МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему: «Проєктування інтерактивного застосунку з використанням технологій штучного інтелекту для стимулювання користувацької взаємодії»

здобувача Шеверун Нікіти Владиславовича _____
(ПІБ) *(Підпис)*

Науковий керівник:

Артемчук В.О._____

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти
завідувач кафедри:
к.е.н., доцент
_____

Київ 2025

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:
Керівник проектної групи(гарант)
освітньо-професійної програми
_____Артемчук В.О.
"____" _____2025 р.

ЗАТВЕРДЖУЮ:
Завідувач кафедри
_____
"____" _____2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувачу вищої освіти ***Шеверун Нікіті Владиславовичу***

очної (денної) форми навчання

на підготовку кваліфікаційної бакалаврської роботи
на тему: «Проєктування інтерактивного застосунку з використанням технологій штучного інтелекту для стимулювання користувацької взаємодії**»**

Тему затверджено наказом ректора Університету від « *7* » *березня 2025* р. № *466- ст*.

Кваліфікаційна бакалаврська робота виконується на матеріалах
Кваліфікаційна бакалаврська робота виконується на матеріалах отриманих під час навчання, навчальних посібників, матеріалів з мережі Інтернет, а також зібраного фактичного матеріалу з обраної теми дослідження під час виконання курсових робіт.

План кваліфікаційної бакалаврської роботи

Розділ І
Аналіз предметної галузі та сучасного стану інтерактивних систем зі штучним інтелектом. Визначення мети, об'єкта та предмета дослідження. Аналіз літературних джерел, існуючих систем і прикладів.

Розділ ІІ
Розробка вимог до інформаційної системи. Формалізація задачі, побудова алгоритмів взаємодії системи з користувачем. Моделювання поведінки та структури системи з використанням UML-діаграм.

Розділ ІІІ
Проєктування і реалізація інформаційної системи. Розробка бази даних, користувацького інтерфейсу, логіки генерації історій. Підключення API генеративного ШІ, перевірка працездатності, тестування системи.

Об'єкт дослідження
Інформаційна система, яка забезпечує різний функціонал, з якого виділяється забезпечення генерації, збереження та персоналізовану взаємодію з текстовими історіями в режимі реального часу.

Предмет дослідження
Процеси генерації динамічного контенту у вигляді історій з можливістю впливати на неї, з використанням генеративних моделей ШІ.

Мета кваліфікаційної бакалаврської роботи
Розробити інтерактивний веб-застосунок із повною реалізацією функціоналу генерації персоналізованих історій на основі ШІ та користувацької взаємодії.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I
Здійснити аналіз предметної галузі, охарактеризувати її основні особливості, а також вивчити приклади схожих систем. Приділити увагу визначенню ключових термінів, що використовуються у сфері інтерактивних застосунків із ШІ, а також факторів, які впливають на прийняття рішень у процесі розробки

У розділі
Провести розробку вимог до інформаційної системи, зокрема формулювання бізнес-вимог, користувацьких, функціональних та нефункціональних вимог. Провести класифікацію та трасування вимог для забезпечення повноти та логічної послідовності в моделюванні системи. Розробити алгоритми взаємодії користувача із системою та побудувати UML-діаграми.

У розділі III
Реалізувати проект. Необхідно розробити структуру бази даних. Реалізація серверної частини з робочими функціями. Розробити стурктуровний інтерфейс. Особливу увагу слід приділити інтеграції з API для генерації контенту. Тестуванням працездатності застосунку в різних сценаріях, перевіркою збереження даних, відповідей ШІ та загальної стабільності системи.

Завдання підготував
науковий керівник **_____Артемчук Володимир Олександрович**
« 10 » березня 2025 р.

Завдання одержав
здобувач **_____ Шеверун Нікіта Владиславович**
« 10 » березня 2025 р..

Відгук
про кваліфікаційну бакалаврську роботу
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми
«Комп'ютерні науки»

_____

на тему
«Проєктування інтерактивного застосунку з використанням технологій штучного інтелекту для стимулювання користувацької взаємодії»

1. Актуальність теми: Тема роботи має актуальність, оскільки відображає сучасні тенденції розвитку штучного інтелекту в різних сферах застосування.

2. Позитивні риси кваліфікаційної бакалаврської роботи: Робота має чітку структуру, добре аргументовану постановку задачі, глибокий аналітичний огляд, а також демонструє практичну реалізацію повнофункціонального веб-застосунку

3. Наявність самостійних розробок автора: В роботі присутня розроблена та структурована база даних, реалізовано механізми генерації та збереження генеративного контенту та створено повноцінну архітектуру клієнто-серверного веб-застосунку.

4. Цінність теоретичних висновків та практичних рекомендацій: У роботі сформульовано низку важливих висновків щодо організації інтерактивної взаємодії користувача з ШІ. Практичні рекомендації можуть бути використані для подальшого вдосконалення проекту.

5. Наявність недоліків: До недоліків віднеситься стилістичні неточності в тексті під час генерації та відсутність автоматизованої модерації згенерованого контенту.

6. Загальна оцінка кваліфікаційної бакалаврської роботи та її допущення до захисту перед ЕК: Кваліфікаційна робота відповідає вимогам до бакалаврських проєктів, відзначається актуальністю, новизною та високим рівнем самостійності. Роботу рекомендовано до захисту перед екзаменаційною комісією з оцінкою “відмінно”.

Науковий керівник

_____ Артемчук В. О.
(підпис)

“___”_____ 20__ р.

Рецензія

на кваліфікаційну бакалаврську роботу
здобувача вищої освіти
Шеверуна Нікіти Владиславовича
(прізвище, ім'я, по батькові)
Тема «Проєктування інтерактивного застосунку з використанням технологій штучного інтелекту для стимулювання користувацької взаємодії»

Актуальність теми кваліфікаційної бакалаврської роботи і доцільність її розроблення: Тема даного бакалаврського проекту є надзвичайно актуальною, оскільки автоматизація процесів взаємодії користувача з інтерактивною платформою є важливою складовою розвитку сучасних технологій. Розробка вебпроекту для генерації історій дозволяє не лише споживати контент, а й активно брати участь у створені сюжету, що збільшує залученість користувачів. Що в свою чергу буде слугувати для аналізу результатів та розробки більшого вебпроекту.

Якість проведеного дослідження: Автор продемонстрував високий рівень проведеного дослідження, здійснивши детальний аналіз потреб кінцевих користувачів і сучасних рішень на ринку. Проект включає в себе не тільки розгляд теоретичних аспектів, але й практичну реалізацію інтерактивної системи, використовуючи сучасні технології. Автор показав здатність самостійно знаходити інформацію, аналізувати її та застосовувати на практиці.

Позитивні риси кваліфікаційної бакалаврської роботи: Проект має декілька значних позитивних рис. Автор успішно поєднав теоретичні знання з практичними навичками, створивши інтуїтивно зрозумілий інтерфейс та ефективний інструмент для користувача. Використання генеративного ШІ для генерації історій є важливим аспектом, який підвищує інтерес до проекту та відкриває нові можливості для розвитку інтерфейсів та контенту.

Зауваження Незважаючи на загальну високу якість роботи, слід більше уваги приділити питанням масштабованості системи та її інтеграції з іншими програмними продуктами компанії. Деталізація цих аспектів дозволить значно покращити проект. Можливо, варто було б приділити більше уваги оптимізації процесів інтеграції з іншими сервісами та базами даних.

Практична значимість висновків і рекомендацій: Висновки та рекомендації цього проекту мають практичне значення для розробки вебзастосунків на основі штучного інтелекту. Запропонований метод персоналізації контенту може покращити залучення користувачів, зменшити витрати на створення контенту та автоматизувати взаємодію. Пропозиції щодо розширення та інтеграції роблять систему більш гнучкою та ефективною, тим самим покращуючи конкурентоспроможність підприємств на ринку

Місце роботи та посада рецензента
Науковий ступінь, учене звання (*за наявності*)
_____

(підпис, ПІБ)

Підпис засвідчую: _____

(посада, підпис)

Місце печатки організації, де працює рецензент

Анотація

випускного бакалаврського проекту студента 4 курсу

Навчально-наукового інституту «Інститут інформаційних технологій в економіці»

Шеверун Нікіти Владиславовича, виконаної на тему: «Проектування та розроблення інтерактивного застосунку з використанням технологій ШІ для стимулювання користувацької взаємодії»

Київ: кафедра інформаційних систем в економіці, 2025 р.

Випускний бакалаврський проект присвячено актуальній проблемі створення інтелектуального програмного засобу, що поєднує генеративні моделі штучного інтелекту та сучасні вебтехнології для побудови персоналізованих історій у режимі реального часу. Основною метою проекту є розробка інтерактивного застосунку, який забезпечує користувачу можливість безперервної взаємодії з історією, впливаючи на сюжет своїми діями.

Данна робота складається з трьох логічно пов’язаних розділів.

У першому охарактеризується предметна галузь, аналіз існуючих рішень, визначено мету та об’єкт дослідження, та головне, що обґрунтовано актуальність розробки системи для українського сегменту аудиторії.

Другий розділ є проектним. У ньому сформовано вимоги до системи, здійснено та обґрунтовано постановку задачі, побудовано алгоритми взаємодії системи з користувачем, а також змодельоване приблизну поведінку системи за допомогою UML-діаграм.

У третьому розділі викладено процес реалізації системи, що охоплює всі етапи розробки: від створення логіки, структури бази даних до інтерфейсу користувача. Описано інформаційне забезпечення, структури даних, принципи їх зберігання та обробки в хмарному середовищі. Розглянуто програмне забезпечення для серверної частини та клієнтського інтерфейсу.

Окремо приділено увагу підключенню ШІ. Та здійснено перевірку на коректність роботи запитів, збереження даних та роботи ШІ.

Результатом виконання бакалаврської роботи є готовий для використання веб-застосунок, що дає можливість створювати, продовжувати та зберігати історії з підтримкою української мови. Проект також має можливості для розширення, масштабування та доповнення функціоналом.

Реферат

Випускний бакалаврський проект містить **116 сторінки**, **9 таблиць**, **43 рисунків**, список літератури з **23 найменувань**, **4 додатки**.

«Проектування та розроблення інтерактивного застосунку з використанням технологій ШІ для стимулювання користувацької взаємодії»

Ключові слова: штучний інтелект (ШІ), генерація тексту, вебзастосунок, інтерактивність, користувацька взаємодія, генеративна модель, хмарний сервіс, Node.js, база даних, PostgreSQL, Gemini, історія, діаграми.

Предмет дослідження є процес генерації динамічного контенту у вигляді історій з можливістю впливати на неї, з використанням генеративних моделей ШІ.

Об'єктом дослідження виступає інформаційна система, яка забезпечує різний функціонал, з якого виділяється забезпечення генерації, збереження та персоналізовану взаємодію з текстовими історіями в режимі реального часу.

Мета випускного бакалаврського проекту полягає в створенні веб-застосунку з повним робочим функціоналом.

Завдання проекту:

- аналіз предметної галузі та сучасних рішень на ринку;
- формалізація вимог і проектування структури системи;
- реалізація генерації контенту через API генеративного ШІ;
- створення інтерфейсу для вибору жанру, сценарію та взаємодії з історією;
- побудова структури бази даних для збереження користувацьких дій;
- тестування працездатності застосунку в різних сценаріях.

Апаратні та програмні засоби, що використовувались при проектуванні: Visual Studio Code, Node.js, React, PostgreSQL, Railway, JavaScript, CSS, HTML,, API Google Gemini.

Результатом роботи став повнофункціональний веб-застосунок, який забезпечує генерацію історій за вибором користувача та створювати нові частини відповідно від прямих запитів користувача. Система дозволяє створення, зберігання, відображення послідовності дій, роботу з власним профілем та роботу запитів до ШІ. Новизна проекту полягає в підтримці української мови, інтерактивності та поєднанні гнучкої генерації з інтерфейсом.

Рік виконання випускного бакалаврського проекту: 2025

Рік захисту випускного бакалаврського проекту: 2025

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних

ДСТУ – державний стандарт України

ЕОМ – електронно-обчислювальна машина

КТЗ – комплекс технічного забезпечення

ОП – організаційне забезпечення

ПЗ – програмне забезпечення

ППЗ – прикладне програмне забезпечення

СКБД – система керування базами даних

ШІ – штучний інтелект

ІС – інформаційна система

ІЗ – інформаційне забезпечення

ВСТУП

Сучасна цифрова епоха характеризується стрімким зростанням обсягів інформації та потребою в нових формах її сприйняття, заснованих на гнучких, адаптивних і персоналізованих підходах. Розвиток штучного інтелекту (ШІ), зокрема генеративних моделей, відкрив нові горизонти для створення інтерактивного контенту, здатного не лише реагувати на дії користувача, а й динамічно змінюватися відповідно до обраного напряму взаємодії. На перетині технологій генеративного ШІ, веброзробки та гейміфікації виникає унікальний підхід до реалізації цифрового досвіду — інтерактивні історії, в яких користувач сам формує сюжет.

Інтерактивні застосунки, що реалізують подібну логіку, стають не лише засобом розваги, а й ефективним інструментом навчання, мотивації, психологічного супроводу та креативного самовираження. За останні роки популярності набули інтерактивні платформим, що поєднують генеративні моделі тексту з ігровим інтерфейсом. Проте більшість подібних систем є англомовними, мають закритий код або обмежені у персоналізації. Це створює потребу в розробці власних рішень, які відповідали б локальним потребам користувачів, підтримували українську мову, мали відкриту архітектуру та можливість масштабування.

Об’єктом дослідження в межах дипломного проекту виступає інформаційна система, орієнтована на генерацію інтерактивних історій із використанням ШІ. Предметом дослідження є моделі й технології створення таких історій, реалізація зберігання дій користувача, взаємодії з ШІ-агентом і візуального інтерфейсу для роботи із сюжетом.

Метою роботи є проектування та створення вебзастосунку, який дозволяє користувачу обрати жанр, тип сюжету, згенерувати початкову історію за допомогою генеративного ШІ, взаємодіяти з нею крок за кроком і зберігати прогрес. При цьому передбачається зручна реєстрація, система профілів, вибір персонажа, можливість продовження історій та їх повторного проходження.

Для досягнення поставленої мети були визначені наступні завдання:

- провести аналіз предметної галузі та подібних інформаційних систем;
- дослідити можливості генеративних моделей ШІ та API Google Gemini;
- розробити структуру бази даних для зберігання користувачів, історій, кроків, жанрів;
- спроектувати архітектуру вебзастосунку та реалізувати її на основі технологій Node.js, React та PostgreSQL;
- реалізувати повноцінну систему користувацького профілю з аватаром, описом, статистикою;
- забезпечити безпеку, масштабованість і зручний інтерфейс;
- провести тестування та оцінити результати реалізації.

Актуальність теми полягає в зростаючому попиті на персоналізований контент і інтелектуальні розважальні платформи, які поєднують креативність, технічну реалізацію та залучення користувача. Запропонований підхід також є прикладом практичного використання новітніх досягнень у сфері ШІ для підвищення якості взаємодії між системою та людиною.

Структура дипломної роботи включає три розділи: перший присвячений аналізу предметної області, другий — розробці вимог та моделюванню системи, третій — безпосередній реалізації програмного продукту. Завершується робота підсумками, оцінкою результатів та визначенням перспектив подальшого розвитку проекту.

РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1. Характеристика предметної галузі та об’єкта дослідження

1.1.1. Суть предметної галузі

Визначають значний вплив та виступають рушієм для більшості сфер діяльності. Ці технології використовуються на більшості етапів різних підприємств, від незначних задач до цілих відділів та систем підприємства. Щодня з’являються нові напрями розвитку інформаційних технологій, але одним з найбільш динамічних і продуктивних з напрямів є впровадження штучного інтелекту в користувацькі системи та застосунки для полегшення користування і більшої зацікавленості.

Саме тому предметна галузь даної роботи охоплює цей напрям, а саме інформаційну систему, що використовує штучний інтелект для генерації унікального контенту та адаптивної взаємодії з користувачем у реальному часі.

Для певного сегменту користувачів зацікавленість направлена на застосунки з генеративним штучним інтелектом, які автоматично створюють історії, тексти, діалоги або інший контент в залежності від запитів користувача в реальному часі, що створює персоналізований досвід спілкування з користувачем.

Такі системи реагують динамічно на запити користувача, аналізуючи контекст минулий досвід взаємодії, а не заздалегідь заплановані дії у певних ситуаціях. Що відкриває можливості для залучення нових користувачів, навчання, геїміфікації, персоналізації та творчості.

Метою більшості таких платформ є реалізація простого інтелектуального та гнучкого діалогу з користувачем, який адаптує свою поведінку та контент на основі побажань, минулих взаємодій, теперішнього завдання у реальному часі.

1.1.2. Об’єкт дослідження

Об’єктом дослідження є процес розробки інтерактивного застосунку, що дозволяє користувачу сформувати новий досвід та зануритися у динамічно створені історії, які формуються у реальному часі за допомогою алгоритмів генеративного штучного інтелекту.

Головною особливістю такого підходу можна вважати, що користувач не просто спостерігає за розвитком сюжету історії яку він вибрав, а активно та безпосередньо впливає на події за допомогою своїх рішень та дій у вигляді текстового діалогу. Що робить взаємодію максимально живою , деталізованою та індивідуальною.

Також у центр уваги потрапляє дослідження механізму побудови відповідей системи на основі запитів користувача та впровадження логічного та “розумного” впровадження подальших подій історії.

1.1.3. Приклади об’єктів предметної галузі

До об’єктів предметної галузі які демонструють схожі характеристики, принципи взаємодії та риси з даним проектом, належать системи які поєднують штучний інтелект, взаємодію з користувачами та генерацію контенту. Найбільш поширеними на сьогоднішній день такі рішення зустрічаються у вигляді чат-ботів, віртуальних помічників, інтерактивні ігрові застосунки, генератори відео-контенту, адаптивні навчальні платформи. Та використовуються в галузях освіти, розваг, комунікації та комерції.

Одним із відомих англомовних прикладів застосування схожих технологій це AI DANGEON - це сайт на якому користувачі створюють власні сценарії на основі своїх думок та допомогою штучного інтелекту та викладають для подальшого читання іншими користувачами. Також подібне реалізовано у проектах схожих на Replica (віртуальний помічник) або схожу ідею можна зустріти у комп’ютерних іграх типу: Detroit Become Human.

Однак більшість існуючих систем дуже складні у використанні, або не реалізовують усю адаптивну логіку сценаріїв, що враховувала б невизначеність і мінливість поведінки користувача під час використання.

1.1.4. Прийняття рішень у предметній галузі

Прийняття рішень стосовно предметної галузі, пов’язаної з розробленням інтерактивного застосунку з використання штучного інтелекту, є складним і багаторівневим. Прийняття рішень охоплює технічні, організаційні, системні аспекти які залежать від різних факторів.

Основні рішення які стосуються цієї галузі, це:

- Вибір архітектури системи - необхідно обрати структуру проекту, як він буде реалізован, та як впливає на масштабованість, продуктивність, зручність для використання та зрозумілість структури. В нашому випадку це класичний клієнт-серверний застосунок.
- Вибір та інтеграція штучного інтелекту - основною задачею є вирішення яку модель ШІ для генерування контенту використовувати відносно їх плюсів та мінусів. Також визначити яким способом підключати API та формат отримування відповіді та запитів. А також обробка результатів.
- Проектування логіки взаємодії - є важливою частиною, адже визначається як буде взаємодіяти користувач з системою, через інтерфейс чи застосунок. Також важливо вирішити як буде реагувати штучний інтелект на дії користувача.
- Побудова інтерфейсу - під час створення інтерфейсу повинно враховуватися кольорова гамма, стиль сторінок, об’єм інформації, структура сторінок, зручність використання, навігація та доступність. Всі ці фактори впливають на зручність та ефективність використання користувачем. Що в свою чергу зацікавить користувача.
- Захист персональних даних та безпека - питання полягає у вирішені яку систему для зберігання використати, яке шифрування використовувати, як правильно створювати запити. Перевірку варіантів і використання найкращого.
- Оптимізація генерації контенту - розробнику потрібно вирішити як оптимізувати запити до ШІ: як правильно сформулювати промти (promt ingenering), яку максимальну довжину відповіді отримувати, в якому форматі відповідь, формат тексту на вивід, який рівень креативності

використовувати. Також потрібно організувати контроль результатів, що б уникати не бажаного контенту. Використання певних стилей під кожну історію, взаємодію з діями користувача, наскільки впливатимуть в історію. Використання різних історій та розгалужень.

1.1.5. Фактори впливу на прийняття рішень

На кожному етапі розробки проекту виникає низка питань на які треба приймати рішення, які визначають функціонал, швидкість, ефективність і зручність системи. Усі рішення приймаються під тиском різних факторів які можемо класифікувати таким чином:

- Технологічні фактори - пов’язані з технічними можливостями реалізації системи (Продуктивність обраної моделі, кількість запитів, швидкість відповіді, сумісністність програмних засобів, тощо)
- Користувацькі очікування - відповідність системи до очікувань користувача і зручності використання (Зручність інтерфейсу, швидкість реакції системи, зрозумілість відповідей ШІ)
- Інформаційні фактори - потрібен обсяг достовірної інформації, тестування системи у реальних ситуаціях та сценаріях.
- Економічні обмеження - на вибір технологій, моделей, можливостей використання впливають економічні фактори, наприклад : вартість використання ШІ, витрати на хостинг та базу даних, рентабельність проекту.
- Конкурентне середовище - значною мірою є впливає аналіз конкурентів та їх рішень, а саме присутність конкурентів, унікальність функціоналу, ціна використовування, демо-версії, відгуки та тренди на сьогоднішній день.

1.1.6. Інформація для прийняття рішень

Розробка проекту повинна включати різні види інформації, особливо актуальну, достовірну, обґрунтовану та різнопланову. Від якості такої інформації залежить зручність та працездатність системи, також адаптивність і відповідність потребам користувача. Класифікуємо цю іноформацію на внутрішню та зовнішню.

Внутрішня інформація - це дані які генерує сама система під час взаємодії з користувачем в процесі розробки та роботи проекту. Найчастіше це: логи користувацької активності (найчастіші жанри та типи історії, скільки часу проведено, тощо.), аналітика взаємодії користувача, запити до ШІ та їх результати, патерни відповіді, виниклі проблеми. Це все дозволить покращити та оптимізувати проект.

Зовнішня інформація - інформація поза межами системи, але впливає на рішення. До такої інформації відноситься документація до мовних моделей, наукові публікації, сайти з відкритим обговоренням, відгуки користувачів, опитування , обмеження у використанні технічних засобів.

Також класифікуємо інформацію за типом та характером.

Емпірична система - дані отримані з використання системи.

Об’єктивна - статистична та аналітична складова, зібрана автоматично.

Суб’єктивна - думка користувачів та інших спеціалістів.

Направлена - зібрана інформація від певного жанру, типу або вибраної взаємодії.

Для обробки інформації буде застосовано імітаційне моделювання, на основі штучно створених умов. Алгоритмізацію та аналітичну методологію, а саме побудову чітких логічних алгоритмів для оброки інформації, побудову моделей, графіків і залежностей. Що забезпечить глибоке розуміння потреб до проекту.

Типи ситуацій прийняття рішень:

У процесі розробки виникають різні ситуації, які класифікуються за характером та ступнем проблеми. Їх можна поділити на закриті задачі, де вже існує вся інформація і алгоритм дій, та відкрити, що допускають різні варіанти вирішення однієї проблеми. Також є кризові ситуації де рішення потребується оперативно, часто без повного списку інформації, лише частина.

В нашому випадку детермінована ситуація, де всі параметри відомі і є побудована логіка - це формування інтерфейсу з певними жанрами та їх типами. Ситуація за умов ризику - це вибір мовної моделі gpt4- яка дорожча але точніша, або gemini - безкоштовна але простіше генерує історії. Або ситуація за умов невизначеності - розробка нової механіки яка не тестувалася з реальним користувачем і неможливо прогнозувати реакцію.

1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

1.2.1. Стан досліджень у галузі інтерактивних ШІ-систем

Основною частиною проекту є штучний інтелект, а отже треба аналіз інформації відносно його.

Все більше і більше у новітніх дослідженнях приділяють увагу застосуванню штучного інтелекту у проектах з активною участю користувачів у реальному часі. Дедалі частіше такі застосунки розглядаються як покращення вже існуючих проектів, так і перспективний напрям усієї галузі ІТ технологій, зокрема в освітній, творчій і комунікаційних сферах застосування.

Серед документів та статей, які формують бачення розвитку Ші в Україні, варто відзначити **"Стратегію розвитку штучного інтелекту в Україні (2023–2030)"**, розроблену Інститутом проблем ШІ НАН України. Документ визначає пріоритетні напрямки впровадження ШІ в державне управління, цифрову трансформацію освіти, медицини та бізнес-налаштування. Стратегія наголошує на інтеграції інтелектуальних систем у

повсякденне життя громадян для створення умов для національних технологічних проривів. [16]

На світовому рівні, з даних отриманих після дослідження компанії **McKinsey & Company**, різні види штучного інтелекту вже на сьогоднішній день має потенціал додати від 2.7 до 4.4 трильйонів доларів до світової економіки. Насамперед генеративні моделі які вміють створювати текст, зображення, відео, програмний код або музики, стає основою інновацій. Загалом це стосується таких сфер, як: клієнто-орієнтовний сервіс, розробка програмного забезпечення, маркетинг, дослідження та розважальна сфера. [17]

Наступне дослідження **"Від пояснюваного до інтерактивного ШІ"** (arXiv), у науковому контексті є важливим для розуміння аналізу еволюзії Штучного інтелекту від традиційних рішень до активної взаємодії з користувачем. Де автори зазначають важливу взаємодію у гібридних системах, користувача з алгоритмами як партнерів для кращого результату. [18]

Від недавна з’явився новий інструмент **Google Stitch,** який чудово демонструє практичне застосування і роботу генеративного штучного інтелекту в автоматичному створені інтерфейсів по запиту користувача. Замість звичного кодування інтерфейсу, розробник вводить текстову інструкція по якій буде згенеровано UI прототип\макет. Що значно спрощує роботу й дозволяє швидко створювати прототипи. [19]

У сфері розваг передовим та яскравим прикладом є проект **AI Dungeon,** який надихнув мене на створення української версії схожого проекту. Він реалізує генерацію сюжетних ліній з прямим втручанням користувача у реальному часі. Користувач активно впливає на історію, а ШІ реагує у стилі живого співрозмовника. Це вдалий приклад взаємодії з ші у цій сфері. [20]

Україна не відстає від іншого світу, та має власні успішні розробки. Grammarly - один із найвідоміших стартапів, який використовує ШІ для обробки природної мови, знаходження та виправлення помилок у текстах, покращення стиля та створення рекомендацій. Також чудовий приклад це -

Reface, яка працює з відеоконтентом та використовує Ші для обробки та аналізу зображень у реальному часі. [21]

1.2.2. Практичний досвід розробки та впровадження

Вже існуючі практичні розробки з використанням штучного інтелекту свідчить про наявні успішні рішення які використовуються в різних галузях - від освіти до розваг. Чудовим прикладом і головним постачальником ідеї мого проекту є AI Dangeon - англомовна платформа, яка дозволяє створювати власні сценраії у реальному часі за допомогою генеративної моделі ChatGPT. За допомогою вибору теми сценарію, та постійної прямої взаємодії користувача з сценарієм у текстовому форматі, що впливає на перебіг подій, в той час ШІ адаптується та продовжує генерувати сценарій. Це забезпечує гнучкість, варіативність та надає ефект живого сценарію. Основні слабі сторони проекту які зустрічаються, це втрата логічного зв'язку у сценаріях під час затяжних сесій, періодичні повторення сцен, неможливість контролювання генерованого тексту без фільтрів, та мінімальна підтримка та взаємодія з мовами, окрім англійської.

Вже інший український стартап Grammarly, що спеціалізується на повноцінному аналізі тексту у реальному часі. Основний задача та функціонал цього проекту полягає у використанні глибоких нейромережевих моделях обробки природньої мови, які виявляють помилки в тексті, покращують стиль написання та пропонують варіанти схожого формулювання тексту. Метою проекту вважається лінгвістичне вдосконалення тексту, а принцип інтерактивності реалізується через постійний діалог користувача і системи під час написання. Дуже важливо зазначити що Grammarly також адаптована під український текст, що робить його одним з числа проектів який адаптований під український ринок.

Також варто зазначити що є інші системи та платформи - Character.AI, Replika, Google Stitch, Reface тощо, що також продемонструють значний прогрес у використанні генеративного ШІ для побудови взаємодії під користувача.

Разом з позитивним досвідом, з'являється ряд викликів. Звідки з'являються такі проблеми:

- Величезна вартість у використанні з потужними мовними моделями.
- Складність використання готової моделі під українську мову та контекст.
- Складність та необхідність контролю над вмістом генерації тексту.
- Підтримка логіки та контексту під час довгих без перервних сесій.

Відштовхуючись від цієї інформації можна засвідчити існування успішної реалізації проектів у сфері генеративного ШІ. Проте зрозуміло що є необхідність розв'язання різних задач на технічному і концептуальному рівнях. Ці фактори впливали на визначення актуальності теми та постанові задач, пов'язаного з розробкою власного проекту на базі ШІ.

1.2.3. Ключові поняття та критичний аналіз підходів

У межах теоретичного та практичного дослідження цієї роботи важливо окреслити ключові поняття, які сформують основу для розуміння створення інтерактивних систем. Насамперед, це поняття **інтерактивність**, **персоналізація**, **штучний інтелект**, **генеративні моделі**, а також **користувацький досвід (UX)**. Розуміння цих термінів дозволить розібратися та сформувати думку та підходи до реалізації функціонального застосунку.

Інтерактивність у нашому випадку - це можливість системи не лише реагувати на дії користувача, а адаптуватися під поточні умови, вибрані характеристики, історію взаємодій та вподобання певного користувача. Це постійна взаємодія з ефектом співпраці з системою, а не просто діалог "запит-відповідь".

Персоналізація - це здатність системи запам'ятовувати індивідуально запити користувача, стиль його спілкування, уподобання та минулі дії. Існує основні два варіанта реалізації на рівні локального збереження, так і навчання моделі на діях користувачів.

Основним і головним елементом системи є штучний інтелект, а саме його генеративна здатність, що дозволяє створювати контент у реальному часі. Генеративні моделі як ChatGPT, Gemini, Cloude та інші, здатні генерувати текст на основі запиту з контекстом, непогано імітувати стиль написання, емоції та логіку. Незважаючи на добре згенеровані тексти, ці моделі і не мають

“усвідомлення”, та продовжують генерація як доречними так і недоречними відповідями. Що знову вказує на потрібність контролювання контенту та чітких установлених правил для генерації.

З сторони користувацької досвіду **(UX)**, інтерактивні Ші-системи повинні створювати враження реального співрозмовника або повноцінної гри зі змістом, де відразу видно що кожна дія має свій наслідок. Основною часткою негативного досвіду, займає незрозумілі, або неправильні відповіді штучного інтелекту, або ситуації коли він забуває що відбувалося раніше. Також важливо забезпечити зрозумілу структуру та плавну взаємодію системи.

Основні стратегії які дозволяє виділити критичний огляд підходів у існуючих системах :

Перше це вільна генерація як у AI Dangeon, де зустрічається дві позиції, користувач має повну свободу, а модель генерує без обмежень на відповідь користувача. Що забезпечує проекту максимальну гнучкість, але водночас більше ситуацій де втрачається сенс історії, логіка і проблеми з етнічності вмісту.

В другому випадку це структурована генерація з контролем контексту, складна модель, як у Grammarly, де генерація відбувається у чітко заданих рамках. Це знижує ризик, але падає варіативність відповіді.

Тому при розробці власного проекту слід комбінувати ці підходи: а саме забезпечити вільну генерацію у межах жанру та сценарію, але за можливості правильно реалізувати фільтрування вмісту відповіді, перевірку на тригери, та збереження логіки через контекст. Проект має створювати ілюзію реальної історії та взаємодії.

1.2.4. Інформаційне, програмне та технічне забезпечення

Розробка проекту з використанням генеративного ШІ вимагає комплексного підходу до вибору та організації всіх видів підтримки: інформаційної, програмної та технологічної. Правильна структура

компонентів забезпечує ефективну взаємодію між користувачами, обчислювальною логікою, зовнішніми службами та базами даних.

- Інформаційне забезпечення включає дані які використовуються в системі, структуру збереження та повернення даних. У контексті проекту це:
- Вхідна інформація від користувачів, вибраний жанр, тип історії, ім’я персонажу.
- Вихідна інформація від ШІ, початкова історія, кожна наступна частина від пов’язана з діями користувача, варіанти подій.
- Допоміжні дані, усі жанри, усі підкласи жанрів, налаштування гравця, минулі взаємодії.
- Збережені історії та всі кроки користувача за минулі сеанси.

Всі вище перечисленні данні зберігаються у базі даних, в нашому випадку це PostgreSQL, яка структурована під проект: таблиці користувачів, їх налаштувань, історій, кроків користувача, жанрів та інших. Також не забуваємо про потребу масштабування, наприклад додавання нових жанрів або підкласів без зміни структури.

Програмне забезпечення реалізується з урахуванням власних побажань та досвіду роботи з різними архітектурами. У проекті це:

Клієнтська частина (Frontend) - реалізована з використанням різних мов програмування, Html, Css, JavaScript. Та відповідає за візуальний контент, відображення інтерфейсу, зібрання інформації від користувача, вивід згенерованих текстів, вибір жанрів та типів історії, та збереження стану.

Серверна частина (Backend) - реалізована на використанні Node.js та Express.js, що обробляє запити користувача, керує логіку аунтифікація та авторизації, виконує збереження даних у базу даних, забезпечує захист маршрутів та передає промти до генеративної моделі штучного інтелекту та отримує відповідь.

Штучний інтелект - проект інтегровано з Google Gemini API, що надається через офіційний сайт. Основний промт створено для загальної структури, але відбувається динамічне форматування на основі дій

користувача. Після отримання відповіді, користувач отримує інформацію на ігровій сторінці, також відбувається збереження базі даних.

База даних - використовується PostgreSQL, розгорнута за допомогою Railway. Де зберігається вся потрібна інформація та зв’язки між нею.

Система загалом забезпечує безпеку до даних користувача, та можливість розширення.

1.2.5. Інформаційні потоки, структура даних і користувачі

Ефективне функціонування інтерактивного проекту забезпечується правильною реалізацією потоків інформації, логічною та простою реалізацією структурі даних і чіткими правилами та дозволами користувачів. Ці етапи мають значний вплив на етапі проектування, та вдосконалення проекту в подальшому.

Інформаційні потоки охоплюють усі аспекти та цикли взаємодії користувача з системою, від самих початкових, вибору жанру, типу історії, до обробки запитів системою (відповідь ШІ, збереження даних, тощо) та виведення інформації до інтерфейсу.

Умовний поділ потоків такий:

- Вхідні дані - дії користувача, вибір, текст, налаштування.
- Внутрішні обробки - побудова запиту до ШІ, збереження та оновлення, прогресу, форматування тексту, правильна обробка.
- Вихідні дані - генерація тексту, відповідь мовної моделі.

Інформація передається за допомогою HTTP(S)-запитів між фронтендом і бекендом, обробляється у вигляді JSON-структур і зберігається у хмарній базі даних (PostgreSQL).

Базові сутності які включає система:

- users - користувачі, що взаємодіють із системою;
- profiles - додаткові атрибути: никнем, фото, обрані жанри;
- stories - основні сюжетні гілки (жанр, назва, вступ);

- steps - послідовні кроки користувача в межах однієї історії;
- genres і story_types - класифікатори, що дозволяють фільтрувати та організовувати контент;

Інформація виникає під час кожного кроку і має свою періодичність, що залежить від завантаженості та використання проекту. Деякі види інформації: введення запиту до ШІ, виникає у реальному часі під час вибору дії користувача. Збереження кроку виникає після результату генерації. Оновлення профілю виникаю за бажання користувача.

Отримані потоки інформації у системі побудовані логічно і просто для розуміння та зі збереженням цілісності інформації. Що в свою чергу забезпечує стабільну і безперервну взаємодію з користувачем.

1.2.6. Формулювання задачі, мети та обґрунтування важливості роботи

На основі усіх вище написаних пунктів, було сформульовану мету дипломного проекту: розробка інтерактивного веб-застосунку який на основі штучного інтелекту, дає змогу користувачу будувати власні історії з постійною взаємодією та впливом на неї, що генерується у реальному часі.

Проект повинен поєднувати базові потреби, як гнучкість системи, простий та зрозумілий інтерфейс, просту структуру даних, можливість постійного функціонального розширення, підтримки української мови. Користувач повинен мати можливість редагування профілю, налаштувань, вибору історії , її типу та імені свого персонажа. Також мати вплив подій, продовжувати історію від своїх побажань та зберігати її.

Актуальність проекту залежить від різних чинників. Головні чинники:

- Зростання зацікавленості та бажання спробувати цифрові продукти з штучним інтелектом у сфері розваг.
- Поширення мовних та генеративних моделей штучного інтелекту та потреби адаптації під український ринок.
- Недостатньою кількістю інтерактивних застосунків які поєднують зручність та цікавість користувачу.

- Можливість застосування проекту у різних середовищах як гнучкий інструмент генерації.

Проблемна ситуація з якою зіштовхується більшість користувачів, і вирішує проект, це - відсутність простого, зрозумілого й водночас технологічного інструменту для генерації історій та взаємодії з ними українською мовою та базується на роботі х штучним інтелектом.

На основі цих критеріїв сформульовані задачі для проекту:

1. Аналіз предметної галузі і існуючих рішень.
2. Приблизне проектування архітектури застосунку.
3. Реалізація повної інтеграції з штучним інтелектом для працездатності проекту через API.
4. Розроблення логіки збереження усіх історій та дій користувача.
5. Створення користувацького інтерфейсу.
6. Тестування системи на різні фактори.

