Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мережних технологій факультету інформатики

ІНТЕЛЕКТУАЛЬНА СИСТЕМА ПІДГОТОВКИ ДІТЕЙ ДО ШКОЛИ НА ОСНОВІ АДАПТИВНОГО НАВЧАННЯ

Текстова частина до кваліфікаційної роботи
за спеціальністю „Інженерія програмного забезпечення” 121

Керівник курсової роботи
кандидат фізико-математичних наук,
доцент Нагірна А. М.
_____
(підпис)
“____” _____ 2026 р.

Виконала студентка ІПЗ-4
Скіп Ю. Я.
“____” _____ 2026 р.

Київ 2026

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Зав. кафедри інформатики
доцент Нагірна А. М.
(підпис)
“____” _____ 2026 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на кваліфікаціну роботу

студенту Скіп Юлії Ярославівні факультету інформатики 4 курсу
ТЕМА: Інтелектуальна система підготовки дітей до школи на основі адаптивного навчання

Зміст ТЧ до кваліфікаційної роботи:

Індивідуальне завдання
Вступ
1 Дослідження предметної області та обгрунтування актуальності розробки
2 Проєктні засади розробки iOS-застосунку «Інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання»
3 Розробка iOS-застосунку «Інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання»
Висновки
Список літератури

Дата видачі “____” _____ 2025 р. Керівник Нагірна А.М.

_____
(підпис)

Завдання отримала _____

(підпис)

Тема: Інтелектуальна система підготовки дітей до школи на основі адаптивного навчання

Календарний план виконання роботи:

№ п/п	Назва етапу кваліфікаційної роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на кваліфікаційну роботу	11.10.2025	виконано
2.	Дослідження галузі дошкільної освіти	11.10.2025 – 30.10.2025	виконано
3.	Аналіз існуючих програмних продуктів для дошкільнят	11.10.2025 – 30.10.2025	виконано
4.	Окреслення основних вимог до iOS-застосунку для «Інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання»	30.10.2025 – 10.11.2025	виконано
5.	Створення інтерактивного прототипу iOS-застосунку	10.11.2025 – 20.11.2025	виконано
6.	Розробка бази тестів	20.11.2025 – 10.12.2025	виконано
7.	Розробка навчальних матеріалів	20.11.2025 – 10.12.2025	виконано
8.	Створення ілюстративних матеріалів для застосунку	10.12.2025 – 13.12.2025	виконано
9.	Тренування моделей машинного навчання	13.12.2025 – 20.12.2025	виконано
10.	Подання індивідуального плану та графіку виконання роботи	14.12.2025	виконано
11.	Проєктування внутрішньої бази даних застосунку	20.12.2025 – 25.12.2025	виконано
12.	Написання програмного коду	01.01.2026 – 15.02.2026	виконано
13.	Подання проміжного звіту про виконання роботи	31.01.2026	виконано
14.	Тестування iOS-застосунку	15.02.2026 – 30.02.2026	виконано
15.	Написання попередньої версії текстової частини роботи	01.03.2026 – 30.03.2026	виконано
16.	Попередній захист кваліфікаційної роботи	30.03.2026- 01.04.2026	виконано
17.	Внесення змін до роботи, відповідно до зауважень	01.04.2026 – 14.05.2026	виконано

18.	Захист кваліфікаційної роботи	25.05.2026 – 27.05.2026	виконано

Студент Скіп Ю.Я. _____

Керівник Нагірна А.М. _____

“_____”_____

ЗМІСТ

Анотація

У роботі продемонстровано процес розробки інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання. Досліджено динаміку використання цифрових пристроїв цільовою аудиторією, переважні типи ресурсів та їх вплив на когнітивний розвиток дітей. Досліджено наявні рішення у предметній області та окреслено їхні переваги й недоліки. Описано повний цикл проєктування системи: формування функціональних вимог, проєктування структури програми, створення контенту, розробку користувацького інтерфейсу. Окреслено процес тренування моделей машинного навчання для класифікації зображень. Проаналізовано поширені адаптивні алгоритми та модифіковано обране рішення до потреб системи. Розглянуто використаний технологічний стек, охарактеризовано реалізацію та роботу модулів програми. Мета роботи – розробка комплексної інтелектуальної системи для дітей дошкільного та раннього шкільного віку на основі адаптивного навчання. Результатом є комплексний iOS-застосунок, який дає основу для формування ключових освітніх компетентностей та забезпечує якісну підготовку дітей до школи.

Ключові слова: інтелектуальна система, адаптивний алгоритм, екосистема Apple, проєктування, інтерфейс користувача, розробка, дошкільна освіта.

Вступ

У зв’язку із цифровізацією значної частини життя суспільства екранний час дітей стрімко зростає. Крім того, більшість з них проводять велику його частину переглядаючи розважальні ресурси чи граючи у відеоігри. На жаль, контент, що доступний в мережі, часто несе загрозу дітям, пропагуючи жорстокість, насильство чи аморальну поведінку. Така тенденція може призвести не лише до погіршення розвитку когнітивних навичок молодого покоління, а й негативно вплинути на психологічний стан дітей та підвищити рівень агресії. Проте, освітні ресурси займають друге місце у рейтингу популярності, що вказує на наявність інтересу до такого роду рішень. Також, враховуючи те, що гаджети слугують засобами зв’язку, що є актуальним в теперішній час – повністю обмежити їхнє використання – недоцільне рішення. Тому варто забезпечити спрямування більшості екранного часу на освітній напрямок. На жаль, українськомовний ринок пропонує малу кількість продуктів у даній ніші. Більше того, наявні рішення часто працюють несправно та не відповідають вимогам користувачів.

Виходячи з потреби користувачів у якісному освітньому ресурсі для дітей, який зміг би зменшити негативний вплив цифровізації на їхній розвиток, за мету даної роботи поставлено розробку комплексної інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання. Створення такої системи не лише забезпечить збільшення кількості екранного часу, спрямованого на освіту, але й ознайомить дітей з інноваційними інтелектуальними технологіями та підвищить ефективність навчального процесу шляхом персоналізації освітньої траєкторії завдяки впровадженню адаптивного підходу.

Робота складається з трьох розділів, кожен з яких містить у собі декілька підрозділів.

Перший розділ присвячено дослідженню предметної області. Проаналізовано динаміку використання цифрових пристроїв дітьми дошкільного та раннього шкільного віку, визначено типи ресурсів, які переважають у їхньому дозвіллі та досліджено їх вплив на розвиток дітей. Проведено порівняльний аналіз існуючих рішень в даній предметній області, визначено їхні переваги, недоліки та відгуки користувачів. Доведено актуальність розробки системи.

У другому розділі окреслено проєктні засади розробки, обґрунтовано мету та визначено постановку завдань. Сформовано функціональні вимоги до продукту та спроєктовану загальну структуру програми. Описано процес розробки баз навчальних та тестових матеріалів та ієрархічних структур для їх збереження. Розроблено базу ілюстративних матеріалів, спроєктовано та реалізовано прототип користувацького інтерфейсу. Окреслено процес створення наборів тренувальних даних та процесу навчання класифікаційних нейронних мереж.

Третій розділ присвячено процесу розробки інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання. Проаналізовано існуючі адаптивні алгоритми та обрано оптимальний варіант в контексті розроблюваної системи. Проведено додаткову модифікацію обраного алгоритму до специфіки цільової аудиторії, структури проєкту та тестового контенту. Обґрунтовано вибір технологічного стеку, описано використані програмні фреймворки, зокрема авторське рішення для вбудовування полотна для малювання. Створено внутрішню базу даних системи. Описано концепції та програмні реалізації усіх розроблених модулів програми та внутрішніх алгоритмів. Окреслено процес впровадження системи.

Створено комплексний програмний продукт, який покриває формування базових освітніх компетенцій для дітей дошкільного та раннього шкільного віку. Розроблений продукт містить модулі для навчання, тестування, дослідження, звітності, реєстрації та навігації. Використано адаптивний алгоритм для персоналізації освітньої траєкторії користувача та впроваджено інтелектуальні елементи для покращення користувацького досвіду і формування початкової технологічної компетентності у користувачів.

РОЗДІЛ 1: Дослідження предметної області та обгрунтування актуальності розробки

1.1. Дослідження предметної області та виявлення потреб користувачів

Перед початком окреслення вимог та розробки програмного продукту важливо дослідити предметну область та потреби потенційних користувачів. Такий підхід дозволить краще виявити проблеми, з якими стикається цільова аудиторія та розробити застосунок, який допоможе їх вирішити.

В цьому розділі проаналізовано тривалість використання цифрових пристроїв дітьми дошкільного і раннього шкільного віку та досліджено, на що саме вони витрачають час, проведений у гаджетах, та як це на них впливає. Також проаналізовано існуючі програмні продукти в галузі дошкільної освіти, щоб виявити їхні переваги й недоліки та розробити комплексне рішення для інтелектуальної системи підготовки дітей до школи, яке покриватиме базові потреби користувачів.

1.1.1. Аналіз тривалості використання цифрових пристроїв дітьми дошкільного віку

У теперішній час електронні пристрої стали невід’ємною частиною життя не лише дорослих, а й молодшого покоління. Більше того, протягом останніх декількох років, кількість дітей, що проводять час у гаджетах стрімко зростає.

У цьому підрозділі розглядається динаміка використання цифрових пристроїв дітьми. Особлива увага приділяється віковій категорії від 5 до 8 років, оскільки саме вони та їхні батьки є цільовою аудиторією досліджуваного продукту. Аналіз базується на результатах дослідження, опублікованого компанією Google щодо безпеки дітей в інтернеті в Україні, проведеного у серпні 2024 року компанією Topline [1]. У даному дослідженні взяли участь

570 українських батьків, віком від 18 до 65 років. За результатами опитування (рисунок 1.1), 1% дітей віком від 5-и до 8-и років проводять понад 9 годин онлайн на день, 5% – від 6 до 9 годин, 26% – від 3 до 6 годин, 59% – від 1 до 3 годин, та лише 10% проводять в мережі менше години.

Рисунок 1.1 – Час, що проводять діти онлайн на день (дослідження Google про безпеку дітей в інтернеті в Україні)

Проаналізувавши динаміку результатів дослідження, можна зробити висновок, що більшість дошкільнят та дітей раннього шкільного віку проводять в гаджетах від 1 до 3 годин, що є дуже високим показником для такої вікової категорії. Враховуючи реалії нашого часу, потрібно розуміти, що повне обмеження використання цифрових пристроїв – це складне та суперечливе завдання, адже вони також слугують засобами зв'язку, що є важливим у даний період. Крім того, такі заходи можуть навіть нашкодити психологічному стану дитини, оскільки гаджети стали важливою частиною життя суспільства й заборона їх використання може призвести до цькування з боку однолітків та відчуття дитиною соціальної ізольованості. Виходячи з цього, оптимальним рішенням може стати гарантування користі часу, проведеного онлайн.

1.1.2. Дослідження типів цифрових ресурсів, які найчастіше використовують діти дошкільного віку, та їх впливу на розвиток дитини

У цьому підрозділі аналізується, як діти проводять екранний час, які ресурси переважають у їхньому дозвіллі, та яким чином це впливає на їхній

розвиток. Під час аналізу використано результати дослідження Google про безпеку дітей в інтернеті в Україні, яке згадувалось раніше.

За даними дослідження (рисунок 1.2), 3% застосунків або сайтів, які використовують діти віком від 5-ти до 8-ми років – новинні, 38% – це соціальні мережі, 46% займають відеоігри, 57% – розважальні ресурси та 52% – освітні продукти.

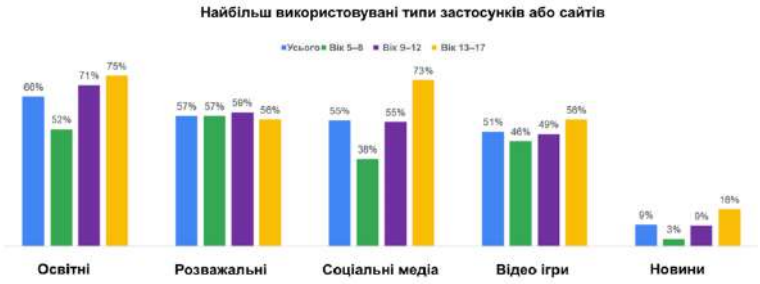


Рисунок 1.2 – Найбільш використовувані типи застосунків або сайтів (дослідження Google про безпеку дітей в інтернеті в Україні)

Така динаміка не викликає суттєвих занепокоєнь, адже освітня галузь займає друге місце за кількістю використання, проте не варто ігнорувати те, що решта типів, що займають такі ж високі позиції – це розважальні ресурси, відеоігри та соціальні медіа, які, на жаль, не завжди позитивно впливають на розвиток дитини та її ментальне здоров'я.

