Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Застосування інтелектуальних агентів для адаптивного фонового вивчення мов
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-2

Вадим Свєркунов
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)

Керівник ст.викл. Тетяна Мірошниченко
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Свєркунову Вадиму Руслановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Застосування інтелектуальних агентів для адаптивного фонового вивчення мов

затверджена наказом університету від 19 травня 20 25 р. № 378Ст

2. Термін подання студентом роботи до екзаменаційної комісії 25 червня 20 25 р.

3. Вихідні дані до роботи Науково-технічні публікації та публічні дослідження стосовно засобів та алгоритмів для інтервального повторення, кривої забуття, моделювання пам'яті та навчання з підкріпленням. Аналіз існуючих програмних рішень для вивчення мов, таких як «Duolingo» та «Anki». Публічні набори даних, зокрема, датасет з додатка для вивчення мов MaiMemo. Технічна документація до програмних засобів та бібліотек, що використовуються: Python, .NET, SvelteKit, Apache Kafka, Docker, Entity Framework Core, DDPG, FSRS, та BKT. Дані з відкритих лінгвістичних сервісів, таких як Cambridge Dictionary та DeepL API.

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі та постановка задачі дослідження

2) Проектування системи адаптивного фонового вивчення мов з використанням інтелектуального агента

3) Програмна реалізація елементів системи адаптивного фонового вивчення мов

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	25.05.2025	виконано
3	Формулювання мети та задач дослідження	26.05.2025	виконано
4	Розробка загальної концепції системи	28.05.2025	виконано
5	Визначення вимог	29.05.2025	виконано
6	Проектування архітектури системи	02.06.2025	виконано
7	Проектування моделі інтелектуального агента	06.06.2025	виконано
8	Програмна реалізація ключових модулів системи	13.06.2025	виконано
9	Тестування розроблених компонентів та системи	13.06.2025	виконано
10	Написання пояснювальної записки	14.06.2025	виконано
11	Перевірка на академічний плагіат	17.06.2025	виконано
12	Підготовка презентації та доповіді	18.06.2025	виконано
13	Нормоконтроль	19.06.2025	виконано
14	Попередній захист	20.06.2025	виконано
15	Рецензування	21.06.2025	виконано
16	Захист перед ЕК	25.06.2025	

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст.викл. Тетяна Мірошниченко
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 125 с., 32 рис., 5 табл., 4 дод., 20 джерел.

ІНТЕЛЕКТУАЛЬНИЙ АГЕНТ, ІНТЕРВАЛЬНЕ ПОВТОРЕННЯ, КОМПОНЕНТИ ПАМ’ЯТІ, КРИВА ЗАБУТТЯ, МІКРОСЕРВІСНА АРХІТЕКТУРА, НАВЧАННЯ З ПІДКРІПЛЕННЯМ, BKT, DDPG, DOCKER, EF CORE, REPLAY BUFFER, KAFKA, MEMORY HALF-LIFE, MEMORY RETENTION, MEMORY STRENGTH, SM2.

Об’єкт дослідження – процес персоналізованого інтервального повторення для підвищення ефективності навчання іноземним мовам.

Предмет дослідження – модифікований з використанням сучасних засобів та алгоритмів для інтервального повторення та моделювання ймовірності знання, а також персоналізований під користувача інтелектуальний агент(и).

Мета роботи – створення адаптивної персоналізованої системи, яка поєднує в собі сучасні засоби та алгоритми у поєднанні з натренованими під користувача інтелектуальними агентами для підвищення ефективності та зручності вивчення іноземних мов.

Методи дослідження – аналіз наукової літератури та публічних досліджень стосовно засобів та алгоритмів для інтервального повторення, кривої забуття та навчання з підкріпленням.

ABSTRACT

Bachelor’s thesis contains: 125 p., 32 fig., 5 tab., 4 app., 20 sources.

BKT, DDPG, DOCKER, EF CORE, FORGETTING CURVE, INTELLIGENT AGENT, KAFKA, MEMORY COMPONENTS, MEMORY HALF-LIFE, MEMORY RETENTION, MEMORY STRENGTH, MICROSERVICE ARCHITECTURE, REINFORCEMENT LEARNING, REPLAY BUFFER, SM2, SPACED REPETITION.

The object of the research is the process of personalized spaced repetition to increase the efficiency of foreign language learning.

The subject of the research is an intelligent agent (or agents) modified with modern tools and algorithms for spaced repetition and knowledge probability modeling, personalized for the user.

The goal of the work is to create an adaptive personalized system that combines modern tools and algorithms with user-trained intelligent agents to enhance the efficiency and convenience of learning foreign languages.

Research methods – analysis of scientific literature and public research regarding tools and algorithms for spaced repetition, the forgetting curve, and reinforcement learning.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІА – інтелектуальний агент;

ІП – інтервальне повторення;

МН – машинне навчання;

ШІ – штучний інтелект;

BKT – Bayesian Knowledge Tracing – баєсове відстеження знань;

DDPG – Deep Deterministic Policy Gradient – глибинний детермінований градієнт політики;

DQL – Deep Q-Learning – глибинне Q-навчання;

EF Core – Entity Framework Core – технологія доступу до даних для .NET;

FSRS – Free Spaced Repetition Scheduler – вільний планувальник інтервальних повторень;

HLR – Half-Life Regression – регресія періоду напіврозпаду (пам'яті);

ITS – Intelligent Tutoring Systems – інтелектуальні навчальні системи;

LSTM – Long Short-Term Memory – довга короткочасна пам'ять (тип рекурентної нейронної мережі);

MVC – Model-View-Controller – Модель-Вид-Контролер;

.NET – платформа розробки програмного забезпечення від Microsoft;

NLP – Natural Language Processing – обробка природної мови;

RL – Reinforcement Learning – навчання з підкріпленням;

SM2 – алгоритм SuperMemo 2;

TS – TypeScript – Тайпскрипт;

UWP – Universal Windows Platform – універсальна платформа Windows;

WinRT – Windows Runtime – середовище виконання Windows.

ВСТУП

Навіть не беручи до уваги сучасні обставини в Україні та той факт, що велика кількість українців мігрували в інші країни, мов яких вони можуть не знати або не знати на достатньому рівні, по всьому світу зростає кількість людей, які прагнуть вивчати нові мови для роботи, подорожей або особистого розвитку, що створює високий попит на прості та ефективні інструменти для вивчення мов.

Навчання новій мові не є простим заняттям, але при досягненні хоча б проміжного рівня, воно відкриває багато можливостей для людини, навіть якщо це не стосується міграції або роботи – це надає доступ до значно більшої кількості інформації у тому ж самому інтернеті, конкретних джерел, які доступні тільки на відповідній мові або ж надає можливість не просто переглядати контент, а й саме зануритися у відповідне середовище та його засоби, ресурси та ком’юніті, що значно підвищує здібності до знаходження потрібної вам інформації та її використанню та розумінню.

Якщо ж торкатися саме міграційних чи робочих переваг, то володіння іноземною мовою не лише спрощує адаптацію в незнайомому середовищі, а й позитивно впливає на рівень життя людини, може надавати переваги у кар’єрі, наприклад, якщо посада, якимось чином пов’язана із комунікацією з клієнтами. Зокрема, не буде зайвим вказати, що деякі сфери праці, для прикладу візьмемо IT, вимагають певне знання мов, в цьому випадку англійської.

Таким чином, виникає потреба у достатньо простому, але ефективному засобі, який не буде поглинати забагато часу та занадто відривати людину від її життя чи праці, як це роблять сучасні засоби. Реалізація такого засобу і стає метою цієї роботи.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Визначення проблеми та актуальність фонового вивчення мов

Фонове вивчення мов – це інноваційний підхід, що інтегрує навчальний процес у повсякденне життя. Його мета – дозволити засвоєння мовного матеріалу паралельно з іншими видами діяльності (роботою, спортом, домашніми справами), мінімізуючи активне відволікання та потребу у виділенні окремого часу.

Актуальність цього підходу зумовлена двома ключовими факторами: стрімким зростанням потреби у володінні іноземними мовами та дефіцитом вільного часу або ж простого бажання на кількагодинні неперервні навчальні сесії з високим рівнем концентрації.

Потреба у вивченні іноземних мов полягає у:

а) глобалізації та кар'єрному розвитку:

– при міграції є необхідність швидкої адаптації до нового мовного середовища, що є особливо актуальним для багатьох українців;

– професійний розвиток, через те, що знання мов є критичним для кар'єрного зростання, особливо в глобалізованих сферах, як IT, де англійська мова є стандартом;

– розширення можливостей для бізнесу, подорожей та культурного обміну;

б) особистісному розвитку:

– доступ до інформації через можливість споживати контент (книги, фільми, наукові статті) мовою оригіналу;

– глибоке занурення в інші культури та спільноти, що також може бути корисним в усіх зазначених вище пунктах.

Проблема існуючих рішень полягає у тому, що незважаючи на високий попит, більшість існуючих рішень на ринку є неефективними. Програми або не мають повноцінного фонового режиму, або їхні фонові функції обмежуються простими сповіщеннями, і ще буде гарно, якщо ці сповіщення це щось окрім нагадування про те, щоб користувач зайшов у застосунок та щось зробив. Такий підхід не забезпечує належного засвоєння матеріалу.

Ключовий недолік – відсутність справжньої адаптивності. Існуючим додаткам бракує інтелектуальних алгоритмів, які б враховували індивідуальні особливості навчання, поточний стан та контекст (відповідно до 2.4) користувача (наприклад, чи зайнятий він, чи може приділити більше уваги).

Фонове вивчення мов – це ефективне рішення цієї проблеми через підхід ненав'язливого споживання інформації. Його ключова ідея – створити інструмент, який би не поглинав забагато часу та занадто не відривав людину від її життя чи праці, але в той же час враховував чи займається користувач роботою, їде в транспорті, відпочиває, чи перебуває в іншому середовищі, яке може впливати на його здатність сприймати та засвоювати інформацію і використовуючи подібні та інші моменти контексту або якісь деталі стосовно використовуваного пристрою намагатиметься інтегрувати навчальний процес у повсякденне життя, мінімізуючи відволікання від основної діяльності.

Таким чином, існує значний, але недостатньо задоволений ринковий попит на інструменти, що ефективно поєднують фонове навчання з глибокою персоналізацією.

Розробка системи, яка б інтелектуально адаптувалася до потреб та контексту користувача, є не тільки актуальною, але й має високий потенціал для задоволення потреб широкого кола користувачів, які прагнуть вивчати мови зручно та результативно.

1.2 Аналіз механізмів засвоєння інформації

1.2.1 Інтервальне повторення (ІП)

1.2.1.1 Крива забуття

Після того, як людина засвоює нову інформацію, з книг, інтернету чи інших джерел вона поступово починає її забувати. Цей процес можна відобразити через криву забуття (рисунок 1.1), яка показує як людська пам’ять зберігає знання, якщо не повторювати вивчений матеріал, спочатку забування відбувається дуже швидко, а згодом уповільнюється.

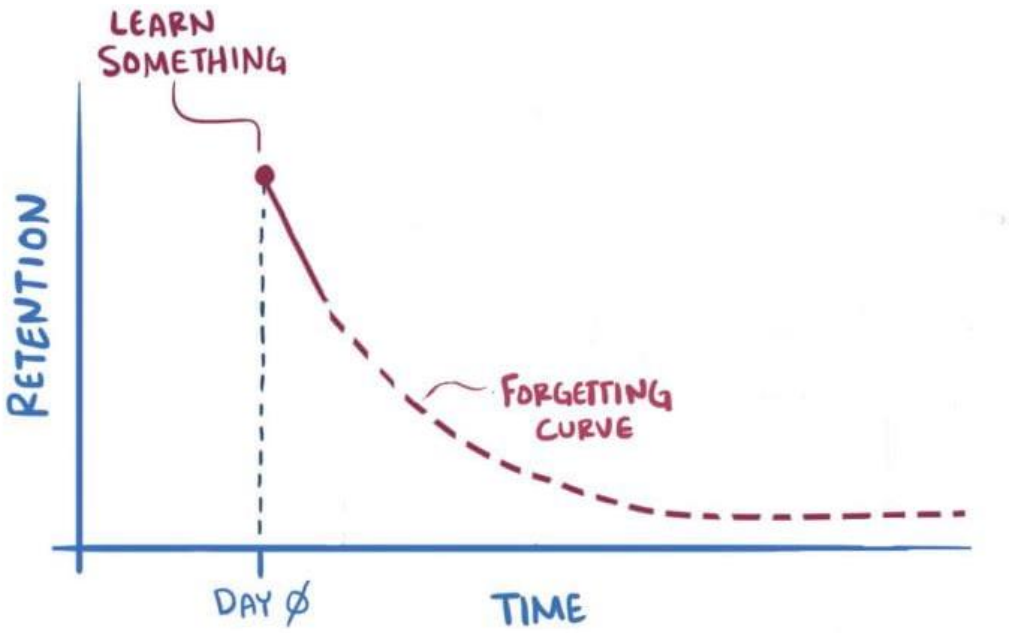

Рисунок 1.1 – Крива забуття, яка показує, як запам'ятовування знань погіршується з часом [1]

Періодичний перегляд вивчаємого матеріалу поступово вирівнює криву забуття (рисунок 1.2), іншими словами, він зменшує ймовірність забуття.

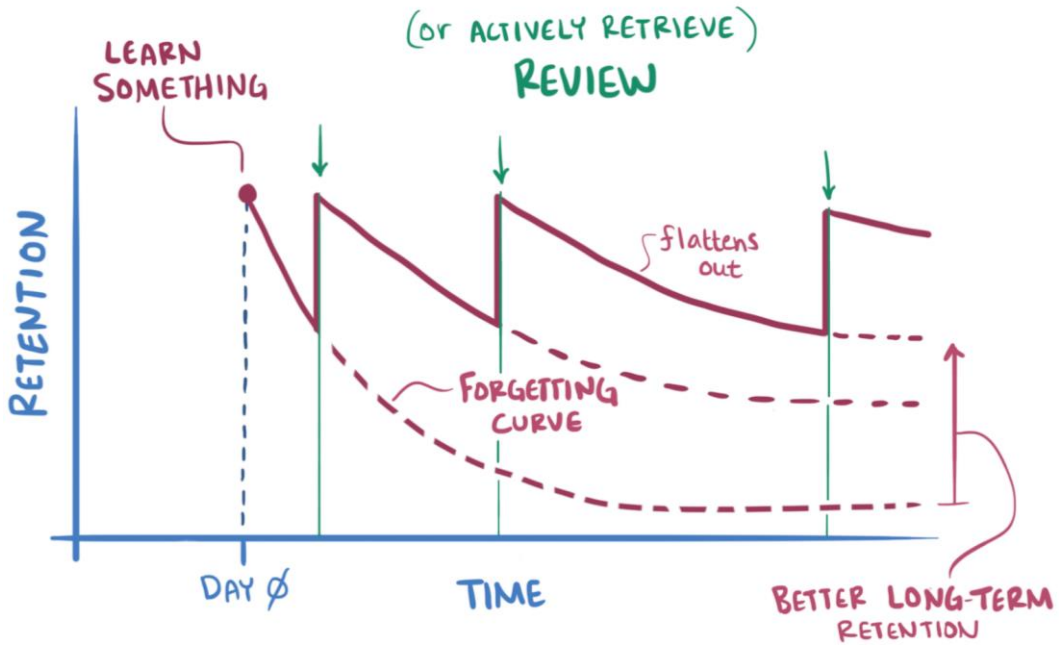

Рисунок 1.2 – Крива забуття при періодичному перегляді вивчаємого матеріалу [1]

Як можна побачити на рисунку 1.3, після кожного повторення, інтервал, який відповідає певному рівню запам'ятовування поступово збільшується. За цим принципом можливо виділити те що, коротші інтервали будуть кращими для незнайомого контенту, тоді як довші підійдуть для більш знайомого матеріалу. Відповідно цей принцип і називається інтервальним повторенням та використовується для довгострокового запам'ятання.

Його ефективність підтвердили декілька робіт, які можна знайти в джерелах [2], [3], також можна навести декілька цитат з цих джерел «A test immediately following the training showed superior performance for the distributed group (70% correct) compared to the massed group (53% correct). These results seem to show that the spacing effect applies to school-age children and to at least some types of materials that are typically taught in school. [2]», та The overall mean weighted effect size was 0.46, with a 95% confidence interval that extended from 0.42 to 0.50. ...the 95% confidence interval for this effect size does not contain zero, indicating that spaced practice was significantly superior

to massed practice in terms of task performance. [3]» якщо коротко, то результати досліджень виявили підвищення ефективності приблизно на ~63% у порівнянні із людьми, які просто використовували звичайне повторення, але у великому обсязі.