Результатом усіх дій, буде отримано веб-застосунок з потенціалом для подальшого розширення, адаптації та інтеграції з іншими сервісами.

1.3 Висновок :

У першому розділі було описано всебічну характеристику предметної галузі, доцільно вивчено об’єкт дослідження та проаналізовано сучасні тенденції у сфері систем, що використовують генеративні моделі ШІ. Також доведено що колючовим рушієм інновацій є генеративний ШІ у сфері цифрових технологій.

Проведено дослідження, що підтвердило актуальність створення інтерактивного веб-застосунку, що дозволяє користувачу зануритися в історії в яких він головний герой та може керувати подіями, створити власний світ. Виявлені проблеми аналогів та їх обмеження у довготривалих сесіях.

Також проведено огляд літературних джерел, публікацій та існуючих прикладів використання ШІ як в Україні, так і за її межами. Що дозволило оцінити існуючи підходи використання, та визначити слабкі та сильні сторони.

Важливою частиною аналізу стало побудова інформаційної моделі проекту, опис потоків даних, ролей, архітектури та інформаційного забезпечення.

Отже, результат аналізу визначив доцільність і практичну значущість в реалізації бакалаврського проекту. Та створення приблизного підходу що дозволяє створити сучасний, адаптивний і масштабований веб-застосунок, який взаємодіє з ШІ та матиме переваги в користувацькому досвіді.

РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ/ПІДСИСТЕМИ

2.1. Аналіз і специфікація вимог до інформаційної системи/підсистеми

2.1.1. Бізнес-вимоги та загальні цілі системи

Все частіше на просторах інтернету з’являються застосунки, що включають взаємодію з штучним інтелектом, і активно впроваджуються у сфери освіти, творчості, психологічної підтримки. Найчастіше зустрічається у сфері розваг. Зважаючи на стрімке зростання популярності штучного інтелекту, виникає потреба у створенні застосунку який повноцінно підтримуватиме українську мову, який дозволить користувачеві зануритися у власну історію яку генеруватиме штучний інтелект.

Бізнес-вимоги в контексті дипломного проекту визначаються на основі аналізу українського ринку, приблизних очікувань та побажань користувачів і можливості вільного доступу до генеративних моделей ШІ.

Головною місією проекту є надання персонального досвіду користувачеві у створенні інтерактивної історі, де сюжет будується на вибраному жанрі та реакції на дії користувача по сюжету. Що дозволить протестувати новий формат взаємодії з використанням ШІ та української мови.

Основні цілі системи:

- Створити гнучку масштабовану платформу з інтуїтивно зрозумілим інтерфейсом для інтерактивного сторітейлінгу.
- Підвищення залученості користувача.
- Реалізувати підтримку різних жанрів та типів сюжетів.
- Забезпечити збереження історій та можливість їх продовження.
- Інтегрувати ШІ для генерації історій.
- Забезпечити безпеку даних.
- Адаптація сценарію до вибору користувача.

Бізнес-вимоги та цілі більш детально представлено на рисунку 2.1 та додатку А.

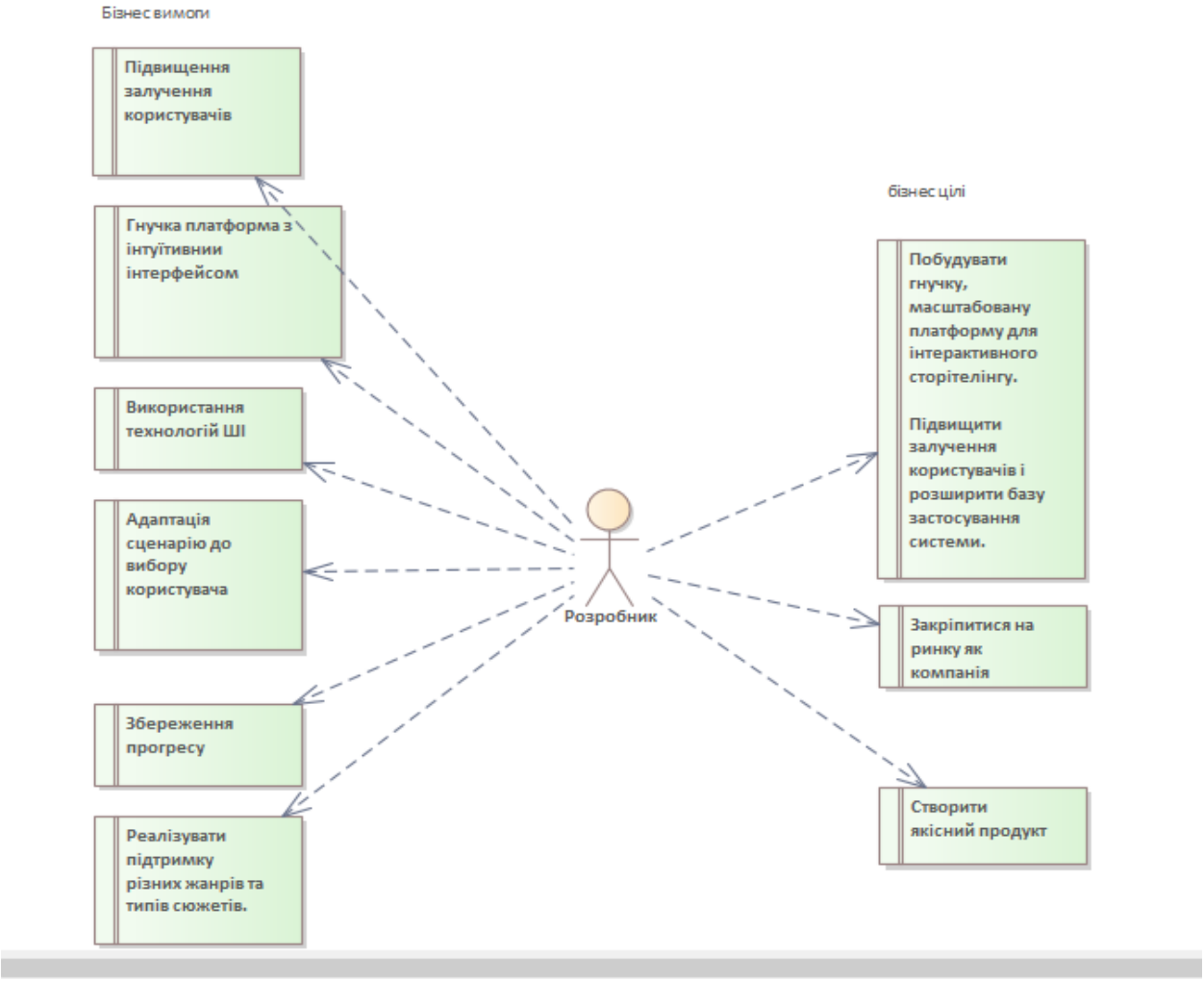


Рисунок 2.1 – Діаграма бізнес-вимог та цілей

Джерело: сформовано автором на основі виконаного дослідження

З огляду на тенденції цифорвізації, система має значний потенціал у подальшому вдосконалені та використані у суміжних сферах – у створені і вдосконалені ігрових процесів, написанні сюжетів для книжок або окремих сцен, використанні для генерації ідей, або відтворені існуючих подій або симуляцій.

Отже, бізнес-вимоги формулюють бачення цінності системи для користувача та показують важливі завдання які повинні взятись до уваги на етапі проектування та розробки проекту.

2.1.2. Користувацькі вимоги

Визначенням очікувань і потребами кінцевого користувача у системі визначаються користувацькими вимогами. Вони визначають та описують функціонал проекту на який розраховує користувач зі своєї сторони та є основною формування функціональних та нефункціональних вимог.

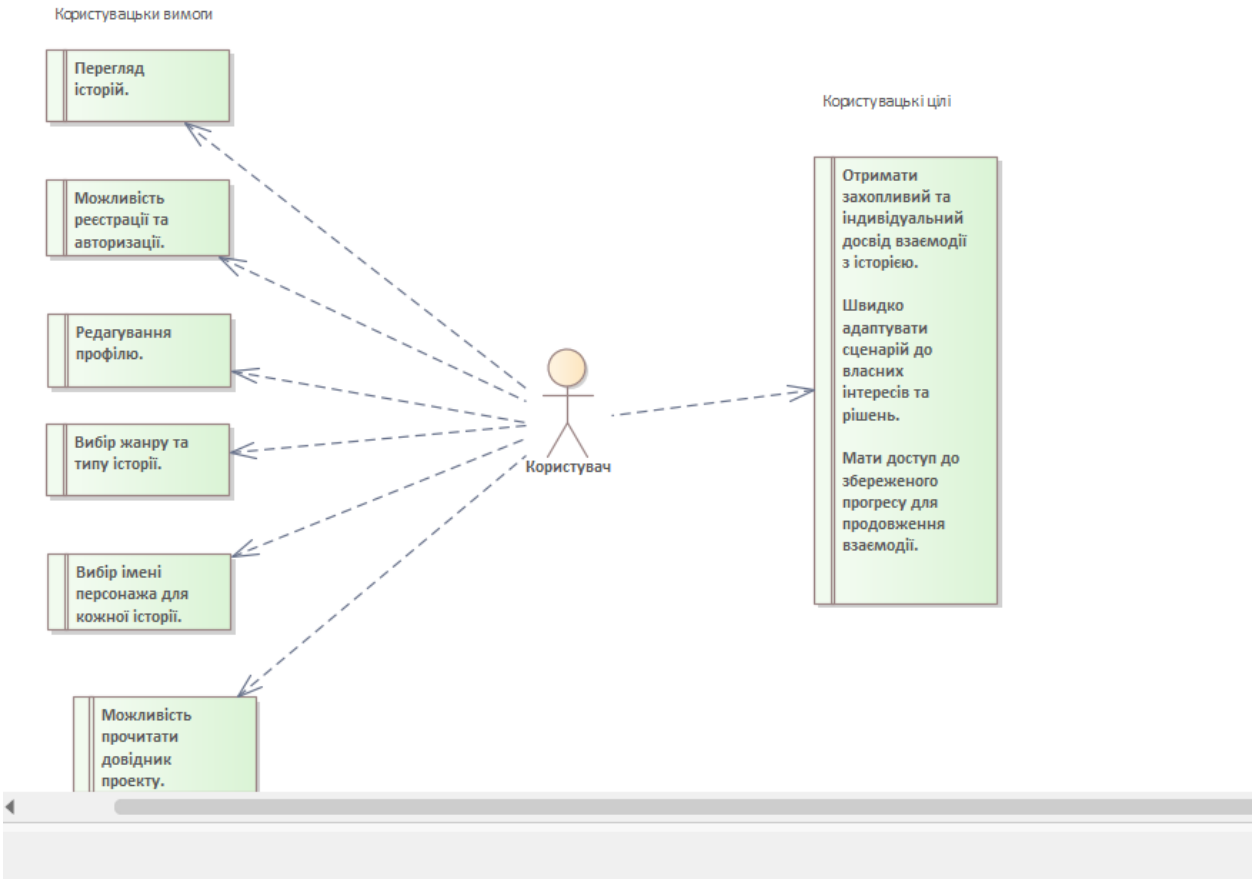
В проекті передбачено лише основний тип користувача, а саме звичайний користувач або гравець, що взаємодіє з системою, створює свої історії, вводять данні. Також присутній розробник який слідкує за проектом та може додавати нові жанри, типи історій і виправляє проблеми проекту.

Звідси на основі очікуваного сценарію формується ключові користувацькі вимоги та цілі.

Для гравця – звичайного користувача:

- Можливість реєстрації та авторизації.
- Вибір типу та жанру історії.
- Пряма взаємодія з історією через інтерфейс.
- Збереження кожної історії та дій користувача.
- Перегляд минулих історій та можливість їх продовжити.
- Редагування профілю.
- Адаптація історії під дії користувача.
- Отримати захопливий та індивідуальний досвід взаємодії з історією.

Кожна вимога є загальною та буде деталізована у подальших розділах, що дозволить здійснити точнішу реалізацію та перевірку завдання. Ознайомитися з існуючими вимогами користувача на рисунку 2.2 та додатку А.



Джерело: сформовано автором на основі виконаного дослідження

Рисунок 2.2 – Користувацькі вимоги.

2.1.3. Функціональні вимоги

Функціональні вимоги визначають, що саме дії системи буде виконувати для досягнення цілей які були поставлені. Вони формуються на основі раніше вказаних користувацьких вимог. Також описується основні функції, які система повинна реалізувати для чіткої взаємодії системи котра використовує штучний інтелект та користувачем.

Основна увага приділяється процесу генерації історії, адаптації історії та змін на основі дій користувача та оновлення історій в реальному часі. Загальні функціональні вимоги представлено на рисунку 2.3 та додатку А.

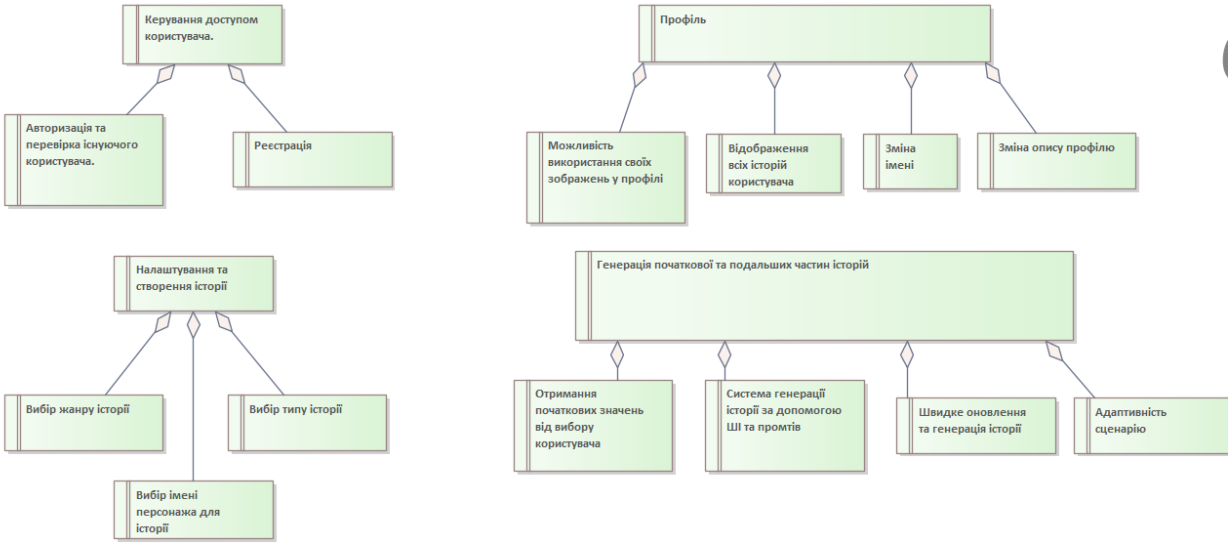


Рисунок 2.3 - Функціональні вимоги системи.

Джерело: сформовано автором на основі виконаного дослідження

Розроблена діаграма побудована на основі важливості реалізації таких функціональних вимог:

- Керування доступом користувача для взаємодії з системою.
- Налаштування та створення початкової історії.
- Можливості Профілю.
- Генерація початкової історії та подальшої генерації частини історії після дії користувача.

2.1.4. Нефункціональні вимоги

Вимоги що відображають технічні, якісні та експлуатаційні характеристики системи, що повинні будуть забезпечені для доцільної роботи проекту, це – нефункціональні вимоги. Отже вимоги визначають якісь функціонування системи, а не її поведінку. У контексті цього дипломного проекту ці вимоги мають особливу увагу, від них залежить рівень задоволеності кінцевого користувача, стабільності та можливості розширення. На рисунку 2.4 подано ключові нефункціональні вимоги системи та додатку А.

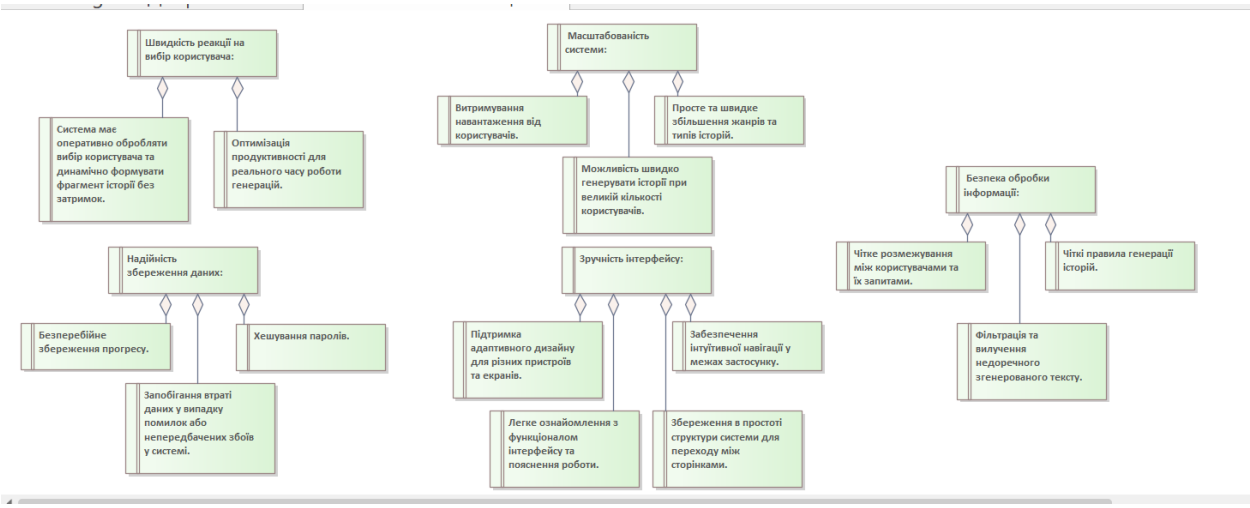


Рисунок 2.4 – Нефункціональні вимоги

Джерело: сформовано автором на основі виконаного дослідження

Розроблена діаграма побудована на основі важливості реалізації таких нефункціональних вимог:

- Швидкість реакції на дії користувача.
- Надійність збереження даних.
- Масштабованість системи.
- Зручність інтерфейсу.
- Безпека обробки інформації.

Виконання цих вимог є критично важливим для існування проекту, впливає на ефективність та прийнятність проекту від користувача. Отже нефункціональні вимоги, повинні чітко враховуватися під час моделювання, тестування та розробки проекту, так само, як функціональні вимоги.

2.1.5. Обмеження та зовнішні умови

Під час проектування та реалізації загальної інформаційної системи необхідно враховувати багато факторів, особливо обмеження та зовнішні умови, що впливають на роботу, ефективність і масштабність системи. Саме вони визначають рамки системи в яких вона буде працювати. Також не забуваємо про зовнішні чинники, що залежать від розробника проекту та мають певний вплив на процес прийняття рішень.

Технічні обмеження з якими зустрінетеся проект:

Обмеження API: значна проблема з використанням генеративного ШІ, це цінна за повноцінні та добре збалансовані моделі, вони також передбачають ліміти на кількість запитів, довжину відповіді та швидкість генерації контенту. Додатковою проблемою виступає важкість знайти безкоштовний тариф для ШІ, на який також діє більш жорсткі обмеження на кількість запитів та обсяг контенту.

Підключення до зовнішнього сервера з базою даних: в свою чергу, це система працює з хмарною базою даних, наприклад Railway, що обмежує прямий доступ до даних без підключення та налаштування авторизаційних IP. Також має значну ціну за використання такої бази даних у великих масштабах.

Обмеження хостингу: безкоштовні платформи є гарним варіантом тільки на початковому рівні, адже мають ліміти на RAM, CPU, та обсяг обробки даних, що вливає на кількість можливих користувачів одночасної активності у системі та призводить до проблем.

Організаційні обмеження проекту:

Використання лише публічних генеративних моделей штучного інтелекту: через важкість створення і навчання власної мовної моделі, система використовує наявні комерційні рішення, що визначає залежність від сторонніх сервісів та їх моделей.

Проблематика бюджету: важливий пункт, адже зумовлює використання і пошуку безкоштовних інструментів, відкритих API або лімітованих аккаунтів.

Ресурс витраченого часу: проект виконується тільки обмежений час, що значно впливає на масштабність виконання роботи, масштабу, функціоналу та виникнення помилок.

Зовнішні фактори:

Правові обмеження: використання кожного сервісу що надає доступ до ШІ, має власну політику використання якої треба дотримуватись. Також важливо відповідати політиці та умовам провайдера і не порушувати законодавство України щодо персональних даних та використання трафіку.

Підтримка української мови: проблема полягає у кращому функціонуванні генеративних моделей на англійській мові. Незважаючи на базову підтримку української, якість генерації відрізняється та зазвичай нижчого рівня, що створює обмеження в сюжеті та діалогах.

Після аналізу та визначення основних факторів, виявлені обмеження формують межі системи та вплинули на прийняття технічних та архітектурних рішень. Їх урахування становить важливу частину та умову для реалістичної реалізації проекту та відповідність поставленим цілям.

2.1.6. Класифікація та трасування вимог

Після визначення усіх основних факторів, для визначення повноти, логічності та відстежуваності вимог, стоїть задача їх класифікувати. Це дозволить розуміти зв’язки між вимогами, цілями, обмеженнями та функціональністю системи та уникнути дублювання, суперечностей.

Визначена класифікація вимог:

Бізнес-вимоги – визначають загальну мету системи.

Користувацькі вимоги – описують систему та її поведінку з боку користувача.

Функціональні вимоги – конкретизують дії, які система повинна виконувати.

Нефункціональні вимоги – визначають якісні характеристики.

Обмеження – описують зовнішні фактори або внутрішні чинники які разом складають межі реалізації системи.

Трасування вимог до системи:

Для забезпечення повноти реалізації функціональних вимог до системи було виконано трасування між основними артефактами моделі: вимогами, варіантами використання (прецедентами), тест-кейсами та елементами користувацького інтерфейсу. Це дозволяє перевірити, що кожна вимога має відповідне покриття в системі та тестах. Переглянути результат діаграми на рисунку 2.5.

На самому рисунку зображено діаграму трасування, яка відображає зв’язки між вимогами, прецедентами, тест кейсами та приблизними елементами інтерфейсу користувача за допомогою зв’язків trace. В результаті чого діаграма дозволяє простежити, як кожна вимога реалізується через прецеденти, перевіряється тестами та відображається у інтерфейсі.

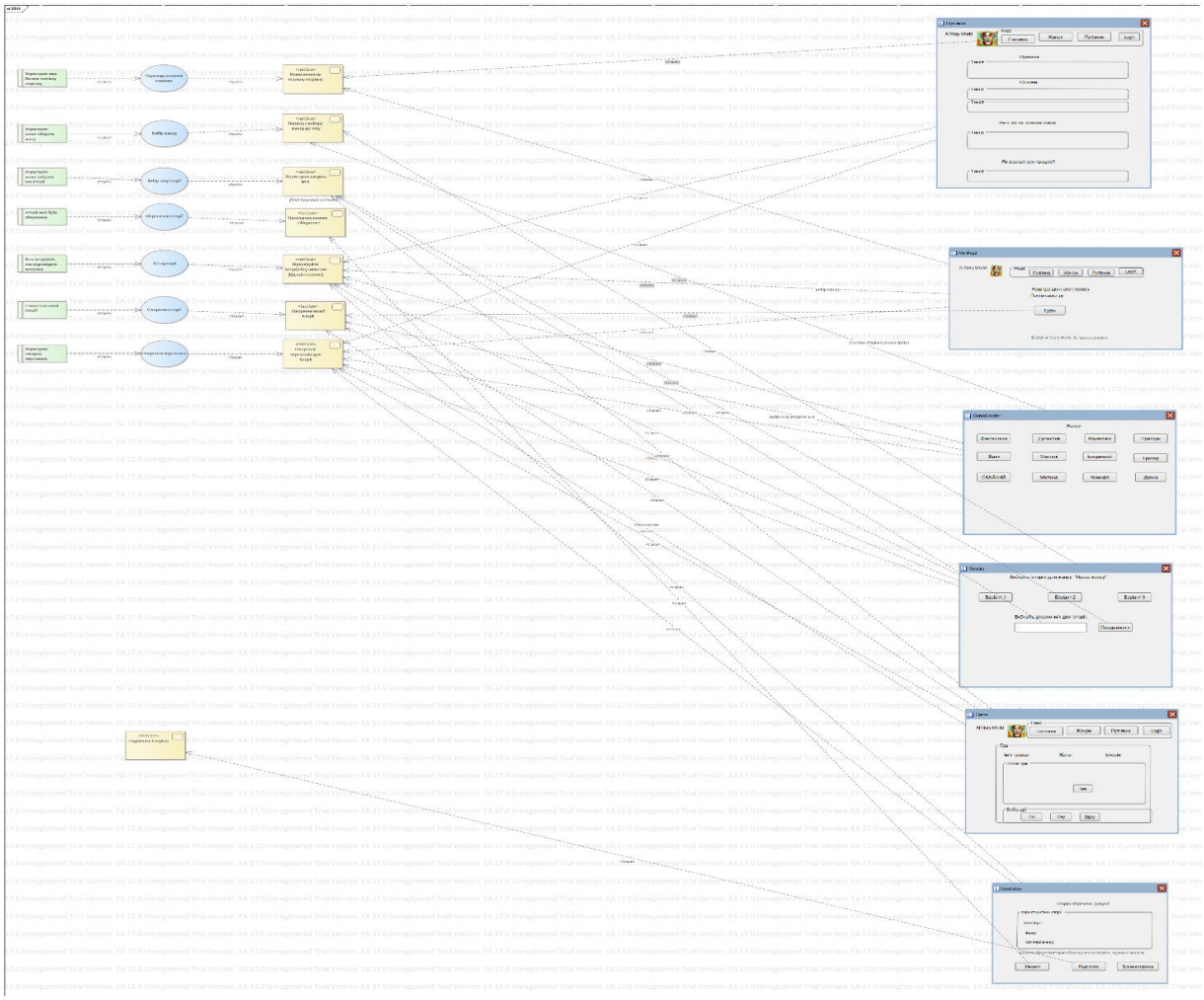


Рисунок 2.5 – Діаграма трасування вимог до системи

Джерело: сформовано автором на основі виконаного дослідження

2.2. Постановка та алгоритм розв’язання задачі

2.2.1. Постановка задачі.

2.2.1.1 Характеристика задачі.

Основною задачі, що розв’язується в цьому проекті, полягає у розробці та створені веб-платформи, що лає змогу користувачеві створювати і брати пряму участь у текстових історіях у режимі реального часу. Основа системи полягає на генеративному ШІ, за допомогою якого і відбувається логічно зв’язаний сюжет з урахуванням дій користувача.

Призначення та сутність задачі:

Мета проекту полягає у створені, оптимізації та нового досвіду у проходженні текстових історій з участю користувача. Система надає можливість отримати унікальний досвід керування історією у реальному часі, за рахунок чого знижує потребу у ручному написанні історій та підвищенні ефективності генерації контенту. Також з підключенням ШІ, проект матиме гнучкість, масштабованість та адаптованість під потреби користувача.

Використання ЕОМ:

ЕОМ це невід’ємна частина проекту, оскільки робота проекту, взаємодія з користувачем у реальному часі, збереження усієї інформації, аналіз існуючих даних, обробка запитів і всі інші процеси проекту вимагають швидкодійної ОЕМ та хмарного середовища для збереження даних.

Перелік Об’єктів управління:

1. Користувач - Реєстрація, редагування профілю, створення власних історій.
2. Історії - Можливість їх створення, генерація продовження за допомогою ШІ, збереження і подальше продовження.
3. Кроки історії - Індивідуальні продовження за результатом запиту користувача.
4. Жанри історій - Десятки різних жанрів для урізноманітнення контенту.
5. Типи історій - Декілька різних варіантів історії в певному жанрі для конкретизації.
6. Профілі користувачів - збереження імені, опису, історії історій користувача.

7. Доступ - контроль дій відносно ролі та реєстрації користувача.
8. База даних - збереження усієї інформації.

Призначення та використання вхідної інформації:

Генерація вихідної інформації формується на основі запитів користувача та взаємодії з ШІ. Вихідна інформація в себе включає: відображення результату історії з діями користувача, зворотній зв'язок з інтерфейсом, можливість продовження минулих історій, формування історій відносно вибору жанру та типу історії, повідомлення про критичні помилки та збереження. Саме використання цієї інформації ділиться на два класи - користувача та системи.

Використання користувачем: Читання згенерованої історії, приймання рішень на основі фрагменту, продовжувати взаємодію з історією, зміни у профілі.

Користування системою: Збереження результатів генерації, використання минулих кроків та фрагментів історії у генерації наступної частини, забезпечення адаптації та безперервність історії.

Важливо зазначити за **періодичність розв'язання задач** у системі, адже вони реалізуються безперервно – кожна нова дія користувача запускає цикли аналізу, генерації що займають від декількох секунд.

Існують умови коли **припиняється автоматизація розв'язування задач**, задача зупиняється коли: користувач виходить з системи, довго не взаємодіє, втрата інтернет зв'язку або збій серверу, в рідких випадках досягнення ліміту взаємодії з ШІ або логічне завершення історії.

Основна задача взаємодіє та поділяєтеся на менші задачі та підсистеми: систему реєстрація та авторизації, систему збереження історій, систему вибору жанру та типу історій, систему генерації історій, систему зміну профілю, системи перевірки на інформацію та інші.

Розподіл задач у проекту відбувається на основі персоналу та технічних засобів у такому форматі: користувач, генеративна ШІ - система, серверна частина та розробник. Кожна частина має свої задачі та цілі.

Також сформовано **просту інформаційну модель задачі** для веб-додатку на рисунку 2.6.

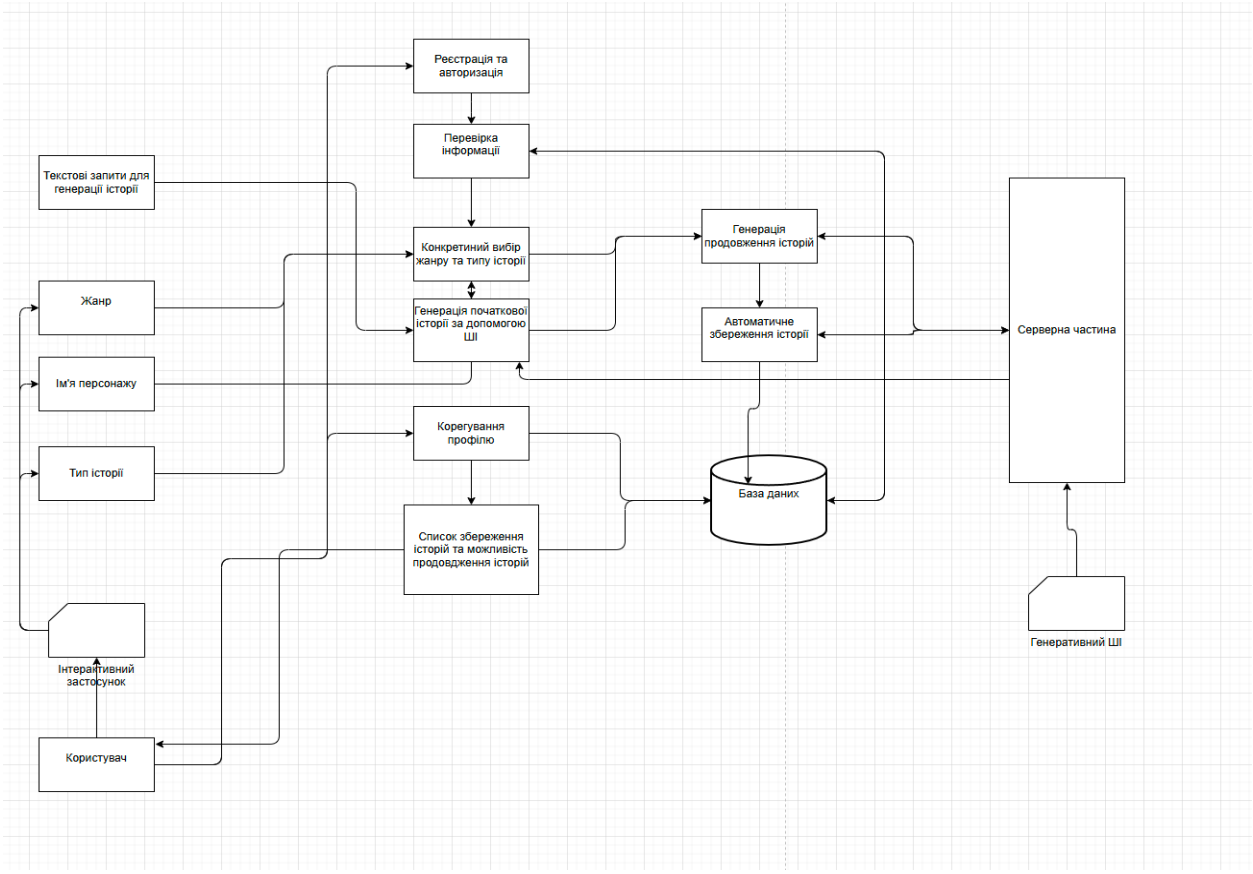


Рисунок 2.6 – Інформаційна модель розв’язання задачі

Джерело: сформовано автором на основі виконаного дослідження

2.2.1.2. Вихідна інформація

Призначення вихідної інформації в системі формується в результаті обробки запитів користувача з генеративною ШІ моделлю. Мета цієї інформації, це забезпечити користувача логічною початковою історією та її змістовним продовженням на основі нових дій та контексту з минулих частин. Також з’являється інформація-повідомлення про поточні дії та стан сесії в інтерфейсі користувача.

Основні характеристики цієї інформації:

- Вона відображається в текстовому інтерфейсі гри.
- Вона може бути збережена для подальшого використання.
- Використовується для аналізу та побудови подальшої логіки та взаємодії.

- Доступна для перегляду у базі даних.

Таблиця 2.1 переліку вихідних повідомлень

№	Назва повідомлення	Ідентифікатор	Форма подання	Періодичність видання	Допустимий термін затримки	Користувачі інформації
1	Відповідь від ШІ	OUT_MSG_01	Текст, JSON\API	Кожен запит від користувача	До 10 секунд	Користувач-Гравець
2	Повідомлення про успішний вхід	OUT_MSG_02	Текст	Після авторизації	Миттєво	Користувач-Гравець
3	Повідомлення про невірні облікові дані	OUT_MSG_03	Текст	Після спроби авторизації	Миттєво	Користувач-Гравець
4	Повідомлення про потребу входу в аккаунт	OUT_MSG_04	Текст	Під час користування сайту без авторизації	Миттєво	Користувач-Гравець
5	Повідомлення про помилку в роботі сервісу	OUT_MSG_05	Текст	За потреби або несправності проекту	Миттєво	Користувач-Гравець, Розробник
6	Службові повідомлення	OUT_MSG_06	Лог-сервіс	У фоновому режимі	До 5 хв	Розробник
7	Збереження історії	OUT_MSG_07	Текст	Текст	Миттєво	Користувач-Гравець

Таблиця 2.2 опис структурних одиниць

Джерело: сформовано автором на основі виконаного дослідження

№	Назва показника	Ідентифікатор повідомлення	Вимоги до точності\ надійності
1	Згенерований текст історії	OUT_MSG_01	Чітко пов’язаний з минулою частиною
2	Дата та час дії та збереження	OUT_MSG_01, OUT_MSG_05, OUT_MSG_06, OUT_MSG_07	До 2-ух секунд точності
3	Тип помилки	OUT_MSG_03, OUT_MSG_04, OUT_MSG_05	Класифіковано по кодах
4	Ідентифікатор користувача	OUT_MSG_02, OUT_MSG_07,	Унікальність ідентифікатора
5	Стан сесії	OUT_MSG_07	Перевірка правильності даних
6	Ідентифікатор історії	OUT_MSG_07	Унікальність ідентифікатора

2.2.1.3. Вхідна інформація

Джерело: сформовано автором на основі виконаного дослідження

Вхідна інформація є неймовірно важливою, адже це дані, які надходять від користувача або внутрішніх систем і використовуються для роботи усіх функцій проекту, генерації історій, збереження історій, налаштування

параметрів гри, жанри, імена та інші. Ця інформація визначає початкові налаштування для генерації історії та логіку взаємодії. Вхідну інформація ділиться на: текстові дії користувача в діалозі з ШІ, вибір жанру, типу історії та імені персонажа, налаштування профіль та службових параметрів (Ідентифікатори, статуси та час).