Журнал Американської Медичної Асоціації (JAMA: The Journal of the American Medical Association) [2] 13 жовтня 2025 року опублікував статтю [3], про вплив соціальних мереж на когнітивний розвиток дітей. У цьому дослідженні вказано, що було виявлено можливий зв'язок між використанням соціальних мереж у ранньому підлітковому віці та нижчими показниками результатів когнітивних тестів. Дослідники стверджують, що діти, які частіше використовують соціальні медіа, гірше виконували тести на читання, пам'ять та словниковий запас, порівняно з тими, хто майже або повністю не

користується ними. Результатом стало те, що діти з низьким зростаючим використанням соцмереж показали в середньому на 1-2 бали, а з високим зростаючим – на 4 бали нижчі показники, у порівнянні з тими, хто майже не користувався соцмережами. Дослідники вважають, що хоч даний результат поки є не критичним, проте ситуація може погіршуватись з часом.

На окрему увагу заслуговує вплив жорстокого контенту в соцмережах та відеоіграх на психіку дітей. Безперечно, не всі такі ресурси пропагують негатив та жорстокість, проте ігнорувати ті, що справді несуть небезпеку не варто. У дослідженні відомого соціального психолога професора Крейга Андерсона [4] описано вплив відеоігор на психіку дітей. У ньому доведено, що насильство у відеоіграх підвищує агресивність дітей незалежно від віку, статі та культури, одночасно знижуючи прояви емпатії та просоціальної поведінки. Вплив таких ігор на психіку набагато небезпечніший за фільми чи книги з таким же змістом, оскільки гравець активно ідентифікує себе з персонажем та може проводити за грою значно більше часу.

Виходячи з усього вищесказаного, варто зазначити, що розважальні ресурси, відеоігри та соціальні медіа, хоч і не завжди, проте можуть негативно впливати на розвиток молодого покоління. Тому батькам варто звернути на це увагу та сприяти спрямуванню більшої частини екранного часу дитини на освітню галузь, а розробникам програмного забезпечення та працівникам освіти, в свою чергу – забезпечити наявність якісних освітніх продуктів на ринку.

1.2. Порівняльний аналіз існуючих рішень у галузі дошкільної освіти

Розглянемо та проаналізуємо існуючі рішення у галузі дошкільної освіти. Перш за все, варто зазначити, що більшість застосунків для дошкільнят, доступних в AppStore – спрямовані на англомовну або російськомовну

аудиторію, що підвищує необхідність створення аналогів українською. Також за запитом «застосунки для дошкільнят» більшість результатів – розважальні, або орієнтовані виключно на розвиток логіки, що робить актуальною розробку продукту, який буде спрямований на формування базових знань дитини, а саме: вивчення букв, чисел, геометричних фігур, кольорів та знайомство з навколишнім середовищем.

Для аналізу обрано три застосунки для дітей дошкільного віку, які найчастіше трапляються в результатах пошуку, а саме: «Мишеняткова Абетка», «Нумо» та «ВИВЧАЮ – НЕ ЧЕКАЮ». Всі вони є нашими потенційними конкурентами, оскільки спрямовані на українськомовну аудиторію та охоплюють формування базових знань. Основну увагу звернемо на відгуки користувачів, зручність використання та напрямки навчання в межах програм.

1.2.1. Аналіз застосунку «Мишеняткова Абетка»

Першим проаналізуємо застосунок «Мишеняткова абетка» [5], оскільки його рейтинг у AppStore (рисунок 1.3) становить 4.4/5.

• *Рисунок 1.3 – Рейтинг застосунку «Мишеняткова абетка» в AppStore.*

Коментарі користувачів (рисунок 1.4) здебільшого позитивні, у них зазначають, що продукт подобається дітям та є корисним для них. Проте, серед найрелевантніших відгуків за версією AppStore можна побачити, що у багатьох людей, під час користування, виникали проблеми зі звуком, що є критичною помилкою, яка негативно впливає на користувацький досвід.

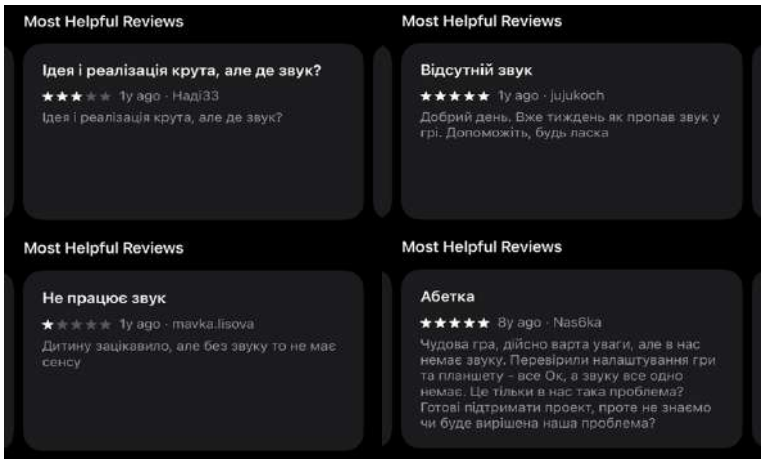


Рисунок 1.4 – Найрелевантніші відгуки на застосунок «Мишеняткова абетка» в AppStore.

Інтерфейс застосунку «Мишеняткова абетка» (рисунок 1.5) приємний, кольори яскраві та зачіпають око, шрифти читабельні. Певні дії та матеріали озвучуються, що може бути великою перевагою, за умови справної роботи, адже дитина зможе використовувати його самостійно. Втім частина дій в межах програми не є інтуїтивно зрозумілими, що може викликати труднощі під час першого використання. Окрему увагу варто звернути на персонажа-символа – мишеня, оскільки такий підхід не тільки підвищує впізнаваність продукту, а й робить його цікавішим для дітей. Щодо напрямку навчання, то цей продукт містить три підходи до вивчення абетки, що відповідають різним рівням знань та вмінь, а саме: вивчення окремих літер, шляхом використання візуальних та аудіо матеріалів (рисунок 1.5 (б)); пошук конкретної літери серед декількох опцій (рисунок 1.5 (в)) та заповнення пропусків у словах (рисунок 1.5. (г)). Рівні визначені заздалегідь та не адаптуються під потреби конкретної дитини.

Рисунок 1.5 – Знімки екрану застосунку «Мишеняткова абетка». Меню (а), перший рівень навчання(б), другий рівень навчання (в), третій рівень навчання (г).

Загалом «Мишеняткова абетка» – чудовий приклад якісного застосунку для дітей. Однак недоліками слугують відсутність адаптації під знання дитини, труднощі при першому використанні та проблеми з озвучуванням.

1.2.2. Аналіз застосунку «Нумо»

Наступним для аналізу обрано застосунок «Нумо» [6]. Його рейтинг у AppStore (рисунок 1.6) становить 3.6/5, що на 0.8 бала нижче, ніж у ресурсу «Мишеняткова абетка» та є досить низькою оцінкою.

Рисунок 1.6 – Рейтинг застосунку «Нумо» в AppStore.

У найбільш популярних відгуках (рисунок 1.7) користувачі зазначають, що застосунок є надто незрозумілим для дітей та важким у користуванні. Також згадується несправність функціональності озвучування, що є схожим до попереднього проаналізованого застосунку.

Рисунок 1.7 – Найрелевантніші відгуки на застосунок «Нумо» в AppStore.

Щодо користувацького інтерфейсу застосунку «Нумо» (рисунок 1.8), то візуально він є приємним завдяки кольорам та ілюстраціям, що зацікавлюють та утримують увагу.

Рисунок 1.8 – Знімки екрану застосунку «Нумо». Вигляд тестів.

Тим не менш користувацький досвід має недоліки. Першою незручністю є складна навігація, адже для кожного окремого завдання у розділі потрібно перейти на нього з меню та, після виконання, самостійно повернутись, щоб мати змогу перейти до наступного. Також між меню та завданнями існує проміжний екран (рисунок 1.9), на який потрібно натиснути, щоб відобразити умову, що є надлишковою дією.

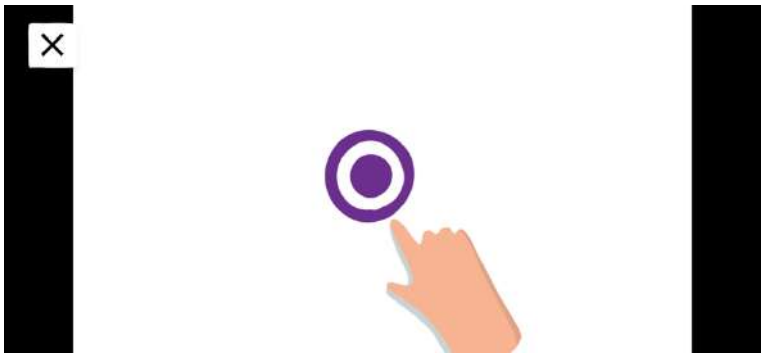


Рисунок 1.9 – Знімок екрану застосунку «Нумо». Екран між меню та завданням.

Напрямком навчання в межах застосунку є вивчення та перевірка розуміння базових понять щодо навколишнього середовища. Завдання різнопланові, цікаві та пізнавальні, проте часто важко зрозуміти, як саме їх виконати. Матеріали поділяються на шість рівнів за віковими категоріями – від 3 до 6 років. Рівні потрібно обирати вручну при реєстрації або в меню (рисунок 1.10). Відслідковування прогресу навчання відсутнє.

а) б)

Рисунок 1.10 – Знімок екрану застосунку «Нумо». Вибір рівня дитини (а) та меню з темами (б).

Застосунок містить звуковий супровід, щоправда іноді він перестає працювати, що унеможливлює самостійне використання його дитиною. Також суттєвим недоліком є відсутність пояснення неправильної відповіді. Загалом «Нумо» – це непоганий застосунок для дітей, однак він має велику кількість недоліків, які погіршують користувацький досвід.

1.2.3. Аналіз застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ»

Останнім для аналізу було обрано застосунок «ВИВЧАЮ – НЕ ЧЕКАЮ» [7], який користувачі AppStore оцінили так само, як і «Нумо» – на 3.6 бала (рисунок 1.11).

Рисунок 1.11 – Рейтинг застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ» в AppStore.

Серед найрелевантіших відгуків для цього застосунку (рисунок 1.12) можна побачити, що він подобається користувачам, проте такі проблеми як аварійні завершення роботи програми, неможливість відкрити розділи для навчання та помилки зі збереженням прогресу негативно впливають на користувацький досвід.

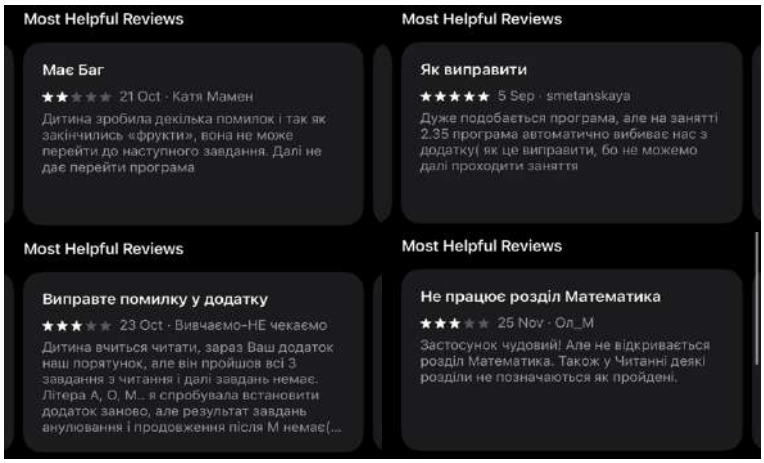


Рисунок 1.12 – Найрелевантніші відгуки на застосунок «ВИВЧАЮ – НЕ ЧЕКАЮ» в AppStore.

Інтерфейс застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ» (рисунок 1.13) викликає позитивні емоції, кольори спокійні, ілюстрації приємні та цікаві, шрифти читабельні.

Рисунок 1.13 – Знімок екрану застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ». Інтерфейс проміжного екрану.

Навігація в межах програми досить складна, але інтуїтивно зрозуміла. Одним з недоліків є довгий процес реєстрації, який містить десять кроків, що унеможливлює самостійний початок використання програми дитиною (рисунок 1.14).

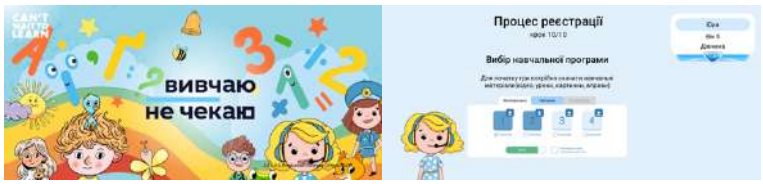
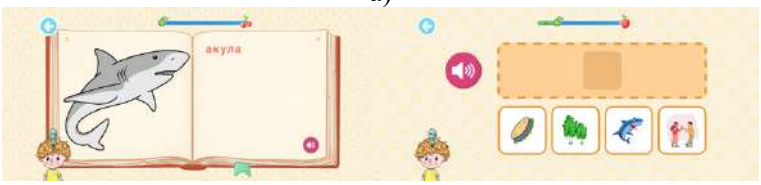


Рисунок 1.14 – Знімок екрану застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ». Онбординг користувача.

Як і в попередніх проаналізованих застосунках, тут є озвучення завдань та дій. Також, як і в «Мишеняткова абетка», в програмі є персонаж-символ – дівчинка Софійка, що слугує додатковим зацікавленням для дітей. Напрямки навчання в межах застосунку – математика та українська мова. Модулі містять дуже якісні та цікаві аудіо, відео та візуальні матеріали для навчання й різні типи тестів (рисунок 1.15). На відміну від попередніх продуктів, у цьому присутнє збереження прогресу навчання. Пояснення неправильних відповідей у тестах відсутнє, що є недоліком, адже протилежний підхід також слугував би додатковим навчанням, а не лише перевіркою знань.

