Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ МОНІТОРИНГУ РОЗУМІННЯ ПРОЧИТАНИХ ХУДОЖНІХ ТВОРІВ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-21-3
Талах В. О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Яковлева О. В.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Талаху Владиславу Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка мобільного застосунку для моніторингу розуміння прочитаних художніх творів

затверджена наказом університету від 19 травня 2025 року № 381Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 21 травня 2025 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література, інтернет джерела, література для створення тестового набору даних, методики оцінювання експертами, бібліотека графічного інтерфейсу с відкритим кодом Gradio, платформа для розміщення вебзастосунків HuggingFace Spaces, мови програмування Python, Java, JavaScript, фреймворки React Native, Spring Boot, база даних PostgreSQL, файлове сховище MinIO, брокер повідомлень RabbitMQ, авторизаційний сервіс Supabase, мовні моделі GPT, Gemini, Claude.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд читацької активності молоді.
2. Огляд ринку читацьких застосунків.
3. Огляд можливостей штучного інтелекту.
4. Проведення тестування із залученням експертів.
5. Дослідження покриття тексту великими мовними моделями.
5. Вибір найбільш відповідної мовної моделі.
6. Розробка мобільного застосунку.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми читання, огляд існуючих додатків, постановка задачі, дослідження мовних моделей, розробка інструкції, експертна оцінка якості, можливості використання Gradio та Hugging Face, результати експертної оцінки, аналіз можливості покриття тексту мовними моделями, проєктування та розробка мобільного застосунку.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	07.04.2025	
2	Аналіз завдання, підбір літератури	08.04.25-10.04.25	
3	Аналіз літератури з досліджуваної проблеми	11.04.25-14.04.25	
4	Аналіз технічних засобів	15.04.25-20.04.25	
5	Розробка методу	21.04.25-27.04.25	
6	Програмна реалізація	28.04.25-11.05.25	
7	Оформлення пояснювальної записки	12.05.25-20.05.25	
8	Перевірка на нормоконтроль	21.05.25-01.06.25	
9	Перевірка на плагіат	21.05.25-01.06.25	
10	Рецензування	21.05.25-01.06.25	
11	Підготовка презентації та доповіді	21.05.25-18.06.25	
12	Занесення роботи в електронний архів	02.06.25-18.06.25	
	Попередній захист кваліфікаційної роботи	02.06.25-18.06.25	

Дата видачі завдання 7 квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ _____ доц. Яковлева О. В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 68 с., 10 табл., 39 рис., 34 джерела.

ЧИТАННЯ, ХУДОЖНЯ ЛІТЕРАТУРА, ВЕЛИКІ МОВНІ МОДЕЛІ, GPT, GEMINI, CLAUDE, ГЕНЕРАЦІЯ ТЕСТІВ, GRADIO, HUGGING FACE, ЕКСПЕРТНЕ ОЦІНЮВАННЯ, МОБІЛЬНИЙ ЗАСТОСУНОК.

Об’єктом роботи є питання використання мовних моделей для генерації тестів з метою моніторингу розуміння прочитаного матеріалу.

Метою роботи є розробка мобільного застосунку для моніторингу розуміння прочитаних художніх творів, призначеного для використання в родині.

Для досягнення мети проаналізовано стан читацької активності молоді та роль цифрових технологій у формуванні мотивації до читання, оглянуто існуючі застосунки для читання книг, досліджено питання використання LLMs для генерації тестів та обрано модель, яка за параметрами якості, швидкості та вартості як найкраще задовольняє вимогам мобільного застосунку для аналізу розуміння прочитаних художніх творів,

У результаті роботи спроєктовано та реалізовано мобільний застосунок для аналізу розуміння прочитаних художніх творів, призначений для використання в родині.

READING, FICTION, LARGE LANGUAGE MODELS, GPT, GEMINI, CLUADE, TESTS GENERATION, GRADIO, HIGGING FACE, EXPERT EVALUATION, MOBILE APPLICATION.

The object of the work is the use of language models to generate tests for monitoring reading comprehension.

The purpose of the work is to develop a mobile application for monitoring reading comprehension of fiction intended for use in the family.

To achieve this goal, we analyzed the state of youth reading activity and the role of digital technologies in shaping reading motivation, reviewed existing book reading apps, investigated the use of LLMs for test generation, and selected a model that best meets the requirements of a mobile application for analyzing reading comprehension in terms of quality, speed, and cost,

As a result of the work, a mobile application for analyzing the comprehension of read works of art was designed and implemented, intended for use in the family.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

LLM – Large Language Model (велика мовна модель)

ШІ – штучний інтелект

CV – Computer Vision (комп’ютерний зір)

ПЗ – програмне забезпечення

JS – JavaScript

API – Application Programming Interface (інтерфейс програмного застосунку)

JSON – JavaScript Object Notation

XML – Extensible Markup Language (розширювана мова розмітки)

REST – Representational State Transfer (передача репрезентативного стану)

JPA – Java Persistence API

ВСТУП

У сучасному світі присутня тенденція на зменшення кількості читачів серед молоді. Вже третє десятиліття молоді люди надають більшу перевагу комп'ютерним іграм, які мають яскраву та динамічну картинку, що захоплює молодь. Деякі дослідження стверджують, що ігри мають свої позитивні сторони, наприклад, покращення реакції, розвиток креативності, тощо. Проте за умови не контрольованого їх споживання вони можуть давати негативні ефекти та погано впливати на інші сфери життя.

Також останні 7 років дуже популярними стали короткі відео на таких платформах як Instagram, TikTok та YouTube. Дослідження показують, що регулярний перегляд таких відео негативно впливає на можливості зосередження та може викликати залежність. Особливо це стосується дітей, мозок яких ще формується. Тому часто діти та підлітки надають перевагу швидкому контенту для стимуляції мозку та отримання швидкого дофаміну.

Окрім того популярність книг, як можливість провести час або дізнатись нову інформацію, регулярно падає. Це пов'язано із розвитком інтернету. На його просторах можна знайти велику кількість інформації та способів для проведення часу і розваг. Але варто зауважити, що інформація, яку можна знайти в інтернеті, може бути недостовірною, особливо на не перевірених платформах, а час, проведений в інтернеті, не завжди є корисним як з точки зору пізнання та навчання, так і з точки зору відпочинку.

В наш час книги нікуди не ділись. Вони присутні в школах, університетах та повсякденному житті. Вміння читати також нікуди не ділось. Люди кожен день сприймають різну інформацію різного обсягу шляхом її читання. Фактори, описані вище, негативно впливають не тільки на популярність читання, а і на сам його досвід. Тому люди можуть мати проблеми з розумінням тексту навіть обсягом у сторінку. Книги ж потребують ще більшої уваги та концентрації на більш довгий період часу, тому ця задача може бути ще більш складною.

Для вирішення цієї проблеми можуть допомогти програмні застосунки, які б допомагали читачам концентруватись та запам’ятовувати інформацію. Наприклад, для кращого запам’ятовування інформації підходять тести. І було б дуже зручно мати можливість проходити їх прямо в застосунку, відразу після прочитання, а не звертатись для цього в інтернет, де присутні відволікаючі фактори.

Актуальність роботи, полягає у тому, що поточний стан ринку мобільних застосунків не може дати альтернативу, яка б підходила людям усіх вікових категорій та регіонів, бібліотека таких застосунків може не підходити за інтересами, а функціонал може бути достатнім.

Дана робота присвячена вирішенню питання розробки мобільного застосунку для моніторингу розуміння прочитаних художніх творів та призначеного для використання в родині.

Особлива увага приділяється дослідженню можливостей великих мовних моделей для генерації тестів та проведенню експертного оцінювання з метою визначення найбільш відповідної моделі, а також питанню можливостей великих мовних моделей до покриття тексту книг з метою визначення оптимального алгоритму створення тестів.

1 СУЧАСНИЙ СТАН І ТЕНДЕНЦІЇ РОЗРОБКИ ОСВІТНІХ МОБІЛЬНИХ ЗАСТОСУНКІВ, СПРЯМОВАНИХ НА РОЗВИТОК ЧИТАЦЬКИХ НАВИЧОК

1.1 Динаміка читацької активності та роль цифрових технологій у формуванні мотивації до читання

Частка художньої літератури, що читається з електронних книг, телефонів та планшетів все ще не перевищує кількості прочитаних паперових книг, але є суттєвою та показує ріст в останні роки.

Статистика продажу паперових (рис. 1.1) та електронних (рис. 1.2) книг в США станом на 2020 рік показує, що продажі електронних книг складають 20% від загальної кількості.

Опитування проведене Українським інститутом книги серед української молоді показало, що 52,8% опитуваних є активними читачами, з яких 78,4% надають перевагу паперовим книгам [1].

Молодь, яка виросла в епоху комп’ютерних ігор та швидкого медіа контенту, надає меншу перевагу читанню, що погано впливає на навички читання. Комп’ютерні ігри привили дітям любов до яскравої та динамічної картинки. Книги, у порівнянні з іграми, можуть здаватись для молоді повільними, нудними та «сірими».

Соціальні мережі, в часності TikTok або Instagram, надають доступ до швидкого медіа контенту, який триває від кількох секунд до хвилини. Різні дослідження та статті експертів показують, що це поганим чином впливає на здатність концентруватись [2]. Наприклад, дослідження проведене Каліфорнійським університетом, показало, що середня концентрація уваги у людей зменшилась з 2,5 хвилин в 2003 році до 47 секунд у проміжок між 2016 та 2023 роками. Постійна стимуляція мозку швидким цифровим контентом заважає нормально займатись повільними діями.

Книги в свою чергу потребують великої кількості зосередженості та уваги протягом тривалого часу. Тому усі ці фактори негативно впливають на читацький досвід, а саме, ускладнюють концентрацію на читанні, розуміння деталей сюжету, аналіз тексту та його запам’ятовування.

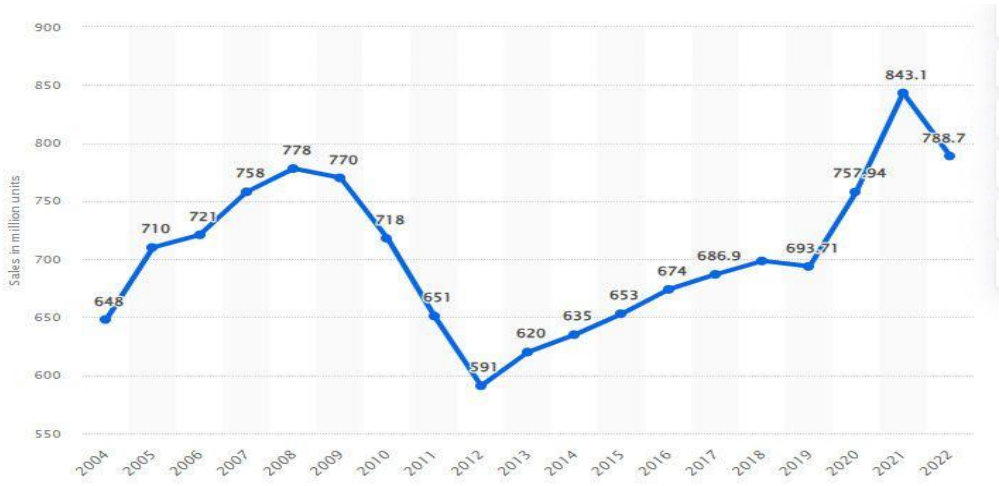

Рисунок 1.1 – Продажі друкованих книг в США [3]

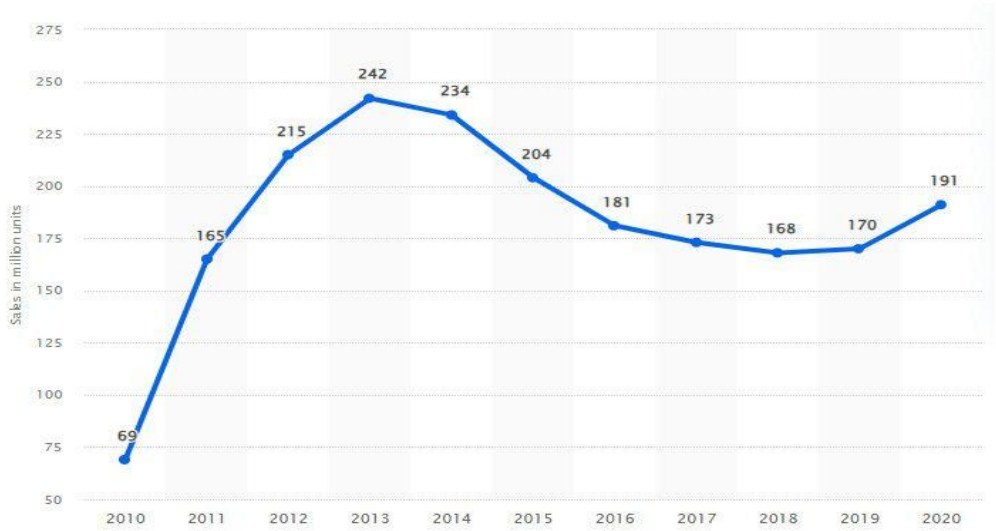

Рисунок 1.2 – Продажі електронних книг в США [4]

Для підвищення інтересу до читання та покращення розуміння тексту можуть допомогти застосунки, які автоматично створюють інтерактивний контент, такий як: тести, ілюстрації, квізи, тощо. Такий контент допомагає запам’ятовувати матеріал шляхом візуалізації в процесі читання та його повторення після прочитання.

Такі застосунки можна використовувати не лише для особистого користування. Вони можуть стати у нагоді в навчальному процесі, де вчителі стикаються з проблемою заохочення дітей до читання, а також у сім’ях, де батьки прагнуть розвивати у дітях любов до книг.

У таких застосунках важлива не лише інтерактивність, що підвищує інтерес та запам’ятовування, а й елементи контролю, зокрема можливість оцінювати швидкість читання, обсяг прочитаного та, найголовніше, розуміння змісту.

1.2 Розвиток штучного інтелекту та його застосування в освітніх застосунках.

Останні роки показують значний прогрес у сфері штучного інтелекту. Великі мовні моделі, системи комп'ютерного зору та мультимодальні моделі досягли вражаючих результатів у різноманітних завданнях [5, 6]. Ці технології знаходять застосування в різних сферах [7 – 11]. Перспективним є застосування великих мовних моделей (LLM, Large Language Models) для аналізу текстів та генерації запитань, що дозволяє створювати інструменти для оцінювання розуміння прочитаного.

1.2.1 Прогрес і можливості штучного інтелекту та сфери його застосування

З 2022 року спостерігається стрімкий розвиток LLM, які демонструють вражаючі можливості в обробці та генерації тексту:

- розуміння контексту та здатність вести змістовний діалог;
- генерація тексту різних стилів та форматів;
- переклад між багатьма мовами;

– аналіз та узагальнення великих обсягів інформації;

– програмування, для написання та пояснення коду.

З 60 років минулого століття та до 2012 року, до прориву AlexNet, для вирішення основних задач комп’ютерного зору, таких як розпізнавання, детекція, сегментація об’єктів, використовувалися евристичні підходи, де для отримання ознак зображення застосовуються заздалегідь підібрані шаблони, функції, маски. Вони якнайкраще підкреслюють особливості зображення, але потребують багато налаштувань та дуже залежать від геометричних перетворень об’єктів та наявності різних завад. Прикладами таких методів є інваріанти Фур’є, ознаки Уолша, моменти Ценріке, вейвлети, дескриптори SIFT, SURF, ORB, матриці збігів, ознаки Лавса та інші [12 – 20].

З 2012 року на арену вийшов нейромережевий підхід, який суттєво підняв точність та розширив спектр задач, які можуть бути вирішені. Найзначніші успіхі комп’ютерного зору були досягнені у розпізнаванні об'єктів, сегментації зображень на рівні пікселів, генерація зображень за текстовим описом, відео аналітиці в реальному часі, розпізнаванні дій та жестів. Але нейромережевий підхід потребує великих датасетів та суттєві потужності обладнання для навчання. Тому для багатьох практичних завдань комп’ютерного зору застосовується комбінація класичних та нейромережевих методів [21, 22].