Таблиця 2.3 переліку вхідних повідомлень

№	Назва повідомлення	Ідентифікатор	Форма подання	Періодичність	Джерело
1	Запит користувача	IN_MSG_01	Текст (введення)	Після кожного введення	Інтерфейс користувача
2	Вибір жанру	IN_MSG_02	Dropdown/menu	Один раз на початку	Інтерфейс гри
3	Вибір типу історії	IN_MSG_03	Dropdown/menu	Один раз на початку	Інтерфейс гри
4	Ім’я персонажа	IN_MSG_04	Текст	Один раз на початку	Інтерфейс гри
	Кнопка дії	IN_MSG_05	Подія	За ініціативою користувача	Користувач
5	Кнопка «Зберегти»	IN_MSG_05	Подія	За ініціативою користувача	Користувач
6	Дані профілю	IN_MSG_06	Форма	За запитом	Редактор профілю
7	Повідомлення про стан сесії	IN_MSG_07	JSON/API	Періодично	Сервер
8	Авторизаційні дані	IN_MSG_08	Текст	При вході	Інтерфейс автентифікації
9	Оновлення даних профілю	IN_MSG_10	Форма	Зазапитом	Інтерфейс профілю

Таблиця 2.4 опис структурних одиниць вхідних повідомлень

Джерело: сформовано автором на основі виконаного дослідження

№	**Назва показника**	**Джерело**	**Ідентифікатор**	**Вимоги до точності**
1	Текст запиту користувача	Клавіатура	IN_MSG_01	До 200 символів
2	Ідентифікатор користувача	База даних	IN_MSG_01	Унікальний
3	Обраний жанр	Dropdown інтерфейсу	IN_MSG_02	Присутній у переліку
4	Тип історії	Dropdown інтерфейсу	IN_MSG_03	Присутній у переліку
5	Ім’я персонажа	Текстове поле	IN_MSG_04	До 100 символів
6	Команда «Зберегти»	Кнопка інтерфейсу	IN_MSG_05	Подія зворотного зв'язку
7	Дата й час	Системний сервер	IN_MSG_01, IN_MSG_08, IN_MSG_09	ISO 8601
8	Команда «Продовжити»	Кнопка інтерфейсу	IN_MSG_01	Подія зворотного зв'язку
9	Команда «Зберегти зміни»	Кнопка інтерфейсу	IN_MSG_01	Подія зворотного зв'язку

2.2.2. Алгоритм розв’язання задачі

Джерело: сформовано автором на основі виконаного дослідження

Використовувана інформація:

Для правильної реалізації алгоритму генерації історій з сюжетом на базі генеративного ШІ, використовується інформація, яка надходить від користувача, проміжні дані та дані сформовані попередніми діями або вже знаходяться в базі даних. Ця інформація обробляється алгоритмами в реальному часі для формування початкової історії, її сюжетного продовження, цілісності та логіки у межах вибраного жанру та типу історії.

Категорії використовуваної інформації діляться на:

- Вхідні запити користувача.
- Метадані сесії (Ідентифікатор, поточний крок, статус збереження, минулі кроки).
- Вибори користувача (Жанр, тип історії, ім'я персонажа).
- Дані профілю.
- Попередні частини історій та їх контекст.
- Конфігурація ШІ.

Для переліку усіх масивів була створена таблиця №

Таблиця 2.5 перелік масивів використовуваної інформації

№	Масив	Ідентифікатор	Максимальна к-сть записів
1	Користувачі	MASV_01	1000
2	Жанри	MASV_02	50
3	Типі історії	MASV_03	200
4	Історії	MASV_04	1000
5	Кроки історії	MASV_05	5000
6	Профільна інформація	MASV_06	400
7	Вхідна текстова інформація	MASV_07	200
8	Контекст сесії	MASV_08	300
9	Обмеження генерації(промт, темп стиль)	MASV_09	10-100

Результати розв'язання:

Джерело: сформовано автором на основі виконаного дослідження

Після автоматизованого розв'язання задачі користувач отримує власну унікальну сюжетну лінію за його запитом, яка є логічним продовженням інтерактивної історії. Також система паралельно генерує супутні службові

результати, що забезпечують збереження прогресу, заповнення масивів інформації, підтримку адаптації історії у межах одного профілю.

Ця інформація використовується для виведення на екран користувача, збереження історій у базі даних, формування усіх кроків історії і подальшого аналізу взаємодії.

Таблиця 2.6 таблиця масивів

№	Масив	Ідентифікатор	Максимальна к-сть записів
1	Згенерований фрагмент історії	MASR_OUT_01	10 000
2	Повні історії	MASR_OUT_02	1 000
3	Системні повідомлення в інтерфейсі	MASR_OUT_03	1 000
4	Записи про помилки та логи	MASR_OUT_04	100
5	Збережений контекст з минулої генерації	MASR_OUT_05	1 000
6	Список усіх минулих історій	MASR_OUT_06	100

Джерело: сформовано автором на основі виконаного дослідження

Алгоритм розв’язання задачі на ЕОМ:

Алгоритм розв’язання задач у даному проекті забезпечує логіку взаємодії користувача з системою, генерацію сюжетних ліній на основі введених запитів, обраних стилів, обробку та збереження результатів у базі даних, а також перегляд інформації.

Процес реалізується у вигляді послідовних етапів, що відповідають усім функціональним блокам системи. Усі дії відбуваються у реальному часі за участю користувача, системи та генеративного ШІ. Дивитись на Рис 2.7

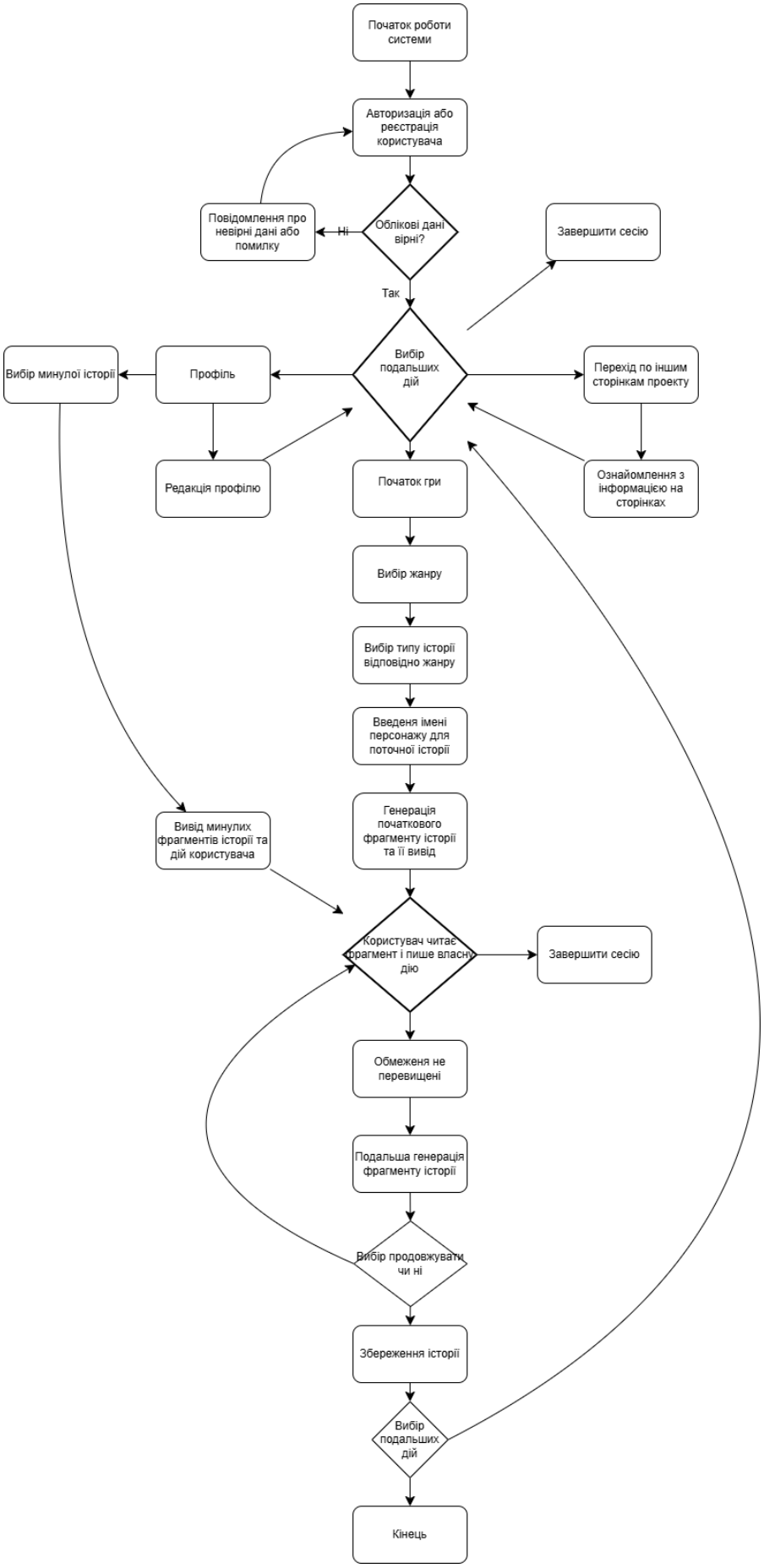


Рисунок 2.7 – Схема алгоритму дій в системі

Джерело: сформовано автором на основі виконаного дослідження

Детальна послідовність етапів алгоритму:

1. **Початок роботи системи.**
2. **Авторизація або реєстрація користувача:**

   - Користувач вводить облікові дані.
   - Перевірка на коректність:

   a. якщо **невірні** — виводиться повідомлення про помилку, повернення до авторизації;
   b. якщо **вірні** — перехід до вибору подальших дій.

3. **Вибір подальших дій після входу:**

   - Перехід на профіль та його редагування;
   - Перехід по інформаційних сторінках або вихід із сесії;
   - Початок гри або вибір раніше збереженої історії.

4. **У разі вибору продовження старої історії:**

   - Система виводить попередні фрагменти та дії користувача;
   - Перехід до етапу генерації нового кроку історії.

5. **У разі початку нової гри:**

   - Вибір жанру історії;
   - Вибір типу історії відповідно до жанру;
   - Введення імені персонажа.

6. **Генерація початкового фрагменту історії та його вивід.**
7. **Очікування взаємодії з користувачем:**

   - Користувач читає згенерований фрагмент та вводить власну дію.

8. **Перевірка обмежень (час, кількість кроків, розмір тексту):**

   - Якщо обмеження не перевищені — продовження генерації;
   - Якщо досягнуто ліміту — можливість завершити історію.

9. **Генерація нового фрагменту історії ШІ.**
10. **Після кожного кроку:**

- Користувач вирішує — **продовжити чи ні**.

11. **Якщо продовження не обране:**

- Відбувається збереження історії в базу даних.

12. **Повернення до вибору подальших дій:**

- Користувач може почати нову історію, відредагувати профіль або вийти з системи.

13. **Кінець (вихід або завершення сесії).**

2.3. Моделювання інформаційної підсистеми

2.3.1. Моделювання поведінки системи

Моделювання поведінки системи розпочато з побудови діаграм, перша це **діаграма прецедентів**, яка дозволить подати функціональні вимоги до системи. Дана діаграма описує сценарії взаємодії користувача з проектом.

На діаграмі зображено користувача, що ініціює дії, пов’язані з параметрами історії та роботою системи. Всі прецеденти знаходяться у межах веб-додатку що відокремлює їх як внутрішні процеси.

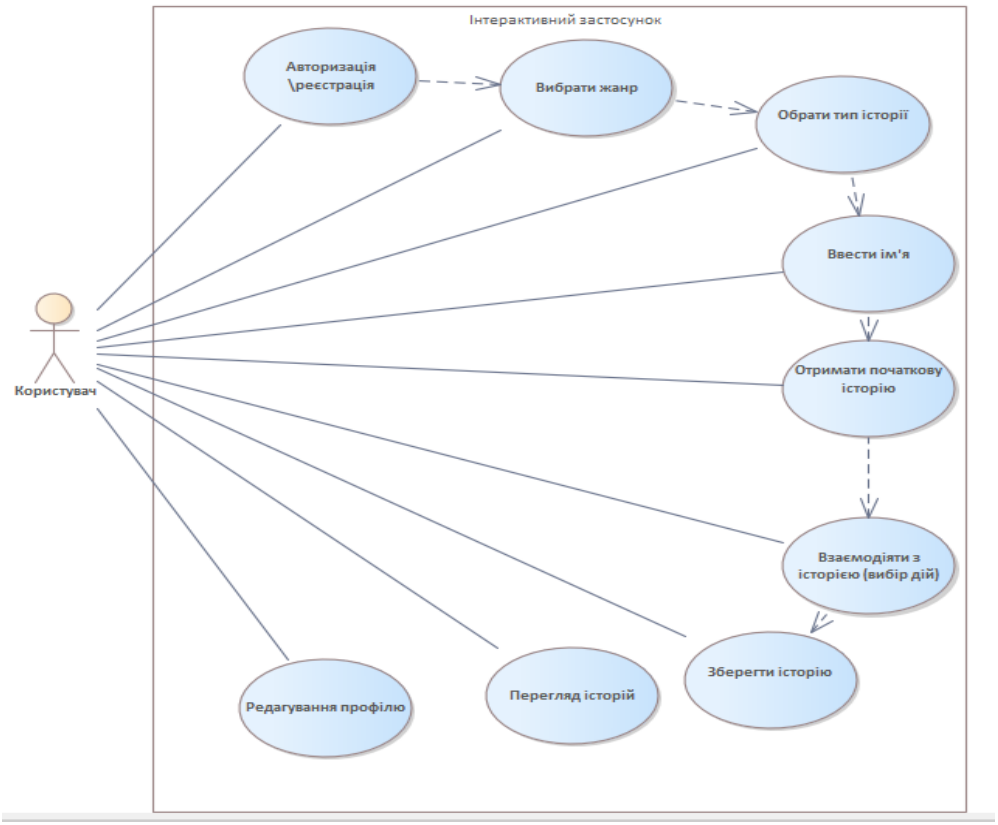


Рисунок 2.7 - діаграма прецедентів для взаємодії користувача з інтерактивним застосунком

Джерело: сформовано автором на основі виконаного дослідження

Також побудовано діаграми послідовності, які відображають динамічну поведінку системи у відповідності до сформованих сценарних варіантів взаємодії користувача. Кожна діаграма демонструє, як об’єкти системи (користувач, форми, сховища) взаємодіють між собою через повідомлення.

Створені такі діаграми: «Введення імені», «Вибір жанру», «Вибір типу історії», «Отримання початкової історії», «Взаємодія з історією», «Збереження історії», «Редагування профілю».

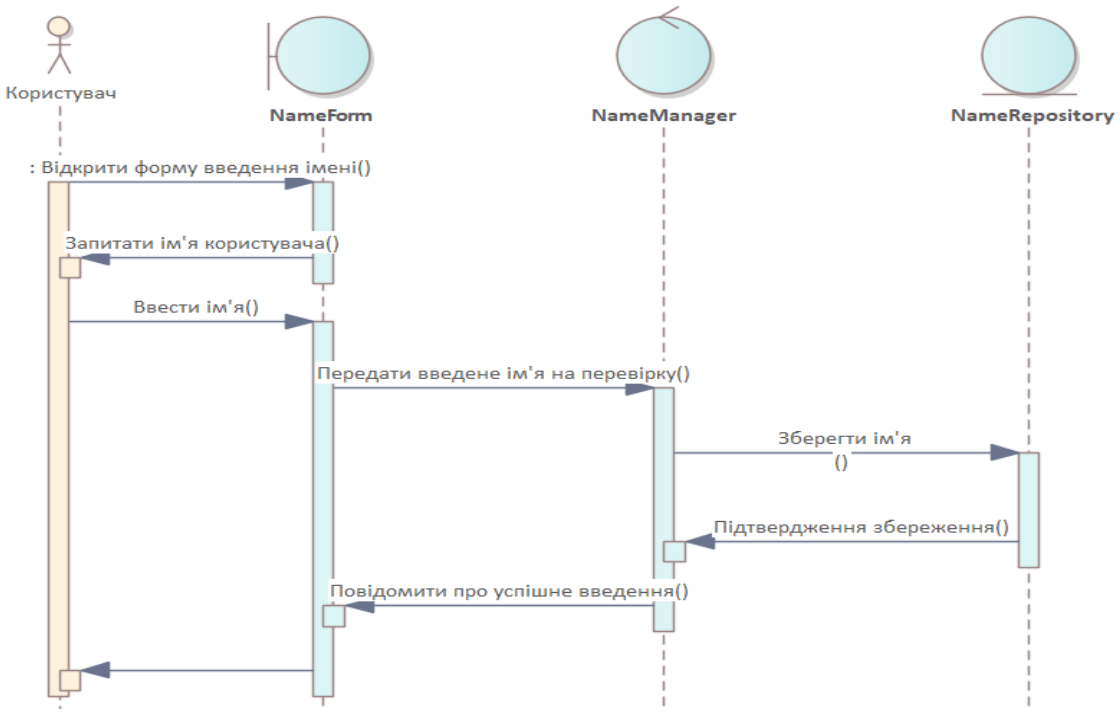


Рисунок 2.8 – Діаграма послідовності «Вибір жанру»\

Джерело: сформовано автором на основі виконаного дослідження

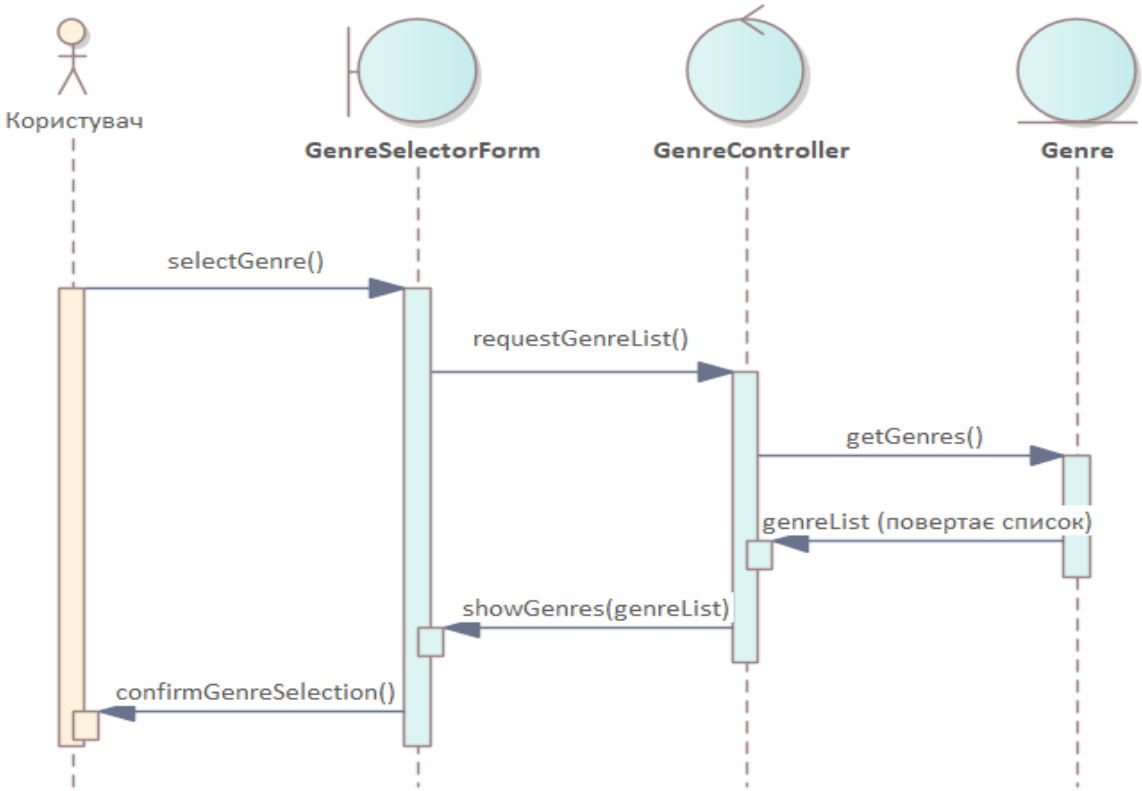


Рисунок 2.9 – Діаграма послідовності «Вибір жанру»

Джерело: сформовано автором на основі виконаного дослідження

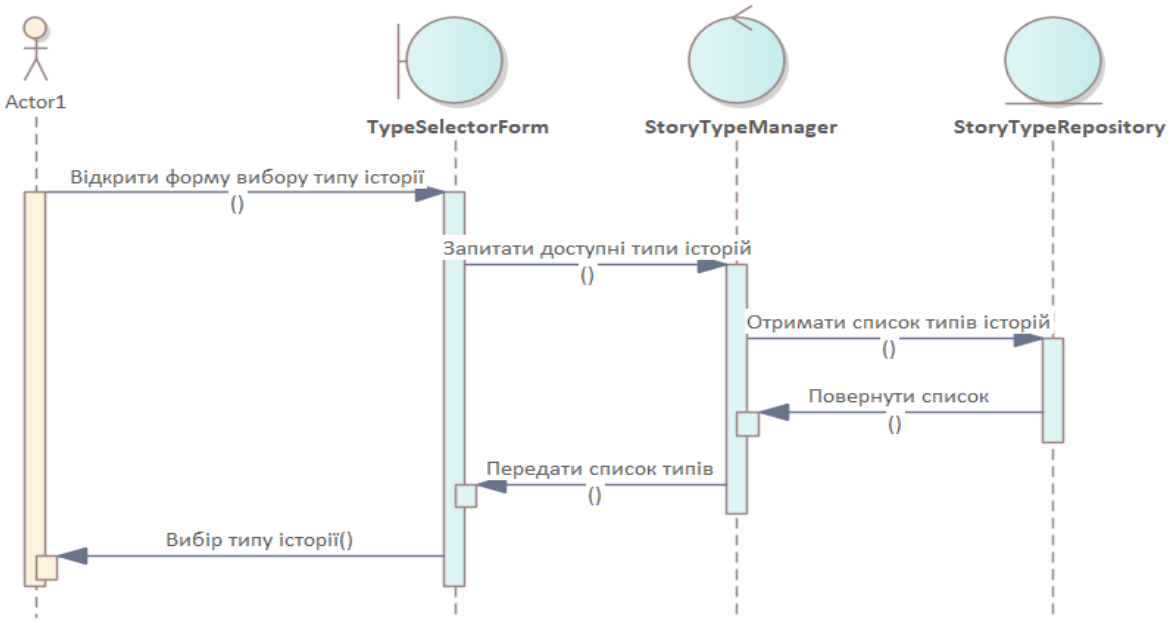


Рисунок 2.10 – Діаграма послідовності «Обрати тип історії»

Джерело: сформовано автором на основі виконаного дослідження

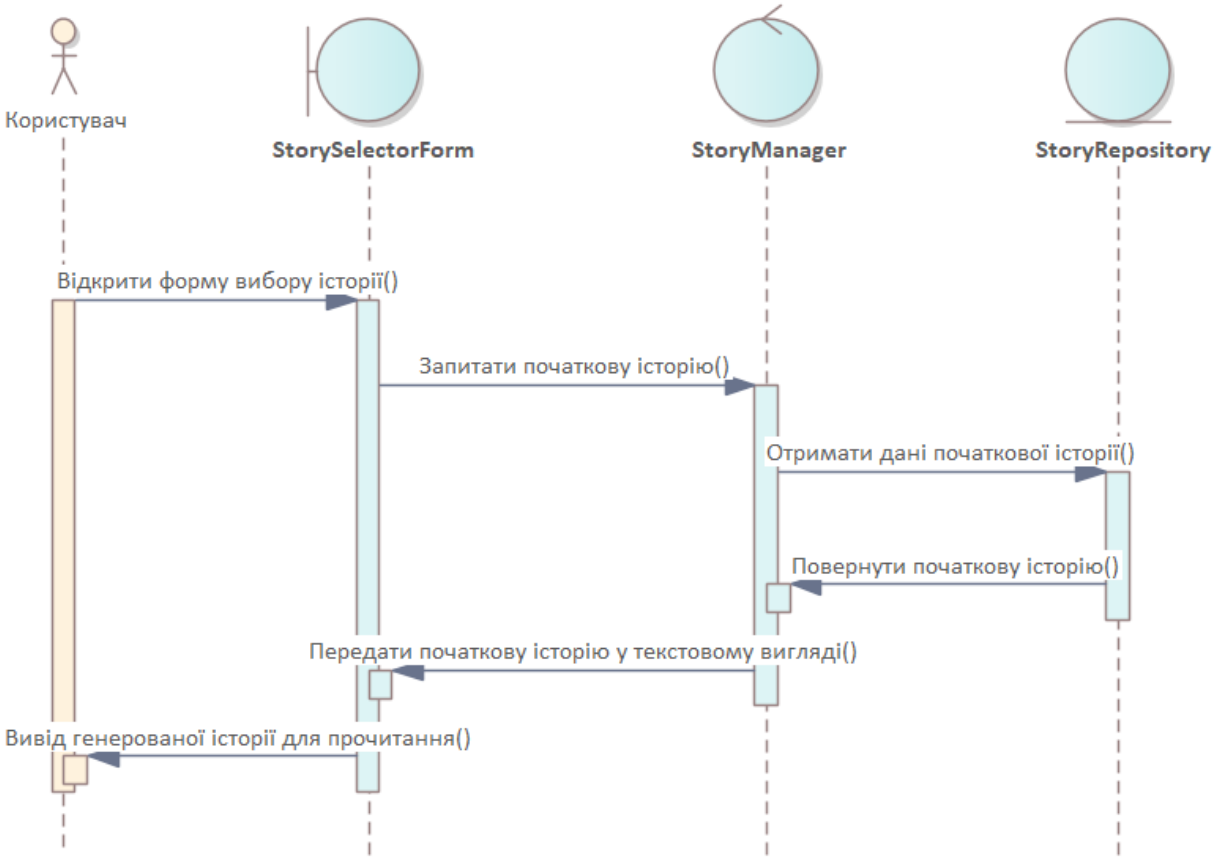


Рисунок 2.11– Діаграма послідовності «Отримання початкової історії»

Джерело: сформовано автором на основі виконаного дослідження

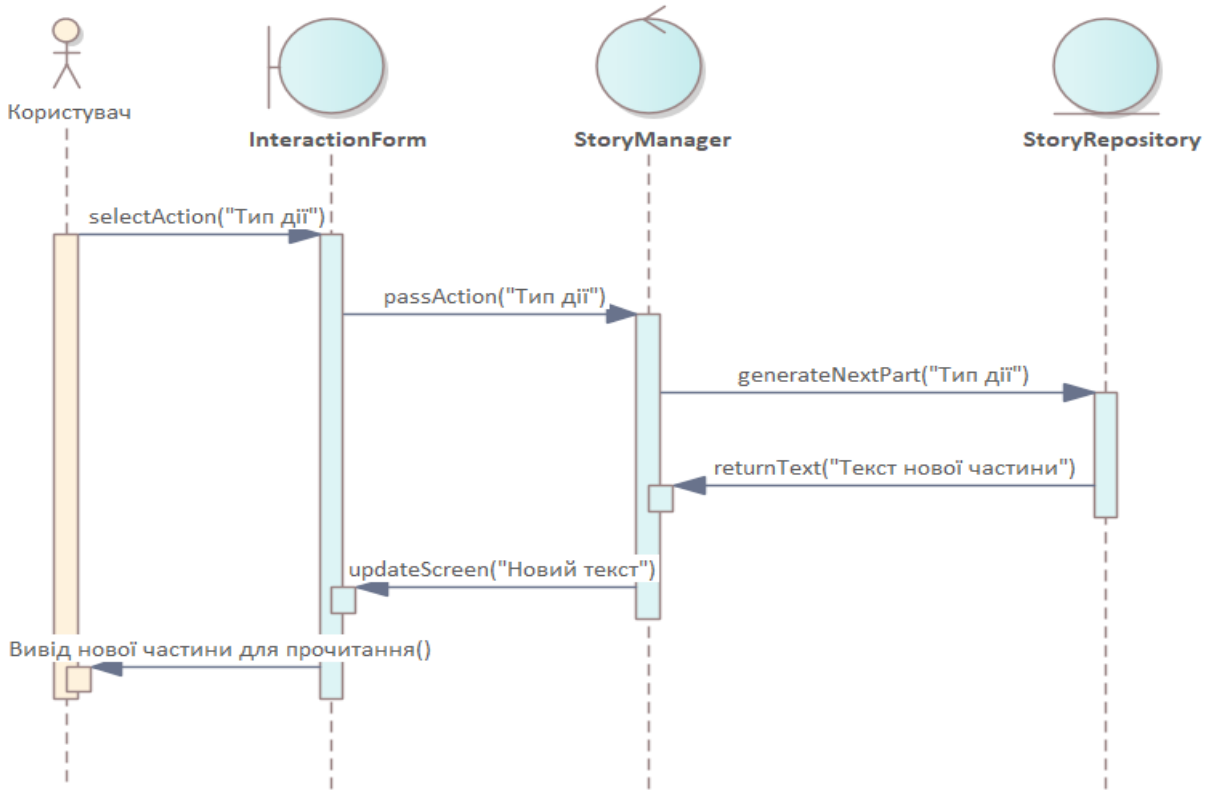


Рисунок 2.12 – Діаграма послідовності «Взаємодія з історією»\

Джерело: сформовано автором на основі виконаного дослідження

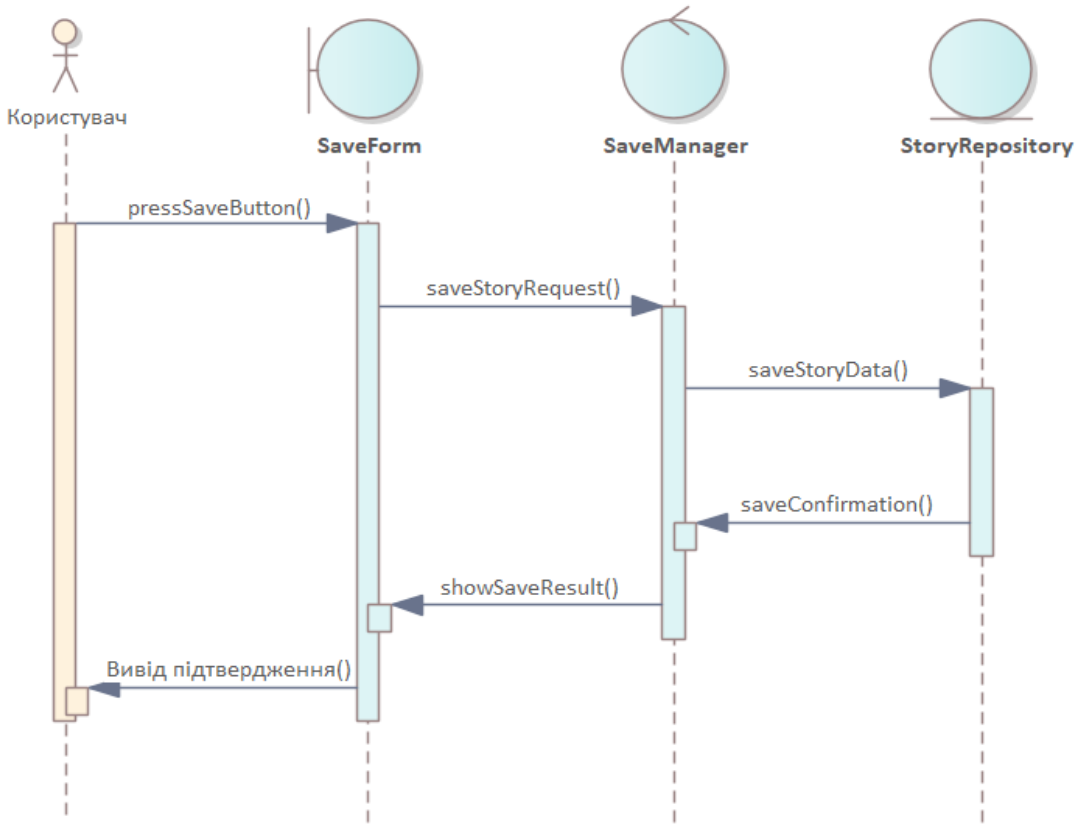


Рисунок 2.13 – Діаграма послідовності «Збереження історії»

Джерело: сформовано автором на основі виконаного дослідження

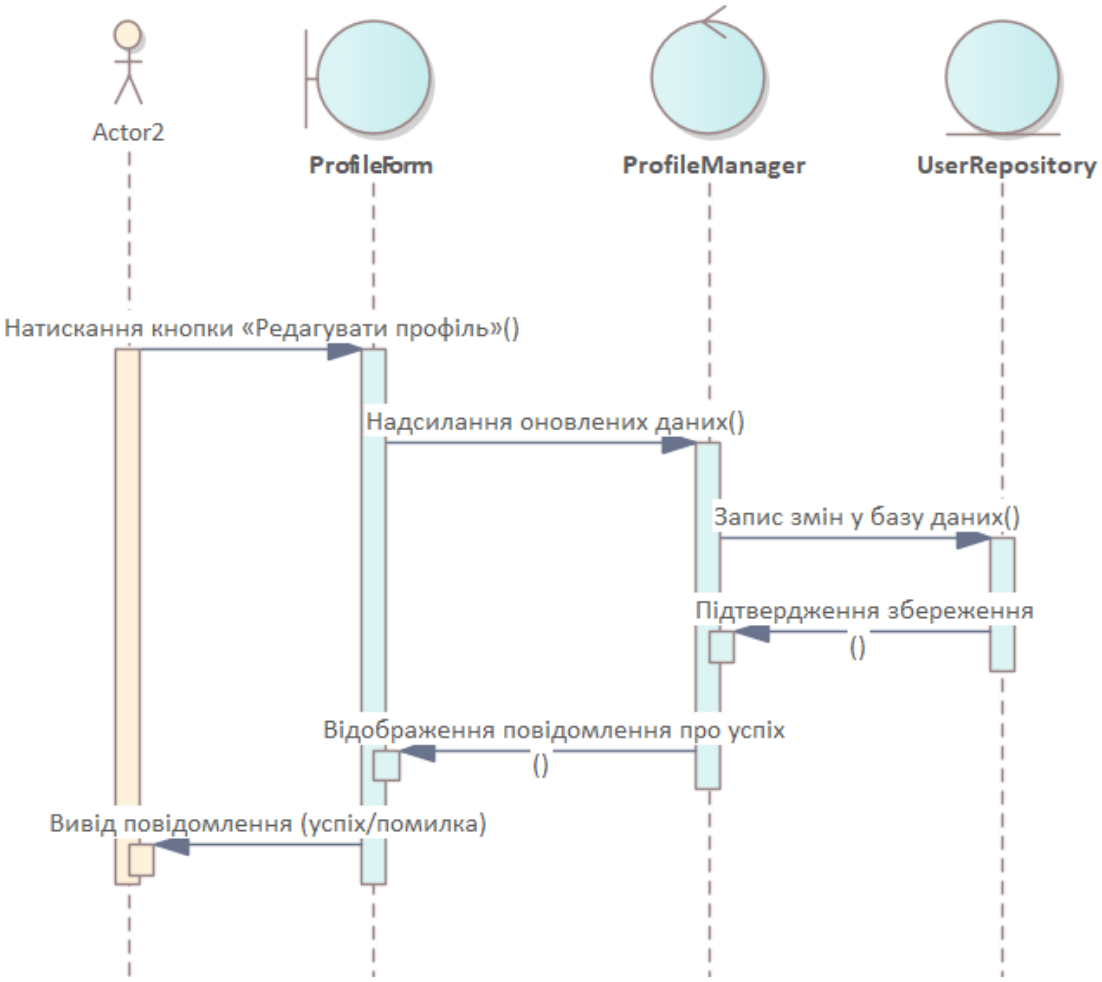
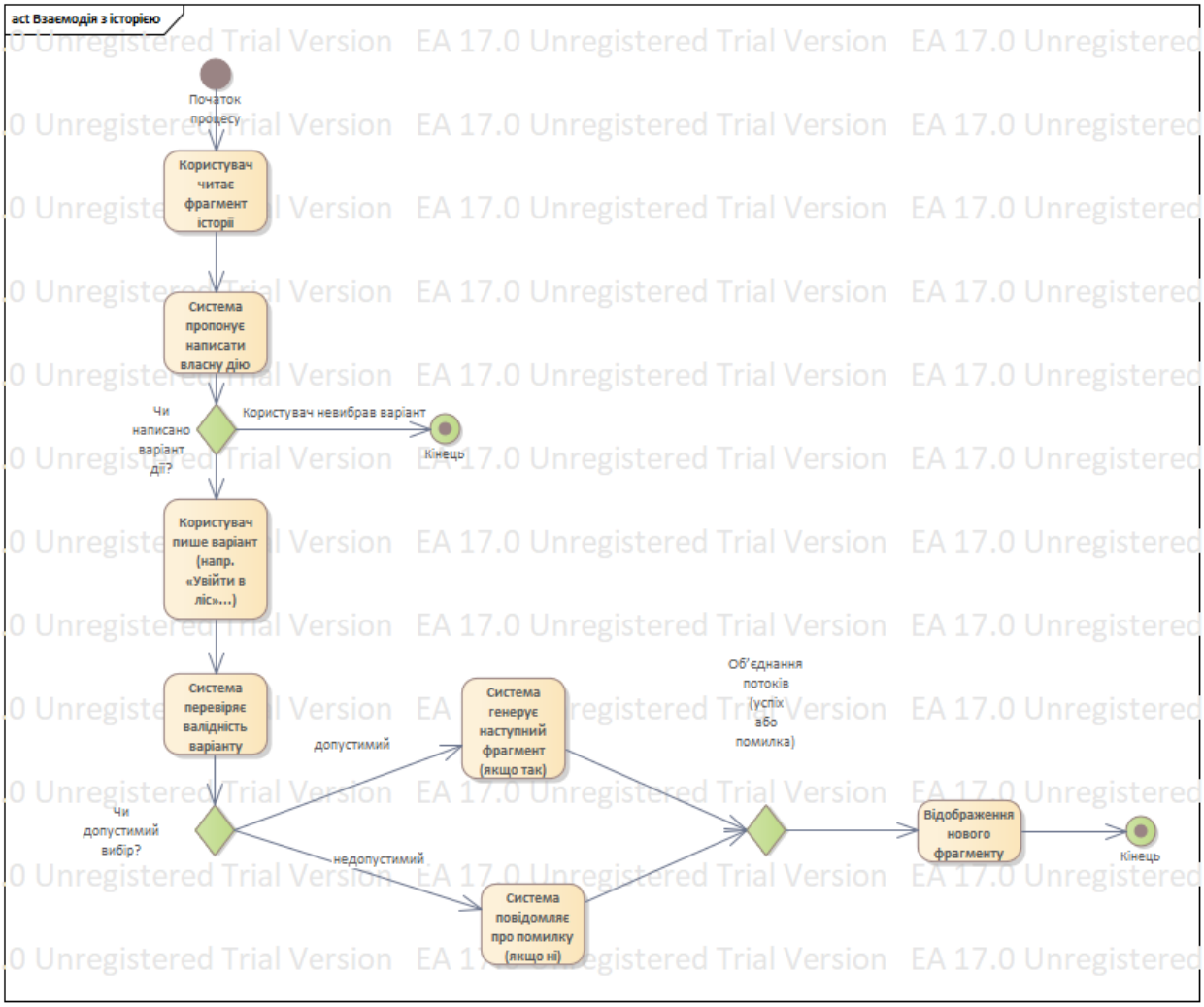


Рисунок 2.14 – Діаграма послідовності «Редагування профілю»

Джерело: сформовано автором на основі виконаного дослідження

Під час моделювання поведінки системи подано у вигляді діаграми діяльності, котра використовується для візуалізації потоку керування між різними системами. У системі сценарію «Взаємодія з історією» передбачається, що користувач читає фрагмент генерованої історії та приймає рішення для дії. На основі вписаних дій користувача, система генерує новий фрагмент або виводить помилку.