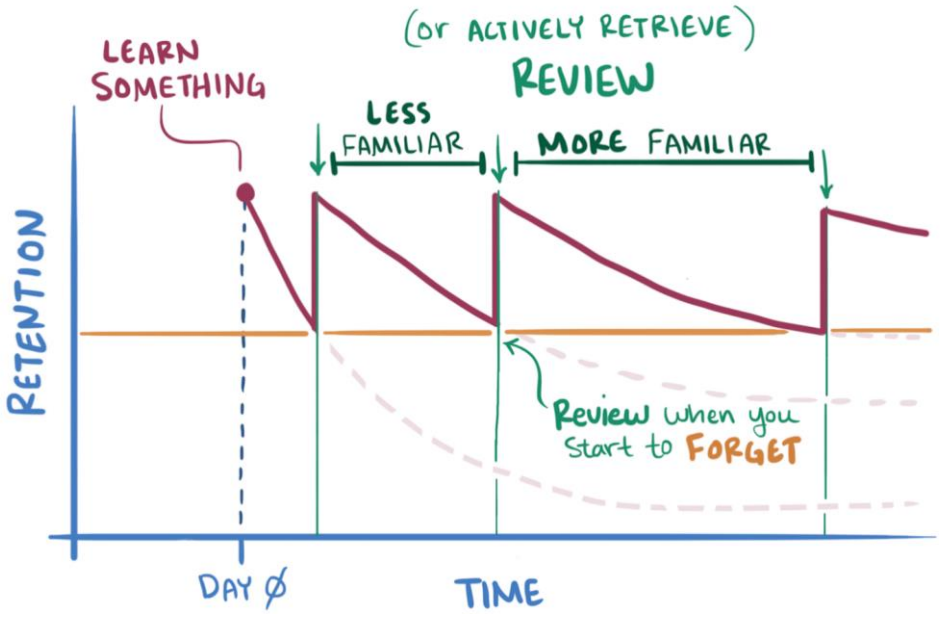

Рисунок 1.3 – Крива забуття після кожного повторення [1]

1.2.1.2 Компоненти пам’яті

По перше, для того, щоб показати наскільки добре людина щось запам'ятала використовують таке поняття як «memory strength».

Через те, що ніхто не може однозначно стверджувати, що слово, запам'ятане сьогодні, буде згадане через тиждень або забуте через місяць. Потрібно вводити ключову змінну, що характеризує стан пам'яті людини, а саме збереження пам'яті, або ж «memory retention (recall probability)».

Наступні досліди в цій області були проведені з використанням датасету у відкритому доступі [4], створеному завдяки додатку для вивчення мов MaiMemo, а також наступним науковим працям [5], [6].

Результати представлені на рисунку 1.4.

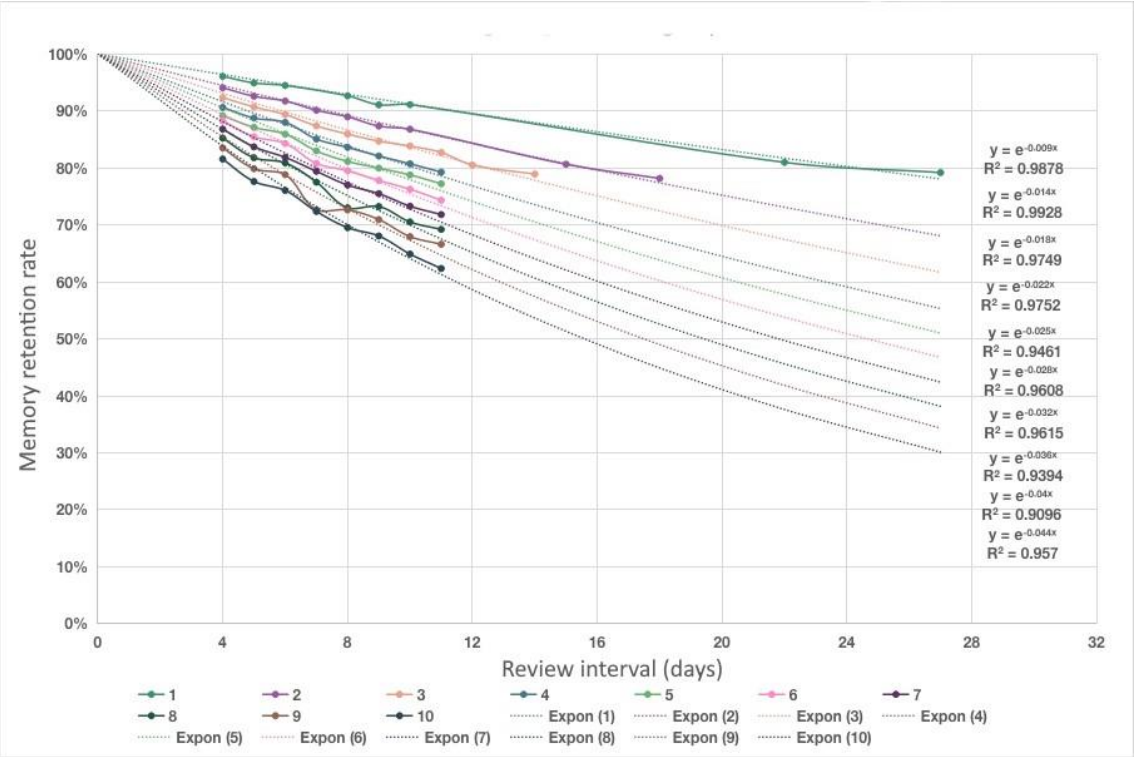

Рисунок 1.4 – Крива забуття на десяти групах з різними рівнями складності [1]

Цей графік можна апроксимувати завдяки від'ємній експоненціальній функції згідно з формулою (1.1) для кривої забуття. А швидкість забуття можна охарактеризувати константою «decay» для цієї функції.

$$R = \exp\left[\frac{t \ln 0.9}{S}\right], \tag{1.1}$$

де R – збереження пам'яті;

S – позначає стабільність пам'яті, або ж «memory stability (memory strength)»;

t – позначає час, що минув з моменту останнього перегляду.

Стабільність пам'яті (S), визначається як час, необхідний для того, щоб ймовірність пригадування (R), впала зі 100% до 90% (рисунок 1.5). У науковій літературі часто використовується значення 50% і в такому випадку вживається термін «memory half-life».

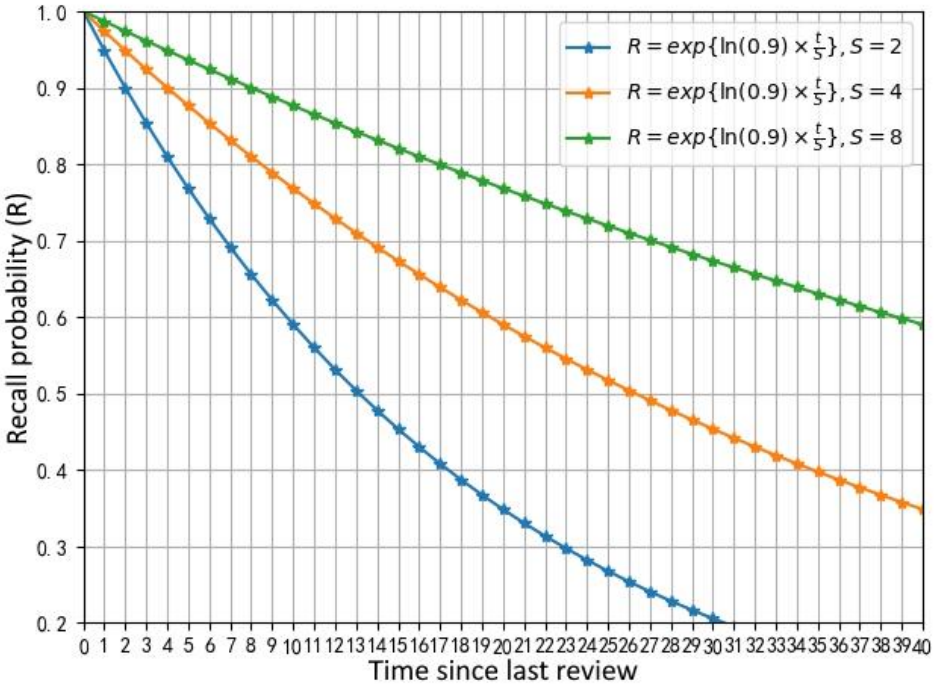

Рисунок 1.5 – Апроксимізована крива забуття, за різними S (стабільність пам'яті) [1]

Два компоненти пам'яті, запропоновані Бйорком – це сила пошуку «retrieval strength» та сила зберігання «storage strength», які відповідають попередньо визначеним ймовірності пригадування та стабільності пам'яті. Але у цьому підході бракує ще однієї компоненти. Проблема у тому, що інформація яку людина намагаєтеся запам'ятати, сама по собі має властивість, яка може впливати на її пам'ять, ця змінна – це складність «difficulty» цієї інформації. Таким чином, було отримано три компоненти пам’яті:

– стабільність пам'яті, вона ж «stability» (час, необхідний для того, щоб ймовірність пригадування певної інформації знизився зі 100% до 90%);

– ймовірність пригадування, вона ж «retrievability (probability of recall)» (ймовірність згадати певний шматок інформації у певний момент);

– складність (невід'ємна складність, пов'язана з конкретним шматком інформації).

Різницю між ймовірністю пригадування та збереженням пам'яті можно виразити як те, що перше стосується ймовірності пригадування конкретного шматка інформації, коли друге стосується середньої ймовірності пригадування для сукупності таких шматків. Тепер можна визначити ймовірність пригадування будь-якого шматка інформації за часом (t), після «n» успішних пригадувань завдяки формулі (1.2):

$$R_n(t) = \exp\left[\frac{t \ln 0.9}{S_n}\right]. \quad (1.2).$$

Якщо ж S_n використовується як інтервал між переглядами – це рівняння може подолати розрив між алгоритмами інтервального повторення та моделями пам'яті використовуючи формулу (1.3):

$$R_n(t) = \exp\left[\frac{t \ln 0.9}{I_1 \prod_{i=2}^{n} C_i}\right], \quad (1.3).$$

де I_1 – початковий інтервал після першого перегляду;

C_i – це відношення і-го інтервалу та попереднього $i - 1$ інтервалу.

Таким чином, головною метою алгоритмів для інтервального повторення стає точне обчислення I_1 та C_i , тим самим визначаючи стабільність пам'яті для різних учнів, шматків інформації та графіків повторення.

1.2.2 Огляд основних алгоритмів для інтервального повторення та перевірка їх ефективності

З моменту створення першого алгоритму для інтервального повторення, яким став SM0 ще у 1985 році [7], [8], який був створений ще без використання комп'ютера, для його реалізації були зібрані

експериментальні дані для визначення найкращих інтервалів перегляду для конкретної особи та конкретного типу матеріалу.

Але так як ручні розрахунки не є ефективними через те, що кожна одиниця інформації має свою криву забуття з’явилася потреба у наступній оцифрованій версії алгоритму SM2 [8], [9], яка була створена двома роками потому. Він являв собою алгоритм для комп'ютерних додатків, що вводить більш деталізований рівень карток і включає адаптивні коефіцієнти «ease factors» та оцінки. Пройшло вже багато часу, але треба зазначити, що SM2 досі використовується у багатьох застосунках для інтервального повторення через простоту його реалізації та має декілька доопрацьованих версій, які надають кращі результати.

З моменту створення першого подібного алгоритму минуло вже більше тридцяти років тож, звісно, за такий проміжок часу була представлена і досить велика кількість інших, більш точних, алгоритмів, для оцінки їх ефективності будуть використані вже існуючі порівняння усіх основних, існуючих алгоритмів у відкритому доступі [10].

Спочатку подивимося на статистичні дані основних алгоритмів у таблиці 1.1, де для тесту використовується дата сет з 9999 користувачів та 349,923,850 їх відгуків, а у стовпчику «Parameters» показано кількість параметрів, що оптимізуються (тренуються). Якщо параметр є константою, він не враховується. Стрілки вказують на те, чи нижчі (↓) або вищі (↑) значення є кращими.

Також, для більшої читабельності, було скорочено стовпець «Input», а саме:

– IL = interval lengths, in days;

– FIL = fractional (aka non-integer) interval lengths;

– G = grades (Again/Hard/Good/Easy);

– SR = same-day (or short-term) reviews;

– AT = answer time (duration of the review), in milliseconds.

Таблиця 1.1 – Порівняння основних алгоритмів для інтервального повторення

Algorithm	Parameters	Log Loss↓	RMSE (bins)↓	AUC↑	Input
LSTM	8869	0.3115±0.0079	0.0354±0.0011	0.7332±0.0038	FIL, G, SR, AT
GRU-P-short	297	0.3195±0.0080	0.0421±0.0013	0.7096±0.0047	IL, G, SR
FSRS-6 recency	21	0.3198±0.0080	0.0437±0.0013	0.7096±0.0041	IL, G, SR
FSRS-rs	21	0.3198±0.0083	0.0437±0.0012	0.7095±0.0040	IL, G, SR
FSRS-6	21	0.3215±0.0082	0.0464±0.0013	0.7060±0.0041	IL, G, SR
FSRS-5	19	0.3273±0.0082	0.0518±0.0016	0.7025±0.0041	IL, G, SR
FSRS-4.5	17	0.3324±0.0084	0.0536±0.0016	0.6918±0.0041	IL, G
FSRS v4	17	0.3378±0.0086	0.0582±0.0017	0.6891±0.0043	IL, G
DASH	9	0.3398±0.0083	0.0627±0.0016	0.6386±0.0047	IL, G
NN-17	39	0.384±0.029	0.0810±0.0037	0.6112±0.0045	IL, G
Ebisu v2	0	0.457±0.012	0.1582±0.0039	0.5942±0.0050	IL, G
Anki-SM-2	0	0.490±0.015	0.1278±0.0036	0.5973±0.0054	IL, G
SM-2	0	0.547±0.017	0.1484±0.0042	0.6005±0.0051	IL, G

З цих даних можна побачити, що LSTM, GRU-P-short, FSRS-6 recency, FSRS-rs є найкращими варіантами, але також можна зазначити, що перші два алгоритми використовують велику кількість параметрів, що значно підвищить затрати на розрахунок, водночас останні версії FSRS алгоритмів показують гарний результат при доволі низької кількості параметрів. Знизу

таблиці також можна побачити результати для вже зазначеного вище SM2 та його покращеної версії від одного з найпопулярніших застосунків для інтервальних повторень, а саме Anki.

Далі можна подивитися на порівняння цих алгоритмів за відсотком на 9999 колекціях відповідей користувачів (рисунок 1.6), де алгоритм A (рядок) перевершує алгоритм B (стовпець). Тобто, наприклад, алгоритм FSRS-5 перевершив FSRS-4.5 на 83.4%.