1.2.2 Можливості штучного інтелекту та інших інновацій для покращення читацьких навичок в освітніх застосунках

Штучний інтелект (ШІ), впроваджений в освітні застосунки, може допомогти в покращенні читацьких навичок. За його допомогою можна додавати в застосунки такий функціонал:

– біонічне читання – методика виділення перших літер, зазвичай жирним шрифтом, яка допомагає швидше читати текст (рис. 1.3). Таким чином

людина читає перші декілька літер, а решту слова розпізнає за його формою та виходячи з контексту;

– генерація тестів на основі тексту книги. Такий підхід дає можливість пройти тестування після прочитання, що допомагає краще та на довше запам’ятати текст. Також це додає цікавості в процес читання та мотивацію отримувати кращі оцінки за тест;

– створення зображень, що ілюструють події. Цей підхід впливає на якість запам’ятовування, оскільки надає візуальну інформацію, яка може допомогти зрозуміти події та сюжет. Особливо корисним це є для сучасної молоді, яка може мати проблеми з фантазією та зосередженням;

– озвучення тексту – сучасні підходи цієї технології дають можливість зробити якісніше озвучування книг. Це може бути корисним для дітей, яким складно дається читання, або для тих дітей, які мають вади зору;

– розпізнавання мови – ця технологія також зазнала покращень за останні роки, тому її можна використовувати для покращення швидкості читання або перевірки промови для дітей, які тільки вчаться читати.

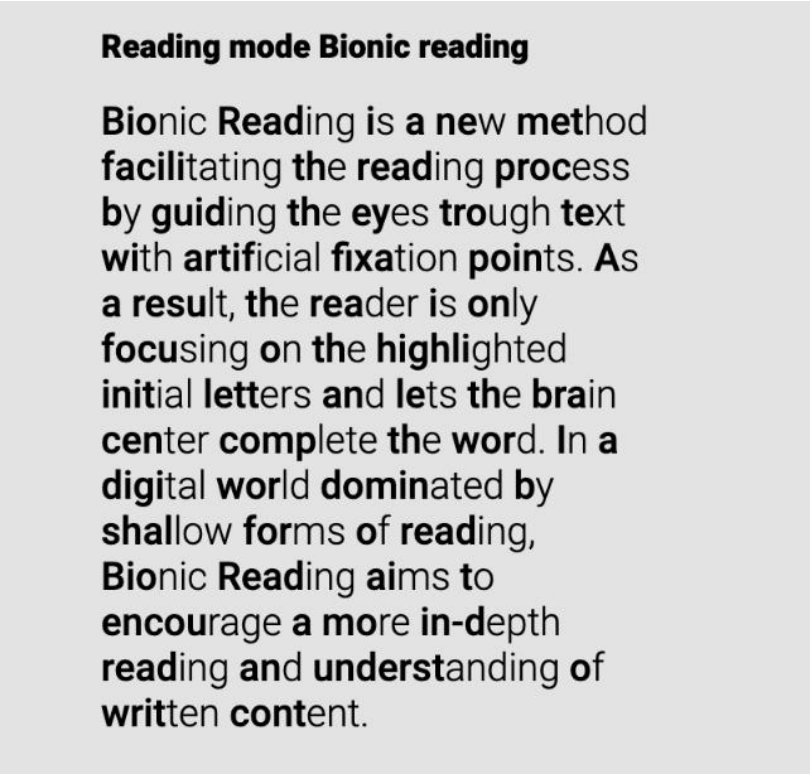

Рисунок 1.3 – Приклад тексту для біонічного читання

1.3 Огляд існуючих застосунків для читання книг

В магазинах застосунків присутня велика кількість програм для читання (табл. 1.1). Зазвичай вони націлені на 3 основні категорії:

– особисте користування;

– взаємодію між батьками та дітьми;

– навчальний процес.

Таблиця 1.1 – Порівняльна характеристика застосунків для читання

Назва	**Бібліотека**	**Тести**	**Родинна взаємодія**	**Статистика**
epic! [23]	Заготовлена	Заготовлені	Перегляд статистики	Години читання
Spark Reading [24]	Заготовлена	Заготовлені	Націлено на викладачів	Години читання
Readability [25]	Заготовлена	Заготовлені	Перегляд статистики	Години читання Швидкість та точність
Khan Academy Kids [26]	Заготовлена	Немає	Немає	Немає
Glose [27]	Заготовлена	Немає	Особисте використання	Кількість прочитаних сторінок, прочитані книги
Rork [28]	Немає	Немає	Націлена на індивідуальних читачів	Швидкість читання, період читання, час

Застосунки другої та третьої групи часто мають якісь інтерактивні елементи: ілюстрації, тести, використання розпізнавання мови для читання. Але в них є 4 основні проблеми:

– не надають можливості завантажувати свої книги, тому бібліотека книг обмежена тим, що додали розробники. Окрім того, що книги можуть бути

не цікавими для дитини або потрібна книга буде відсутня, такі застосунки підійдуть дітям не усіх регіонів, оскільки можуть не мати потрібної мови;

– відсутність динамічно створеного інтерактивного контенту. Усі тести є підготовленими заздалегідь, а деякі книги їх просто не мають тестів;

– недостатня взаємодія між батьками та дітьми. Наприклад, не можна вказати пріоритет читання, слідкувати за прогресом, давати завдання, налаштовувати графік читання, тощо;

– недостатня статистика. Для більшості користувачів це не є дуже важливим функціоналом, або він не потрібен у повній мірі, але за умови впровадження в освітній процес мати детальну статистику важливо для відстежування прогресу всіх дітей і підлаштовування цього самого процесу.

Приклади роботи застосунку epic! показано на рисунку 1.4.

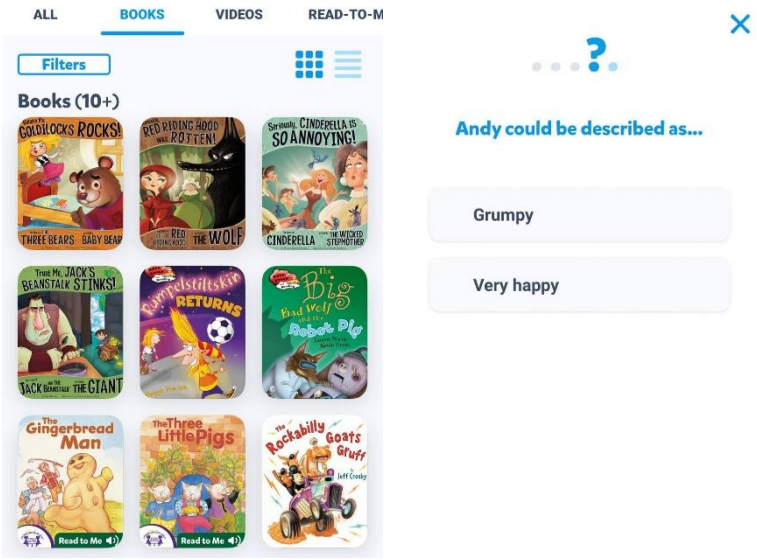

Рисунок 1.4 – Приклади бібліотеки та тесту в застосунку epic!

Поточний стан на ринку подібних застосунків показує відсутність універсального застосунку для створення інтерактивного контенту на основі книг. Усі застосунки, в яких присутня можливість проходити тести, не дають можливості завантажувати свої книги. З цього можна зробити висновок, що тести є заготовленими заздалегідь. Протилежна ситуація у застосунках, де можна завантажувати свої книги, а саме, вони не надають функціоналу

автоматичного створення тестів на основі книг і здебільшого націлені на просте читання.

Подібний застосунок, з можливістю завантаження книг та автоматичним створенням тестів на їх основі, можна було б використовувати як в сімейному колі, так і в навчальному процесі для заохочення дітей до читання.

1.4 Програмне забезпечення та технології для створення мобільних застосунків та застосунків штучного інтелекту

В сучасних реаліях розробки програмного забезпечення існує велика кількість технологій, фреймворків та іншого програмного забезпечення (ПЗ), яке полегшує та пришвидшує створення застосунків.

В останні роки React займає перші позиції серед бібліотек та фреймворків для створення вебзастосунків. Розробники React активно розвивають свою бібліотеку, а також займаються розробкою інших продуктів. Одним з них є React Native – бібліотека, яка дозволяє створювати мобільні застосунки, використовуючи звичний синтаксис React, всі його методики, мати доступ до великої кількості JavaScript (JS) бібліотек, а також компілювати застосунок як для Android, так і для iOS, використовуючи один і той самий програмний код. Тому технології React є гарним варіантом для створення клієнтської частини застосунків з використанням штучного інтелекту.

Моделі штучного інтелекту поділяються на 2 категорії за типом використання:

- використання локально запущеної моделі;
- використання API сервісів.

До моделей першої категорії частіше всього відносяться моделі для задач комп’ютерного зору, розпізнавання тексту, озвучення тексту, тощо. Тобто моделі, які не потребують великої кількості ресурсів для роботи. Такі

моделі можуть бути запущені як на серверах з ціллю постобробки, так і локально на пристроях користувачів з метою виконання задач в реальному часі. Наприклад, бібліотека opencv-js дозволяє використовувати OpenCV з коду JavaScript як в вебзастосунках, так і в мобільних.

До моделей другої категорії здебільшого відносяться мовні моделі, проте туди можна віднести також і моделі комп'ютерного зору, розпізнавання тексту, озвучення тексту, тощо. Сучасні мовні моделі потребують великої кількості ресурсів, тому такі компанії як Google, Anthropic, OpenAI та інші надають доступ до своїх моделей, які запущені на їх серверах. Для доступу до них вони надають бібліотеки, через які можна взаємодіяти з моделями. Бібліотеки є для різних мов, проте найпопулярнішою мовою для цих бібліотек є Python. Тому є сенс створювати застосунки з серверною частиною повністю на цій мові або створювати мікро-сервіси на ній для роботи з моделями. Також часто бібліотеки роблять для мови JavaScript, що дає можливість працювати з моделями напряму з вебзастосунків або з мобільних застосунків. Використання бібліотек робить роботу з мовними моделями набагато зручніше і швидше, проте якщо модель потрібно використовувати з мови, на якій немає підтримки бібліотеки, завжди можна відправляти запити на відповідні API адреси з використанням звичних для мови засобів бо бібліотек.

Також є різні бібліотеки та фреймворки, які спрощують роботу з моделями для деяких задач, надаючи різний готовий функціонал. Одним з таких фреймворків є LangChain [29]. Він надає єдиний інтерфейс для роботи з різними моделями, велику кількість інструментів, наприклад, для читання різних форматів файлів, зчитування даних з різних сайтів, форматування та перетворення даних різних типів, інтерфейси баз даних, об'єднання цих інструментів в ланцюги для простого використання і багато чого іншого. Все це допомагає швидко створювати застосунки з використанням ШІ для різних задач.

1.5 Постановка задачі

Таким чином, на сьогодні існує потреба у застосунках, що пов’язані з заохоченням молоді до читання, поліпшенням розуміння прочитаного матеріалу. Такі застосунки можуть використовуватися у родині, де у батьків була б змога завантажувати потрібні книжки, ставити завдання дитині щодо об’єму, який необхідно прочитати, та мати можливість автоматично оцінити розуміння дитиною прочитаного матеріалу та передивитися статистику.

Для моніторингу розуміння прочитаного матеріалу в роботі запропоновано використовувати автоматично згенеровані тести за допомогою LLM.

Об’єктом роботи є питання використання мовних моделей для генерації тестів з метою моніторингу розуміння прочитаного матеріалу.

Мета роботи – розробка мобільного застосунку для моніторингу розуміння прочитаних художніх творів, призначеного для використання в родині.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз поточного стану ринку споживання книг, проаналізувати рівень читацької активності молоді та роль цифрових технологій у формуванні мотивації до читання;

– оглянути існуючі застосунки для читання, їхні можливості, сфери застосування, визначити переваги та недоліки;

– ознайомитися з досягненнями штучного інтелекту та сфери його застосування, визначити можливості штучного інтелекту та інших інновацій для покращення читацьких навичок в освітніх застосунках;

– оглянути сучасні технології, які допомагають створювати мобільні застосунки з використанням штучного інтелекту та мовних моделей;

– визначити мету та спосіб використання LLM для покращення читацьких навичок;

– дослідити питання використання LLM для генерації тестів та обрати модель, яка за параметрами якості, швидкості та вартості як найкраще задовольняє вимогам мобільного застосунку для аналізу розуміння прочитаних художніх творів:

1) розробити запит, за допомогою якої модель буде генерувати тести на основі тексту книги;

2) сформувати набір даних для проведення експериментів щодо дослідження LLM на предмет створення тестів;

3) оцінити час генерації тестів;

4) провести тестування за участі експертів з метою визначення кращої моделі для даної задачі;

5) розглянути питання покриття великого обсягу тексту мовними моделями з використанням ін’єкцій;

– визначити набір технологій та розробити архітектуру майбутнього мобільного застосунку;

– розробити мобільний застосунок для моніторингу розуміння прочитаних художніх творів.

2 ДОСЛІДЖЕННЯ LLM ДЛЯ ВИРІШЕННЯ ЗАДАЧІ МОНІТОРИНГУ РОЗУМІННЯ ПРОЧИТАНОГО МАТЕРІАЛУ ТА СТВОРЕННЯ ДОСЛІДНИЦЬКОГО ЗАСТОСУНКУ З ВИКОРИСТАННЯ GRADIO ТА HUGGING FACE

2.1 Мета використання великих мовних моделей для покращення читацьких навичок

Метою використання LLM в даній роботі є генерація тестів на основі тексту книги з метою покращення читацьких навичок, розуміння та запам’ятовування прочитаного матеріалу. Даний підхід можна використовувати для особистого користування, в колі сім’ї та в навчальному процесу, з метою його покращення.

2.2 Напрямки досліджень для визначення відповідної моделі

З розвитком LLM з’являються і методи оцінювання їх якості, що включають в себе автоматичні метрики, такі як BLEU, BERTScore, MAUVE та інші, набори еталонних даних, такі як GLUE, MMLU, HumanEval та інші. Проте вони націлені на загальні здібності моделі.

Мовні моделі почали використовувати в різноманітних задачах для яких часто складно знайти відповідні метрики. Ще одним з методів оцінювання якості генерації мовних моделей є людська оцінка, що включає в себе порівняльні оцінки, абсолютні оцінки, використання шкал Ліберта, ранжування, тощо. Тому для подібних випадків, коли важко оцінити якість генерації моделей наявними метриками, можна залучити людей, так званих експертів, до тестування моделей.

Задача генерації тестів якраз відноситься до випадків, коли важко підібрати відповідну метрику для порівняння моделей, тому для дослідження

і збору статистики якості генерації тестів буде залучено 85 експертів, а саме учнів 7, 9 та 10 класів.

Не менш важливим показником при виборі мовної моделі є ціна. Загальна ціна за генерацію залежить від кількості інформації, яка подається на вхід та кількості інформації, яку модель надає на виході. Цінобудова за вхідні та вихідні дані залежить від самої моделі і визначається її провайдером.

Швидкість також є важливим показником, особливо в задачах, де взаємодія між користувачем та мовною моделлю очікується в реальному часі. Швидкість генерації залежить першочергово від самої моделі, а також від кількості даних, які їй треба обробити перед початком генерації, та кількості даних, які їй треба генерувати. Публічні мовні моделі, наприклад, від OpenAI, Google чи Anthropic, робота з якими здійснюється через API сервіси, інколи мають перевантаження в залежності від дня або години доби, що впливає на швидкість аналізу та генерації даних. Тому важливо зауважити, що, за умови одних і тих самих даних на вхід та вихід, одна й та сама модель може показувати різний час генерації.