Джерело: сформовано автором на основі виконаного дослідження

Рисунок 2.15 – Діаграма діяльності для сценарію «Взаємодія з історією»

2.3.2. Моделювання структури системи

Для представлення архітектури у повному форматі, у даному підрозділі здійснено структурне моделювання, що охопило логічну, так і фізичну складову. Було створено декілька діаграма : діаграми визначення блоків (Block Definition Diagram, BDD) та діаграми внутрішніх блоків (Internal Block Diagram, IBD). Також враховуючи орієнтацію проекту на інформаційну систему на взаємодію між сервісними модулями та користувачами, створена діаграма класів.

Задачею цих діаграм є визначення складу основних компонентів системи, ролей, взаємозв'язків, інтерфейс-взаємодій та даних що циркулюють в системі. Проаналізувавши структурне моделювання, забезпечило можливість узгодити основу для подальшої реалізації архітектури проекту.

Діаграма класів: Ця діаграма дозволяє візуалізувати статичну будову важливої частини системи яка відповідає за головну суть проекту. Також створено таблицю з загальною структурою класів.

Таблиця 2.7 – Структура класів

Назва об'єкту	Атрибути	Методи
NameForm	inputField: string	submitName(), validateInput()
NameManager	–	checkName(String): boolean
StoryManager	currentStory: String	generateNextFragment(), loadStory(), save()
StoryRepository	–	loadStory(): String, saveStory(String)
GenreSelectorForm	selectedGenre: String	confirmGenre(), renderGenres()
GenreController	–	applyGenre(String), getGenres()
StartForm	startButton: Button	startStory()
SaveForm	saveButton: Button	onSaveClick()
StoryTypeForm	selectedType: String	chooseType()
InteractionForm	choices: String, storyText: String	displayChoices(), submitChoice()

Джерело: сформовано автором на основі виконаного дослідження

На діаграмі **центральним елементом** є клас StoryManager, який координує логіку взаємодії з історією. Він отримує дані від інтерфейсних форм, керується перевіркою валідності, жанром, а також взаємодіє з репозиторієм для збереження й завантаження історій.

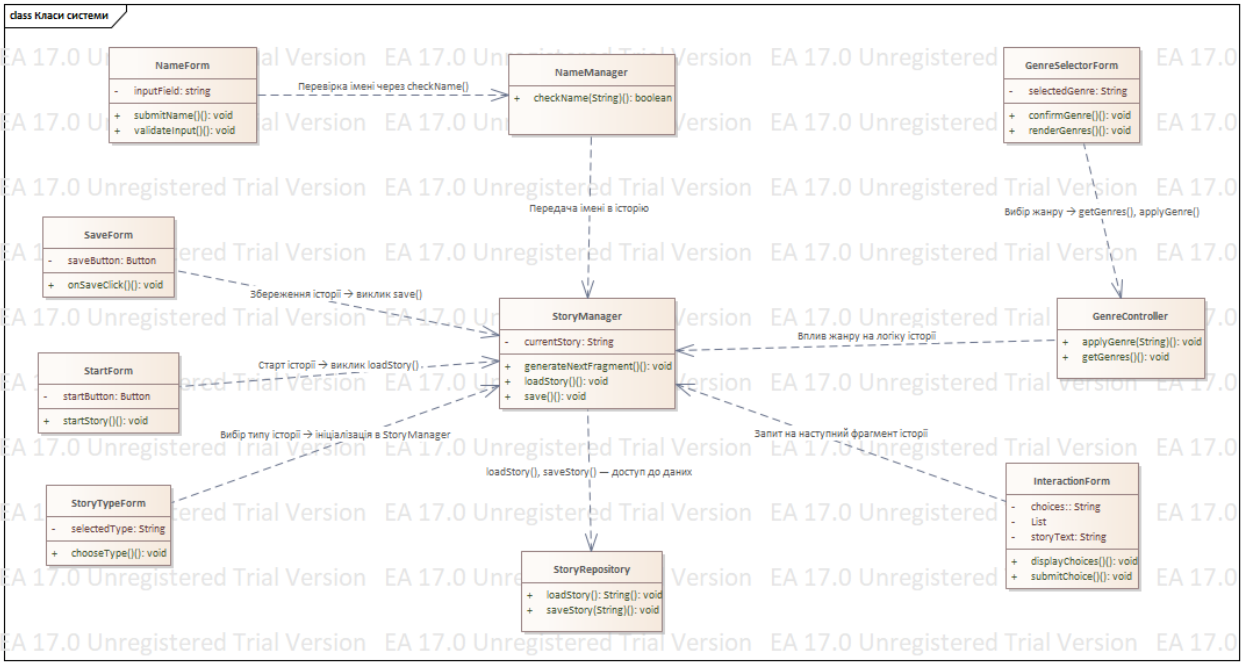


Рисунок 2.16 – Діаграма класів системи

Джерело: сформовано автором на основі виконаного дослідження

Діаграма визначення блоків (BDD) – в цій діаграмі подано загальну архітектуру системи, побудовану ієрархію технічних та програмних

продуктів. Див рис №2.17. З діаграми виділяється дві головні підсистеми: технічне забезпечення та програмне забезпечення.

Технічне включає в себе: ClientDevice – клієнтська частина (пристрій користувача) та ServerNode – серверна частина, де розгорнута основна логіка.

Програмне забезпечення в свою чергу включає: UI – інтерфейс користувача, API Gateway – точка входу запитів GenreService, StoryService, NameManager – прикладні модулі та Database – сховище даних.

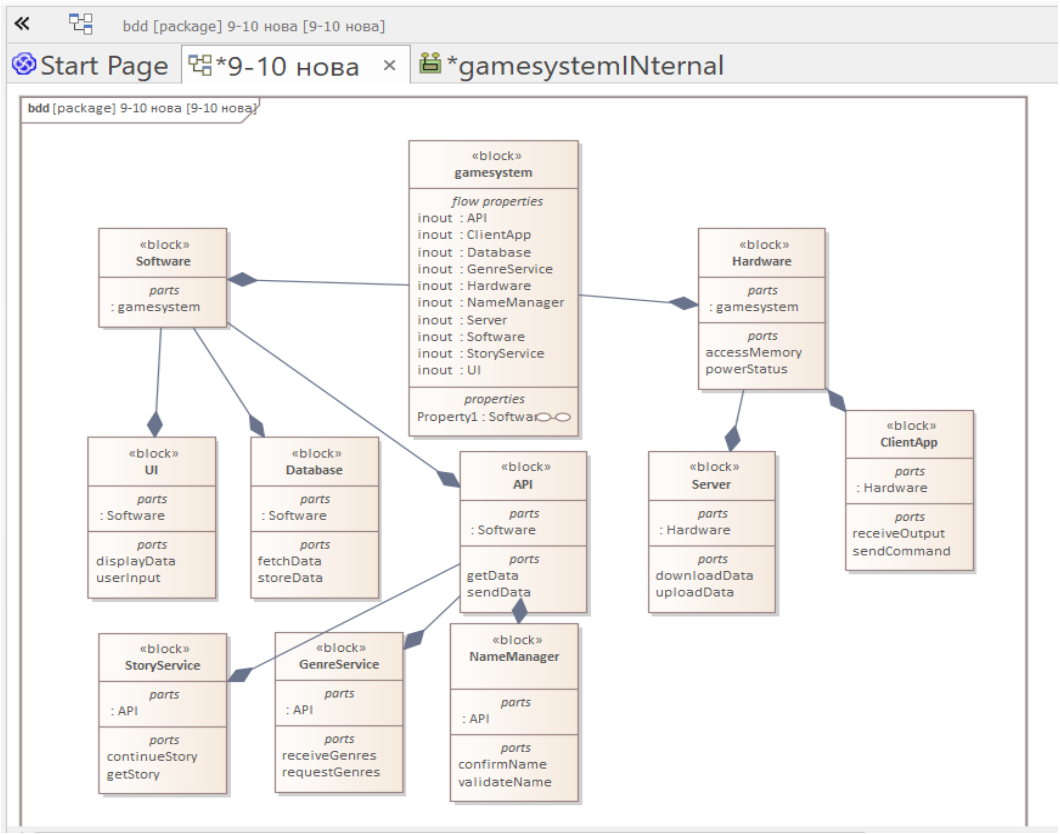


Рисунок 2.17 – Діаграма визначення блоків системи

Джерело: сформовано автором на основі виконаного дослідження

Діаграма внутрішніх блоків (IBD) - деталізовано взаємодію між компонентами програмного забезпечення. Особливу увагу зосереджено на чіткій демонстрації потоків запитів та відповідей між UI, API, ServiceLayer та Database.

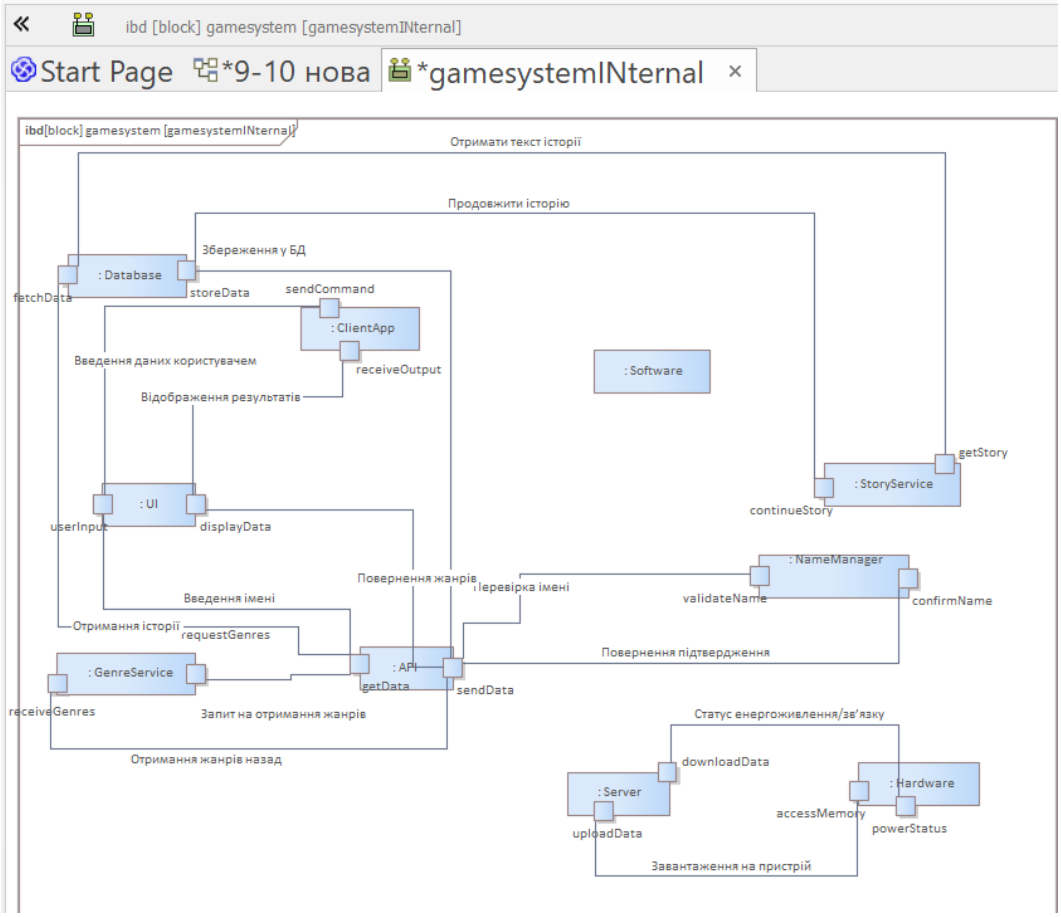


Рисунок 2.18 – Діаграма внутрішніх блоків системи

Джерело: сформовано автором на основі виконаного дослідження

2.3.3. Розподіл вимог за компонентами системи

Для забезпечення повноти реалізації функціональних вимог до системи було виконано трасування між основними артефактами моделі: вимогами, варіантами використання (прецедентами), тест-кейсами та елементами користувацького інтерфейсу. Це дозволяє перевірити, що кожна вимога має відповідне покриття в системі та тестах.

Для результативності трасування вимог, побудовано дві діаграми:

Рис №2.19 Відображає співвідношення між системними вимогами, функціональними прецедентами та реалізаційними компонентами бази даних. Та друга (Див рис №2.20), що відображає додаткове трасування між прецедентами, UI- компонентами та сценаріями, що дає змогу провести верифікацію відповідностей функціоналу.

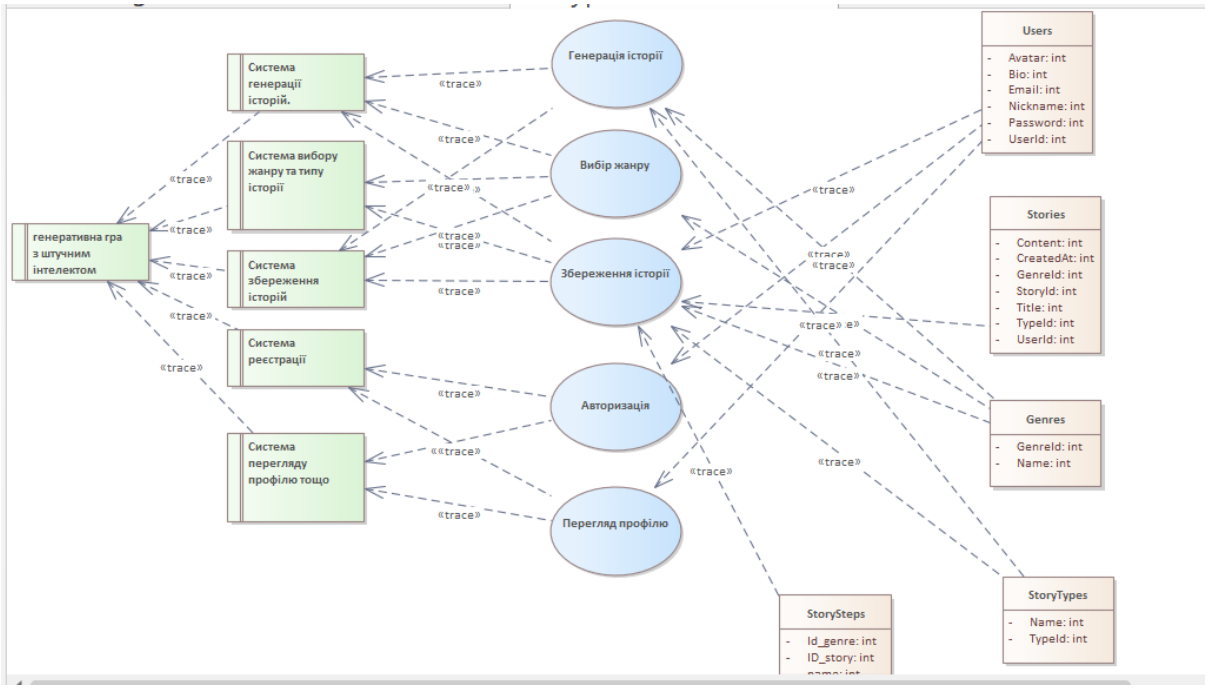


Рисунок 2.19 – Діаграма трасування

Джерело: сформовано автором на основі виконаного дослідження

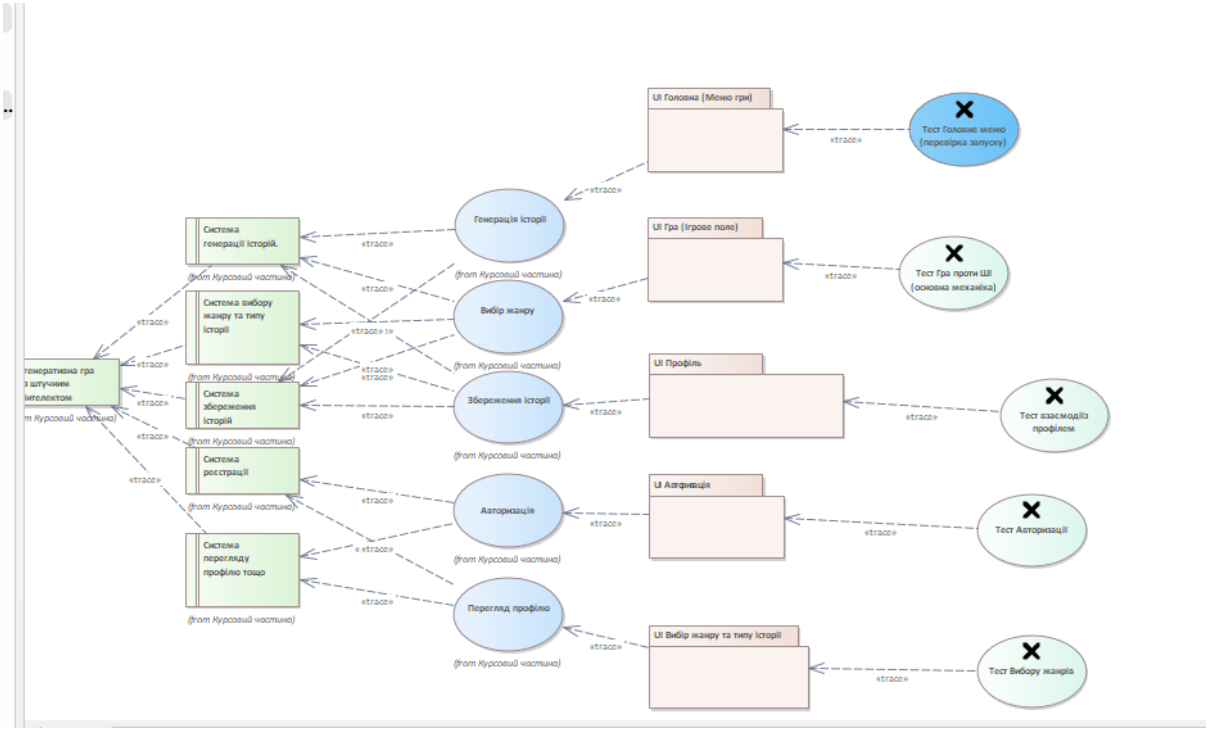


Рисунок 2.20 – Діаграма трасування доповнена

Джерело: сформовано автором на основі виконаного дослідження

Згідно створеним діаграмам було побудовано чотири матриці взаємозв’язків. Що дозволило детально проаналізувати діаграми, забезпечити верифікацію системної моделі, а також відстеження реалізації вимог.

Матриця відповідності варіантів використання функціональним вимогам на рис№2.21 відображає трасування між основними прецедентами та

вимогами, демонструє реалізацію функціональних вимог в кожному варіанті використання.

Матриця відповідності варіантів використання класам системи на рис№2.22 пов’язує прецеденти з класами UML. Показує, які саме класи беруть участь у реалізації кожного варіанту використання.

Матриця відповідності варіантів використання інтерфейсним елементам (пакетам UI) на рис№2.23 встановлює зв’язок між варіантами використання та пакетами інтерфейсів.

Матриця повного трасування між варіантами використання і всіма типами компонентів на рис№2.24, це універсальна матриця дозволяє здійснити повну перевірку відповідностей системи вимогам, виявити дублювання чи відсутність потрібних частин.

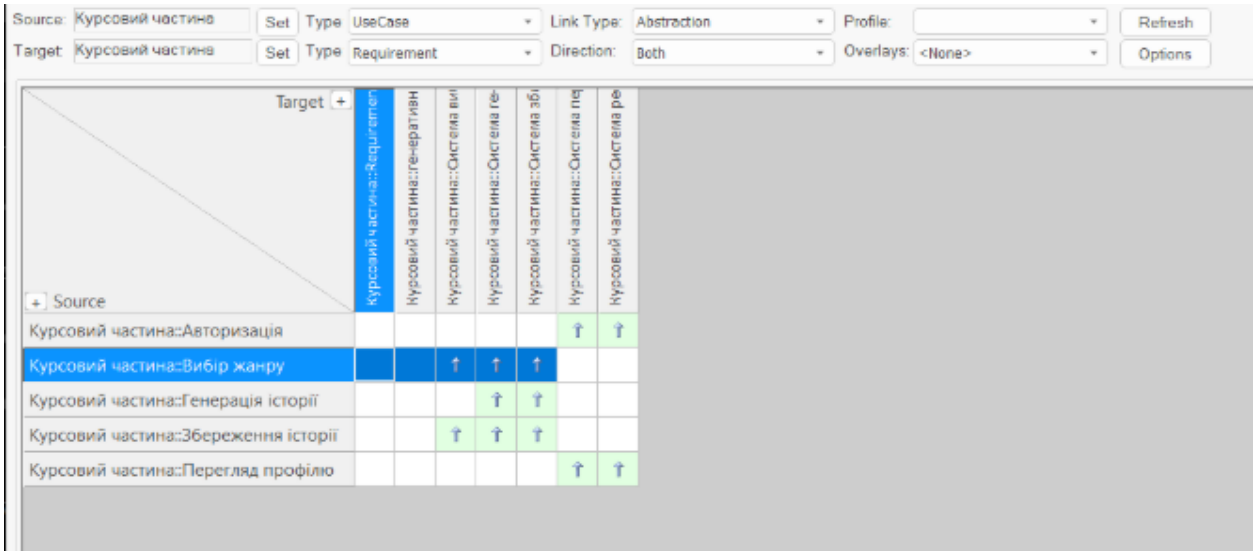


Рис №2.21 - Матриця відповідності варіантів використання функціональним вимогам

Джерело: сформовано автором на основі виконаного дослідження

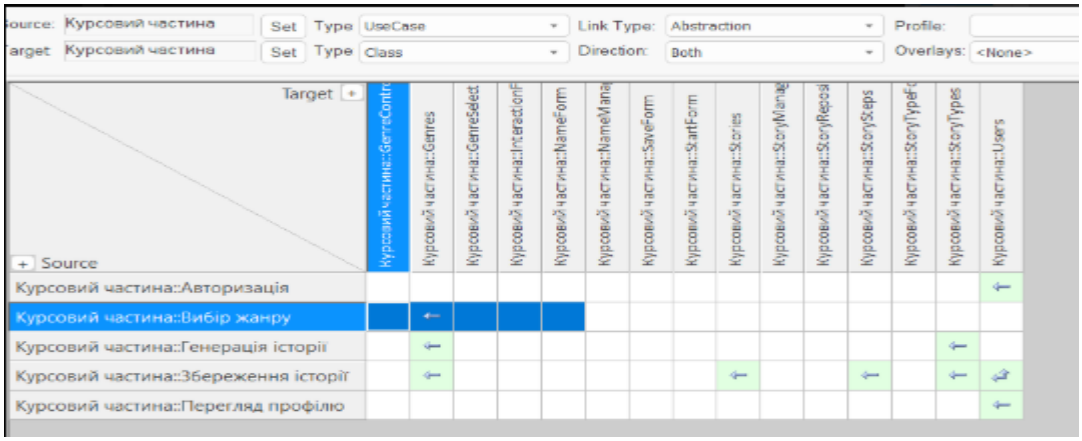


Рис №2.22 - Матриця відповідності варіантів використання класам системи

Джерело: сформовано автором на основі виконаного дослідження

Рис №2.23 - Матриця відповідності варіантів використання інтерфейсним елементам (пакетам UI)

Джерело: сформовано автором на основі виконаного дослідження

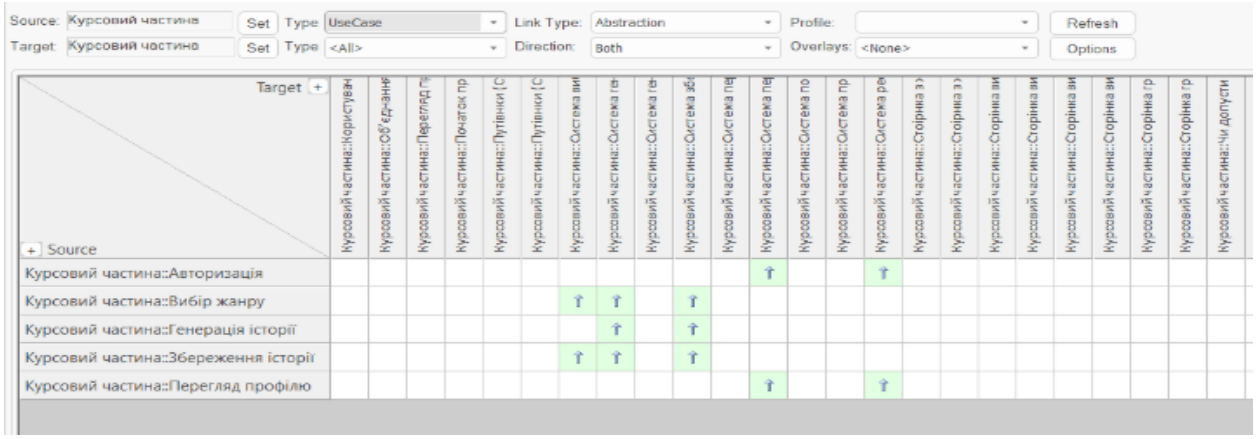


Рис №2.24 - Матриця повного трасування між варіантами використання і всіма типами компонентів

Джерело: сформовано автором на основі виконаного дослідження

2.4 Висновок

У цьому розділу було проведено аналітичне дослідження вимог до створенні інформаційної системи та здійснено багатогранне моделювання її компонентів. Було визначено різні вимоги, що формують загальні цілі проекту. Приділено значну увагу аспектам гнучкості, інтерактивності та персоналізації проекту, що є критично важливими характеристиками.

На основі поставлених вимог, побудовано класифікаціію та їх трасування до відповідних функціональних модулів, що дало можливість чітко все перевірити та уникнути дублювання. Під час аналізу було враховано різні показники та обмеження, що дозволило сформувати реалістичну архітектуру для системи.

Сформульовано задачу проекту, визначено ключові об’єкти, вхідну та вихідну інформацію, побудовано алгоритми системи та їх взаємодію. Також відбулася візуалізація інформації за допомоги діаграм, що забезпечило прозорість проектування та дозволило розробити етапи розробки.

Отже, на основі результатів сформована основа для реалізації проекту, тестування системи. Всі етапи моделювання тісно пов’язані з попередньо сформульованими вимогами, що відповідає технічному рішенню.

РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ ПІДСИСТЕМИ

3.1. Інформаційне забезпечення

3.1.1. Загальна характеристика інформаційного забезпечення

Інформаційне забезпечення (ІЗ) є неоціненно важливим компонентом системи, адже воно охоплює сукупність вхідних, проміжних і вихідних інформаційних об’єктів та структур даних. Адже це все використовується у функціонуванні програмного продукту.

До складу ІЗ входять чотири типи даних:

- Вхідні дані – інформація отримана від користувача.
- Проміжні дані – збережені дані системи (промти, кроки історій, дані профілю, тощо).
- Вихідні дані – результат генерації ШІ, минулі історії, різні статуси за запитом.
- Метадані – службова інформація для ідентифікації користувача.

Також до інформаційного забезпечення входять два носії даних, а саме локальні обчислення в браузері користувача та серверну базу PostgresSql для загального, упорядкованого зберігання.

PostgresSql була вибрана для реалізації сховища даних, адже це система з якою вам роботу в минулому. Система PostgresSql представляє з себе надійну СКБД з відкритим кодом, підтримує зовнішні ключі, індекси, JSON-формати та поля, з можливістю масштабування. В системі реалізовується збереження інформації про користувачів та всю інформацію системи яка надходить.

Перевагами системи вважаю: підтримку структурованих і напів структурованих даних, високу швидкість системи, зручний і простий контроль доступу та технічне супроводження.

- Методами контролю достовірності й цілісності інформації виступає :
- Перевірка введених даних на стороні клієнта і сервера.
- Унікальні обмеження

- Реалізація зовнішніх ключів
- Хешування паролів

До структури ІЗ входить: Користувачі, історії, жанри, типи історій, кроки історій, генеровані фрагменти.

Була створена загальна схема функціонування ІЗ, що демонструє основні джерела даних, етапи оброки, використання у системі, масиви та канали передачі результатів користувачеві та приблизні дії користувача у деяких випадках. Схему дивитись на рис №3.1

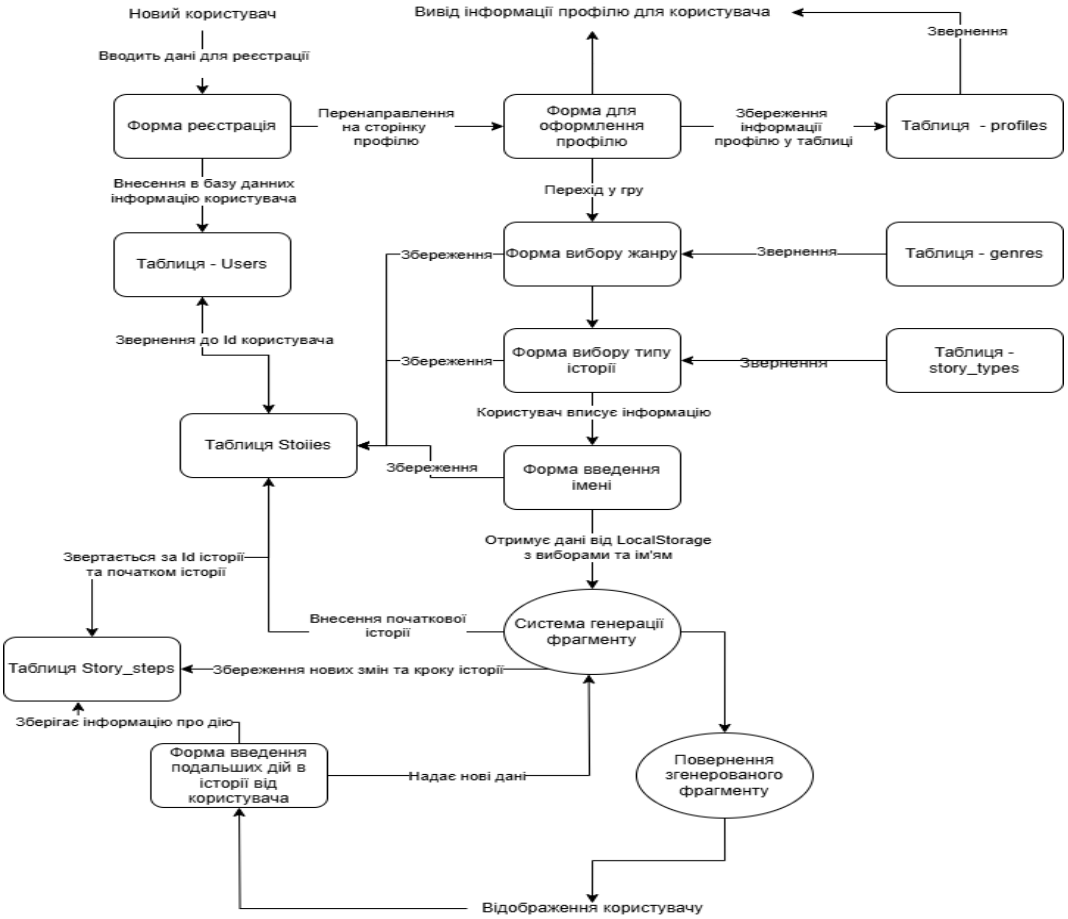


Рис №3.1 - Загальна схема функціонування інформаційного забезпечення

Джерело: сформовано автором на основі виконаного дослідження

3.1.2. Організація збору і передавання первинної інформації

У розробленій системі з використанням ШІ первинна інформація надходить безпосередньо від користувача системи через веб інтерфейс. Інформація що була введена передається системі з метою збереження,

оновлення даних та формування запитів до генеративної моделі або подальшого виведення.

Джерела та носії первинної інформації представлені у таблиці №3.1:

Таблиця №3.1 первинна інформація

Джерело	**Тип носія**	**Дані, що подаються**
Користувач	Веб-форма	Дані реєстрації (логін, email, пароль)
Користувач	Веб-форма	Дані профілю (нікнейм, аватар, опис профілю)
Користувач	Веб-форма	Вибір жанру
Користувач	Веб-форма	Вибір типу історії
Користувач	Веб-форма	Ім’я персонажа
Користувач	Веб-форма	Дії в історії у формі тексту (репліка, команда, дія)
Система LocalStorage	Локальний кеш	Проміжні вибори, які використовуються при генерації
Користувач	Веб-форма	Редагування профілю(нікнейм, опис аватар)

Джерело: сформовано автором на основі виконаного дослідження

В організація збору інформації виступають:

- Інструмент збору – вебінтерфейс React, у якому реалізовані форми з полями вводу.
- Механізм передачі – HTTP запити для відправлення даних на сервер.
- Формат подання – Json – структури.
- Перевірка – здійснюється валідація на фронтенді і бекенді.
- Проміжне збереження – використовується LocalStorage для зручності.

Періодичність та способи передачі інформації :

Спосіб передачі даних у системі реалізовано через HTTP-запити типу Post, PUT, PUTHC та GET, які відправляють з фронтенду до серверної частини форматі JSON. Усі запити проходять через API, для перевірки даних та збереження інформації за певним користувачем. Періодичність на кожну дію дивитись у таблицю №3.2**:**

Таблиця 3.2 Події та їх періодичність

Дія / Подія	**Періодичність**
Реєстрація	Одноразово при створені облікового запису
Авторизація	Кожного разу при вході
Оновлення профілю	За потреби користувача
Створення історії	Кожного разу при створені гри
Кроки історії	Після кожної взаємодії користувача
Завантаження історії	За запитом користувача
Збереження історій	Після кожного кроку історії

Джерело: сформовано автором на основі виконаного дослідження

Передана інформації використовується для формування запитів до ШІ, для збереження у відповідних таблицях БД, для відновлення прогресу минулих історій, та редагування профілю.

3.1.3. Побудова системи класифікації та кодування

У рамках системи для ефективного зберігання, пошуку, оброки та взаємодії з даними реалізовано систему класифікації та кодування в таблиця БД. Класифікування відбувається та забезпечується за ознаками: жанр, тип історії, кроки, користувач та інші. Система кодувань використовує унікальну ідентифікацію кожного об’єкта в системі, зо зменшує обсяг переданих і збережених даних, забезпечує простоту зв’язності і контроль цілісності. Таблицю класів об’єктів і використані класифікатори дивитись у табл №3.3.

Таблиця 3.3 Класи об'єктів і використовувані класифікатори.

Об’єкт класифікації	**Класифікатор**	**Призначення**
Користувачі	Власний Id	Унікальна ідентифікація користувача
Жанри історії	Власний Id	Визначення жанру для класифікації історій
Типи історії	Власний Id	Визначення типу сюжету в межах жанру
Історії користувача	Власний Id	Ідентифікація кожної створеної історії
Кроки історій	Власний Id	Визначення кожного кроку/фрагмента в історії

Джерело: сформовано автором на основі виконаного дослідження

Структуро кодів представляє з себе реалізацію ідентифікатора як ціле число зі зростанням, що забезпечує унікальність кожного запиту та зв’язки між таблицями. Для зв’язків між таблицями використовуються зовнішні ключі. Обмеження на значення задаються відповідно до логіки бізнес-правил, має бути унікальність, що запобігає дублюванню. Довжина коду значень 32 біти, а тип кодування Integer.

Прикладом такого коду є: user.id = 5 -унікальний користувач або genres.id = 5 – це жанр фентезі. Приклади у розробленій базі даних наведено у **додатку В**

3.1.4. Проектування форм первинних документів, машинограм та відеокадрів

В проекті форми введення та виведення первинної інформації, що забезпечує взаємодію користувача з системою на різних етапах: реєстрація, створення історій, генерацій часини історії, керування профілем. Дивитись таблицю №3.4

Таблиця №3.4 - Основні типи форм

№	Назва форми	Ідентифікатор	Тип документа	Користувачі	Періодичність використання	Призначення
1	Форма реєстрації	FormRegister	Ввідна форма	Нові користувачі	Один раз (при вході)	Введення email та пароля
2	Форма авторизації	FormLogin	Ввідна форма	Існуючі користувачі	Після запуску додатку	Доступ до збережених історій
3	Форма створення профілю	FormProfile	Ввідна форма	Всі користувачі	Один раз або при зміні даних	Заповнення профілю: ім'я, опис, аватар
4	Вибір жанру та типу історії	FormGenreSelect	Ввідна форма	Всі користувачі	Перед кожною новою історією	Вибір жанру, типу сюжету, назви персонажа
5	Генерація стартової історії	StoryStart	Машинограма	Всі користувачі	Один раз на історію	Вивід тексту, сформованого ШІ
6	Введення дій користувача	StoryStepInput	Ввідна форма	Всі користувачі	У кожному кроці історії	Введення дій/вибору користувача
7	Виведення згенерованого фрагменту	StoryStepOutput	Вивідна форма	Всі користувачі	У кожному кроці історії	Показ фрагменту, створеного ШІ
8	Вивід історій у профілі	ProfileView	Вивідна форма	Всі користувачі	У профілі	Перегляд списку всіх збережених історій

Джерело: сформовано автором на основі виконаного дослідження

Вимоги до форми мають жорсткі критерії які повинні бути у проекту. Основними вимогами є: уніфікованість інтерфейсу, достовірність введення інформації, автоматичне збереження та машинограми (результат дій користувача та текстові повідомлення).