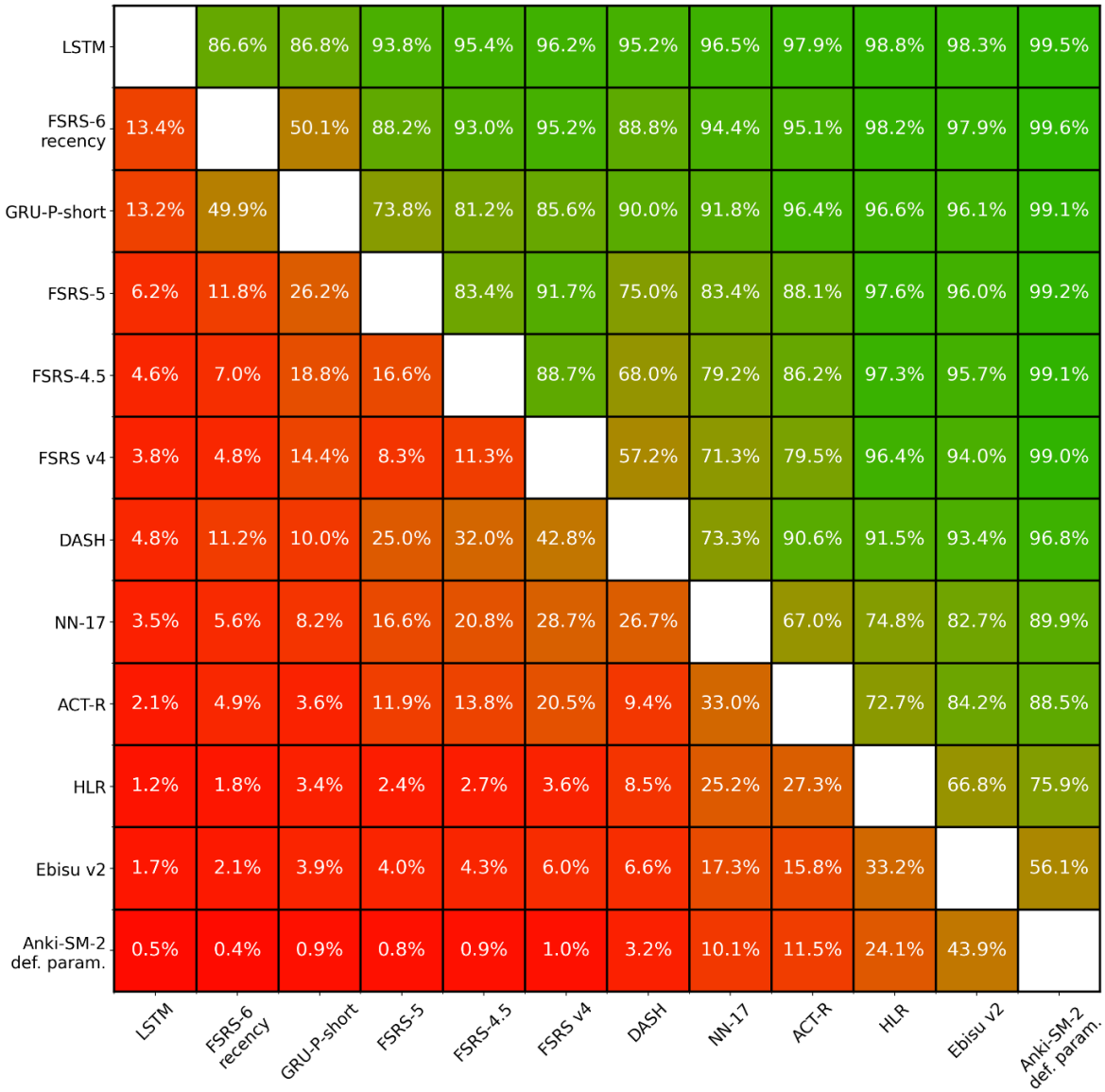

Рисунок 1.6 – Відсоток колекцій (базується на 9 999 колекціях), де алгоритм A (рядок) перевершує алгоритм B (стовпець). Включено лише основні алгоритми

1.3 Перспективи застосування інтелектуальних агентів

1.3.1 Загальний опис інтелектуальних агентів

Взагалі-то, інтелектуальні агенти призначені для того, щоб автономно діяти в заданому середовищі для досягнення якихось конкретних цілей.

Для них можна виділити наступні ключові характеристики:

– дії без прямої інструкції;

– можливість навчатися на досвіді використовуючи «Experience Replay»;

– відповідь на зміни середовища;

– ініціативність у досягненні цілей.

Існують як прості агенти, для яких встановлюються конкретні прості правила із жорстко закодованими значеннями параметрів, так і складніші типи агентів по типу:

– агентів з поведінкою, заснованою на внутрішній моделі середовища для прийняття рішень у умовах неповної інформації;

– цілеспрямованих та практичних агентів, які оцінюють стани через функції корисності, вона ж «reward function»;

– навчуваних агентів (напр., на основі нейромереж): які здатні до самооптимізації.

Агентів нерідко використовують для створення роботизованих систем, фінансових систем (наприклад, для виявлення шахрайства), медичних діагностичних систем (для задач по типу аналізу зображень МРТ), логістики (для маршрутизації автономних транспортних засобів), а також задач по типу вдачної посадки зонда на Марс.

Загалом, інтелектуальні агенти здатні вирішувати складні завдання, що вимагають адаптації до змінюваних умов середовища та оптимізації своїх дій для досягнення поставлених цілей.

1.3.1.1 Навчання з підкріпленням (Reinforcement Learning, далі RL)

Будь-який з алгоритмів побудованих завдяки навчанню з підкріпленням має основні концепції, яких треба притримуватися при реалізації.

По перше, агент буде взаємодіяти із середовищем, отримуючи спостереження «state», винагороду «reward» та виконуючи дії «action». Далі завдяки функції винагороди оцінює свою дію в певному стані (наприклад, у робототехніці – винагорода за точність руху). Ці дії агент обирає завдяки заданій для нього політиці «policy», яка фактично є стратегією агента для вибору дій.

Для прикладу подивимося на діаграму [11] на рисунку 1.7, що показує агента, який взаємодіє з навколишнім середовищем, використовуючи політику, яка оновлюється алгоритмом навчання з підкріпленням. Показано дії та спостереження за час t, а також винагороди та спостереження за час t+1.

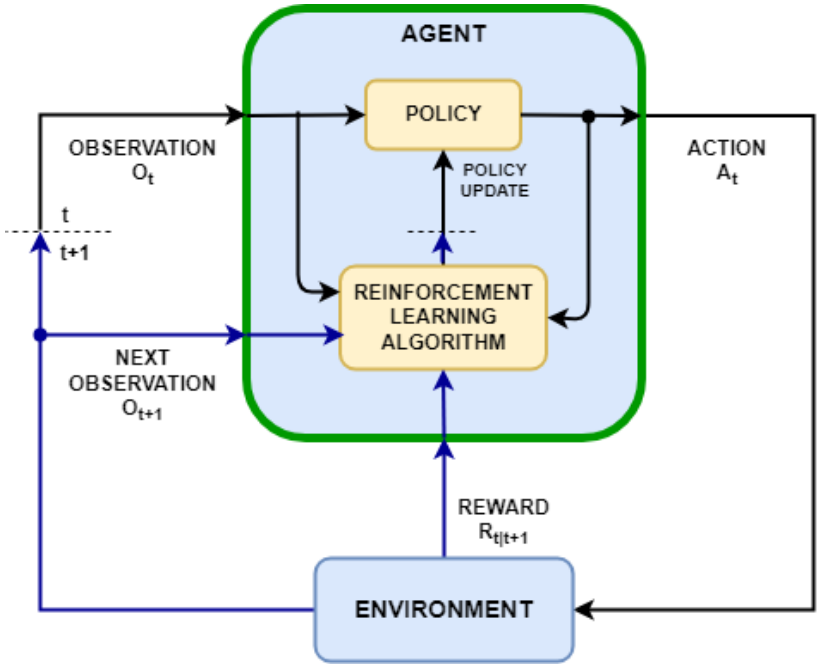

Рисунок 1.7 – Діаграма взємодії агенту із середовищем

Також одним з головних моментів при обранні агенту стає його вміння працювати з дискретними або ж неперервними станами та діями. Наприклад, дискретними діями буде скінченна кількість варіантів (вибір між кнопками «вліво/вправо» у грі), а неперервними – нескінченний простір значень (кут повороту роботизованої руки).

Для наглядності та демонстрації переваг та недоліків існуючих інтелектуальних агентів [11] предсталені рисунки 1.8, 1.9.

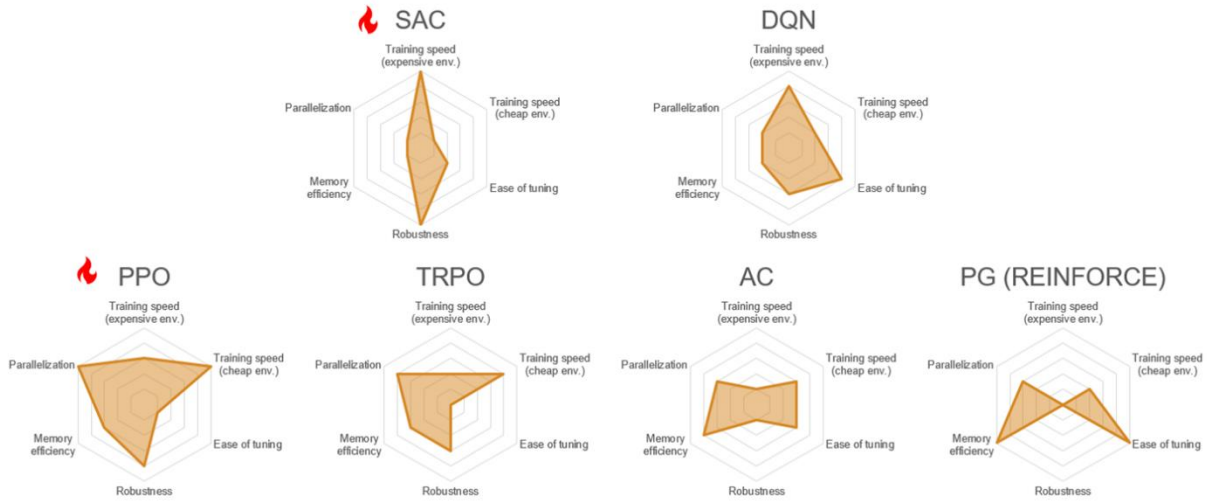

Рисунок 1.8 – Агенти які працюють з дискретним простором дій

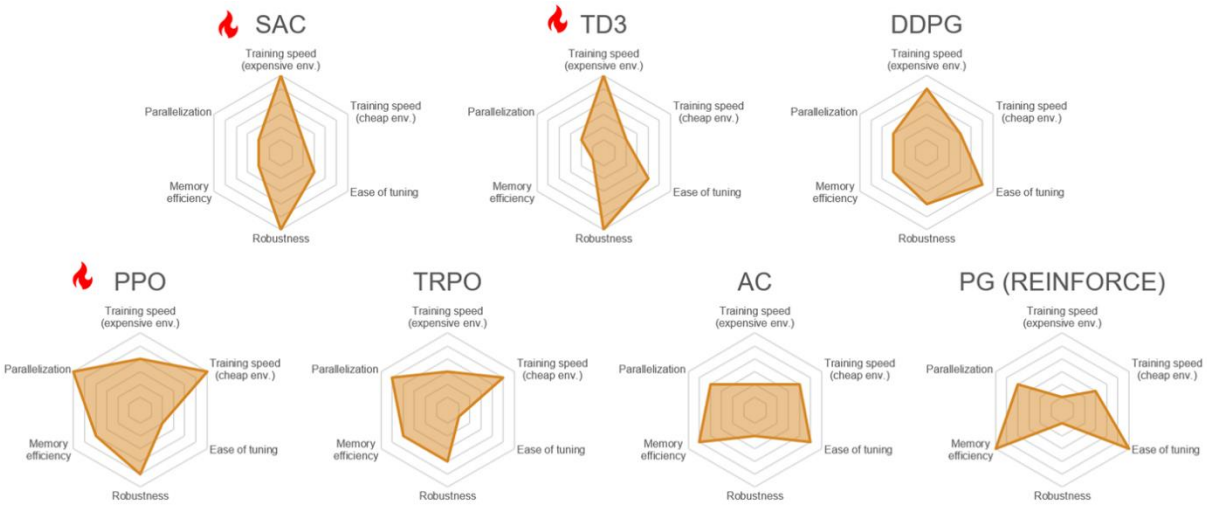

Рисунок 1.9 – Агенти, які працюють з неперервним простором дій

1.3.1.2 Глибоке навчання з підкріпленням (Deep RL)

Основний момент за яким відрізняються Deep RL від RL це використання глибоких нейронних мереж для апроксимації, що дозволяє працювати з високорозмірними даними (наприклад, зображення, сенсори робота) та неперервними просторами дій.

Таким чином, Deep RL також включає механізми стабілізації навчання такі як «Replay buffer», являє собою сховище (буфер) досвіду агента, де зберігаються попередні переходи, які вже були зазначені раніше: стан (state), дія (action), винагорода (reward), а також новий (наступний) стан (next state). Та «Target network», яка є допоміжною нейромережею, яка також є копією основної мережі (наприклад, критика «Critic network» в DDPG [12], [13]), але її ваги оновлюються з певним запізненням.

Оскільки було згадано поняття «Critic network», треба пояснити, що воно із себе уявляє. Але, по перше, треба ще представити поняття Q-Значення «Q-Value» або просто «Q». Це значення буде мати різні імплементації залежно від агента, але, взагалі-то, сенс у тому, що він визначає оптимальність дій у довгостроковій перспективі. Тобто, у контексті простої гри, Q-значення може визначати, чи призведе дія, для прикладу візьмемо «рух вгору», до перемоги в раунді. Таким чином, Q-значення є основою для оцінки стратегій для багатьох агентів.

Повертаючись до Critic network – це нейромережа, яка буде оцінювати якість дій на основі Q-значень. Але, для того, щоб було, що оцінювати також потрібно додати і актора «Actor network», який в свою чергу визначає політику (стратегію дій агента).

Таким чином, створюється система, де актор оптимізує дії для максимізації винагороди, а критик підказує, які з цих дій ефективні. Replay buffer – зберігає досвід для стабільного навчання. А також у випадку DDPG, через те, що його основна ціль це мінімізування різниці між Q та Q_{next}, Target network як раз і існує для цієї мінімізації.

1.3.2 Застосування інтелектуальних агентів у вивченні мов

Інтелектуальні агенти (ІА) є ключовим компонентом для створення описаної у цій робиті системи вивчення мов, завдяки ним можливо досягити глибокої персоналізації та ефективності навчального процесу.

Тож ключовими напрямками та завданнями, які потрібно вирішити стають:

а) оптимізація процесу запам'ятовування досягається завдяки:

– ІА, які розраховують оптимальні інтервали для повторення матеріалу індивідуально для кожного користувача, виходячи за рамки стандартних алгоритмів описаних у 1.2.2, також є варіант використовувати вже існуючі алгоритми для ІП, щоб задати напрямок тренування ІА, тим самим покращивши ефективність на початкових етапах та частково позбавивши від проблеми «холодного старту»;

– завдяки навчанню з підкріпленням (Reinforcement Learning, RL) та агентам, навченим, безпосередньо, за допомогою RL (наприклад, DDPG), можливо ще більше покращіти ІА, надавши їм можливості донавчання у реальному часі під час взаємодії з користувачем;

б) глибока персоналізація навчання досягається коли:

– на даних усіх користувачів тренується потужна глобальна модель, яка слугує фундаментом, який володіє загальними знаннями про процес навчання. Коли новий користувач реєструється, він починає взаємодіяти з цією глобальною моделлю. Після того, як він накопичує достатню кількість даних (наприклад, 100-200 взаємодій), система робить для нього копію глобальної моделі і донавчає (fine-tunes) її тільки на його персональних даних, такий підхід також позбавляє від проблеми «холодного старту», коли для нових користувачів, про яких ще немає даних, моделі ІА повинні починати з евристичних підходів або налаштувань за замовчуванням, поступово

збираючи інформацію та переходячи до повністю персоналізованої стратегії;

в) відстеження знань:

– моделі знань (наприклад, Bayesian Knowledge Tracing – BKT): ІА використовують BKT для отримання ймовірнісної оцінки того, наскільки добре користувач засвоїв матеріал. Це дозволяє приймати обґрунтовані рішення щодо наступних завдань;

г) інтелектуальні навчальні системи (Intelligent Tutoring Systems, ITS):

– ІА діагностує рівень знань учня в реальному часі та надає індивідуалізовану підтримку (підказки, зворотний зв'язок), керуючи навантаженням;

– застосування у фоновому режимі;

– такий ІА також приймає рішення щодо типу, складності та способу подачі контенту, враховуючи поточний контекст (див. 2.4) користувача (чи він зайнятий, чи вільний). Наприклад, агент може надіслати коротке текстове сповіщення або запропонувати інтерактивне завдання.

Таким чином, інтелектуальні агенти дозволяють створювати по-справжньому адаптивні системи вивчення мов, здатні ефективно функціонувати навіть у фоновому режимі, підлаштовуючись під динамічне життя сучасної людини.

1.4 Аналіз існуючих рішень

Ринок застосунків для вивчення мов є доволі насиченим, проте аналіз типових представників виявляє суттєві обмеження, що обґрунтовує необхідність розробки нової, більш досконалої системи. Серед популярних застосунків можна виділити такі платформи, як «Duolingo» та «Anki».

У той час коли «Duolingo» відомий своїм ігровим підходом та широким охопленням мов, «Anki» є потужним інструментом для

інтервального повторення, що базується на картках і дозволяє користувачам створювати власний навчальний контент або використовувати готові набори. Хоча повний огляд усіх існуючих рішень виходить за межі даної роботи, аналіз типових представників дозволяє виявити загальні тенденції та недоліки.