Таким чином ці три показники, а саме, якість, ціна та швидкість, є основними при виборі оптимальної моделі, незалежно від типу задач, і їх буде використано в даній роботі для вибору найбільш відповідної мовної моделі для генерації тестів на основі тексту книг.

2.3 Огляд великих мовних моделей

LLMs з кожним роком набуває все більшого і більшого розвитку, нові моделі виходять все частіше, а старі втрачають актуальність.

Для задачі генерації тестів на основі тексту книги, за даними відкритих джерел [30, 31] станом на 25 лютого 2025 року, було відібрано 5 популярних моделей, які показували гарні результати в різних задачах, високо оцінювались користувачами та не мали занадто високої вартості (табл. 2.1).

Таблиця 2.1 – 5 відібраних моделей з характеристиками якості, швидкості та ціни

Модель	**Artificial Analysis Intelligence Index**	**Швидкість (tk/s)**	**Ціна вхідних токенів (USD/1m)**	**Ціна вихідних токенів (USD/1m)**	**LMSys Chatbot Arena ELO Score**
Gemini 2.0 Flash	48	179	0,1	0,4	1359
Claude 3.7 Sonnet	48	79	3	15	1313
Gemini 1.5 pro	45	59	1,25	5	1302
Claude 3.5 Sonnet	44	81	3	15	1283
GPT 4o	41	48	2,5	10	1377

Artificial Analysis Intelligence Index (Індекс інтелекту штучного аналізу) – комбінована метрика, запропонована сайтом Artificial Analysis, яка включає в себе кілька популярних метрик, таких як: GPQA Diamond, Humanity's Last Exam, LiveCodeBench, SciCode, AIME, MATH-500, MMLU і тд. Швидкість моделей вимірюється у кількості токенів, які вони можуть генерувати за секунду. Ця метрика напряму впливає на швидкість генерації відповідей і є важливою для деяких задач з використанням мовних моделей.

Використання публічних мовних моделей не є безкоштовним, тому провайдери моделей стягують гроші за кількість токенів, які були подані на вхід моделі, та кількість токенів, які були повернені моделлю.

Chatbot Arena – вебсайт, на якому представлена велика кількість мовних моделей, і який надає можливість користувачам платформи порівнювати моделі між собою. Використовуючи ці дані, будується рейтинг, на який можна спиратись під час вибору моделі.

2.4 Розробка запиту та його роль у генерації тестових запитань

Найважливішою частиною в задачах з використанням LLM є запит. Запит (prompt) – це інструкція, яка надається моделі штучного інтелекту, щоб

дати вказівки для її відповіді. Запит фактично визначає, яку інформацію та в якому вигляді має генерувати модель. В залежності від задачі, запит може складатись з простих запитань або містити складні інструкції та приклади. Якість відповіді моделі напряму залежить від якості підготовки запиту. Створення запитів для конкретних задач при роботі зі штучним інтелектом називають інженерією запитів (Prompt engineering).

Для задачі генерації тестів на основі тексту книги запит складається з таких елементів:

– роль моделі, ким вона є, яка її мета та задача. Це допомагає моделі краще зрозуміти з чим вона працює і що від неї очікується;

– частина тексту, на основі якої буде згенеровано тест. В інженерії запитів це називається контекст, тобто додаткові дані, які потребує модель для роботи;

– можливі теми для запитань, наприклад, зовнішній вигляд персонажів, їх взаємозв’язки або події в тексті;

– правила для генерації тестів, наприклад, кількість запитань, кількість відповідей, мова, на якій потрібно генерувати та інші додаткові правила, які допомагають отримати очікуваний результат;

– кроки для постановки питання, де описується, що модель повинна вибрати декілька ключових моментів з тексту, теми запитань, які до них підходять, пояснити чому ці питання гарні та вибрати найкраще;

– формат відповіді. Ця частина є дуже важливою, оскільки, якщо не обмежити модель у форматі відповіді, то вона буде використовувати довільний, подальшу роботу з яким буде вести неможливо. Варто зауважити, що формат може бути різним і відрізнятися в залежності від задачі, очікуваного результату та подальших дій із відповіддю. Для цієї задачі використовується JavaScript Object Notation (JSON) формат, який дозволяє в подальшому зручно працювати з об’єктами.

Також для досягнення кращих результатів існують додаткові правила інженерії запитів:

– використання розміток, таких як Markdown, EXtensible Markup Language (XML), JSON та інші для форматування структури запиту;

– форматування тексту, яке включає в себе перенесення строк, пунктуацію, виділення важливих моментів великими літерами, нумерацію списків, розділення елементів запиту на логічні блоки, тощо. Запит повинен бути чітко структурований, легко читатись та розумітись;

– розставляти елементи запиту в правильній послідовності, оскільки це також може впливати на результати. Загалом для отримання гарних результатів достатньо розставити елементи за логічним порядком, проте за потреби можна провести додаткові експерименти, розставляючи елементи в різній послідовності.

В рамках цієї роботи для створення запиту було використано Markdown та XML. За допомогою Markdown можна чітко структурувати тест, в особливості списки, а XML допомагає в тих випадках, коли в процесі написання запиту потрібно посилатись на якісь його частини, наприклад, в даному випадку список тем, частина тексту та JSON формат для відповіді.

Інколи модель може забути правила, які були їй надані на початку запиту, тому в самий його кінець можна продублювати ці правила, наприклад, написавши «Запам’ятай» або «Не забудь». Також сюди можна вставити різні додаткові правила, які було складно віднести до якоїсь окремої частини.

Моделі потрібно давати більше конкретики. Якщо очікується конкретний формат – треба його описати, якщо потрібно дотримуватись якихось правил із однієї генерації в іншу – ці правила також потрібно описати. Чим більш конкретним є запит, тим більш якіснішою та очікуванішою буде відповідь моделі.

На рисунку 2.1 показано приклад роботи моделі за розробленим запитом. Можна побачити процес створення запитань, який складається з виділення ключових частин, вибору тем запитань для них та вибору найкращого. Також можна побачити, як модель обгортає різні частини своєї

відповіді в XML форматування. Це допомагає зробити пост обробку відповіді, а саме отримати JSON з запитанням та відповідями.

```
<question_development>
1. Key quotes from excerpt:
- "Карпо був широкий в плечах, з батьківськими карими гострими очима, з блідуватим лицем."
- "Лаврінове молоде довгасте лице було рум'яне. Веселі сині, як небо, очі світились привітно й ласкаво."
- "Веселому, жартовливому меншому братові хотілось говорити; старший знехотя кидав йому по кілька слів."

2. Relevant topics:
- Topic 1: questions about the personalities of the characters
- Topic 2: questions about the appearance or look of characters
- Topic 5: the attitude of some characters to other characters

3. Potential questions evaluation:
a) Question about brothers' physical differences
- Too straightforward, might not engage critical thinking

b) Question about brothers' personality contrast
- More complex, requires understanding of character traits
- Allows for clear distinction between answers
- Supported by multiple text examples
- Age-appropriate and encourages comprehension

4. Selected best question:
Focusing on the personality contrast between brothers as it:
- Combines physical and personality descriptions
- Shows clear character differences
- Has strong textual support
- Appropriate difficulty level for children
</question_development>

<json_format>
{
    "question": "Як відрізнявся характер Лавріна від характеру Карпа?",
    "textPart": "Веселому, жартовливому меншому братові хотілось говорити; старший знехотя кидав йому по кілька слів.",
    "answers": [
        {
            "answer": "Лаврін був веселий і говіркий, а Карпо – неговіркий і похмурий",
            "isCorrect": true
```

Рисунок 2.1 – Приклади роботи моделі за розробленим запитом

2.5 Оцінка часу, необхідного для підготовки тестів

З кожною ітерацією моделі стають дедалі швидшими, але, нажаль, їх відповідь не є моментальною. Робота моделей з великою кількістю інформації, наприклад, текстами книг, є відносно довгим процесом оскільки моделі потрібен час для аналізу тексту та для того, щоб згенерувати відповідь.

Для проведення тестування було попередньо згенеровано тести на основі 4 творів (табл. 2.2) зі шкільної програми. Для кожної книги було згенеровано по 10 запитань, використовуючи один запит для усіх моделей з

метою забезпечення рівних умов. Зафіксовані результати часу, витраченого на генерацію тестів, представлено в таблиці 2.2. Також для подальшого вибору моделі було зафіксовано і вартість генерації тестів (табл. 2.3).

Таблиця 2.2 – Час генерації тестів

Модель	**Захар Беркут**	**Катерина**	**Сон**	**Кайдашева сім'я**
Gemini 1.5 Pro	02:14	02:01	02:13	02:21
Gemini 2.0 Flash	01:17	00:58	00:57	00:59
Claude 3.5 Sonet	02:47	01:52	01:33	02:20
Claude 3.7 Sonet	03:15	03:01	02:54	04:07
GPT 4o	04:02	02:28	02:34	02:26

Таблиця 2.3 – Вартість генерації тестів у доларах

Модель	**Захар Беркут**	**Катерина**	**Сон**	**Кайдашева сім'я**	**Сумарні витрати**
Gemini 1.5 Pro	0,06	0,06	0,06	0,09	0,27
Gemini 2.0 Flash	0,01	0,01	0,01	0,01	0,03
Claude 3.5 Sonet	0,15	0,14	0,13	0,22	0,64
Claude 3.7 Sonet	0,26	0,22	0,21	0,35	1,04
GPT 4o	0,13	0,11	0,11	0,17	0,52

2.6 Експертна оцінка якості згенерованих запитань та відповідей

2.6.1 Мета дослідження

Метою проведення експериментів із залученням експертів є збір оцінок та статистики з метою вибору моделі, яка найкраще підходить для задачі генерації тестів на основі тексту книги.

Для оцінювання якості генерації моделей було використано абсолютне оцінювання за показниками: цікавість запитань, коректність запитань та коректність відповідей. Для оцінювання експертам було надано шкалу Ліберта із зсувом, а саме від -2 до 2, де: -2 – дуже погано, -1 – погано, 0 – задовільно,

1 – добре, 2 – відмінно. Така шкала є більш зрозумілою для користувача, порівняно зі шкалою від 1 до 5, оскільки має чітке розділення на негативний та позитивний досвід використання.

За результатами оцінювання буде отримано такі результати: середня оцінка за кожним показником та за всіма, медіана, 75-й персентиль, мінімальні та максимальні значення. Також під час підготовки тестів буде зібрано статистику з часу та вартості генерації тестів. Усі ці показники будуть залучені під час вибору моделі, яка найбільше підходить для задачі генерації тестів.

2.6.2 Можливості спільного використання Gradio та HuggingFace для проведення експериментів у сфері штучного інтелекту

Gradio – це Python бібліотека з відкритим вихідним кодом, яка дозволяє швидко і легко створювати вебсторінки для демонстраційних версій застосунків, тестування моделей машинного навчання та будь-якого іншого Python коду. Gradio надає можливість розробникам конструювати вебзастосунки, не вимагаючи глибоких знань у веброзробці, забезпечуючи їх великою кількістю готових компонентів. Таким чином Gradio значно пришвидшує початкові етапи створення застосунків з використанням інтелектуальних систем, спрощує комунікацію між технічними та нетехнічними фахівцями. Завдяки легкості використання та широким можливостям, Gradio є корисним інструментом для розробників, які прагнуть швидко створювати прототипи та отримувати зворотній зв’язок від користувачів.

HuggingFace – це вебсайт, на якому зібрано велику кількість моделей машинного навчання та різноманітні набори даних різного характеру. Проте однією з найцікавіших та найпопулярніших сервісів цієї платформи є HuggingFace Spaces.

HuggingFace Spaces надає функціонал хостингу вебзастосунків, створених за допомогою бібліотек Gradio або Streamlit. За принципом роботи обидві бібліотеки дуже схожі, проте більшість користувачів надає перевагу Gradio через більшу кількість функціоналу та більшій можливості налаштування візуальної частини. Запущеним на HuggingFace Spaces застосунком можна легко поділитись з іншими людьми, так як він не потребує ніякої реєстрації для використання подібних застосунків [32].

2.6.3 Проєктування застосунку для тестування якості генерації тестів

Для створення платформи, де учні могли б проходити тестування, комбінація технологій Gradio і HuggingFace Spaces є найкращим варіантом. Застосунок повинен мати функціонал для конфігурації тестів (рис. 2.2), їх завантаження та проходження (рис. 2.3), збирати відгуки про тести (рис. 2.4), які будуть зберігатися в базу даних, а також показувати експертам результати тестування (рис. 2.5).

Весь цей функціонал можна реалізувати за допомогою мови програмування Python, а інтерфейс розробити з використанням бібліотеки Gradio.

Сам же застосунок буде розміщено на платформі HuggingFace Spaces. Це дасть можливість завантажити усі необхідні для тестування дані та поширити застосунок серед експертів. Безкоштовних ресурсів, які надає HuggingFace Spaces, вистачає для того, щоб витримувати навантаження для проведення подібних експериментів.

