково варто реалізувати інтегровані нагадування, які допомагатимуть користувачам стежити за своїм розкладом без надмірного тиску. Наприклад, замість звичайних текстових нагадувань можна використати анімаційні підказки від вихованця, які будуть привертати увагу м'яко, без надмірного стресу. Для аналізу впливу виконаних завдань на рівень задоволеності користувачів було побудовано графік (рис. 1.6), що показує, як часте виконання завдань позитивно впливає на загальну залученість у гру. Як видно з даних, користувачі, які виконують більше завдань, демонструють вищий рівень задоволеності та зацікавленості у використанні додатку.


Рис 1.6 Залежність рівня задоволеності від кількості виконаних завдань

Крім того, гра повинна мати високий рівень персоналізації. Кожен користувач має свої унікальні особливості сприйняття інформації, тому варто передбачити можливість вибору рівня складності планування. Наприклад, користувач може вибрати короткі або довгі проміжки між завданнями, а також змінювати частоту нагадувань. Також корисною буде функція кастомізації персонажа та вихованця, що підвищить залученість користувача (рис. 1.7). Інтерактивні звіти та прогрес-бари допоможуть візуалізувати досягнення та мотивуватимуть користувача дотримуватися розкладу.

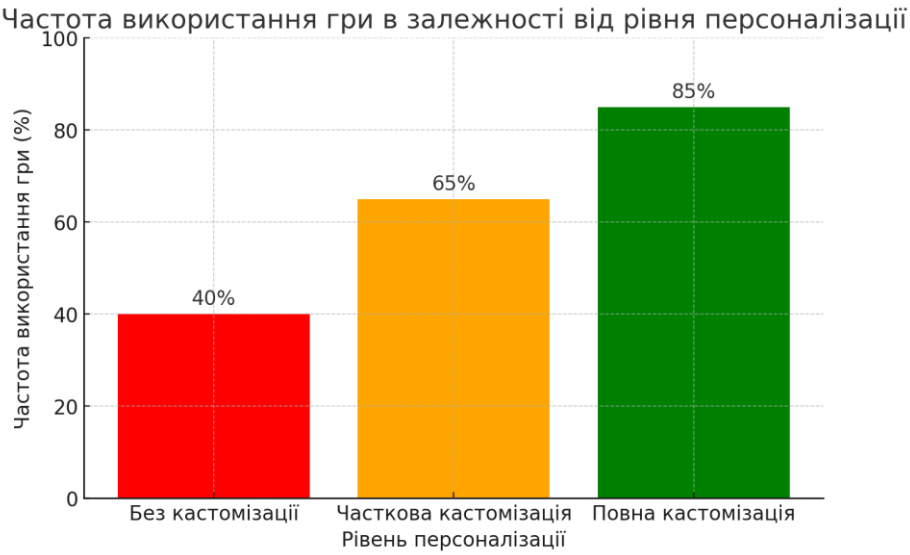



Рис1.7 Частота використання гри в залежності від рівня персоналізації

Інтеграція гри з реальним життям є ще одним важливим фактором. Для цього варто передбачити можливість синхронізації з існуючими календарями та планувальниками, щоб користувачі могли зручно переносити свої справи з інших додатків. Додатково варто впровадити елементи психологічної підтримки, такі як вправи для зниження стресу або техніки релаксації. Це допоможе користувачам краще адаптуватися до змін у планах та зменшить тривожність, пов’язану з управлінням часом.

Таким чином, розробка інклюзивної гри-тамагочі для планування часу має враховувати особливості цільової аудиторії, зокрема складнощі з управлінням часом, необхідність візуальної підтримки та потребу в мотивації. Використання

гейміфікації, адаптивного інтерфейсу, механізмів концентрації уваги, персоналізації та інтеграції з реальним життям дозволить створити ефективний інструмент для покращення тайм-менеджменту людей із СДУГ та аутизмом. Поєднання цих елементів у грі створить не лише корисний, а й захопливий досвід, який сприятиме формуванню звичок, необхідних для ефективного керування своїм часом.

Також не можна забути про доступність і адаптація додатку для різних груп користувачів, що також є важливим аспектом. Люди з когнітивними порушеннями часто мають різні потреби у взаємодії з інтерфейсом, тому гра має містити налаштування сенсорного зменшення, можливість змінювати рівень візуального навантаження та використовувати голосові підказки. Це дозволить зробити гру зручною для широкого кола користувачів.

Система підтримки звичок та довгострокового планування може допомогти користувачам у формуванні корисних звичок. Запровадження щоденних, щотижневих та довгострокових цілей допоможе створити відчуття поступового прогресу. Крім того, наявність шаблонів завдань, адаптованих під різні стилі життя, та автоматичне відстеження прогресу сприятиме кращому управлінню часом.

Соціальна взаємодія у грі може відігравати мотиваційну роль. Додавання функцій обміну бонусами, дружнього рейтингу та групових викликів допоможе користувачам підтримувати один одного та підвищить рівень залученості у процес планування часу. Це добре показано на графіку, що наведено нижче (рис. 1.8). Він показує, як соціальні функції (обмін бонусами, рейтинги, групові виклики) впливають на залученість користувачів. Лінія синього кольору демонструє стрімке зростання активності після впровадження соціальних механік, тоді як червона лінія (без них) показує більш повільний ріст. Це наочно підтверджує мотиваційний ефект соціальної взаємодії в грі. [3]

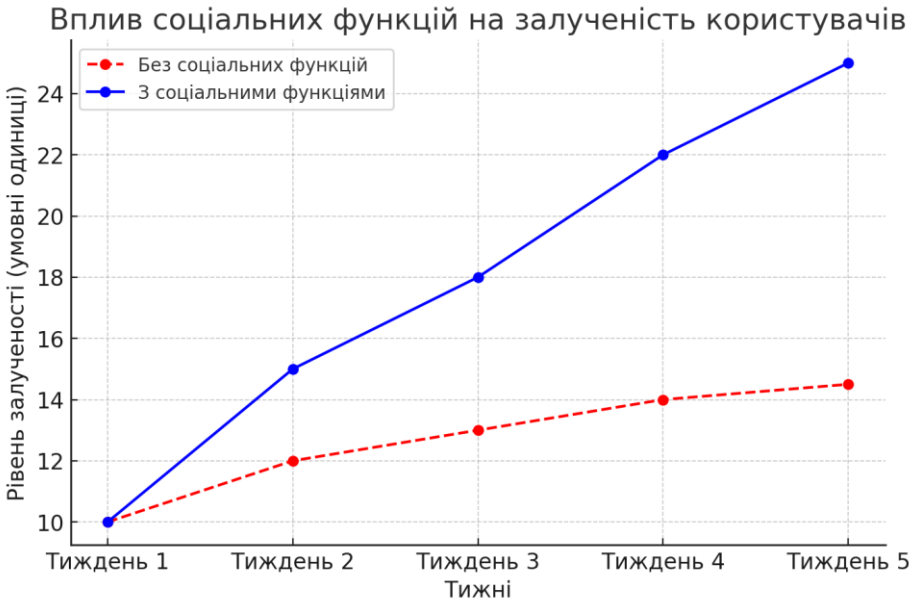



Рис 1.8 Графік впливу соціальних функцій на залученість користувачів

Інтеграція додатку з іншими сервісами є важливою для зручності користувачів. Вбудована синхронізація з календарями, мобільними нагадуваннями та іншими планувальниками дозволить користувачам легко поєднувати ігровий процес з реальним життям. Це сприятиме створенню цілісної системи управління часом, яка буде не тільки ефективною, а й приємною у використанні.

При створенні інтерфейсу інклюзивного додатку важливо враховувати особливості сприйняття інформації користувачами із синдромом дефіциту уваги з гіперактивністю (СДУГ) та розладами аутичного спектру (РАС). Люди з такими особливостями часто стикаються з труднощами у навігації через перевантажені інтерфейси, складні меню та велику кількість текстової інформації.

Одним із ключових аспектів є використання кольорів та контрастності. Дослідження показують, що яскраві кольори можуть відволікати увагу, тоді як недостатній контраст між елементами ускладнює їхнє розпізнавання. Оптимальним рішенням є використання м’яких пастельних тонів із добре вираженим контрастом для кнопок і важливих елементів. Наприклад, важливі нагадування можна підсвічувати одним кольором, а виконані завдання – іншим, що дозволить користувачам легше орієнтуватися у вмісті.

Ще одним важливим фактором є шрифти та розмір тексту. Дрібний текст або надто декоративні шрифти можуть ускладнювати читання та викликати втому очей. Рекомендується використовувати шрифти без засічок, наприклад, Open Sans, Roboto або Arial, з мінімальним розміром 14-16 pt для основного тексту.

Також слід враховувати мінімізацію зайвих візуальних елементів. Додаток має містити лише необхідну інформацію, розташовану у логічному порядку. Великі кнопки, прості форми та інтуїтивно зрозумілі піктограми допоможуть користувачам швидко орієнтуватися у програмі. Крім того, важливо використовувати єдину стилістику для всіх елементів: однаковий дизайн кнопок, стандартне розташування меню та передбачувану логіку переходів між екранами.

Ще одним рішенням для покращення доступності є адаптація під сенсорну чутливість. Деякі люди з аутизмом можуть негативно реагувати на різкі візуальні ефекти або гучні звукові сигнали. Тому в додатку має бути можливість налаштовувати інтенсивність анімацій, гучність звуків і навіть колірну гамму відповідно до індивідуальних потреб користувача.

# РОЗДІЛ 2 РОЗРОБКА КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ ДОДАТКУ З УРАХУВАННЯМ ІНДИВІДУАЛЬНИХ ПОТРЕБ ОСНОВНОЇ ГРУПИ КОРИСТУВАЧІВ

## 2.1. Розробка концепції ігрової платформи

Розроблюваний додаток являє собою мобільну гру-симулятор догляду за віртуальним вихованцем, яка поєднує елементи тайм-менеджменту та гейміфікації. Основна ідея полягає у тому, щоб допомогти користувачам з захворюваннями аутистичного спектру (ЗАС) та синдромом дефіциту уваги з гіперактивністю (СДУГ) структурувати свій час та підвищити мотивацію до виконання завдань.

Ключовий принцип гри – "дбай про свого вихованця, виконуючи реальні справи". Користувач створює власного віртуального персонажа (пет-симулятор), який потребує уваги та догляду. Щоб вихованець розвивався, гравець має додавати та виконувати заплановані завдання: навчальні, робочі чи побутові. Цей формат гри поєднує дві важливі складові - емоційний зв’язок із вихованцем та гейміфікована система тайм-менеджменту. Перша складова створює додаткову мотивацію, в той час як друга - допомагає користувачам запам’ятовувати, структурувати та виконувати свої завдання у доступній та цікавій формі.

Цільова аудиторія (ЦА) цього додатку - це люди з когнітивними особливостями, для яких традиційні системи планування є неефективними. В першу чергу це люди з СДУГ та ЗАС, а також будь-хто, хто має проблеми з організацією часу. Гра також може бути корисна тим, хто просто шукає додаткове джерело мотивації для виконання щоденної рутини.

Люди з СДУГ та ЗАС часто уникають складних додатків, тому гра має бути інтуїтивно зрозумілою, простою у використанні та адаптивною. Простий інтерфейс з мінімумом зайвої інформації, зрозумілі іконки, адаптивний розмір тексту, гейміфіковане планування, де замість списку справ користувач отримує віртуального вихованця, візуальні підказки та анімації замість текстових

нагадувань, гнучкість у налаштуваннях - все це сприяє досягненню поставленої мети.

Вибір піксельної графіки (рис. 2.1) як основи дизайну для додатку зумовлений кількома факторами. По-перше, цей стиль дозволяє досягти візуальної простоти та зрозумілості, що особливо важливо для користувачів з когнітивними особливостями, такими як СДУГ та ЗАС. Піксельна графіка мінімізує візуальне перевантаження, дозволяючи зосередитись на ключових елементах інтерфейсу та ігровому процесі.

По-друге, піксельна графіка має свою естетичну привабливість та асоціюється з ретро-іграми, що може викликати позитивні емоції та ностальгію у користувачів. Цей стиль не втрачає своєї актуальності та може бути використаний для створення сучасного та стильного дизайну.

По-третє, піксельна графіка полегшує процес створення ассетів для гри. Створення спрайтів та фонів у цьому стилі не вимагає високого рівня художньої майстерності та може бути виконано навіть за допомогою простих графічних редакторів. Це дозволяє зосередитись на геймдизайні та механіках гри, а не на складних технічних аспектах створення графіки.

Таким чином, піксельна графіка є оптимальним вибором для даного проекту, оскільки дозволяє створити простий, привабливий та доступний інтерфейс, який буде зручним для користувачів з когнітивними особливостями.

Рис 2.1 Піксельна графіка на прикладі гри “Stardew Valley” [4]

Гра буде пропонувати різноманітні завдання, які користувач може виконувати для догляду за своїм віртуальним вихованцем. Наприклад, побутові (помити посуд, прибрати в кімнаті, винести сміття, полити квіти, приготувати їжу), навчальні (прочитати заданий текст, виконати домашнє завдання з математики, підготуватися до контрольної роботи, написати реферат), робочі (відповісти на електронні листи, підготувати презентацію, завершити звіт, провести зустріч), особисті (зайнятися спортом, погуляти на свіжому повітрі, почитати книгу, помедитувати). Крім цього у користувачів буде можливість додати свої, кастомні, завдання і свій список справ на день що оновлюється.

Гра буде адаптуватися до потреб користувачів з різними когнітивними особливостями за допомогою рівнів складності (легкий, середній, складний) та налаштувань доступності (сенсорне зменшення, голосові підказки, зміна кольорової гами, адаптивний розмір тексту). Додатково буде розглянуто можливість додавання функції, яка дозволить користувачам самостійно налаштовувати параметри гри відповідно до своїх потреб.

На зображенні (рис. 2.2) представлений інтерфейс мобільної гри Tuggle [5], яка є симулятором віртуального вихованця. Ліва частина екрану демонструє

головного персонажа - милого пухнастого створіння з великими очима, що стоїть на фоні пейзажу з деревами та пагорбами. Над персонажем розташовані іконки, що відображають його ім'я (Petto), рівень, кількість зібраних зірок, стан здоров'я та енергії, а також іконку магазину. Під персонажем знаходиться шкала прогресу, позначена "3s". Внизу екрану розташовані іконки різних активностей, які можна виконати з персонажем: фізичні вправи, стрільба з лука, бій на мечах. Права частина екрану демонструє магазин ігрових предметів (Store), де на полицях розташовані різні зілля, магічні палички, кільця, броня та одяг. В горі екрану відображається кількість ігрової валюти (9999).