Серед таких недоліків можна особливо виділити:

– часто адаптивність є поверхневою і зводиться до зміни складності завдань або використання застарілих алгоритмів інтервального повторення (як SM2 чи інші алгоритми для ІП вказані у 1.2.2);

– більшість масових застосунків не використовують передові моделі пам'яті (напр., FSRS) і не враховують комплексно індивідуальні когнітивні особливості, поточний стан чи контекст користувача;

– можливості фонового режиму в існуючих застосунках часто є дуже обмеженими. У кращому випадку, це зводиться до періодичного надсилання сповіщень із нагадуванням про необхідність повторити матеріал або з окремими словами для запам'ятовування. Такий підхід є лише поверхневим втіленням ідеї фонового навчання і не забезпечує глибокої інтеграції навчального процесу в повсякденну діяльність користувача. Відсутнє врахування реальних можливостей користувача взаємодіяти з навчальним контентом у різні моменти часу, а також адаптація типу та обсягу інформації до поточного контексту;

– відсутня глибока інтеграція в повсякденну діяльність та адаптація типу контенту до реальних можливостей користувача взаємодіяти з ним;

– критичним обмеженням є відсутність ІА, що використовують навчання з підкріпленням (RL) для динамічної оптимізації довгострокових навчальних стратегій. Системи покладаються на жорстко закодовані правила;

– «холодний старт», коли нові користувачі отримують стандартизований контент через відсутність даних для персоналізації, що знижує мотивацію;

– недостатня інтеграція моделей знань (по типу ВКТ), що унеможливлює точне відстеження прогресу та оптимальний вибір завдань.

Тож проведений аналіз свідчить про наявність значного, але недостатньо задоволеного ринкового попиту на інструменти, що ефективно поєднують можливості фонового навчання з глибокою персоналізацією, керованою справжніми інтелектуальними агентами. Існує чітка ніша для розробки системи, де ІА є центральним елементом, що керує навчальним процесом на основі глибокого розуміння користувача, його контексту та сучасних принципів навчання (включаючи передові алгоритми інтервального повторення та моделі відстеження знань).

Ключова інновація полягатиме не просто у створенні ще одного застосунку для вивчення мов, а в розробці цілісної системи, здатної забезпечити високоефективне та зручне засвоєння мовного матеріалу в адаптивному фоновому режимі.

1.5 Постановка задачі

На підставі проведеного в попередніх підрозділах аналізу проблемної області, що охоплював дослідження механізмів засвоєння інформації (див. 1.2), огляд сучасних алгоритмів інтервального повторення (див. 1.3), оцінку перспектив застосування інтелектуальних агентів та аналіз наявних на ринку рішень (див. 1.4), було визначено, що проблема адаптивного фонового вивчення іноземних мов є актуальною через те, що велика частина ефективності алгоритмів для інтервального повторення залежить саме від адаптивності під користувача та його здібності.

Зростаюча важливість володіння іноземними мовами в сучасному глобалізованому світі зумовлена як об'єктивними професійними потребами, такими як міжнародна міграція, розширення ділових контактів, кар'єрне зростання, особливо в динамічній ІТ-сфері, так і прагненням до

особистісного розвитку та отримання доступу до значно ширшого кола інформаційних ресурсів та культурних надбань. Водночас традиційні методи вивчення мов часто характеризуються високою ресурсомісткістю, вимагаючи значних часових затрат та активної концентрації уваги, що стає суттєвою перешкодою для людей із щільним графіком роботи та повсякденних справ. У цьому контексті концепція фонового вивчення мов виглядає як перспективний напрямок, що потенційно дозволяє інтегрувати навчальний процес у повсякденне життя, мінімізуючи відволікання від основної діяльності та створюючи умови для більш органічного засвоєння матеріалу.

Актуальність такого підходу посилюється не лише дефіцитом вільного часу у сучасних користувачів, але й зміною загальної парадигми споживання інформації: люди звикли до багатозадачності та так званого «ненав'язливого» отримання контенту через різноманітні цифрові канали. Перенесення цього принципу на освітній процес є логічним кроком, проте його ефективність критично залежить від здатності системи інтелектуально адаптуватися до потреб користувача, а не просто пасивно транслювати навчальний матеріал. Існує значний, але недостатньо задоволений ринковий попит на інструменти, що ефективно поєднують можливості фонового навчання з глибокою персоналізацією.

При цьому існуючі методи розв'язання цієї проблеми та відповідні програмні застосунки, попри їхню численність та популярність (наприклад, «Duolingo» та «Anki», аналіз яких проведено в розділі 1.4), мають декілька суттєвих обмежень та недоліків, особливо в контексті реалізації по-справжньому адаптивного фонового навчання. Одним із ключових недоліків є недостатня глибина адаптації навчального процесу. Багато систем покладаються на спрощені алгоритми інтервального повторення, такі як класичний SM2, який, хоч і довів свою корисність, значно поступається за точністю та гнучкістю новітнім розробкам по типу FSRS. Такі системи часто не враховують комплексно індивідуальні когнітивні

особливості користувача, його поточний емоційний стан, рівень втоми чи навіть контекст (відповідно до 2.4), у якому відбувається навчання. Можливості фонового режиму в існуючих застосунках часто зводяться до, максимум, періодичного надсилання сповіщень, що є лише поверхневим втіленням ідеї і не забезпечує глибокої інтеграції в повсякденну діяльність користувача та врахування його реальних можливостей взаємодіяти з навчальним контентом у різні моменти часу.

Критичним обмеженням є також повна відсутність або лише поверхове використання інтелектуальних агентів (ІА). Більшість популярних застосунків не використовують повноцінних ІА, здатних до навчання з підкріпленням (Reinforcement Learning, RL) для динамічного коригування навчальної траєкторії та оптимізації довгострокових навчальних стратегій, обмежуючись переважно евристичними підходами, коли точні алгоритми або процедури відсутні або не можуть бути використані. Це різко контрастує з потенціалом сучасних ІА, наприклад, агентів на основі алгоритму «Deep Deterministic Policy Gradient» (DDPG), які здатні навчатися складним політикам поведінки в динамічному середовищі. Окремою проблемою є так званий «холодний старт»: нові користувачі часто отримують стандартизований, неперсоналізований контент через відсутність достатньої кількості даних для індивідуалізації навчального процесу, що суттєво знижує ефективність та мотивацію на початкових етапах. Відсутність ефективного вирішення цієї проблеми призводить до високого відсотку відмов нових користувачів, оскільки початковий досвід не демонструє розуміння їхніх унікальних потреб. Крім того, спостерігається недостатня інтеграція моделей знань користувача, таких як «Bayesian Knowledge Tracing» (BKT), що лише зменшує ймовірність точного моделювання поточного рівня засвоєння матеріалу та, як наслідок, призводить до неоптимального навчання.

Варто зазначити, що обмеження існуючих систем часто пов'язані не стільки з відсутністю окремих технологій, скільки зі складністю їхньої

комплексної інтеграції в єдину, зручну для користувача та ефективну систему, особливо для реалізації дієвого фонового режиму. Термін «адаптивність» у багатьох комерційних продуктах є радше маркетинговим ходом, аніж глибоко реалізованою функцією, заснованою на складних моделях користувача та передових алгоритмах машинного навчання.

Виходячи з актуальності окресленої проблеми та виявлених недоліків існуючих рішень, ставиться задача розробити систему адаптивного фонового вивчення іноземних мов з використанням інтелектуальних агентів. Метою даної кваліфікаційної роботи є створення програмної системи, яка б забезпечувала високоефективне та зручне засвоєння іншомовного лексичного та граматичного матеріалу в режимі фонового навчання. Це досягатиметься шляхом застосування персоніфікованих навчальних стратегій, що динамічно адаптуються до індивідуальних особливостей, когнітивного стану та прогресу кожного користувача за допомогою спеціально розроблених інтелектуальних агентів.

Ключовий момент, або ж, інновація полягає не просто у створенні ще одного застосунку для вивчення мов, а в розробці цілісної системи, де інтелектуальний агент є центральним елементом, що керує навчальним процесом на основі глибокого розуміння користувача та сучасних принципів навчання. Успіх такої системи значною мірою залежатиме від здатності ІА не лише оптимізувати інтервали повторень, але й адаптувати тип, складність контенту та спосіб його подачі у фоновому режимі, враховуючи різні формати взаємодії.

Для досягнення поставленої мети необхідно розв’язати наступні основні підзадачі, які відображають ключові етапи проектування та реалізації системи і будуть детально розглянуті в розділах 2 та 3 даної роботи:

– розробити архітектуру програмної системи для адаптивного фонового вивчення мов. Це завдання включає обґрунтування вибору мікросервісної архітектури, яка забезпечить необхідну гнучкість,

масштабованість і відмовостійкість. Необхідно визначити ключові компоненти системи, такі як фронтенд та бекенд сервіси для функціонування інтелектуального агента, обробки даних користувача, управління навчальним контентом та надсилання фонових сповіщень. Також слід обрати оптимальний технологічний стек та визначитися із взаємодією між компонентами;

– визначити та описати детальні вимоги до системи. Це передбачає формулювання функціональних вимог, специфічних вимог користувачів, а також загальних системних та нефункціональних вимог. Особливу увагу необхідно приділити механізмам адаптації навчального процесу, персоналізації контенту, реалізації ефективного фонового режиму взаємодії та інтеграції з зовнішніми сервісами для отримання необхідної інформації. На основі цих вимог слід розробити прототипи користувацького інтерфейсу та провести аналіз завдань, що стоять перед користувачами;

– спроектувати модель інтелектуального агента (або групи агентів), що буде відповідати за динамічну адаптацію навчального процесу. Це включає вибір та обґрунтування алгоритмів навчання з підкріпленням, для оптимізації стратегії навчання, зокрема, вибору оптимального моменту, типу та змісту навчальної взаємодії. Важливою складовою є інтеграція моделей відстеження знань користувача та алгоритмів ІП, для точної оцінки рівня засвоєння матеріалу та надання релевантної інформації для ІА. Також необхідно розробити механізми для ефективного вирішення проблеми «холодного старту» для нових користувачів, можливо, шляхом початкового використання евристичних підходів або параметрів за замовчуванням для моделей інтервального повторення типу FSRS, з поступовим переходом до повністю керованої ІА стратегії. Ефективність ІА безпосередньо залежить від якості визначених вимог та повноти зібраних даних, а також від гнучкості архітектури, яка має підтримувати збір необхідної інформації та оперативне оновлення моделей агента;

– реалізувати ключові програмні модулі системи відповідно до розробленої архітектури та детальних специфікацій. Це завдання охоплює розробку програмного модуля базової версії інтелектуального агента, створення підсистеми управління навчальним контентом (картками), розробку механізму генерації та надсилання детальних фонових сповіщень, реалізацію користувацького інтерфейсу, а також забезпечення їхньої злагодженої взаємодії. Необхідно також реалізувати надійне збереження даних користувача, його прогресу та налаштувань;

– провести комплексне тестування та аналіз працездатності й ефективності розробленого прототипу системи, а саме тестування окремих модулів та системи в цілому.

2 ПРОЕКТУВАННЯ СИСТЕМИ АДАПТИВНОГО ФОНОВОГО ВИВЧЕННЯ МОВ З ВИКОРИСТАННЯМ ІНТЕЛЕКТУАЛЬНОГО АГЕНТА

2.1 Вимоги до системи

2.1.1 Визначення вимог до функціоналу

FR-01: основний функціонал побудований на використанні інтелектуальних агентів для розрахунку інтервалів повторення, а також використання додаткових алгоритмів, таких як алгоритми для інтервального повторення та алгоритми моделювання ймовірності знання по типу «Bayesian Knowledge Tracing (BKT)» [14], що дозволяють компенсувати неточність прогнозів агента та підвищити його загальну ефективність.

FR-02: користувач має можливість переглядати план повторень та обирати редагуєму «колоду» карток для повторення. Картки формуються як вручну, так і з використанням сторонніх сервісів для отримання додаткової інформації, таких як «Cambridge Dictionary» [15], «DeepL» [16].

FR-03: система підтримує CRUD-операції над картками. Користувач може додавати, редагувати, видаляти та переглядати свої картки.

FR-04: інтерфейс реалізовано у вигляді списку карток завантаженого з обраної користувачем «колоди», де поле для додавання нових елементів розташоване по центру, а картки із найбільшим часом до моменту повторення розміщуються над ним, з найменшим – під ним.

FR-05: уся навігація по «колоді» здійснюється шляхом циклічного прокручування карток. Також присутня можливість вибору кількості карток, що відображаються одночасно.

FR-06: повторення карток здійснюється з використанням push-сповіщень, які надсилаються відповідно до розрахованих інтервалів. Завдяки адаптивному підходу, система здатна ефективно оптимізувати

навчальний процес, забезпечуючи персоналізовані інтервали повторення на основі аналізу відповідей кожного користувача.

FR-07: система повинна вміти працювати з акаунтами (створення / видалення, вхід / вихід, збереження / редагування / видалення даних) використовуючи «Google Firebase», а також вміти зберігати дані безпосередньо на пристрої у разі нехватки місця та їх вигрузки як місце з’явиться.

FR-08: система повинна мати мінімальний набір мов: англійська та українська; перелік мов складається з переліків доступних мов сервісів, які впливають на поля вводу та виведення, також має бути присутнє авто-визначення мови для поля вводу.

FR-09: система повинна надавати можливість перегляду «повної інформації» чи «стислої інформації» по кожній «картці».

FR-10: система повинна використовувати «Fuzzy Search» (Нечіткий пошук) при виконанні пошуку по збереженим «карткам».

FR-11: система повинна вміти використовувати декілька атрибутів при пошуку по збереженим «карткам».

FR-12: контекстно-адаптивна ініціація навчальних сесій. Система проактивно пропонує навчальні сесії, аналізуючи дані сенсорів пристрою для визначення сприятливого контексту (наприклад, низька активність, тиша). Вона також навчається на основі минулих реакцій користувача, щоб ініціювати взаємодію в найбільш вдалий момент, підвищуючи ефективність навчання.

FR-13: динамічне коригування модальності взаємодії. Система динамічно обирає модальність подачі контенту (візуальну, аудіальну, тактильну), враховуючи середовище та активність користувача. Наприклад, у шумному місці замість аудіо-завдання буде запропоновано текстове, щоб забезпечити ефективне сприйняття інформації та зменшити когнітивне навантаження.

FR-14: адаптація презентації контенту залежно від активності користувача. Система адаптує обсяг та складність навчального матеріалу до фізичної активності користувача. Під час інтенсивних дій пропонуються короткі сесії, а періоди «простою», як очікування в черзі, використовуються для мікро-навчання, інтегруючи процес у повсякденне життя.

FR-15: адаптивна щільність інформації у навчальних матеріалах. Система дозволяє динамічно регулювати рівень деталізації навчальних матеріалів, перемикаючись між стислим та повним поданням інформації. Ця адаптація може залежати від етапу вивчення матеріалу або когнітивного стану користувача, оптимізуючи навантаження.

FR-16: конфігурація користувачем використання контекстних даних та чутливості адаптації. Система надає користувачеві прозорі налаштування для контролю над збором контекстних даних (доступ до сенсорів, активність). Користувач може вмикати/вимикати окремі функції та регулювати інтенсивність адаптації системи відповідно до власних уподобань.

FR-17: адаптація контенту та формату системних сповіщень залежно від середовища. Система адаптує зміст та формат push-сповіщень до поточного контексту користувача, перетворюючи їх на інтерактивні навчальні модулі. Наприклад, у шумному середовищі аудіо-сповіщення замінюються на текстові з кнопками для швидкої відповіді, мінімізуючи відволікання.