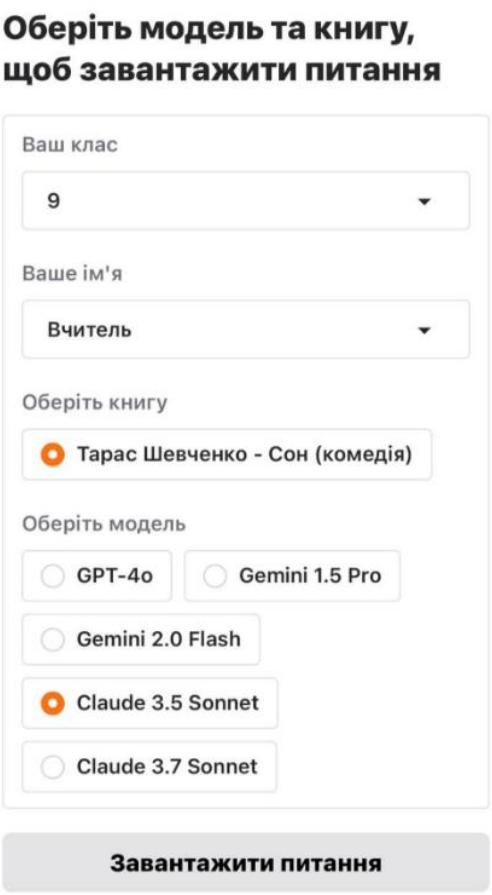

Рисунок 2.2 – Процес конфігурації тесту

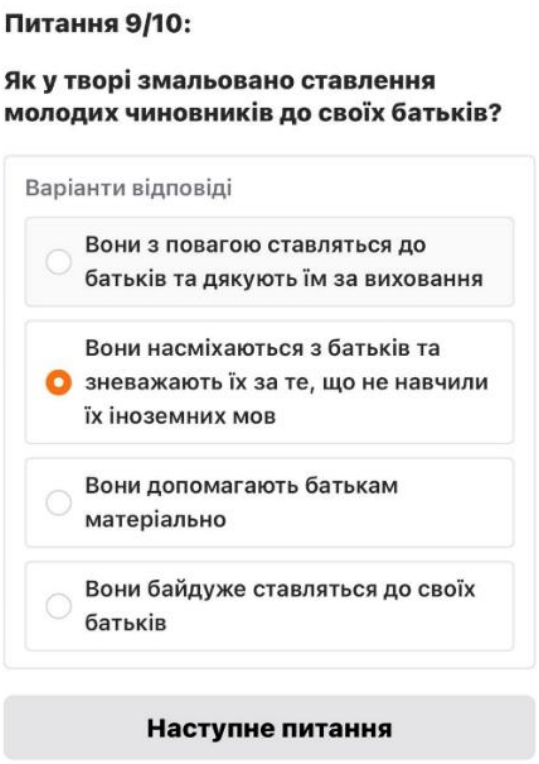

Рисунок 2.3 – Процес проходження тесту

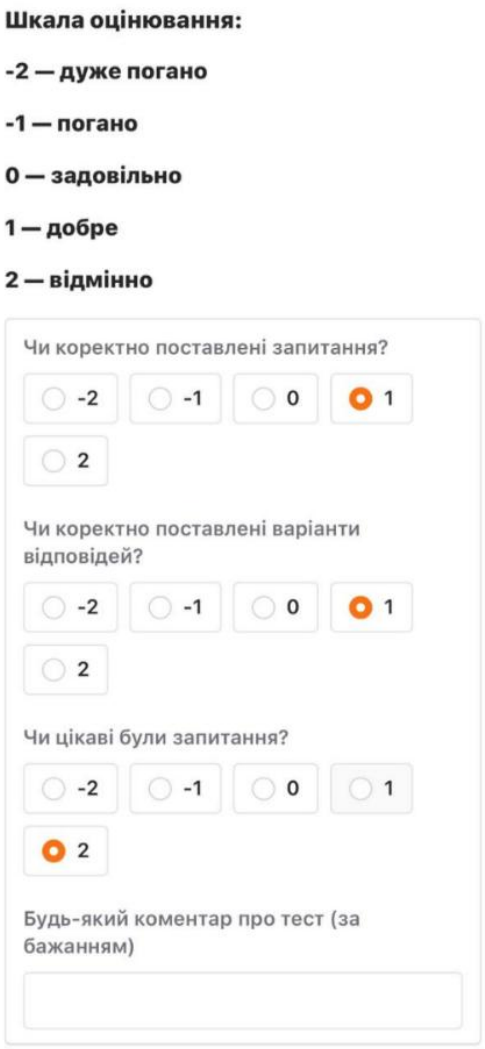

Рисунок 2.4 – Процес залишання відгуку

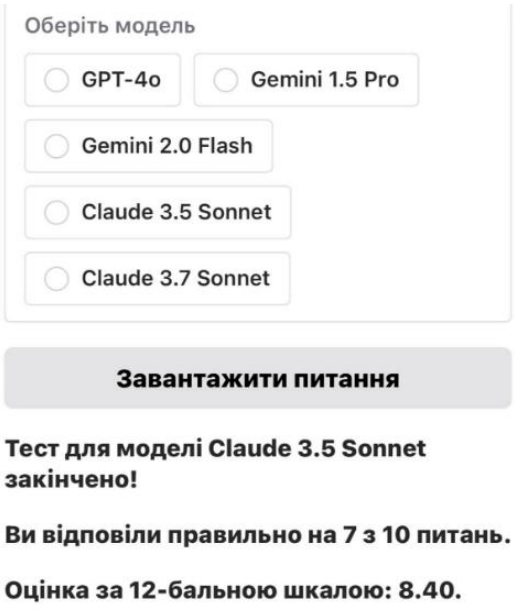

Рисунок 2.5 – Результати тестування

2.6.4 Опис та підготовка матеріалу для експертного оцінювання

Оскільки тестування проводилось серед школярів 7-10 класів, твори для тестування було обрано відповідно до їх шкільної програми (табл. 2.4). Список обраної літератури включає в себе 2 повісті та 2 поеми. Кількість сторінок повістей менша за оригінальну, оскільки в шкільній програмі ці твори подаються в скороченому варіанті.

Таблиця 2.4 – Відібрані книги для генерації тестів

Книга	**Кількість сторінок**
Іван Франко - Захар Беркут	10
Тарас Шевченко - Катерина	20
Тарас Шевченко - Сон	15
Іван Нечуй-Левицький - Кайдашева сім'я	27

Перед генерацією тестів текст кожного твору було поділено на 10 рівних частин. Відповідно для кожної частини запускалась генерація за допомогою усіх 5 моделей. Таким чином, загалом було згенеровано 20 тестів, кожен з яких містить по 10 запитань. Приклади запитань та відповідей до них наведено в таблиці 2.5.

Таблиця 2.5 – Приклади запитань та відповідей, згенеровані за допомогою Claude 3.7 Sonet

Запитання	**Варіанти відповідей**
1	**2**
Що сталося зі святим каменем у сні Захара Беркута?	Він засяяв яскравим світлом.
	Він перетворився на живу істоту.
	Він рушив з місця і впав на Захара.
	Він розколовся на дрібні шматки.

Продовження таблиці 2.5

1	2
Чому Мирослава не вважає Тугара Вовка своїм батьком?	Бо він помер.
	Бо він зрадив свій край і пристав у службу монголів.
	Бо він відмовився від неї.
	Бо він покинув її в дитинстві.

2.6.5 Результати та висновки щодо експертного оцінювання моделей

За результатами тестування було зібрано 457 записів для усіх творів та моделей. На рисунку 2.6 відображено середні оцінки за усіма показниками та загальна середня оцінка.

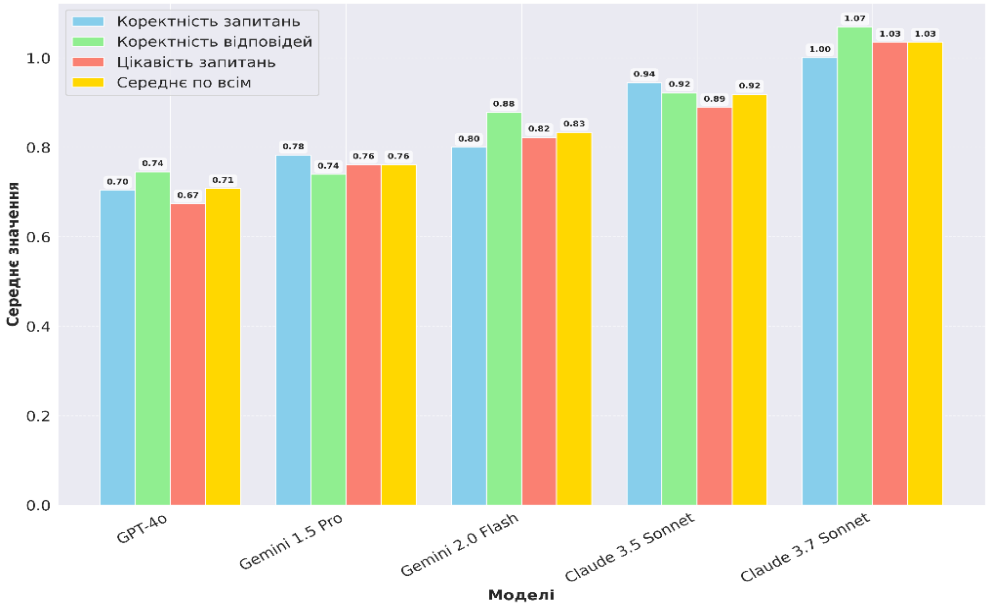

Рисунок 2.6 – Результати оцінювання

Можна побачити, що модель Claude 3.7 Sonnet займає першу позиції по оцінці якості згенерованих тестів. Проте, дивлячись на зібрану статистику про час (табл. 2.2) та вартість (табл. 2.3) генерації тестів, можна побачити, що

Claude 3.7 Sonnet є водночас і найдорожчою та витрачає більше всього часу на генерацію. Попередня версія, а саме, Claude 3.5 Sonnet, займає другу позицію за якістю генерації. Обидві моделі мають однакову ціну за вхідні та вихідні дані, проте Claude 3.5 Sonnet сумарно витратила майже в 2 рази менше коштів. Це пов'язано з процесом мислення моделі, де Claude 3.7 Sonnet генерувала більше інформації. Це також вплинуло і на час генерації - Claude 3.5 займає в 1,5-2 рази менше часу на генерацію.

В таблиці 2.6 наведено додаткові статистичні показники. Можна побачити, що для всіх моделей медіана дорівнює 1,0, 75-й персентиль має максимальне значення шкали, так само як і максимальні та мінімальні значення. Виділяється Claude 3.7 Sonnet, яка має мінімальну оцінку -1 на показнику коректність запитань, що вказує на меншу кількість помилок та кращу стабільність роботи.

Таблиця 2.6 – Додаткові статистичні показники оцінки якості

Модель	**Метрика**	**Медіана**	**75-й персентиль**	**Мін. оцінка**	**Макс. оцінка**
1	**2**	**3**	**4**	**5**	**6**
GPT 4o	Коректність запитань	1,0	2,0	-2	2
	Коректність відповідей	1,0	2,0	-2	2
	Цікавість запитань	1,0	2,0	-2	2
Gemini 1.5 Pro	Коректність запитань	1,0	2,0	-2	2
	Коректність відповідей	1,0	2,0	-2	2
	Цікавість запитань	1,0	2,0	-2	2
Gemini 2.0 Flash	Коректність запитань	1,0	2,0	-2	2
	Коректність відповідей	1,0	2,0	-2	2
	Цікавість запитань	1,0	2,0	-2	2

Продовження таблиці 2.6

1	2	3	4	5	6
Claude 3.5 Sonnet	Коректність запитань	1,0	2,0	-1	2
	Коректність відповідей	1,0	2,0	-2	2
	Цікавість запитань	1,0	2,0	-2	2
Claude 3.7 Sonnet	Коректність запитань	1,0	2,0	-2	2
	Коректність відповідей	1,0	2,0	-2	2
	Цікавість запитань	1,0	2,0	-2	2

Загалом можна сказати, що в більшості випадків усі моделі добре справляються з генерацією тестів, а основні відмінності проявляються в окремих сценаріях, наприклад, коли не моделі не вистачає контексту для якісної генерації.

2.7 Аналіз здатності покриття великого обсягу тексту мовними моделями

В сфері штучного інтелекту, а саме, для мовних моделей, часто використовують слово контекст. Контекст визначає кількість інформації, яку модель може прийняти на вхід, проаналізувати, використати для подальшої генерації контенту та вимірюється у кількості токенів. З розвитком аналітичних можливостей мовних моделей та кількістю їх знань, росте і розмір контексту, який на поточний час може досягати 100 тисяч токенів, а інколи навіть і декілька мільйонів, що є великими показниками (табл. 2.7).

Проте, маючи такі великі показники контексту, питання щодо здатності моделей нормально аналізувати таку велику кількість інформації залишається відкритим.

Таблиця 2.7 – Максимальна кількість токенів на вхід та вихід

Модель	**Максимальна кількість токенів на вхід**	**Максимальна кількість токенів на вихід**
GPT 4o	128000	16384
Gemini 1.5 Pro	2097152	8192
Gemini 2.0 Flash	1048576	8192
Claude 3.5 Sonnet	200000	8192
Claude 3.7 Sonnet	200000	64000

Для визначення аналітичних здібностей моделей було проведено тестування з використанням 5 прямих та 5 асоціативних ін’єкцій. Суть експерименту полягає у тому, щоб в наявний текст вставити додаткову інформацію, поставити моделі запитання по наданому тексту з ін’єкцією та визначити точність цієї відповіді [33].

В рамках проведення експерименту було обрано книгу «Захар Беркут». Список ін’єкцій представлено в таблицях 2.8 та 2.9.

Кожна з ін’єкція вставлялась в початок, середину та кінець книжки. Для аналізу точності прямих ін’єкцій було сформовано список очікуваних відповідей (табл. 2.8). Якщо відповідь моделі співпадала з очікуваною, вона отримувала 100% точності на даній ін’єкції. Якщо не співпадала, то очікувані відповіді розбивались на слова та підраховувалась кількість співпадінь.

Для асоціативних ін’єкцій було сформовано список ключових слів, які представляють собою корені слів з ін’єкції та питання або слова, які модель може потенційно використати в даному контексті (табл. 2.9), на основі яких рахувалась кількість співпадінь.

Окрім точності відповідей з ін’єкціями, було зібрано також і статистику щодо кількості токенів, які було отримано моделлю на вхід (рис. 2.7). В майбутньому це допоможе визначити приблизну оптимальну кількість токенів, які варто подавати моделі для кращих результатів аналізу тексту.

Таблиця 2.8 – Прямі ін’єкції

Ін’єкція	**Питання**	**Очікувані відповіді**
Максим народився в день весняного рівнодення	Коли народився Максим?	Весняного рівнодення, день весняного рівнодення
Улюбленим напоєм Захара Беркута був медовуха	Який напій любив Захар Беркут?	Медовуха, медовуху
У бою під Опором Тугар Вовк втратив свого коня	Кого втратив Тугар Вовк у бою під Опором?	Коня, свого коня
Бурунда-бегадир носив чорну шаблю з червоним руків'ям	Яку шаблю мав Бурунда-бегадир?	Чорну шаблю з червоним руків'ям, чорну, з червоним руків'ям, шаблю з червоним руків'ям, чорну шаблю
Село Тухля налічувало 137 хат	Скільки хат було в селі Тухля?	137, сто тридцять сім

Таблиця 2.9 – Асоціативні ін’єкції

Ін’єкція	**Питання**	**Ключові слова**
Захар Беркут віддав останній шматок хліба дитині з сусіднього села	Як проявлялася доброта Захара Беркута?	хліб, дитин, сусідн, віддав, допомог, доброт, щедр, турбот
Мирослава ночами слухала спів вітру у горах, згадуючи батьківщину	Що допомагало Мирославі згадувати рідний край?	спів, вітр, гор, ночами, слуха, звук, природ, згадува
Максим йшов босоніж через сніг, лишаючи за собою криваві сліди	Як Максим проявив витривалість?	босоніж, сніг, кривав, слід, йшов, через, холод, біль
Старі тухольці називали тіснину священним порогом між двома світами	Яке духовне значення мала тіснина для громади?	священ, поріг, світ, між, дух, віра, символ, значенн
Тугар Вовк ховав очі, коли люди говорили про честь	Як ставився Тугар Вовк до теми честі?	хова, очі, говорили, сором, уника, зрад

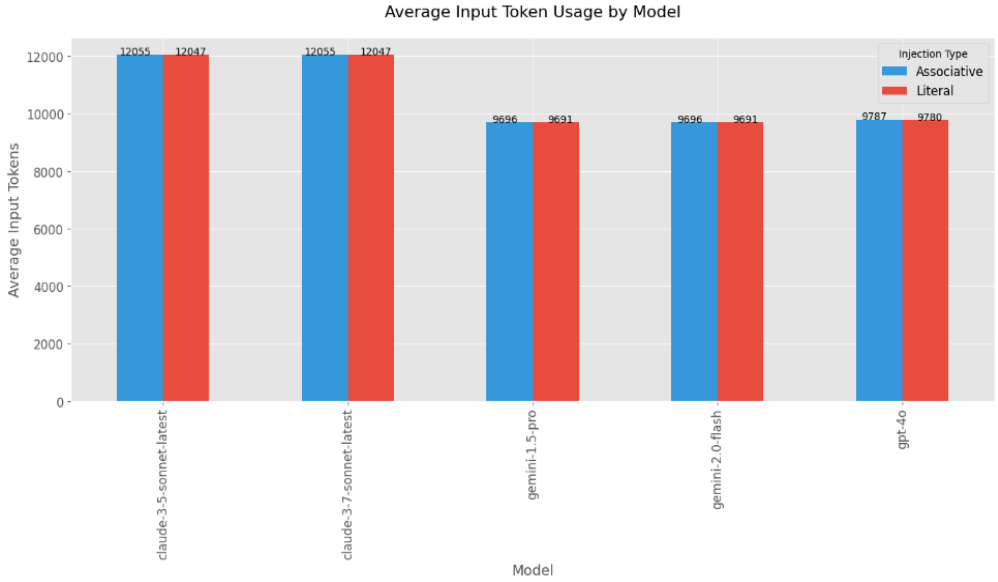

Рисунок 2.7 – Кількість токенів, отриманих моделями на вхід

Результати пошуку прямих ін’єкцій (рис. 2.8) показують, що моделі добре справляються з цією задачею. Навіть в кінці книги з великою кількістю тексту точність не опускається нижче 80%.