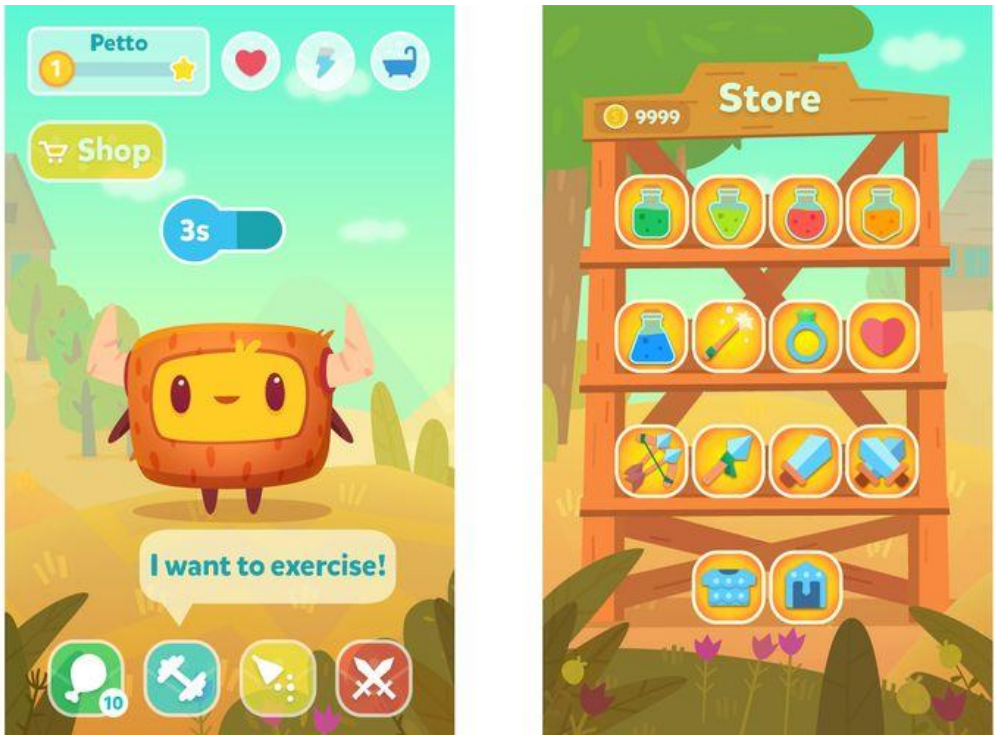



Рис 2.2 Приклад інтерфейсу мобільної гри Tuggle [5]

Інтерфейс Tuggle виконаний в яскравій кольоровій гамі з використанням великих кнопок та чітких іконок, що робить його інтуїтивно зрозумілим та легким для сприйняття. Персонаж виглядає привабливо та дружелюбно, що сприяє створенню позитивної емоційної атмосфери. Візуальні елементи чітко відокремлені один від одного, що допомагає уникнути сенсорного перевантаження та полегшує навігацію для користувачів з когнітивними особливостями. Яскраві кольори та великі елементи привертають увагу та допомагають користувачам з

СДУГ краще концентруватися на завданнях. Чіткі іконки та простий інтерфейс полегшують розуміння та взаємодію з грою для користувачів з ЗАС. Дружелюбний персонаж створює позитивну атмосферу та може знижувати рівень тривожності у користувачів з когнітивними особливостями. Важливо враховувати можливість адаптації інтерфейсу під індивідуальні потреби користувачів. Наприклад, можна передбачити можливість змінювати розмір тексту, колірну гаму, а також відключати анімації або звукові ефекти для тих, хто чутливий до сенсорного перевантаження. Інтерфейс Tuggle є вдалим прикладом дизайну, який враховує потреби користувачів з когнітивними особливостями.

## 2.2. Розробка ігрових механік, необхідних для правильної організації планування часу та менеджменту

Аналіз популярних ігор з віртуальними вихованцями, таких як Pou та My Talking Tom, виявляє ключові механіки залучення користувачів. У Pou успіх забезпечується різноманіттям активностей: годування з вибором страв, догляд за чистотою, інтерактивні ігри, персоналізація зовнішнього вигляду та кімнати, а також широкий спектр міні-ігор для заробітку внутрішньоігрової валюти. Соціальні функції, що дозволяють відвідувати друзів, дарувати подарунки та змагатися, також значно підвищують залученість. My Talking Tom робить акцент на розвитку персонажа, який росте та змінюється, відкриваючи нові можливості. Інтерактивність через повторення слів та реакції на дотики створює відчуття "живого" вихованця. Симуляція базових потреб, таких як годування, гігієна та сон, а також зміна часу доби, сприяє розвитку відповідальності гравця. Соціальна взаємодія з елементами змагання та співпраці також є важливою складовою гри. Обидві гри демонструють, що різноманітні активності, персоналізація, розвиток персонажа, симуляція потреб та соціальна взаємодія є потужними інструментами для утримання інтересу користувачів у віртуальних вихованцях.

Майбутній додаток для управління часом поєднуватиме елементи гри та реального життя, створюючи унікальний досвід для користувачів, особливо для

тих, хто має розлади аутичного спектру та синдром дефіциту уваги і гіперактивності (рис. 2.3). Основна ідея полягає в тому, щоб зробити процес організації часу захопливим та мотивуючим, використовуючи ігрові механіки, адаптовані до когнітивних особливостей цільової аудиторії (рис. 2.4).

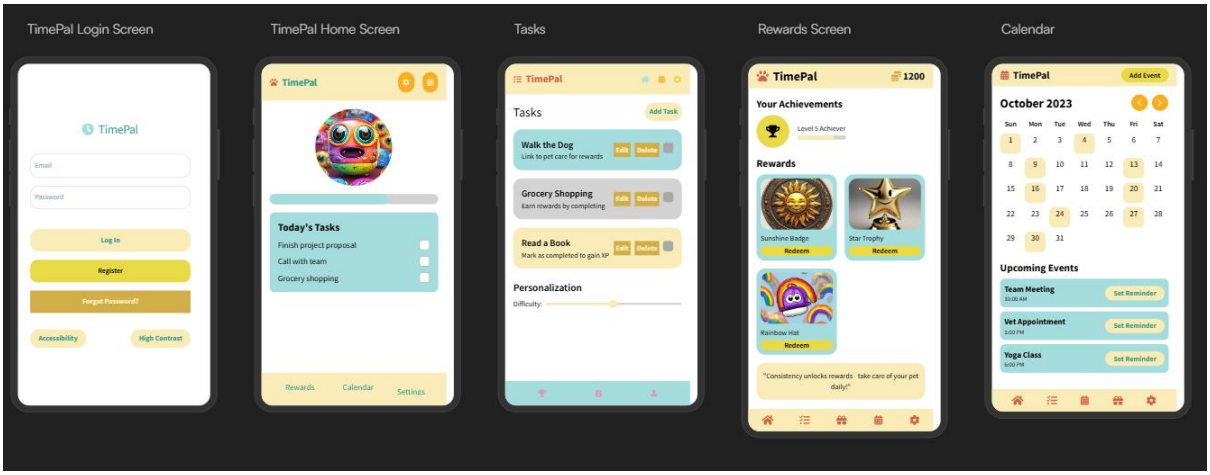



Рис. 2.3 Приклад дизайну додатку, розроблений у Figma

Центральною функцією стане взаємодія з віртуальним вихованцем. Користувач візьме на себе відповідальність за догляд за ним, де кожен аспект піклування буде тісно пов'язаний з виконанням реальних завдань. Наприклад, щоб "нагодувати" віртуальну тварину, потрібно буде виконати заплановане приготування їжі, а для підтримки її "здоров'я" - зайнятися фізичною активністю. Такий підхід має на меті зробити рутинні справи більш відчутними та значущими.

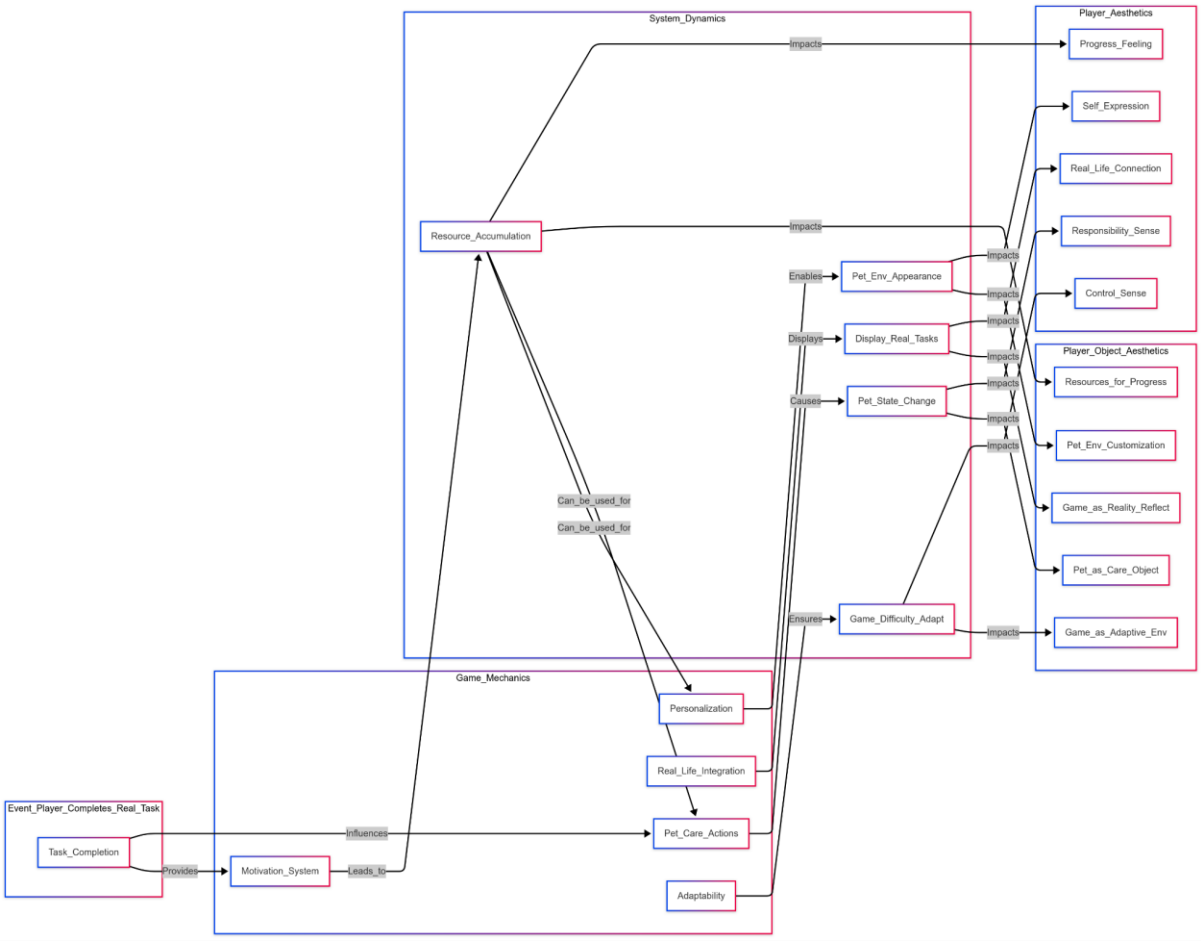



Рис. 2.4 Діаграма взаємозв’язку компонентів між собою

Система мотивації відіграватиме ключову роль у підтримці зацікавленості користувачів. За кожне виконане завдання передбачаються віртуальні винагороди, такі як внутрішньоігрова валюта, очки досвіду та різноманітні бонуси. Зароблені ресурси можна буде використовувати для покращення віртуального життя вихованця, придбання для нього нових предметів та аксесуарів. Накопичення досвіду призведе до "зростання" та розвитку віртуальної істоти, відкриваючи нові ігрові можливості. Додатково будуть передбачені досягнення за певні успіхи, щоденні завдання з додатковими заохоченнями та сезонні події зі спеціальними місіями та унікальними призами.

Інтеграція з повсякденним життям буде забезпечена через синхронізацію з особистим календарем та списком справ користувача. Завдання, додані в реальному світі, автоматично відображатимуться в ігровому середовищі, створюючи цілісну систему планування. Також передбачена можливість встановлення ігрових

нагадувань, які надходитимуть на мобільний пристрій. Користувачі зможуть ділитися своїми ігровими успіхами з друзями через соціальні мережі, що може додати елемент соціальної взаємодії та підтримки.

Персоналізація стане важливою складовою додатку, дозволяючи кожному користувачеві створити унікального віртуального компаньйона. Можна буде обирати різні види тварин, кожна з яких матиме свої особливості та потреби. Зовнішній вигляд вихованця можна буде налаштовувати за допомогою різноманітних опцій. Крім того, користувачі зможуть змінювати візуальне оформлення ігрового простору та обирати музичний супровід.

Адаптивність забезпечить комфортне використання додатку для широкого кола користувачів. Різні рівні складності дозволять кожному обрати оптимальний режим гри відповідно до своїх можливостей. Налаштування доступності враховуватимуть потреби людей з різними когнітивними особливостями, роблячи додаток максимально інклюзивним.

### 2.3. Візуальний стиль гри та продумка необхідної кількості асетів

Візуальний стиль гри буде виконаний в піксельному стилі, що забезпечить простоту та зрозумілість інтерфейсу, а також полегшить процес створення асетів. Для створення позитивної та спокійної атмосфери буде використана яскрава, але не надто насичена кольорова палітра, з переважанням пастельних відтінків. Основними кольорами будуть блакитний та зелений, що асоціюються з природою та спокоєм, а також рожевий та жовтий, що додадуть яскравості та позитиву. Кольори будуть ретельно підібрані таким чином, щоб бути приємними для ока та не викликати перевантаження у користувачів з когнітивними особливостями (рис. 2.5).


Рис 2.5 Приклад палітри кольорів [6]

Віртуальний вихованець буде милим та привабливим (рис. 2.6), з простими формами та великими виразними очима, щоб викликати у гравця емоційний зв'язок та бажання піклуватися про нього . Дизайн вихованця буде простим та зрозумілим, без зайвих деталей, щоб уникнути візуального перевантаження. При цьому, вихованець буде мати достатньо індивідуальності, щоб гравець міг відчути до нього прив'язаність.


Рис 2.6 Приклад спрайт-листа кролика

Фони гри будуть деталізованими, але не перевантаженими, зображаючи різні локації, такі як кімната вихованця, двір або парк, ігрові майданчики, лікарня, магазин тощо. Фони будуть створювати атмосферу затишку та комфорту, але при цьому не будуть відволікати гравця від основного ігрового процесу. Деталі фонів будуть ретельно продумані, щоб створити відчуття реалістичності та занурення у гру.

Прості анімації, такі як рух хвоста, вух, або моргання, додадуть динаміки та життя у гру. Анімації будуть плавними та природними, щоб не викликати дискомфорту у гравців. Для підкреслення певних дій або подій будуть використані ненав'язливі візуальні ефекти, такі як блиск, сяйво, або частинки. Ефекти будуть доповнювати візуальний стиль гри та робити його більш виразним.

При продумуванні кількості асетів важливо знайти баланс між різноманітністю та кількістю, щоб уникнути візуального перевантаження та ускладнення процесу розробки, але при цьому зробити гру цікавою та різноманітною. Для початку буде створено базовий набір асетів, достатній для реалізації основних механік гри, а пізніше, в процесі розробки, будуть додані нові асети, якщо в цьому буде потреба. Додатково можна буде створити тематичні набори асетів, наприклад, для Хелловіну, Різдва, або дня народження. Також можна буде дозволити гравцям персоналізувати асети, наприклад, змінювати колір шерсті вихованця, або вибирати різні меблі для кімнати. Використання анімованих асетів допоможе створити більш динамічне та живе оточення.

Базовий набір асетів для тамагочі включатиме все необхідне для повноцінного запуску гри та демонстрації її основних механік. Спочатку буде створено один вид вихованця (наприклад, кішка) з усыма анімаціями. Для кожної варіації тваринки буде доступно три базових костюми. Фони гри будуть включати кімнату вихованця з базовим набором меблів (ліжко, миска, іграшка) та один варіант фону для двору або парку.

Щодо предметів, гра міститиме три види їжі, три види іграшок та базовий набір ліків для лікування хвороб вихованця. Інтерфейс гри буде складатися з кнопок для основних дій (годування, гра, лікування тощо), іконок для відображення

стану вихованця (голод, настрій, здоров'я), шрифту для відображення тексту та простого прогрес-бару для відстеження розвитку та стану вихованця (рис. 2.7).

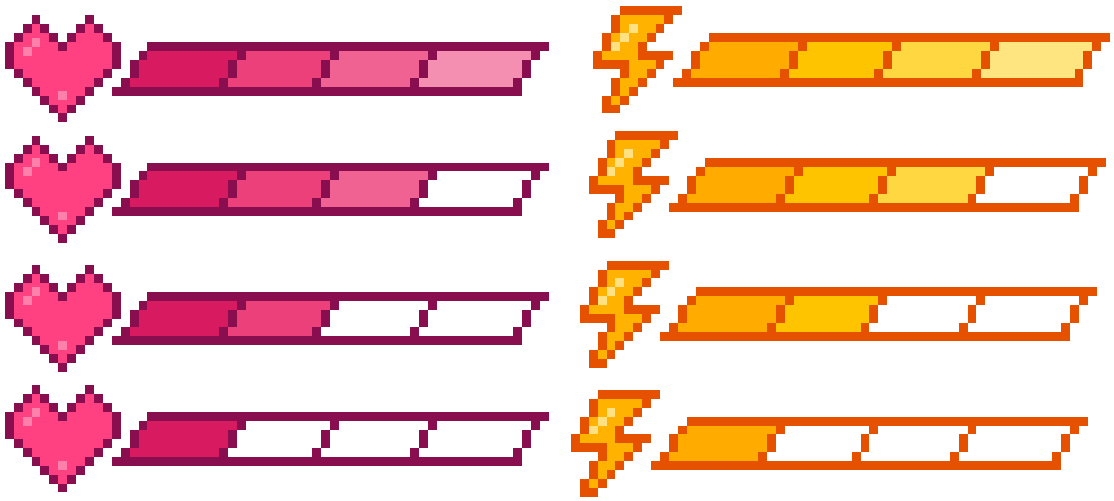


Рис 2.7 Приклади статус-бару з асетів проекту

Цей базовий набір дозволить гравцям доглядати за вихованцем, годувати його, грати з ним, лікувати та спостерігати за його розвитком. Він також дасть можливість протестувати основні механіки гри та зібрати відгуки від користувачів для подальшого покращення і тестування. В подальшому, цей набір може бути розширений за рахунок додавання нових видів вихованців, емоцій, костюмів, фонів, предметів та інтерактивних елементів.