2.1.2 Визначення конкретних задач через вимоги користувачів

Ця система має наступні вимоги користувачів:

а) як користувач, який хоче скористуватися перекладачем, я хочу мати можливість;

– перекласти те, що я вказую у полі вводу на ту мову яку я обрав;

– обирати мову з якої я перекладаю як автоматично, так і з переліку доступних мов;

– одразу після перекладу побачити альтернативні його варіанти;

– для одиночних слів у полі вводу та виведення побачити таку «стислу інформацію»: частина мови, транскрипция, вимова (аудіо), приклади та теги;

– прослухати вимову слова;

б) як користувач, якому не подобається те, який результат був отриманий після перекладу, я хочу мати можливість;

– використовувати різні ресурси для пошуку даних;

– змінювати наповнення поля з перекладом;

– якщо мені, наприклад, не сподобалася вимова або транскрипція слова, чи його приклади або визначення, я міг би подивитися відповіді альтернативних постачальників даних для цього поля та якщо мені щось сподобалось більше – змінити на нього;

– як CRUD операції для кожного поля;

– редагувати параметри використання, та пріорітету сторонніх ресурсів, які використовуются для пошуку даних;

в) як користувач, який хоче подивитися детальну інформацію по якомусь слову, я хочу мати можливість;

– отримати таку «повну інформацію»: частина мови, транскрипція, вимова (аудіо), теги, рівень (A1–C2), мовний стиль (архаічний, формальний, неформальний, рідкий, сленг), синоніми, частота використання, приклади;

– відкрити/закрити «словник» (перемикач, за замовчуванням – вимкнено);

– якщо це єдине конкретне слово у полі вводу чи виведення – надавати можливість відкрити його у «словнику» (кнопка «показати у словнику»);

– якщо це фраза/ідіома/речення/текст – виділити конкретне слово або декілька окремих слів для автоматичного додавання їх у «пул» з якого можна буде перемикатися між цими словами та якщо «словник» відкрит – переглядати активне слово у ньому; пул єдиний для усіх виділених слів в межах застосунку;

– використати всі слова з «пулу» або їх частину для формування специфічного запиту до LLM;

– при формуванні специфічнного запиту до чату – обрати з пресету питань: «Чим слова відрізняються за значенням? », «Типове використання слів», «Наведи декілька прикладів до слів»; або створити нове питання та добавити його до пресету;

г) як користувач, який хоче зберігати отриману перекладачем інформацію, я хочу мати можливість;

– увійти у застосунок або створити акаунт з використанням мого Google аккаунту та використовувати цей акаунт для збереження даних;

– зберігти інпут та аутпут як «картки» та мати до них постійний доступ;

– при натисканні на «картку» – відкрити її у перекладачі;

– видалити «картку» зі збережених;

д) як користувач, який хоче знайти інформацію по якомусь слову або группі слів, я хочу мати можливість;

– відкрити пошук в будь-який момент часу, наприклад, через натискання клавіши «пробіл»;

– виконати швидкий пошук по усім збереженим «карткам»;

– побачити «стислу інформацію» на «картці»;

– вказати «fluent» мови, щоб не бачити інформацію навподобі прикладів та транскрипцій для мови яку добре знаю;

– викиристовувати один чи більше атрибут для пошуку, наприклад: «Хочу знайти всі збережені формальні

слова/фрази/ідіоми/речення/текст на конкретну тему, рівня B2, по конеретним тегам»;

– знайти потрібну мені інформацію навіть якщо в моєму пошукому запросі маються помилки;

е) як користувач, який хоче використовувати «картки» задля подальшого запам'ятовування слов з них, я хочу мати можливість;

– задля запомятання «карток», отримувати повідомленя від застосунку раз на певний (встановлений мною) час (5~10 хвилин) з «карткою» зі статусом «на вивченні» або «на повторенні», таким чином, я отримаю можливісь невимушено і без особливого напруження вивчати збережені «картки»;

– подивитися або відредагувати персоналізований під мене «календарний план» вивчення збережених «карток»;

– сховати переклад та «стислу інформацію», щоб не бачити підказок при «вивченні»;

– побачити або завантажити зображення яке описує наповнення «картки»;

– створити/обрати «теги» для «картки»;

– обирати конкретні теги, рівні, мовниі стилі, частоту використання для процесу «вивченя» через повідомлення.

2.1.3 Загальні вимоги до системи

BR-01: застосунок повинен мати доступ до сторонніх сервісів, які виконують роль постачальників даних для переліку доступних для перекладання мов. Сторонні сервіси додаються до застосунку програмно, повинні бути вказані мови, які підтримують ці сервіси.

BR-02: перелік мов складається з переліків доступних мов сервісів, які впливають на поля вводу та виведення.

BR-03: доступ до сервісів отримується завдяки публічному API або через парсингові скрипти якщо до API немає доступу.

BR-04: доступ до сховища має лише сам користувач.

BR-05: потрібен безкоштовний спосіб зберігання даних збережених користувачем, завжди потрібно бути місце та доступ до сховища. Безкоштовне сховище досягається завдяки збереженню даних у хмарі на подобі Google Firebase.

BR-06: якщо місце в хмарі закінчилось, тереба зберігати дані безпосередньо на устройстві до тих пір як місце з’явиться, перевірку виконувати раз на день, після збереження даних у хмару – видаляти їх з пристрою.

NFR-01: система, її структура та кількість використовуємих сторонніх сервісів повинні бути масштабованими.

NFR-02: система та пошук (не довше 2 секунд) в її межах повинні бути швидкими.

NFR-03: система повинна бути безпечною.

NFR-04: інтерфейс користувача повинен бути інтуїтивно зрозумілим та зручним.

2.2 Специфікації до варіантів використання

Перед тим як перейти до визначення варіантів використання та створення специфікації за ними, треба визначити основних акторів, які будуть безпосередньо брати участь у створюваній системі. Таким чином, у цій системі користувачем-людиною є:

– користувач: використовує систему для перекладу та вивчення іноземних мов;

– адміністратор: керує налаштуваннями, масштабованістю та поновленнями системи та забезпечує її працездатність.

Та користувачем-задачею:

– перекладач: введення тексту для перекладу, вибір мови, отримання альтернативних варіантів перекладу;

– сховище: створення, редагування та видалення «карток» з перекладеною інформацією;

– календарний план: отримання повідомлень з «картками» за персоналізованим календарним планом вивчення;

– пошукач: виконання швидкого та ефективного пошуку по збереженим «карткам».

Таким чином, за визначиними раніше вмогами (див. 2.1) можна створити use-case діаграму (рисунок 2.1) для візуалізації взаємодії між користувачами та системою.

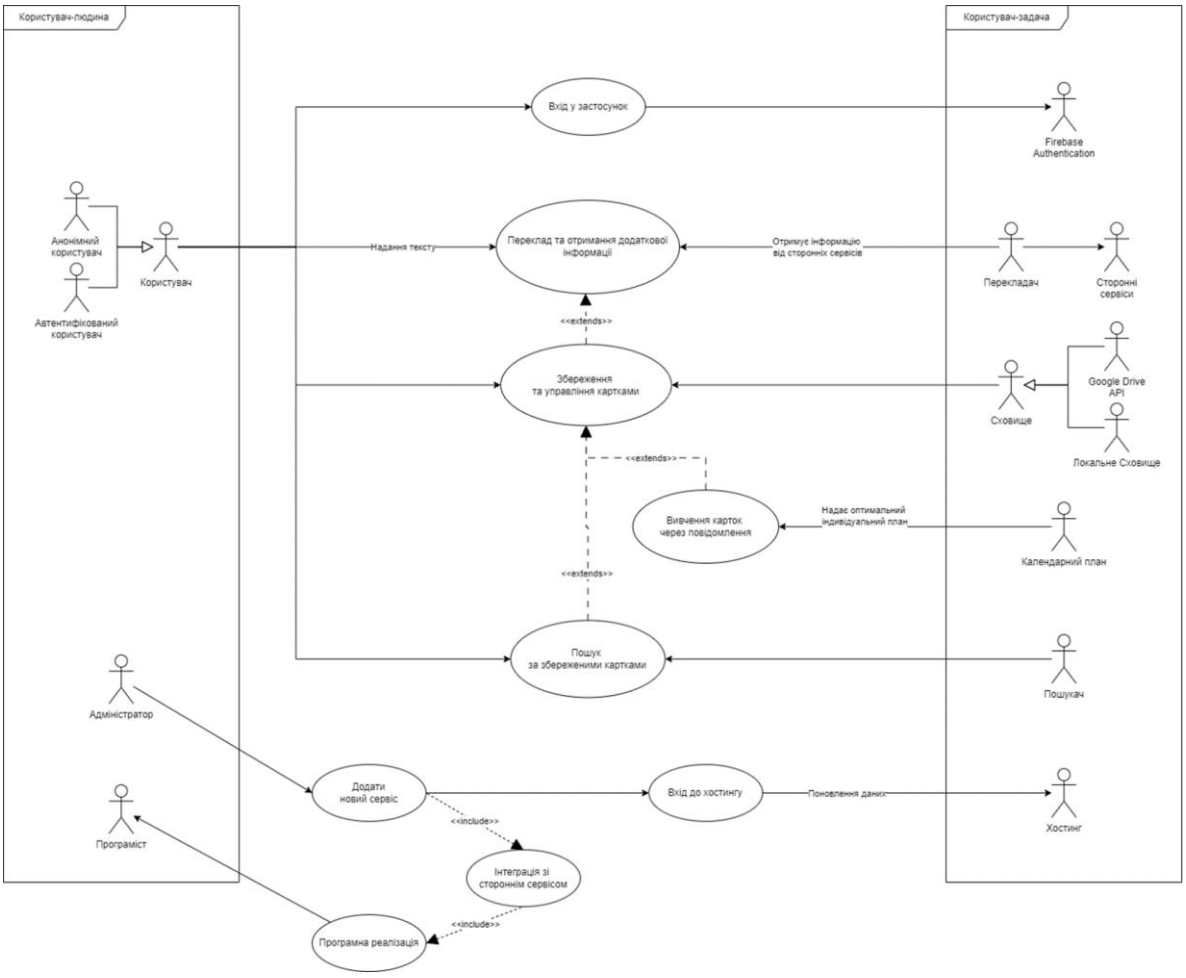

Рисунок 2.1 – Use-case діаграма ІС

Діаграма варіантів використання стане основою для подальшого проектування архітектури системи (див. 3.2.1) та розробки інтерфейсів (див. 2.3). Вона також допоможе визначити межі відповідальності кожного компонента та уникнути дублювання функціоналу.

Тепер коли визначено та візуалізовано основні взаємодії між користувачами та системою можна перейти до створення необхідних специфікації для інформаційної системи. Які включатимуть саму діаграму, її короткий опис, що містить загальну інформацію про основні актори (користувач, адміністратор, автоматичні задачі) та ключові сценарії їхньої взаємодії з системою, такі як переклад тексту, керування картками, навчання за планом тощо. А також основний та альтернативний потік дій, де основний описує стандартний сценарій виконання варіанту використання, тоді як альтернативний потік вказує на відхилення від норми (наприклад, помилка перекладу або відсутність інтернет-з’єднання) та способи їх обробки.

В кінці також потрібно надати передумови та післяумови. Передумови визначають умови, необхідні для запуску сценарію (наприклад, авторизація користувача або наявність підключення до бази даних), а післяумови фіксують очікуваний стан системи після виконання дій (наприклад, оновлення сховища карток або запис логів операції).

Таким чином, подивимося на рисунок 2.2, який відображає вхід у застосунок. Даний варіант використання дозволяє користувачу увійти у застосунок для більш зручного використання.

Основний потік подій:

а) вхід за допомогою Google;

– користувач входить до системи через Firebase Authentication за допомогою Google;

– після успішного входу система отримує токен користувача;

– сиситема використовує цей токен для авторизації користувача в Google Drive API;

– система отримує можливість пацювати з файлами на Google Диску після успішної автентифікації та авторизації;

б) вхід через Facebook, Twitter, GitHub та інші;

– користувач входить до системи за допомогою вибраного провайдера (наприклад, Facebook) через Firebase Authentication;

– після успішного входу система отримує токен Firebase;

– система використовує підпроцес «Отримати токен користувача»;

– користувач проходить «Пройти OAuth 2.0 flow для Google» у відповідному підпроцесі, де надає дозвіл на доступ до Google Диску;

– система отримує токен доступу Google API;

– система використовує отриманий токен доступу для роботи з Google Drive API;

в) анонімна автентифікація;

– користувач входить до системи через Firebase Authentication за допомогою Google завдяки опції анонімного входу;

– після успішного входу система отримує токен користувача.

Альтернативні потоки:

– неправильне ім'я або пароль. Якщо під час виконання Основного потоку виявиться, що користувач ввів неправильне ім'я і/або пароль, система виводить повідомлення про помилку. Користувач може повернутися до початку Основного потоку або відмовитися від входу в систему, при цьому виконання варіанта використання завершується.

Передумови:

– користувач має гугл акаунт.

Післяумови:

– якщо варіант використання завершився успішно, користувач успішно увійде до застосунку.

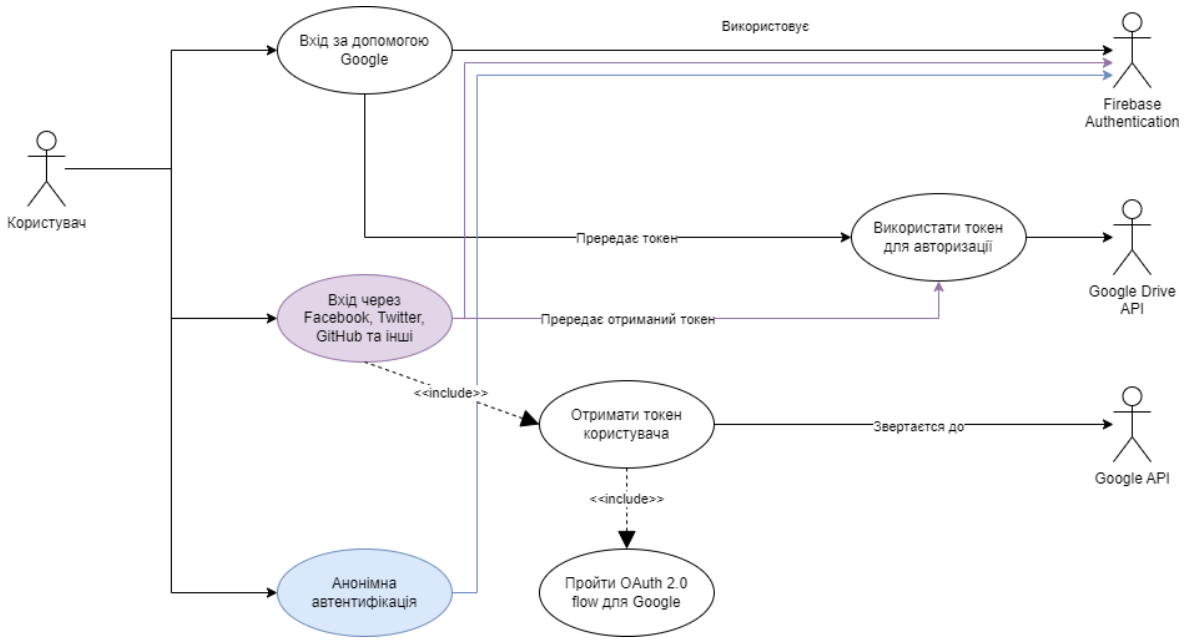

Рисунок 2.2 – Діаграма варіантів використання для входу у застосунок

Далі подивимося на рисунки 2.3, 2.4, які відображають переклад та отримання додаткової інформації.

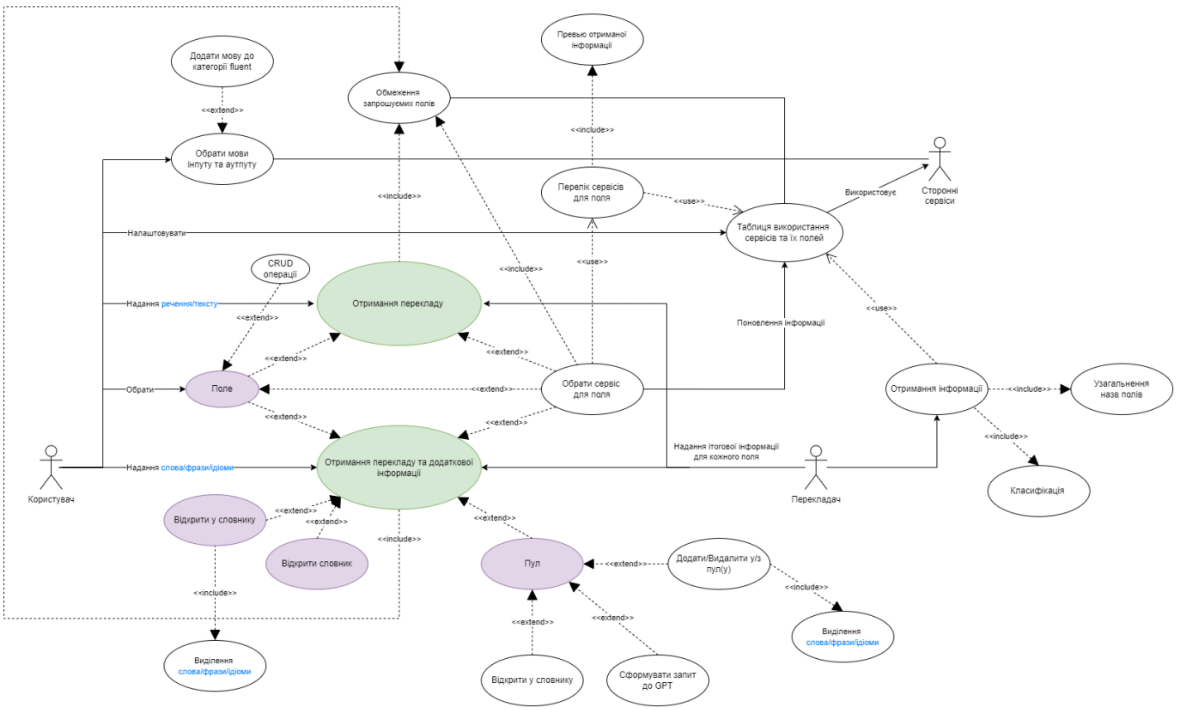

Рисунок 2.3 – Діаграма варіантів використання для перекладу та отримання додаткової інформації

Даний варіант використання дозволяє користувачу отримувати переклад речень, тексту або окремих слів/фраз/ідіом, а також отримувати додаткову інформацію від сторонніх сервісів.