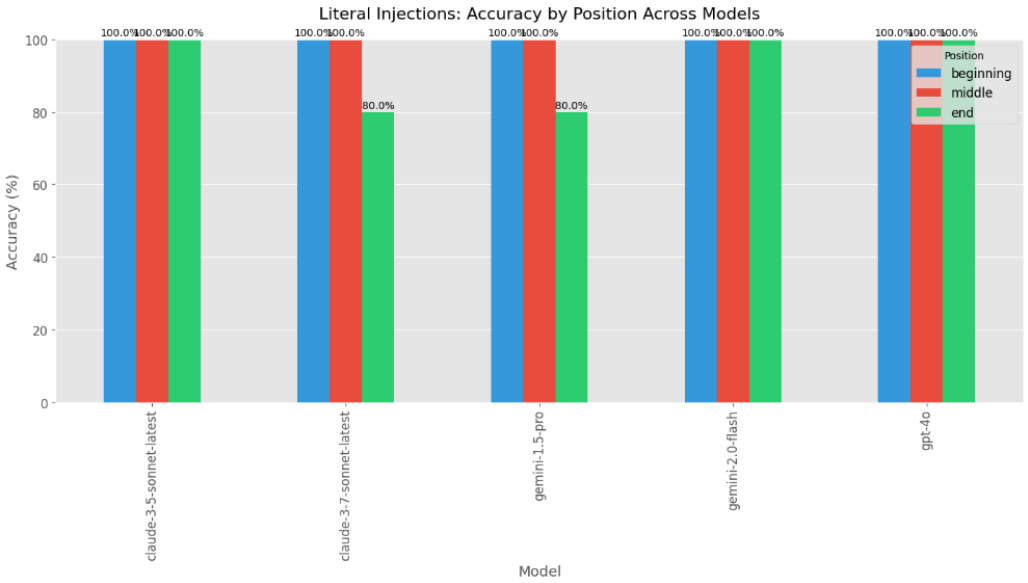

Рисунок 2.8 – Точність з прямими ін’єкціями

Проте процес генерації тестів вимагає від моделі глибшого аналізу тексту, який більше схожий на процес виявлення асоціативних ін’єкцій. В цьому ж випадку моделі показують не настільки гарні результати (рис. 2.9).

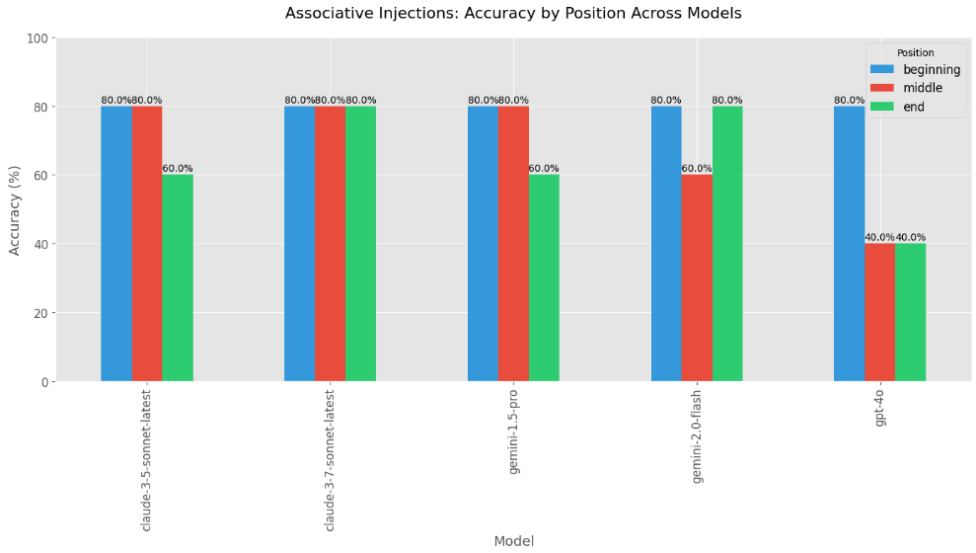

Рисунок 2.9 – Точність з асоціативними ін’єкціями

Навіть на початку тексту немає 100% точності, а до кінця тексту точність падає до 40-60%. Виділяється серед усіх моделей gemini-2.0-flash, в якої точність падає в середині тексту.

За результатами проведення тестування з ін’єкціями було виявлено, що мовні моделі показують не найкращі результати при роботі з асоціаціями в середині та кінці тексту. Тому з метою покращення якості генерації запитань для подальшого алгоритму створення тестів було запропоновано розділяти текст на декілька частин, кількість яких залежить від кількості запитань у майбутньому тесті. Таким чином модель буде отримувати невелику частину тексту, з якої буде створювати запитання, і яка буде в знаходитись рамках припустимої кількості токенів, яку модель може нормально аналізувати.

2.8 Висновки щодо визначення найбільш відповідної моделі

За результатами підготовки даних для тестування було виявлено, що генерація тесту з 10 питаннями та обсягом тексту від 10 до 27 сторінок займає від 1 до 4 хвилин в залежності від моделі. Враховуючи майбутню архітектуру застосунку, де тести будуть підготовлюватись заздалегідь і генеруватись в

фоновому режимі, показник швидкості не є впливовим для фінального вибору відповідної моделі.

Таким чином, для вибору залишається 2 показники: якість та ціна. Лідерство за якістю займають моделі від Claude, проте вони є і найдорожчими. Використовуючи ці моделі, майбутньому застосунку доведеться стягувати гроші для існування. Навіть за умови стягування середньої вартості підписки у розмірі 5 доларів, ця сума покриє генерацію до 40 тестів для Claude 3.5 Sonnet та до 20 тестів для Claude 3.7 Sonnet, що не є гарним показником прибутковості.

Наступна за якістю, а саме Gemini 2.0 Flash, яка займає 3 місце по якості генерації тестів, має доволі непогані показники. За показником якості вона відстає всього на 5% від Claude 3.7 Sonnet, проте витрачає в 34 рази менше коштів.

Таким чином, за найбільш оптимальну мовну модель для задачі генерації тестів на основі тексту книги було обрано Gemini 2.0 Flash і подальша розробка мобільного застосунку буде вестись з нею.

3 РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ МОНІТОРИНГУ РОЗУМІННЯ ПРОЧИТАНИХ ХУДОЖНІХ ТВОРІВ

3.1 Функціональна специфікація застосунку

В застосунку присутні 2 головні ролі, а саме, батько та дитина. Функціональні можливості користувачів з цими ролями зображено на рисунках 3.1 та 3.2.

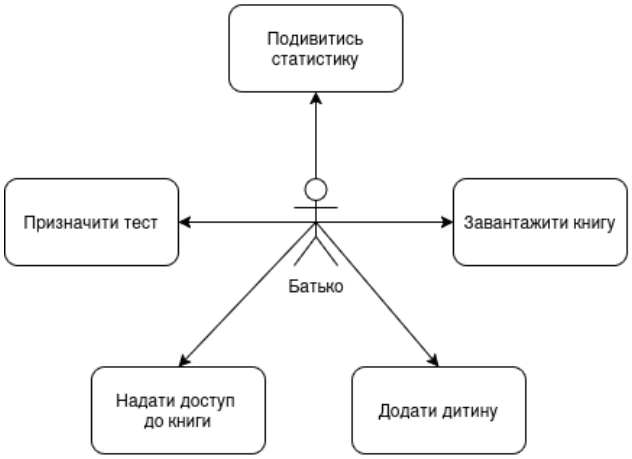

Рисунок 3.1 – Функціональні можливості користувача з роллю «Батько»

Рисунок 3.2 – Функціональні можливості користувача з роллю «Дитина»

Основні функції користувача з роллю «Батько» включають в себе:

– завантаження книг;

– додавання дітей;

– надання доступу до книг;
– призначення тестів;
– перегляд прогресу читання дітей;
– перегляд оцінок дітей;
– перегляд статистики.

Користувачі з роллю «Дитина» мають такий функціонал:

– читання книг;
– проходження тестів;
– перегляд свого прогресу читання;
– перегляд своїх оцінок;
– перегляд своєї статистики.

Для використання застосунку, батько повинен зареєструватись у ньому, після чого додати дітей. Перемикання на роль дитини виконується з бокового меню застосунку.

3.2 Проєктування архітектури застосунку

Сучасні діти ознайомлюються з технологіями у ранньому віці, особливо зі смартфонами, тому для багатьох із них використання телефонів та застосунків на них не викликає труднощів.

Таким чином в цій роботі застосунок буде створено для мобільних платформ.

3.2.1 Користувацька частина застосунку

Найбільш популярними операційними системами для мобільних пристроїв є Android та iOS. В останніх реаліях розробки програмного забезпечення найбільш популярною мовою програмування для Android є

Kotlin, а для iOS – Swift. Тому зазвичай створення застосунків для цих платформ потребує від розробників знання цих двох мов та, відповідно, усіх їх особливостей.

React – бібліотека, створена компанією Meta з метою оптимізації та покращення процесу створення вебзастосунків. Бібліотека вже довгий час користується великою популярністю і багато веброзробників працюють з нею. В 2015 році Meta представила новий продукт – React Native. Цей фреймворк надає можливість створювати мобільні застосунки, використовуючи вже звичні для розробників методики зі звичайного React. Окрім того, що процес створення застосунків полегшується, код, написаний за допомогою React Native, можна виконувати як на Android та і на iOS пристроях. Таким чином від розробників не вимагається знання різних мов програмування і, що найважливіше, це полегшує підтримку застосунків, оскільки додавання нового функціоналу відбувається відразу на 2 платформи.

Часто в парі з React Native використовують платформу Expo, яка додатково полегшує створення мобільних застосунків, керуючи усіма процесами зборки застосунку для iOS та Android, надає додаткові бібліотеки та бібліотеки, які розширюють функціонал стандартних, а також дозволяє збирати застосунки у своєму хмарному середовищі. Все це дозволяє зосередитись на розробці застосунку і не витрачати час на додаткові дії для його створення. Таким чином, для користувацької частини застосунку буде використовуватись React Native разом з Expo.

3.2.2 Серверна частина застосунку

Специфіка даного застосунку неодмінно потребує створення серверної частини для збереження файлів книг, дітей, прогресу читання, генерації тестів та результатів їх проходження, тощо.

Основний функціонал серверної частини застосунку буде реалізовано в мікро-сервісі з використанням мови програмування Java та фреймворку Spring. Ця комбінація технологій перевірена часом та дає можливість створювати надійні та стабільні застосунки. Цей мікро-сервіс буде відповідати за збереження та отримання даних.

Усі публічні мовні моделі зазвичай мають бібліотеки для роботи з ними. Ці бібліотеки представлені для різних мов програмування, але Java не є популярним вибором для них. Найпопулярнішою мовою для подібних бібліотек є Python. Таким чином, було прийнято рішення створити другий мікро-сервіс, з використанням мови програмування Python, який буде відповідати за генерацію тестів. Також для цього мікро-сервісу було обрано використовувати фреймворк FastAPI, який дозволяє швидше обробляти запити до мікро-сервісу, а також підтримує асинхронне виконання коду, що в майбутньому дозволить пришвидшити генерацію тестів, запускаючи її в декількох потоках відразу.

Оскільки в застосунку присутні 2 мікро-сервіси, потрібно налаштувати взаємодію між ними. Обидва мікро-сервіси підтримують Representational State Transfer (REST) протокол, проте він має декілька недоліків в даній архітектурі:

– залежність від доступності обидвох сервісів одночасно, що створює вразливості та можливу втрату даних;

– важко масштабується для великих навантажень та балансування;

– зазвичай використовується для синхронної комунікації.

Для вирішення цих проблем можуть допомогти брокери повідомлень, наприклад RabbitMQ. Він забезпечує надійну асинхронну комунікацію, гарантуючи доставку повідомлень навіть під час збоїв в у системі. Це досягається тим, що усі повідомлення між мікро-сервісами відправляються не напряму, а через брокера, який зберігає усі повідомлення в своєму сховищі. Таким чином, якщо мікро-сервіс, який отримує повідомлення, з якихось причин не доступний в конкретний момент часу, то повідомлення залишається в системі і чекає обробки. При використанні REST, в такому випадку

повідомлення було б втрачено. Окрім того брокери повідомлень дозволяють балансувати навантаження на систему, розподіляючи повідомлення між декількома отримувачами.

На рисунку 3.3 зображено діаграму послідовності процесу генерації тестів, на якому можна побачити взаємодію мікро-сервісів між собою за допомогою RabbitMQ.

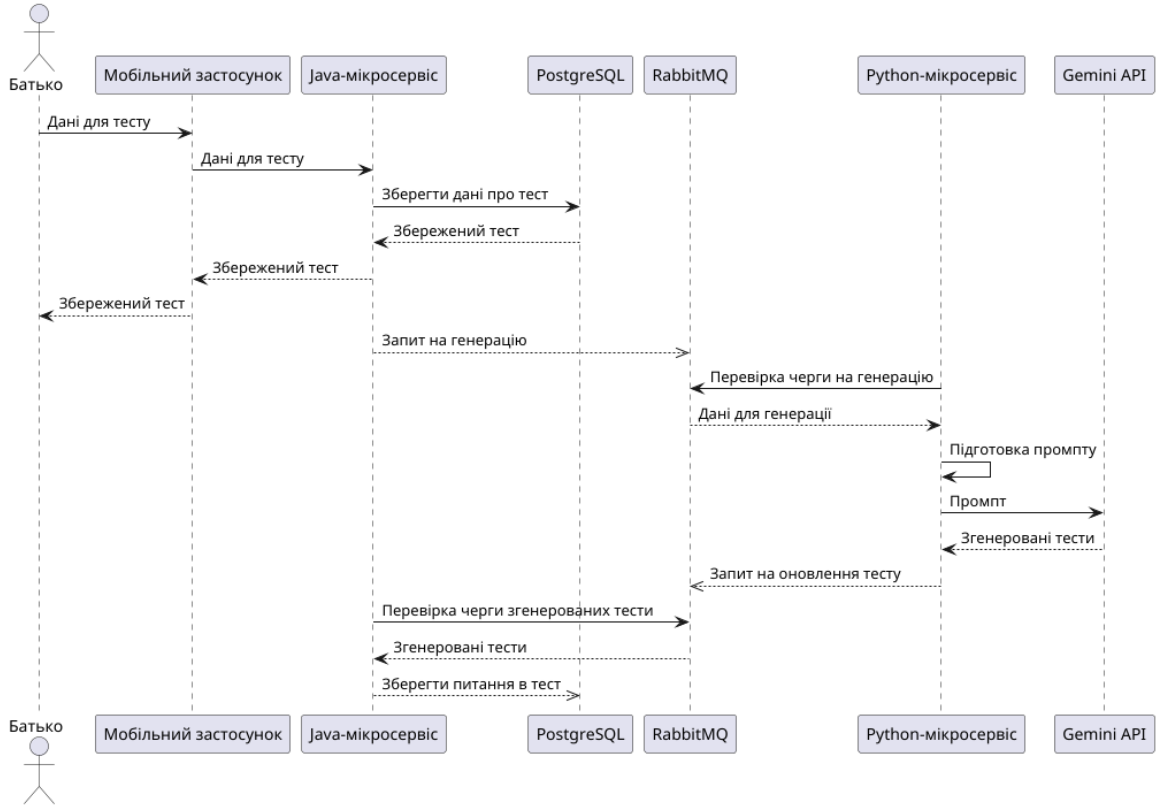

Рисунок 3.3 – Діаграма послідовності процесу генерації тестів

На рисунку зображено, що користувач, відправивши запит на генерацію тесту, не чекає повної відповіді від серверу, а отримує лише часткові дані про тест, які було збережено до бази даних. В цей же час, запит на генерацію тесту вже відправлено. Також можна побачити, що і мікро-сервіси не чекають відповідь один від одного, а лише відправляють запити. Отримання ж повідомлень виконується періодичною перевіркою потрібної черги.

При використанні брокерів повідомлень мікро-сервіси можуть відігравати ролі виробника (producer), який відправляє дані в чергу, та

споживача (consumer), який отримує дані з черги. В даному випадку, обидва мікро-сервіси відіграють відразу 2 ролі. Java мікро-сервіс відправляє запити на генерацію тестів та приймає повідомлення з уже згенерованими тестами, а Python мікро-сервіс приймає запити на генерацію тестів та підправляє повідомлення із згенерованими.