Для електронної взаємодії в проекті реалізовано вебінтерфейс. Всі форми розміщаються на клієнтській частині, а обмін інформації відбуваються через API- програмний інтерфейс з підключенням бази даних PostgreSql. Вся передача даних відбувається у форматі JSON. Сама авторизація до цього інтерфейсу відбувається за допомогою токенів доступу.

Усі форми, що створені, для проекту мають графічне представлення у додатку Б у вигляді скріншоту інтерфейсу. Наприклад, форми вибору жанру

та типу історії із випадаючого списку, що дозволяє робити вибір користувачу, також з'являється форма з ім'ям яке користувач вводить власноруч. Інтерфейси побудовані логічним чином для легкого розуміння та мінімізації дій з системою.

3.1.5. Структура інформаційних масивів

Інформаційні масиви в розробленій системі формуються на основі таблиць бази даних PostgreSQL. Кожен масив реалізований у вигляді логічно зв'язаних таблиць, які забезпечують повний цикл обробки та зберігання даних: від реєстрації користувача до збереження згенерованих історій та їх кроків. Структура масивів визначена проектною моделлю й відповідає принципам нормалізації, забезпечуючи мінімальну надмірність даних та підтримку цілісності зв'язків. Нижче подано таблицю №3.5**,** у якій детально представлено опис кожного інформаційного масиву.

Таблиця №3.5 - Структура Інформаційних масивів

Найменування масиву	**Найменування Даних та індефікатор**	**Формат**	**Первинний\ вторинний ключ**	**Обов'язкове поле**	**Індексне поле**	**Логічні чи семант ичні зв'язки**
Users	id	int	PK	Так	Інд	
Users	email	varchar		Так	Інд	
Users	password_hash	varchar		Так		
Users	created_at	timestamp		Так		
Profiles	id	int	PK	Так	Інд	users
Profiles	user_id	int	FK	Так		
Profiles	nickname	varchar		Hi		
Profiles	bio	Varchar		Hi		
profiles	avatar_url	Varchar		HI		
Profiles	followers_count	int		Так		
Profiles	following_count	int		Так		
Profiles	friends_count	int		Так		
Profiles	updated_at	timestamp		Так		
Stories	id	int	PK	Так	Інд	users
Stories	user_id	int	FK	Так		Genres
Stories	genre_id	int	FK	Так		Story_types
Stories	story_type_id	int	FK	Так		
Stories	title	Varchar		Так		
Stories	start_text	Text		Hi		
Stories	created_at	timestamp		Так		
story_steps	id	int	PK	Так	Інд	stories
story_steps	story_id	int	FK	Так		
story_steps	step_number	int		Так		
story_steps	player_action	Varchar		Hi		
story_steps	generated_text	Text		Hi		
story_steps	created_at	timestamp		Так		
Genres	id	int	PK	Так	Інд	
Genres	name	Varchar		Так		

Найменування масиву	Найменування Даних та індефікатор	Формат	Первинний\ вторинний ключ	Обов'язкове поле	Індексне поле	Логічні чи семант ичні зв'язки
Genres	description	Text		Ні		
story_types	id	int	PK	Так	Інд	genres
story_types	name	Varchar		Так		
story_types	genre_id	int	FK	Так		
story_types		Text		Ні		

Джерело: сформовано автором на основі виконаного дослідження

3.1.6. Вибір системи керування базами даних (СКБД)

Під час розробки веб-застосунку було використано сучасну об'єктно-реляційну систему з відкритим вихідним кодом PostgreSQL, що є системою керування базами даних. Вона надає потужні інструменти для роботи з даними, дотримується стандарту SQL, підтримує транзакційність, зовнішні ключі, перевірку цілісності, користувацькі функції, індекси, роботу з напівструктурованими даними (JSON, JSONB) та багатокористувацький режим з високою паралельністю запитів. PostgreSQL активно підтримується світовою спільнотою з 1986 року і має відмінну документацію [8].

У рамках даного проекту база даних [10, 11, 14] була розгорнута на хмарній платформі Railway, яка забезпечує стабільний хостинг PostgreSQL без необхідності локального налаштування серверного середовища. Railway підтримує швидке розгортання, зручний графічний інтерфейс для управління базою, автоматичне резервне копіювання та доступ з будь-якого середовища розробки. Такий підхід забезпечив ефективну командну роботу та доступність даних під час усіх етапів розробки — як з локального середовища, так і при інтеграції з вебінтерфейсом застосунку [7].

Замість інших популярних рішень, зокрема таких як MySQL, була обрана Система PostgreSQL. Ключовими причинними вибору стали: саме PostgreSQL є рекомендованою системою керування базами даних для попередньо обраного фреймворку Django (усі основні функції Django – система об'єктно-реляційного відображення, міграції, агрегації, фільтрація тощо найбільш повно підтримуються саме PostgreSQL); PostgreSQL дозволяє легко масштабувати систему в майбутньому та демонструє стабільну роботу з великими обсягами даних та великою кількістю одночасних запитів;

PostgreSQL повніше дотримується офіційного стандарту мови запитів SQL, ніж MySQL, у якій деякі механізми реалізовані частково або з обмеженнями (наприклад, обмеження на підтримку зовнішніх ключів у таблицях) [8].

За аналітичним порівнянням [9], PostgreSQL суттєво переважає MySQL у категоріях розширюваності, аналітичних можливостей та точності виконання складних запитів.

Використання у проекті хмарного рішення Railway додатково дозволило підвищити доступність під час розробки та скоротити час на налаштування та розгортання бази, а також автоматизувати інтеграцію з серверною частиною застосунку. Адже саме Railway підтримує безперервну роботу в мережі та резервне копіювання та надає готові параметри підключення до бази, що відповідає сучасним стандартам DevOps-підходів.

Використання PostgreSQL у поєднанні з хмарною платформою Railway для зберігання даних програмного продукту забезпечило масштабовану, стабільну та надійну основу. З точки зору функціональності та продуктивності, так і з точки зору безпеки, підтримки та сумісності з іншими компонентами системи це рішення було найоптимальнішим.

3.1.7. Інфологічна модель бази (сховища) даних

Інфологічна модель це логічне подання структури інформації. Ця модель ґрунтується на сутностях предметної області, атрибутах та зв’язках між ними. Метою моделі є формування структури даних до вигляду таблиць реляційної бази даних.

Для побудови інфологічної моделі використовувався підхід «сутність зв’язок», що дозволило описати систему за допомогою таблиць, атрибутів цих таблиць та зв’язків. Модель розроблена відповідно до існуючої бази даних вебзастосунку.

Розробка відбувалась в програмному середовищі DBeaver, яке зручне для візуалізації інформації та автоматичної побудови ER-діаграм на основі створених таблиць в PostrgresSQL. Для ознайомлення з структурою бази даних Див рис 3.2.

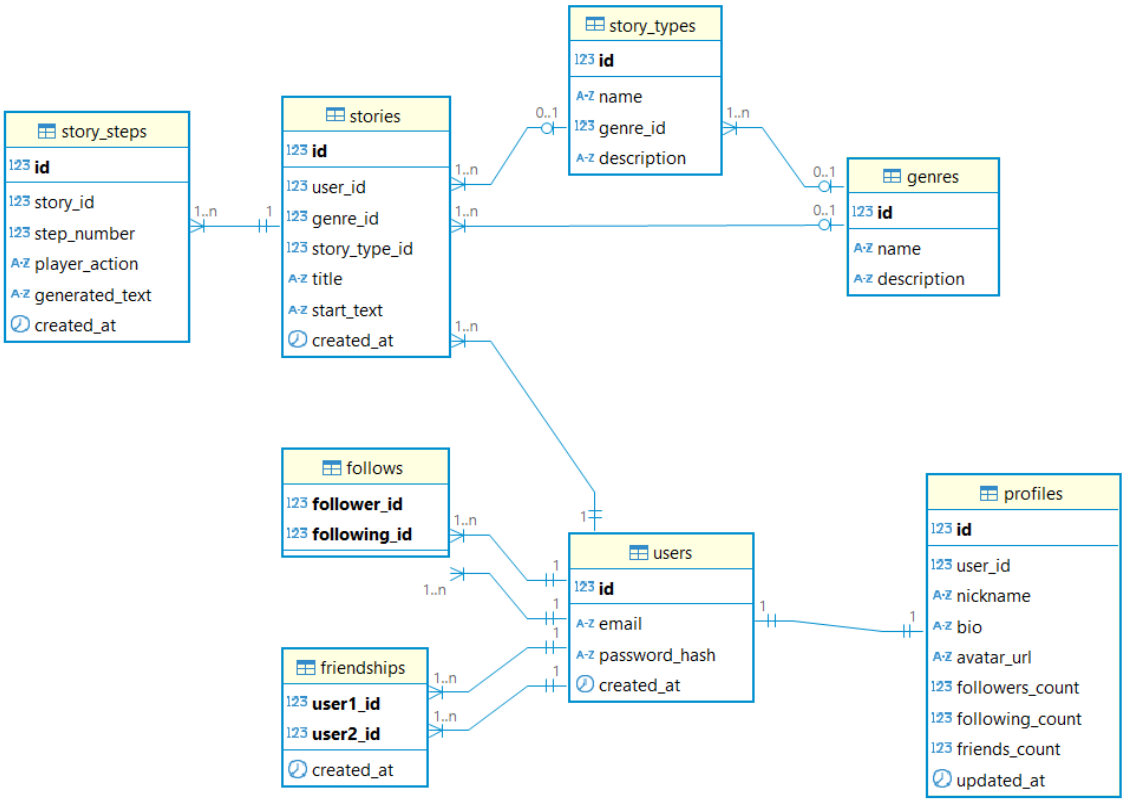


Рисунок 3.2- Інфологічна модель бази даних

Джерело: сформовано автором на основі виконаного дослідження

3.1.8. Даталогічна модель бази (сховища) даних

Даталогічна модель представляє фізичне подання структури бази даних, що показує саме реалізацію таблиці, які поля, типи даних, ключі, обмеження використовуються в керуванні БД. На відміну від інфологічної моделі яка є логічною структурою, ця модель представляє конкретну реалізацію в середовищі PostgreSQL. Для створення даталогоічної моделі використовувався також DBeaver та розгорнутої хмари на Railvay для зручного керування базою.

Для розуміння основних характеристик реалізації дивитись таблицю №3.6. Також приклад реалізації представлено на рисунку №3.3 та всю реалізацію та код в додатку №В

Таблиця 3.6 - Основні характеристики реалізації.

Primary Key	Усі таблиці мають чітко визначені первинні ключі.
Foreign Key	Для забезпечення зв'язків між таблицями використовуються зовнішні ключі.
INT, TEXT,VARCHAR, TIMESTAMP	Чітко визначені типи даних.
NOT NULL, UNIQUE	Визначені обов'язкові поля та унікальні обмеження.
Індексація	Для певних полів реалізована індексація для швидкого пошуку.

Джерело: сформовано автором на основі виконаного дослідження

```
CREATE TABLE public.genres (
    id serial4 NOT NULL,
    "name" varchar(100) NOT NULL,
    description text NULL,
    CONSTRAINT genres_name_key UNIQUE (name),
    CONSTRAINT genres_pkey PRIMARY KEY (id)
);
```

Рис 3.3 - Реалізація таблиці жанрів.

Джерело: сформовано автором на основі виконаного дослідження

3.2 Технічне забезпечення

3.2.1 Загальні положення та схема автоматизації

Інформаційна система проекту – це інтерактивний застосунок, що включає та поєднує технологій ШІ, хмарних обчислень і кросплатформеного вебінтерфейсу для створення історій з динамічним сюжетом за участі користувача. У фокусі автоматизації процесів перебуває повний цикл від авторизації, генерації історій до збереження та персоналізації під потреби.

Архітектурною особливістю системи є мережевий клієнт-серверний підхід, що дає змогу обчислювальній логіці та керуванню даними відбуватися на стороні серверу, а клієнтська частина виконує роль інтерфейсу з яким взаємодіє користувач. За цих умов модель дозволяє досягти високої масштабованості, управління даними та безперебійного оновлення функціоналу без втручання користувача.

Серверна частина розміщена у хмарному середовищі Railway, що дозволило безперервну доступність, ізоляцію середовищ і підтримку різних процесів. На серверній частині розгорнути чотири основні компоненти:

- Логіка обробки запитів користувача.
- Зберігання історій, інформації профілю та іншої інформації у БД.
- Взаємодія з API.
- Обробка станів системи.

Користувацький доступ реалізован через вебінтерфейс, без повноцінної програми та інсталяції. Користувачу достатньо мати лише браузер з підтримкою JS. Це дозволяє користуватися проектом практично з будь-якого пристрою, незалежно від ОП системи. Основна умова – це мати доступ до стабільного інтернету.

Автоматизація полягає в тому, шо всі основні процеси, реалізуються повністю або частково в деяких випадках за допомогою ШІ, а не як в традиційних системах, вимагали ручного написання. Система формує нові сюжетні повороти, адаптує сценарій під дії користувача, зберігає прогрес і забезпечує розвиток подій без ручного адміністрування.

Умовна схема автоматизації включає наступні вузли:

- Клієнт – взаємодія через браузер.
- Інтернет – канал передачі даних.
- Хмарний сервер – обробка запитів, збереження, API запити.
- ШІ – генерація контенту.
- База даних –збереження інформації.

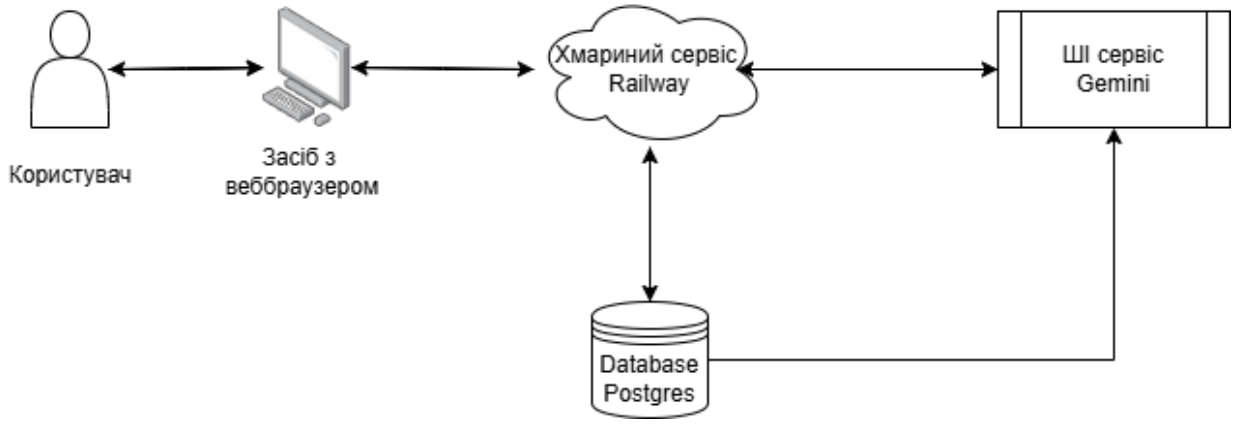


Рис №3.4 - візуалізація схеми автоматизації проекту

Джерело: сформовано автором на основі виконаного дослідження

Ця конфігурація дозволяє легко зрозуміти структуру роботи, масштабувати систему під зростаючу кількість користувачів, але й забезпечити гнучкість – наприклад підключення нових ШІ-сервісів, нові запити або зміну інтерфейсу.

3.2.2 Структура комплексу технічних засобів

Комплекс технічних засобів (КТЗ), що забезпечує функціонування системи, базується на концепції централізованої обробки даних у середовищі з доступом до інтернету. Такий підхід дозволяє мінімізувати витрати на фізичну інфраструктуру і досягти високого рівня доступності

Обґрунтування вибору структури КТЗ:

Архітектура клієнт-сервер була обрана, адже є найбільш придатною для реалізації інтерактивного веб застосунку з використання ШІ. Основні процеси проходять на сервері, тоді як користувач працює тільки з інтерфейсом в браузері.

Чотири основних компоненти КТЗ – це хмарний сервер, база даних, клієнт з власним пристроєм, ШІ-сервіс. Детальне пояснення компонентів, технічних характеристик , безпеки та захисту системи представлено в таблиці №.3.7

Таблиця №3.7 - Основні компоненти КТЗ

Основні компоненти КТЗ	
Хмарний сервер	розміщений на платформі Railway або аналогічному хостингу. Він виконує роль основного обчислювального вузла, приймає запити від клієнтів, взаємодіє з базою даних та ШІ-сервісами.
База даних PostgreSQL	використовується для зберігання згенерованих історій, дій користувачів, їх профілів, прогресу, жанрових уподобань тощо.
Клієнтські пристрої	персональні комп'ютери, ноутбуки, планшети та смартфони користувачів, які підключаються до системи через браузер. Жодне додаткове апаратне забезпечення з боку користувача не потрібне.
ШІ-сервіс GEMINI	зовнішній API, до якого сервер надсилає запити на генерацію історій, описів, варіантів дій тощо.
Технічні характеристики	
Хостинг	Node.js backend з підключенням до PostgreSQL, доступ через HTTPS.
Кількість АРМ	система розрахована на необмежену кількість одночасних користувачів, але первинна інфраструктура проекту підтримує до 100 паралельних сесій без втрати продуктивності.
Масштабованість	завдяки використанню хмарної інфраструктури можливе динамічне масштабування ресурсів.
Безпека та захист	
Шифрування	вся передача даних здійснюється через HTTPS-протокол.
Захист від втрат	автоматичне резервне копіювання даних на рівні Railway і PostgreSQL.
Контроль доступу	облікові записи, доступ до яких захищений паролем; підтримується автентифікація через токени (JWT).

Джерело: сформовано автором на основі виконаного дослідження

Оскільки система проекту працює в хмарному середовищі та загалом автоматизована, потреба в постійному залучені обслуговуючого персоналу мінімальна. Для технічного супроводу достатньо одного адміністратора системи – розробника, який відповідатиме за моніторинг та оновлення застосунку. Такий випадок дозволяє знизити витрати й забезпечити роботу системи за обмеженого людського ресурсу.

3.2.3 *Опис автоматизованого робочого місця (АРМ)*

Система розроблена з урахуванням принципів доступності, простоти в користування, кросплатформеності, то автоматизоване робоче місце користувача не потребує апаратного чи програмного забезпечення. Достатньо мати пристрій з доступом в мережу інтернет та веббраузер з підтримкою JS, що забезпечить доступ з будь-якої точки світу. Мінімальні вимоги пристрою та підключення мережі дивитись табличку №.3.8

Таблиця №3.8 - мінімальні потреби

Мінімальні системні вимоги до АРМ користувача	
Операційна система	Windows 10+, macOS 10.13+, Android 8+, iOS 12+
Процесор	Intel Core i3 або ARM-еквівалент
Оперативна пам’ять	від 4 ГБ
Накопичувач	1 ГБ вільного простору для кешу браузера та тимчасових файлів
Браузер**:**	Google Chrome, Mozilla Firefox, Safari або Edge останніх версій
Інтернет-з'єднання	постійне, зі швидкістю не менше 5 Мбіт/с
Підключення до мережі	
Wi-Fi	**Wi-Fi** (802.11n/ac
Інтернет	Ethernet
Мобільного інтернету	(4G/5G)

Джерело: сформовано автором на основі виконаного дослідження

АРМ у системі *проекту* може бути відбуватися у кількох формах залежно від ролі користувача та сценарію використання. Основним варіантом є персональний комп’ютер або ноутбук, який підходить для активних користувачів, що працюють з історіями та потребують більш зручного інтерфейсу для введення і редагування контенту. Для звичайних або зайнятих користувачів достатньо мобільні пристроїв — смартфони або планшети, що дає змогу взаємодіяти з історіями у будь-якому місці та у зручний час.

До периферійних пристроїв які потрібні користувачу відноситься: мишка, клавіатура, монітор якщо це власник персонального комп’ютера, якщо користувач використовує ноутбук або телефон, такі пристрої не є обов’язковими.

3.3 Програмне забезпечення

3.3.1 *Структура програмного забезпечення*

Структура проекту охоплює всі рівні, від базового системного забезпечення до прикладних програмних компонентів, що реалізують функціональність генерації контенту та взаємодії з користувачем та інтелектуальним ядром системи. Система базується за принципом модульності, що дає змогу розділити ПЗ за функціональним призначенням. Структуру системи подано на рисунку №3.5.

На першому рівні розміщено системне програмне забезпечення, яке забезпечує стабільність обчислювального середовища. До цього рівня входить: операційні системи що використовуються клієнтами, хмарна система Railway для збереження бази даних, та PosetgreSQL для керування базами даних.

На другому рівні розташоване прикладне ПЗ, що реалізує функціональність проекту. До нього належать: **React** для створення інтерфейсу**, Node.js** для реалізації логіки запитів, зв’язок з базою даних та зовнішнім API, саме **API** яке виконує генерацію контенту.

На третьому рівні зосереджена програмна документація, вона містить опис функціональних можливостей, вимоги до середовища, конфігурації, графіки та діаграми та технічну документацію.

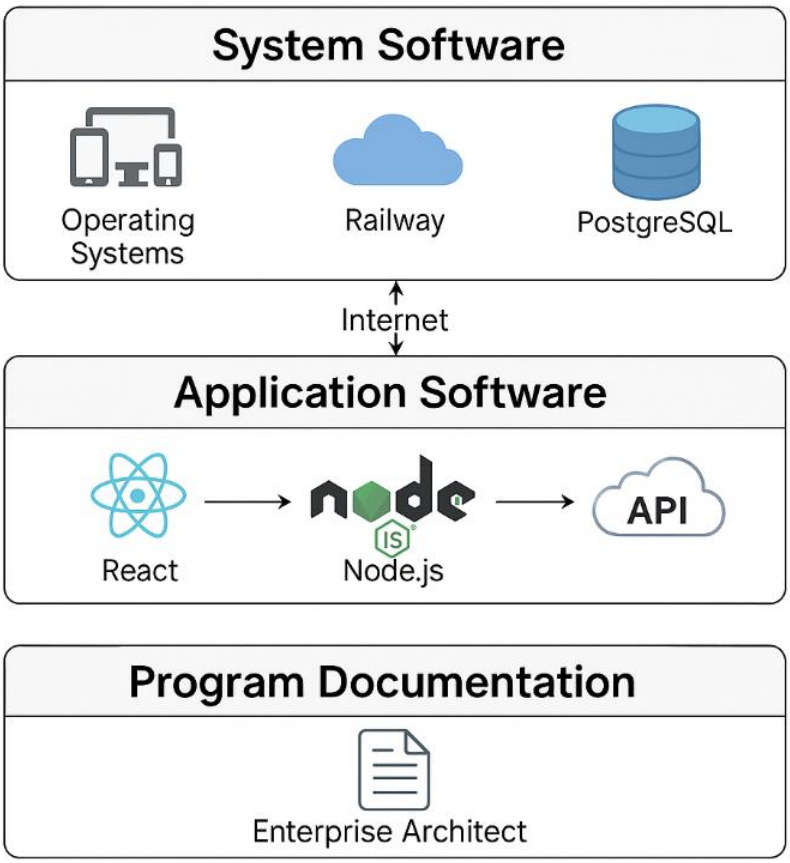


Рисунок 3.5 - Структура програмного забезпечення інформаційної системи

Джерело: сформовано автором на основі виконаного дослідження

3.3.2 *Системне програмне забезпечення*

Системне програмне забезпечення – це основа функціонування проекту. Саме воно забезпечує створення, виконання та підтримку прикладних програм, стабільну роботу компонентів, обробку всіх запитів, зберігання даних, взаємодію з хмарними сервісами та захист інформації. Вибір ПЗ здійснювався на основі власного досвіду, вимог до системи, та зручності у використані. Також було використано деяке забезпечення за поради більш розвинутих користувачів.

Операційні системи потребують можливість кросплатформеності, тому в системі використовується різні операційні системи: для клієнтських пристроїв це може бути будь-яка система windows чи Linux або android, головне мати сучасний браузер. Для серверної частини застосовується Ubuntu Linux адже є частиною Railway. Та для розміщення бази даних використовується хмарне середовище Railway у віртуалізованому середовищі Linux.

Інструменти що швидко дозволяють візуалізувати та формалізувати програмну архітектуру це RAD-засоби та CASE-системи, до них в проекті відноситься та використовувалися Enterprise Architect, pgAdmin, Railway CLI .

У випадку Веб-технологій у проекті використовується власноруч розроблена систем з використанням таких технологій:

- **HTML5, CSS3, JavaScript** — основа клієнтського інтерфейсу;
- **React** — фреймворк для створення SPA (Single Page Application);
- **Node.js + Express** — для побудови серверної частини та реалізації REST API;
- **PostgreSQL** — як основна СКБД;
- **Google Gemini API** — для інтеграції функцій генерації тексту штучним інтелектом.

Також використовується базовий список сервісних програм з якими мав досвід для розробки та тестування системи:

- **Visual Studio Code** — основне середовище розробки;
- **Git + GitHub** — для контролю версій та командної розробки;
- **DBeaver –** для керування базою даних.

Для захисту даних та взаємодії між клієнтом і сервером використовуються:

- **HTTPS** — захищений протокол обміну даними;
- **JWT (JSON Web Tokens)** — для авторизації користувачів;
- **CORS-захист** та перевірка CSRF через middleware на Node.js.

3.3.3 Прикладне програмне забезпечення

Прикладне ПЗ системи становить сукупність взаємозв’язків між програмними модулями призначених для реалізації всіх функціональних вимог системи – від авторизації до керування профілем користувача та генерації історій. Створене ПЗ тісно інтегровано з хмарним середовищем та базою даних, що формує масштабовану та взаємо надійну структуру

компонентів, що забезпечила зрозумілу логіку взаємодії з користувачем і стабільну обробку процесів та логічних зв'язків застосунку. До класифікації ППЗ відноситься:

- ПЗ загального призначення
- Методо-орієнтоване ПЗ
- Проблемно-орієнтоване ПЗ
- ПЗ для роботи в глобальних мережах
- ПЗ для адміністрування обчислювального процесу.

Для ознайомлення та детального опису всіх класифікацій дивитись в таблиці №3.9.

Таблиця №3.9 - Класифікація прикладного ПЗ системи

ПЗ загального призначення
Клієнтський інтерфейс, реалізований у вигляді вебзастосунку на основі фреймворку React. Забезпечує візуалізацію інтерфейсу, навігацію, взаємодію з історією та надсилання вибору користувача на сервер.
Система автентифікації користувачів, що підтримує реєстрацію, вхід, захист даних за допомогою JWT.
Методоорієнтоване ПЗ:
API-шлюз, побудований на базі Node.js + Express, який обробляє запити клієнта, звертається до бази даних PostgreSQL та виконує запити до зовнішнього ШІ-сервісу **Google Gemini**.
М**одуль управління сесіями** — реалізує логіку обліку стану користувача, збереження прогресу та подій, що відбулися в історії.
Проблемно-орієнтоване ПЗ:
Модуль генерації історій, який інтегрується з API Google Gemini та формує сюжетні гілки відповідно до вибору користувача.
Модуль збереження даних, який реалізує логіку взаємодії з PostgreSQL: запис дій, профілю користувача, вибраного жанру тощо.
ПЗ для роботи в глобальних мережах:
Використання HTTPS-протоколу для всіх обмінів між клієнтською та серверною частинами.
Зовнішнє API Google Gemini**,** що викликається через інтернет за захищеним каналом.
ПЗ для адміністрування обчислювального процесу:
Логування подій і запитів на стороні сервера.
Модуль моніторингу продуктивності та статистики сесій (у перспективі).
Адмін-панель (у розробці) — для керування історіями, користувачами та налагодженням API.

Джерело: сформовано автором на основі виконаного дослідження

Вибір інструментів для створення ППЗ проекту здійснювалось на основі концепції RAD, яка передбачає максимально швидке та гнучке створення програмного продукту. Ця концепція передбачає повторне використання компонентів, інтеграція різних середовищ розробки та збільшення автоматизації процесів.

В якості клієнтського середовища було обрано React – популярний фреймворк JavaScript, який дозволяє створювати динамічні інтерфейси користувача за допомогою компонентного підходу. Його переваги: велика спільнота, сумісність із сучасними браузерами, підтримка JSX, гнучке розгортання та проста інтеграція з бекендом через REST API.

Серверна частина реалізована за допомогою Express поверх Node.js — це дозволяє обробляти одночасні запити з мінімальною затримкою, легко масштабувати логіку, створювати гнучкі RESTful API та інтегрувати сторонні бібліотеки для AI, JWT, журналювання, безпеки тощо. Express — це легкий фреймворк, який дозволяє швидко розгорнути серверну частину та налаштувати маршрутизацію, підключення до бази даних, проміжне програмне забезпечення та автентифікацію.

Для зберігання даних було обрано PostgreSQL, який ідеально підходить для структури даних системи, яка містить сутності користувачів, історії, дії, профілі, жанри тощо. PostgreSQL підтримує складні запити, транзакції, поля JSON та індекси, що покращує продуктивність запитів під час великих сеансів.

Особливу увагу приділено інтеграції з API Gemini AI від Google, який може генерувати динамічний текст у режимі реального часу на основі вибору користувача. Усі запити до ШІ відбуваються асинхронно, з обробкою помилок і тайм-ауту.

3.4. Результати реалізації інформаційної підсистеми

За результатами описаних вище пунктів було спроектовано, розроблено та реалізовано інформаційну систему дипломного проекту, яка призначена для створення унікального досвіду користувачів українського ринку, за допомогою створення та управління сюжетними історіями на події в яких користувач може впливати напряму за допомогою власних текстових уточнень або команд які модуль генерації з ШІ обробляє та створює продовження.

Проект реалізовано для онлайн використання, що забезпечує швидкий доступ для кожного користувача в систему з будь-якого пристрою. У якості серверної частини на момент реалізації використовується середовище Node.js, з базою даних PostgreSQL, що розміщена на хмарному середовищі Railway, що забезпечує зручне адміністрування та можливість масштабування. Для

реалізації фронтенду було використано React. На github наведено повний лістинг програмного коду для даної розробки.

Основний функціонал системи було протестовано та доведено до робочого результату.

Для початку взаємодії з веб-застосунком користувач переходить на сайт, де і попадає на головну сторінку, див рис 3.6. На цій сторінці користувач бачить частину інформації для ознайомлення та заблокований профіль і кнопку яка просить увійти у зареєстрований аккаунт. Користувач без реєстрації може перейти для ознайомлення на інші сторінки, а саме жанри та путівник, див рисунки 3.7 та 3.8. Саме на сторінці жанри користувач може зрозуміти які на даний момент часу активні жанри історій та короткий опис до них. На сторінці путівник вже цікавіше, адже ця сторінка ознайомлює користувача з важливою та цікавою інформацією, котра описує початок роботи, основи в які входить: «Вибір власної пригоди» та «Власний внесок в історію», речі які повинен знати користувач та як загалом працює проект. Вибором користувача котрий зацікавився і вирішив спробувати буде сторінка входу у систему, якщо аккаунта не було створено то сторінка реєстрації дивитись рис 3.9 та 3.10. Де користувач вводить власну пошту та паролі для входу або створення аккаунту. П

Після створення користувача автоматично перенаправляє на сторінку профілю дивитись рис 3.11, де користувач власноруч може відредагувати власний профіль, поставити своє зображення, змінити ім’я профілю та опис. В подальшому також побачити список всіх історій які було початі або пройдені нижче.

Для початку гри користувачу потрібно повернутися на головну сторінку, де матиме вже кнопку Почати, дивитись рис 3.12. В результаті натискання кнопки користувач попаде на сторінку вибору жанру історії, вона ідентична сторінці на рис 3.7 - Жанри, лише немає верхнього меню. Після вибору жанру користувач попадає на сторінку вибору типу історії дивитись рис 3.13 , де описані варіанти для більш чіткої класифікації історії, де після вибору типу з’являється вікно для внесення імені персонажа для історії, дивитись рис 3.14,

користувач може написати будь-яке ім’я і воно буде присвоєне головному герою історії.

Під час виборів користувача все зберігається для використання у новій згенерованій історії і впливає на всі концепції які були запрограмовані. Отже користувач потрапляє на сторінку гри , за середнім результатом близько трьох секунд користувач отримує початкову історію з якою повинен ознайомитися, дивитись рис 3.15. В результаті прочитання користувач повинен ввести власні дії продовження історії які будуть використанні для побудови наступної частини і може продовжувати такий цикл вже скільки побажає. Приблизну структуру історій та результатів дивитись на рис 3.16.

Користувач за бажанням може повернутися до профілю через меню по натисканню картинки профілю біля назви сторінки, дивитись рис 3.17. На сторінці профілю він вже матиме власну історію і може її продовжити за натисканням на неї. Та опиниться на новій сторінці де може ознайомитися з минулими кроками які робив раніше.