В цілому, візуальний стиль гри має бути привабливим, зрозумілим та адаптованим до потреб цільової аудиторії. Продуманий підбір та створення асетів допоможе створити гру, яка буде не тільки корисною, але й естетично приємною.

# РОЗДІЛ 3 ТЕХНІЧНА РЕАЛІЗАЦІЯ ГРИ

## 3.1. Проєктування архітектури та огляд структури програмного забезпечення

Ефективна архітектура програмного забезпечення є фундаментом для успішного розвитку та життєвого циклу будь-якого складного проекту, і гра "Тамагочі-Планер" не є винятком. Її важливість полягає у забезпеченні трьох ключових характеристик: масштабованості, підтримуваності та зрозумілості кодової бази. Масштабованість дозволяє розширювати функціональність гри в майбутньому, додавати нові можливості та контент без необхідності значних переробок існуючого коду. Підтримуваність робить процес виявлення та виправлення помилок, а також внесення невеликих змін, більш простим та ефективним, зменшуючи ризик виникнення нових проблем при модифікації коду. Зрозумілість архітектури та коду, в свою чергу, полегшує загальне розуміння принципів роботи програми.

У даному проекті для досягнення цих цілей було обрано модульний підхід до архітектури (рис. 3.1). Суть цього підходу полягає у розбитті всієї функціональності гри на відносно незалежні модулі або системи, кожна з яких відповідає за певну чітко визначену область відповідальності. Такий поділ дозволяє уникнути створення монолітної структури, де всі частини коду тісно переплетені між собою, що ускладнює будь-які зміни або оновлення. Завдяки модульній архітектурі, кожна система може розроблятися, тестуватися та модифікуватися відносно незалежно від інших, що значно спрощує процес розробки та зменшує ймовірність виникнення побічних ефектів при внесенні змін.

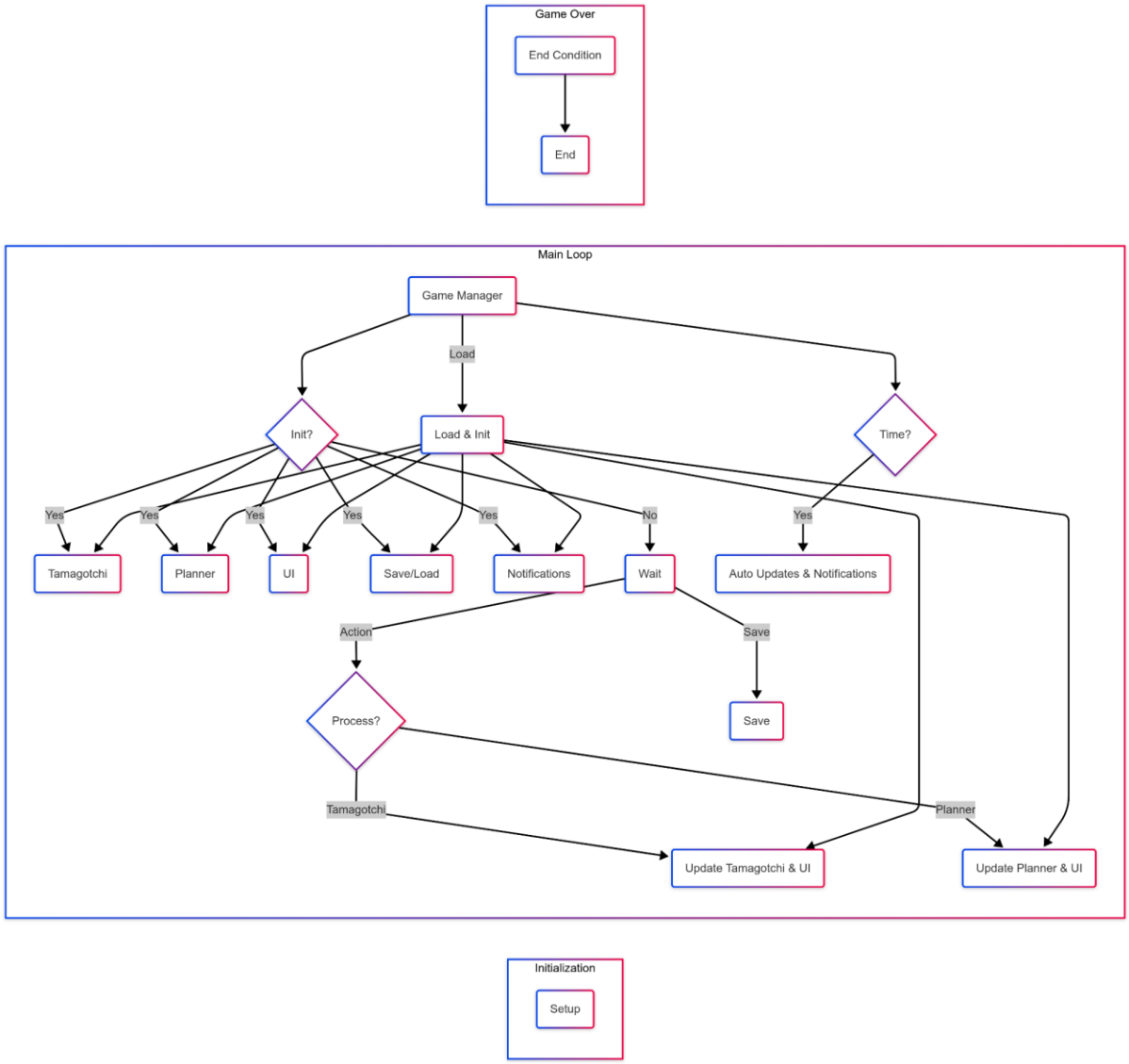



Рис. 3.1 - Діаграма залежностей в архітектурі додатку

Центральним елементом гри є Система Тамагочі, яка відповідає за симуляцію віртуального улюбленця. Вона охоплює керування всіма аспектами його життєдіяльності, включаючи базові потреби (голод, спрага, розваги, енергія, гігієна), емоційний стан (настрій, щастя), фізичні параметри (здоров'я, вік), а також його поведінку та реакцію на взаємодію з гравцем. Ця система також відповідає за моделювання природних процесів, таких як поступове зниження потреб з часом та старіння віртуальної істоти.

Паралельно з системою тамагочі функціонує Система Планера, яка надає гравцеві інструменти для організації свого часу та завдань. Вона дозволяє створювати нові завдання та події, встановлювати терміни їх виконання,

переглядати їх у форматі календаря або списку, редагувати існуючі записи та відмічати їх як виконані. Ця система інтегрована з ігровим світом, надаючи гравцеві практичну користь від використання планера в контексті гри.

Для забезпечення візуальної взаємодії з гравцем розроблено Систему Інтерфейсу Користувача (UI). Ця система відповідає за відображення всієї необхідної інформації, включаючи поточний стан тамагочі (рівень потреб, настрій, здоров'я), елементи планера (календар, список завдань, форми для введення даних), а також надає інтерактивні елементи керування (кнопки, перемикачі тощо) для взаємодії з обома основними системами гри. Ефективний UI є критично важливим для забезпечення зручного та інтуїтивно зрозумілого ігрового досвіду.

Збереження прогресу гравця та можливість повернення до гри в будь-який момент забезпечується Системою Збереження та Завантаження. Ця система відповідає за періодичне або за запитом гравця збереження поточного стану гри, включаючи всі дані про тамагочі та список завдань планера, у постійне сховище. При наступному запуску гри ця ж система відповідає за відновлення збереженого стану, дозволяючи гравцеві продовжити з того місця, де він зупинився.

Для підтримки залученості гравця та своєчасного інформування про важливі події в грі реалізовано Систему Сповіщень. Її функціональність полягає у генерації та відображенні повідомлень, які можуть інформувати про критичне зниження потреб тамагочі, нагадувати про наближення терміну виконання запланованих завдань або повідомляти про інші важливі ігрові події. Сповіщення можуть бути представлені у різних формах, включаючи візуальні підказки та звукові сигнали.

Координацію та управління всіма вищеописаними системами здійснює Система Керування Гри (Game Manager). Ця система відіграє роль центрального оркестратора, відповідаючи за ініціалізацію всіх інших систем при запуску гри, керування загальним ігровим циклом, обробку глобальних ігрових подій та забезпечення їхньої коректної взаємодії.

Застосування модульного підходу в грі "Тамагочі-Планер" дозволило чітко відокремити такі функціональні аспекти, як керування віртуальним улюбленцем (його станом, потребами, поведінкою), планування завдань та подій, відображення

інформації для користувача через інтерфейс, збереження та завантаження ігрового прогресу, а також система сповіщень. Таке відокремлення не тільки полегшує розуміння загальної структури гри, але й значно спрощує подальшу розробку та тестування окремих її компонентів. Крім того, модульна архітектура підвищує стійкість гри до помилок, оскільки проблема в одному модулі менш імовірно вплине на роботу інших незалежних частин системи.

Взаємодія між цими шістьма основними системами здійснюється через чітко визначені інтерфейси та механізми обміну даними (рис. 3.2). Наприклад, коли гравець через елементи UI ініціює дію, таку як годування тамагочі або створення нового завдання в планері, відповідний запит передається до Системи Тамагочі або Системи Планера. Ці системи, у свою чергу, оновлюють свій внутрішній стан відповідно до отриманої команди. Зміни, що відбулися в стані цих систем, потім відображаються в UI для гравця, забезпечуючи зворотний зв'язок. Аналогічно, Система Сповіщень може підписуватися на події, що відбуваються в Системі Тамагочі та Системі Планера, щоб своєчасно інформувати гравця про важливі зміни або наближення термінів.

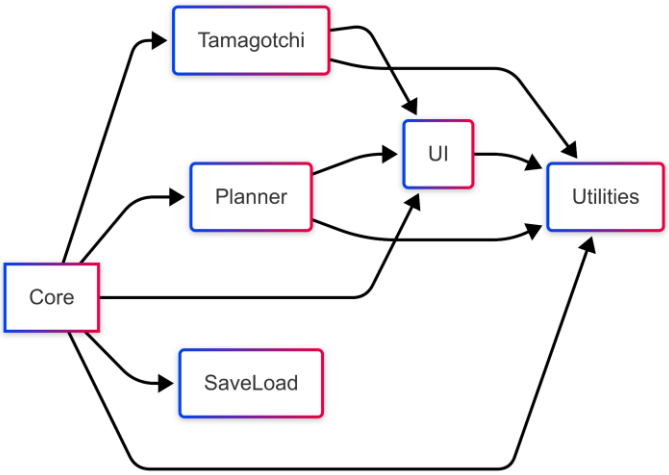



Рис. 3.2 Візуалізація зв’язку модулів коду між собою

Для забезпечення високого рівня організованості кодової бази, полегшення навігації та підтримки чистоти коду, було прийнято рішення про її структурування

за функціональними ознаками. Всі файли проекту розміщені у відповідних папках, назви яких відображають їхнє призначення (рис. 3.3).

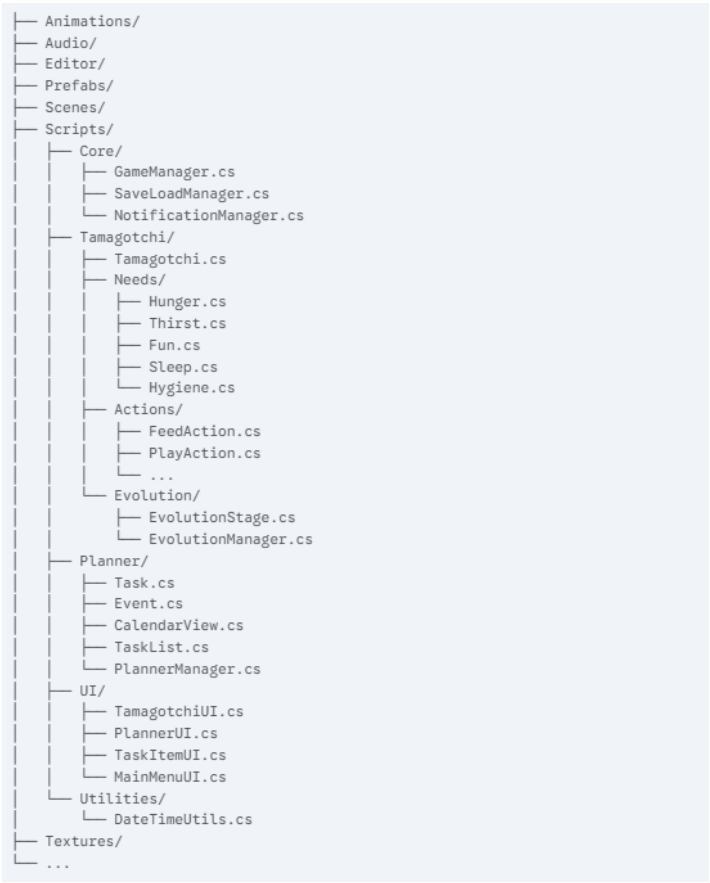



Рис. 3.3 Візуалізація структури гри

Папка “Animations/” містить всі файли анімацій, які використовуються для візуалізації різноманітних дій та станів віртуального улюбленця, а також для анімації елементів користувацького інтерфейсу, що сприяє створенню більш живого та інтерактивного ігрового досвіду. У папці “Audio/” зберігаються всі аудіоресурси гри, включаючи звукові ефекти, що супроводжують дії гравця та реакції тамагочі, а також фонова музика, яка створює відповідну атмосферу під час ігрового процесу.

Папка “Editor/” містить спеціалізовані скрипти, призначені для розширення стандартної функціональності редактора Unity. Ці скрипти можуть включати кастомні інспектори для спрощення налаштування компонентів, інструменти для автоматизації рутинних завдань та інші утиліти, що полегшують процес розробки

та налаштування ігрових об'єктів і сцен. У папці “Prefabs/” зберігаються попередньо налаштовані шаблони ігрових об'єктів, такі як сам тамагочі, різні елементи користувацького інтерфейсу (кнопки, індикатори, вікна), а також інші часто використовувані ігрові сутності. Використання префабів дозволяє багаторазово використовувати готові об'єкти на різних ігрових сценах, що значно прискорює процес розробки та забезпечує консистентність ігрового світу.

Файли, що описують різні екрани або рівні гри, такі як головна сцена з тамагочі та планером, а також інші можливі ігрові сцени (наприклад, міні-ігри або екрани налаштувань), зберігаються у папці “Scenes/”. Основна логіка гри, написана на мові програмування C#, розміщена у папці “Scripts/”. Для подальшої організації ця папка поділяється на кілька підпапок, кожна з яких відповідає за певну функціональну область гри.

У підпапці “Core/” знаходяться основні класи, які керують загальною логікою гри та забезпечують взаємодію між різними системами. До них належать, наприклад, “GameManager.cs”, який відповідає за ініціалізацію гри та керування її життєвим циклом, “SaveLoadManager.cs”, який реалізує механізми збереження та завантаження ігрового стану, та “NotificationManager.cs”, який відповідає за відображення сповіщень для гравця. Підпапка “Tamagotchi/” містить всі скрипти, пов'язані з функціональністю віртуального улюбленця. Тут знаходиться основний клас “Tamagotchi.cs”, а також підпапки “Needs/”, що містить класи для керування окремими потребами тамагочі (наприклад, “Hunger.cs”, “Thirst.cs”), “Actions/”, що описує можливі дії гравця по відношенню до тамагочі (наприклад, “FeedAction.cs”, “PlayAction.cs”), та “Evolution/”, що відповідає за процеси розвитку та еволюції віртуальної істоти.