Таке розділення на мікро-сервіси має декілька переваг:

– Java та Spring дуже швидко оброблюють запити;

– Hibernate, який використовується в Spring, дозволяє швидко та оптимізовано працювати з базою даних;

– окремий мікро-сервіс на Python дозволяє зняти навантаження з основного мікро-сервісу на Java;

– використання брокерів повідомлень підвищує стресостійкість серверної частини.

Оскільки застосунок передбачає завантаження файлів книг, потрібно також продумати і механізм їх збереження. PostgreSQL надає можливість збереження файлів в базу, проте в даному випадку це не найкраще рішення, оскільки книги можуть багато важити, в залежності від типу файлу та його вмісту, що призведе до великого розміру базу, а відповідно сповільнить її роботу. Зберігати книги в файловій системі серверу теж не є кращим рішенням, оскільки це не є безпечним. Найкращим варіантом буде використання спеціалізованих файлових сховищ. Популярним рішенням серед подібних технологій є MinIO, оскільки він надає високу продуктивність при роботі з великими обсягами даних, не потребує великої кількості ресурсів та має вбудований захист даних шифруванням. Додатковим плюсом використання файлового сховища буде можливість його використання з обох мікро-сервісів. Таким чином при надсиланні запитів для генерації тестів, Java мікро-сервісу потрібно буде вказати лише ідентифікатор книги в сховищі, а не надсилати весь файл, а Python мікро-сервіс зможе легко і швидко його завантажити для подальшої роботи. Таким чином виконання запитів генерації тестів значно пришвидшиться.

3.2.3 База даних

Для збереження інформації застосунок потребує бази даних.

Гарним вибором для роботи з Java та Spring буде база даних PostgreSQL. Вона виділяється своєю надійністю, безпекою та гнучкістю, а також забезпечує високу продуктивність та масштабованість.

Окрім того, технології Java Persistence API (JPA) та Hibernate, які використовуються у Spring, добре оптимізовані для роботи з цією базою даних.

На рисунку 3.4 представлена архітектура бази даних, з якою буде вестись уся подальша робота застосунку.

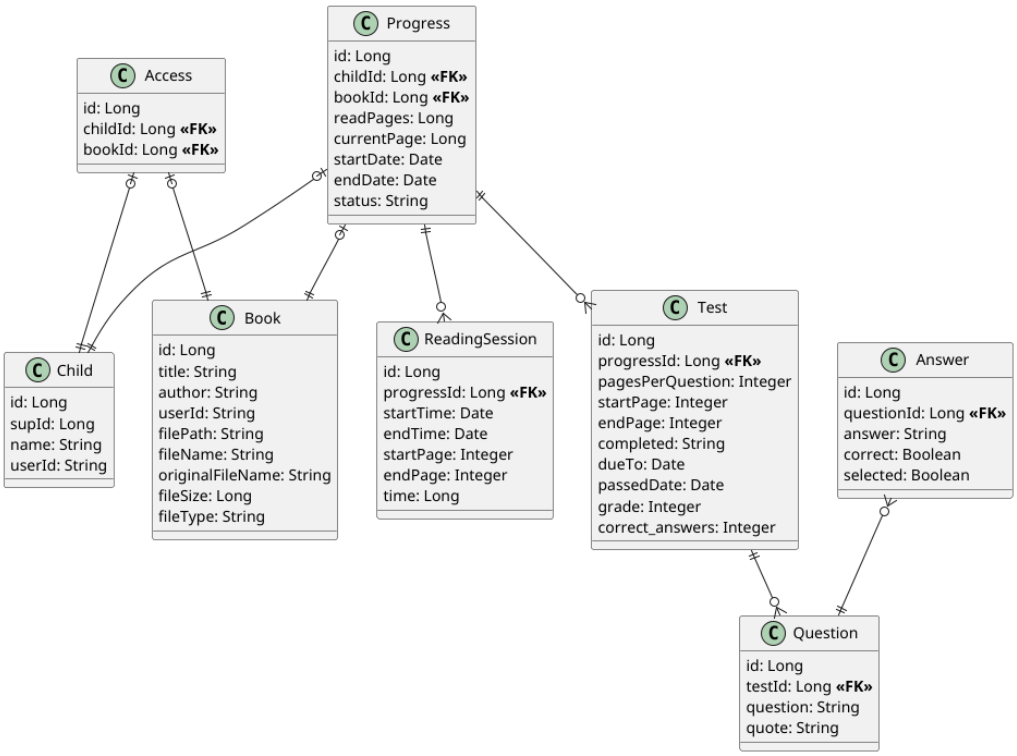

Рисунок 3.4 – Головний екран користувача в ролі «Батько»

База даних включає в себе такі структури:

– книга – зберігає деталі книги та інформацію про файл;

– дитина – відповідає за збереження дітей;

– доступ – доступ дитини до якоїсь книги зберігається в цій таблиці;

– прогрес – зберігає дані про читання книги, а саме статус прочитання, дату початку та закінчення читання, останню сторінку, тощо;

– сесія читання – зберігає в інформацію прочитані сторінки та час читання. Потрібна для створення статистики;

– тест – створені тести зберігаються в цю таблицю та в майбутньому оновлюються, записуючи дані про результати проходження;

– питання – таблиця для згенерований моделлю питань;

– відповідь – таблиця для згенерованих моделлю відповідей.

3.3 Ілюстрація роботи застосунку

3.3.1 Роль «Батько»

Заходячи в застосунок, користувача у ролі «Батько» зустрічає головний екран застосунку, на якому присутні кнопки для переходу на сторінки книг, тестів, дітей та статистики (рис. 3.5).

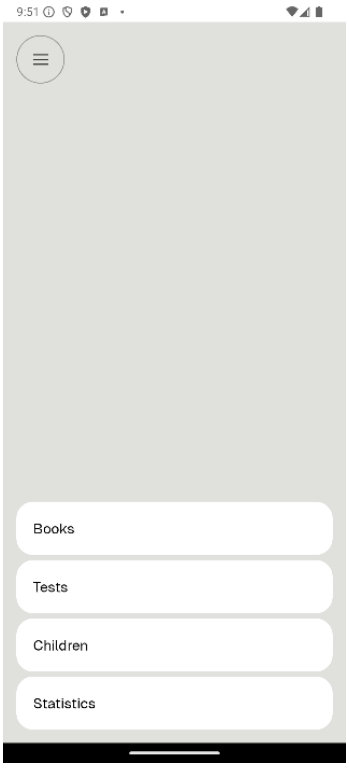

Рисунок 3.5 – Головний екран користувача в ролі «Батько»

Перейшовши на сторінку книг, користувач може побачити список вже завантажених творів, а також додати новий (рис. 3.6).

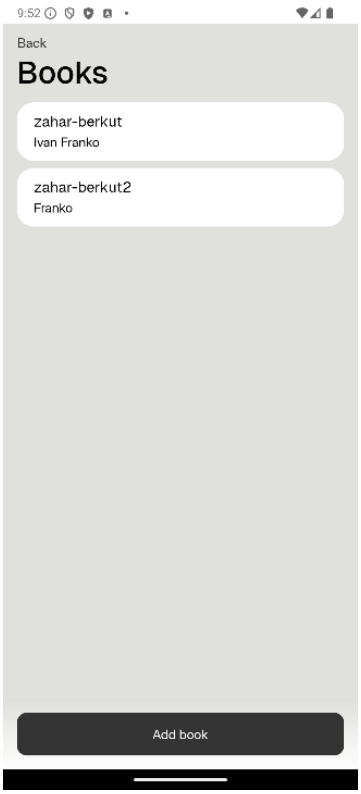

Рисунок 3.6 – Сторінка книг користувача в ролі «Батько»

Натиснувши на кнопку додавання книги, користувачу відкривається модальне вікно, в якому він може обрати файл книги з пристрою, та додати відомості про книгу, а саме, назву та автора (рис. 3.7).

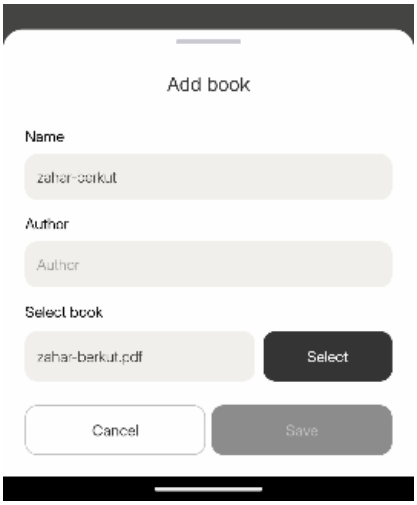

Рисунок 3.7 – Модальне вікно додавання книги

Натиснувши на якусь з доданих книг, користувач потрапить на сторінку цієї книги, де зможе побачити список дітей, яким надано до неї доступ. Також з цієї сторінки можна назначити цей самий доступ або прочитати книгу (рис. 3.8).

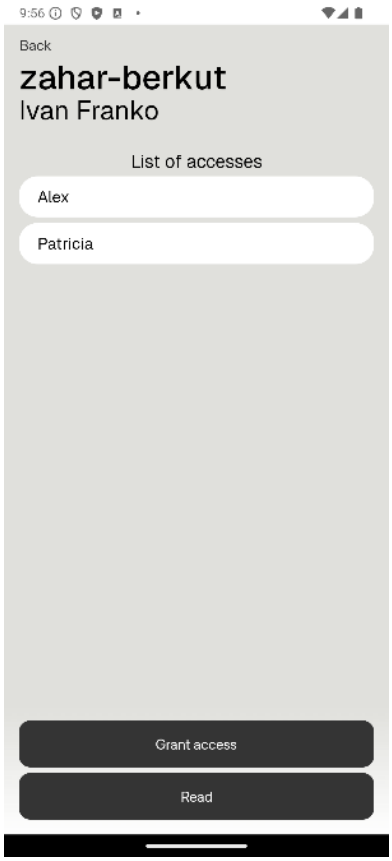

Рисунок 3.8 – Сторінка деталей книги

При наданні доступу користувачу надається модальне вікно зі списком дітей, в якому він може обрати потрібну дитину (рис. 3.9).

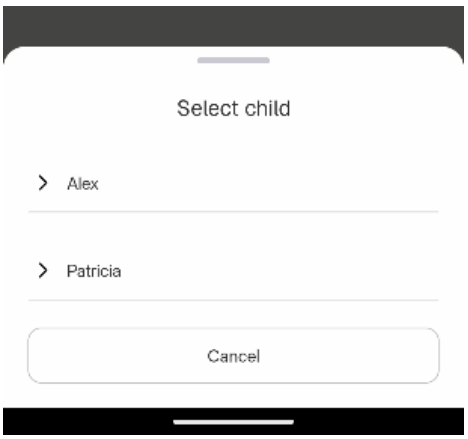

Рисунок 3.9 – Модальне вікно надання доступу

Читання книги відбувається на окремій сторінці, де відображаються сторінки книги, які можна перегортати, та прогрес читання (рис. 3.10).

Рисунок 3.10 – Сторінка читання книги

Натиснувши на конкретну дитину на сторінці книги, користувач потрапить на сторінку, де зможе побачити список тестів, які призначені цій дитині для даної книги. Картки тестів відображають деталі тесту, а саме, сторінки та кількість запитань, статус тесту та оцінку, у тому випадку, якщо тест пройдено (рис. 3.11). Також на цій сторінці присутня можливість призначити новий тест для цієї книги.

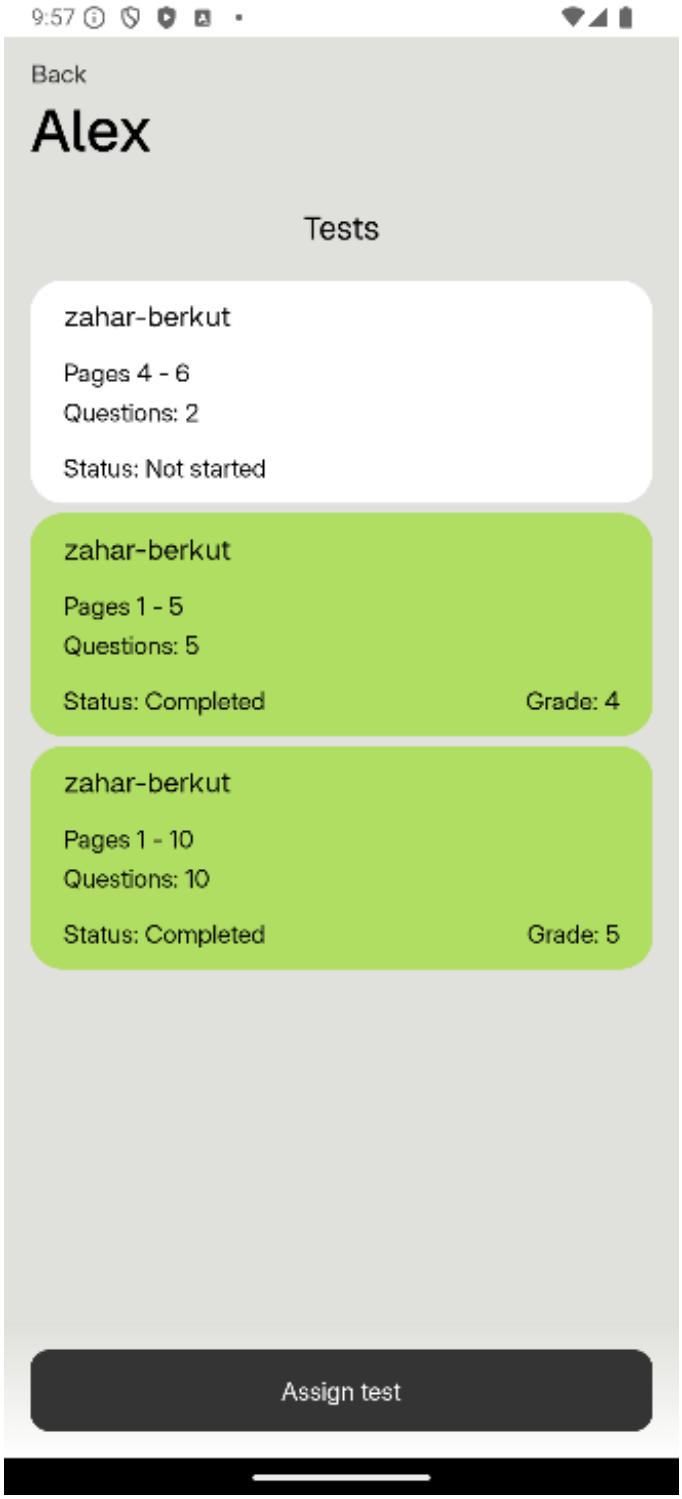

Рисунок 3.11 – Список тестів, призначених дитині для обраної книги

Для призначення нового тесту потрібно вказати початкову та кінцеву сторінки та кількість запитань (рис. 3.12). Книгу в цьому модальному вікну не можна змінити, оскільки користувач потрапив сюди зі сторінки конкретної книги, відповідно тест призначається для неї.

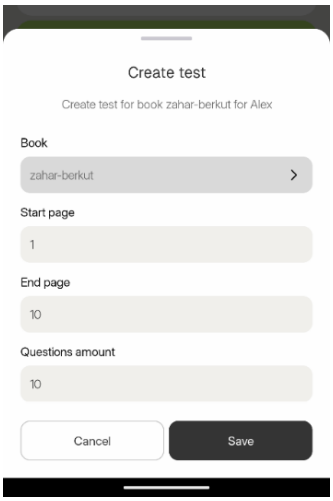

Рисунок 3.12 – Модальне вікно призначення тесту

Окрема сторінка тестів схожа на ту, яка відображається на рисунку 3.12, але на ній присутня можливість обрати дитину, а тести будуть завантажуватись по всім книгам, а не по конкретній (рис. 3.13).