Також у меню профілю можна вийти з аккаунту за потреби та перейти до налаштування профілю, на момент написання це лише сторінка без чіткої роботи, але є перспективою доопрацювання та реалізації повноцінних функція, як налаштування дозволу на різні рівня контенту, зміну статусу користувача (мандрівник), зміну паролю та пошти за потреби. Рис 3.18

Важливо зазначити перспективи продовження розробки проекту, до них відноситься:

- додавання функції багатокористувацької гри (спільне створення історій);
- реалізація системи досягнень та рейтингів;
- інтеграція з мобільними платформами Android/iOS;
- локалізація на інші мови;
- додавання можливості створення власних жанрів та типів історій (Story Editor);
- використання більш просунутих моделей (наприклад, GPT-4 або Claude AI) для поліпшення контекстуальних відповідей;

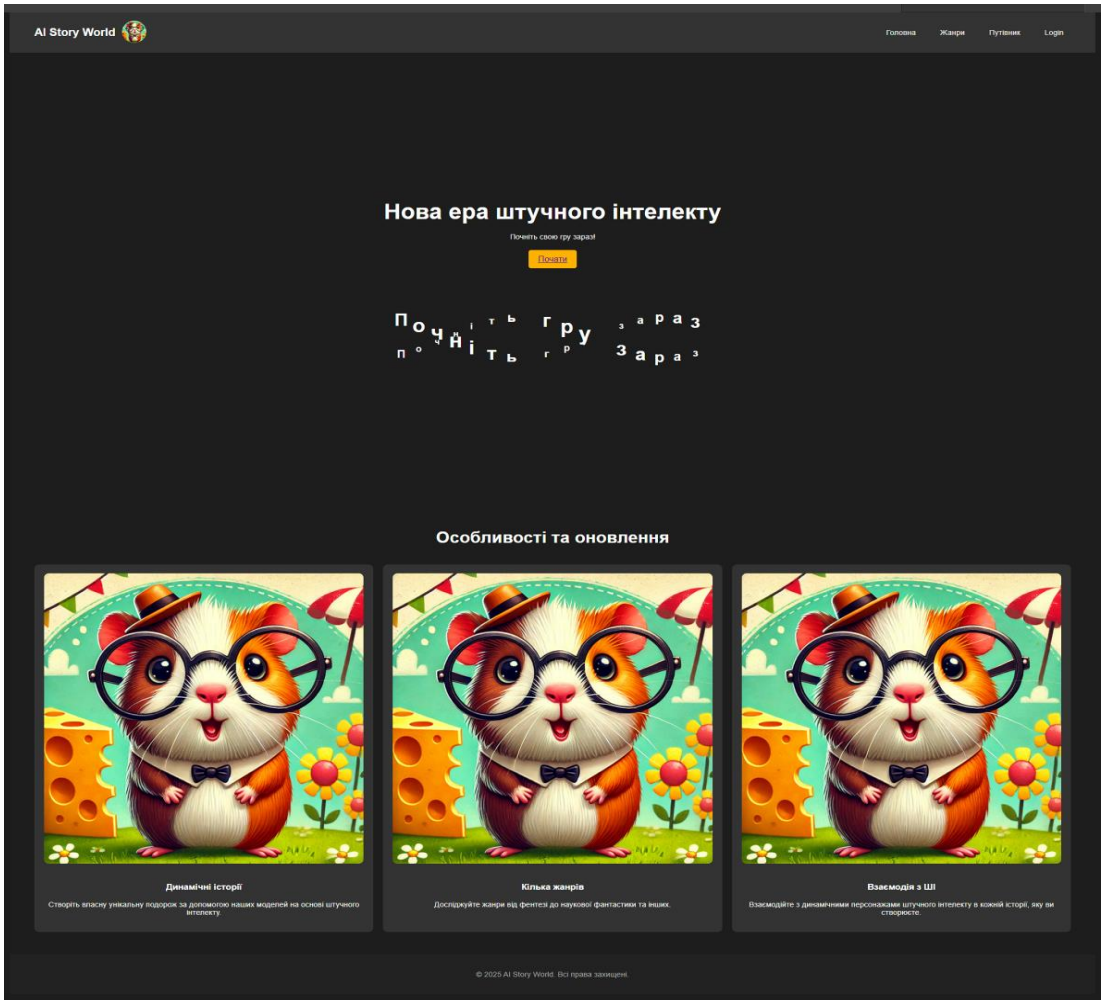


Рисунок 3.6 - головна сторінка

Джерело: сформовано автором на основі виконаного дослідження

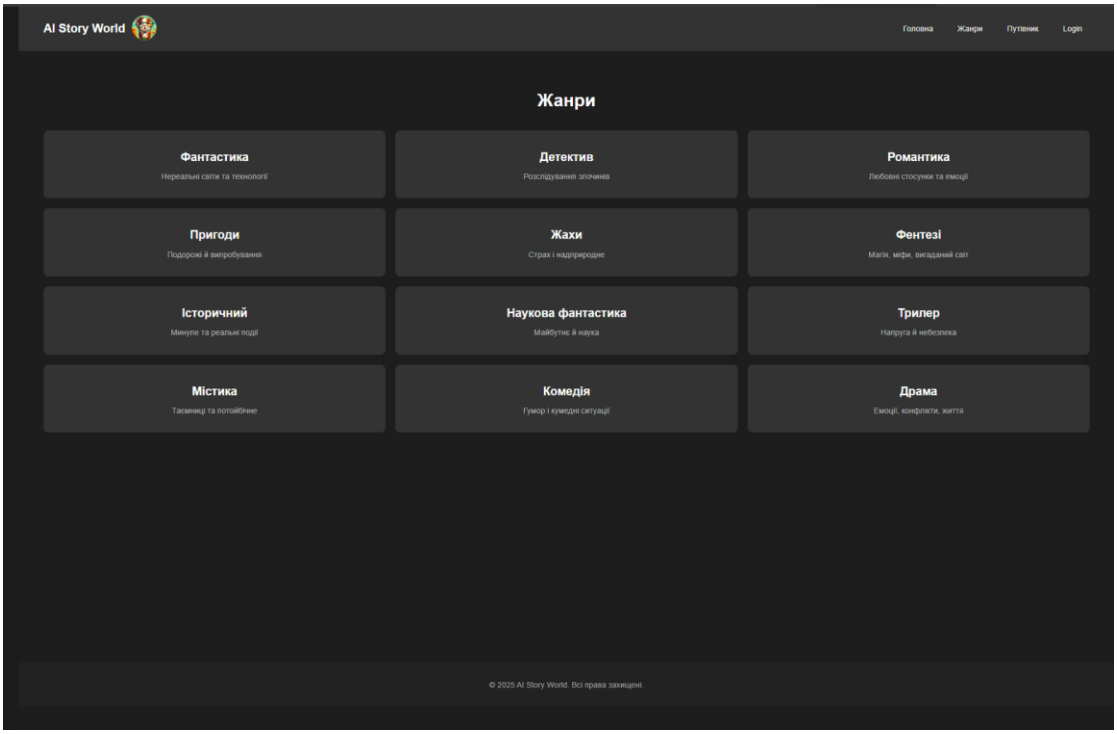


Рисунок 3.7 – Жанри

Джерело: сформовано автором на основі виконаного дослідження

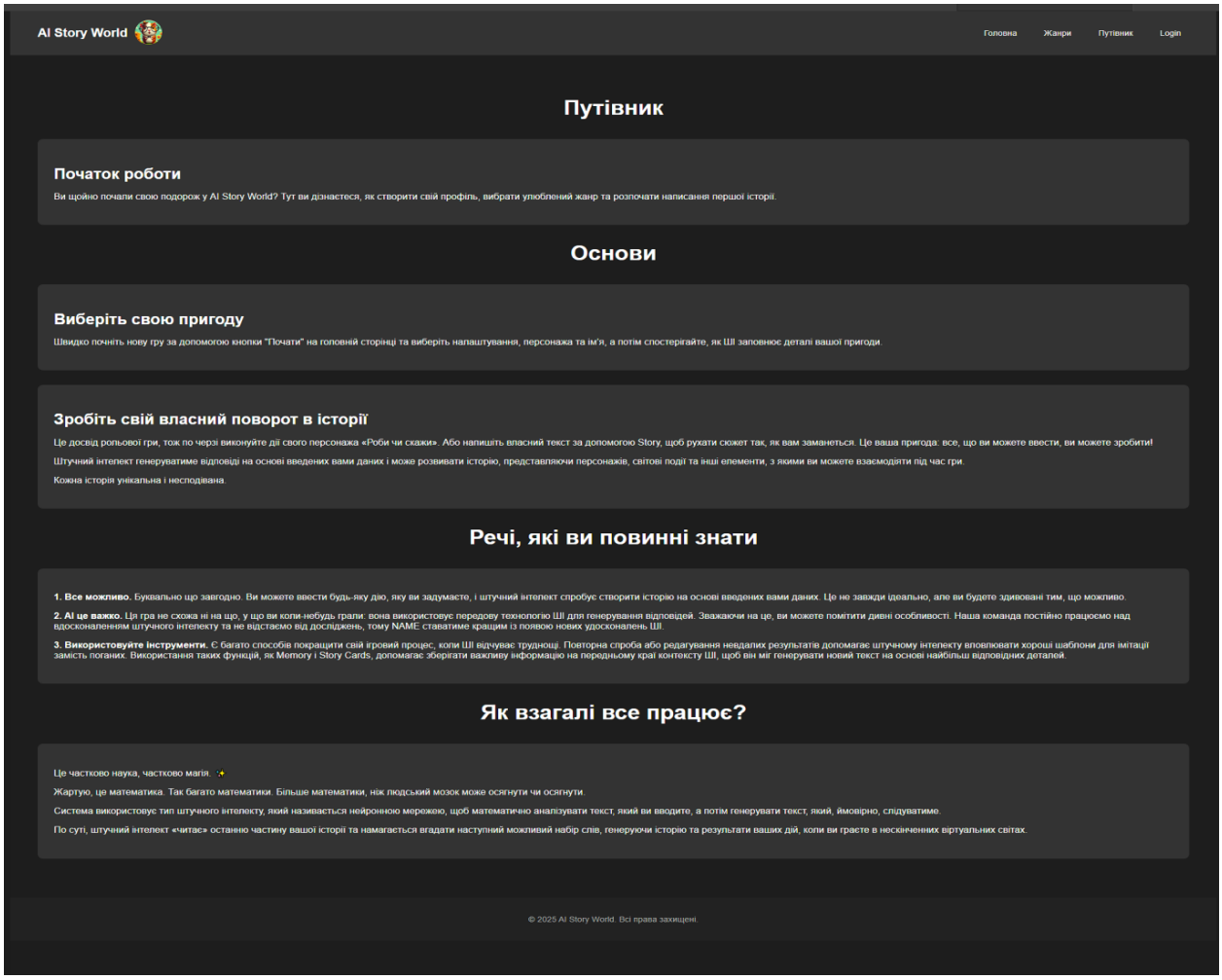


Рисунок 3.8 – Путівник

Джерело: сформовано автором на основі виконаного дослідження

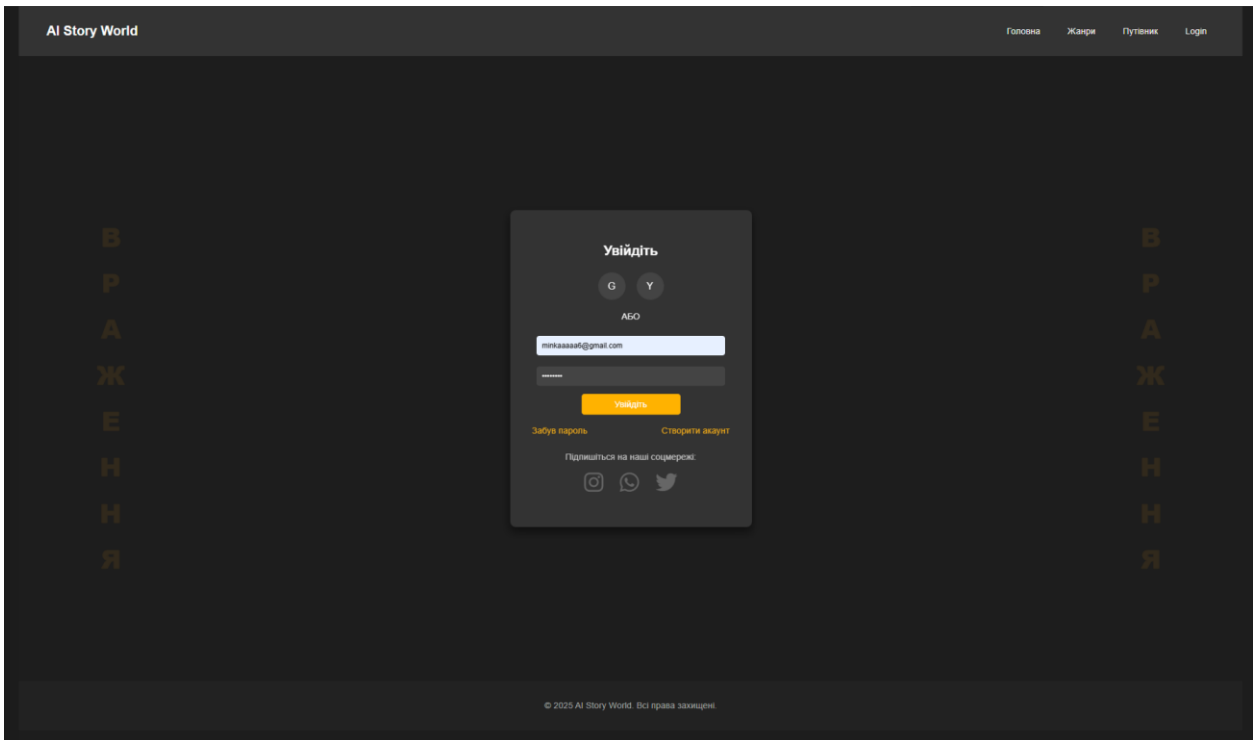


Рисунок 3.9 - Вхід в акаунт

Джерело: сформовано автором на основі виконаного дослідження

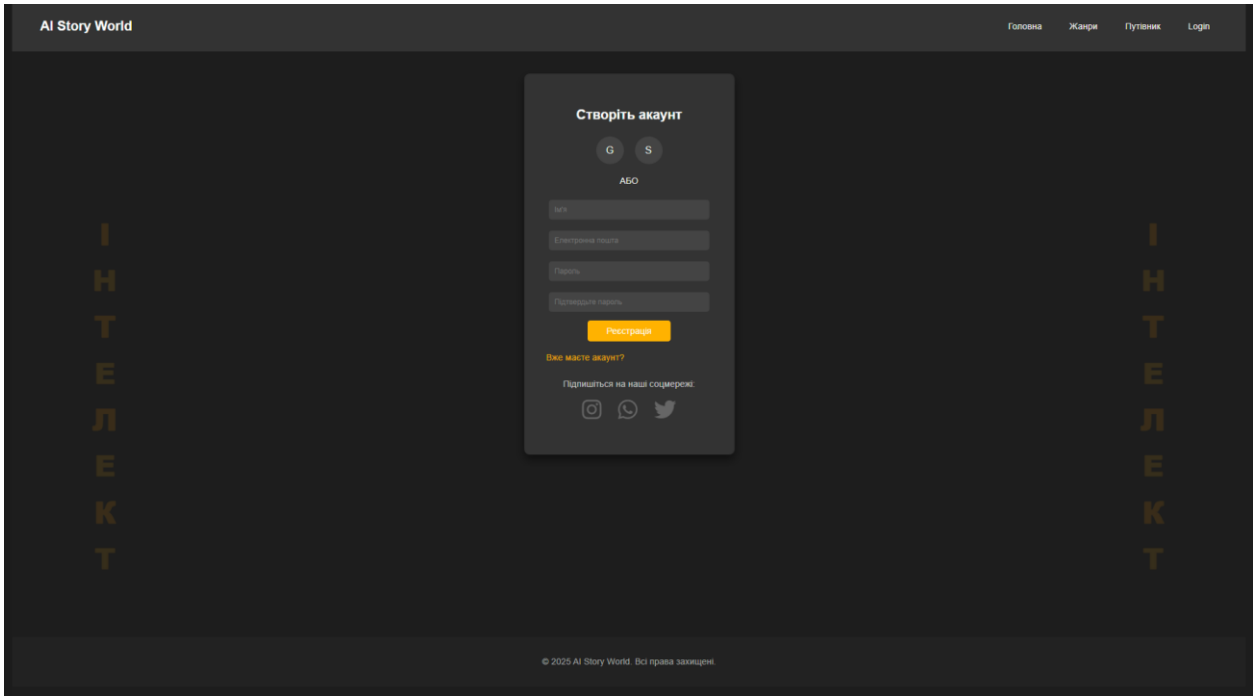


Рисунок 3.10 - Реєстрація

Джерело: сформовано автором на основі виконаного дослідження

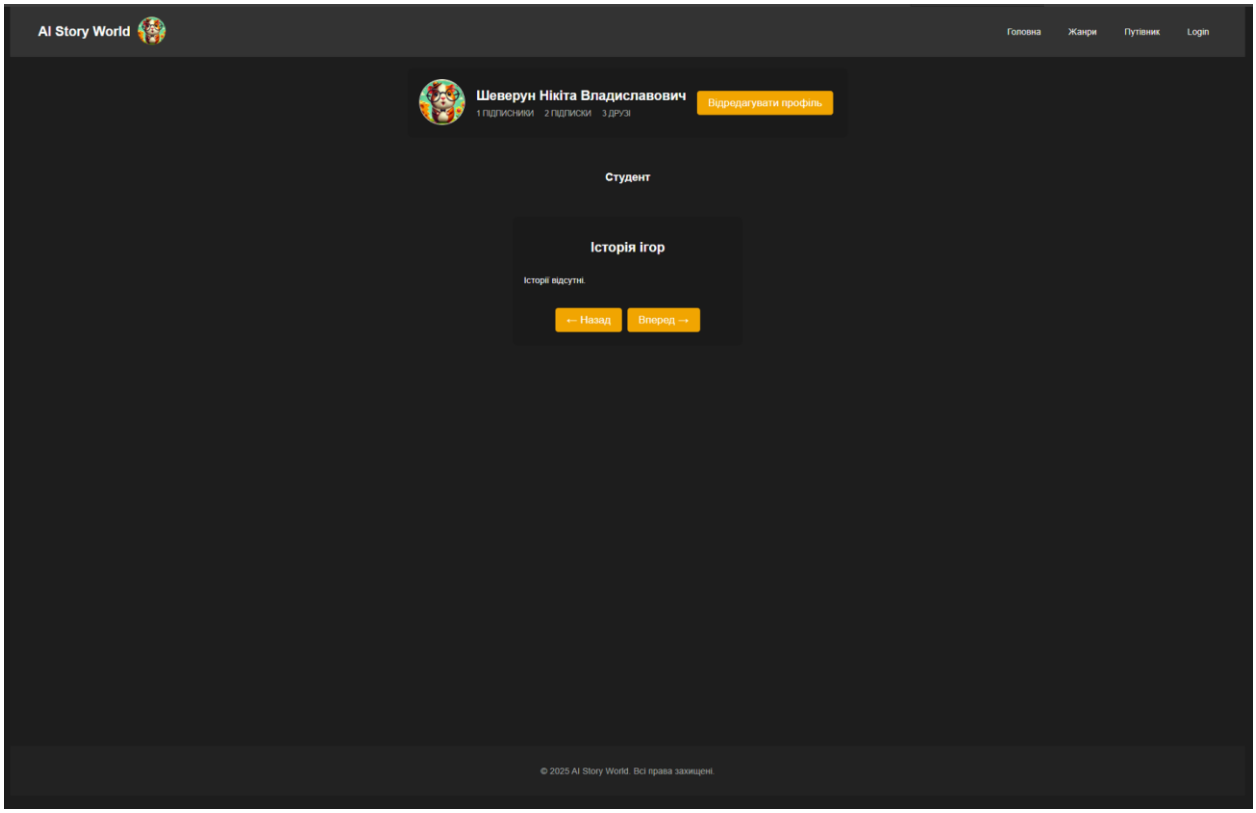


Рисунок 3.11 - Профіль

Джерело: сформовано автором на основі виконаного дослідження

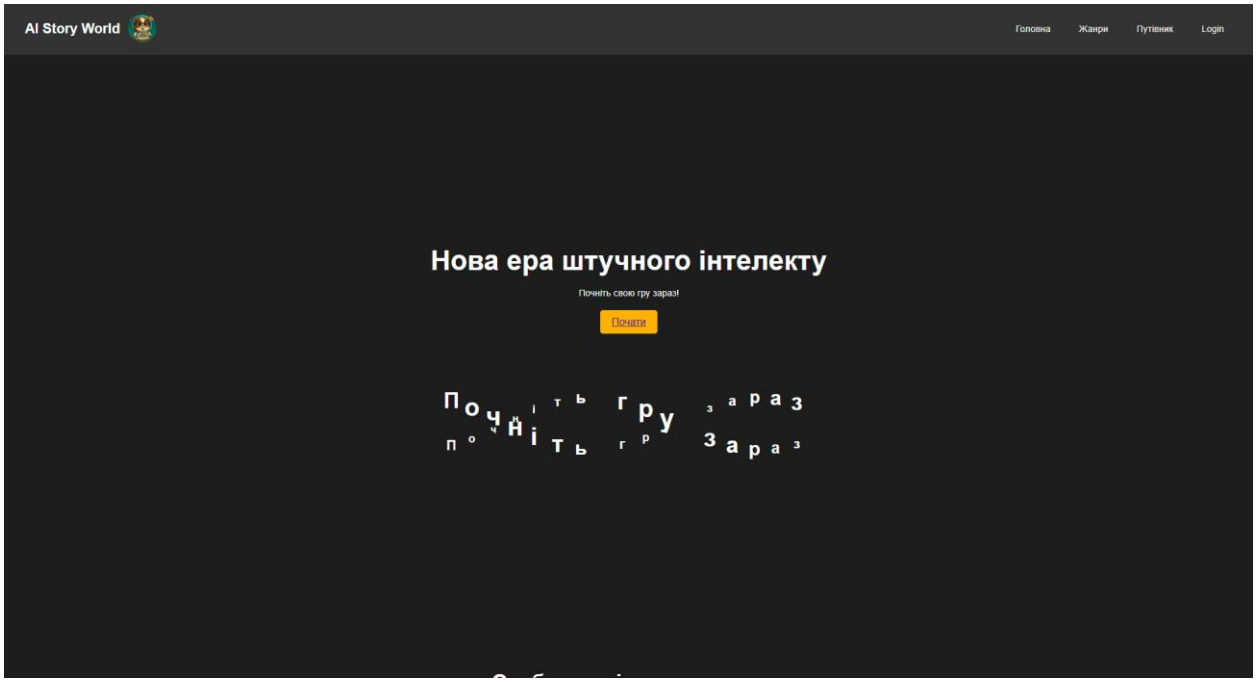


Рисунок 3.12 - головна сторінка з авторизованим аккаунтом

Джерело: сформовано автором на основі виконаного дослідження

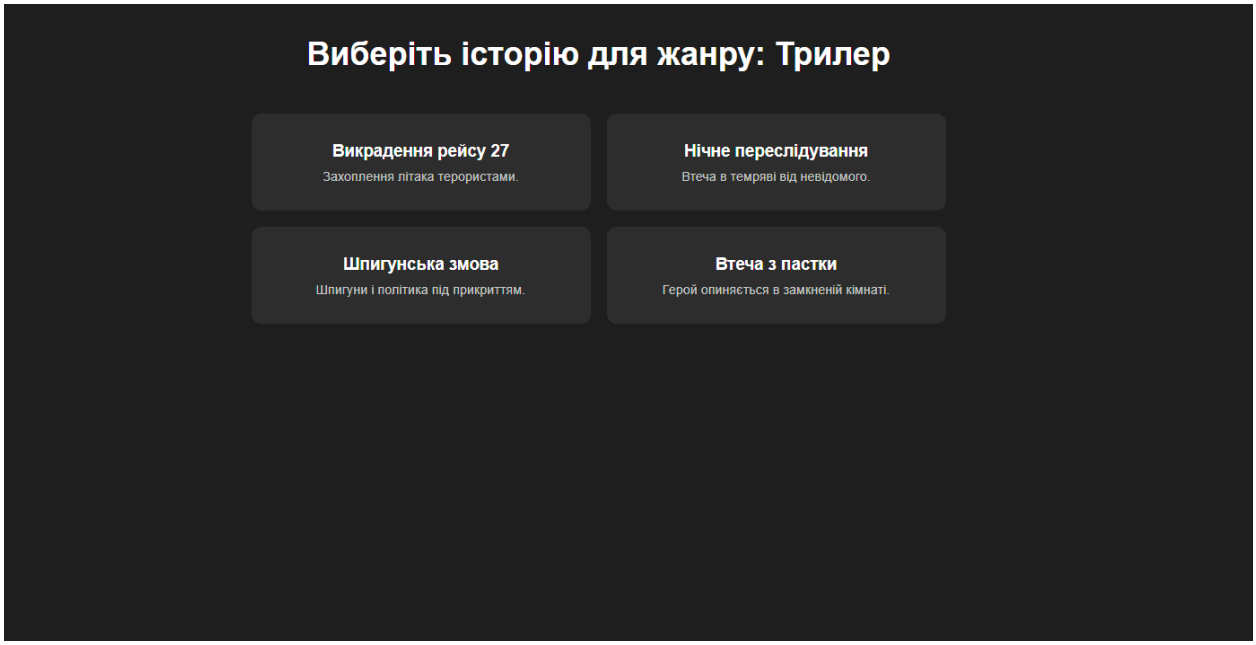


Рисунок 3.13 - Вибір типу історії

Джерело: сформовано автором на основі виконаного дослідження

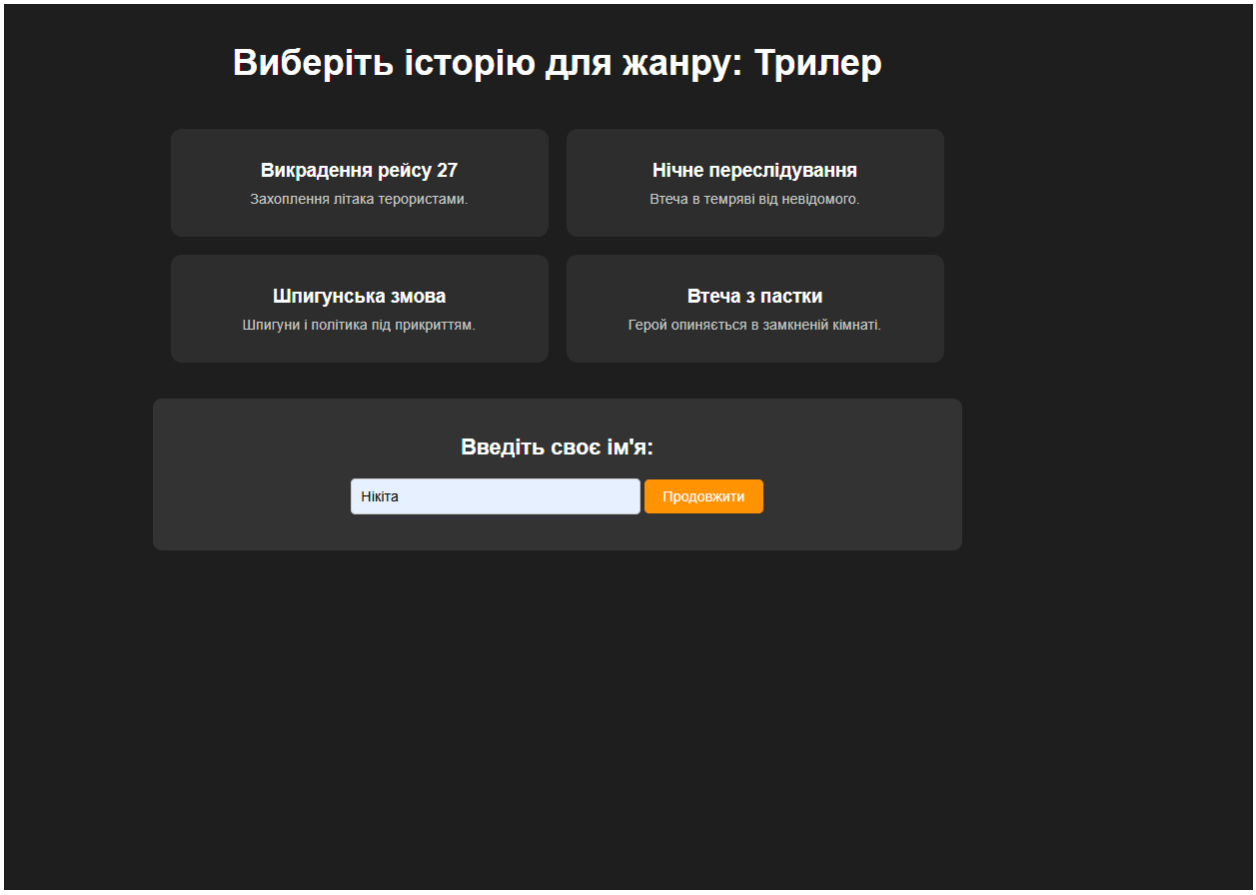


Рисунок 3.14 - Вибір типу історії та написання імені

Джерело: сформовано автором на основі виконаного дослідження

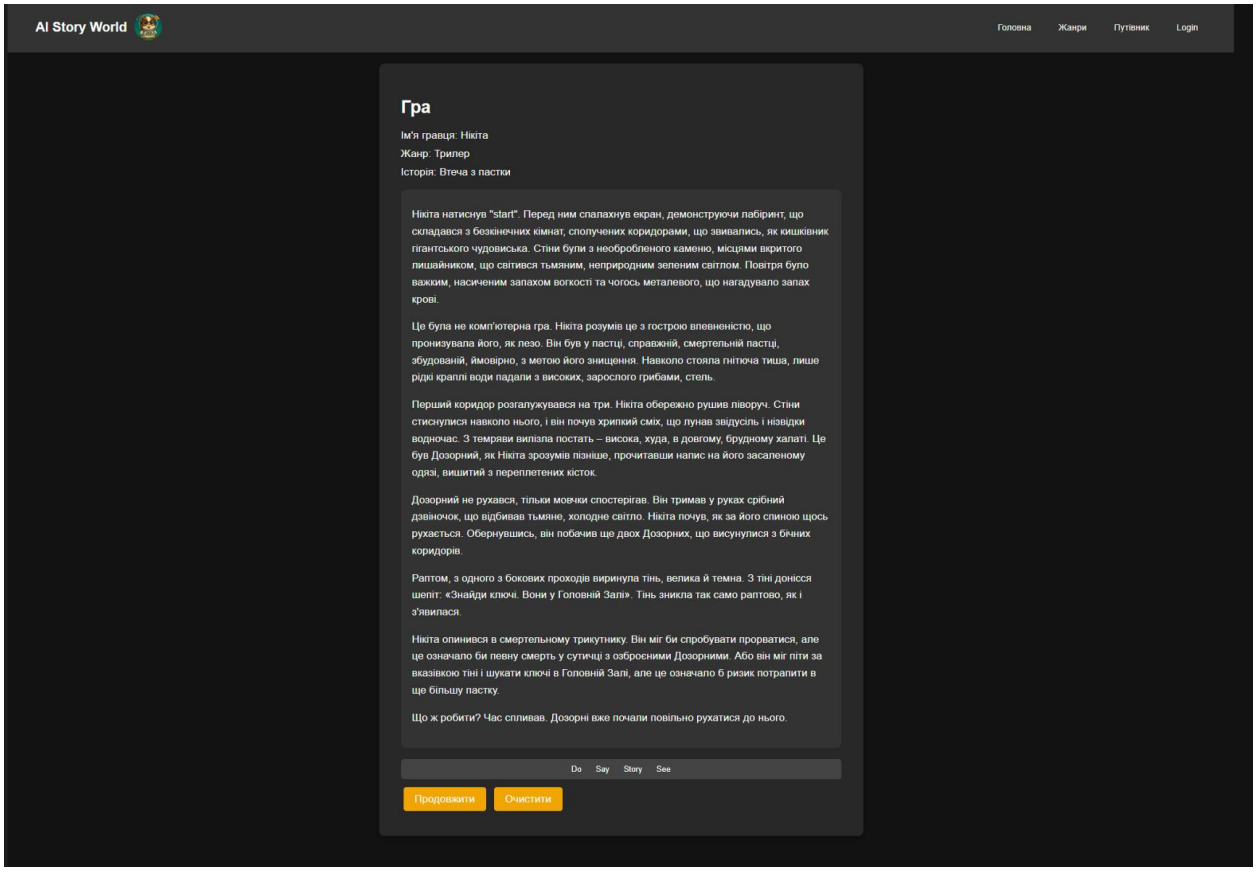


Рисунок 3.15 - Сторінка гри з початковою історією

Джерело: сформовано автором на основі виконаного дослідження

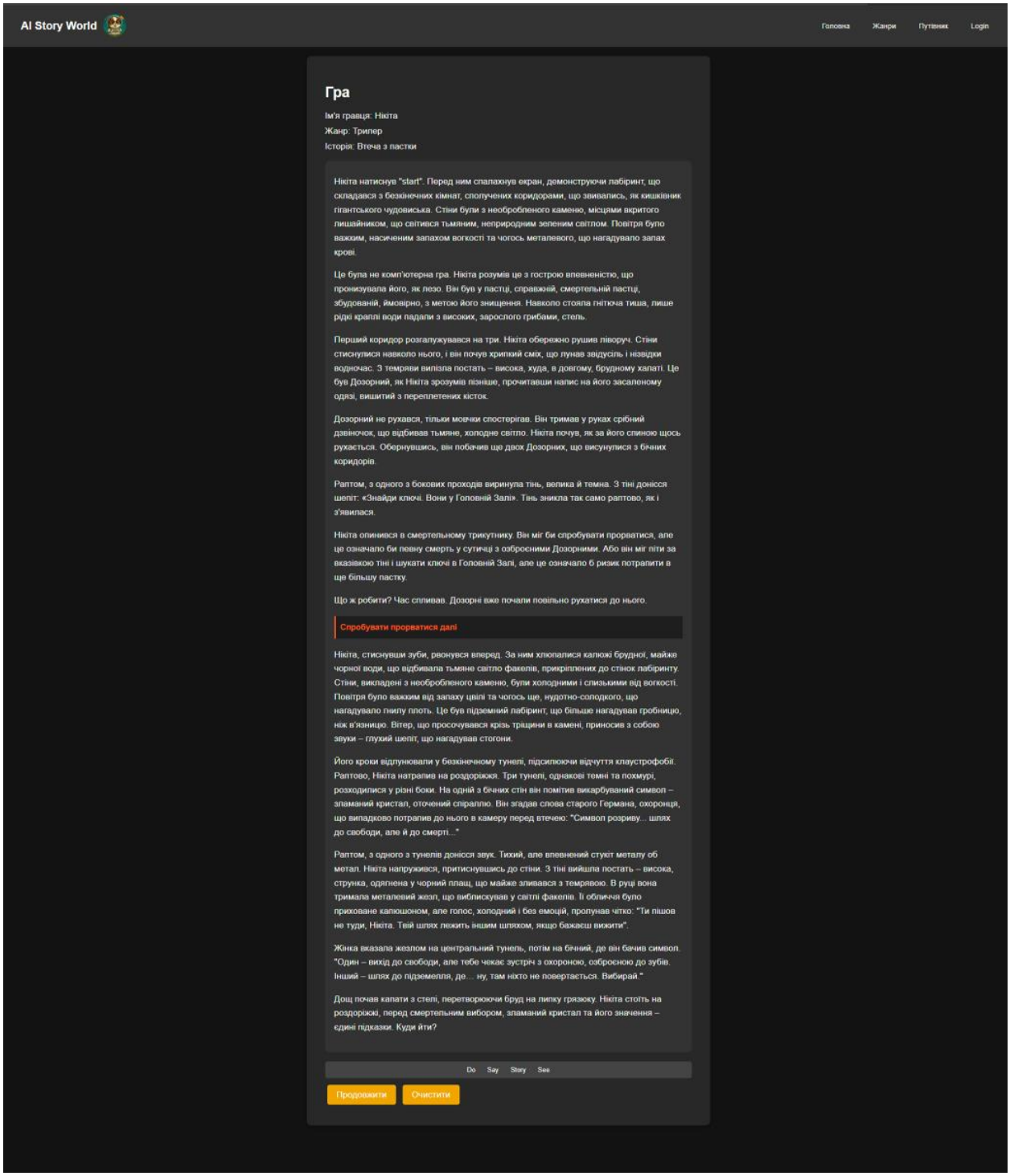


Рисунок 3.16 - Сторінка гри з продовженнями

Джерело: сформовано автором на основі виконаного дослідження

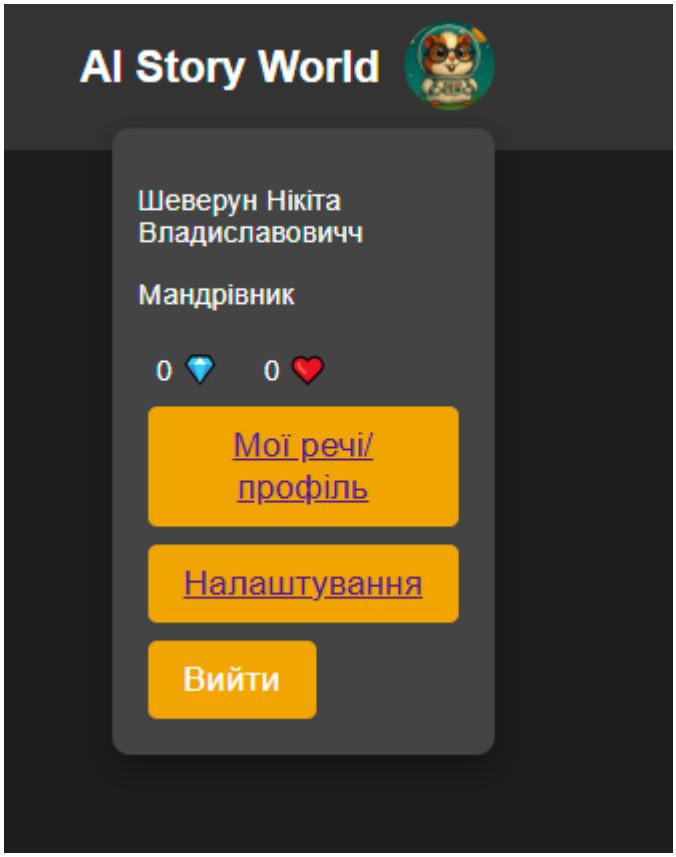


Рисунок 3.17 - меню профілю біля назви сторінки\

Джерело: сформовано автором на основі виконаного дослідження

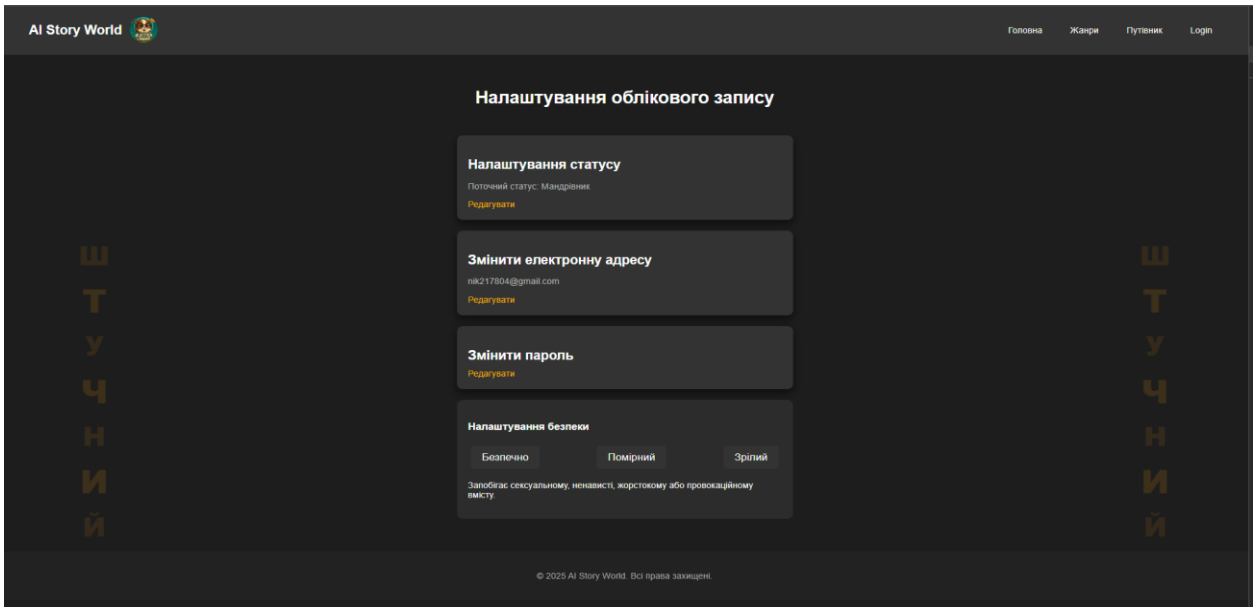


Рисунок 3.18- сторінка налаштування профілю

Джерело: сформовано автором на основі виконаного дослідження

3.4 Висновок

У цьому розділі було здійснено повноцінне проектування та реалізацію всього проекту від загальних сторінок до різного функціоналу та підключення ШІ. Визначено усі види забезпечення, необідні для функціонування системи.

Реалізована база даних PostgreSQL у хмарному середовищі, що забезпечило зручне та швидкодійне збереження даних.

Розроблена архітектура базується на модульному підході з використанням різних технологій: React, Node.js та Express, та важливою частиною стало підключення API Google Gemini для генерації історій. Було розроблено реалізацію взаємодії користувача з системою на різних етапах, від реєстрації до взаємодії з історією. Та розроблено досить зручний інтерфейс для взаємодії.

У рамках розділу побудовано різні моделі бази даних, структуру інформаційних масивів, таблиці та схеми автоматизації. Детально описано роботу кожної системи. Забезпечено логічну цільність компонентів, діаграм, алгоритмів да даних системи.

Отже у результаті всіх дій, цей розділ демонструє завершену реалізацію проекту на практичному рівні, підтверджує її функціональність системи та створює ґрунт для подальшого вдосконалення, експлуатації та інтеграції нових ШІ.