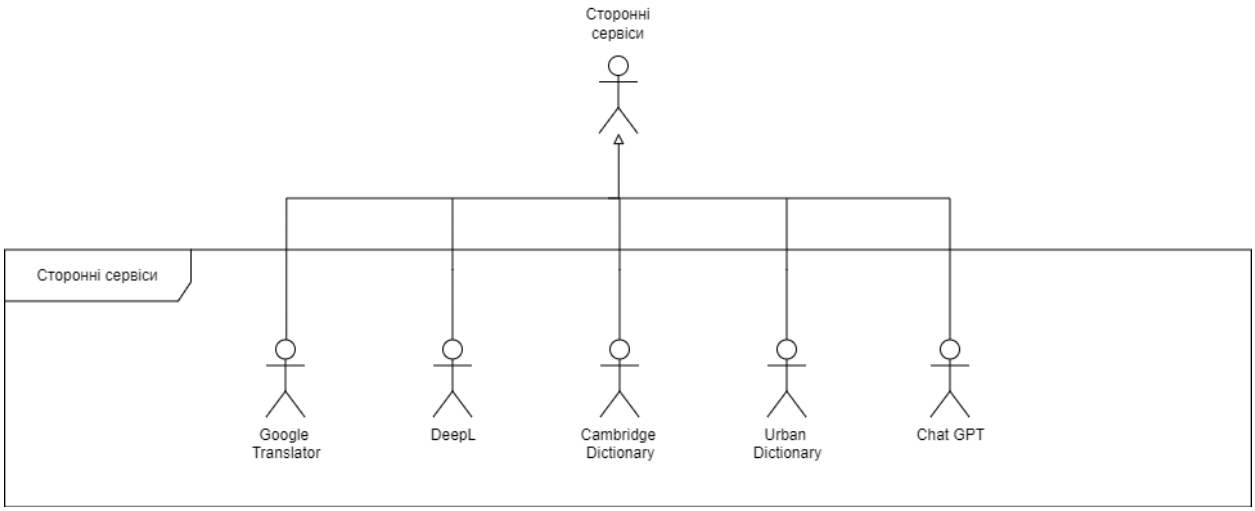

Рисунок 2.4 – Сторонні сервіси [Узагальнення]

Основний потік подій:

а) обрати мови інпуту та аутпуту:

– система надає список усіх доступних мов, отриманий зі списку вказаних мов у стороніх сервісів мовного перекладу (які впливають на поля інпут та аутпут);

– користувач обирає з переліків потрібні йому мови, для поля інпуту можливо обрати автоматичне визначення мови;

– після вибору для усіх процесів у перекладачі будуть використовуватися обрані мови;

б) отримання перекладу та додаткової інформації:

– користувач надає слово, фразу або ідіому для перекладу та отримання додаткової інформації;

– виконується один з підлеглих потоків «Обмеження запрошуємих полів» (вимикає зайві (при обранні нових сервісів для

полів, запрошувати дані тільки для цих полів) поля з запросу до сторонніх сервісів);

– перекладач отримує інформацію зі сторонніх сервісів;

– виконується один з підлеглих потоків «узагальнення» (так як назви полів у різних сервісів та цьому застосунку можуть відрізнятися);

– перекладач отримує узагальнену інформацію по усім полям;

– виконується один з підлеглих потоків «Класифікація» (отримання тегів та специфічних атрибутів);

– інформація надається до кожного з видимих полів (у даному випадку: поля інпут, аутпут та стисла інформація);

в) поле:

– перекладач отримує інформацію зі сторонніх сервісів для кожного поля;

– інформація надається до кожного з видимих полів за пріоритетом (порядков) сервісів з таблиці використання.

Альтернативні потоки:

а) у будь-який момент користувач може додати мову з переліку у fluent категорію, додати мову до категорії «fluent»:

– система надається список усіх доступних мов, отриманий зі списку вказаних мов у стороніх сервісів мовного перекладу (які впливають на поля інпут та аутпут);

– користувач обирає зі списку потрібну мову;

– після вибору для цієї мови вже не будє наводитися стисла інформація (приклади, синоніми, теги, транскрипція, вимова (аудіо)), але залишається можливість відкрити у «словнику»;

б) отримання перекладу:

– користувач надає речення або текст для перекладу;

– у разі надання користувачем речення або тексту, які не міститяться у сторонніх сервісах як фрази чи ідіоми буде

виконуватися простий переклад без отримання додаткової інформації та можливості відкрити словник;

– виконується один з підлеглих потоків «Обмеження запрошуємих полів» (вимикає зайві поля з запросу до сторонніх сервісів – усі крім інпуту та аутпуту);

– перекладач отримує інформацію зі сторонніх сервісів;

– виконується один з підлеглих потоків «Узагальнення» (так як назви полів у різних сервісів та цьому застосунку можуть відрізнятися);

– перекладач отримує узагальнену інформацію по усім полям;

– виконується один з підлеглих потоків «Класифікація» (отримання тегів та специфічних атрибутів);

– інформація надається до кожного з видимих полів (у данному випадку поля інпут та аутпут);

– виводяться теги;

в) користувач може обрати сервіс для кожного поля:

– виконується один з підлеглих потоків «Таблиця використання сервісів та їх полей», який формує таблицю за вказаними у сервісах полях «впливу»;

– таблиця надає перелік стороніх сервісів для кожного поля за вказаними у ній сервісах для цього поля;

– виконується один з підлеглих потоків «Превью отриманої інформації» для візуального представлення стислої сводки за інформацією наданою сервісом;

– користувач обирає подрібний сервіс з переліку;

– виконується один з підлеглих потоків «Обмеження запрошуємих полів», який вимикає усе окрім полів зі зміною у сервісі у таблиці використання сервісів;

г) додання слова/фрази/ідіоми у пул:

– виконується один з підлеглих потоків «Виділення слова/фрази/ідіоми»;

– до пулу автоматично додається виділене;

д) відкрити у словнику (якщо користувач надав слово/фразу/ідіому):

– переклач надає вже отриману після запросу до сервісів інформацію «словнику»;

– відображається словник;

е) відкрити у словнику (якщо користувач надав речення/текст):

– виконується один з підлеглих потоків «Виділення слова/фрази/ідіоми»;

– перекладач отримує інформацію зі сторонніх сервісів;

– виконується один з підлеглих потоків «узагальнення» (так як назви полів у різних сервісів та цьому застосунку можуть відрізнятися);

– перекладач отримує узагальнену інформацію по усім полям;

– виконується один з підлеглих потоків «Класифікація» (отримання тегів та специфічних атрибутів);

– відображається «словник» з усієї отриманої інформації;

ж) видалення слова/фрази/ідіоми з пулу:

– користувач обирає або видалення конкретного елементу пула через натискання кнопки видалення на елементі, або виділяє одразу декілька елементів;

– натискає на кнопку «видалити обрані», або видаляє усі елементи через відповідну кнопку;

з) відкрити у словнику слово/фразу/ідіому з пулу:

– користувач обирає конкретний елемент пулу;

– обирає дію з переліку дій – відкриття у словнику;

и) сформувати запит до LLM:

– користувач обирає ствrення запиту за словом(ами), фразою(ами) чи ідіомою(ми) з пулу;

– після вдачного запиту – отримує запитану інформацію, інакше отримує – помилку при виконанні запиту;

к) якщо сторонньому сервісу не вдається отримати інформацію протягом 3 секунд, система припиняє запит до цього сервісу та змінює його стан у «таблиці використання» для цього перекладу на «вимкнено»;

л) відсутність поля заявленого у сервісі як «впливного» (поле для якого сервіс має надавати інформацію) в повернутої з сервісу інформації викликає зміну його стану у «таблиці використання» для цього перекладу на «вимкнено».

Передумови:

– користувач увійшов в систему;

– система має доступ до сторонніх сервісів для перекладу та отримання додаткової інформації.

Післяумови:

– якщо варіант використання завершився успішно, користувач отримує переклад тексту (та додаткову інформацію). В іншому випадку стан системи не змінюється;

– система оновлює таблицю використання сервісів та полів відповідно до отриманих даних;

– слова/фрази/ідіоми додані або видалені з пула відповідно до запиту користувача.

Ця специфікація описує детальний сценарій використання системи для перекладу та отримання додаткової інформації, враховуючи всі можливі варіанти та потоки подій.

Наступні діаграми варіантів використання зображені на рисунках 2.5, 2.6, які відображають збереження та управління картками для автентифікованого користувача (рисунок 2.5) та збереження та управління картками для анонімного користувача (рисунок 2.6), відповідно. Даний варіант використання дозволяє користувачу зберігати, редагувати та видаляти інформацію у вигляді карток.

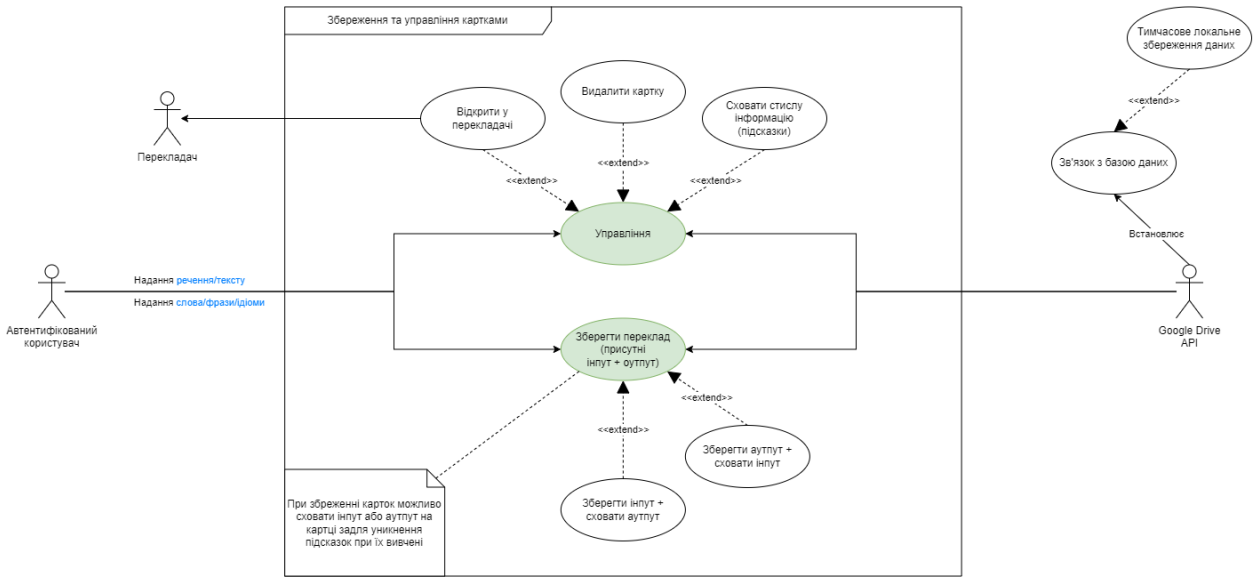

Рисунок 2.5 – Діаграма варіантів використання для збереження та управління картками для автентифікованого користувача

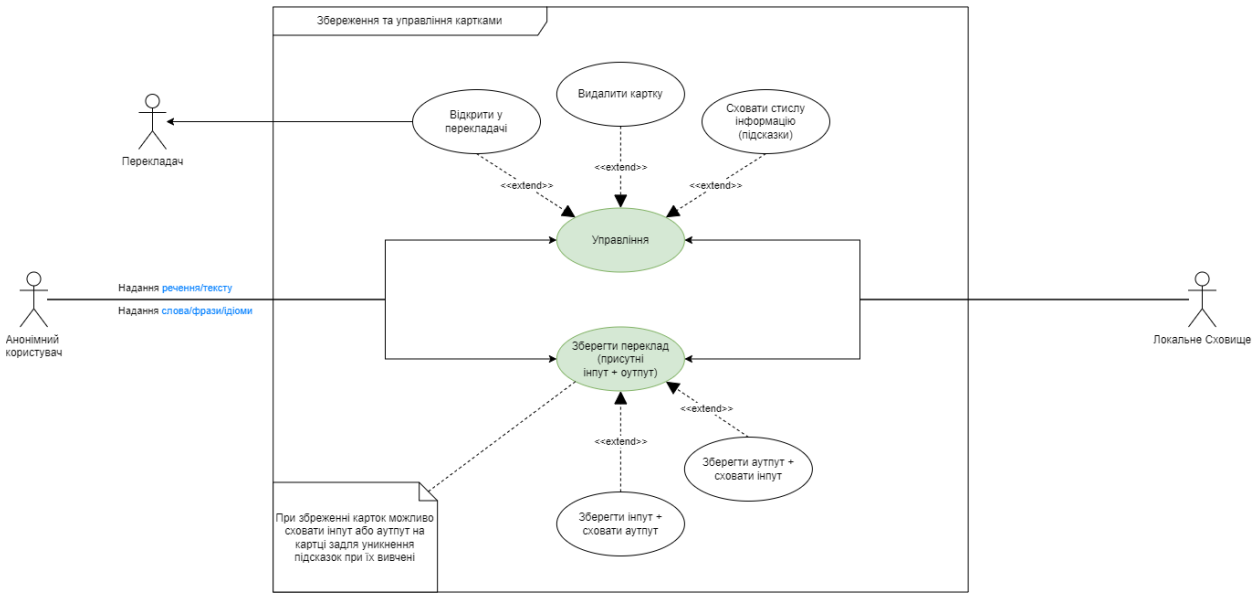

Рисунок 2.6 – Діаграма варіантів використання для збереження та управління картками для анонімного користувача

Основний потік подій:

а) зберегти переклад, коли присутні інпут та аутпут поля;

– виконується один з підлеглих потоків «Зв'язок з базою даних»;

– користувач натискає кнопку збереження перекладу;

– окрім усієї необхідної інформації також зберігаються відредаговані поля та використані сервіси для всіх полів;

– до «сховища» надається новий запис;

б) видалити картку;

– виконується один з підлеглих потоків «Зв'язок з базою даних»;

– користувач натискає кнопку «Видалити картку»;

– зі «сховища» вилучається зазначена картка;

«Зберегти інпут» та «Сховати аутпут», а також «Зберегти аутпут» та «Сховати інпут» надають можливість сховати інпут або аутпут на картці, при збреженні карток, задля уникнення підсказок при їх вивчені, але також подивитися сховану частину при наведенні курсором миши на неї.

Альтернативні потоки:

а) відкрити у перекладачі:

– користувач натискає на заголовок збереженої картки;

– виконується запит до Перекладача за полем інпуту та використаними сервісами для всіх полів картки;

– до інформації отриманої Перекладачем для полів додаються збережені значення відредагованих полів та обираються як активні;

– колишня кнопка збереження перекладу змінюється на кнопку його видалення;

б) сховати стислу інформацію (підсказки):

– користувач натискає на відповідну кнопку на картці;

– слисла інформація ховається з можливістю повторного її відкриття;

в) помилка збереження картки на пристрій видається, якщо система не може зберегти картку;

г) помилка редагування картки на пристрії видається, якщо система не може знайти картку для редагування;

д) помилка видалення картки з пристрою, видається якщо система не може знайти картку для видалення.

Локальне збереження даних – використовується тільки у випадках коли:

– помилка збереження картки у базу даних;

– помилка редагування картки з бази даних;

– помилка видалення картки з бази даних;

– користувач зайшов анонімно.

Зберігає дані на пристрої доки зв’язок з базою даних або місце на не неї з’явиться, після чого вигружає у базу даних та видаляє їх.

Передумова – користувач увійшов в систему.

Післяумова – якщо варіант використання завершився успішно, картка буде збережена, відредагована або видалена.