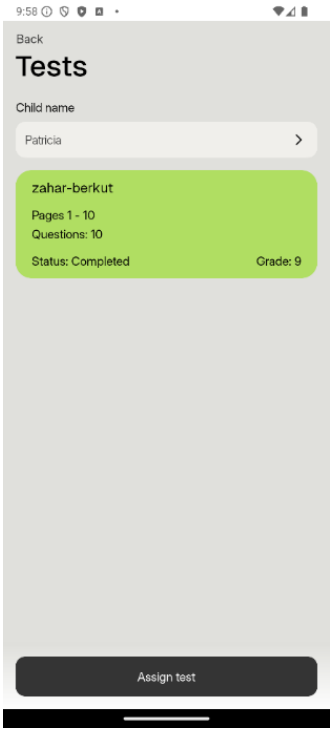

Рисунок 3.13 – Сторінка тестів

З цієї сторінки так само можна призначити тест для обраної дитини, як на рисунку 3.12, проте в цьому випадку потрібно обрати книгу за допомогою додаткового модального вікна (рис. 3.14).

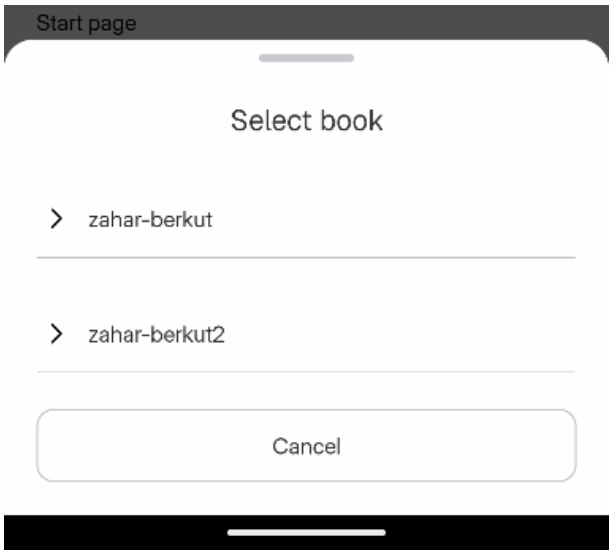

Рисунок 3.14 – Модальне вікно вибору книги для створення тесту

Сторінка дітей (рис. 3.15) відображає список створених користувачем дітей, звідки можна також і додати нову дитину, вказавши її ім’я.

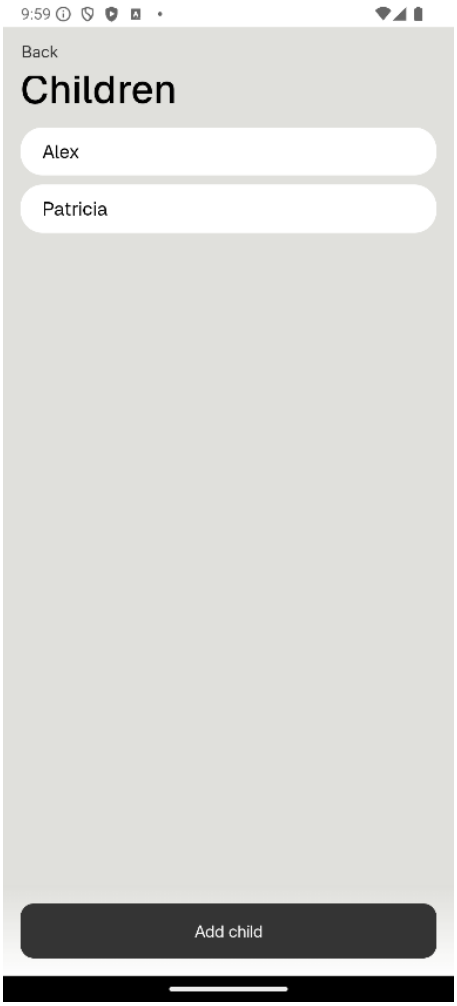

Рисунок 3.15 – Сторінка списку дітей

Натиснувши на дитину зі списку, користувач потрапить на сторінку, де буде відображено список тестів дитини по всім книгам (рис. 3.16). З цієї сторінки можна також і призначити новий тест таким саме чином, як на рисунку 3.14.

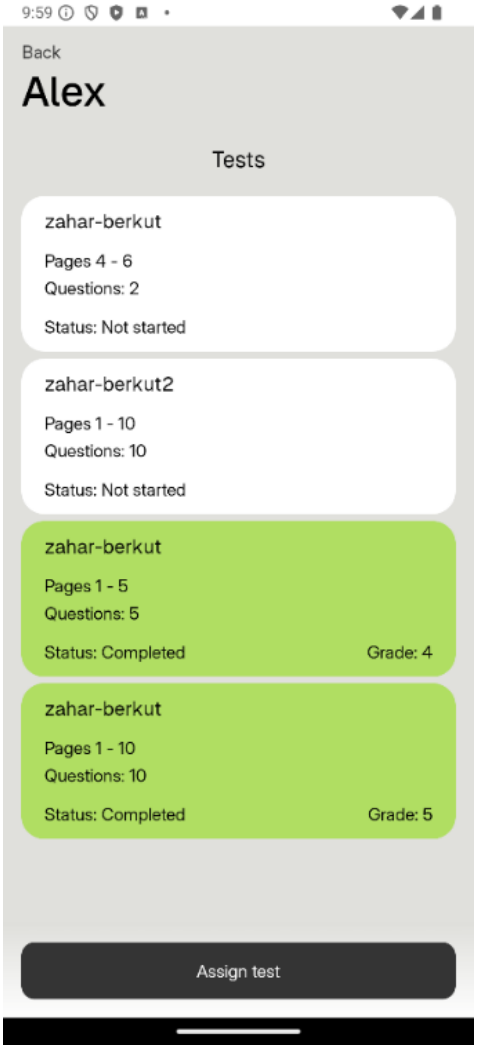

Рисунок 3.16 – Список тестів дитини

3.3.2 Роль «Дитина»

Для перемикання на роль дитини, користувач повинен зайди в бокове меню на головній сторінці, де буде відображено кнопку перемикання режиму (рис. 3.17), після натискання на яку, відобразиться модальне вікно зі списком існуючих дітей, в якому треба обрати потрібну дитину (рис. 3.18).

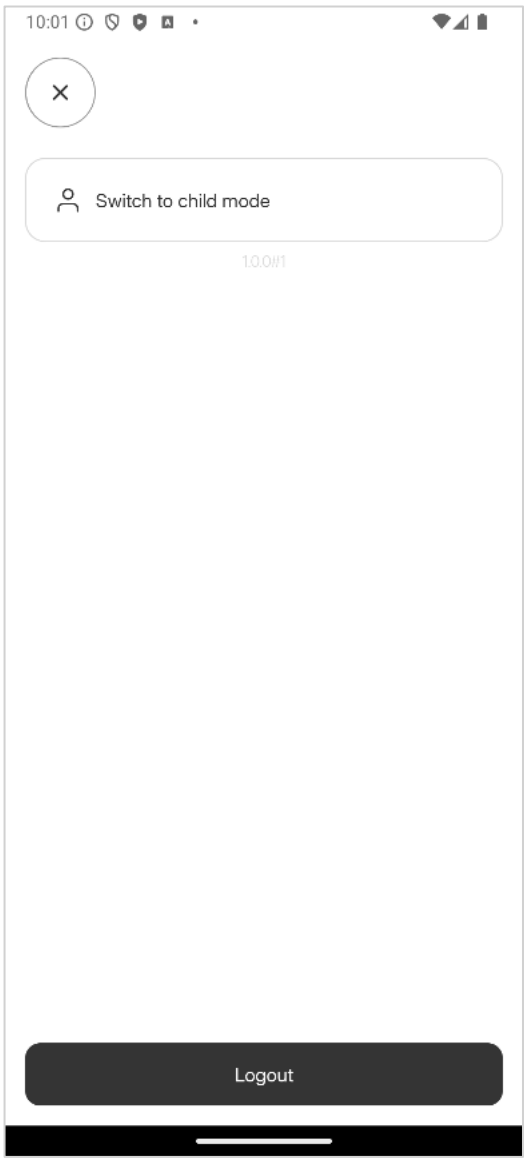

Рисунок 3.17 – Бокове меню головного екрану

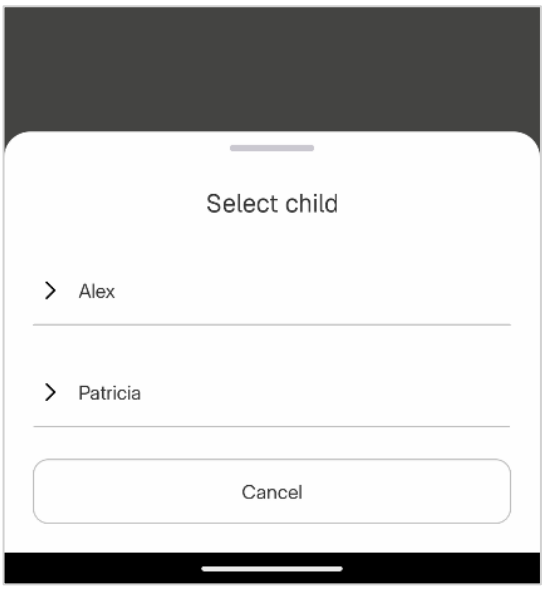

Рисунок 3.18 – Модальне вікно вибору дитини

Користувача, який переключився на роль дитини, зустрічає головний екран, з якого він має доступ до сторінок книг, тестів та статистики (рис. 3.19).

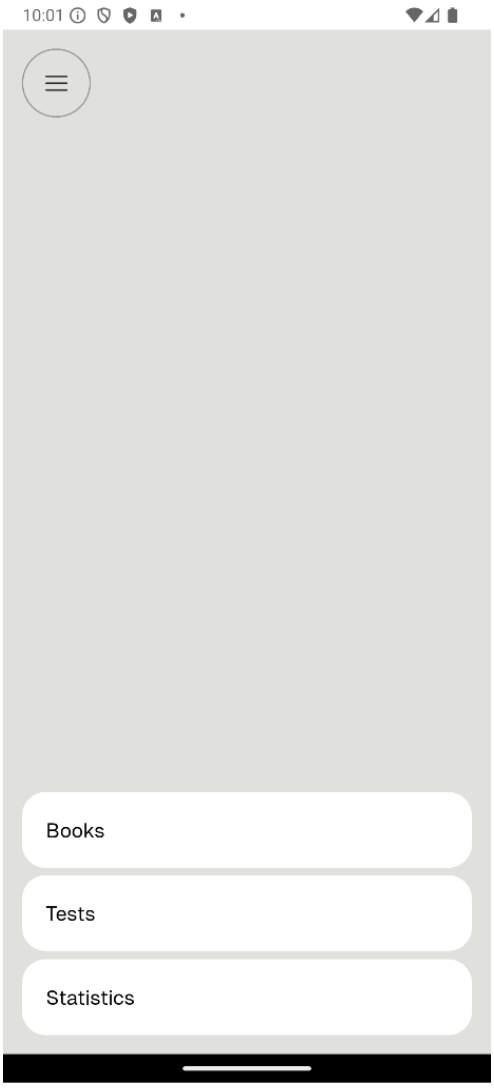

Рисунок 3.19 – Головний екран користувача у ролі «Дитина»

Перейшовши на сторінку книг, користувач побачить таке саме відображення як на рисунку 3.6, де буде виведено список книг, до яких ця дитина має доступ. Також відрізняється те, що користувач у ролі дитини не має кнопки для завантаження книг, цей функціонал доступний лише батькам.

Обравши якусь книгу, користувач потрапить на сторінку її деталей (рис. 3.20), проте, на відміну від цієї ж сторінки для батьків, він не матиме доступ до списку дітей, яким надано до неї доступ, а також не зможе надати доступ.

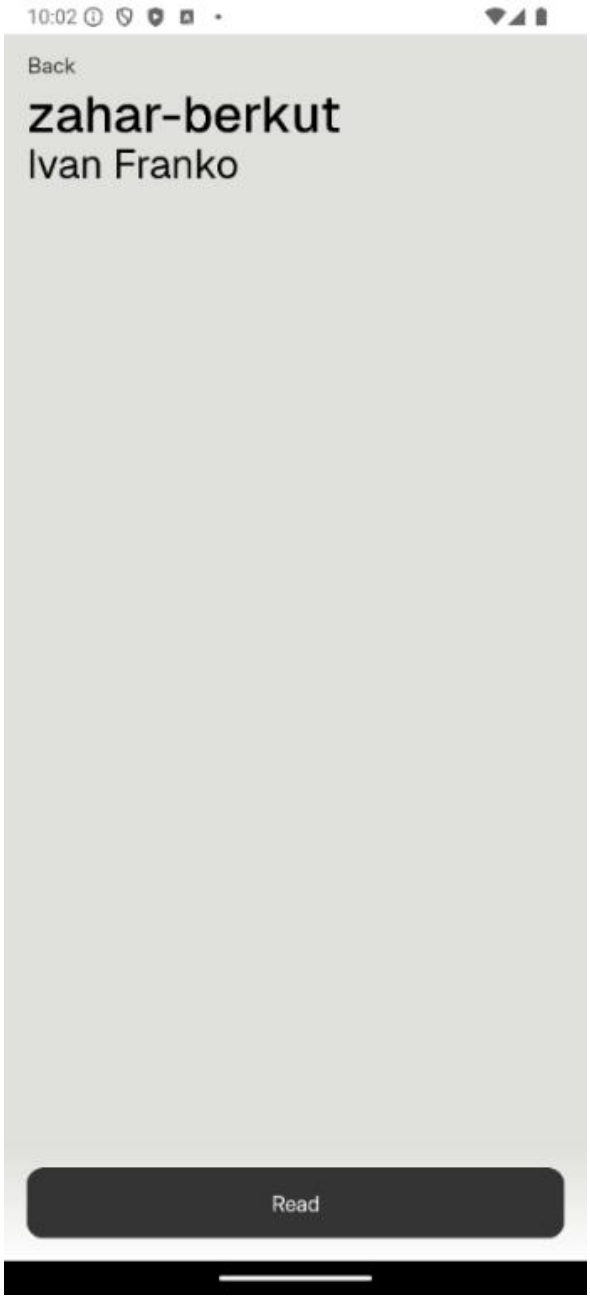

Рисунок 3.20 – Сторінка деталей книги для користувача у ролі «Дитина»

Коли дитина починає читати книгу, вона потрапляє на ту ж саму сторінку, що і батько. Основною відмінністю є те, що у випадку читання з роллю дитини застосунок запитує з серверу доступні тести. Якщо такі є, то повертається тест, який має найменше значення кінцевої сторінки. Отримавши наявний тест, застосунок повідомляє користувача, що після заданої сторінки треба буде пройти тестування, а дійшовши до цієї сторінки, замість кнопки перегортання з’явиться кнопка для початку тесту (рис. 3.21).

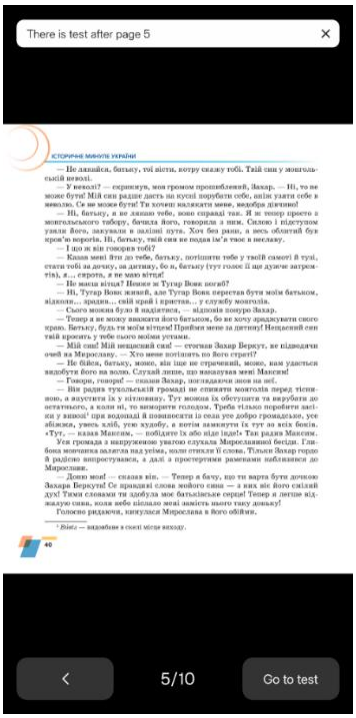

Рисунок 3.21 – Сторінка читання для дитини

Після переходу до тестування, користувачу послідовно надаються запитання та 4 варіанти відповідей (рис. 3.22). Відповівши на останнє запитання, дитина отримує результати у вигляді кількості правильних відповідей та оцінки за десятибальною шкалою (рис. 3.23).