Функціональність планера реалізована у скриптах, розміщених у підпапці “Planner/”. Тут знаходяться класи для представлення завдань (“Task.cs”) та подій (“Event.cs”), скрипт “CalendarView.cs”, що відповідає за логіку відображення календаря, “TaskList.cs”, який керує списком завдань, та “PlannerManager.cs”, що здійснює загальне управління системою планера. Скрипти, що відповідають за логіку роботи елементів користувацького інтерфейсу, розміщені у підпапці “UI/”.

Тут знаходяться класи, такі як “TamagotchiUI.cs”, що відображає стан тамагочі, “PlannerUI.cs”, що відповідає за відображення планера, “TaskItemUI.cs”, що представляє окремий елемент завдання у списку, та “MainMenuUI.cs”, що керує головним меню гри. Допоміжні класи та функції, які використовуються в різних частинах коду, розміщені у підпапці “Utilities/”. Прикладом такого класу є “DateTimeUtils.cs”, який може містити корисні методи для роботи з датою та часом.

Нарешті, папка “Textures/” містить всі файли текстур, які використовуються для візуалізації ігрових об'єктів та елементів користувацького інтерфейсу, забезпечуючи візуальну привабливість гри. Інші асети, такі як матеріали та шрифти, також зберігаються у відповідних папках.

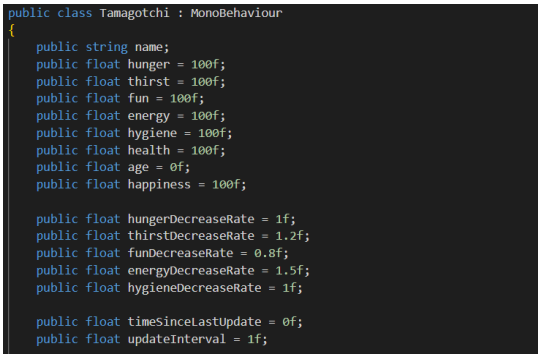
Така детальна та логічна структура кодової бази значно полегшує процес розробки, тестування та підтримки гри, дозволяючи швидко знаходити необхідні скрипти та інші ресурси, а також чітко розуміти функціональну належність різних частин коду.

### 3.2. Реалізація базових механік догляду за віртуальним улюбленцем

В основі функціональності догляду за віртуальним улюбленцем у грі "Тамагочі-Планер" лежить ретельно розроблена система, що моделює ключові потреби та їхню динаміку. Центральним елементом цієї системи є скрипт “Tamagotchi.cs”, який відповідає за зберігання та оновлення внутрішнього стану віртуальної істоти. Цей скрипт оголошує ряд публічних змінних (рис. 3.4), що відображають поточний рівень основних потреб тамагочі: “hunger”, “thirst”, “fun”, “energy” та “hygiene”. Кожна з цих змінних є числовим значенням з плаваючою комою в діапазоні від 0 до 100, де 100 представляє повністю задоволену потребу, а 0 – критичний рівень. Крім того, скрипт містить змінні “health”, що відображає загальний стан здоров'я тамагочі, “age”, що представляє його вік в умовних ігрових одиницях часу, та “happiness”, що відображає його емоційний стан.

Для імітації природного процесу зниження потреб з часом у класі Tamagotchi визначено змінні, що контролюють швидкість цього зниження:

“hungerDecreaseRate”, “thirstDecreaseRate”, “funDecreaseRate”, “energyDecreaseRate” та “hygieneDecreaseRate”. Ці змінні визначають, наскільки одиниць відповідна потреба зменшується за кожну одиницю ігрового часу. Для керування частотою оновлення стану використовується змінна “updateInterval”, яка визначає інтервал часу в секундах між послідовними оновленнями потреб.



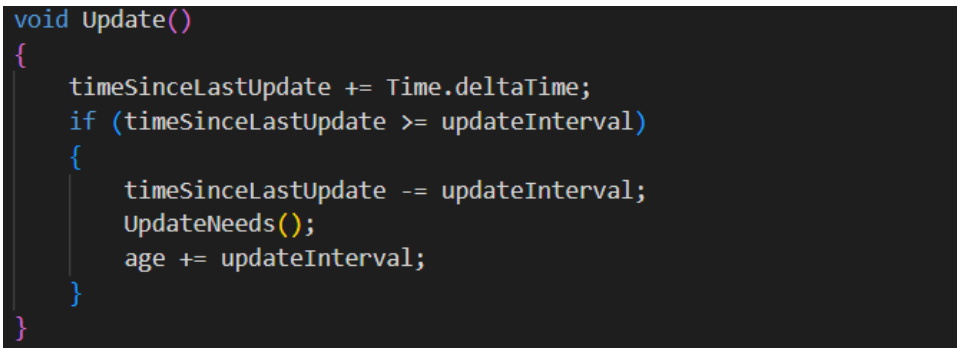
```
public class Tamagotchi : MonoBehaviour
{
    public string name;
    public float hunger = 100f;
    public float thirst = 100f;
    public float fun = 100f;
    public float energy = 100f;
    public float hygiene = 100f;
    public float health = 100f;
    public float age = 0f;
    public float happiness = 100f;

    public float hungerDecreaseRate = 1f;
    public float thirstDecreaseRate = 1.2f;
    public float funDecreaseRate = 0.8f;
    public float energyDecreaseRate = 1.5f;
    public float hygieneDecreaseRate = 1f;

    public float timeSinceLastUpdate = 0f;
    public float updateInterval = 1f;
```

Рис. 3.4 Оголошення ряду публічних змінних

Основна логіка життєвого циклу та оновлення стану тамагочі реалізована в методі “Update()”, який є стандартним методом Unity, що викликається на кожному кадрі. У цьому методі відбувається накопичення часу, що минув з моменту останнього оновлення, у приватній змінній “timeSinceLastUpdate”. Коли значення “timeSinceLastUpdate” досягає або перевищує значення “updateInterval”, викликається приватний метод “UpdateNeeds()”, який і відповідає за зміну рівнів потреб. (рис. 3.5)



```
void Update()
{
    timeSinceLastUpdate += Time.deltaTime;
    if (timeSinceLastUpdate >= updateInterval)
    {
        timeSinceLastUpdate -= updateInterval;
        UpdateNeeds();
        age += updateInterval;
    }
}
```

Рис. 3.5 Метод “Update()”

У методі “UpdateNeeds()” кожна потреба зменшується на величину, що є результатом множення відповідної швидкості зниження на інтервал часу між

оновленнями (“updateInterval”). (рис. 3.6) Для запобігання від'ємним значенням використовується функція “Mathf.Max(0, ...)”. Після оновлення потреб викликаються методи “CalculateHappiness()” та “CalculateHealth()” для перерахунку поточного рівня щастя та здоров'я тамагочі відповідно. Важливою частиною механізму є використання делегатів та подій (“OnNeedsChanged” та “OnHealthChanged”). Після кожного оновлення потреб та здоров'я викликаються ці події, що дозволяє іншим компонентам гри (наприклад, елементам інтерфейсу користувача) реагувати на ці зміни та оновлювати відображену інформацію.

```
void UpdateNeeds()
{
    hunger = Mathf.Max(0, hunger - hungerDecreaseRate * updateInterval);
    thirst = Mathf.Max(0, thirst - thirstDecreaseRate * updateInterval);
    fun = Mathf.Max(0, fun - funDecreaseRate * updateInterval);
    energy = Mathf.Max(0, energy - energyDecreaseRate * updateInterval);
    hygiene = Mathf.Max(0, hygiene - hygieneDecreaseRate * updateInterval);

    CalculateHappiness();
    CalculateHealth();

    NeedsChanged?.Invoke(hunger, thirst, fun, energy, hygiene, happiness);
}
```

Рис. 3.6 Робота методу “UpdateNeeds()”

Рівень щастя тамагочі обчислюється в методі “CalculateHappiness()” як усереднене значення всіх основних потреб, нормалізоване до діапазону від 0 до 100. Здоров'я тамагочі, що визначається в методі “CalculateHealth()”, залежить від того, наскільки задоволені його базові потреби. (рис. 3.7) Якщо будь-яка з потреб опускається нижче певного критичного значення (у цьому прикладі 20), здоров'я тамагочі починає зменшуватися. Якщо значення здоров'я досягає нуля або менше, викликається метод “Die()”, який сигналізує про смерть віртуального улюбленця та може ініціювати відповідні ігрові наслідки.

```
void CalculateHappiness()
{
    happiness = Mathf.Clamp01((hunger + thirst + fun + energy + hygiene) / 500f) * 100f;
}

void CalculateHealth()
{
    health = 100f;
    if (hunger <= 20 || thirst <= 20 || fun <= 20 || energy <= 20 || hygiene <= 20)
    {
        health -= 10f;
    }
    health = Mathf.Clamp(0f, 100f, health);
    HealthChanged?.Invoke(health);
    if (health <= 0)
    {
        Die();
    }
}
```

Рис. 3.7 Робота методів “CalculateHappiness()” та “CalculateHealth()”

Для забезпечення можливості гравця піклуватися про тамагочі та задовольняти його потреби, у класі Tamagotchi реалізовано ряд публічних методів, що відповідають основним діям догляду: “Feed(float amount)”, “Drink(float amount)”, “Play(float amount)”, “SleepAction(float amount)” та “Clean(float amount)”. Кожен з цих методів приймає параметр amount, що визначає кількість ресурсу, який гравець надає тамагочі. При виклику одного з цих методів відповідна потреба тамагочі збільшується на передане значення, але не може перевищувати максимальне значення 100. Наприклад, виклик методу “Feed(20f)” збільшить поточний рівень голоду тамагочі на 20 одиниць, але якщо поточний рівень голоду становить 90, то після виклику він стане 100. (рис. 3.8)

```
public void Feed(float amount)
{
    hunger = Mathf.Min(100f, hunger + amount);
}

public void Drink(float amount)
{
    thirst = Mathf.Min(100f, thirst + amount);
}

public void Play(float amount)
{
    fun = Mathf.Min(100f, fun + amount);
}

public void SleepAction(float amount)
{
    energy = Mathf.Min(100f, energy + amount);
}

public void Clean(float amount)
{
    hygiene = Mathf.Min(100f, hygiene + amount);
}
```

Рис. 3.8 Ряд публічних методів, що відповідають основним діям догляду

Для візуалізації стану тамагочі та забезпечення інтерфейсу для взаємодії з ним використовується скрипт “TamagotchiUI.cs”. Цей компонент відповідає за відображення поточних рівнів потреб та здоров'я тамагочі за допомогою елементів UI, таких як повзунки (Slider) та текстові поля (TMP_Text). У методі “Start()” скрипт отримує посилання на компонент Tamagotchi, підписується на події “NeedsChanged” та “HealthChanged” (рис. 3.9) для автоматичного оновлення інтерфейсу при зміні стану тамагочі, а також відображає ім'я тамагочі, якщо відповідне текстове поле призначено. У методах “UpdateNeedsUI()” та “UpdateHealthUI()” відбувається оновлення значень повзунків на основі отриманих даних про потреби та здоров'я.

```
void Start()
{
    if (tamagotchi != null)
    {
        UpdateUI();
        Tamagotchi.NeedsChanged += UpdateNeedsUI;
        Tamagotchi.HealthChanged += UpdateHealthUI;
        if (nameText != null)
        {
            nameText.text = tamagotchi.name;
        }
    }
    else
    {
        Debug.LogError("TamagotchiUI: Tamagotchi компонент не призначено!");
    }
}
```

Рис. 3.9 Метод “Start()”, події “NeedsChanged” та “HealthChanged”

Крім того, скрипт “TamagotchiUI.cs” містить обробники подій для кнопок інтерфейсу (рис. 3.10), що відповідають діям годування (“OnFeedButton()”), напування (“OnDrinkButton()”), гри (“OnPlayButton()”), сну (“OnSleepButton()”) та гігієни (“OnCleanButton()”). При натисканні однієї з цих кнопок викликається відповідний метод класу “Tamagotchi” (наприклад, “tamagotchi.Feed(20f)”) з певним фіксованим значенням кількості наданого ресурсу.

```
public void OnFeedButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Feed(20f);
    }
}

public void OnDrinkButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Drink(20f);
    }
}

public void OnPlayButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Play(30f);
    }
}

public void OnSleepButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.SleepAction(40f);
    }
}

public void OnCleanButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Clean(25f);
    }
}
```

Рис. 3.10 Обробники подій для кнопок інтерфейсу

Реалізація базових механік догляду у грі "Тамагочі-Планер" формує нерозривний зв'язок між гравцем та віртуальною істотою, створюючи ігровий цикл, що базується на відповідальності та емпатії. Потреби тамагочі не є статичними величинами; вони характеризуються постійною динамікою зменшення з плином

внутрішньоігрового часу. Ця безперервна деградація потреб імітує біологічні імперативи живого організму, такі як голод, спрага, потреба у розвагах, відпочинку та гігієні. Саме ця невпинна зміна стану вимагає від гравця регулярної уваги та активного втручання в життя віртуального улюбленця. Ігнорування цих сигналів та несвоєчасне задоволення потреб призводить до поступового погіршення самопочуття тамагочі, що може проявлятися у зниженні настрою, рівня щастя та, зрештою, у погіршенні здоров'я.

Успішний та своєчасний догляд, навпаки, є ключем до підтримання високого рівня щастя та здоров'я тамагочі. Регулярне годування, надання води, забезпечення розваг, можливість відпочинку та підтримання гігієни не лише запобігають негативним наслідкам, але й сприяють емоційному благополуччю віртуального улюбленця. Високий рівень задоволення потреб безпосередньо впливає на його настрій, роблячи його більш активним та комунікабельним, що, в свою чергу, буде винагороджувати гравця позитивними ігровими реакціями. Таким чином, механіка догляду стає центральним елементом ігрового процесу, навколо якого обертається більшість дій гравця. Вона визначає основні завдання, такі як моніторинг поточного стану тамагочі, прогнозування майбутніх потреб та своєчасне реагування на них. Гравець виступає в ролі опікуна, чиї дії безпосередньо впливають на якість життя та тривалість існування віртуального улюбленця, формуючи почуття відповідальності та прив'язаності. Можливість настання негативних наслідків, аж до "смерті" тамагочі внаслідок недбалого догляду, підкреслює важливість цієї системи та робить її не просто набором рутинних дій, а ключовим елементом, що визначає емоційну глибину та залученість гравця в ігровий процес, завдяки чому користувач буде частіше заходити в додаток і, відповідно, частіше бачити свої завдання і події, що записані.

### 3.3. Інтеграція планувальника та системи мотивації

Базова система догляду за віртуальним улюбленцем у "Тамагочі-Планер" створює фундаментальний цикл взаємодії, де гравець несе відповідальність за добробут своєї віртуальної істоти. Однак для поглиблення ігрового процесу та стимулювання більш активної та цілеспрямованої діяльності гравця, передбачається інтеграція функціональності планера з додатковою системою мотивації. Ця система базуватиметься на винагородженні гравця за успішне виконання запланованих завдань у вигляді внутрішньоігрової валюти, наприклад, "монеток". Зароблені таким чином ресурси зможуть бути використані для придбання різноманітних благ для віртуального улюбленця, таких як покращена їжа або цікаві розваги, створюючи прямий зв'язок між продуктивністю гравця у плануванні та рівнем задоволення потреб його тамагочі. Для інтеграції функціональності планера та системи мотивації гравця знадобиться кілька нових скриптів, а також оновлення існуючих скриптів “Tamagotchi.cs” та “TamagotchiUI.cs”.

Для початку варто почати з системи планування (рис. 3.11), адже вже до неї буде підключена система мотивації внутрішньоігрової валюти.