а)



б)

Рисунок 1.15 – Знімок екрану застосунку «ВИВЧАЮ – НЕ ЧЕКАЮ». Тести та матеріали з вивчення математики (а). Тести та матеріали з вивчення літер (б).

Загалом застосунок «ВИВЧАЮ – НЕ ЧЕКАЮ» – це гарний приклад програмного продукту для навчання дітей, проте, опираючись на відгуки користувачів він часто працює несправно, що погіршує користувацький досвід.

1.3. Висновки до Розділу 1

Аналіз тривалості використання цифрових пристроїв дітьми віком від 5 до 8 років показав, що більшість з них проводять від 1 до 3 годин у гаджетах, що є високим показником для такої вікової категорії. Оскільки заборона використання ЦП в теперішніх реаліях неможлива, тому потрібно шукати підходи та способи зробити цей час цікавим та корисним для дитини. Одним з рішень є забезпечення проведення максимальної кількості екранного часу в процесі навчання та розвитку.

Досліджено типи електронних ресурсів, які використовують діти дошкільного та раннього шкільного віку, та їхній вплив на розвиток когнітивних навичок. Визначено, що більшість використовуваних ресурсів – розважальні та можуть нести потенційну шкоду дітям. Однак освітні ресурси

займають друге місце в рейтингу використання, що підтверджує наявність інтересу до такого типу застосунків у дітей.

Проаналізувавши ринок освітніх продуктів для дошкільнят та дітей раннього шкільного віку, було визначено, що українських застосунків такого типу є небагато, а існуючі не завжди задовольняють потреби користувачів. Серед основних недоліків є складність навігації та самостійного використання їх дітьми, несправність звукового супроводу, відсутність персоналізації та адаптації під знання дитини, відсутність відслідковування прогресу та навіть аварійні завершення роботи програм.

Отже, виникає необхідність у розробці інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання, яка б забезпечувала ефективне та персоналізоване навчання відповідно до індивідуальних особливостей кожної дитини.

РОЗДІЛ 2: Проєктні засади розробки iOS-застосунку «Інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання»

Важливим кроком у процесі створення програмного продукту є етап планування, від якісного виконання якого залежить успіх проєкту, оскільки він формує фундаментальні засади розробки. У цьому розділі окреслено мету роботи, постановку завдання та функціональні вимоги. Також описано процес проєктування структури застосунку, внутрішньої бази даних та прототипу користувацького інтерфейсу; створення ілюстративних, навчальних та тестових матеріалів. Продемонстровано процедуру розробки та тренування моделей машинного навчання для задачі класифікації зображень.

2.1. Обґрунтування мети та постановка завдань розробки

Опираючись на дослідження, проведене у першому розділі, метою роботи є розробка комплексної інтелектуальної системи для дітей дошкільного та раннього шкільного віку на основі адаптивного навчання. Впровадження інтелектуальних складових у програмний продукт сприятиме покращенню користувацького досвіду та ознайомленню дитини з сучасними технологіями. Адаптивний підхід до навчання, в свою чергу, забезпечить персоналізацію під потреби конкретної дитини та підвищить ефективність навчального процесу. Такий продукт дозволить дітям проводити екранний час з користю та зменшить негативний вплив цифровізації на розвиток їхніх когнітивних навичок шляхом компенсації кількості часу, який міг потенційно бути витраченим на відеоігри та соціальні мережі. Визначено основні завдання роботи, а саме:

– Сформувати функціональні вимоги до програмного продукту.

– Спроєктувати структуру застосунку.

– Спроєктувати та розробити внутрішню базу даних.

– Спроєктувати та розробити інтерактивний прототип користувацького інтерфейсу.

– Спроєктувати та розробити бази навчальних матеріалів та тестів.

– Створити ілюстративні матеріали для застосунку.

– Розробити та натренувати моделі машинного навчання для задач класифікації зображень.

– Розробити комплексний iOS застосунок для інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання.

2.2. Формування функціональних вимог

Перед початком проєктування складових частин програмного продукту важливо окреслити очікуваний результат, описавши функціональні вимоги, що визначають, як саме повинна або не повинна поводитись система в конкретних випадках. Опираючись на вищенаведені дослідження та твердження, визначено такі вимоги до нашої системи:

– Користувач повинен мати можливість реєстрації.

1. Система повинна надавати форму для заповнення особистих даних користувача.

2. Система повинна гарантувати безпеку, цілісність та несуперечливість даних.

3. Система повинна дозволяти користувачеві відкласти процес реєстрації на потім (з обмеженням певних дій всередині програми до моменту реєстрації).

– Система повинна надавати користувачеві інтуїтивну та просту навігацію між модулями програми.

– Користувач повинен мати можливість перегляду навчальних матеріалів.

1. Система повинна містити навчальні матеріали різних типів.

2. Система повинна коректно відображати усі складові частини навчальних матеріалів.

3. Система повинна озвучувати текстові складові навчальних матеріалів та аудіо матеріали.

– Користувач повинен мати можливість проходити адаптивні тести за різними темами.

1. Система повинна містити тестові матеріали спрямовані на перевірку формування базових навичок.

2. Тести повинні бути різнотипові, відповідати окремим рівням знань і вмінь.

3. Система повинна забезпечувати динамічну адаптацію тестів відповідно до знань дитини.

4. Система повинна адаптуватись під відображення різних типів тестів.

5. Система повинна озвучувати умови завдань та текстові варіанти відповідей.

6. Система повинна коректно визначати та зберігати результати виконання завдань.

7. Система повинна вираховувати час, витрачений на виконання окремих завдань.

8. Система повинна повідомляти про результат виконання кожного тесту та використовувати мотивуючі фрази для заохочення дитини.

9. Система повинна показувати та пояснювати результат у разі неправильної відповіді.

– Користувач повинен мати можливість вивчати букви, цифри та геометричні фігури шляхом використання моделей машинного навчання.

1. Система повинна надавати інтерфейс для розпізнавання та класифікації букв, цифр та геометричних фігур за допомогою камери телефона.

2. Система повинна озвучувати результат класифікації цифр та фігур двома мовами – українською та англійською.

3. Система повинна озвучувати результат класифікації букв українською мовою.

– Користувач повинен мати можливість переглядати сформовані навички на основі виконаних тестів.

1. Система повинна надавати інтерфейс для структурованого перегляду сформованих навичок.

2. Система повинна динамічно вираховувати сформовані навички.

– Система повинна коректно вираховувати рівень знань користувача.

1. Система повинна комплексно аналізувати результати виконання різних типів тестів користувачем.

2. Система повинна динамічно змінювати рівень для окремих навчальних модулів.

– Система повинна забезпечувати персистентне збреження даних та відновлення стану після перезапуску.

2.3. Опис структури застосунку

На основі сформованих функціональних вимог окреслено основні складові елементи системи, а саме:

– Модуль реєстрації – форма для заповнення даних (з можливістю відтермінування етапу).

– Модуль головного меню – сторінка з навігацією до основних складових частин програми.

– Модуль проміжного меню – сторінка з вибором тем навчання, тестування або дослідження та переходом на окремі сторінки для кожної теми.

– Модуль навчальний – інтерактивні картки з навчальними матеріалами та можливістю вільної навігації між ними.

– Модуль тестування – відображення динамічно підібраних тестів відповідно до рівня знань користувача. Тести відображаються по одному з можливістю переходу до наступного, після виконання поточного.

– Модуль дослідницький – екран з доступом до камери для розпізнавання та класифікації об’єктів.

– Модуль аналітики – сторінка з відображенням сформованих навичок на основі пройдених тестів.

2.4. Проєктування, формування та наповнення баз тестів і навчальних матеріалів

Перед початком розробки баз тестів та навчальних матеріалів опрацьовано професійну педагогічну літературу [8], щоб забезпечити відповідність контенту вимогам та нормам освітніх стандартів, навчальних програм і методичних рекомендацій. Також в процесі розробки проведено

консультації з працівниками освітньої галузі, які допомагали покращувати якість матеріалів та їхню відповідність потребам дітей дошкільного та раннього шкільного віку. Всі створені матеріали та тести спрямовані на формування ключових компетенцій дитини [9], а саме: особистісної, мовленнєво-комунікативної, соціальної та громадянської, математичної, природничої, предметно-практичної та технологічної, екологічної та мистецької. Опираючись на описані компетенції визначено основні напрямки навчання та вміння, на які буде спрямована основна увага в межах інтелектуальної системи підготовки дітей до шкоди на основі адаптивного навчання.

Першим таким напрямком є вивчення математики. В межах цього блоку дитина повинна опанувати такі навички:

– Розпізнавати, називати та писати усі числа від 0 до 10.

– Рахувати кількість об’єктів на зображенні.

– Знаходити певне число серед інших.

– Обчислювати арифметичні вирази з числами та предметами в межах 10.

– Розпізнавати та називати основні геометричні фігури.

– Визначати кількість кутів геометричних фігур.

– Знаходити певну геометричну фігуру серед інших.

Другим напрямком навчання є формування базових знань української мови. Навичками, які дитина має опанувати в межах цього блоку є:

– Розпізнавати на зображенні та на слух усі літери українського алфавіту.

– Називати усі літери українського алфавіту.

– Писати усі літери українського алфавіту.

– Знаходити певну літеру серед інших.
– Заповнювати пропущені літери у словах.
– Розпізнавати слова на слух.
– Складати слова із запропонованих літер.
– Пояснювати реченням, що відбувається на зображенні.

Третім напрямком є розвиток логічного мислення та знань про навколишнє середовище. В межах цього блоку дитина повинна опанувати такі навички:

– Розпізнавати та класифікувати людські емоції.
– Аналізувати та визначати властивості об’єктів.
– Знаходити відсутню частину зображення.
– Аналізувати зображення та знаходити логічні невідповідності.
– Комплексно аналізувати послідовності та визначати логічний наступний або зайвий елемент.
– Аналізувати та знаходити предмети, тварин або рослини на зображеннях, які відносяться до одного класифікаційного ряду.
– Знати пори року та їх ознаки.
– Знати основні професії та вміти їх описати.
– Розв’язувати базові задачі на логіку.
– Знати, вміти розпізнавати та називати основні кольори.
– Вміти передбачувати результати змішування основних кольорів.

Останнім напрямком є загальне та патріотичне виховання, в межах якого дитина повинна опанувати базові знання про себе, свою родину та країну. Результатом опрацювання цього блоку буде формування таких вмінь як:

– Знати прізвище ім’я та по батькові батьків.

– Знати своє прізвище ім’я та по батькові.
– Знати свій вік.
– Знати країну, населений пункт та адресу свого проживання.
– Знати державні, національні та традиційні символи України.

2.4.1. Обґрунтування вибору формату для проміжного збереження матеріалів застосунку

Перед початком розробки бази навчальних матеріалів і тестів, необхідно визначити оптимальний формат зберігання цих даних. Враховуючи, що підготовлений контент має завантажуватися до локальної бази даних застосунку під час першого запуску, обраний формат повинен бути достатньо легким, зручним для обробки та забезпечувати просту серіалізацію та десеріалізацію даних, підтримувати ієрархічні структури і дозволяти легко модифікувати або розширювати структуру контенту в майбутньому без суттєвих змін у коді застосунку. Проаналізувавши доступні на сьогодні формати зберігання даних, такі як XML [10], CSV[11] та JSON [12], було визначено, що найбільш доцільним варіантом є використання останнього формату. Він вирізняється простою структурою, компактністю та високою читабельністю, що значно спрощує як розробку, так і налагодження системи. Додатковою перевагою JSON є його нативна підтримка в екосистемі iOS.

2.4.2. Розробка компонентів та моделювання ієрархічних структур даних для систематизації бази навчальних матеріалів

Опираючись на описані раніше напрямки навчання сформовано п’ять окремих блоків навчальних матеріалів, а саме: букви, цифри, фігури, кольори,

патріотичне виховання та навчання письма. Визначено основні складові кожного з компонентів блоків та узагальнено їх, визначивши елементи, що є спільними. На основі цих компонентів створено базову структуру JSON (рисунок 2.1), яка слугуватиме для проміжного збереження матеріалів.

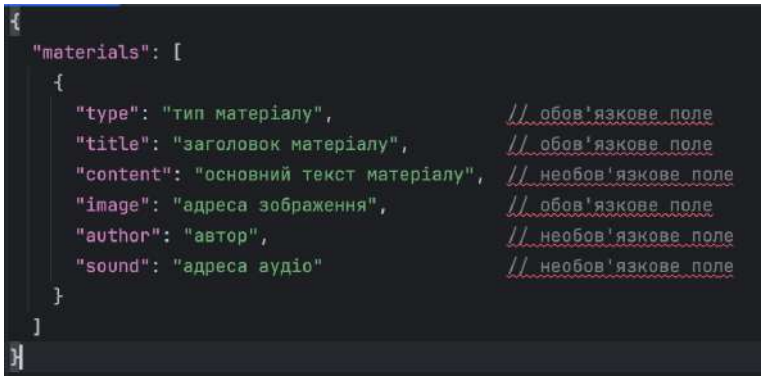


Рисунок 2.1 – Структура JSON для збереження матеріалів.