Рисунок 3.22 – Процес проходження тесту

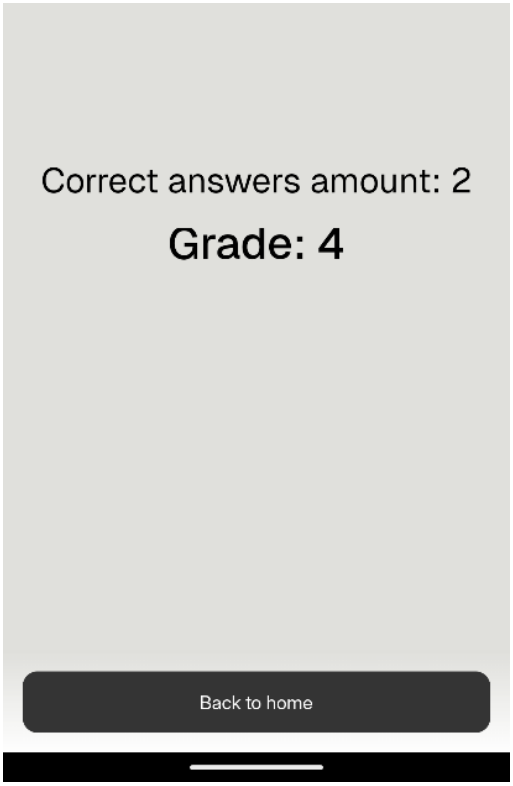

Рисунок 3.23 – Результати тестування

Друга та остання сторінка для дитини представляє собою список тестів, де відображені тести для усіх книг (рис. 3.24). Ця сторінка схожа на сторінку тестів для батьків, проте не дає можливості обрати дитину та призначити тест.

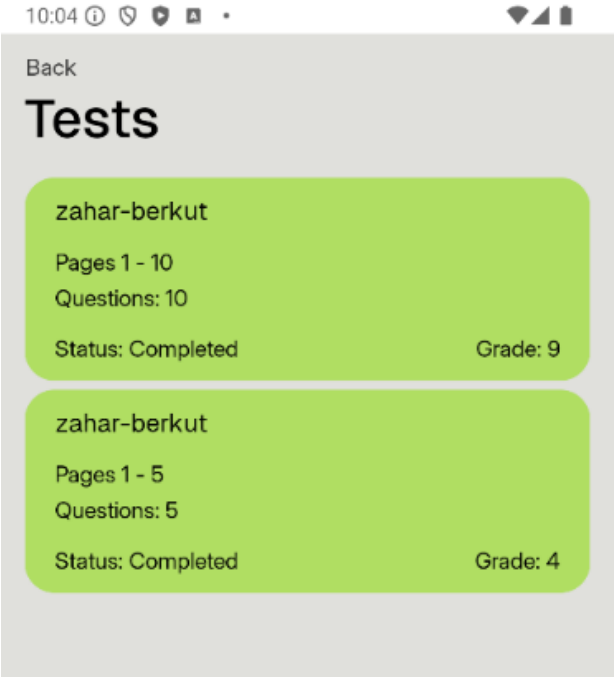

Рисунок 3.24 – Сторінка тестів для дитини

3.3.3 Статистика

Доступ до сторінки статистики мають як діти, так і батьки. Ця сторінка представляє собою статистику, яка складається з двох вкладок. Перша вкладка відображає графік зі стовпчиками, де кожен показує середню оцінку дитини, розраховану за пройденими тестам (рис. 3.25) Під графіком відображаються картки з додатковою інформацією у вигляді пройденої та загальної кількості тестів, відсотку пройдених тестів та числове відображення середньої оцінки дитини. Список карток можна гортати по горизонталі або швидко перейти до потрібної, натиснувши на якийсь із стовпчиків.

Друга ж вкладка містить в собі статистику читання по днях, де можна обрати дитину. За обраною дитиною відобразяться 2 графіки, а саме, графік, відображаючий час читання в хвилинах, та графік, відображаючий кількість прочитаних сторінок (рис. 3.26).

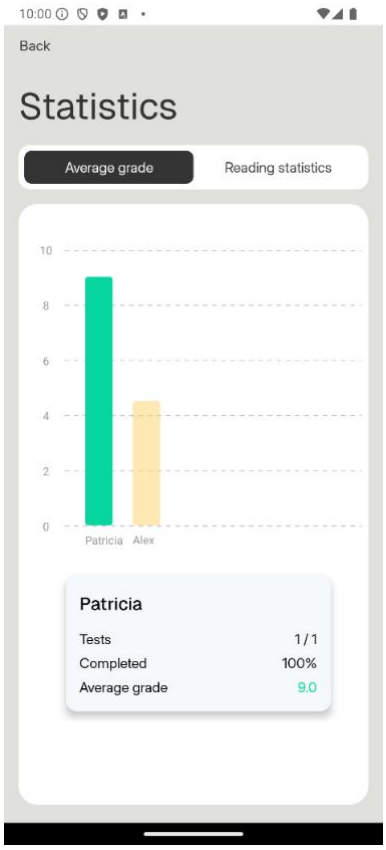

Рисунок 3.25 – Графік середніх оцінок дітей та карточки з додатковою інформацією

Рисунок 3.26 – Графік статистики читання з часом ті кількістю прочитаних сторінок по днях

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено мобільний застосунок для моніторингу розуміння прочитаних художніх творів, який містить в собі інтерактивні елементи, а саме, тести, які допомагають краще запам’ятати прочитане та оцінити якісь читання.

Для створення застосунку було вирішено такі завдання:

– проведено аналіз ринку споживання книг, рівня читацької активності молоді та роль цифрових технологій у формуванні мотивації до читання;

– оглянуто існуючі застосунки для читання та їхні можливості, визначено переваги та недоліки;

– визначено можливості штучного інтелекту та інших інновацій для покращення читацьких навичок в освітніх застосунках;

– оглянуто сучасні технології, які допомагають створювати мобільні застосунки з використанням штучного інтелекту та мовних моделей;

– визначено мету використання LLM та спосіб її застосування для покращення читацьких навичок;

– досліджено питання використання LLM для генерації тестів та обрано модель, яка за параметрами якості, швидкості та вартості як найкраще задовольняє вимогам мобільного застосунку для аналізу розуміння прочитаних художніх творів:

1) розроблено інструкцію для моделі, яка вказує як правильно генерувати тести;

2) сформовано набір даних для проведення експериментів щодо дослідження LLMs на предмет створення тестів;

3) оцінено час генерації тестів;

4) проведено тестування за участі експертів з метою визначення кращої моделі для даної задачі;

5) розглянуто питання покриття великого обсягу тексту мовними моделями з використанням ін’єкцій;

– визначено набір технологій та розроблено архітектуру майбутнього мобільного застосунку;

– розроблено мобільний застосунок для моніторингу розуміння прочитаних художніх творів.

В рамках роботи було проведено дослідження за участі експертів з метою визначення найбільш відповідної моделі для задачі генерації тестів на поточний час. В результаті проведеного тестування моделі порівнювались за показниками швидкості, якості та ціни. Найкращі результати якості показали моделі від Anthropic, проте їхнім суттєвим недоліком виявилась велика вартість. В той же час модель Gemini 2.0 Flash відстає за якістю від найкращої моделі Anthropic всього на 5%, але показує набагато більшу швидкість і в рази меншу вартість. Таким чином в застосунку було використано саме Gemini 2.0 Flash для генерації тестів. Також було проведено дослідження з метою визначення можливостей покриття великого обсягу тексту мовним моделями, результати якого показали, що моделі добре працюють з прямими ін’єкціями, проте мають проблеми при роботі з асоціативними ін’єкціями в середині та наприкінці тексту. Результати дослідження дали зрозуміти що для покращення якості тестів потрібно розділяти текст твору на невеликі частини, на основі яких будуть генеруватись запитання.

В майбутньому застосунок може зазнати значних покращень, наприклад, налаштування пріоритету та графіку читання, додавання термінів проходження тестів, розширення можливостей тестування, наприклад, відповідь у вільному форматі з подальшим аналізом мовною моделлю, збір додаткової аналітики читання та розширення статистики, генерація зображень до книг з метою покращення запам’ятовування.

Результати роботи апробовано у вигляді 2 тез доповідей під час четвертої національної наукової і практичної конференції «Scientific practice: modern and classical research methods» [32], 29-го Міжнародного молодіжного форуму «Радіоелектроніка і молодь у XXI столітті», де робота зайняла 1 місце [34].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Опитування: Близько половини української молоді читає книги щодня або декілька разів на тиждень. РеІнформ. URL: https://reinform.com.ua/52174/opytuvannya-blyzko-polovyny-ukrayinskoyi-molodi-chytaye-knygy-shhodnya-abo-dekilka-raziv-na-tyzhden/ (дата звернення 18.04.2024).

2. Book sales by year: Print books in the U.S. 2024| statista. Statista. URL: https://www.statista.com/statistics/422595/print-book-sales-usa/ (дата звернення 18.04.2024).

3. E-book unit sales in the U.S. 2020| statista. Statista. URL: https://www.statista.com/statistics/426799/e-book-unit-sales-usa/ (дата звернення 18.04.2024).

4. ДС редакція. Швидкий контент, повільний розум? Як короткі відео викликають залежність та впливають на мозок — dsnews.ua. «Ділова столиця» українською – найсвіжіші новини України та світу. URL: https://www.dsnews.ua/ukr/society/shvidkiy-kontent-povilniy-rozum-yak-korotki-video-viklikayut-zalezhnist-ta-vplivayut-na-mozok-22022025-517271 (дата звернення 18.04.2024).

5. Gorokhovatskyi, V., Tvoroshenko, I., Yakovleva, O., Hudáková, M., & Gorokhovatskyi, O. (2024). Application a Committee of Kohonen Neural Networks to Training of Image Classifier Based on Description of Descriptors Set. IEEE Access, 1. https://doi.org/10.1109/access.2024.3404371.

6. Gorokhovatskyi, O., & Yakovleva, O. (2024). MEDOIDS AS A PACKING OF ORB IMAGE DESCRIPTORS. Advanced Information Systems, (Vol. 8, pp 5-11).

7. Yakovleva, O., Kovtunenko, A., Liubchenko, V., Honcharenko, V., & Kobylin, O. (2023). Face Detection for Video Surveillance-based Security System. In COLINS (3) (pp. 69-86).

8. Yakovleva O., Nebeský L., Kirichenko A. (2023) Using the GPT models for responses based on custom content to develop neural consultant for university applicants. Abstracts of V International Scientific and Practical Conference.

Madrid, Spain. Pp. 172-178. URL: https://eu-conf.com/ua/events/trends-in-science-regarding-the-creation-of-new-teaching-methods/.

9. Yakovleva, O., Kovač, M., Ardasov, V. & Yeremenko, I. (2023). Study on adding functionality to the Zoom online conference system for monitoring the participant activities. Public Administration and Regional Development, 19(1), pp. 158–184.

10. Naumenko V., Shelest V., Yakovleva O. (2024). Combination of .Net technology and Angular framework to develop application for testing SQL language knowledge. Proceedings of the XVth International Scientific and Practical Conference «Free And Open Source Software», Ukraine, Kharkiv, February 13-14, 2024. pp.63-66.

11. Yakovleva Olena, Matúšová Silvia, Táncošová Judita (2024, December 16-18). Investigation of LLMs for generating answers based on user-provided content to support educational and organizational processes. Abstracts of XVI International Scientific and Practical Conference «Modern and new technical trends that help humanity». Thessaloniki, Greece, Pp. 289-295.

12. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., and Hudáková M. (2025) Image description compression in classification structural methods, IEEE Access, vol. 13, pp. 43631-43641.

13. Gorokhovatskyi V., Tvoroshenko I., and Yakovleva O. (2024) Transforming image descriptions as a set of descriptors to construct classification features, Indonesian Journal of Electrical Engineering and Computer Science, vol. 33, no. 1, pp. 113-125.

14. Gorokhovatskyi , O., & Yakovleva , O. (2024). Medoids as a packing of ORB image descriptors. Advanced Information Systems, 8(2), 5–11.

15. Yakovleva, O., & Nikolaieva, K. (2020). Research Of Descriptor Based Image Normalization And Comparative Analysis Of SURF, SIFT, BRISK, ORB, KAZE, AKAZE Descriptors. Advanced Information Systems, 4(4), 89-101.

16. Yakovleva O., Nebeský L, Liakhov P. (2023) Research methods of texture image analysis to solve the texture search problem. Proceedings of the IV International Scientific and Practical Conference. Vienna, Austria. pp. 252-261.

17. Яковлева, О. В., & Кускова, І. В. (2006) Дослідження результатів сегментації зображень методом матриць збігів. *Вісник Національного технічного університету "ХПІ"*, 39, С.164 -171.

18. Яковлева, О. В., & Панченко, І. А. (2007) Застосування енергетичних характеристик Лавса для сегментації зображень. *Біоніка інтелекту: науково-технічний журнал*, 2(67), С.94-98.

19. Яковлева О.В., Нестерова О.П. (2009) Порівняльний аналіз методів характеристик Лавса і матриць збігів у задачах сегментації текстурних зображень. Прикладна радіо-електроніка: науч.-техн. журнал, Том 8, №2. - С.181 - 187.

20. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., Hudáková M., and Gorokhovatskyi O. (2024) Application a committee of Kohonen neural networks to training of image classifier based on description of descriptors set, IEEE Access, vol. 12, pp. 73376-73385.

21. Ковтуненко, А. Р., Яковлева, О. В., Любченко, В. А., & Янголенко, О. В. (2020). Дослідження сумісного використання математичної морфології та згорткових нейронних мереж для вирішення задачі розпізнавання цінників. *Вісник Національного технічного університету ХПІ*, 3, С. 24-31.

22. Yakovleva O., Matúšová S., Tvoroshenko I., Isaiev Y. (2024). Visitor counting based on video stream analysis from surveillance cameras. Scientific Journal of Bratislava University of Economics and Management «Public Administration and Regional Development, Economics, Management and Marketing», vol. 20, no. 1, pp. 67–87.

23. Epic - Books for Kids. URL: https://www.getepic.com/ (дата звернення 29.09.2024).

24. Pearson school. URL: https://www.pearsoncanadaschool.com/ (дата звернення 29.09.2024).

25. Readability. URL: https://www.readabilitytutor.com/ (дата звернення 29.09.2024).

26. Khan Academy Kids. URL: https://learn.khanacademy.org/khan-academy-kids/ (дата звернення 29.09.2024).

27. Gloose. URL: https://glose.com/what-is-glose (дата звернення 29.09.2024).

28. Rork. URL: https://rork.ua/ (дата звернення 29.09.2024).

29. Любченко В., Талах В. (2024) Створення асистентів на базі штучного інтелекту. Радіоелектроніка та молодь у XXI столітті: Конференція "Інформаційні інтелектуальні системи", 2024, Харків, Україна, С. 50-51.

30. Artificial Analysis. URL: https://artificialanalysis.ai/guide (дата звернення 28.02.2025).

31. Chatbot Arena. URL: https://lmarena.ai/ (дата звернення 28.02.2025).

32. Yakovleva, O., Matúšová, S., & Talakh, V. (2025). GRADIO AND HUGGING CAPABILITIES FOR DEVELOPING RESEARCH AI APPLICATIONS. Collection of scientific papers «ΛΌГОΣ», (February 14, 2025; Boston, USA), 202-205.

33. Modarressi, A., Deilamsalehy, H., Dernoncourt, F., Bui, T., Rossi, R. A., Yoon, S., & Schütze, H. (2025). NoLiMa: Long-Context Evaluation Beyond Literal Matching. arXiv preprint arXiv:2502.05167.

34. Талах В. О. (2025) Дослідження питання використання LLMs для генерації тестів з метою моніторингу розуміння прочитаного матеріалу. *Радіоелектроніка та молодь у XXI столітті: тези доповідей 29-го Міжнародного молодіжного форуму (Харків, 16–19 квітня 2025 р.)*. Харків: ХНУРЕ, 2025. Т. 7. С. 136-138.