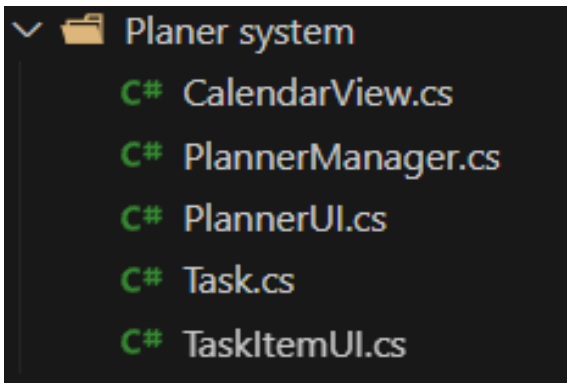



Рис. 3.11 Коди, необхідні для реалізації планера

Система планування в додатку буде складатися з п’яти зв’язаних між собою кодів. “Task.cs” визначає структуру одного завдання планера (назва, опис, дата виконання, статус виконання). “PlannerManager.cs” - керує списком завдань, додаванням, видаленням, редагуванням та відмічанням виконання завдань. Код “CalendarView.cs” відповідає за відображення календаря та завдань на ньому.

“TaskItemUI.cs” відображає одне завдання у списку завдань в інтерфейсі користувача. Ну і код “PlannerUI.cs”, в свою чергу, керує відображенням планера, списку завдань та елементів керування. Давайте розглянемо кожен з цих кодів детальніше:

Почнемо з коду для визначення структури кожного завдання (рис.3.12). “Task.cs” використовується для представлення окремого завдання планера. Він визначає структуру даних, необхідних для опису завдання.

```
using System;

Windsurf: Refactor | Explain
[Serializable]
1 reference
public class Task
{
    1 reference
    public string title;
    1 reference
    public string description;
    1 reference
    public DateTime dueDate;
    2 references
    public bool isCompleted;
    0 references
    public int rewardPoints = 10; // Кількість монеток за виконання завдання

    0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
    public Task(string title, string description, DateTime dueDate)
    {
        this.title = title;
        this.description = description;
        this.dueDate = dueDate;
        this.isCompleted = false;
    }

    0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
    public void MarkAsCompleted()
    {
        isCompleted = true;
    }
}
```

Рис. 3.12 Повний код для структури завдань

На початку цього коду ми бачимо задані публічні змінні. Кожне завдання має назву (title), опис (description), дату виконання (dueDate), статус виконання (isCompleted) та кількість нагородних балів (rewardPoints, за замовчуванням стоїть 10 балів(“монеток”)), які гравець отримує за його виконання. публічний об’єкт “Task” ініціалізує кожне нове завдання, таким чином задаючи йому потрібні характеристики. Особливою деталлю є те, що у кожного нового завдання статус виконання є заздалегідь негативним - не виконаним. Це зроблено для того, щоб користувач не наплутав нічого при створенні. Окремо також прописано метод

“MarkAsCompleted()”, який перемикає статус завдання на “виконано” після відповідної дії з UI елементом конкретного завдання.

Керування колекцією завдань та операціями над ними здійснює скрипт “PlannerManager.cs”. Цей скрипт є ключовим компонентом системи планера, відповідаючи за керування колекцією всіх завдань. Він реалізований як Singleton, що означає, що в грі існуватиме лише один екземпляр цього класу, забезпечуючи централізований доступ до функціональності планера (рис. 3.13). Статична властивість Instance надає цей єдиний екземпляр. Внутрішньо клас містить приватну змінну tasks, яка є списком об'єктів типу Task, де зберігаються всі активні завдання користувача. Для сповіщення інших частин гри про зміни у списку завдань використовується делегат OnTaskListChanged та статична подія TaskListChanged. Будь-який компонент, зацікавлений у відстеженні змін у списку завдань (наприклад, UI), може підписатися на цю подію.

```
1 reference | Windsurf: Refactor | Explain
5  public class PlannerManager : MonoBehaviour
6  {
       2 references
7      public static PlannerManager Instance { get; private set; }
       9 references
8      public List<Task> tasks = new List<Task>();
9
       1 reference
10     public delegate void OnTaskListChanged();
       3 references
11     public static event OnTaskListChanged TaskListChanged;
```

Рис. 3.13 Ініціалізація в класі PlannerManager

Метод “Awake()” є стандартним методом Unity, який викликається при створенні екземпляра скрипта. У ньому реалізована логіка Singleton: якщо екземпляр Instance ще не існує (значення дорівнює null), поточний об'єкт стає цим екземпляром і автоматично позначається самим Unity як такий, що не буде знищуватися при завантаженні нових сцен (DontDestroyOnLoad). Якщо ж екземпляр вже існує, поточний об'єкт знищується, щоб запобігти дублюванню. У цьому методі також передбачено місце для завантаження збережених завдань, яка визивається з коду “SaveLoadManager.cs”.

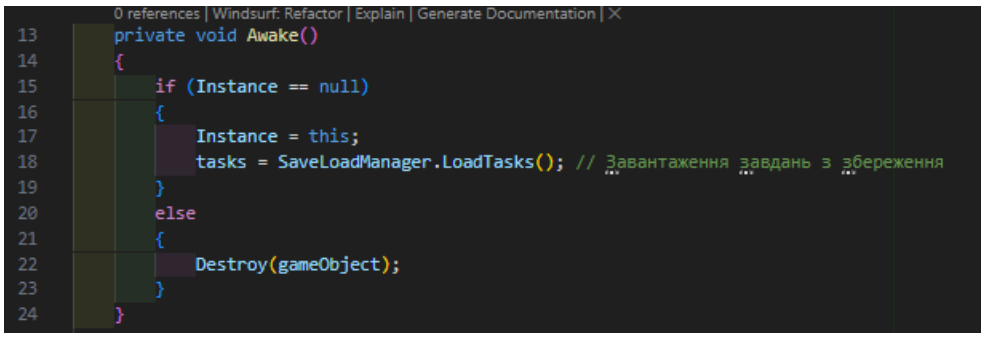



Рис. 3.14 Метод Awake()

Метод “AddNewTask()” (рис. 3.15) створює новий об'єкт Task з переданими назвою, описом та датою виконання, додає його до списку tasks та викликає подію “TaskListChanged”, щоб повідомити про зміни.

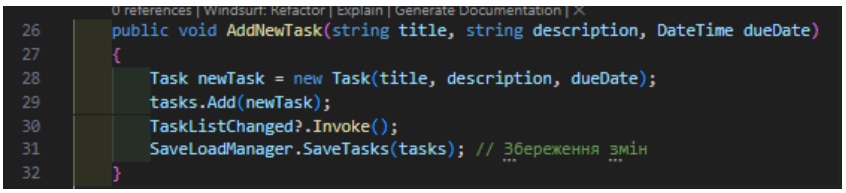



Рис. 3.15 Метод “AddNewTask()”

Аналогічно, метод “RemoveTask()”, що видаляє передане завдання зі списку, та метод “ToggleTaskCompletion()”, що змінює статус виконання переданого завдання на протилежний, також викликають подію про зміни та їх збереження (рис. 3.16).

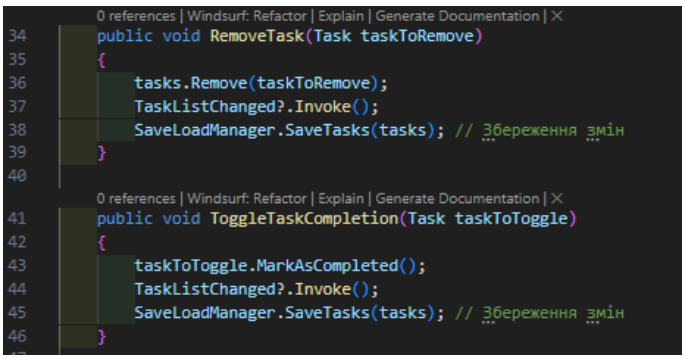



Рис. 3.16 Методи “RemoveTask()” та “ToggleTaskCompletion()” в коді

Метод “GetCurrentTasks()” (рис. 3.17) повертає поточний список усіх завдань.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | X
48  public List<Task> GetCurrentTasks()
49  {
50      return tasks;
51  }
52
```

Рис. 3.17 Метод “GetCurrentTasks()”

Метод “GetTasksForDate()” (рис. 3.18) приймає дату як аргумент та повертає новий список, що містить лише ті завдання, дата виконання яких співпадає з переданою датою.

```
53  public List<Task> GetTasksForDate(DateTime date)
54  {
55      return tasks.Where(task => task.dueDate.Date == date.Date).ToList();
56  }
57
```

Рис. 3.18 Метод “GetTasksForDate()”

Метод “GetRewardForCompletedTasks()” (рис. 3.19) призначений для обчислення загальної кількості монеток, які гравець отримав за виконані завдання.

```
1 reference | Windsurf: Refactor | Explain | Generate Documentation | X
58  public int GetRewardForCompletedTasks()
59  {
60      return tasks.Where(task => task.isCompleted && !task.Equals(GetRewardForCompletedTasks())).Sum(task => task.rewardPoints);
61  }
62  }
```

Рис. 3.19 Метод “GetRewardForCompletedTasks()”

Наступним розглянемо “CalendarView.cs”. Цей скрипт відповідає за візуалізацію календаря в інтерфейсі користувача. Він має публічні змінні (рис. 3.20) для префаба одного дня (dayPrefab), контейнера для розміщення днів (calendarContainer) та текстового поля для відображення назви місяця та року (monthYearText). Приватна змінна currentDate типу DateTime зберігає поточний відображуваний місяць. Приватний список tasks зберігає посилання на поточний список завдань з PlannerManager.

```
0 references | Windsurf: Refactor | Explain
public class CalendarView : MonoBehaviour
{
    2 references
    public Transform dayPrefab;
    3 references
    public Transform calendarContainer;
    1 reference
    public TextMeshProUGUI monthYearText;

    11 references
    private DateTime currentDate = DateTime.Now;
    1 reference
    private List<Task> tasks = new List<Task>();
```

Рис. 3.20 Публічні змінні коду CalendarView

Метод InitializeCalendar() (рис. 3.21) викликає метод UpdateCalendar() для початкового відображення календаря.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
public void InitializeCalendar()
{
    UpdateCalendar();
}
```

Рис. 3.21 Метод InitializeCalendar()

Методи ShowNextMonth() та ShowPreviousMonth() (рис. 3.22) змінюють значення currentDate на наступний або попередній місяць відповідно та викликають UpdateCalendar() для оновлення відображення.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
public void ShowNextMonth()
{
    currentDate = currentDate.AddMonths(1);
    UpdateCalendar();
}

0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
public void ShowPreviousMonth()
{
    currentDate = currentDate.AddMonths(-1);
    UpdateCalendar();
}
```

Рис. 3.22 Методи ShowNextMonth() та ShowPreviousMonth()

Основна логіка відображення календаря знаходиться в методі UpdateCalendar() (рис. 3.23). Він спочатку очищає контейнер від попередніх відображень днів. Потім встановлює текст назви місяця та року. Визначає кількість

днів у поточному місяці та день тижня, з якого починається місяць. Створює порожні клітинки на початку календаря для правильного вирівнювання першого дня місяця. Далі, у циклі для кожного дня місяця, створює новий екземпляр префаба дня, отримує доступ до текстового компонента для відображення номера дня та компонента зображення для можливої індикації. Відображає номер дня у текстовому полі. Потім отримує список завдань на поточний день з PlannerManager і, якщо на цей день є завдання, змінює колір фону клітинки дня для візуальної індикації.

```
33      void UpdateCalendar()
34      {
35          foreach (Transform child in calendarContainer)
36          {
37              Destroy(child.gameObject);
38          }
39
40          monthYearText.text = currentDate.ToString("MMMM yyyy");
41
42          int daysInMonth = DateTime.DaysInMonth(currentDate.Year, currentDate.Month);
43          DateTime firstDayOfMonth = new DateTime(currentDate.Year, currentDate.Month, 1);
44          int firstDayOfWeek = (int)firstDayOfMonth.DayOfWeek == 0 ? 7 : (int)firstDayOfMonth.DayOfWeek;
45
46          for (int i = 1; i < firstDayOfWeek; i++)
47          {
48              Instantiate(dayPrefab, calendarContainer);
49          }
50
51          for (int day = 1; day <= daysInMonth; day++)
52          {
53              DateTime currentDay = new DateTime(currentDate.Year, currentDate.Month, day);
54              Transform dayObject = Instantiate(dayPrefab, calendarContainer);
55              TextMeshProUGUI dayText = dayObject.GetComponentInChildren<TextMeshProUGUI>();
56              Image background = dayObject.GetComponent<Image>();
57
58              if (dayText != null)
59              {
60                  dayText.text = day.ToString();
61              }
62
63              List<Task> tasksOnDay = PlannerManager.Instance.GetTasksForDate(currentDay);
64              if (tasksOnDay.Count > 0)
65              {
66                  background.color = Color.yellow; // Візуальна індикація наявності завдань
67              }
68          }
69      }
70
```

Рис. 3.23 Метод UpdateCalendar()

Методи OnEnable() та OnDisable() (рис. 3.24) відповідають за підписку та відписку від події TaskListChanged в PlannerManager. При активації скрипта він отримує поточний список завдань та підписується на подію, а при деактивації відписується, щоб уникнути витоків пам'яті. Метод UpdateCalendar() також викликається при виникненні події TaskListChanged, щоб оновити відображення календаря при зміні списку завдань.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
71      void OnEnable()
72      {
73          if (PlannerManager.Instance != null)
74          {
75              tasks = PlannerManager.Instance.GetCurrentTasks();
76              PlannerManager.TaskListChanged += UpdateCalendar;
77              UpdateCalendar();
78          }
79      }
80
        0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
81      void OnDisable()
82      {
83          if (PlannerManager.Instance != null)
84          {
85              PlannerManager.TaskListChanged -= UpdateCalendar;
86          }
87      }
88  }
```

Рис. 3.24 Методи OnEnable() та OnDisable()

Скрипт “TaskItemUI.cs” керує відображенням одного елемента завдання у списку завдань. Він має публічні змінні (рис. 3.25) для текстових полів назви (titleText) та дати виконання (dueDateText), а також для перемикача статусу виконання (completionToggle). Приватна змінна task зберігає посилання на відповідний об'єкт Task.

```
0 references | Windsurf: Refactor | Explain
5   public class TaskItemUI : MonoBehaviour
6   {
        1 reference
7       public TMP_Text titleText;
        1 reference
8       public TMP_Text dueDateText;
        1 reference
9       public Toggle completionToggle;
        9 references
10      private Task task;
```