Об’єкти розробленої структури матеріалів містять обов’язкові поля – тип матеріалу, заголовок і зображення, та необов’язкові – основний текст матеріалу, ім’я автора та адреса аудіо супроводу. Тип матеріалу має обмежений перелік значень та відповідає назві окремого блоку, що слугує для поділу контенту в межах застосунку.

2.4.3. Розробка компонентів та моделювання ієрархічних структур даних для систематизації бази тестів

Відповідно до раніше визначених напрямів навчання виокремлено сім блоків тестових завдань, спрямованих на розвиток мислення, знань про навколишнє середовище, вивчення букв, цифр, кольорів, фігур, а також засвоєння персональних даних. Спроєктовано узагальнену структуру JSON тестового завдання (рисунок 2.2) та визначено обов’язкові та необов’язкові поля.

Рисунок 2.2 – Структура JSON для збереження тестів.

Кожен об’єкт має складну ієрархічну структуру, що включає обов’язкове поле тип, вкладений список умови, масив відповідей та поле складності. Список умови містить обов’язкове поле з текстом умови, а також два опціональні поля – альтернативний текст і масив зображень. Масив відповідей складається з вкладених структур з опціональними полями: зображенням (за наявності тексту), текстом (за наявності зображення), булевою позначкою правильної відповіді (за наявності тексту або позиції), номером позиції (за наявності тексту або позначки правильності) та поясненням. Поле складності використовується для визначення рівня складності завдання та підтримки функціонування адаптивного алгоритму.

Всі тести поділено на одинадцять типів, які визначають формат тесту, наявність опціональних полів та майбутній вигляд на користувацькому інтерфейсі. Нумерація типів починається з нуля.

– Тип 0: тестове завдання даного типу містить умову, представлену у вигляді тексту та зображення. Варіанти відповіді подані двома текстовими елементами, один з яких позначений як правильний.

– Тип 1: умова тестового завдання представлена виключно текстовою інформацією. Варіанти відповіді подані у вигляді двох зображень, серед яких одне визначене як правильне.

– Тип 2: умова включає текстовий опис та ілюстративний матеріал у вигляді зображення. Варіанти відповіді представлені чотирма зображеннями, одне з яких є правильним.

– Тип 3: умова тестового завдання містить текст та зображення. Відповідь надається користувачем шляхом голосового або текстового введення.

– Тип 4: структура умови аналогічна до попереднього типу. Відмінністю слугує відсутність відображення тексту умови на екрані.

– Тип 5: умова тестового завдання складається з основного тексту та альтернативного текстового опису. Варіанти відповіді представлені чотирма зображеннями, серед яких визначено правильний варіант. Особливістю цього типу є те, що альтернативний текст не відображається у візуальному інтерфейсі, а використовується виключно для звукового відтворення.

– Тип 6: умова також містить основний текст та альтернативний текстовий опис. Варіанти відповіді представлені чотирма зображеннями з позначенням правильного варіанта. На відміну від попереднього типу, альтернативний текст відображається безпосередньо в інтерфейсі користувача.

– Тип 7: умова тестового завдання складається з текстового та альтернативного текстового опису. Варіанти відповіді представлені чотирма текстовими полями, кожне з яких має відповідну позначку правильності. Альтернативний текст лише озвучується, проте не відображається на інтерфейсі.

– Тип 8: умова містить текстову інформацію та чотири зображення. Варіанти відповіді також представлені чотирма зображеннями, серед яких визначено правильний варіант.

– Тип 9: умова тестового завдання включає текстовий опис та декілька зображень. Виконання завдання передбачає впорядкування поданих зображень у правильній послідовності.

– Тип 10: умова містить текстовий опис. Варіанти відповіді представлені чотирма зображеннями, причому система допускає наявність кількох правильних варіантів відповіді.

2.4.4. Розробка компонентів та моделювання ієрархічних структур даних для систематизації бази досягнень

Опираючись на навички, які має опанувати дитина в межах застосунку, окреслено компоненти бази досягнень та створено для них структуру JSON (рисунок 2.3) для зручного проміжного збереження.

```
{
    "skills": [
        {
            "type": 0,                          // обов'язкове поле, тип тесту
            "module": "назва навчального блоку", // обов'язкове поле
            "difficulty": 0,                    // обов'язкове поле, складність
            "title": "опис навички"             // обов'язкове поле
        }
    ]
}
```

Рисунок 2.3 – Структура JSON для збереження досягнень.

Розроблена структура містить чотири обов’язкові поля. Тип відповідає типу тестового матеріалу, модуль – назвам навчальних та тестових блоків. Складність визначається на основі пройдених тестів. Опис досягнення формується відповідно до навичок, описаних раніше.

2.5. Проєктування й реалізація прототипу користувацького інтерфейсу застосунку та ілюстративних матеріалів

Перед тим як розпочати написання програмного коду, важливо спочатку створити прототип користувацького інтерфейсу. Такий підхід дозволяє структурувати попередні напрацювання та чітко окреслити очікуваний результат. Прототипування забезпечує візуалізацію концепції, що дозволяє виявити недосконалості в дизайні та логіці навігації до початку розробки, що економить час і ресурси.

2.5.1. Обґрунтування вибору стилістичних рішень з урахуванням специфіки аудиторії

Проаналізувавши дослідження впливу колірної гами навчальних матеріалів на сприйняття дитиною інформації [13] визначено основні кольори, що будуть використовуватись в інтерфейсі користувача, а саме: блакитний, жовтий, зелений, білий та помаранчевий. Така палітра сприятиме кращій концентрації уваги та активізації розумової діяльності. Головними шрифтами обрано «Comic Sans MS» для основного та «Inter» для додаткового тексту. Вибір основного шрифту зумовлений його декоративністю та креативністю, що зацікавлює дітей, проте не порушує читабельності тексту. Додатковий шрифт обрано простим та універсальним для використання в інформаційних блоках, де потрібна висока розбірливість та нейтральний стиль.

2.5.2. Характеристика компонентів прототипу застосунку

Для розробки прототипу застосунку використано інструмент Figma [14]. Вибір цієї платформи зумовлений її широкими можливостями для створення високоякісних макетів та інтерактивних прототипів. В процесі роботи основну увагу звернуто на принципи орієнтації на користувача, простоти та інтуїтивності, візуальної ієрархії, доступності та адаптивності. Прототип

створено за компонентним підходом, що підвищує його якість та масштабованість. Розроблено окремі компоненти (рисунок 2.4) для фонів, кнопок, опцій меню, шаблонів тестів та персонажа-символа застосунку.

Рисунок 2.4 – Компоненти прототипу застосунку.

Навігаційні шляхи в межах прототипу спроєктовано з урахуванням специфіки цільової аудиторії та забезпечено їхню інтуїтивність та простоту. Вхідною точкою є форма реєстрації (рисунок 2.5), яка складається з двох кроків. Підхід розділення процесу реєстрації та відображення прогресу забезпечує кращий користувацький досвід. Така сегментація зменшує когнітивне навантаження, мотивує користувачів та дозволяє легко повернутися до попередніх етапів для виправлення можливих помилок.

Рисунок 2.5 – Форма реєстрації.

Основним навігаційним вузлом слугує меню застосунку, яке поділене на головне (рисунок 2.6 (а)) та допоміжні (рисунок 2.6 (б)). Перше слугує для навігації між блоками програми, інші – для навігації всередині блоків. Опції супроводжуються ілюстраціями для підвищення інтуїтивності користування. Всі елементи мають великий масштаб для запобігання помилковим переходам.

а) *б)*

Рисунок 2.6 – Головне меню (а) та додаткові меню (б) застосунку

Блок з навчальними матеріалами (рисунок 2.7) містить представлення ілюстративних та текстових даних. Всі об’єкти відображаються як картки з можливістю вільної навігації між ними.

Рисунок 2.7 – Приклад відображення навчального блоку.

Прототип тестового блоку (рисунок 2.8) демонструє відображення усіх одинадцяти типів тестів та визначає розташування, стиль та розмір усіх відображуваних елементів кожного тесту.

Рисунок 2.8 – Приклад відображення тестового блоку.

Додатково в межах блоку тестів розроблено компоненти відображення успіху та невдачі (рисунок 2.9), що представлені текстовим повідомленням та зображенням персонажа-символа.

а) *б)*

Рисунок 2.9 – Відображення успіху (а) та невдачі (б).

Дослідницький блок (рисунок 2.10) представлений компонентом доступу до камери телефона, результатом класифікації об’єкта моделлю машинного навчання та кнопками для озвучування результату.

Рисунок 2.10 – Приклад відображення дослідницького блоку.

Блок досягнень (рисунок 2.11) представлений списком сформованих навичок у форматі робочого зошита, що слугує асоціацією з навчальним процесом.

Рисунок 2.11 – Приклад відображення блоку досягнень.

Додатково прийнято рішення розробити адаптивний дизайн, що змінюється залежно від пори року (рисунок 2.12). Сезонна тематика дизайну підвищує залученість дітей та органічно інтегрує освітній контент про природні явища та сезонні зміни.

Рисунок 2.12 – Приклад адаптації дизайну під пору року.

На окрему увагу заслуговує персонаж-символ (рисунок 2.13), що супроводжує користувача в окремих блоках застосунку та адаптує свій вигляд разом із рештою елементів дизайну. Такий підхід підвищує зацікавленість дітей, слугує додатковим елементом гейміфікації та підвищує впізнаваність продукту серед користувачів. В ролі символа обрано кота, оскільки таким чином можна не лише зацікавити дитину, а й сформувати прихильне ставлення та повагу до тварин.

Рисунок 2.13. – Персонаж-символ застосунку.

В межах створення прототипу також розроблено символіку застосунку, а саме назву та логотип (рисунок 2.14).

Рисунок 2.14 – Логотип застосунку.

Назвою обрано словосполучення «До школи», що несе в собі декілька значень. По-перше, це підготовка до школи як процес. По-друге, це вказівка на те, що потрібно робити перед тим, як піти до школи. Така назва робить бренд запам'ятовуваним та одночасно висловлює основну мету продукту. Логотип містить декоративний елемент – дзвінок, що асоціюється безпосередньо з навчанням. Шрифт обрано декоративний, що справляє враження крейдового надпису та формує додаткову асоціацію.

2.5.3. Створення ілюстративних матеріалів для застосунку

Ілюстративні матеріали для застосунку розроблено з використанням комбінованого підходу, що поєднує можливості штучного інтелекту та професійних дизайн-інструментів. Складні ілюстрації (рисунок 2.15) створювались за допомогою генеративної моделі штучного інтелекту Gemini [15], що дозволило швидко створити різноманітні варіанти і знайти оптимальний стиль, відповідний загальному дизайну продукту.

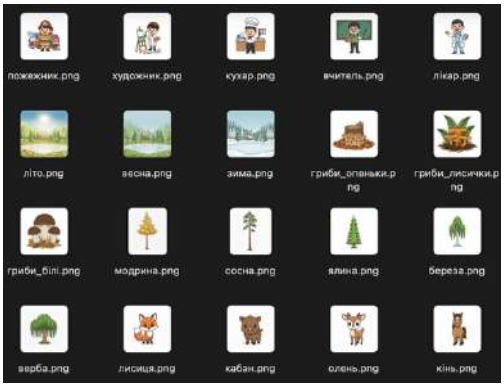


Рисунок 2.15 – Приклад ілюстрацій, створених за допомогою Gemini.

Прості ілюстрації (рисунок 2.16) – зокрема геометричні фігури, літери, числа та поодинокі об'єкти – створювались у Figma, оскільки в їхньому випадку час на генерацію у ШІ не виправдовував себе та вручну було швидше й ефективніше скласти їх із базових елементів.

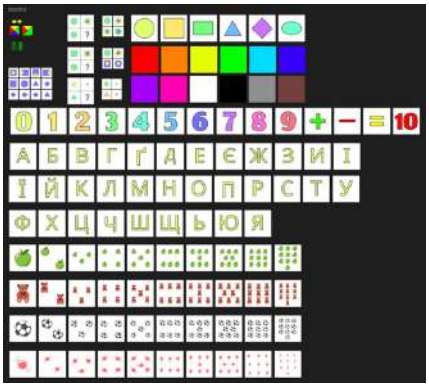


Рисунок 2.16 – Ілюстрації, створені у Figma.

Такий підхід до створення ілюстративних матеріалів забезпечив оптимальний баланс між якістю дизайну, швидкістю розробки та економією ресурсів.

2.6. Розробка та тренування моделей машинного навчання для задачі класифікації зображень

Для додаткової гейміфікації навчального процесу та ознайомлення дітей з сучасними технологіями вирішено інтегрувати до застосунку модуль машинного навчання, спеціалізований на розпізнаванні та класифікації візуальних об'єктів.

2.6.1. Формування та попередня обробка наборів даних для навчання

Під час дослідження доступних публічних наборів тренувальних даних для задач класифікації літер українського алфавіту, цифр та геометричних фігур виявлено їх недостатність для потреб проєкту. Аналіз показав, що існуючі датасети часто демонструють низькі показники точності навчання та

валідації або не відповідають специфіці продукту. З огляду необхідність забезпечити високу точність розпізнавання в умовах реального використання, прийнято рішення про самостійне створення наборів тренувальних даних. Для цього створено спеціальні скрипти на мові програмування Python [16].