Висновок:

Під час виконання кваліфікаційної бакалаврської роботи було успішно спроектовано та реалізовано веб-застосунок, що поєднує сучасні технології штучного інтелекту та засобів інтерактивної взаємодії з користувачем. Основна ідея полягала у створені української системи, яка надає можливості створювати унікальні історії в режимі реального часу, використовуючи генеративну модель ШІ.

На відміну від деяких існуючих рішень, запропонований проект орієнтований на підтримку української мови, простоту у використані, можливості логічного продовження та побудови дій на основі дій користувача. Проведене моделювання системи дозволило визначити ключові вимоги, структуру даних, інформацію та як буде відбуватися взаємодія з ШІ. Реалізовано робочий проект з технологіям – React, Node.JS, PostgreSQL, Realway та API Gemini.

В системі наявні різні функціональні модулі, такі як : реєстрація та авторизація, вибір жанру та типу історії, генерацію історій, збереження прогресу, вибору імені персонажу та написання правильних промтів. В підтвердження роботи системи було надано різні рисунки та приклади у роботі. Особливу увагу приділено до інтерфейсу, забезпечити його зручністю та адаптивності системи, що позитивно утримуватиме аудиторію.

Суттєвим етапом розрлобки стало тестування працездатності системи. Було протестовано всі основні модулі. Під час написання різних частин проекту, проводилось тестування для перевірки функціоналу кожної сторінки в пошуках проблем. Основну увагу приділялось перевірці взаємодії з інтерфейсом, а також правильності роботи штучного інтелекту. Початковими проблемами було невдале розміщення кнопок та деяких блоків на сторінці HTML. Неуступною проблемою виникла робота штучного інтелекту. Було перероблено логіку гри декілька разів, адже шукав найкращу мовну модель для мого проекту, зупинився саме на Gemini від Google. Після написання логіки, було багато виправлень з правильною роботою над написанням історії з діями які вводив користувач, і після десятків спроб

знайшовши правильний підхід отримав значний результат. В результаті отримано стабільність проекту.

Результатом проекту свідчить реальна можливість застосування інтерактивних ШІ-систем у різних сферах використання, особливо у сфері розваг, де знаходиться реалізований проект. Отриманий продукт має можливість для основи подальшого розвитку з залученням нових систем.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ 2293:2014. Охорона праці. Терміни та визначення основних понять.
2. ДСТУ EN ISO 7010:2019. Графічні символи. Кольори та знаки безпеки. Зареєстровані знаки, безпеки.
3. ДСТУ 7237:2011. Система стандартів безпеки праці. Електробезпека. Загальні вимоги та номенклатура видів захисту.
4. ДСТУ EN 13306:2019. Технічне обслуговування. Термінологія технічного обслуговування (EN 13306:2017, IDT)
5. ДСТУ 2506-94. Засоби обчислювальної техніки. Відмовостійкість і живучість. Загальні технічні вимоги.
6. ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання.
7. Shipping great products is hard. Scaling infrastructure is easy. URL: https://railway.com/ (дата звернення: 16.05.2025).
8. PostgreSQL: The World’s Most Advanced Open Source Relational Database. URL: https://www.postgresql.org/ (дата звернення: 16.05.2025).
9. PostgreSQL vs MySQL: The Critical Dsfferences. URL: https://www.integrate.io/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/ (дата звернення: 16.05.2025).
10. Пасічник В. В., Резніченко В. А. Організація баз даних та знань. К. : BHW, 2006. 384 с.
11. Hasselbring W. Information system integration / Communications of the ACM. June 2000. Vol. 43, No. 6. P. 33-38.
12. Nwasuka N.C., Nwaiwu U. Computer-based production planning, scheduling and control: A review / Journal of Engineering Research. March 2024. Volume 12, Issue 1, P. 275-280.
13. Berrington J. Databases / Anaesthesia & Intensive Care Medicine. March 2017. Volume 18, Issue 3, P. 155-157.
14. IT-рішення для цифрової трансформації бізнес-процесів. URL: https://www.it.ua/ (дата звернення: 16.05.2025).

15. Швидке впровадження автоматизації: як покращити бізнес-процеси. https://keepincrm.com/quick-automatization-with-apix (дата звернення: 16.05.2025).\
16. Штучний інтелект: майбутнє України / Журнал «Інновації». – URL: https://jai.in.ua/archive/2023/ai_mono.pdf (дата звернення: 16.05.2025).
17. The economic potential of generative AI: the next productivity frontier // McKinsey Digital. – URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier (дата звернення: 16.05.2025).
18. Karpathy A. The generalist agent: demo release // arXiv. – 2024. – URL: https://arxiv.org/abs/2405.15051 (дата звернення: 16.05.2025).
19. Google Labs. Stitch: UI + Code = Design tool. – URL: https://www.theverge.com/news/670773/google-labs-stitch-ui-coding-design-tool (дата звернення: 16.05.2025).
20. AI Dungeon – офіційний сайт інтерактивної гри. – URL: https://play.aidungeon.com/ (дата звернення: 16.05.2025).
21. Штучний інтелект і держава: які проєкти реалізуються в Україні // Its-time.com.ua. – URL: https://its-time.com.ua/innovations/shtuchnyj-intelekt-i-derzhava-yaki-proyekty-realizuyutsya-v-ukrayini/ (дата звернення: 16.05.2025).
22. Штучний інтелект: ворог чи союзник? / Колонка на УП. – URL: https://www.pravda.com.ua/columns/2025/02/5/7496889/ (дата звернення: 16.05.2025).
23. Посилання на GitHub з проектом: https://github.com/Nikita2178/Bakalavr_project

ДОДАТКИ

Додаток А до роздлілу 2.1 – таблиці вимог

	A	B	C	D	E	F	G	H	I	J
1	ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
2	BREQ-01	Залучення користувачів	Система повинна підвищувати рівень залучення користувачів через персоналізовану взаємодію	Бізнес-вимога	Розробник	Актуальна	Середня	Висока	Середня	Ключовий показник ефективності системи — активність користувача
3	BREQ-02	Інтуїтивний інтерфейс	Платформа повинна мати простий і зрозумілий для користувача інтерфейс	Бізнес-вимога	Розробник	Актуальна	Середня	Висока	Низька	Покращує досвід користувача і знижує бар'єр входу
4	BREQ-03	Використання ШІ	Система повинна застосовувати генеративний штучний інтелект для створення історій	Бізнес-вимога	Розробник	Актуальна	Висока	Висока	Середня	Основа системи — інтерактивна генерація контенту в реальному часі
5	BREQ-04	Адаптація сценарію	Система повинна адаптувати сценарій історії відповідно до дій та вибору користувача	Бізнес-вимога	Розробник	Актуальна	Висока	Висока	Середня	Підвищує глибину персоналізації та зацікавленість користувача
6	BREQ-05	Збереження прогресу	Система повинна забезпечувати збереження прогресу користувача в історії	Бізнес-вимога	Розробник	Актуальна	Середня	Висока	Середня	Критично важливо для зручності і довготривалого використання

Рисунок А.1 – таблиця бізнес вимог та цілей частина 1

Джерело: сформовано автором на основі виконаного дослідження

	A	B	C	D	E	F	G	H	I	J
7	ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
8	BREQ-06	Підтримка жанрів	Система повинна підтримувати різні жанри та типи сюжетів	Бізнес-вимога	Розробник	Актуальна	Середня	Середня	Низька	Забезпечує різноманітність і розширює цільову аудиторію
9	BREQ-07	Гнучка платформа	Система повинна бути гнучкою і масштабованою для адаптації під нові функції	Бізнес-вимога	Розробник	Актуальна	Висока	Висока	Середня	Дозволяє легко інтегрувати нові модулі і розширення
10	BREQ-08	Розширення бази користувачів	Система повинна бути орієнтована на широке коло користувачів	Бізнес-вимога	Розробник	Актуальна	Середня	Висока	Середня	Розробка повинна враховувати потреби різних типів аудиторій
11	BREQ-09	Якісний продукт	Система повинна відповідати сучасним вимогам якості щодо надійності, дизайну та продуктивності	Бізнес-вимога	Розробник	Актуальна	Висока	Висока	Середня	Створення довготривалого продукту, який буде конкурентним на ринку

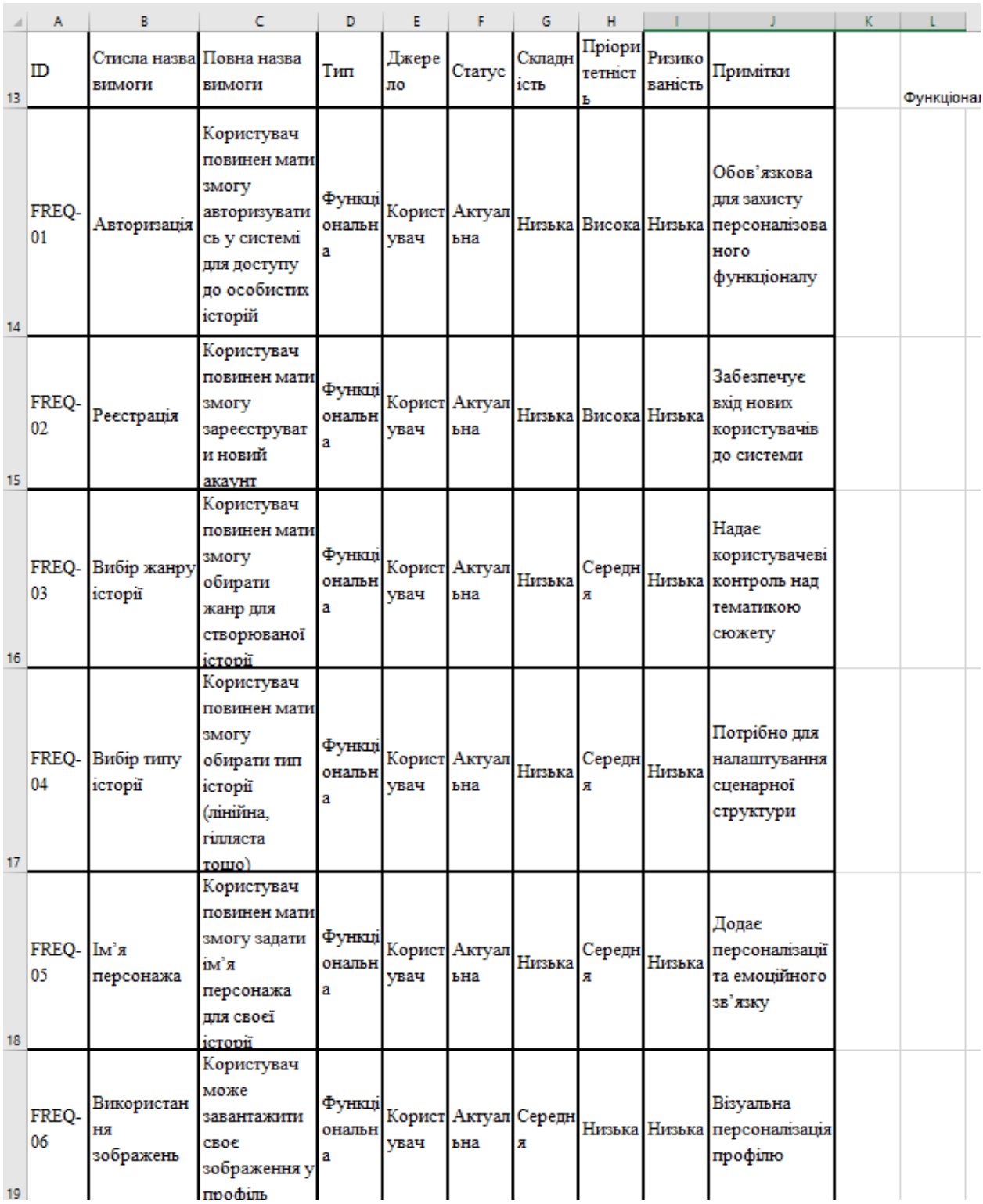
Рисунок А.2 – таблиця бізнес вимог та цілей частина 2

Джерело: сформовано автором на основі виконаного дослідження

ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
UREQ-1	Перегляд історій	Користувач повинен мати змогу переглядати свої або створені історії	Користувацька	Користувач	Актуальна	Низька	Середня	Низька	Дозволяє користувачеві повернутися до історій та продовжити їх
UREQ-2	Реєстрація та авторизація	Користувач повинен мати змогу зареєструватися та авторизуватись у системі	Користувацька	Користувач	Актуальна	Низька	Висока	Низька	Створює основу для персоналізованого використання системи
UREQ-3	Редагування профілю	Користувач повинен мати змогу редагувати інформацію у своєму профілі (ім'я, опис)	Користувацька	Користувач	Актуальна	Низька	Низька	Низька	Допомагає формувати індивідуальний образ у системі
UREQ-4	Вибір жанру і типу	Користувач повинен мати змогу обрати жанр і тип історії перед її початком	Користувацька	Користувач	Актуальна	Низька	Середня	Низька	Формує тематичне спрямування та структуру історії
UREQ-5	Ім'я персонажа	Користувач повинен мати змогу вказати ім'я персонажа для кожної нової історії	Користувацька	Користувач	Актуальна	Низька	Середня	Низька	Додає особистісний вимір до взаємодії зі сценарієм
UREQ-6	Довідник	Користувач повинен мати змогу ознайомитися з основною інформацією у довіднику проєкту	Користувацька	Користувач	Актуальна	Низька	Низька	Низька	Сприяє кращому розумінню функціональності платформи

Рисунок А.3 – таблиця користувацьких вимог

Джерело: сформовано автором на основі виконаного дослідження



	A	B	C	D	E	F	G	H	I	J	K	L
13	ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки		Функціонал
14	FREQ-01	Авторизація	Користувач повинен мати змогу авторизуватись у системі для доступу до особистих історій	Функціональна	Користувач	Актуальна	Низька	Висока	Низька	Обов'язкова для захисту персоналізованого функціоналу		
15	FREQ-02	Реєстрація	Користувач повинен мати змогу зареєструвати новий акаунт	Функціональна	Користувач	Актуальна	Низька	Висока	Низька	Забезпечує вхід нових користувачів до системи		
16	FREQ-03	Вибір жанру історії	Користувач повинен мати змогу обирати жанр для створюваної історії	Функціональна	Користувач	Актуальна	Низька	Середня	Низька	Надає користувачеві контроль над тематикою сюжету		
17	FREQ-04	Вибір типу історії	Користувач повинен мати змогу обирати тип історії (лінійна, гілляста тощо)	Функціональна	Користувач	Актуальна	Низька	Середня	Низька	Потрібно для налаштування сценарної структури		
18	FREQ-05	Ім'я персонажа	Користувач повинен мати змогу задати ім'я персонажа для своєї історії	Функціональна	Користувач	Актуальна	Низька	Середня	Низька	Додає персоналізації та емоційного зв'язку		
19	FREQ-06	Використання зображень	Користувач може завантажити своє зображення у профіль	Функціональна	Користувач	Актуальна	Середня	Низька	Низька	Візуальна персоналізація профілю		

Рисунок А.4 – таблиця функціональних вимог частина 1

Джерело: сформовано автором на основі виконаного дослідження

ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
FREQ-07	Перегляд історій	Користувач повинен мати змогу переглядати створені ним історії	Функціональна	Користувач	Актуальна	Низька	Середня	Низька	Дозволяє аналізувати і продовжувати історії
FREQ-08	Зміна імені	Користувач повинен мати змогу змінити ім'я у профілі	Функціональна	Користувач	Актуальна	Низька	Низька	Низька	Гнучке управління особистими даними
FREQ-09	Зміна опису	Користувач повинен мати змогу редагувати опис профілю	Функціональна	Користувач	Актуальна	Низька	Низька	Низька	Допомагає відобразити інтереси користувача
FREQ-10	Генерація за допомогою ШІ	Система повинна створювати історію на основі дій користувача	Функціональна	Користувач	Актуальна	Висока	Висока	Середня	Основна функція взаємодії користувача з системою
FREQ-11	Швидке оновлення	Система повинна швидко генерувати продовження історії	Функціональна	Користувач	Актуальна	Середня	Середня	Низька	Забезпечує безперервність сюжету
FREQ-12	Адаптація сценарію	Історія повинна змінюватися згідно з попередніми виборами користувача	Функціональна	Користувач	Актуальна	Висока	Висока	Середня	Забезпечує персоналізовану динаміку оповіді

Рисунок А.5 – таблиця функціональних вимог частина 2

Джерело: сформовано автором на основі виконаного дослідження

ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
NREQ-01	Оперативна обробка	Система має оперативно обробляти дії користувача та динамічно формувати фрагмент історії без затримок	Нефункціональна	Розробник	Актуальна	Середня	Висока	Середня	Забезпечує швидку реакцію системи на введення
NREQ-02	Оптимізація продуктивності	Оптимізація продуктивності для роботи в реальному часі	Нефункціональна	Розробник	Актуальна	Середня	Середня	Низька	Підвищує комфортність взаємодії для користувача
NREQ-03	Збереження даних	Система повинна безперебійно зберігати прогрес користувача	Нефункціональна	Розробник	Актуальна	Середня	Висока	Середня	Ключова вимога для збереження ходу історії
NREQ-04	Захист даних	Система повинна запобігати втраті даних у випадку збоїв	Нефункціональна	Розробник	Актуальна	Середня	Середня	Середня	Важливо для стабільної роботи системи
NREQ-05	Хешування паролів	Система повинна зберігати паролі у захищеному вигляді	Нефункціональна	Розробник	Актуальна	Низька	Висока	Низька	Забезпечує базовий рівень безпеки
NREQ-06	Масштабованість	Система повинна масштабуватись із ростом кількості жанрів і типів історій	Нефункціональна	Розробник	Актуальна	Середня	Середня	Низька	Важливо для розширення функціоналу без переробки архітектури

Рисунок А.7 – таблиця нефункціональних вимог частина 1

Джерело: сформовано автором на основі виконаного дослідження

48	ID	Стисла назва вимоги	Повна назва вимоги	Тип	Джерело	Статус	Складність	Пріоритетність	Ризикованість	Примітки
49	NREQ-07	Швидка генерація	Можливість швидко генерувати історії при великій кількості користувачів	Нефункціональна	Розробник	Актуальна	Висока	Висока	Середня	Забезпечує стабільність під навантаженням
50	NREQ-08	Адаптивний дизайн	Система повинна мати адаптивний інтерфейс для різних пристроїв	Нефункціональна	Користувач	Актуальна	Середня	Середня	Низька	Покращує доступність і UX
51	NREQ-09	Зручність навігації	Забезпечення інтуїтивної навігації в межах застосунку	Нефункціональна	Користувач	Актуальна	Низька	Середня	Низька	Сприяє швидкому освоєнню системи
52	NREQ-10	Розмежування запитів	Система повинна чітко розмежовувати запити користувачів	Нефункціональна	Розробник	Актуальна	Середня	Середня	Середня	Необхідно для ізоляції сесій
53	NREQ-11	Фільтрація контенту	Система повинна фільтрувати та вилучати неприйнятний згенерований текст	Нефункціональна	Розробник	Актуальна	Висока	Висока	Середня	Забезпечує етичність і безпеку контенту

Рисунок А.8 – таблиця нефункціональних вимог частина 2

Джерело: сформовано автором на основі виконаного дослідження

genres

Add column

id	name	description
1	Фантастика	Нереальні світи та технології
2	Детектив	Розслідування злочинів
3	Романтика	Любовні стосунки та емоції
4	Пригоди	Подорожі й випробування
5	Жахи	Страх і надприродне
6	Фентезі	Магія, міфи, вигаданий світ
7	Історичний	Минуле та реальні події

Рисунок Б.1 – таблиця жанрів

Джерело: сформовано автором на основі виконаного дослідження

story_types

Add column

id	name	genre_id	description
1	Місто майбутнього	1	Герої досліджують мегаполіс з високими технологія...
2	Космічна колонія	1	Життя на далекій колонії під загрозою.
3	Світ після апокаліпсису	1	Світ, що вижив після катастрофи.
4	Галактична імперія	1	Імперії борються за контроль над космосом.
5	Злочин у потязі	2	Злочин скоєно під час подорожі.
6	Таємнича кімната	2	Закрита кімната приховує відповідь.
7	Вбивство в маєтку	2	Класичне вбивство серед еліти.