Рис. 3.25 Публічні змінні коду TaskItemUI

Метод SetTask() (рис. 3.26) приймає об'єкт Task як аргумент та заповнює текстові поля назви та дати виконання, а також встановлює стан перемикача відповідно до стану виконання завдання.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | ×
12      public void SetTask(Task currentTask)
13      {
14          task = currentTask;
15          titleText.text = task.title;
16          dueDateText.text = task.dueDate.ToString("dd.MM.yyyy");
17          completionToggle.isOn = task.isCompleted;
18      }
19
```

Рис. 3.26 Метод SetTask()

Метод OnToggleValueChanged() (рис.3.27) викликається при зміні значення перемикача статусу виконання. Якщо значення перемикача змінилося порівняно зі станом виконання завдання, він викликає метод ToggleTaskCompletion() в PlannerManager для оновлення статусу завдання.

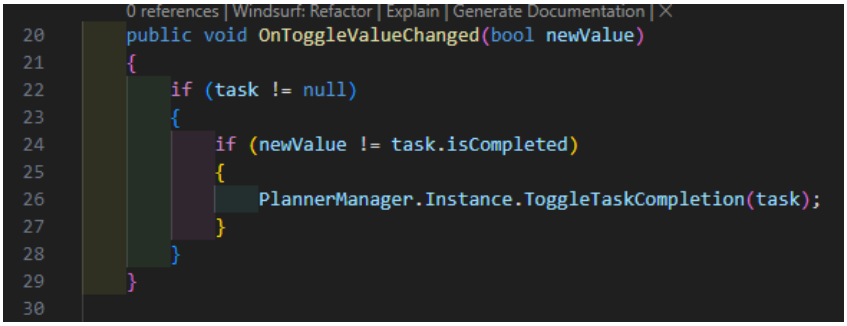



Рис. 3.27 Метод OnToggleValueChanged()

Метод OnRemoveButtonClicked() (рис. 3.28) викликається при натисканні кнопки видалення. Якщо пов'язане завдання існує, він викликає метод RemoveTask() в PlannerManager для його видалення та знищує сам GameObject елемента завдання.

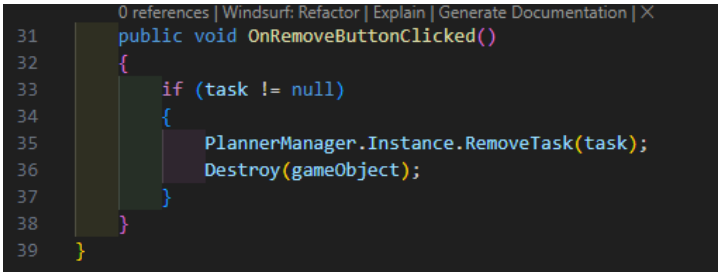



Рис. 3.28 Метод OnRemoveButtonClicked()

Скрипт “PlannerUI.cs” відповідає за керування відображенням усього інтерфейсу планера, включаючи список завдань та форму для додавання нових завдань. Він має публічні змінні (рис. 3.29) для контейнера списку завдань (taskListContainer), префаба елемента завдання (taskItemPrefab), полів введення назви (newTaskTitleInput) та опису (newTaskDescriptionInput) нового завдання, змінної для зберігання дати виконання нового завдання (newTaskDueDate) та текстового поля для відображення цієї дати (dueDateText).

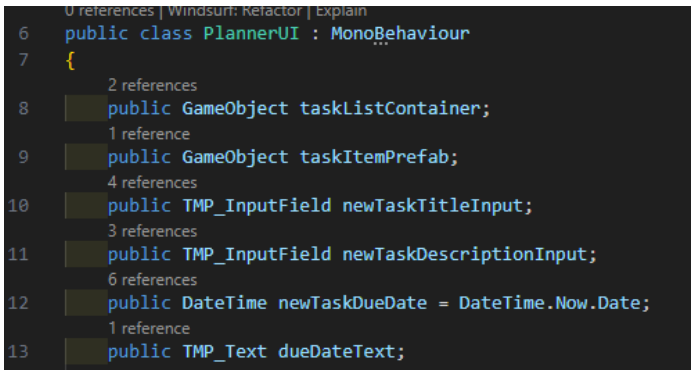



Рис. 3.29 Публічні змінні скрипту PlannerUI

Методи OnEnable() та OnDisable() (рис. 3.30) керують оновленням списку завдань при активації панелі планера та очищенням полів введення та скиданням дати при деактивації.

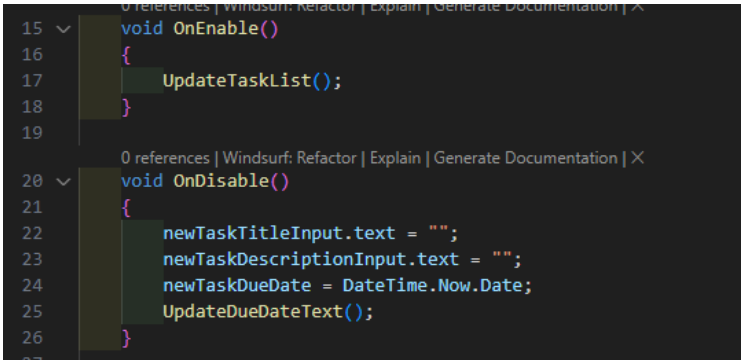



Рис. 3.30 Методи OnEnable() та OnDisable()

Метод UpdateTaskList() (рис. 3.31) очищає контейнер списку завдань від попередніх елементів та створює нові елементи на основі поточного списку завдань, отримуючи його з PlannerManager, і встановлює для кожного елемента відповідний об'єкт Task за допомогою TaskItemUI.

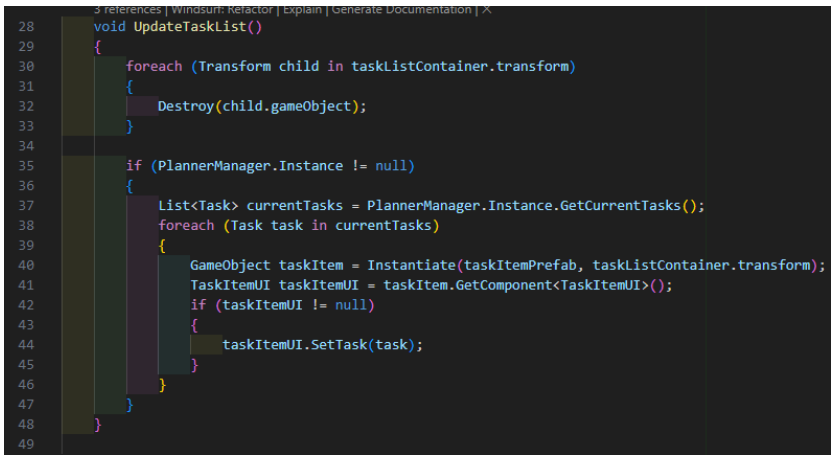



Рис. 3.31 Метод UpdateTaskList()

Метод OnAddTaskButtonClicked() (рис. 3.32) викликається при натисканні кнопки додавання нового завдання. Він перевіряє наявність тексту в полі назви та, якщо є, викликає метод AddNewTask() в PlannerManager для додавання нового завдання з введеними даними та поточною датою виконання, після чого очищає поля введення та скидає дату.

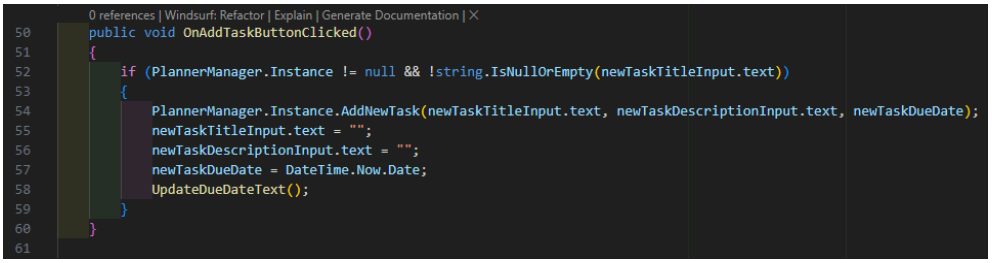



Рис. 3.32 Метод OnAddTaskButtonClicked()

Метод OnSelectDueDateButtonClicked() (рис. 3.33) призначений для вибору дати виконання нового завдання (у наданому коді реалізовано лише перехід на наступний день для прикладу).

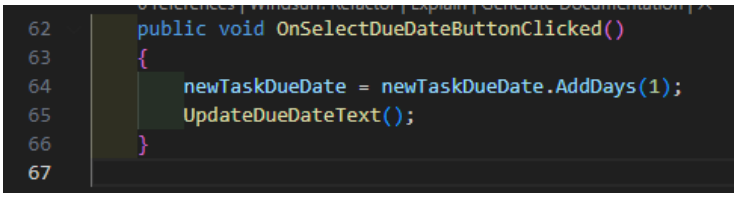



Рис. 3.33 Метод OnSelectDueDateButtonClicked()

Метод UpdateDueDateText() (рис. 3.34) оновлює відображення поточної вибраної дати виконання.

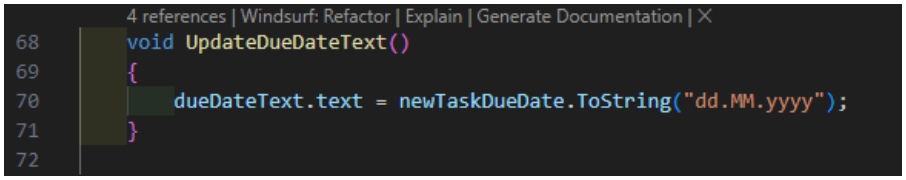



Рис. 3.34 Метод UpdateDueDateText()

Метод Awake() (рис. 3.35) підписується на подію TaskListChanged в PlannerManager для автоматичного оновлення списку завдань при його зміні та викликає UpdateDueDateText() для початкового відображення дати.

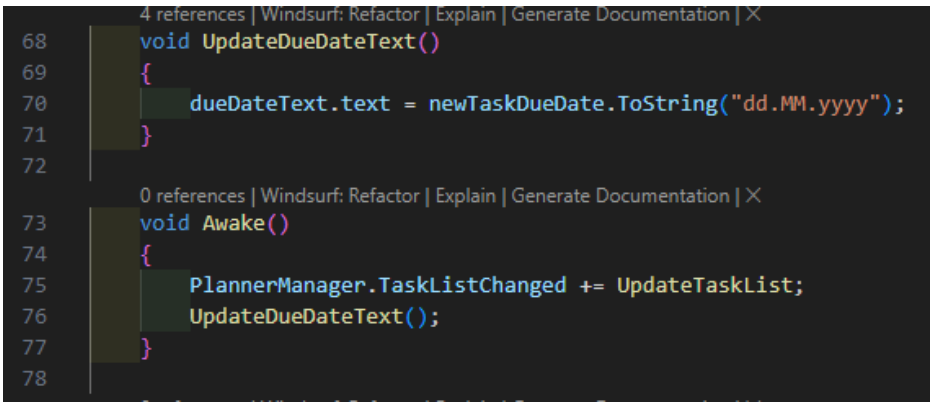



Рис. 3.35 Методи Awake() та UpdateDueDateText()

Метод OnDestroy() (рис. 3.36) відписується від події TaskListChanged при знищенні об'єкта.

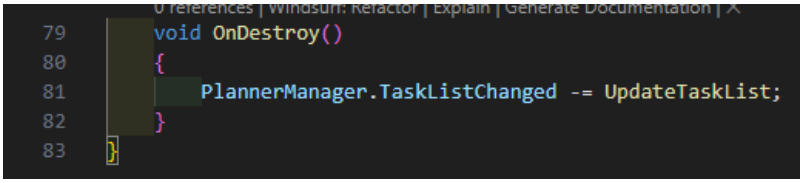



Рис. 3.36 Метод OnDestroy()

Після розгляду кодів, що реалізують основну функціональність планера, перейдемо до опису скриптів, які відповідають за систему мотивації гравця, а саме нарахування та використання внутрішньоігрової валюти. За це відповідають два скрипти, а саме “RewardManager.cs”, який відповідає за нарахування та облік внутрішньоігрової валюти (монеток) за виконання завдань, та “CurrencyUI.cs”, який відображає поточну кількість монеток гравця в інтерфейсі користувача

Скрипт “RewardManager.cs”, як і минулий скрипт-менеджер, є Singleton-класом, що керує внутрішньоігровою валютою. Він має статичну властивість Instance для доступу до єдиного екземпляра. Публічна змінна currentCurrency зберігає поточну кількість монеток гравця. Для сповіщення про зміни валюти використовуються делегат OnCurrencyChanged та статична подія CurrencyChanged. (рис. 3.37)

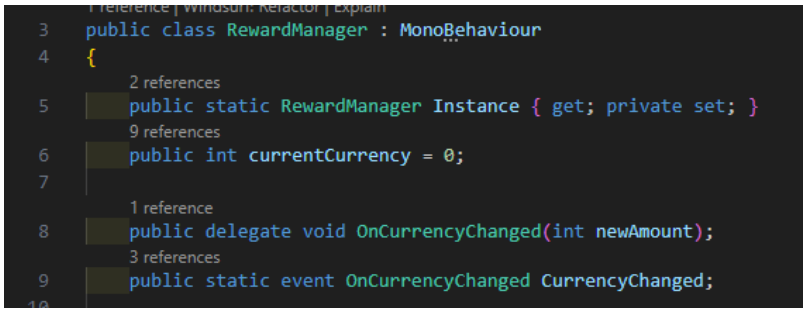


Рис. 3.37 Початок коду RewardManager

Метод Awake() (рис. 3.38) реалізує логіку Singleton та передбачає завантаження збереженої кількості валюти.

```
11      private void Awake()
12      {
13          if (Instance == null)
14          {
15              Instance = this;
16              currentCurrency = SaveLoadManager.LoadCurrency();
17              CurrencyChanged?.Invoke(currentCurrency);
18          }
19          else
20          {
21              Destroy(gameObject);
22          }
23      }
24
```

Рис. 3.38 Метод Awake()

Метод AddCurrency() (рис. 3.39) збільшує поточну кількість валюти на передане значення та викликає подію CurrencyChanged.

```
25      public void AddCurrency(int amount)
26      {
27          currentCurrency += amount;
28          CurrencyChanged?.Invoke(currentCurrency);
29          SaveLoadManager.SaveCurrency(currentCurrency);
30      }
31
```

Рис. 3.39 Метод AddCurrency()

Метод SpendCurrency() (рис. 3.40) намагається витратити передану кількість валюти. Якщо на балансі достатньо коштів, він зменшує currentCurrency, викликає подію та повертає true; інакше повертає false.

```
32      public bool SpendCurrency(int amount)
33      {
34          if (currentCurrency >= amount)
35          {
36              currentCurrency -= amount;
37              CurrencyChanged?.Invoke(currentCurrency);
38              SaveLoadManager.SaveCurrency(currentCurrency);
39              return true;
40          }
41          return false;
42      }
43
```

Рис. 3.40 Метод SpendCurrency()

Методи OnEnable() та OnDisable() (рис. 3.41) керують підпискою та відпискою від події TaskListChanged в PlannerManager.


Рис. 3.41 Методи OnEnable() та OnDisable()

Метод CheckForCompletedTasks() (рис. 3.42) викликається при зміні списку завдань та перевіряє, чи є нові виконані завдання, за які ще не було нараховано нагороду (потрібна додаткова логіка для відстеження). Якщо такі завдання є, він отримує сумарну нагороду та додає її до балансу гравця за допомогою AddCurrency().

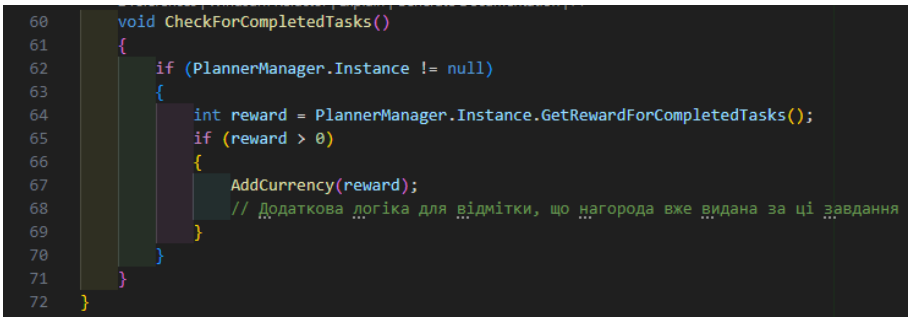



Рис. 3.42 Метод CheckForCompletedTasks()

Скрипт “CurrencyUI.cs” відповідає за відображення поточної кількості валюти в інтерфейсі користувача. Він має публічну змінну currencyText (рис. 3.43) типу TMP_Text для відображення значення.

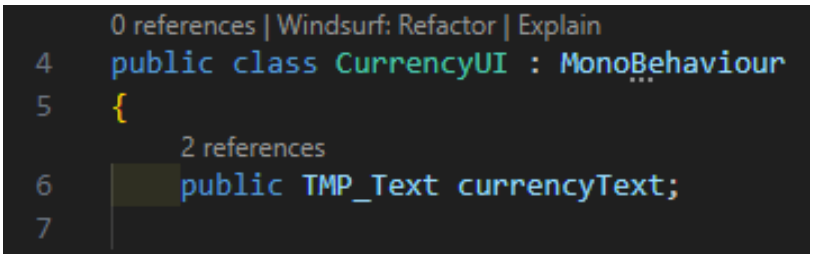



Рис. 3.43 Публічна змінна currencyText в коді CurrencyUI

У методі Start() (рис. 3.44) він отримує початкове значення валюти з RewardManager та підписується на подію CurrencyChanged для оновлення відображення при зміні балансу.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | X
8   void Start()
9   {
10      UpdateCurrencyText(RewardManager.Instance.currentCurrency);
11      RewardManager.CurrencyChanged += UpdateCurrencyText;
12  }
```

Рис. 3.44 Метод Start()

Метод OnDestroy() (рис.3.45) відписується від події.

```
0 references | Windsurf: Refactor | Explain | Generate Documentation | X
14  void OnDestroy()
15  {
16      RewardManager.CurrencyChanged -= UpdateCurrencyText;
17  }
18
```

Рис. 3.45 Метод OnDestroy()

Метод UpdateCurrencyText() (рис. 3.46) оновлює текст currencyText новим значенням валюти.

```
3 references | Windsurf: Refactor | Explain | Generate Documentation | X
19  void UpdateCurrencyText(int newAmount)
20  {
21      if (currencyText != null)
22      {
23          currencyText.text = newAmount.ToString();
24      }
25  }
26 }
```

Рис. 3.46 Метод UpdateCurrencyText()

Врахувавши все вищесказане, інтеграція системи планера та мотивації гравця в "Тамагочі-Планер" досягається за допомогою набору взаємопов'язаних скриптів. Разом ці компоненти створюють цілісну систему, яка не лише допомагає гравцеві організовувати свій час, але й мотивує його до виконання запланованих справ через винагороди, які безпосередньо впливають на добробут його тамагочі, поглиблюючи ігровий процес та додаючи йому стратегічної глибини. Для забезпечення безперервності ігрового досвіду реалізовано механізми збереження та завантаження всіх ключових даних, включаючи стан планера та баланс валюти.