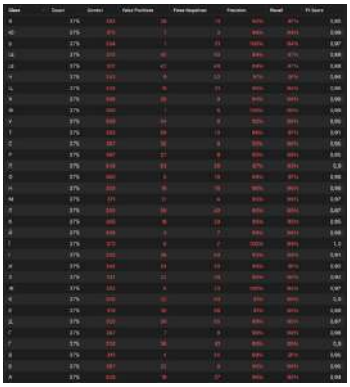
Для геометричних фігур створено синтетичні зображення десяти типів, а саме: круг, квадрат, трикутник, шестикутник, зірка, овал, ромб, трапеція, паралелограм, серце. Малювання для кожного типу фігури виконується з використанням вбудованих методів бібліотеки PIL [17]. Для цифр від 0 до 9 та літер від А до Я створено синтетичні зображення, шляхом растеризації тексту на основі різних шрифтів. Для збільшення різноманітності та імітації реальних умов застосовується комплексна аугментація даних. Зображення піддаються випадковому масштабуванню, зміщенню від центру у різних напрямках та повороту. Додатково 30% зображень розмиваються за допомогою гаусівського фільтра, а 30% отримують синтетичний шум для симуляції низької якості камери мобільного пристрою. Усі згенеровані зображення розподілено на три підмножини для різних етапів розробки моделі. Загальний обсяг кожного з наборів даних становить 25 000 зображень, які розділяються у пропорції 70% для тренування, 15% для валідації та 15% для тестування. Всі зображення розміщені у окремих іменованих папках, відповідно до назв класів.

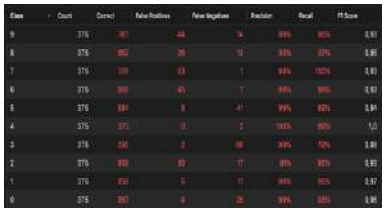
2.6.2. Процедура навчання та валідація інтелектуальних моделей

Навчання моделей виконувалось за допомогою інструмента Create ML [18], який інтегрований у середовище розробки Xcode [19] та спеціалізований на створенні компактних моделей машинного навчання для екосистеми Apple. Цей інструмент забезпечує візуальний моніторинг процесу через графіки точності та втрат на тренувальному та валідаційному наборах, що дозволяє контролювати процес тренування. Результатом навчання стали три

скомпільовані моделі у форматі .mlmodel, придатні для прямого інтегрування в iOS-застосунок.

Модель для класифікації літер показала 98% точності під час тренування та 97% під час валідації, для цифр – 99% та 98% відповідно, для фігур – 100% в обох випадках. Точність тестування для літер та цифр однакова – 94% (рисунок 2.17).





а) б)

Рисунок 2.17 – Звіт щодо тестування моделі класифікації літер (а) та цифр (б).

Тестування моделі класифікації геометричних фігур (рисунок 2.18) показало точність 95%.

Class	Count	Correct	False Positives	False Negatives	Precision	Recall	F1 Score
triangle	375	369	0	6	100%	98%	0,99
trapezium	375	352	6	23	98%	94%	0,96
star	375	375	0	0	100%	100%	1,0
square	375	254	2	121	99%	68%	0,81
rhombus	375	367	17	8	96%	98%	0,97
parallelogram	375	372	146	3	72%	99%	0,83
oval	375	374	0	1	100%	100%	1,0
hexagon	375	359	1	16	100%	96%	0,98
heart	375	375	0	0	100%	100%	1,0
circle	375	375	6	0	98%	100%	0,99

Рисунок 2.18 – Звіт щодо тестування моделі класифікації геометричних фігур.

Такі показники якості забезпечують впевненість у коректності роботи модуля для кінцевих користувачів.

2.7. Висновки до Розділу 2

Опираючись попередні дослідження обґрунтовано мету та описано постановку завдань розробки. Сформовано функціональні вимоги, в яких

описано, як саме має поводитись система та, на основі цього, окреслено структуру застосунку.

Ґрунтуючись на опрацьованій професійній педагогічній літературі та консультаціях з працівниками освітньої галузі створено бази тестів та навчальних матеріалів. Описано напрямки навчання та навички, що будуть формуватися в межах системи. Обґрунтовано вибір формату для проміжного збереження матеріалів застосунку та розроблено ієрархічні структури для них. Розроблено та окреслено одинадцять типів тестових матеріалів, формат яких забезпечує адаптивність системи під потреби конкретного користувача. На основі створених баз матеріалів спроєктовано внутрішню базу даних застосунку для забезпечення персистентного збереження даних та відновлення стану після перезапуску.

Базуючись на попередніх напрацюваннях спроєктовано та реалізовано прототип користувацького інтерфейсу застосунку та створено ілюстративні матеріали. Опираючись на специфіку цільової аудиторії обґрунтовано вибір стилістичних рішень, охарактеризовано компоненти створеного прототипу і символіку програмного продукту, описано процес створення ілюстрацій.

Окреслено процес формування наборів тренувальних даних для навчання інтелектуальних моделей для задач класифікації зображень та результати їхнього навчання й валідації.

Проведена робота охопила ключові аспекти проєктування системи. Послідовне виконання всіх етапів забезпечило комплексний підхід до розробки, а отримані результати підтвердили обґрунтованість обраної методології та створили надійну основу для подальшої реалізації проєкту.

РОЗДІЛ 3: Розробка iOS-застосунку «Інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання»

3.1. Теоретичні засади розробки

Адаптивне навчання [20] – це концепція, яка передбачає корегування освітньої траєкторії під конкретного здобувача, опираючись на безперервний аналіз його дій, прогресу та динаміки. В інформаційних системах такий підхід забезпечується використанням комп'ютерних алгоритмів адаптивного навчання.

3.1.1. Аналітичний огляд алгоритмів адаптивного навчання

Для вибору оптимального адаптивного алгоритму проаналізовано три поширені існуючі рішення, а саме: Bayesian Knowledge Tracing (BKT) [21], Performance Factor Analysis (PFA) [22] та Item Response Theory (IRT) [23].

Перший підхід – BKT є одним із класичних алгоритмів, що стоять в основі численних адаптивних систем. Він базується на чотирьох показниках:

- $p(L_0)$: ймовірність того, що навичка вже опанована;
- p(T): ймовірність опанування навички під час наступної практики;
- p(S): ймовірність помилки для опанованої навички;
- p(G): ймовірність правильної відповіді для неопанованої навички;

Після кожної відповіді алгоритм обчислює апріорну ймовірність опанування навички на основі правильності відповіді (рисунок 3.1. (1b), (1c)), а кінцева ймовірність коригується з урахуванням можливості засвоєння навички в конкретний момент часу та прогнозується майбутній результат (рисунок 3.1 (1d), (1e)).

$$p(L_1)_u^k = p(L_0)^k, \tag{1a}$$

$$p(L_{t+1}|obs = correct)_u^k = \frac{p(L_t)_u^k \cdot (1 - p(S)^k)}{p(L_t)_u^k \cdot (1 - p(S)^k) + (1 - p(L_t)_u^k) \cdot p(G)^k}, \tag{1b}$$

$$p(L_{t+1}|obs = wrong)_u^k = \frac{p(L_t)_u^k \cdot p(S)^k}{p(L_t)_u^k \cdot p(S)^k + (1 - p(L_t)_u^k) \cdot (1 - p(G)^k)}, \tag{1c}$$

$$p(L_{t+1})_u^k = p(L_{t+1}|obs)_u^k + (1 - p(L_{t+1}|obs)_u^k) \cdot p(T)^k, \tag{1d}$$

$$p(C_{t+1})_u^k = p(L_t)_u^k \cdot (1 - p(S)^k) + (1 - p(L_t)_u^k) \cdot p(G)^k \tag{1e}$$

Рисунок 3.1 – Формули для обчислення ймовірностей BKT.

Підхід PFA (рисунок 3.2) базується на логістичній регресії та прогнозує ймовірність правильної відповіді (p), опираючись на базову легкість навички (β_j), історію попередніх успіхів ($s_{i,j}$) та невдач ($f_{i,j}$) та відповідні вагові коефіцієнти (γ_j, ρ_j)

$$\mathrm{m}(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \qquad \mathrm{p}(m) = \frac{1}{1 + e^{-m}}$$

Рисунок 3.2 – Формули для PFA.

Даний підхід, як і BKT є динамічним, проте більш поступовим та забезпечує вищу точність результатів.

Основою IRT є взаємодія рівня здібностей конкретного студента та характеристики окремого запитання. Існує три основні моделі IRT (рисунок 3.3).

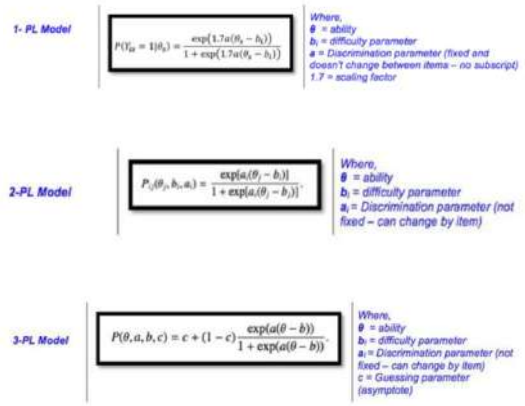


Рисунок 3.3 – Формули для IRT моделей.

Перша складається з одного параметра, що описує здібність (θ) особи, яка відповідає на завдання, а також параметра складності (b_i). Друга додатково враховує дискримінацію (a_i), а третя модель також враховує параметр вгадування (c_i). Даний алгоритм працює статично та визначає поточний рівень здібностей на основі взаємодії людини з конкретним завданням. Для його коректного функціонування необхідно заздалегідь визначити параметри кожного завдання шляхом попередньої калібрації на репрезентативній вибірці користувачів.

Проаналізувавши дані підходи, визначено, що жоден з них не відповідає повною мірою структурі розроблюваної системи у своєму класичному вигляді, проте їхня математична гнучкість дозволяє провести необхідну адаптацію для точнішого моделювання знань. Враховуючи переваги PFA над BKT та складність початкової калібрації IRT, прийнято рішення адаптувати для подальшого впровадження у систему перший з перелічених алгоритмів.

3.1.2. Адаптація обраного рішення до структури системи

Опираючись на створену структуру тестових матеріалів, внесено низку змін у класичну роботу алгоритму PFA. Перш за все, впроваджено адаптацію даного рішення до модульної структури тестового контенту та забезпечення ізоляції прогресу для кожного типу завдань в межах навчального блоку. Визначено модулі, до яких відноситься конкретна навичка та тип відповідних їй тестових матеріалів. Додатково адаптовано поняття базової легкості та обернено її на вагу складності для відповідності розробленій структурі тестів (рисунок 3.4). Також змінено підхід до визначення даного показника, адже в класичному варіанті він зафіксований для всіх матеріалів певної навички, в той

час, як в даному випадку – може варіюватись в межах окремих питань конкретного модуля та типу, що відносяться до однієї навички.

$$\beta_{test} = d_{test} \cdot t_{test} \quad (3.4)$$

де β_{test} *– вага конкретного питання за складністю(*d_{test}*) та типом(*t_{test}*);*

d_{test} *– складність конкретного питання;*

t_{test} *– вага тестового типу;*

Також для кожного тесту, опираючись на оптимальний (рисунок 3.5) та фактичний час виконання, що різниться для кожного окремого типу завдань, додано обчислення рівня впевненості (рисунок 3.6). Таким чином правильні відповіді, на які було витрачено меншу кількість часу мають більшу вагу в обчисленні прогресу, а ті, на які було витрачено більше часу – меншу.

$$T_{target} = \beta_{test} \cdot T_{base} \quad (3.5)$$

$$c_{test} = \omega \cdot (T_{target} / T_{actual}) \quad (3.6)$$

де T_{target} *– оптимальний час відповіді на питання (секунди);*

T_{base} *– еталонний час для виконання завдання зі складностю 1 (секунди);*

c_{test} *– впевненість при виконанні тесту;*

ω *– вага впевненості, константа;*

T_{actual} *– фактичний час, за який користувач відповів (секунди) обмежений знизу пороговим значенням у 30 с;*

Опираючись на усі внесені зміни, виведено формулу (рисунок 3.7), що відповідає специфіці системи. Формула для обчислення кінцевої ймовірності опанування навички залишилась незмінною.

$$L(t, m, d) = (\gamma \cdot s_{t,m,d} + \rho \cdot f_{t,m,d}) - \beta_{test} + c_{test} \quad (3.7)$$

де γ – вага успіху, константа;

$s_{t,m}$ – кількість успіхів для конкретного типу (t) у модулі (m) зі складністю (d);

ρ – вага невдачі, константа;

$f_{t,m}$ – кількість невдач для конкретного типу (t) у модулі (m) зі складністю (d);

Роботу алгоритму відображено у вигляді обчислювального графу (рисунок 3.8), де сірі елементи є стандартними для PFA, а зелені – модифікаціями для забезпечення відповідності структурі тестових матеріалів системи та специфіці цільової аудиторії.