Рисунок Б.2 – таблиця типів історії

Джерело: сформовано автором на основі виконаного дослідження

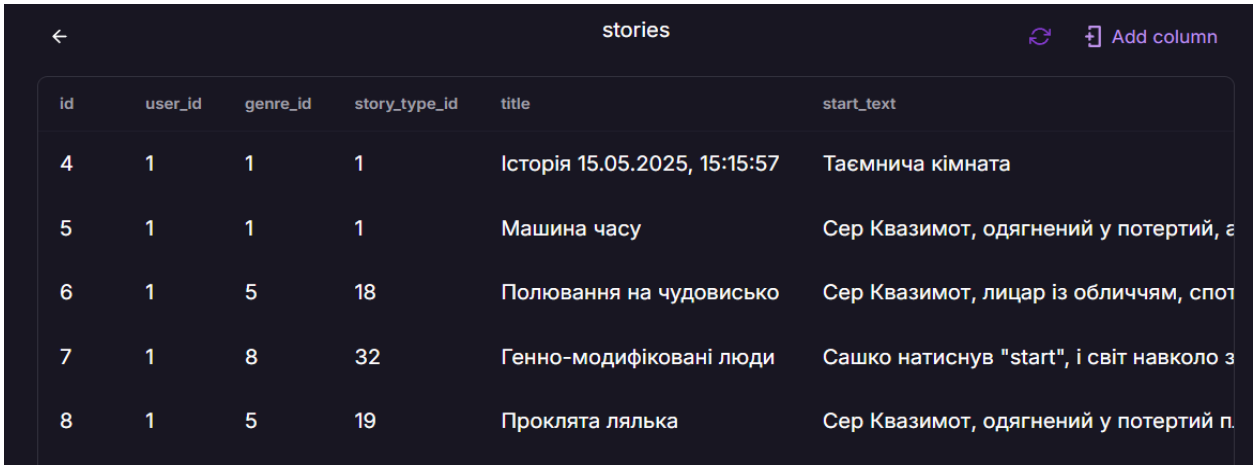


Рисунок Б.3 – таблиця історій

Джерело: сформовано автором на основі виконаного дослідження

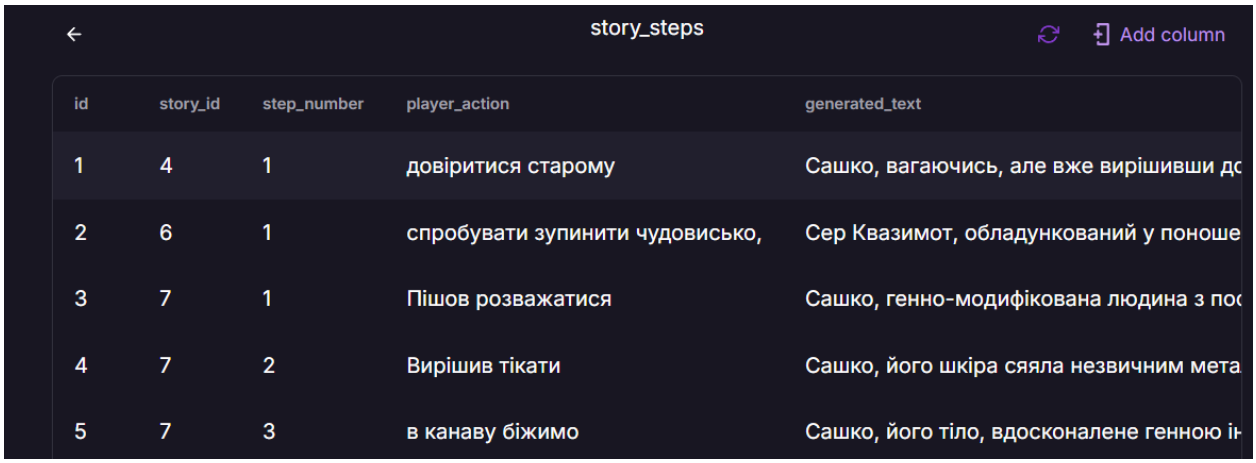


Рисунок Б.4 – таблиця кроків історій

Джерело: сформовано автором на основі виконаного дослідження

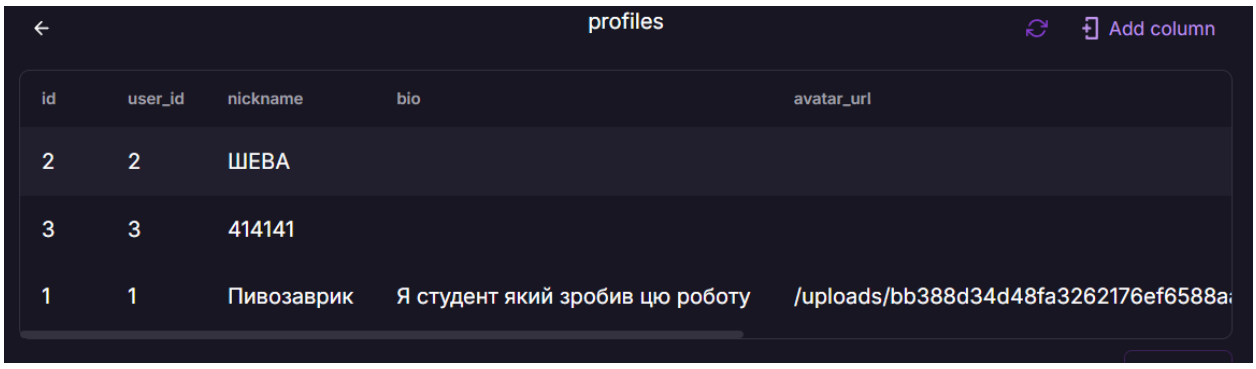


Рисунок Б.5 – таблиця профілю

Джерело: сформовано автором на основі виконаного дослідження

Додаток В до роздлілу 3.1.4 – реалізація інтерфейсу

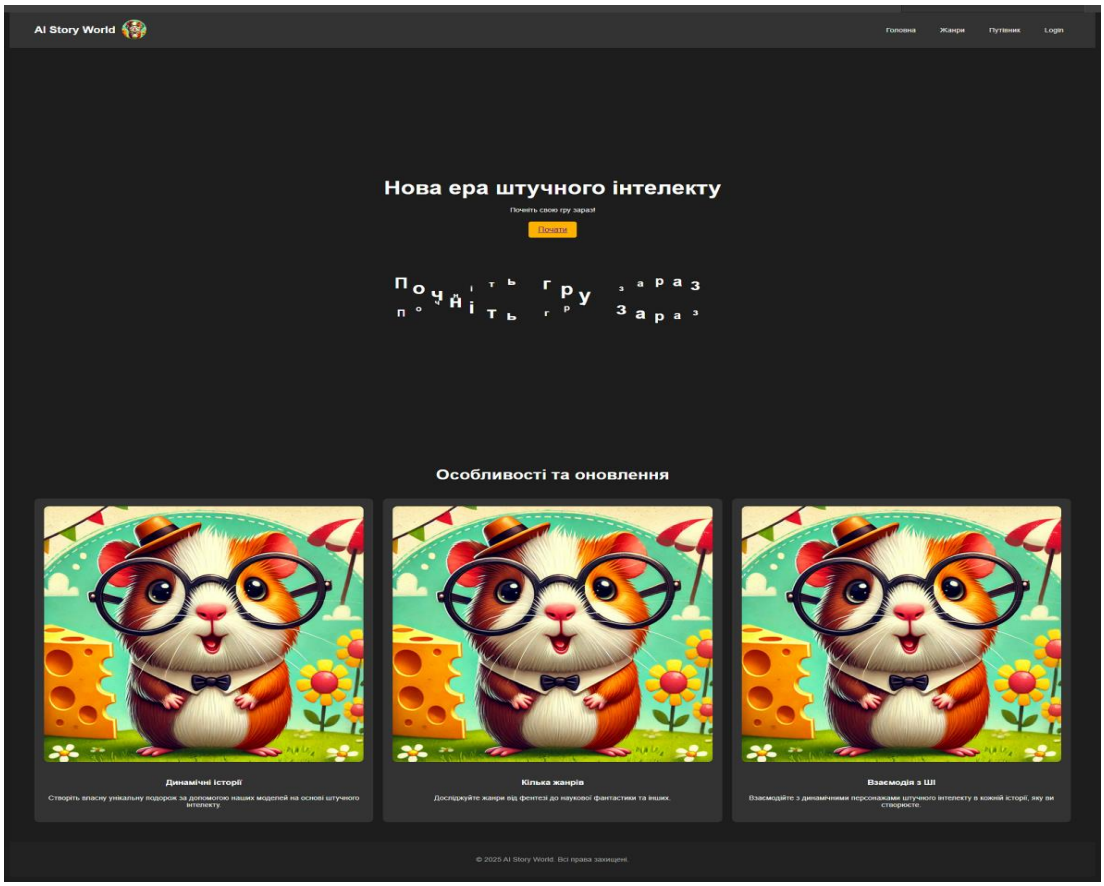


Рисунок В.1 - головна сторінка

Джерело: сформовано автором на основі виконаного дослідження

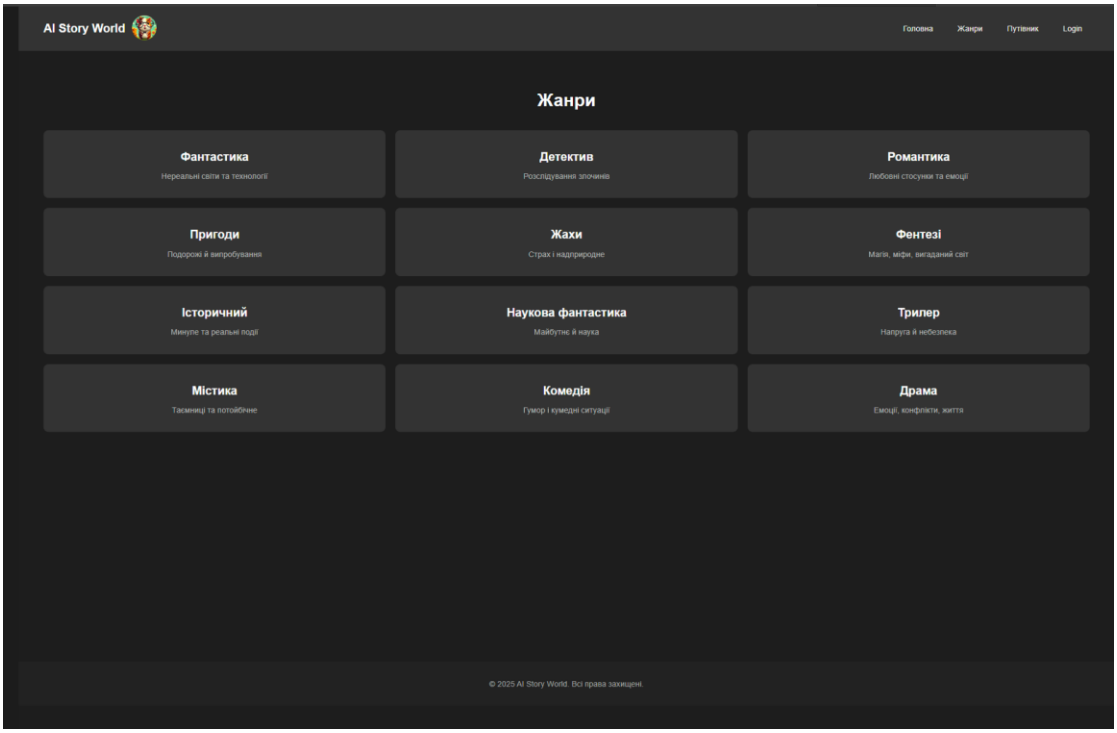


Рисунок В.2 – Жанри

Джерело: сформовано автором на основі виконаного дослідження

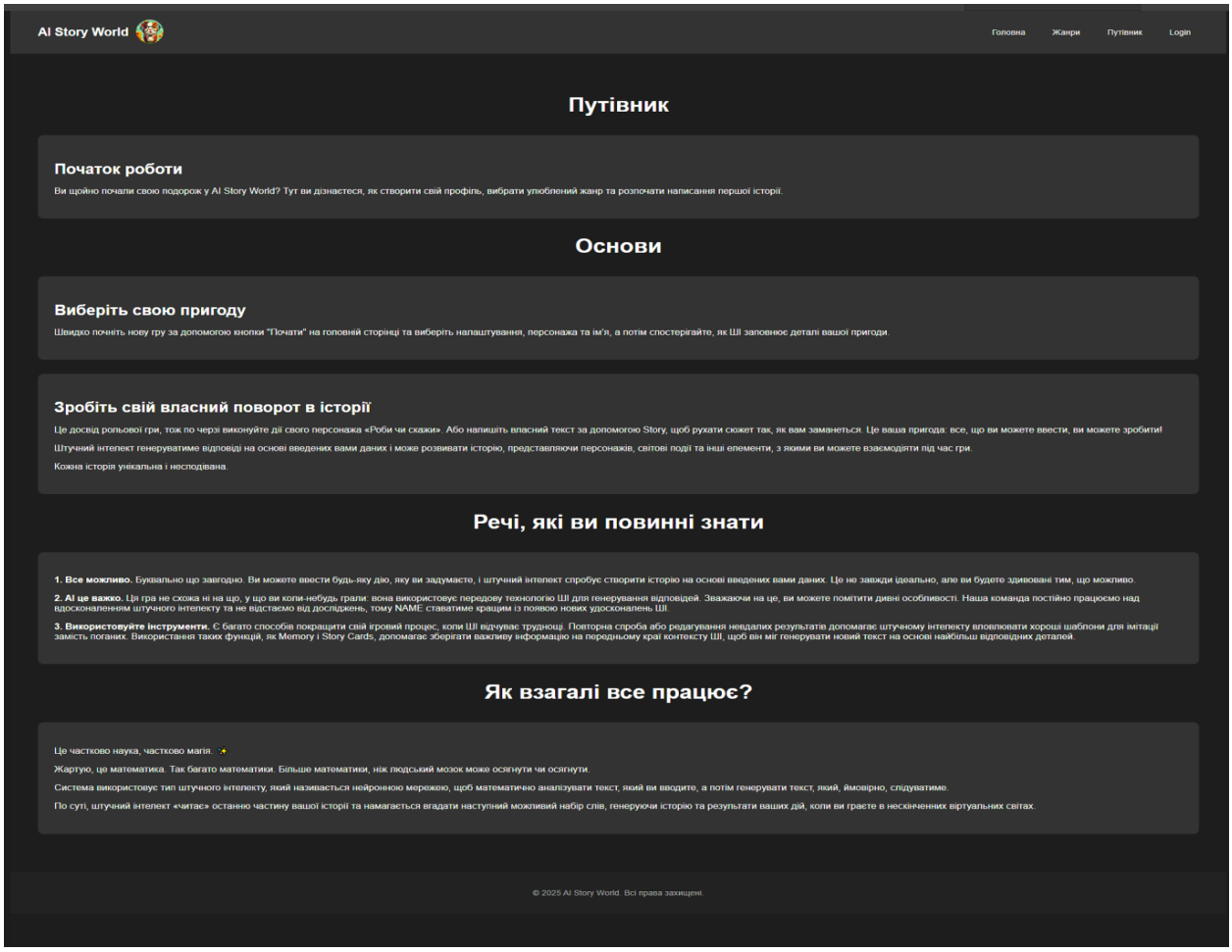


Рисунок В.3 – Путівник

Джерело: сформовано автором на основі виконаного дослідження

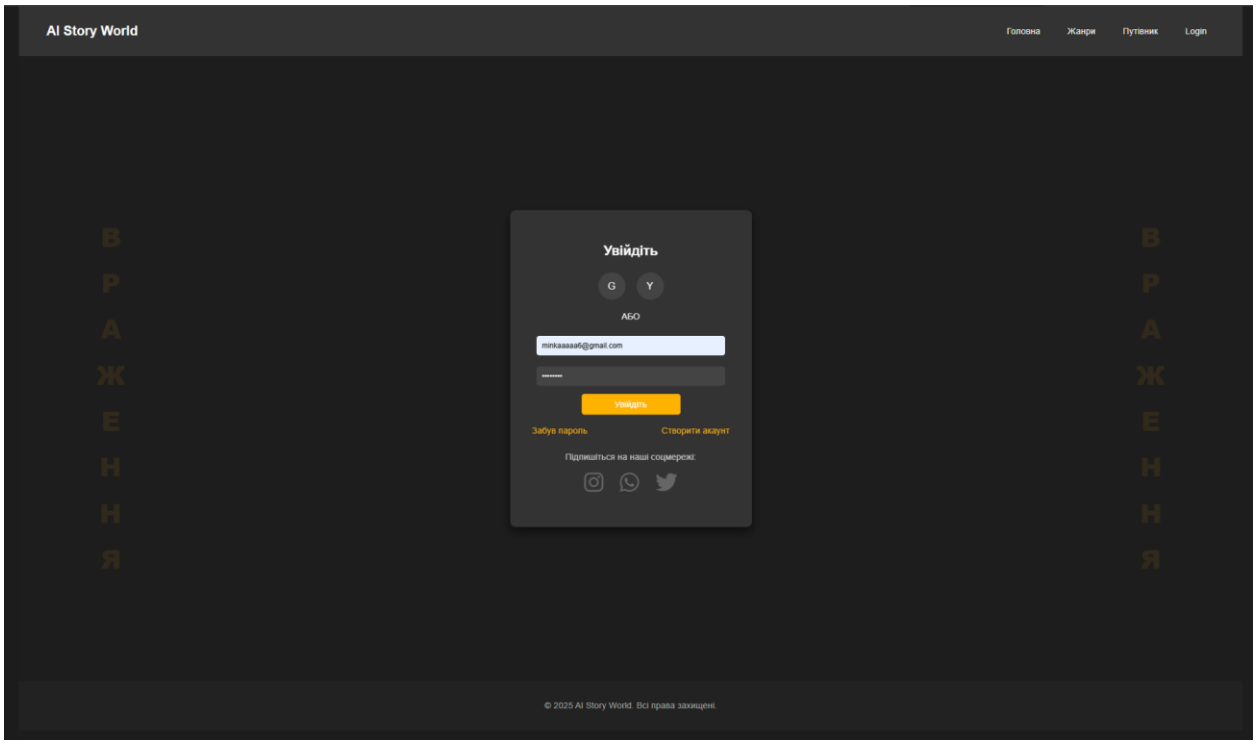


Рисунок В.4 - Вхід в акаунт

Джерело: сформовано автором на основі виконаного дослідження

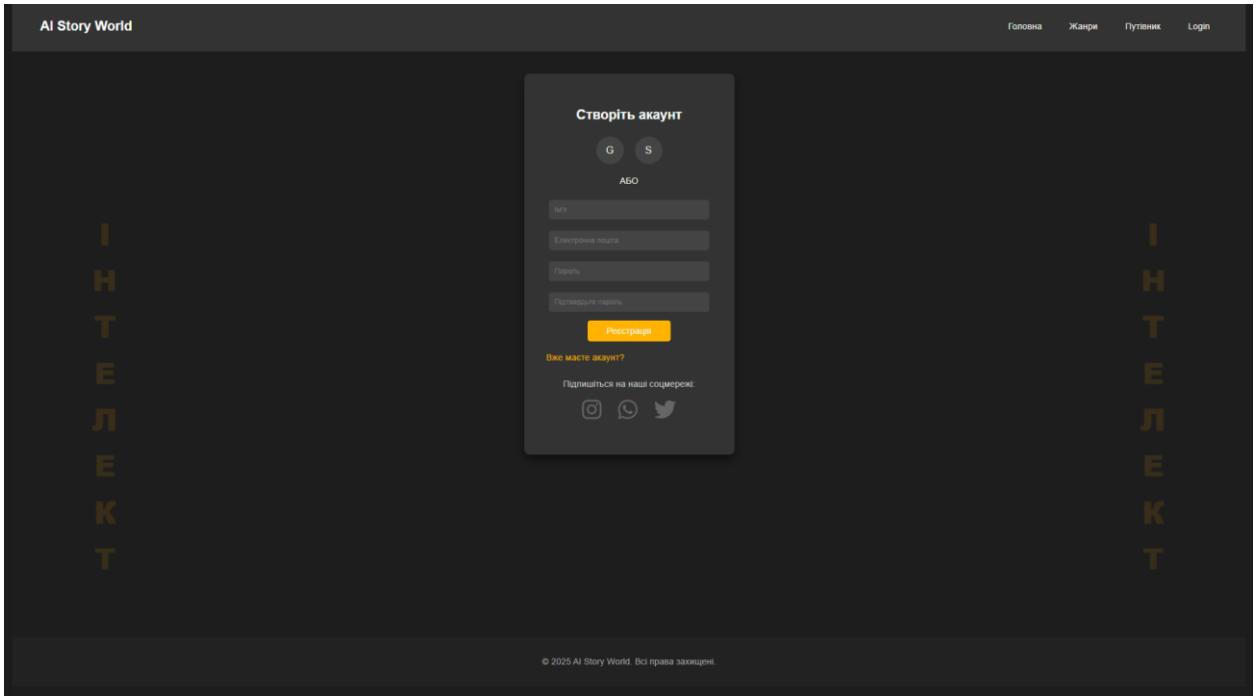


Рисунок В.5 - Реєстрація

Джерело: сформовано автором на основі виконаного дослідження

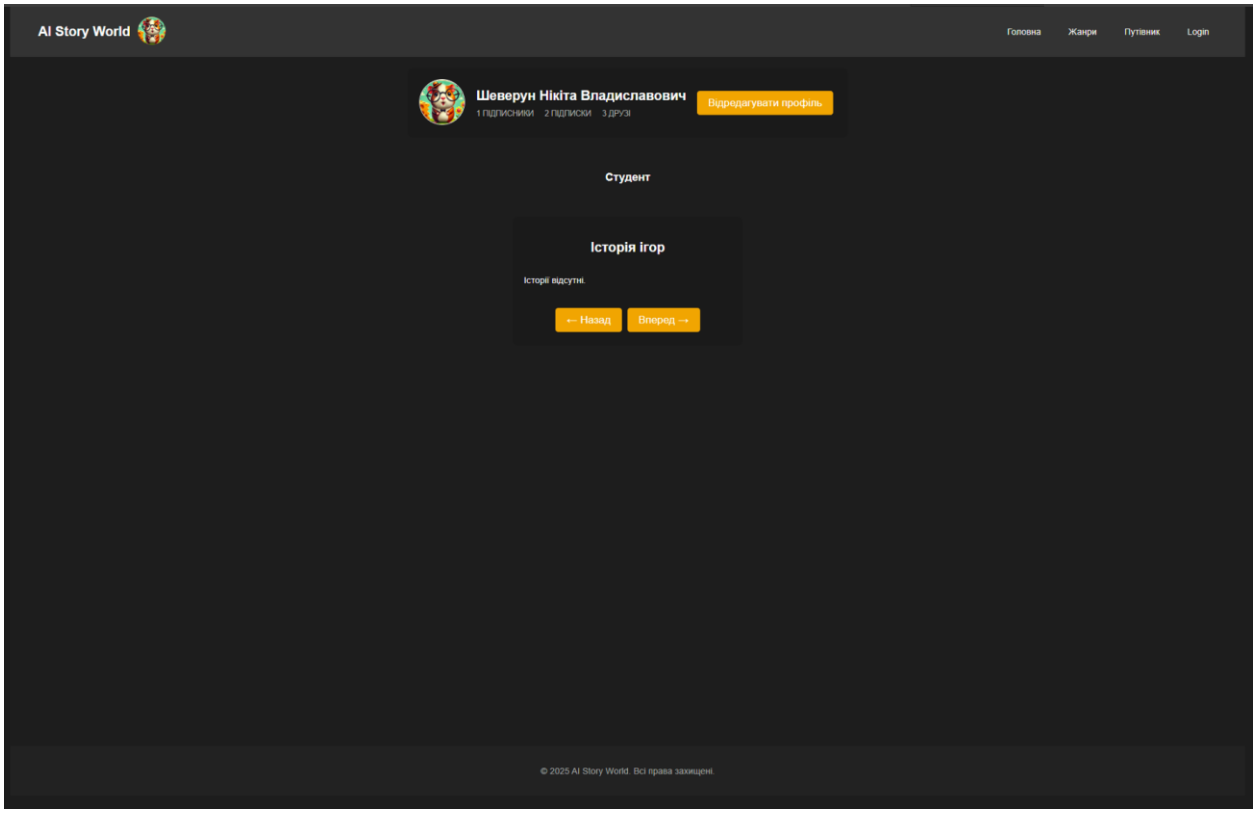


Рисунок В.6 - Профіль

Джерело: сформовано автором на основі виконаного дослідження

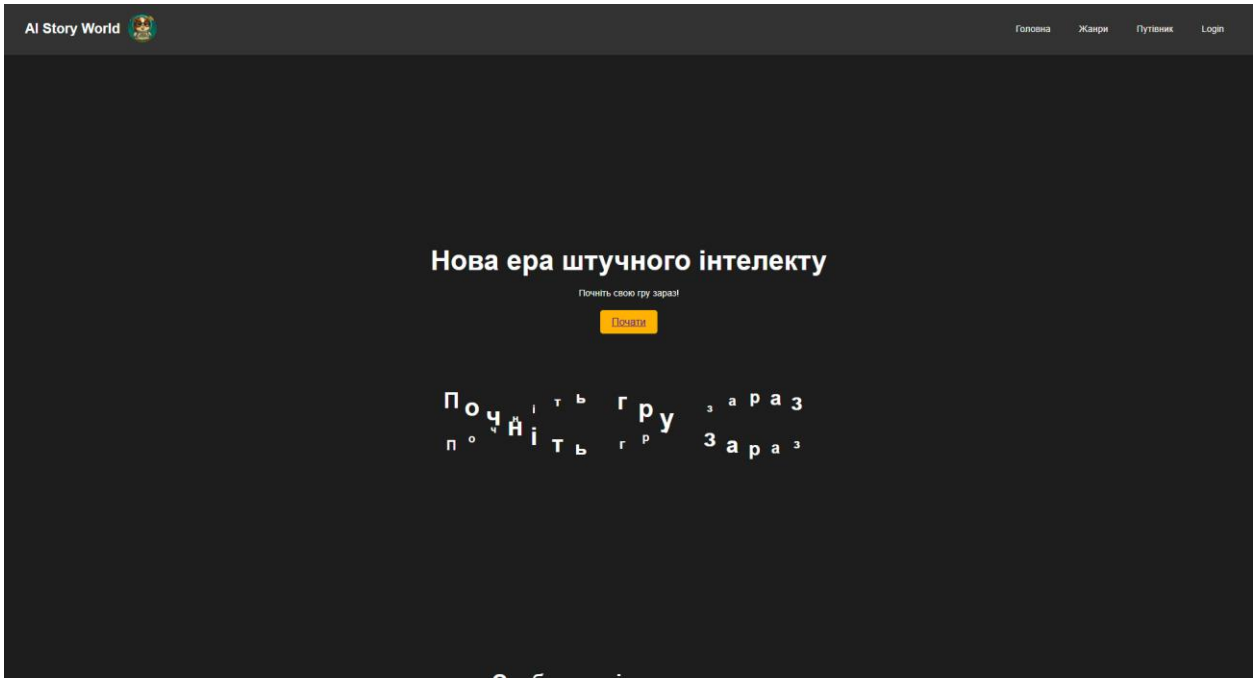


Рисунок В.7- головна сторінка з авторизованим аккаунтом

Джерело: сформовано автором на основі виконаного дослідження

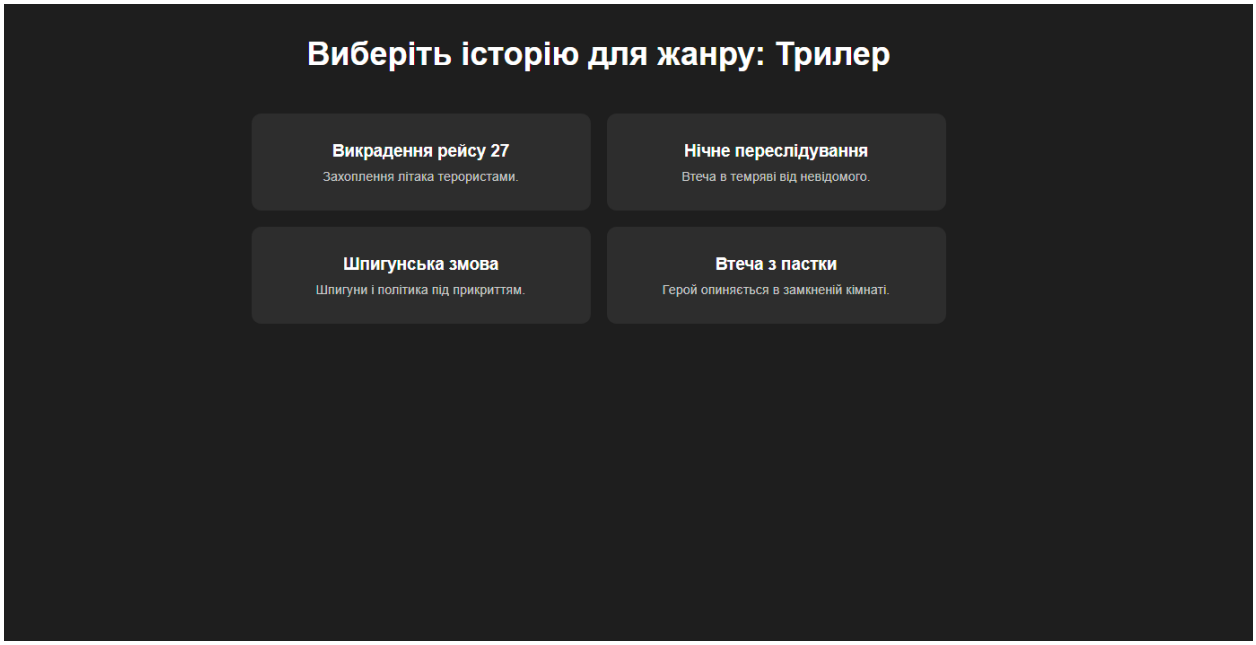


Рисунок В.8- Вибір типу історії

Джерело: сформовано автором на основі виконаного дослідження

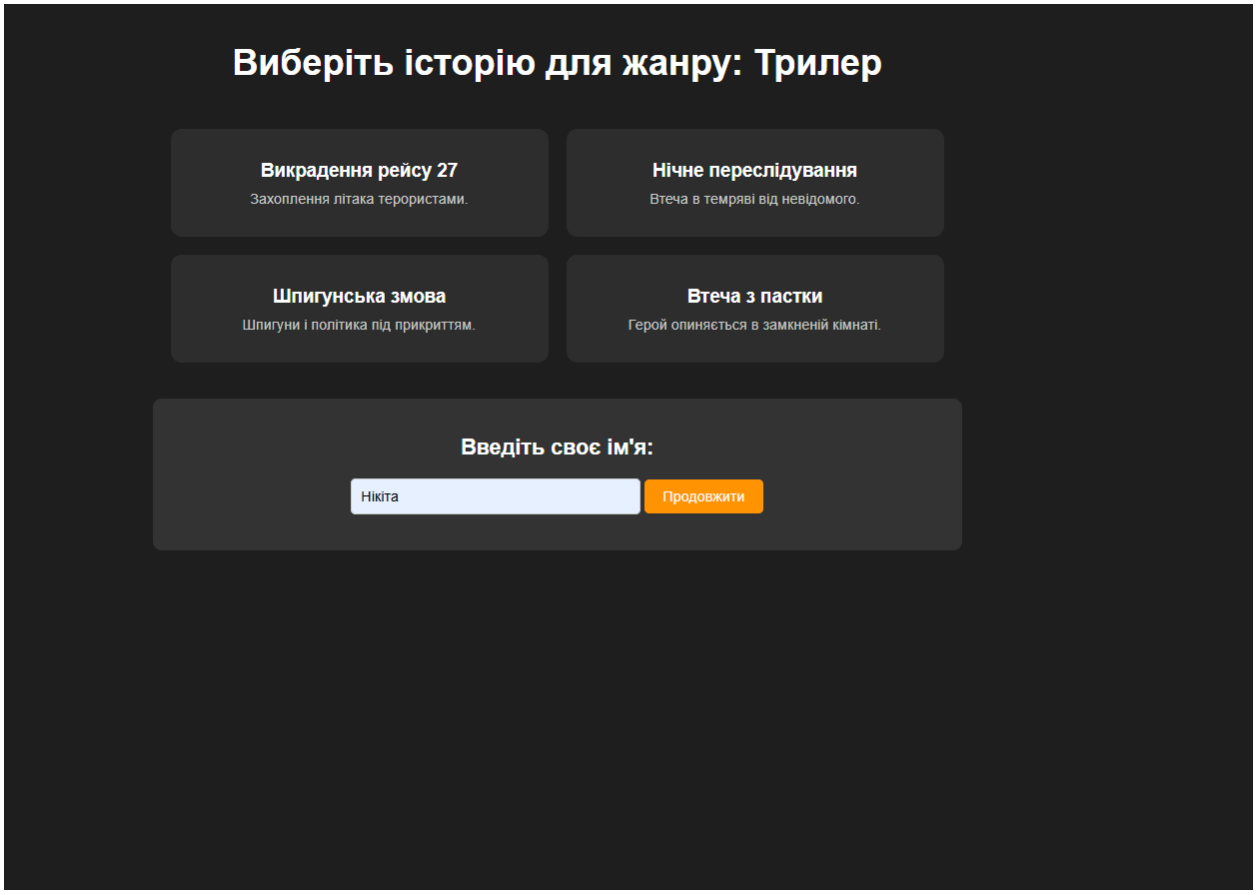


Рисунок В.9- Вибір типу історії та написання імені

Джерело: сформовано автором на основі виконаного дослідження

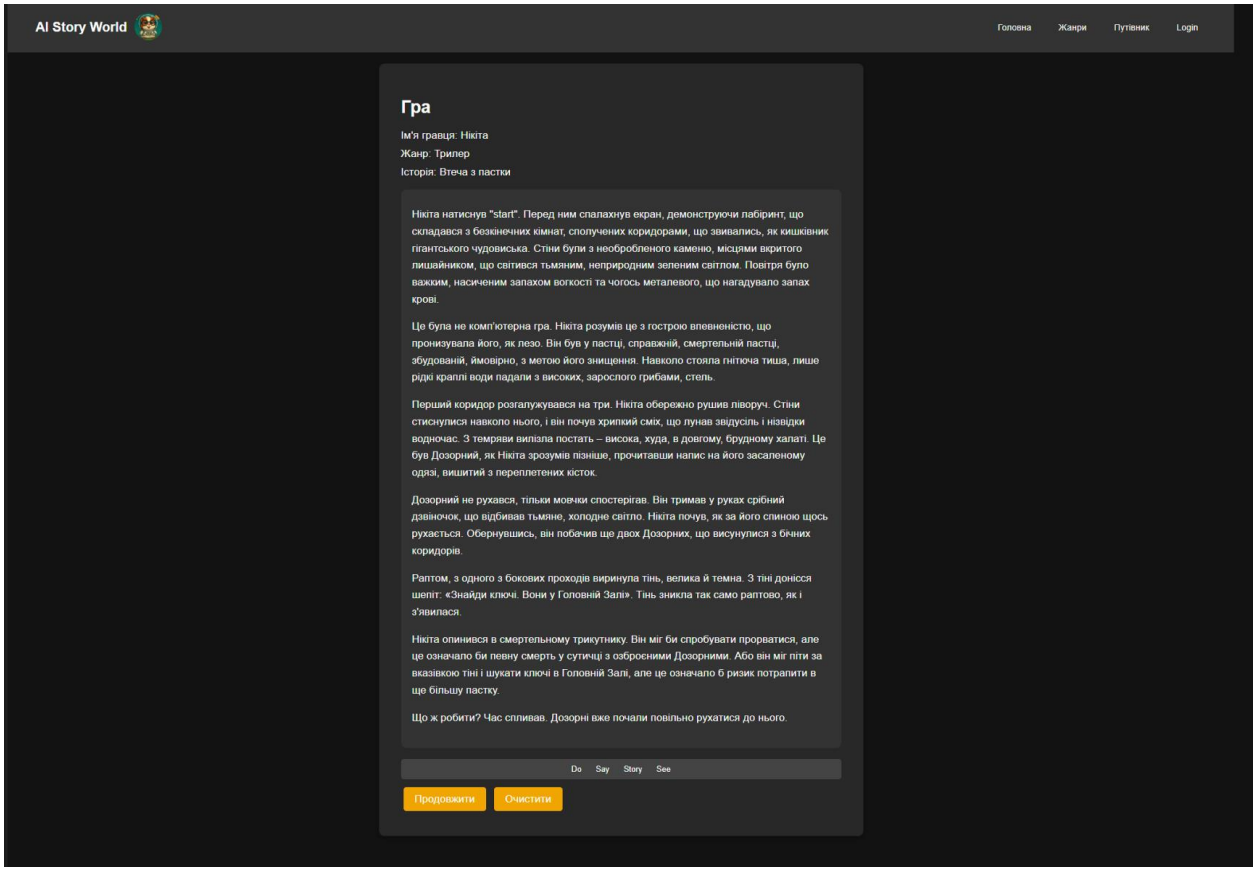


Рисунок В.10- Сторінка гри з початковою історією

Джерело: сформовано автором на основі виконаного дослідження

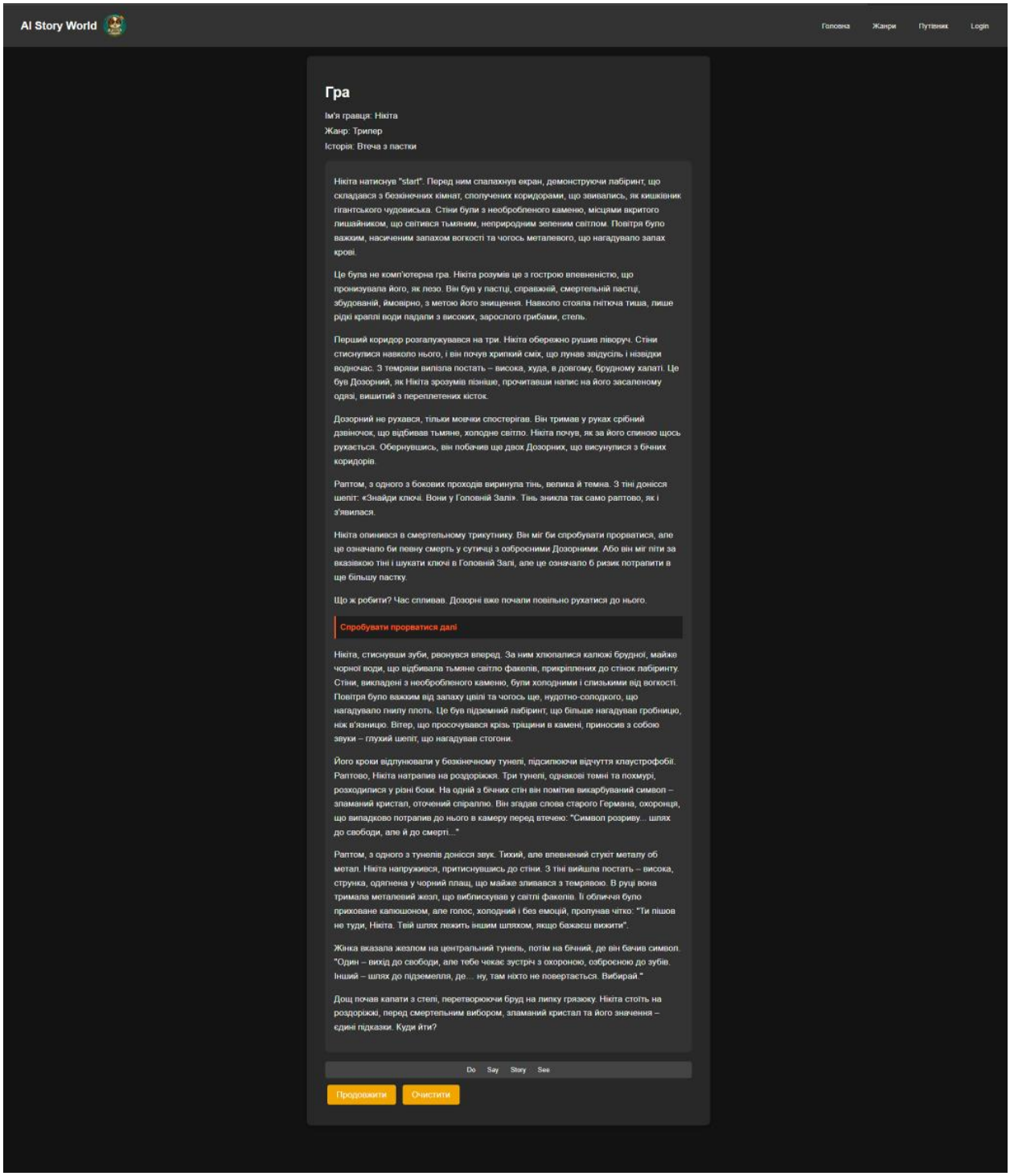


Рисунок В.11 - Сторінка гри з продовженнями

Джерело: сформовано автором на основі виконаного дослідження

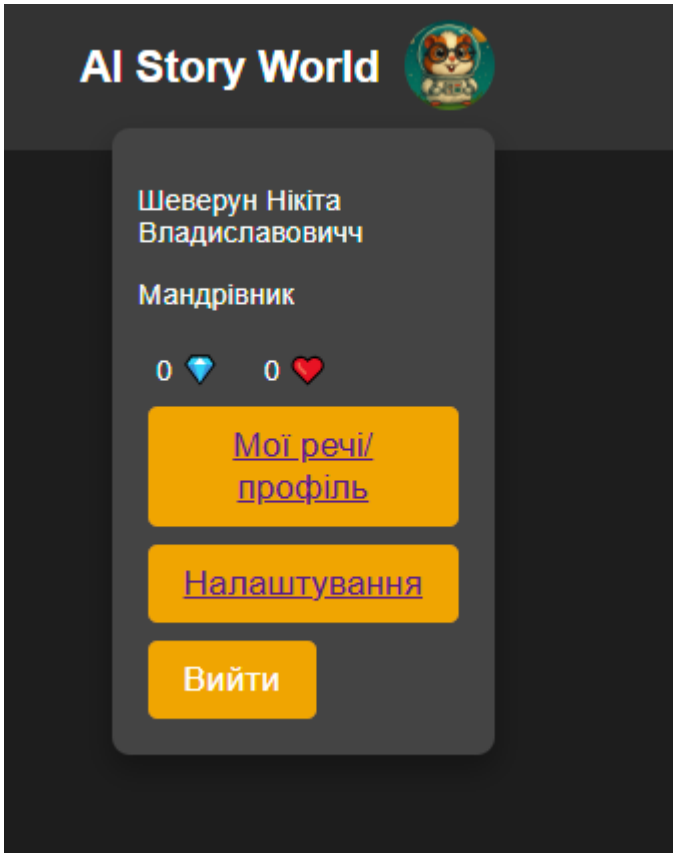


Рисунок В.12 - меню профілю біля назви сторінки

Джерело: сформовано автором на основі виконаного дослідження

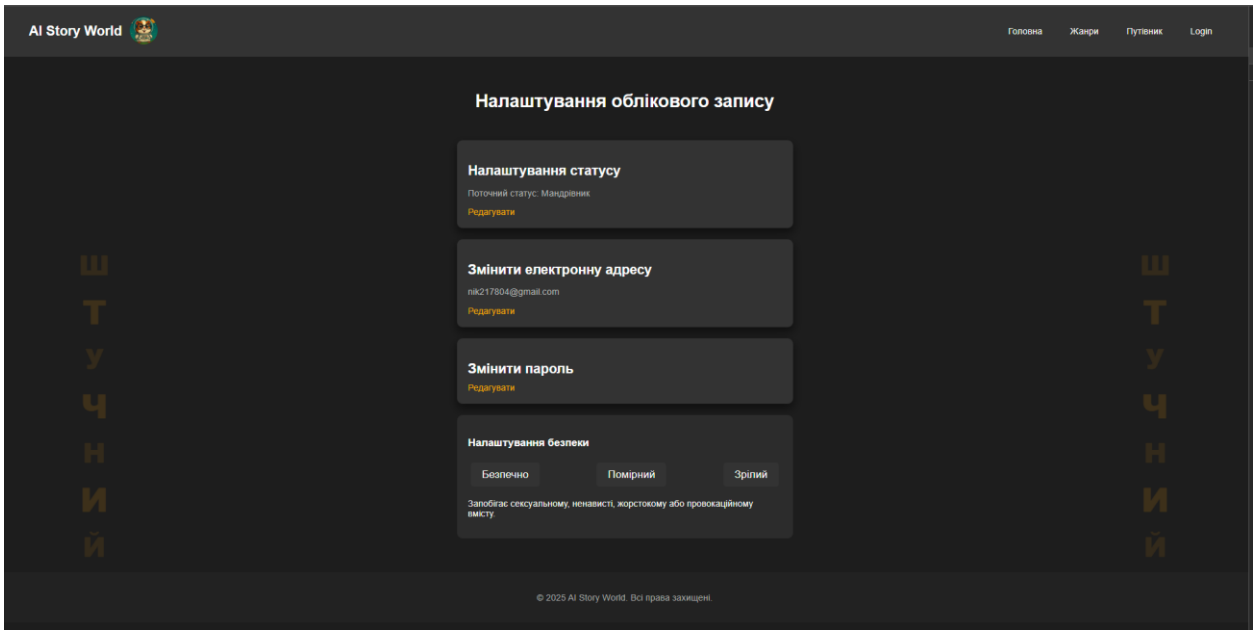


Рисунок В.13 - сторінка налаштування профілю

Джерело: сформовано автором на основі виконаного дослідження

```sql
CREATE TABLE public.users (
    id serial4 NOT NULL,
    email varchar(255) NOT NULL,
    password_hash text NOT NULL,
    created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT users_email_key UNIQUE (email),
    CONSTRAINT users_pkey PRIMARY KEY (id)
);


-- public.follows визначення

-- Drop table

-- DROP TABLE public.follows;

CREATE TABLE public.follows (
    follower_id int4 NOT NULL,
    following_id int4 NOT NULL,
    CONSTRAINT follows_pkey PRIMARY KEY (follower_id, following_id),
    CONSTRAINT follows_follower_id_fkey FOREIGN KEY (follower_id) REFERENCES public.users(id) ON DELETE CASCADE,
    CONSTRAINT follows_following_id_fkey FOREIGN KEY (following_id) REFERENCES public.users(id) ON DELETE CASCADE
);


-- public.friendships визначення

-- Drop table

-- DROP TABLE public.friendships;

CREATE TABLE public.friendships (
    user1_id int4 NOT NULL,
    user2_id int4 NOT NULL,
    created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT friendships_check CHECK ((user1_id < user2_id)),
    CONSTRAINT friendships_pkey PRIMARY KEY (user1_id, user2_id),
    CONSTRAINT friendships_user1_id_fkey FOREIGN KEY (user1_id) REFERENCES public.users(id) ON DELETE CASCADE,
    CONSTRAINT friendships_user2_id_fkey FOREIGN KEY (user2_id) REFERENCES public.users(id) ON DELETE CASCADE
);
```

Рисунок Г.1 – SQL код для створення таблиць частина 1

Джерело: сформовано автором на основі виконаного дослідження

Перегляд SQL:

```sql
CREATE TABLE public.profiles (
    id serial4 NOT NULL,
    user_id int4 NOT NULL,
    nickname varchar(50) NOT NULL,
    bio text NULL,
    avatar_url text NULL,
    followers_count int4 DEFAULT 0 NULL,
    following_count int4 DEFAULT 0 NULL,
    friends_count int4 DEFAULT 0 NULL,
    updated_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT profiles_nickname_key UNIQUE (nickname),
    CONSTRAINT profiles_pkey PRIMARY KEY (id),
    CONSTRAINT profiles_user_id_key UNIQUE (user_id),
    CONSTRAINT profiles_user_id_fkey FOREIGN KEY (user_id) REFERENCES public.users(id) ON DELETE CASCADE
);


-- public.story_types визначення

-- Drop table

-- DROP TABLE public.story_types;

CREATE TABLE public.story_types (
    id serial4 NOT NULL,
    "name" varchar(100) NOT NULL,
    genre_id int4 NULL,
    description text NULL,
    CONSTRAINT story_types_name_key UNIQUE (name),
    CONSTRAINT story_types_pkey PRIMARY KEY (id),
    CONSTRAINT story_types_genre_id_fkey FOREIGN KEY (genre_id) REFERENCES public.genres(id)
);
```

Рисунок Г.2 – SQL код для створення таблиць частина 2

Джерело: сформовано автором на основі виконаного дослідження

```sql
CREATE TABLE public.stories (
    id serial4 NOT NULL,
    user_id int4 NOT NULL,
    genre_id int4 NULL,
    story_type_id int4 NULL,
    title varchar(255) NULL,
    start_text text NOT NULL,
    created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT stories_pkey PRIMARY KEY (id),
    CONSTRAINT stories_genre_id_fkey FOREIGN KEY (genre_id) REFERENCES public.genres(id),
    CONSTRAINT stories_story_type_id_fkey FOREIGN KEY (story_type_id) REFERENCES public.story_types(id),
    CONSTRAINT stories_user_id_fkey FOREIGN KEY (user_id) REFERENCES public.users(id) ON DELETE CASCADE
);


-- public.story_steps визначення

-- Drop table

-- DROP TABLE public.story_steps;

CREATE TABLE public.story_steps (
    id serial4 NOT NULL,
    story_id int4 NOT NULL,
    step_number int4 NOT NULL,
    player_action text NULL,
    generated_text text NOT NULL,
    created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT story_steps_pkey PRIMARY KEY (id),
    CONSTRAINT story_steps_story_id_fkey FOREIGN KEY (story_id) REFERENCES public.stories(id) ON DELETE CASCADE
);
```

Рисунок Г.3 – SQL код для створення таблиць частина 3

Джерело: сформовано автором на основі виконаного дослідження

Увесь код для SQL:

```sql
-- DROP SCHEMA public;

CREATE SCHEMA public AUTHORIZATION pg_database_owner;

-- DROP SEQUENCE public.genres_id_seq;

CREATE SEQUENCE public.genres_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;
-- DROP SEQUENCE public.profiles_id_seq;

CREATE SEQUENCE public.profiles_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;
-- DROP SEQUENCE public.stories_id_seq;

CREATE SEQUENCE public.stories_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;
-- DROP SEQUENCE public.story_steps_id_seq;
```

```
CREATE SEQUENCE public.story_steps_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;
-- DROP SEQUENCE public.story_types_id_seq;

CREATE SEQUENCE public.story_types_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;
-- DROP SEQUENCE public.users_id_seq;

CREATE SEQUENCE public.users_id_seq
INCREMENT BY 1
MINVALUE 1
MAXVALUE 2147483647
START 1
CACHE 1
NO CYCLE;-- public.genres визначення

-- Drop table

-- DROP TABLE public.genres;

CREATE TABLE public.genres (
id serial4 NOT NULL,
"name" varchar(100) NOT NULL,
description text NULL,
CONSTRAINT genres_name_key UNIQUE (name),
CONSTRAINT genres_pkey PRIMARY KEY (id)
);


-- public.users визначення

-- Drop table

-- DROP TABLE public.users;

CREATE TABLE public.users (
id serial4 NOT NULL,
email varchar(255) NOT NULL,
password_hash text NOT NULL,
created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT users_email_key UNIQUE (email),
CONSTRAINT users_pkey PRIMARY KEY (id)
);


-- public.follows визначення

-- Drop table

-- DROP TABLE public.follows;

CREATE TABLE public.follows (
follower_id int4 NOT NULL,
following_id int4 NOT NULL,
CONSTRAINT follows_pkey PRIMARY KEY (follower_id, following_id),
CONSTRAINT follows_follower_id_fkey FOREIGN KEY (follower_id) REFERENCES
public.users(id) ON DELETE CASCADE,
```

```
CONSTRAINT follows_following_id_fkey FOREIGN KEY (following_id) REFERENCES
public.users(id) ON DELETE CASCADE
);


-- public.friendships визначення

-- Drop table

-- DROP TABLE public.friendships;

CREATE TABLE public.friendships (
user1_id int4 NOT NULL,
user2_id int4 NOT NULL,
created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT friendships_check CHECK ((user1_id < user2_id)),
CONSTRAINT friendships_pkey PRIMARY KEY (user1_id, user2_id),
CONSTRAINT friendships_user1_id_fkey FOREIGN KEY (user1_id) REFERENCES
public.users(id) ON DELETE CASCADE,
CONSTRAINT friendships_user2_id_fkey FOREIGN KEY (user2_id) REFERENCES
public.users(id) ON DELETE CASCADE
);


-- public.profiles визначення

-- Drop table

-- DROP TABLE public.profiles;

CREATE TABLE public.profiles (
id serial4 NOT NULL,
user_id int4 NOT NULL,
nickname varchar(50) NOT NULL,
bio text NULL,
avatar_url text NULL,
followers_count int4 DEFAULT 0 NULL,
following_count int4 DEFAULT 0 NULL,
friends_count int4 DEFAULT 0 NULL,
updated_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT profiles_nickname_key UNIQUE (nickname),
CONSTRAINT profiles_pkey PRIMARY KEY (id),
CONSTRAINT profiles_user_id_key UNIQUE (user_id),
CONSTRAINT profiles_user_id_fkey FOREIGN KEY (user_id) REFERENCES public.users(id) ON
DELETE CASCADE
);


-- public.story_types визначення

-- Drop table

-- DROP TABLE public.story_types;

CREATE TABLE public.story_types (
id serial4 NOT NULL,
"name" varchar(100) NOT NULL,
genre_id int4 NULL,
description text NULL,
CONSTRAINT story_types_name_key UNIQUE (name),
CONSTRAINT story_types_pkey PRIMARY KEY (id),
CONSTRAINT story_types_genre_id_fkey FOREIGN KEY (genre_id) REFERENCES
public.genres(id)
);


-- public.stories визначення
```

```sql
-- Drop table

-- DROP TABLE public.stories;

CREATE TABLE public.stories (
id serial4 NOT NULL,
user_id int4 NOT NULL,
genre_id int4 NULL,
story_type_id int4 NULL,
title varchar(255) NULL,
start_text text NOT NULL,
created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT stories_pkey PRIMARY KEY (id),
CONSTRAINT stories_genre_id_fkey FOREIGN KEY (genre_id) REFERENCES public.genres(id),
CONSTRAINT stories_story_type_id_fkey FOREIGN KEY (story_type_id) REFERENCES
public.story_types(id),
CONSTRAINT stories_user_id_fkey FOREIGN KEY (user_id) REFERENCES public.users(id) ON
DELETE CASCADE
);


-- public.story_steps визначення

-- Drop table

-- DROP TABLE public.story_steps;

CREATE TABLE public.story_steps (
id serial4 NOT NULL,
story_id int4 NOT NULL,
step_number int4 NOT NULL,
player_action text NULL,
generated_text text NOT NULL,
created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT story_steps_pkey PRIMARY KEY (id),
CONSTRAINT story_steps_story_id_fkey FOREIGN KEY (story_id) REFERENCES
public.stories(id) ON DELETE CASCADE
);
```

Джерело: сформовано автором на основі виконаного дослідження