Після написання всіх необхідних кодів та завершення візуальної складової в Unity можна перейти до тестування додатку для виявлення багів та розуміння ефективності прийнятих рішень.

### 3.4. Проведення тестування гри на вибірці користувачів з цільової аудиторії. Оцінка користувацького досвіду та рівня залученості. Перспективи подальших розробок

З метою оцінки користувацького досвіду та рівня залученості гри "Тамагочі-Планер" для її цільової аудиторії – осіб з розладами дефіциту уваги та гіперактивності (РДУГ), розладами аутистичного спектру (РАС) та подібними особливостями нейророзвитку – було проведено якісне та кількісне тестування. До участі у тестуванні було залучено 15 осіб (8 з діагнозом РДУГ, 5 з діагнозом РАС та 2 особи з іншими особливостями нейророзвитку), віком від 16 до 35 років. Процес тестування складався з двох етапів: ознайомлення з грою та самостійне використання протягом однієї години, після чого проводилося індивідуальне структуроване інтерв'ю та заповнення опитувальника, що складався з загальних питань про враження від гри та її ефективності для кожної групи тестувальників.
Під час ознайомлення тестувальникам надавалася коротка інструкція щодо основних механік гри, включаючи догляд за тамагочі та використання планера. Протягом місяця самостійного використання фіксувалися основні дії користувачів, частота їхньої взаємодії з різними елементами гри та будь-які помічені труднощі у навігації чи розумінні механік. Після завершення ігрового часу проводилося інтерв'ю, що охоплювало питання щодо загального враження від гри, зручності інтерфейсу, зрозумілості механік догляду та планера, рівня мотивації, викликаної системою винагород, а також будь-які пропозиції щодо покращення. Опитувальник містив шкали Лайкерта для оцінки різних аспектів користувацького досвіду, таких як легкість використання, рівень залученості, зрозумілість інструкцій та візуальна привабливість.

Аналіз спостережень під час ігрового процесу (таб. 3.1) показав, що більшість тестувальників виявили інтерес до механіки догляду за тамагочі вже на початковому етапі. Взаємодія з віртуальним улюбленцем, така як годування та гра, здійснювалася досить активно. Однак, використання
планера викликало дещо більші труднощі у деяких учасників, особливо у розумінні взаємозв'язку між плануванням завдань та отриманням винагород для тамагочі. Кількість створених завдань варіювалася серед учасників, причому деякі зосереджувалися виключно на догляді за тамагочі, ігноруючи функціональність планера.

Таблиця 3.1

Частота використання основних функцій гри за годину тестування (середнє значення на користувача)

| Функція гри | Середня кількість використань |
|---|---|
| Годування тамагочі | 5.2 |
| Гра з тамагочі | 3.8 |
| Створення завдання | 1.5 |
| Відмітка завдання як виконаного | 0.8 |
| Перегляд календаря | 0.5 |

Дані, отримані з опитувальників, були узагальнені для кожної шкали Лайкерта (таб. 3.2). Наприклад, оцінка "легкості використання" за шкалою від 1 (дуже складно) до 5 (дуже легко) показала середній бал 3.8, що свідчить про загальну позитивну оцінку, але з наявністю певних аспектів, які можуть бути покращені. Рівень залученості отримав середній бал 4.1, вказуючи на те, що гра в цілому викликала інтерес у цільової аудиторії. Зрозумілість інструкцій була оцінена на 3.5, що підкреслює необхідність більш чіткого та, можливо, візуального

представлення деяких ігрових механік. Візуальна привабливість гри отримала високу оцінку – 4.5.

Таблиця 3.2

Усереднені оцінки користувацького досвіду за шкалою Лайкерта (1-5)

| Аспект користувацького досвіду | Середня оцінка | Стандартне відхилення |
|---|---|---|
| Легкість використання | 3.8 | 0.7 |
| Рівень залученості | 4.1 | 0.6 |
| Зрозумілість інструкцій | 3.5 | 0.8 |
| Візуальна привабливість | 4.5 | 0.5 |

Результати індивідуальних інтерв'ю надали більш якісну інформацію про конкретні аспекти гри, які сподобалися користувачам та які викликали труднощі. Багато учасників відзначили позитивний вплив віртуального улюбленця, який створював відчуття відповідальності та спонукав до регулярної взаємодії. Система винагород була сприйнята як цікавий мотиватор, проте деякі користувачі висловили нерозуміння, як саме виконання завдань у планері впливає на можливість отримання цих винагород. Були також висловлені пропозиції щодо спрощення інтерфейсу планера, додавання більш наочних інструкцій та візуалізації прогресу виконання завдань.

Дані про частоту використання функцій, показані на графіку 3.1, демонструють, що на початковому етапі тестування користувачі значно частіше взаємодіяли з механіками догляду за тамагочі, ніж з функціями планера. Це може свідчити про більшу інтуїтивність перших або про недостатнє розуміння переваг використання планера для покращення стану тамагочі. Низька частота відмітки завдань як виконаних може бути пов'язана як з невеликою кількістю створених завдань, так і з труднощами у розумінні механізму отримання винагород.

Середні оцінки за шкалою Лайкерта вказують на загалом позитивне сприйняття гри цільовою аудиторією, особливо щодо її візуальної складової та рівня залученості. Проте, оцінка зрозумілості інструкцій свідчить про необхідність покращення навчального процесу та представлення деяких механік. Розподіл оцінок рівня залученості, що показаний на графіку 3.2, підтверджує загальний інтерес, але також показує наявність користувачів, які оцінили його нижче середнього.

Результати цього тестування вказують на те, що гра "Тамагочі-Планер" має потенціал зацікавити цільову аудиторію завдяки привабливій візуальній складовій та механіці догляду за віртуальним улюбленцем. Проте, для підвищення рівня залученості та повноцінного використання функціональності планера необхідно зосередитися на покращенні зрозумілості інструкцій, спрощенні інтерфейсу планера та більш наочному представленні взаємозв'язку між плануванням завдань та отриманням винагород, які мотивують як догляд за тамагочі, так і використання планера. Подальші ітерації розробки повинні враховувати ці висновки для оптимізації користувацького досвіду.

Потенційні покращення та оновлення гри "Тамагочі-Планер" можуть бути спрямовані на оптимізацію користувацького досвіду, підвищення рівня залученості цільової аудиторії та розширення функціональних можливостей. Враховуючи результати проведеного тестування, одним з пріоритетних напрямків є покращення зрозумілості інтерфейсу планера та механізму взаємодії між плануванням завдань та системою винагород. Це може бути досягнуто шляхом впровадження більш наочних візуальних елементів, інтерактивних навчальних посібників або поетапного введення функціональності планера. Додатково, розгляд можливості кастомізації інтерфейсу, зокрема розміру шрифтів, кольорової гами та анімацій, може підвищити доступність гри для осіб з особливостями нейророзвитку.

Розширення механіки догляду за тамагочі може включати введення нових потреб, стадій розвитку, унікальних рис характеру та поведінкових реакцій віртуального улюбленця, що зробить взаємодію з ним більш різноманітною та емоційно забарвленою. З метою підвищення мотивації використання планера

можна розглянути можливість надання гравцеві додаткових переваг, окрім внутрішньоігрової валюти, наприклад, вплив на настрій або стан тамагочі в залежності від успішності виконання запланованих завдань. Введення елементів соціальної взаємодії, таких як можливість обміну досягненнями або порівняння прогресу з іншими гравцями (з урахуванням анонімності та контролю над рівнем соціальної взаємодії), може сприяти підвищенню довготривалої залученості.

Подальший розвиток функціональності планера може включати розширені можливості для категоризації завдань, встановлення нагадувань, інтеграцію з реальними календарями (за згодою користувача) та візуалізацію прогресу виконання завдань у більш наочній формі, наприклад, через графіки або діаграми. Розгляд можливості адаптації складності завдань та системи винагород відповідно до індивідуальних потреб та прогресу гравця може сприяти підтримці оптимального рівня складності та мотивації.

З точки зору технічної реалізації, оптимізація продуктивності гри, покращення стабільності та розширення підтримки різних платформ будуть важливими напрямками для майбутніх оновлень. Впровадження системи збору зворотного зв'язку від користувачів безпосередньо в грі може забезпечити постійний потік інформації для подальшого вдосконалення та адаптації гри до потреб цільової аудиторії.

# ВИСНОВКИ

У цій дипломній роботі було проведено комплексне дослідження проблеми ефективного тайм-менеджменту для людей з розладами нейророзвитку, такими як СДУГ та РАС. Аналіз психологічних та когнітивних особливостей цієї групи користувачів дозволив виявити специфічні труднощі, з якими вони стикаються при плануванні та організації свого часу. Дослідження існуючих програмних рішень для тайм-менеджменту показало, що більшість універсальних інструментів не враховують потреби людей з когнітивними розладами, що робить їх використання неефективним. Водночас, спеціалізовані додатки мають різний ступінь ефективності, що залежить від таких факторів, як простота використання, адаптивність та мотиваційний аспект.

На основі проведеного аналізу було сформульовано вимоги до інклюзивного додатку для планування часу, орієнтованого на потреби цільової аудиторії. Було визначено, що такий додаток повинен поєднувати елементи гейміфікації, персоналізації, візуалізації та мати простий, інтуїтивно зрозумілий інтерфейс. У рамках роботи було розроблено концепцію мобільної гри-симулятора догляду за віртуальним вихованцем з елементами тайм-менеджменту. Технічна реалізація гри передбачала використання модульної архітектури коду, що забезпечує її масштабованість та підтримуваність.

Проведене тестування гри на вибірці користувачів з цільової аудиторії показало загалом позитивне сприйняття гри, особливо щодо її візуальної складової та рівня залученості. Водночас, було виявлено певні аспекти, які потребують покращення, зокрема, зрозумілість інтерфейсу планера та механізму взаємодії між плануванням завдань та системою винагород. Результати дослідження підтверджують перспективність використання гейміфікованих додатків для підтримки людей з когнітивними розладами в управлінні часом. Подальші дослідження можуть бути спрямовані на розробку більш адаптованих та персоналізованих рішень, а також на вивчення довготривалого впливу таких додатків на навички тайм-менеджменту та якість життя користувачів.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The Rise in Autism and ADHD. *Brain Health from Birth*. URL: https://brainhealthfrombirth.com/prevalence/.

2. Time management skills in relation to generalself-efficacy and parental sense of competencein individuals with and without cognitivedisabilities / H. R. Afsaneh et al. *Cogent Psychology*. URL: https://www.researchgate.net/publication/335242641_Time_management_skills_in_relation_to_general_self-efficacy_and_parental_sense_of_competence_in_individuals_with_and_without_cognitive_disabilities.

3. Психологія гейміфікації та її вплив на залученість користувачів. *peerdh.com*. URL: https://peerdh.com/uk/blogs/programming-insights/the-psychology-of-gamification-and-its-impact-on-user-engagement.

4. Stardew Valley on Steam. *Welcome to Steam*. URL: https://store.steampowered.com/app/413150/Stardew_Valley/.

5. Tuggle. Behance. URL: https://www.behance.net/gallery/27601867/Tuggle.

6. Кольорова палітра №4579 | *IN COLOR BALANCE. IN COLOR BALANCE | Підбір кольорів*. URL: https://incolorbalance.com/kolorova-palitra-4579/.

7. Unity Technologies. (2024). *Unity Documentation*. URL: https://docs.unity3d.com/Manual/index.html

8. American Psychiatric Association. (2022). *Diagnostic and Statistical Manual of Mental Disorders (DSM-5-TR)*. American Psychiatric Publishing.

9. World Health Organization. *International Classification of Diseases 11th Revision (ICD-11)*. URL: https://icd.who.int/

10. Shaw, P., Eckstrand, K., Sharp, W., Liu, S., Greenstein, D., Nelson, J., ... & Giedd, J. N. Attention-deficit/hyperactivity disorder is characterized by a delay in cortical maturation. *Proceedings of the National Academy of Sciences*, 108(42), 16900-16905

11. Frith, U. *Autism: Explaining the Enigma*. Blackwell Publishing.

12. The National Institute of Mental Health. (2023). *Autism Spectrum Disorder (ASD)*. URL: https://www.nimh.nih.gov/health/topics/autism-spectrum-disorder-asd

13. Макаренко, О. А. *Використання ігрових технологій у корекції психологічних особливостей дітей з СДУГ*. Науковий вісник Ужгородського національного університету. Серія: Психологія, 44, 151-155.

14. Буковська, О. І. (2021). *Психокорекція агресивної поведінки підлітків з СДУГ*. [Дисертація на здобуття наукового ступеня кандидата психологічних наук]. Київський національний університет імені Тараса Шевченка.

15. Lord, C., Elsabbagh, M., Charman, T., & Nevill, R. (2020). Annual Research Review: The autism phenotype revisited – The first two decades of autism research in the 21st century. *Journal of Child Psychology and Psychiatry*, 61(3), 265-299.

16. Cortese, S., Faraone, S. V., Adamo, N., & Rostain, A. L. (2022). Attention-deficit/hyperactivity disorder. *The Lancet*, 399(10332), 1619-1635.

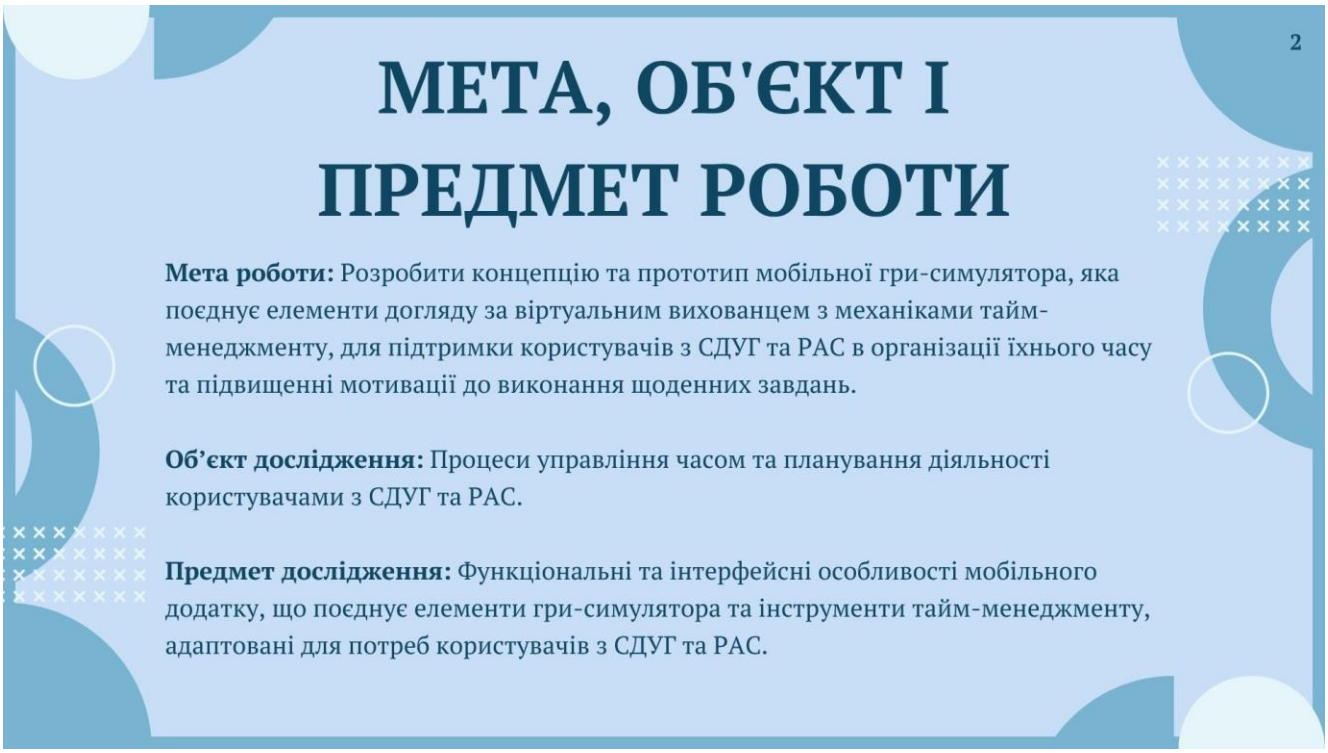
17. Lai, M. C., Kassee, C., Besney, J., Bonato, S., Dewan, S., Fraser, C., ... & Szatmari, P. (2020). Prevalence of autism spectrum disorder in the past 50 years: a systematic review. *Molecular Autism*, 11(1), 1-22.