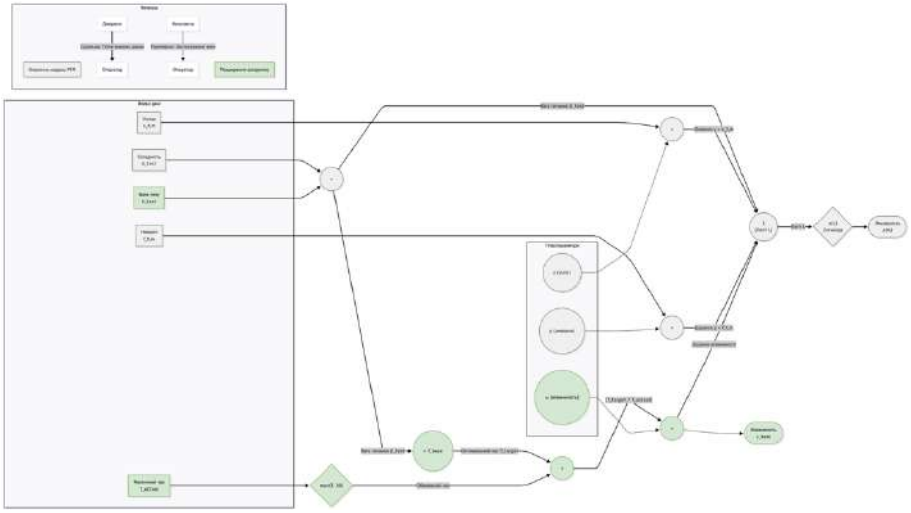


Рисунок 3.8 – Обчислювальний граф роботи алгоритму.

Підсумовуючи, на відміну від класичної моделі PFA, яка базується виключно на історії успіхів та невдач, розроблена модифікація враховує метрику швидкості, що дозволяє алгоритму оцінювати не лише факт наявності знань, а й рівень впевненості користувача, що є важливим для дошкільної освіти. Завдяки переходу від статичного параметра ваги до розрахунку на основі складності конкретного питання та ваги тестового типу – підвищено гнучкість алгоритму для застосування в контексті розроблюваної системи. Обернено поняття базової легкості навички на складність та внесено відповідні зміни до кінцевого обрахунку. Також забезпечено ізоляцію та акумуляцію прогресу в специфічних зрізах – для конкретного типу завдання, в межах конкретного модуля та для певного рівня складності. Це зумовлено специфікою структури тестового контенту та дозволяє алгоритму опрацьовувати ситуації, коли користувач успішно вирішує в межах теми завдання одного типу для певної складності, але помиляється в завданнях іншого типу на тому ж рівні.

3.2. Обґрунтування вибору технологічного стеку та інструментів розробки

Для розробки інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання було обрано екосистему iOS, а саме мову програмування Swift та поєднання фреймворків для побудови користувацьких інтерфейсів SwiftUI та UIKit. Такий вибір зумовлений низкою технічних і практичних переваг.

Перш за все, Swift забезпечує високу продуктивність виконання, що є критично важливим для дитячих освітніх застосунків, оскільки швидка реакція системи на дії користувача забезпечує підтримання уваги дитини та запобігає зниженню зацікавленості в процесі навчання. Додатковою перевагою є широка екосистема Apple для розробників [24], яка містить велику кількість

фреймворків, зокрема, інструменти для розпізнавання і синтезу мовлення та інтеграції моделей машинного навчання, які є важливими для реалізації інтелектуальних функцій розроблюваної системи. На окрему увагу заслуговує рівень безпеки платформи iOS, адже вона забезпечує ізоляцію застосунків у межах власного середовища виконання, а також надає надійні механізми для зберігання конфіденційних даних, зокрема, через використання Keychain.

Щодо переваг використання SwiftUI, то важливим чинником є його декларативний підхід до побудови інтерфейсів, який дозволяє значно зменшити обсяг програмного коду, знизити ймовірність виникнення помилок та прискорити процес розробки й супроводу застосунку. UIKit, в свою чергу, є більш гнучким та зрілим інструментом із ширшою підтримкою та краще підходить для створення складних елементів інтерфейсу. Поєднання цих двох фреймворків дозволяє ефективно створювати сучасні, якісні та комплексні користувацькі інтерфейси.

Середовищем розробки обрано Xcode – це нативний інструмент для створення застосунків в екосистемі Apple. Його використання зумовлене зручністю написання, налагодження та тестування коду. Головними перевагами слугують можливості попереднього перегляду розроблюваного інтерфейсу за допомогою SwiftUI Preview та відлагодження роботи застосунку на симуляторах та реальних пристроях. Це дозволяє оперативно оцінювати зовнішній вигляд і поведінку програми в умовах, максимально наближених до реального використання.

3.3. Обґрунтування вибору використаних фреймворків

В процесі розробки використано нативні та сторонні програмні фреймворки, що полегшують розробку та надають інструменти для виконання різних задач, а саме:

– Foundation надає базову функціональність для програм, зокрема роботу з даними та мережею.

– SwiftUI та UIKit використовуються для розробки користувацького інтерфейсу. Перший з них є основним та за його допомогою створено більшу частину застосунку, в той час як другий використовується виключно для роботи з камерою та налагодження авторського фреймворку для малювання.

– CoreData забезпечує збереження даних про тести, матеріали та досягнення користувача у локальній базі та надає зручні інструменти для роботи з ними. Цей фреймворк забезпечує персистентність об'єктів, керування змінами та виконання важких операцій у фоні.

– KeychainAccess [25] використовується для безпечного збереження персональних даних користувача. Він надає інструменти для легкого доступу до Keychain з коду та виконання дій з даними, що там знаходяться.

– Combine слугує інструментом для впровадження реактивного підходу у застосунок та використовується для швидкого реагування на розпізнавання об’єктів інтелектуальними моделями та відображення результату класифікації на користувацькому інтерфейсі.

– Vision забезпечує взаємодію між зображенням та інтелектуальними моделями. Цей фреймворк приймає на вхід фото, передає в Core ML-модель для розпізнавання, обробляє запит та повертає результат класифікації.

– AVFoundation використовується для інтеграції роботи з камерою у дослідницький модуль. Він охоплює роботу з перевірки, відтворення, обробки та створення аудіовізуальних медіа файлів.

– NaturalLanguage слугує інструментом для перетворення слів у їх базові форми – леми. Цей підхід використовується для спрощення та покращення аналізу текстових відповідей користувача.

– MicrosoftCognitiveServicesSpeech [26] це фреймворк від компанії Microsoft, який надає інструменти для синтезу мовлення. Він використовує передові моделі, включаючи українськомовну модель Остапа, яка вважається однією з найкраще натренованих для синтезу української мови. Мовлення генерується на серверах Microsoft, тому фреймворк працює виключно з доступом до мережі. Він слугує основним інструментом для озвучування текстового контенту системи.

– Speech використовується для забезпечення безперебійності роботи озвучування в застосунку, оскільки він є нативним для екосистеми Apple та працює без доступу до інтернету.

– Network забезпечує моніторинг підключення пристрою до мережі. Такий підхід використовується для автоматичного перемикання механізму озвучування з MicrosoftCognitiveServicesSpeech на нативний Speech в разі відсутності доступу до інтернету.

3.3.2. Архітектурні особливості та функціональне призначення авторського фреймворку SkipFrame

З метою забезпечення функціональності для навчання письма використано авторський фреймворк «SkipFrame» [27], що створений в межах курсу бакалаврської програми НаУКМА – «Вибрані фреймворки для iOS». Він

розроблений на базі UIKit та доступний для вільного використання через Swift Package Manager та CocoaPods. Даний фреймворк дозволяє легко додавати шар для малювання у будь-який контролер представлення та адаптувати його вигляд та функціональність під потреби застосунку. В його межах забезпечено функціональність для використання різних типів пензлів; зміни їх розміру, прозорості та кольору; можливість скасовувати й відновлювати зміни та повністю очищувати полотно; отримувати доступ до створених малюнків з коду та зберігати їх у системну галерею. Можливості адаптації включають в себе зміну піктограм елементів інтерфейсу, розміру полотна, перевпорядкування або приховування опцій меню та зміну його позиції. Фреймворк містить повну документацію та інструкції з інсталяції й використання.

Впровадження даного інструментального середовища в інтелектуальну систему підготовки дітей до школи на основі адаптивного навчання дозволило легко інтегрувати інструменти малювання, суттєво зменшило час на розробку та підвищило якість коду. Функціональність, додана за допомогою нього, збільшує практичну користь застосунку шляхом забезпечення розвитку дрібної моторики та формування навички письма.

3.4. Структура внутрішньої бази даних

Важливою характеристикою якісного програмного продукту є забезпечення персистентного збереження даних та відновлення стану після перезапуску. Цього можна досягти шляхом збереження усієї необхідної для роботи інформації у базі даних. В інтелектуальній системі підготовки дитини до школи на основі адаптивного навчання необхідно зберігати таку інформацію як навчальні матеріали, тести та досягнення користувача.

Опираючись на розроблені структури JSON кожного з блоків інформації та обраний адаптивний алгоритм, створено концептуальну модель «сутність-зв’язок» (рисунок 3.9), яка демонструє сутності, що зберігатимуться у базі. Для кожної з них визначено первинний ключ, обов’язкові й опціональні атрибути та зв’язки.

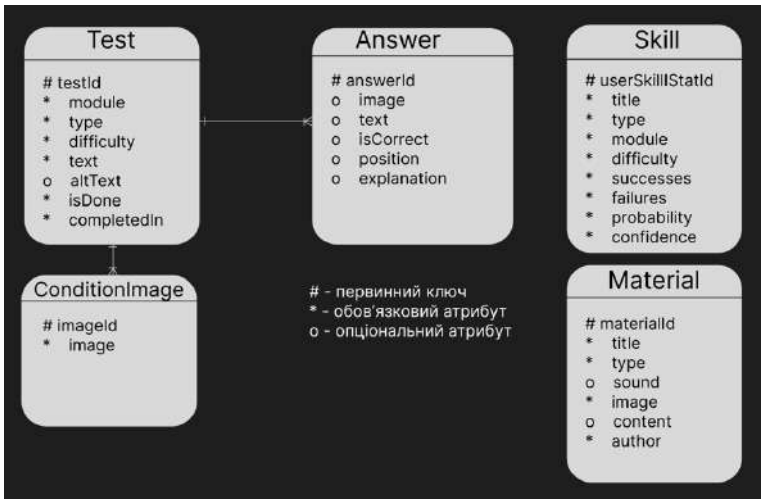


Рисунок 3.9 – Концептуальна модель «сутність-зв’язок» бази даних застосунку.

У даній моделі представлено п’ять класів-сутностей, три з яких – основні, а інші два – допоміжні. Першою з них є «Test» – це основна сутність, що являє собою представлення тестів у системі. Для неї створено допоміжну сутність «ConditionImage», що є зображенням в умові тесту. Між тестом і зображенням встановлено зв’язок типу «один-до-багатьох»: один тест може мати кілька зображень, але кожне зображення пов’язане лише з одним тестом. Ще однією допоміжною сутністю є «Answer», що є представленням варіанту відповіді до тесту. Між тестом і варіантом, як і у випадку із зображенням, встановлено зв’язок типу «один-до-багатьох». Також створено основну незалежну сутність «Material», яка слугує для збереження навчальних матеріалів у системі. Останньою основною сутністю є «Skill», що представляє навички, що формуватимуться в процесі роботи. Всі поля сутностей співвідносяться до

відповідних полів ієрархічних структур. Сутність «Test» доповнено двома полями, які визначають чи виконане завдання та скільки часу витрачено на його виконання. Для «Skill» додано лічильники успіхів та невдач і показники впевненості та ймовірності опанування.

3.5. Опис модулів програми

Програму побудовано з використанням модульної архітектури, де кожен модуль може функціонувати незалежно від інших та має окрему зону відповідальності. Таким чином виокремлено 6 частин застосунку, а саме: модулі реєстрації, меню, навчання, тестування, дослідження та звітності.

Структура проєкту побудована відповідно до архітектурного патерну MVVM (Model-View-ViewModel) [28], ключовим принципом якого є чітке розмежування відповідальності між шарами обробки даних та представлення інтерфейсу. У межах кожного модуля виокремлено три компоненти: Models, що інкапсулюють основну бізнес-логіку застосунку; Views, відповідальні за формування та відображення користувацького інтерфейсу; та ViewModels, які виконують роль посередника між Views та відповідними Models, забезпечуючи їх взаємодію та синхронізацію даних.

Додатково розроблено окремі сервіси для керування сезонними змінами дизайну програми, роботи з базою даних, початкового завантаження контенту, роботи з інтелектуальними моделями, синтезом та розпізнаванням мовлення, аналізом тестових відповідей, визначенням поточного рівня знань та забезпеченням роботи адаптивного механізму. Також створено окремі перевикористовувані компоненти інтерфейсу для повторюваних елементів відображення тестів, опції керування звуковим супроводом, камери та полотна для малювання. Всі розроблені модулі відповідають створеному дизайну.

Система придатна для використання на різних моделях мобільних пристроїв компанії Apple.