На рисунку 2.7 можна побачити діаграму варіантів використання для вивчення карток через повідомлення. Даний варіант використання дозволяє користувачу отримувати повідомлення та вивчати збережені картки через системні повідомлення від застосунку. Система надсилає користувачу повідомлення за індивідуальним календарним планом зі збереженими картками.

Далі йде основний потік подій де, «Почати вивчення» – стартує «Процес вивчення», який має наступний потік дій:

– виконується один з підлеглих потоків «Отримати збережені картки» – отримується повний список карток;

– користувач вказує або не вказує критерії звуження;

– користувач вказує інтервали між повідомленнями (5 ~ 10 хвилин);

– виконується один з підлеглих потоків «Інтервали між повідомленнями»;

– виконується один з підлеглих потоків «Звузити вібірку карток» задля відсіння речень/текстів;

– виконується один з підлеглих потоків «Створення індивідуального плану»;

– календарний план надає (звужений) ідивідуальний план;

– користувач натискає кнопку «Почати вивчення»;

– виконується один з підлеглих потоків «Надсилати повідомлення з картками».

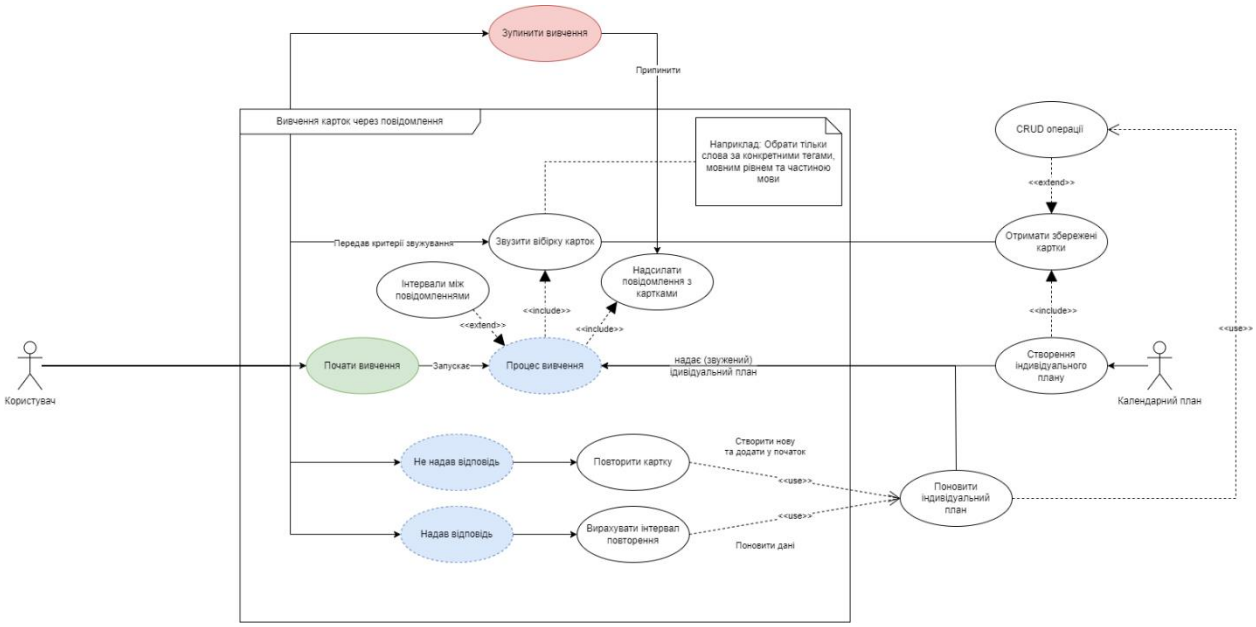

Рисунок 2.7 – Діаграма варіантів використання для вивчення карток через повідомлення

Зупинити вивчення включає:

– користувач натискає відповідну кнопку;

– припиняється робота одного з підлеглих потоків «Надсилати повідомлення з картками»;

– після початку вивчення стають доступні колаборативні події.

Коли користувач «Надав відповідь»:

– виконується один з підлеглих потоків «Вирахувати інтервал повторення»;

– вираховані інтервали задаються відповідним карткам та зберігаються;

– виконується один з підлеглих потоків «Поновити індивідуальний план» для внесення змін на базі нових даних.

Коли користувач не надав відповідь:

– виконується один з підлеглих потоків «Повторити картку»;

– задається інтервал рівний –1 (тобто, додатється у початок календарного плану) задаються відповідним карткам та зберігаються;

– виконується один з підлеглих потоків «Поновити індивідуальний план» для внесення змін на базі нових даних.

Альтернативні потоки, включають:

– звузити вібірку карток – користувач може обрати, які саме картки повинні ідти на вивчення, наприклад: обрати тільки слова за конкретними тегами, мовним рівнем та частиною мови.;

– CRUD операції – користувач може вручну відредагувати та реорганізувати календарний план;

– користувач не налаштував отримання повідомлень від застосунку – повідомити його про цей факт та запропонувати зробити це, щоб мати можливість користуватися цим модулем.

Передумова – користувач увійшов в систему та налаштував отримання повідомлень від застосунку.

Післяумова – якщо варіант використання завершився успішно, процес вивчення пройшов вдачно, а дані картки буде оновлено.

Пошук за збереженими картками (рисунок 2.8) є останнім варіантом використання. Даний варіант використання дозволяє користувачу шукати інформацію серед збережених карток.

Основний потік подій:

а) нечіткий пошук;

– виконується один з підлеглих потоків «Отримати збережені картки» – отримується повний список карток;

– користувач вводить запит для пошуку;

– пошукач виконує пошук серед збережених карток;

– виконується один з підлеглих потоків «Представлення знайдених карток»;

– користувач переглядає результати та обирає потрібну картку.

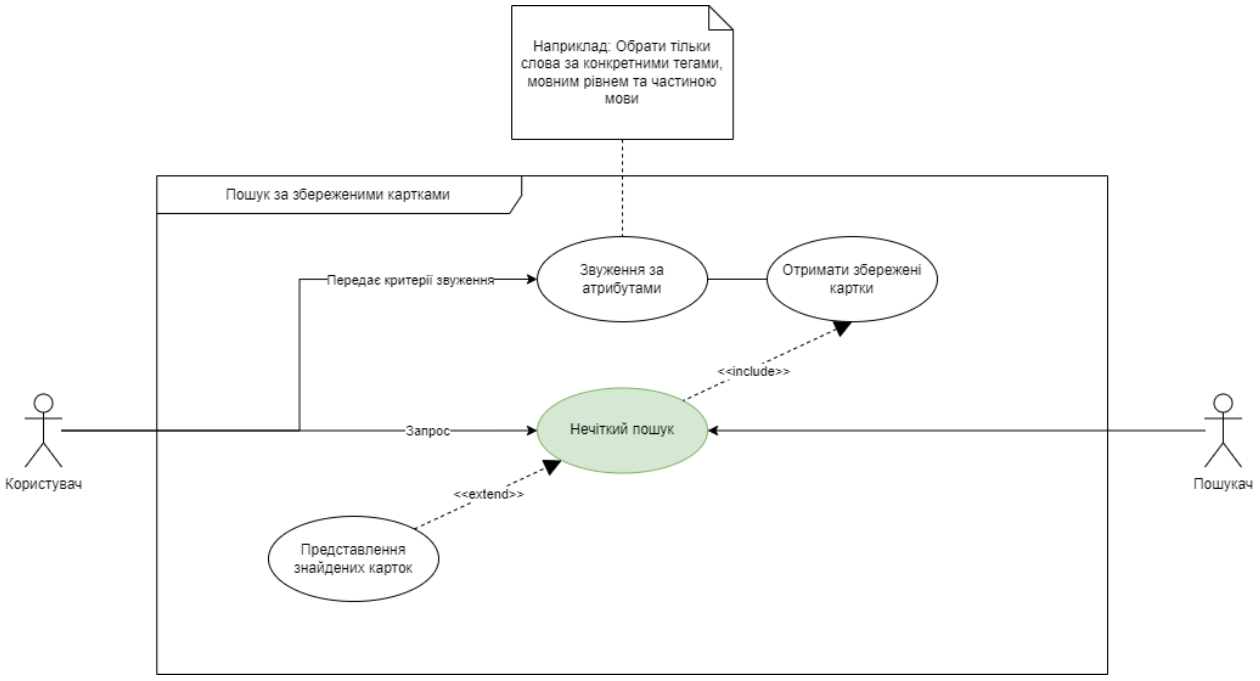

Рисунок 2.8 – Діаграма варіантів використання для пошуку за збереженими картками

Альтернативні потоки:

а) звуження за атрибутами (для звужування пошуку при нечіткому пошуку), наприклад: обрати тільки слова за конкретними тегами, мовним рівнем та частиною мови

– виконується один з підлеглих потоків «Отримати збережені картки» – отримується повний список карток;

– користувач вводить запит для пошуку;

– користувач задає критерії звужування;

– пошукач виконує пошук серед збережених карток;

– виконується один з підлеглих потоків «Представлення знайдених карток»;

– користувач переглядає результати та обирає потрібну картку.

б) картки не знайдено – якщо система не знаходить жодної картки за запитом, вона виводить повідомлення про відсутність результатів.

Передумова – користувач увійшов в систему.

Післяумова – якщо варіант використання завершився успішно – користувач отримає результати пошуку.

2.3 Прототипи інтерфейсу

Після того, як було з'ясувано основні завдання і вимоги, складемо структуру і макет інтерфейсу. Створення прототипів є ключовим етапом проектування, що дозволяє візуалізувати майбутній застосунок та перевірити логіку взаємодії користувача з системою ще до початку повноцінної розробки. Вони слугують візуальним втіленням функціональних вимог та сценаріїв використання, описаних у відповідних специфікаціях.

Основна мета цього етапу – створити інтуїтивно зрозумілий та ефективний користувацький інтерфейс, що відповідає задачам, які стоять перед користувачем. На основі аналізу варіантів використання було розроблено макети для ключових екранів системи: основного екрану керування картками та системного сповіщення для фонового навчання. Ці прототипи демонструють розташування основних елементів керування, навігаційні потоки та способи подання навчальної інформації, що є основою для подальшої програмної реалізації.

Для того, щоб користувач зміг обрати цільові мови (з якої на яку) для перекладання додамо відповідні поля (рисунок 2.9, №1). Після обрання необхідних мов він повинен мати можливість ввести слова, фрази чи ідеоми, які йому потрібно додати (рисунок 2.9, №2), але для того, щоб користувач

не міг вводити стінки тексту для яких точно не будуть знайдені визнячення чи інша інформація треба додати деякі обмеження, наприклад, лічильних максимальної кількості символів (рисунок 2.9, №3).

Так як цей застосунок напрямлений на вивчення доданого користувачем через інтервальні системні фонові повідомлення – потрібно надати користувачу можливість переглядати час до відправлення наступного повідомлення (рисунок 2.9, №4).

При перегляді детальної інформації за якоюсь карткою користувачу потрібно бачити основну інформацію у стислому вигляді тож додамо випадаючі списки з визначеннями (рисунок 2.9, №5), які він сам зможе розкривати за необхідністю та корисною інформацією по цьому визначенню у стислому та інформативному вигляді, наприклад, його мовним рівнем, частиною мови та можливим тлумаченням слова (рисунок 2.9, №6).

Коли така необхідність виникла треба показати користувачу основі відомості по визначенню, а саме стисло показати транскрипції для наданого слова та надати можливість їх прослухати (рисунок 2.9, №7), показати декілька прикладів використання у реченнях (рисунок 2.9, №8), а також надати можливість швидко та зрозуміло переглядати визначення завдяки секціям з клікабельними мініатюрами визначень за їх корисною інформацією, вище по списку визначень (рисунок 2.9, №9) та нижче (рисунок 2.9, №10).

Ще не можна забувати про можливості додання, видалення, перегляду інших карток та поновлення інформації у них. Для цього можна зробити відповідні кнопки а також, для зручності, шорткати, які можна представити користувачу у відповідних секціях. Нижня (рисунок 2.9, №11) допоможе користувачу створити да додати картку (надав корисну інформацію та підсказки) у початок календарного плану, а верхня (рисунок 2.9, №12) покаже як додати у кінець.

Таким чином, після створення декількох корисувачу буде потрібна можливість переключатися та переглядати різні картки. Для цього можна

додати циклічний список усіх карток, який користувач зможе прокручувати та обирати конкретні картки (рисунок 2.9, №13). Тепер же можна замислитися над тим, що картки можуть стати зайвими або просто відібражати некоректну або неточну інформацію, тож треба надати користувачу можливість видалити чи змінити їх, або ж просто зробити якісь додаткові дії на ними, для цього реалізуємо секцію з цими спеціальними функціями та розмістимо їх так, щоб при перегляді якоїсь картки ці дії автоматично показувалися при необхідності (рисунок 2.9, №14).

Рисунок 2.9 – Прототип «плану» (переліку карток, які підшружаються залежно від обраної «колоди» карток)

Тепер розглянемо процес навчання, під час якого користувачу надходять системні тост повідомлення від застосунку (рисунок 2.10, №1), де йому потрібно надати як слово, фразу чи ідіому для повторення (рисунок 2.10, №2), так і привести визначення (рисунок 2.10, №3) та приклад використання (рисунок 2.10, №4) у реченні.

Для того, щоб дізнатися чи користувач знає цю картку, чи ні, а також надати йому можливість відкласти її потрібно надати відповідні кнопки з діями (рисунок 2.10, №5).

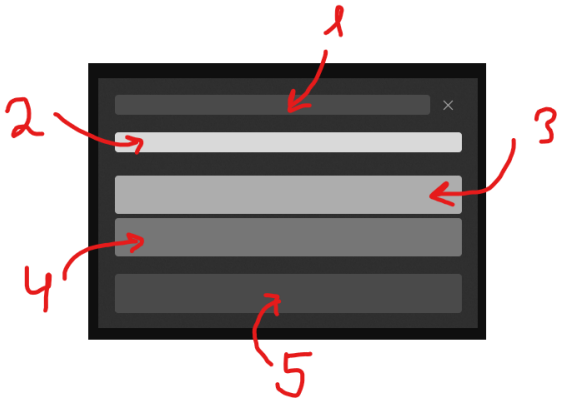

Рисунок 2.10 – Прототип системного тост повідомлення з карткою

Пілся створення основних прототипів інтерфейсу користувача можна перейти до визначення структури для користувачів, відповідно до таблиці А.1 у додатку А.

Таким чином, було проаналізовано завдання, що стоять перед користувачами.

2.4 Проектування інтелектуального агента

2.4.1 Загальна архітектура та технології

В основі розроблюваної інтелектуальної системи навчання лежить синергія кількох передових технологій штучного інтелекту, що забезпечує

її інноваційність та потенціал для глибокої персоналізації навчального процесу. Центральним компонентом системи є алгоритм «Deep Deterministic Policy Gradient» (DDPG), який пропонується використовувати для прогнозування неперервних інтервалів повторення навчального матеріалу. Вибір DDPG обґрунтований його здатністю ефективно працювати в просторах неперервних дій, що точно відповідає задачі визначення оптимального часового проміжку для наступного повторення, який може бути будь-яким значенням (наприклад, кілька годин, днів або тижнів).

Для підвищення загальної ефективності DDPG-агента та, що особливо важливо, для часткового вирішення проблеми «холодного старту» (ситуації, коли система має недостатньо даних про нового користувача для якісної персоналізації), планується інтеграція моделей «Free Spaced Repetition Scheduler» (FSRS) та «Bayesian Knowledge Tracing» (BKT) безпосередньо у функцію винагороди DDPG-агента.

Моделі FSRS та BKT, що спеціалізуються на моделюванні процесів запам'ятовування та рівня знань користувача відповідно. Таким чином, замість того, щоб винагорода базувалася лише на бінарному результаті відповіді користувача (правильно/неправильно), вона також враховуватиме прогнози FSRS щодо стабільності пам'яті та ймовірності пригадування, а також оцінку BKT щодо ймовірності знання матеріалу. Такий підхід дозволяє агенту швидше навчатися ефективним стратегіям інтервального повторення, особливо на початкових етапах взаємодії з новими користувачами.

Окрім визначення оптимального часу повторення, система передбачає розробку інтелектуальної системи навчання «Intelligent Tutoring System» (ITS) з використанням алгоритму «Deep Q-Learning» (DQL) [18]. DQL призначений для управління дискретним простором дій, що означає вибір конкретного навчального контенту (яку картку, якої скадності або типу завдання запропонувати) та способу його подачі (який формат, який

рівень складності). Такий розподіл завдань, де DDPG визначає «коли», а DQL – «що» і «як», створює комбінаціію з декількох ІА, де кожен з них використовується відповідно до своїх сильних сторін. Це дозволить надавати кожному учню персоналізований навчальний досвід, адаптуючись до його унікальних потреб у реальному часі.