18. Mitsea E. A Systematic Review of Serious Games in the Era of Artificial Intelligence, Immersive Technologies, the Metaverse, and Neurotechnologies: Transformation Through Meta-Skills Training.

19. Häggström, J., Tärneberg, S., & Bergström, K. (2023). Virtual Reality for Enhancing Social Skills in Individuals with Autism Spectrum Disorder: A Systematic Review. *Journal of Autism and Developmental Disorders*, 53(7), 2824-2840.

20. DuPaul, G. J., Weyandt, L. L., & Janus, D. (2022). ADHD in adults: A review of the diagnosis, epidemiology, and treatment. *Journal of Attention Disorders*, 26(10), 1709-1721.

# ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

3

# АКТУАЛЬНІСТЬ

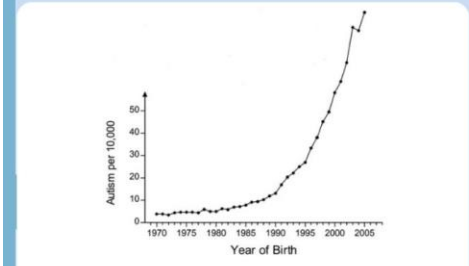



Рис. 1.1 Поширеність аутизму в Каліфорнії за роком народження. [1]

**Table 5. Differences in Time Management Skills and General Self-efficacy among participants with and without cognitive disability**

| | Participants without cognitive disability n = 147 | Participants with cognitive disability n = 86 | |
|---|---|---|---|
| **Variable** | **Mean (Sd)** | **Mean (Sd)** | **t-value** |
| ATMS-s subscales (possible range 30-120) | | | |
| Time management | 56.59 (10.89) | 40.49 (10.62) | 11.037** |
| Organization & planning | 56.32 (6.69) | 52.67 (6.19) | 4.073** |
| Regulation of emotions | 50.12 (9.64) | 43.81 (12.18) | 4.354** |
| S-GSE | | | |
| (possible range 10-40) | 30.35 (4.54) | 24.97 (5.34) | 8.311** |

**p ≤ .001

Рис. 1.2 Результати дослідження відмінності в навичках управління часом [2]

**Зростання кількості діагностованих випадків** когнітивних розладів, зокрема СДУГ та РАС (рис. 1.1), зумовлює **необхідність розробки нових підходів та інструментів** для їх ефективної адаптації.

**Ключовою проблемою** для людей з цими розладами є труднощі в управлінні часом, організації та плануванні діяльності. (рис. 1.2)

Традиційні методи тайм-менеджменту часто **неефективні, оскільки не враховують специфічні когнітивні особливості** - проблеми з концентрацією уваги, імпульсивність, труднощі з плануванням, схильність до сенсорного перевантаження).

4

# ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ

Focus@Will


Habitica


Toggl Track

5

# ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ

Порівняння трьох спеціалізованих додатків

| Показник | Focus@Will | Habitica | Toggl Track |
|---|---|---|---|
| **Простота використання** | 8/10 | 7/10 | 9/10 |
| **Адаптивність** | 6/10 | 9/10 | 7/10 |
| **Мотиваційний аспект** | 5/10 | 10/10 | 6/10 |
| **Доступність** | Платний (від $3/міс) | Безкоштовний (платні функції) | Безкоштовний (платні функції) |

7

# ВИМОГИ ДО ІНТЕРФЕЙСУ ТА ФУНКЦІОНАЛЬНИХ ОСОБЛИВОСТЕЙ

### Адаптація під сенсорну чутливість

- Можливість налаштовувати інтенсивність анімацій.
- Можливість налаштовувати гучність звуків.
- Можливість налаштовувати колірну гамму відповідно до індивідуальних потреб користувача.

### Інтегровані нагадування

- Анімаційні підказки від вихованця замість звичайних текстових.

### Високий рівень персоналізації

- Можливість вибору рівня складності планування (короткі або довгі проміжки між завданнями).
- Можливість змінювати частоту нагадувань.
- Функція кастомізації персонажа та вихованця

7

# ВИМОГИ ДО ІНТЕРФЕЙСУ ТА ФУНКЦІОНАЛЬНИХ ОСОБЛИВОСТЕЙ

**Механізми підтримки концентрації уваги**

- Режим "Якісний час із улюбленцем" з штрафами за залишення режиму до завершення таймера.

**Гейміфікація**

- Система внутрішніх нагород (віртуальна валюта, бонуси) за своєчасне виконання завдань.

**Інтеграція з реальним життям**

- Можливість синхронізації з існуючими календарями та планувальниками.

8

# РОЗРОБКА КОНЦЕПЦІЇ ІГРОВОЇ ПЛАТФОРМИ

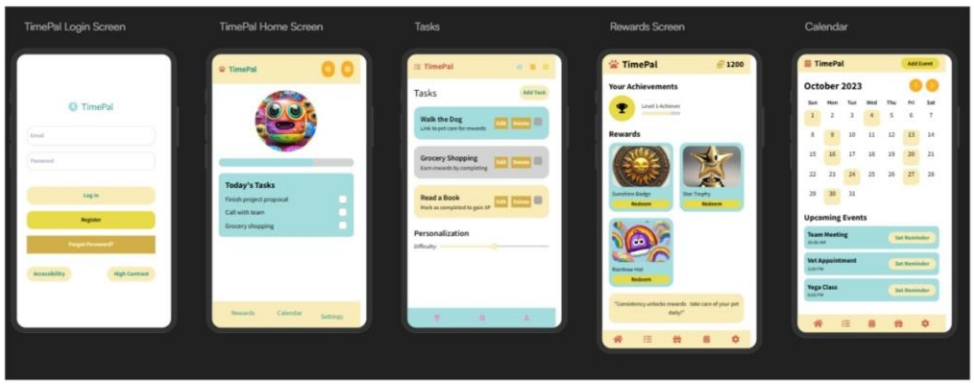
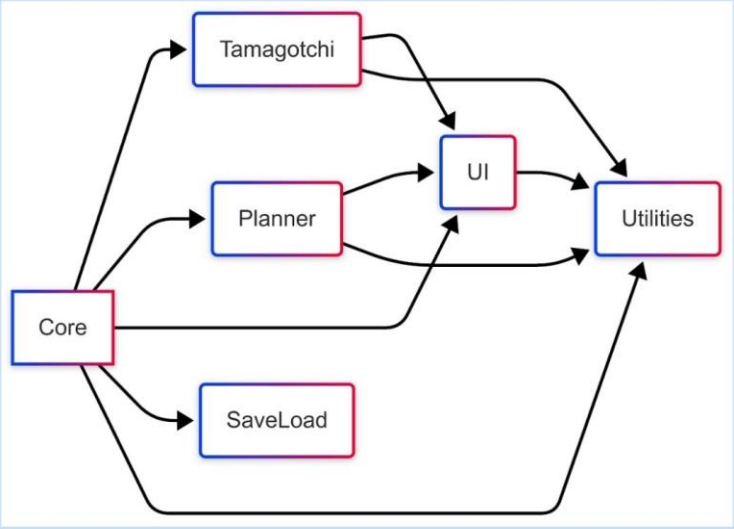



Рис. 2.3 Приклад дизайну додатку, розроблений у Figma

9

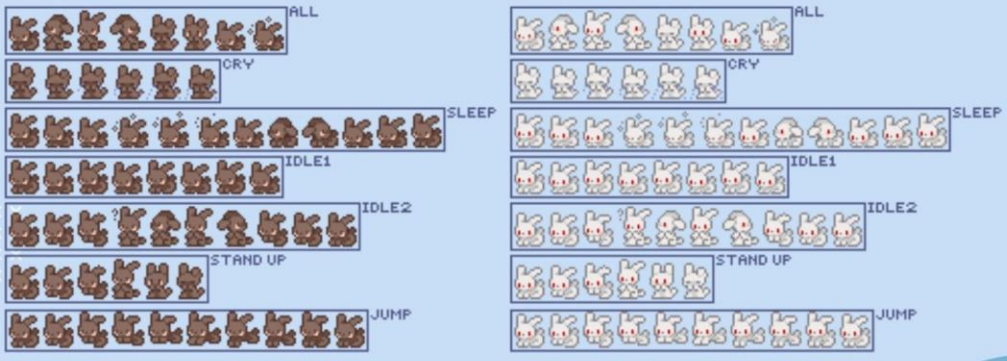
# РОЗРОБКА КОНЦЕПЦІЇ ІГРОВИХ МЕХАНІК




10

# ВІЗУАЛЬНИЙ СТИЛЬ ГРИ

Вибір **піксельної графіки як основи дизайну** зумовлений візуальною простотою та зрозумілістю, що мінімізує сенсорне перевантаження для цільової аудиторії. Крім того, цей стиль має естетичну привабливість, асоціюється з ретро-іграми, а також полегшує процес створення асетів.



11

# ПРОЄКТУВАННЯ АРХІТЕКТУРИ ТА ОГЛЯД СТРУКТУРИ ПЗ

**Основними системами гри є:**

- **Система Тамагочі**
- **Система Планера**
- **Система Інтерфейсу Користувача (UI)**
- **Система Збереження та Завантаження**
- **Система Сповіщень**
- **Система Керування Гри (Game Manager)**

```
├── Animations/
├── Audio/
├── Editor/
├── Prefabs/
├── Scenes/
├── Scripts/
│   ├── Core/
│   │   ├── GameManager.cs
│   │   ├── SaveLoadManager.cs
│   │   └── NotificationManager.cs
│   ├── Tamagotchi/
│   │   ├── Tamagotchi.cs
│   │   ├── Needs/
│   │   │   ├── Hunger.cs
│   │   │   ├── Thirst.cs
│   │   │   ├── Fun.cs
│   │   │   ├── Sleep.cs
│   │   │   └── Hygiene.cs
│   │   ├── Actions/
│   │   │   ├── FeedAction.cs
│   │   │   ├── PlayAction.cs
│   │   │   └── ...
│   │   └── Evolution/
│   │       ├── EvolutionStage.cs
│   │       └── EvolutionManager.cs
│   ├── Planner/
│   │   ├── Task.cs
│   │   ├── Event.cs
│   │   ├── CalendarView.cs
│   │   ├── TaskList.cs
│   │   └── PlannerManager.cs
│   ├── UI/
│   │   ├── TamagotchiUI.cs
│   │   ├── PlannerUI.cs
│   │   ├── TaskItemUI.cs
│   │   └── MainMenuUI.cs
│   └── Utilities/
│       └── DateTimeUtils.cs
├── Textures/
└── ...
```

12

# РЕАЛІЗАЦІЯ БАЗОВИХ ДЛЯ ТАМАГОЧІ МЕХАНІК

```
public void OnFeedButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Feed(20f);
    }
}

public void OnDrinkButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Drink(20f);
    }
}

public void OnPlayButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Play(30f);
    }
}

public void OnSleepButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.SleepAction(40f);
    }
}

public void OnCleanButton()
{
    if (tamagotchi != null)
    {
        tamagotchi.Clean(25f);
    }
}
```

```
public void Feed(float amount)
{
    hunger = Mathf.Min(100f, hunger + amount);
}

public void Drink(float amount)
{
    thirst = Mathf.Min(100f, thirst + amount);
}

public void Play(float amount)
{
    fun = Mathf.Min(100f, fun + amount);
}

public void SleepAction(float amount)
{
    energy = Mathf.Min(100f, energy + amount);
}

public void Clean(float amount)
{
    hygiene = Mathf.Min(100f, hygiene + amount);
}
```

13

# ІНТЕГРАЦІЯ ПЛАНУВАЛЬНИКА ТА СИСТЕМИ МОТИВАЦІЇ

**Система планування**

- “Task.cs”
- “PlannerManager.cs”
- “CalendarView.cs”
- “TaskItemUI.cs”
- “PlannerUI.cs”

**Система мотивації**

- “RewardManager.cs”
- “CurrencyUI.cs”

14

# ТЕСТУВАННЯ ГРИ НА ВИБІРЦІ КОРИСТУВАЧІВ З ЦА

Частота використання основних функцій гри за годину тестування (середнє значення на користувача)

| Функція гри | Середня кількість використань |
|---|---|
| Годування тамагочі | 5.2 |
| Гра з тамагочі | 3.8 |
| Створення завдання | 1.5 |
| Відмітка завдання як виконаного | 0.8 |
| Перегляд календаря | 0.5 |

15

# ТЕСТУВАННЯ ГРИ НА ВИБІРЦІ КОРИСТУВАЧІВ З ЦА

Усереднені оцінки користувацького досвіду за шкалою Лайкерта (1-5)

| Аспект користувацького досвіду | Середня оцінка | Стандартне відхилення |
|---|---|---|
| Легкість використання | 3.8 | 0.7 |
| Рівень залученості | 4.1 | 0.6 |
| Зрозумілість інструкцій | 3.5 | 0.8 |
| Візуальна привабливість | 4.5 | 0.5 |

16

# ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗРОБОК

**Оптимізація користувацького досвіду**

Покращення зрозумілості інтерфейсу планера та механізму взаємодії між плануванням завдань і винагородами, включаючи більш наочні візуальні елементи, інтерактивні навчальні посібники або поетапне введення функціональності планера

**Посилення мотивації**

Введення нових потреб, стадій розвитку, унікальних рис характеру та поведінкових реакцій віртуального улюбленця

**Розширення ігрових механік**

Введення нових потреб, стадій розвитку, унікальних рис характеру та поведінкових реакцій віртуального улюбленця

17

# ВИСНОВКИ

Ця дипломна робота **досліджує ефективність тайм-менеджменту для людей з СДУГ та РАС**, виявляючи їхні специфічні труднощі та неефективність традиційних рішень.

Було розроблено концепцію мобільної гри-симулятора з елементами тайм-менеджменту, що **поєднує гейміфікацію, персоналізацію та інтуїтивний інтерфейс** на основі модульної архітектури.

Тестування **підтвердило позитивне сприйняття гри**, хоча виявило аспекти для покращення інтерфейсу планера та системи винагород. **Результати доводять перспективність гейміфікованих додатків** для підтримки людей з когнітивними розладами у керуванні часом.

18

# АПРОБАЦІЯ РЕЗУЛЬТАТІВ РОБОТИ

Апробація дослідження здійснювалась у формі участі в **VI науково-технічній конференції «Сучасний стан та перспективи розвитку IoT»**

З поданням тези на тему **“Використання ігор в терапії нейровідмінних осіб: аналіз та розробка рекомендацій”**


ДЯКУЮ ЗА
УВАГУ!