3.5.1. Концепція та програмна реалізація навчального модуля

Навчальний модуль має два ключові напрямки – представлення наочних, аудіо й текстових матеріалів для запам’ятовування та функціональність для навчання письма.

Навчальні матеріали поділені за темами та представлені у форматі карток з ілюстративними матеріалами та текстовим описом. Дані для відображення у LearningView динамічно завантажуються з локальної бази даних за допомогою CoreDataMaterialsManager та передаються через MaterialsViewModel, відповідно до обраної теми. Користувач має можливість вільної навігації між картками. Навігація супроводжується анімацією перегортування, що забезпечує покращення користувацького досвіду та наближує схожість до взаємодії з фізичними навчальними матеріалами.

Для забезпечення функціональності навчання письма, в представлення матеріалів вбудовано описане раніше полотно для малювання – SkipFrame. Дане полотно відображається лише у випадку вибору опції «Письмо» та вбудоване в картки з наочностями, на яких відображено покрокові схеми написання чисел та літер.

Всі текстові дані в межах окремої навчальної одиниці озвучуються за допомогою SpeechManager, що працює на базі MicrosoftCognitiveServicesSpeech за наявності підключення до мережі та нативного Speech в разі його відсутності для забезпечення безперебійності роботи звукового супроводу. Додатково при першому відкритті опції навчання письма озвучуються голосові інструкції з описом алгоритму дій.

3.5.2. Концепція та програмна реалізація модуля тестування

Модуль тестування поділений на окремі теми, в межах яких тестування відбувається незалежно від результатів для інших тем. Для кожної теми передбачено відображення одинадцяти типів тестів, описаних раніше. Під час рендерингу окремого тесту програма опираючись на тип завдання динамічно визначає і відмальовує елементи інтерфейсу та опрацьовує специфічні голосові матеріали. Всі текстові умови завдань, варіанти відповіді та інструкції, як і випадку з навчальним модулем – озвучуються за допомогою SpeechManager.

3.5.2.1. Алгоритм визначення рівня знань дитини

Перед початком кожного тесту система динамічно обчислює поточний рівень знань дитини для навичок конкретного навчального модуля за допомогою класу LevelAnalyser, використовуючи формулу, наведену раніше (рисунок 3.10).

```
func calculateSkillResults(skill: Skill, latestTestActualTime: Double) -> (probability:Double, confidence:Double) {
    let beta = getBeta(for: skill.type, difficulty: skill.difficulty)
    let tTarget = getTargetTime(for: beta)

    let tActual = max(latestTestActualTime, 30.0)

    // Formula: L(t,m,d) = (γ * s + ρ * f) - β_test + ω * (T_target / T_actual)
    let successTerm = gamma * Double(skill.successes)
    let failureTerm = rho * Double(skill.failures)
    let confidence = omega * (tTarget / tActual)

    let logitL = (successTerm + failureTerm) - beta + confidence

    // Formula: p(m) = 1 / (1 + e^-L)
    let probability = 1.0 / (1.0 + exp(-logitL))

    return (probability, confidence)
}
```

Рисунок 3.10 – Програмний код роботи адаптивного алгоритму.

Значення константи ваги успіху (γ) становить 0,8, що дозволяє тримати алгоритм на стабільному рівні для цільової аудиторії та не допускає можливості поспішного переведення на новий рівень навичок. Для константи ваги невдачі (ρ) значення становить -0,5, що також сприяє стабільності ритму

навчання та не допускає різких знижень рівнів володіння навичками. Таким чином успіх більшою мірою впливає на роботу алгоритму, ніж невдача, що є оптимальним рішенням для дітей дошкільного віку. Вагу впевненості (ω) встановлено на рівні 0,3, що зумовлено специфікою цільової аудиторії, адже діти можуть відволікатися або демонструвати нерівномірну концентрацію уваги під час виконання завдань, що може спричинити часові аномалії у зібраних даних. Водночас це значення не є мінімальним, оскільки показники виконання все-таки є важливими під час обчислень. Значення ваги конкретного питання (β_{test}) обраховується за формулою для визначеної складності завдання, що знаходиться в межах від 1 до 5 та ваги типу цього завдання, що приймає значення від 0 до 1, залежно від формату та кількості варіантів відповідей. Для обрахунку оптимального часу відповіді на питання (T_{target}) визначено значення еталонного часу (T_{base}) – 30 секунд, що зумовлено специфікою цільової аудиторії та складністю тестових матеріалів.

В результаті роботи алгоритм повертає два значення – ймовірність опанування навички та впевненість при виконанні завдань, що обраховуються на основі раніше описаних формул (рисунок 3.11).

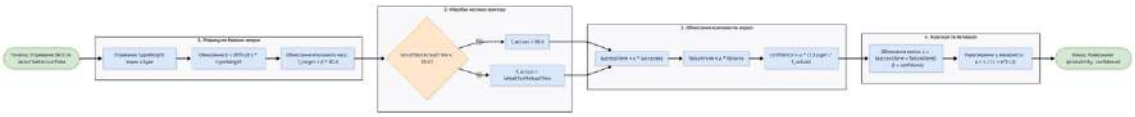


Рисунок 3.11 – Процес прийняття рішень алгоритму.

3.5.2.2. Алгоритм адаптивного вибору тестів

Відбір тестів працює на базі TestsViewModel та має 3 режими – початковий, адаптивний та випадковий, відповідно до поточної кількості пройдених користувачем тестів та сформованих навичок в межах теми.

Під час першого етапу система завантажує усі непройдені тести для даного блоку та вибирає з них 10 випадкових, щоб зібрати дані та визначити початковий рівень користувача.

Наступним етапом є основний – адаптивний режим. Під час нього система динамічно шукає навички для кожного тесту, аналізує їх, використовуючи попередньо описаний алгоритм, та вираховує ймовірність правильної відповіді на обраний тест. На наступному кроці вираховується відстань до оптимального показника, що становить абсолютне значення різниці константи 0,6 та попередньо визначеної ймовірності. Таким чином підтримується принцип "зони найближчого розвитку", суть якого полягає в тому, що оптимальними завданнями є ті, ймовірність виконання яких становить 60%, оскільки більший відсоток може бути занадто простим, а менший – складним. Система сортує обрані тести за принципом – спочатку ті, що найбільш наближені до оптимальної ймовірності. Фінальним кроком роботи алгоритму є вибір 10 найперших тестів, які пересортовують випадковим чином.

Останній режим активується після того, як дитина успішно засвоїла всі необхідні навички, і пропонує їй пройти тест із 10 випадково відібраних завдань із повного набору для обраної теми, з метою закріплення вивченого матеріалу.

3.5.2.3. Програмна реалізація процесу тестування

Кожна тестова сесія включає в себе виконання 10 тестів з конкретної теми. Процес проходження тестів поступовий, користувач може перейти до наступного завдання лише після виконання поточного, до попередніх завдань повертатись не можна. В момент появи умови на екрані – запускається таймер, який вираховує час виконання конкретного тесту.

Після виконання, відповідь надсилається на перевірку класом AnswerValidator, який виконує аналіз правильності на основі конкретного тесту, його типу та відповідей, даних користувачем, та повертає булеве значення (true або false) результату. Для відкритих питань процес перевірки має декілька етапів. На першому з них система перевіряє повну схожість відповіді користувача з еталонною, та в разі невдачі відбувається глибший аналіз. Надана відповідь нормалізується шляхом приведення усіх символів у нижній регістр, видалення розділових знаків та зайвих пробілів. Наступним кроком є переведення числових даних в буквенну форму. Якщо після даного кроку відповіді не збігаються – алгоритм розбиває відповіді на слова, які перетворює у їхні базові форми – леми та вираховує коефіцієнт схожості для кожної пари слів еталонної відповіді та відповіді користувача, використовуючи алгоритм відстані Левенштейна. Даний алгоритм рахує мінімальну кількість необхідних операцій для перетворення одного слова в інше. Якщо слова схожі хоча б на 80% – вони вважаються збігом. Якщо кількість схожих слів становить більше 80% речення – відповідь вважається правильною.

Після перевірки система записує час і статус виконання для конкретного тесту та оновлює значення кількості похибок та успіхів для навички, до якої належить тест. Додатково вираховується та записується в базу нова ймовірність опанування навички та впевненість під час вирішення пов’язаних з нею завдань.

На основі результату виконання система виводить повідомлення про успіх або невдачу, яке супроводжується анімацією та звуковим повідомленням. У разі неправильної відповіді також проводиться аналіз помилок, під час якого дитині відображаються правильні та неправильні відповіді та надається можливість прослухати пояснення помилки.

Після закінчення тестового блоку система виводить нагадування про відпочинок для запобігання перевантаження нервової системи дитини.

3.5.3. Концепція та програмна реалізація дослідницького модуля

Дослідницький модуль поділений на 3 теми – український алфавіт, цифри та геометричні фігури. Основою є доступ до камери пристрою, реалізований за допомогою AVFoundation та відображення результату класифікації.

Перед початком роботи проводиться звукова інструкція з використання. Користувач повинен навести камеру на відповідний об’єкт та натиснути на кнопку для розпізнавання. Система фіксує поточне зображення та, залежно від обраної теми, передає його на обробку у відповідний клас ImageClassifier, для нормалізації, шляхом перетворення у чорно-білий формат та обрізання. Нормалізоване зображення опрацьовується відповідною заздалегідь натренованою нейронною мережею через VNCoreMLRequest та VNImageRequestHandler (рисунок 3.12).

```
func classify(image: UIImage, completion: @escaping (String?) -> Void) {
    guard
        let bwImage = ImageEditor.makeBlackAndWhite(image),
        let cgImage = bwImage.cgImage
    else {
        completion(nil)
        return
    }

    let request = VNCoreMLRequest(model: model) { request, _ in
        let result = (request.results as? [VNClassificationObservation])?.first
        completion(result?.identifier)
    }

    request.imageCropAndScaleOption = .centerCrop

    let handler = VNImageRequestHandler(
        cgImage: cgImage,
        orientation: CGImagePropertyOrientation(image.imageOrientation),
        options: [:]
    )

    DispatchQueue.global(qos: .userInitiated).async {
        try? handler.perform([request])
    }
}
```

Рисунок 3.12 – Програмний код класифікації зображень.

Результатом виконання запиту є два значення – клас, до якого відноситься об’єкт, та показник впевненості моделі. Дані значення надсилаються до ClassificationStore, який є спостережуваним об’єктом з @Published

властивостями у відображенні дослідницького модуля – DiscoverView. Даний підхід базується на принципах реактивного програмування та використовує можливості фреймворку Combine, що дозволяє підписуватись на відстеження подій у системі та миттєво реагувати на них, змінюючи інтерфейс користувача.

Усі результати класифікації озвучуються. Для українського алфавіту озвучування доступне лише українською мовою, а для цифр та геометричних фігур – українською та англійською мовами.

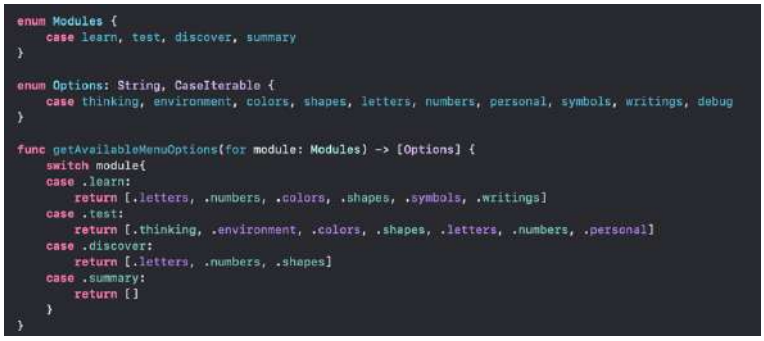
3.5.4. Концепція та програмна реалізація модуля звітності

Модуль звітності містить один статичний екран, який не передбачає жодної взаємодії з користувачем, окрім перегляду інформації. Ймовірність опанування навичок динамічно обраховується в процесі проходження відповідних тестів. За допомогою CoreDataSkillsManager вони оновлюються, завантажуються та передаються у SkillsViewModel, де відбувається фільтрація за статусом опанування. Ті навички, ймовірність виконання тестів для яких становить більше 95%, вважаються опанованими та відображаються на SkillsView.

3.5.5. Концепція та програмна реалізація модуля навігації

Модуль навігації складається з двох основних частин – головного меню, що слугує для навігації між окремими частинами застосунку, та допоміжних меню, які відповідають за навігацію між темами всередині цих частин. Вся навігація уніфікована та централізована. Створено перевикористовувані MainMenuView та ModulesMenuView, що відповідають за відображення опцій меню. MenuViewModel адаптує роботу модуля та керує навігацією у

застосунку, даний клас містить переліки доступних опцій навігації та визначає доступ до них залежно від частини програми (рисунок 3.13).



```
enum Modules {
    case learn, test, discover, summary
}

enum Options: String, CaseIterable {
    case thinking, environment, colors, shapes, letters, numbers, personal, symbols, writings, debug
}

func getAvailableMenuOptions(for module: Modules) -> [Options] {
    switch module{
    case .learn:
        return [.letters, .numbers, .colors, .shapes, .symbols, .writings]
    case .test:
        return [.thinking, .environment, .colors, .shapes, .letters, .numbers, .personal]
    case .discover:
        return [.letters, .numbers, .shapes]
    case .summary:
        return []
    }
}
```

Рисунок 3.13 – Програмний код керування опціями навігації.

Такий підхід гарантує можливість легкого розширення функціональності застосунку в майбутньому.

3.5.6. Концепція та програмна реалізація модуля реєстрації

Модуль реєстрації реалізовано у вигляді форми SignUpView, яка складається з двох послідовних етапів: збір інформації про склад родини та зазначення відомостей про місце проживання користувача. Дані кроки можна пропустити з можливістю повернутись до них пізніше. У разі відсутності збережених даних система пропонує пройти реєстрацію при кожному запуску програми або перед доступом до опції засвоєння персональної інформації.

Всі дані користувача використовуються виключно для навчання та безпечно зберігаються у Keychain для запобігання витоків чутливої інформації. Для збереження та обробки введених даних створено окремий клас UserDataManager, який використовує KeychainAccess для доступу до Keychain.

3.5.7. Керування виглядом програми

Для забезпечення узгодженості вигляду користувацького інтерфейсу розроблено окремий клас SeasonsManager, що відповідає за зміну вигляду елементів інтерфейсу залежно від пори року. Система динамічно вираховує поточний час та повертає відповідні ілюстрації та кольорові пари для відображення на інтерфейсі користувача.

Для забезпечення узгодженості дизайну та запобігання дублюванню коду створено перелічення ColorConstants, що містить кодування основних кольорів у моделі RGB. Щоб гарантувати адаптивність програми для різних розмірів дисплеїв створено перелік SizeConstants, де збережено розміри полотна представлення та тексту, які залежать від моделі пристрою, на якому запущено програму. Перелік TextConstants відповідає за керування основними та допоміжними шрифтами у програмі. Використання створених переліків забезпечує легкість корегування та узгодженість дизайну застосунку.

3.6. Впровадження системи

В результаті роботи розроблено повнофункціональну інтелектуальну систему підготовки дітей до школи на основі адаптивного навчання, що покриває розвиток базових компетенцій, які повинні бути сформованими для дітей дошкільного віку.

Коректність роботи системи верифіковано методами ручного функціонального тестування, проведено перевірку основних сценаріїв використання системи, які забезпечують навчальний процес. Виконана робота дозволила забезпечити справність інтерфейсу користувача та перевірити зручність взаємодії з системою.

З метою підтвердження відповідності вимогам та практичної значущості розробленого рішення проведено експертне оцінювання системи залученим методистом. Експертний аналіз підтвердив відповідність системи потребам освітнього процесу та перспективу її подальшого впровадження. Продукт рекомендовано до використання у закладах дошкільної освіти як додатковий цифровий інструмент підготовки до школи, а також для використання вдома самостійно та під супроводом батьків.

3.7. Висновки до Розділу 3

Для вибору оптимального адаптивного навчального алгоритму проаналізовано три поширені існуючі рішення: PFA, BKT та IRT. В ході аналізу виявлено, що жодне з них повною мірою не відповідає структурі розроблюваної системи у своєму класичному вигляді та, враховуючи математичну гнучкість PFA, його переваги над BKT та складність початкової калібрації IRT – прийнято рішення про адаптацію першого з них. Основні рішення щодо корегування обраного алгоритму включають: забезпечення відповідності модульній структурі програми та типам тестових матеріалів; врахування рівня впевненості користувача на основі часу виконання; та зміна поняття складності, враховуючи її варіацію в межах окремих питань конкретного модуля та типу.

Обґрунтовано вибір мови програмування Swift, фреймворків для створення користувацького інтерфейсу UIKit та SwiftUI, та середовища розробки XCode, враховуючи принципи швидкодії, безпеки та зручності розробки. Описано практичні переваги та прикладне використання нативних та сторонніх програмних фреймворків під час розробки. Висвітлено функціональне призначення авторського фреймворку SkipFrame в контексті розроблюваної системи.

Окреслено структуру внутрішньої бази даних. Описано архітектурні рішення та особливості системи. Висвітлено концепцію та програмну реалізацію усіх модулів програми, а саме: реєстрації, меню, навчання, тестування, дослідження та звітності. Розглянуто використання алгоритмів для визначення рівня знань користувача, адаптивного вибору тестових матеріалів, процесу тестування, аналізу та перевірки результатів виконання завдань. Окреслено процес роботи інтелектуальних моделей. Охарактеризовано механізм керування виглядом програми.

За результатами експертного рецензування та успішного тестування, систему визнано такою, що відповідає окресленим вимогам.

Висновки

У результаті виконаної роботи реалізовано комплексний програмний продукт – інтелектуальну систему підготовки дітей до школи на основі адаптивного навчання, що підвищує користь екранного часу дітей та підвищує ефективність навчання шляхом персоналізації освітньої траєкторії відповідно до індивідуальних особливостей та прогресу кожної дитини.

Результати дослідження підтвердили актуальність теми та необхідність створення освітнього програмного продукту для підготовки дітей до школи.

В процесі проєктування сформульовано чітку постановку задачі, визначено функціональні вимоги до розроблюваної системи, спроєктовано архітектуру продукту, розроблено базу навчального контенту, створено прототип інтерфейсу користувача з урахуванням специфіки цільової аудиторії. Створено набори тренувальних даних та проведено навчання класифікаційних нейронних мереж для класифікації букв, цифр та геометричних фігур з метою навчання шляхом дослідження оточення дитини.

Досліджено існуючі адаптивні алгоритми, та їхні математичні особливості, обрано найоптимальніше рішення в контексті розроблюваної системи. Впроваджено додаткові модифікації для підвищення його відповідності структурі програми та формату тестових матеріалів. Реалізовано повнофункціональний iOS-застосунок для інтелектуальної системи підготовки дітей до школи на основі адаптивного навчання. Впроваджено інтелектуальні складові для покращення користувацького досвіду, зокрема: розпізнавання об’єктів нейронними моделями,

розпізнавання та синтез мовлення. Забезпечено адаптацію роботи системи під потреби конкретного користувача. Охоплено формування ключових освітніх компетенцій для дітей дошкільного віку.

Верифіковано справність роботи системи та отримано рецензію від експерта у предметній області. За результатами експертного рецензування фахівцем-методистом, розроблену інтелектуальну систему визнано методично обґрунтованою та рекомендовано до використання у закладах дошкільної освіти та вдома як додатковий цифровий інструмент підготовки до школи.

Результати виконаної роботи відкривають перспективи подальшого розвитку системи – вдосконалення наявної функціональності, розширення баз навчальних матеріалів та додавання нових функцій відповідно до змін потреб та запитів користувачів.

Таким чином, поставлену мету досягнуто, а результати розробки мають високий потенціал практичного застосування з метою підготовки дітей до шкільного навчання. Також існує перспектива впровадження системи у закладах дошкільної освіти як додаткового навчального інструменту.

Список літератури

1. День безпечного Інтернету: Google презентує дослідження про онлайн-безпеку дітей в Україні [Електронний ресурс] // Офіційний Блог Google Україна. – 2025. – Режим доступу : https://ukraine.googleblog.com/2025/02/google.html
2. JAMA Network [Електронний ресурс] : [офіційний сайт]. – Режим доступу : https://jamanetwork.com
3. Social Media Use Trajectories and Cognitive Performance in Adolescents [Електронний ресурс] / [J. M. Nagata, J. H. Wong, K. E. Kim та ін.] // JAMA. – Режим доступу : https://jamanetwork.com/journals/jama/fullarticle/2839941?guestAccessKey=c8bce59a-f799-4c36-817e-dd2c05cf6ae4
4. Video Games and Aggressive Thoughts, Feelings, and Behavior in the Laboratory and in Life [Електронний ресурс] / Craig A. Anderson, Karen E. Dill // Journal of Personality and Social Psychology. – 2000. – Т. 78, № 4. – С. 772-790. – Режим доступу: https://www.apa.org/pubs/journals/releases/psp784772.pdf.
5. Мишеняткова абетка [Електронний ресурс] : мобільний застосунок / Thundermark Ukraine. – Режим доступу : https://apps.apple.com/ua/app/мишеняткова-абетка/id553447059?l=uk
6. НУМО: розвивальні ігри [Електронний ресурс] : мобільний застосунок / UNICEF Україна. – Режим доступу : https://apps.apple.com/ua/app/нумо-розвивальні-ігри/id1635803298?l=uk
7. Вивчаю – не чекаю [Електронний ресурс] : мобільний застосунок / War Child Holland. – Режим доступу : https://apps.apple.com/ua/app/вивчаю-не-чекаю/id1638116843?l=uk

8. Білан О. І. Програма розвитку дитини дошкільного віку «Українське дошкілля» / О. І. Білан ; за заг. ред. О. В. Низковської. – Тернопіль : «Мандрівець», 2017. – 256 с.
9. Про затвердження Положення про організацію освітнього процесу в закладах дошкільної освіти [Електронний ресурс] : Постанова Кабінету Міністрів України від 17 груд. 2024 р. № 1445 / Кабінет Міністрів України. – Режим доступу: https://zakon.rada.gov.ua/laws/show/1557-2025-%D0%BF#Text.
10. Extensible Markup Language (XML) 1.0 (Fifth Edition) [Електронний ресурс] : W3C Recommendation / World Wide Web Consortium (W3C). – 2008. – Режим доступу: https://www.w3.org/TR/xml/.
11. Common Format and MIME Type for Comma-Separated Values (CSV) Files [Електронний ресурс] : RFC 4180 / Y. Shafranovich; ; SolidMatrix Technologies, Inc. – 2005. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc4180.
12. Introducing JSON [Електронний ресурс] / JSON.org. – Режим доступу: https://www.json.org/json-en.html.
13. Вовк О. В. Вплив колірної гами навчальної літератури на сприйняття дитиною шкільного матеріалу [Електронний ресурс] / Вовк О. В., Чеботарьова І. Б., Шипова М. К. // Харківський національний університет радіоелектроніки. – Режим доступу: https://openarchive.nure.ua/server/api/core/bitstreams/55d597ed-2c97-478f-84e8-f978ce72ae6e/content.
14. Figma [Електронний ресурс]. – Режим доступу: https://www.figma.com/design/.

15. Ілюстрації до проєкту [Електронний ресурс] : згенеровано нейронною мережею Gemini / Google DeepMind. – 2026. – Режим доступу: https://gemini.google.com.
16. Python 3.14.3 documentation [Електронний ресурс] / Python Software Foundation. – 2026. – Режим доступу: https://docs.python.org/3/.
17. Pillow (PIL Fork) 12.1.1. documentation [Електронний ресурс] / Jeffrey A. Clark (Alex) and contributors. – 2026. – Режим доступу: https://pillow.readthedocs.io/en/stable/.
18. Create ML [Електронний ресурс] / Apple Developer ; Apple Inc. – 2026. – Режим доступу: https://developer.apple.com/machine-learning/create-ml/.
19. Xcode [Електронний ресурс] / Apple Developer ; Apple Inc. – 2026. – Режим доступу: https://developer.apple.com/xcode/.
20. Kerr P. Adaptive learning [Електронний ресурс] / Philip Kerr // ELT Journal. – 2016. – Т. 70, № 1. – С. 88-93. – Режим доступу: https://academic.oup.com/eltj/article-abstract/70/1/88/2450155.
21. Corbett A. T. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge [Електронний ресурс] / A. T. Corbett, J. R. Anderson // User Modeling and User-Adapted Interaction. – 1995. – Т. 4, № 4. – С. 253-278. – Режим доступу: https://act-r.psy.cmu.edu/wordpress/wp-content/uploads/2012/12/893CorbettAnderson1995.pdf.
22. Pavlik P. I. Performance Factors Analysis – A New Alternative to Knowledge Tracing [Електронний ресурс] / Pavlik P. I., Cen H., Koedinger K. R. – 2009. – С. 531-538. – Режим доступу: https://pact.cs.cmu.edu/koedinger/pubs/AIED%202009%20final%20Pavlik%20Cen%20Keodinger%20corrected.pdf.
23. Item Response Theory [Електронний ресурс] / Columbia Public Health; Columbia University Mailman School of Public Health. – Режим доступу:

https://www.publichealth.columbia.edu/research/population-health-methods/item-response-theory.

24. Apple Developer Documentation [Електронний ресурс] / Apple Inc. – 2026. – Режим доступу: https://developer.apple.com/documentation.
25. Kishikawa K. KeychainAccess [Електронний ресурс] : Simple Swift wrapper for Keychain that works on iOS, watchOS, tvOS and macOS / Kishikawa Katsumi. – 2026. – Режим доступу: https://github.com/kishikawakatsumi/keychainaccess.
26. About the Speech SDK [Електронний ресурс] / Microsoft Learn ; Microsoft Corp. – 2026. – Режим доступу: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-sdk.
27. Skip Y. SkipFrame [Електронний ресурс] : Swift framework for adding an interactive drawing overlay on top of any UIViewController / Yuliia Skip. – 2026. – Режим доступу: https://github.com/JuliaSkip/SkipFrame.git.
28. Parmar G. MVVM Architecture in iOS App [Електронний ресурс] / Gaurav Parmar // Medium. – 2024. – Режим доступу: https://medium.com/@gauravios/mvvm-architecture-in-ios-app-b03405e2ead4.