Важливою особливістю системи є її глибока контекстна обізнаність. Ця функціональність дозволяє адаптувати тип та спосіб подачі навчального контенту до поточних умов користувача, таких як рівень навколишнього шуму, вид поточної діяльності (наприклад, робота, поїздка в транспорті, відпочинок) та інші фактори, що будуть детальніше розглянуті в підрозділі 2.4.4.

Для ефективного тренування RL-агентів (DDPG та DQL) пропонується застосувати гібридний підхід. Цей підхід поєднує тренування потужної глобальної моделі на великому анонімізованому масиві даних, зібраних від усіх користувачів, та подальшу персоналізацію шляхом тонкого налаштування (fine-tuning) цієї моделі на індивідуальних даних кожного конкретного користувача. Деталі цього підходу розглядаються в підрозділі 2.4.3.

Ключові технології штучного інтелекту, що використовуються в системі, та їх основні функції наведені в таблиці 2.1.

Таблиця 2.1 – Ключові технології ШІ та їх функції в системі

Технологія	Акровнім	Основна функція в системі
1	2	3
Deep Deterministic Policy Gradient	DDPG	Прогнозування оптимальних неперервних інтервалів повторення навчального матеріалу.
Free Spaced Repetition Scheduler	FSRS	Моделювання стабільності пам'яті, складності матеріалу, ймовірності пригадування; інтеграція у винагороду DDPG.

Продовження таблиці 2.1

1	2	3
Bayesian Knowledge Tracing	BKT	Моделювання ймовірності знання; інтеграція у винагороду DDPG.
Deep Q-Learning	DQL	Управління дискретним простором дій: вибір навчального контенту (що?) та способу його подачі (як?) для персоналізації.
Інтелектуальна система навчання	ITS	Загальна система для надання персоналізованого навчального досвіду.
Модуль контекстної обізнаності	-	Збір та аналіз даних про оточення та активність користувача для адаптації навчального процесу.

Ця таблиця систематизує інформацію про технологій та слугує зручним довідником для розуміння архітектури застосованих алгоритмів та ІА у створюємій системи.

2.4.2 Компонентна взаємодія в системі

Ефективна робота розроблюваної інтелектуальної системи навчання залежить від злагодженої та чітко визначеної взаємодії між її основними компонентами. Ця взаємодія забезпечує адаптивність та персоналізацію навчального процесу на різних рівнях.

DDPG-агент відіграє роль стратегічного планувальника, визначаючи оптимальний момент для повторення кожної окремої навчальної картки. Він відповідає на ключове питання: «коли повторювати?». Його рішення базуються на складній моделі, що враховує історію взаємодії користувача та прогнози щодо кривої забування.

Коли настає час для повторення, визначений DDPG-агентом, або коли система самостійно ідентифікує сприятливий момент для навчання (наприклад, користувач вільний і знаходиться у відповідному середовищі, що визначається модулем контекстної обізнаності), активується DQL-агент. Цей агент відповідає за наповнення навчальної сесії: він приймає рішення про те, що саме запропонувати користувачеві (яку конкретну картку, який тип завдання) і як це зробити (який формат подання використати, який рівень складності обрати). Така структура дозволяє ефективно розділити складну проблему оптимізації навчання на більш керовані підзадачі.

Моделі FSRS та BKT функціонують як допоміжні інтелектуальні компоненти, що постійно оновлюються на основі відповідей та взаємодій користувача з системою. Оновлені оцінки стабільності пам'яті, складності матеріалу та ймовірності пригадування (від FSRS), а також ймовірності актуального знання матеріалу (від BKT), слугують важливими вхідними даними для обох RL-агентів (DDPG та DQL).

Це забезпечує їх актуальною інформацією, необхідною для прийняття обґрунтованих рішень. Наприклад, DDPG-агент може скоригувати запропонований інтервал, якщо BKT вказує на низьку ймовірність знання, навіть якщо попередній інтервал був стандартним. Аналогічно, DQL-агент може обрати простіший тип завдання, якщо FSRS сигналізує про високу складність картки для даного користувача.

Модуль контекстної обізнаності, що використовує дані з сенсорів мобільного пристрою, надає DQL-агенту інформацію про поточне оточення та активність користувача. Ця інформація дозволяє DQL-агенту адаптувати навчальну взаємодію, наприклад, запропонувати коротшу сесію, якщо користувач зайнятий, або змінити формат подачі матеріалу.

Таким чином, у системі реалізовано циклічний потік інформації: дії користувача призводять до оновлення моделей FSRS та BKT; ці оновлені моделі надають актуальні дані для RL-агентів; RL-агенти, враховуючи

також контекст, приймають рішення щодо наступних дій системи; система пропонує адаптований контент та час, що знову призводить до дій користувача.

Такий цикл адаптації є ключовою для поступового самовдосконалення системи та підвищення ефективності навчання для кожного окремого користувача. Важливо, що система може діяти не лише реактивно (за сигналом DDPG), а й проактивно, ініціюючи навчання у сприятливі моменти, що сприяє інтеграції навчального процесу в повсякденне життя користувача та відповідає сучасним тенденціям мікронавчання.

2.4.3 Гібридна модель тренування RL-агентів

Для вирішення фундаментальної дилеми між необхідністю створення узагальненої моделі, що ефективно працює для широкого кола користувачів, та прагненням до глибокої персоналізації навчального досвіду, а також для подолання проблеми «холодного старту» для нових користувачів буде використано гібридний підхід до тренування RL-агентів. Цей підхід є компромісом, що поєднує переваги узагальнення та індивідуалізації, і базується на основних концепціях МН, таких як попереднє навчання на великих масивах даних з подальшим тонким налаштуванням.

На першому етапі на великому анонімізованому наборі даних, зібраному від усіх користувачів системи, тренується потужна глобальна модель. Ця модель призначена для вивчення загальних патернів функціонування людської пам'яті, типових кривих забуття, ефективних навчальних стратегій та інших закономірностей, що є спільними для більшості людей.

Глобальна модель слугує міцним фундаментом, забезпечуючи високу якість початкової взаємодії для нових користувачів. Саме завдяки наявності

такої попередньо навченої моделі ефективно вирішується проблема «холодного старту», оскільки новий користувач одразу отримує доступ до системи, що вже володіє значним обсягом узагальнених знань про процеси навчання.

Коли новий користувач починає працювати з системою, він спочатку взаємодіє з вищезгаданою якісною глобальною моделлю. Після того, як система накопичує достатню кількість його персональних даних про взаємодію (наприклад, 100–200 відповідей на картки), для нього створюється персональна копія глобальної моделі. Ця індивідуальна копія надалі донавчається (fine-tunes) виключно на даних цього конкретного користувача.

Такий підхід дозволяє швидко та ефективно адаптувати модель до індивідуальних особливостей когнітивних процесів учня, його темпу навчання, специфічних труднощів та вподобань. При цьому зберігаються потужні узагальнюючі знання, отримані на глобальному етапі тренування. Це поєднання забезпечує як переваги потужного узагальнення глобальної моделі, так і максимальну персоналізацію та швидку адаптацію, характерну для індивідуальних моделей.

Запропонований гібридний підхід до тренування (глобальна модель + fine-tuning) є потужною та перспективною стратегією. Він ефективно вирішує проблему «холодного старту», надаючи новим користувачам якісну, узагальнену модель з самого початку їхньої взаємодії з системою. Подальше тонке налаштування на персональних даних дозволяє досягти глибокої персоналізації та швидкої адаптації до індивідуального стилю навчання кожного користувача.

Однак, існують певні нюанси та потенційні виклики, які потребують уваги при реалізації такого підходу:

– достатність даних для fine-tuning є першим викликом, хоча 100-200 взаємодій можуть бути достатніми для початкового тонкого налаштування, особливо якщо глобальна модель є міцною та добре узагальненою основою,

для складних RL-агентів, таких як DDPG та DQL, що мають велику кількість параметрів, цього обсягу даних може виявитися замало для повноцінної адаптації без ризику перенавчання (overfitting).

Перенавчання на специфічних, але потенційно нерепрезентативних початкових даних користувача може призвести до погіршення якості персоналізованої моделі. Визначення оптимального порогу кількості даних для початку fine-tuning та розробка адаптивних стратегій тонкого налаштування (наприклад, поступове «розморожування» шарів моделі або використання технік мета-навчання) є важливими напрямками для подальших досліджень;

– необхідно визначити оптимальну періодичність перетренування глобальної моделі новими анонімізованими даними від усіх користувачів. Це важливо для того, щоб глобальна модель залишалася актуальною, відображала еволюцію загальних патернів навчання та враховувала нові знання, отримані системою;

– також існує ризик катастрофічного забування під час тонкого налаштування на обмежених персональних даних, коли модель «забуде» корисні узагальнені знання, отримані на етапі тренування глобальної моделі. Це відома проблема в навчанні нейронних мереж, яка може призвести до деградації моделі. Для запобігання цьому необхідно застосовувати спеціальні техніки, такі як використання менших темпів навчання (learning rates) для fine-tuning, регулярне порівняння продуктивності персоналізованої моделі з глобальною моделлю, або більш складні методи збереження знань (наприклад, «Elastic Weight Consolidation» (EWC) або «Experience Replay» (навчання з реплеєм попереднього досвіду)). Успішне вирішення цієї проблеми є ключовим для створення по-справжньому адаптивних систем, здатних до безперервного навчання протягом усього життєвого циклу. У таблиці 2.2 наведено порівняння ключових характеристик, переваг та викликів обох етапів гібридного тренування.

Таблиця 2.2 – Порівняння етапів гібридного тренування та їх виклики

	Етап 1: Тренування глобальної моделі	Етап 2: Персоналізація через fine-tuning
Мета	Вивчення загальних патернів та вирішення «холодного старту».	Адаптація до індивідуальних особливостей учня.
Дані	Великий анонімізований набір даних від усіх користувачів.	Персональні дані конкретного користувача (наприклад, після 100-200 взаємодій).
Переваги	Міцний фундамент, висока якість початкової взаємодії.	Швидка адаптація, максимальна персоналізація, збереження узагальнюючих знань (в ідеалі).
Потенційні виклики та нюанси	Частота оновлення для підтримки актуальності.	Достатність даних для fine-tuning (ризик overfitting), ризик катастрофічного забування.
Стратегії подолання викликів	Визначення оптимальної періодичності перетренування.	Техніки запобігання перенавчанню, техніки проти катастрофічного забування (менші темпи навчання, методи збереження знань тощо).

2.4.4 Реалізація контекстної обізнаності та адаптації

Ключовою перевагою розроблюваної інтелектуальної системи навчання є її здатність глибоко адаптуватися до умов, в яких перебуває користувач під час навчального процесу. Ця адаптація виходить за межі простого налаштування інтервалів повторення чи вибору контенту на основі попередніх відповідей. Вона враховує реальний фізичний та ситуативний контекст користувача, перетворюючи систему на інтелектуального компаньйона, що ненав'язливо інтегрується в повсякденне життя.

Для реалізації контекстної обізнаності передбачається використання вбудованих сенсорів мобільного пристрою користувача. Зокрема, можуть бути задіяні акселерометр та гіроскоп для детекції руху та визначення

поточної фізичної активності (наприклад, ходьба, біг, поїздка в транспорті, стан спокою), а також мікрофон для оцінки рівня навколишнього шуму. Існуючі дослідження демонструють високу точність (до 97.85% у тестових умовах) у визначенні подібних активностей на основі даних акселерометра та гіроскопа з використанням моделей глибокого навчання. Хоча досягнення такої точності в реальних умовах експлуатації може бути пов'язане з певними викликами (різноманітність пристроїв, індивідуальні особливості рухів, енергоспоживання), технологічна база для розпізнавання контексту є достатньо зрілою.

Система повинна враховувати широкий спектр контекстуальних факторів, серед яких:

– поточна діяльність користувача, а саме чи займається користувач роботою, перебуває в транспорті, відпочиває, чи знаходиться в іншому середовищі, яке може впливати на його здатність сприймати та засвоювати інформацію;

– фізичне оточення, що включає: рівень шуму, освітлення, наявність інших людей, що можуть відволікати;

– час доби та рівень енергії користувача, наприклад, втома або, навпаки, бадьорість користувача можуть суттєво впливати на ефективність навчання. Рівень енергії може оцінюватися опосередковано через патерни взаємодії з пристроєм або системою;

– доступні ресурси та пристрої, чи використовує користувач мобільний телефон, планшет або комп'ютер; чи є стабільний доступ до Інтернету;

– емоційний стан користувача, і хоча пряме визначення емоційного стану є складною задачею, система, все ще, може враховувати його через аналіз швидкості реакцій, кількості помилок тощо, або надавати користувачеві можливість самостійно вказати свій стан.

Взагалі то, контекст, у якому відбувається навчання може охоплювати сукупність усіх обставин та умов, що оточують користувача під час

взаємодії з навчальною системою. Головна мета врахування контексту полягає в тому, щоб адаптувати спосіб подачі навчального матеріалу, його обсяг та інтенсивність таким чином, аби навчання було максимально ефективним та мінімально нав'язливим у конкретний момент часу для конкретного користувача. Наприклад, під час роботи або заняття спортом система може пропонувати лише дуже короткі навчальні сесії або менш складний матеріал, тоді як у спокійній домашній обстановці – більш глибоке занурення в тему.

Важливо відзначити, що використання деяких сенсорів, зокрема, мікрофона для оцінки рівня шуму, може викликати у користувачів питання щодо приватності. Тому реалізація модуля контекстної обізнаності повинна супроводжуватися прозорою політикою використання даних та наданням користувачеві повного контролю над тим, які сенсори та дані використовуються системою, з можливістю відключення певних функцій моніторингу.

2.4.5 Адаптивні механізми та формат взаємодії з користувачем

Адаптація системи до контексту користувача реалізується через гнучкий вибір механізмів та формату взаємодії. Інтелектуальний DQL-агент, отримуючи інформацію від модуля контекстної обізнаності, обирає найдоцільніший спосіб донесення навчального контенту.Приклади адаптації можуть бути такими:

– якщо система визначає, що користувач знаходиться в гучному середовищі (наприклад, у метро, що визначено за рівнем шуму з мікрофона та/або патерном руху з акселерометра), вона не буде пропонувати завдання, що потребують прослуховування аудіо (наприклад, вимова слів). Натомість може бути надіслано коротке текстове сповіщення з варіантами відповіді або картка для візуального опрацювання;

– якщо користувач ідентифікований як зайнятий (наприклад, під час робочого процесу або активного пересування), система може запропонувати значно коротші навчальні сесії, менш складний матеріал або завдання, що потребують мінімальної взаємодії;

– у спокійній обстановці, коли користувач має достатньо часу та уваги, система може запропонувати більш тривалі сесії, складніші завдання або контент, що передбачає глибоке занурення та активне осмислення.

Адаптація також стосується вибору оптимального «формату взаємодії». Це поняття охоплює різні способи, якими користувач може отримувати інформацію від системи та взаємодіяти з нею:

– візуальна форма подачі, а саме сприйняття інформації через зір. Це може бути читання тексту на картках, перегляд зображень, що ілюструють зміст картки, взаємодія з графічним інтерфейсом застосунку;

– аудіо формат подачі через сприйняття інформації завдяки слуху. Наприклад, прослуховування вимови іноземних слів, аудіо-пояснення до матеріалу;

– текстова/графічна взаємодія через отримання навчального контенту (карток) через системні push-сповіщення, які можуть містити текст, визначення, приклади та навіть прості інтерактивні елементи;

– взаємодія з інтерфейсом через дотик (наприклад, гортання карток, натискання кнопок для відповіді на запитання у сповіщенні).

Спосіб подачі інформації також адаптується: це може варіюватися від повноцінного, багатофункціонального інтерфейсу застосунку до коротких, стислих повідомлень у фоновому режимі (наприклад, через сповіщення). Система може динамічно вирішувати, чи показувати «повну інформацію» по картці (з усіма деталями, прикладами, зображеннями) чи «стислу інформацію» (лише ключове слово та його переклад/визначення).

У таблиці 2.3 наведено приклади зв'язку між сенсорними даними, контекстуальними факторами та можливими адаптаціями з боку DQL-агента.

Таблиця 2.3 – Сенсори, контекстуальні фактори та приклади адаптації

Тип сенсора або джерело даних	Збирані дані / Визначуваний фактор	Приклад адаптації DQL-агентом
Акселерометр, Гіроскоп	Вид фізичної активності (ходьба, біг, поїздка в транспорті, спокій)	Зміна тривалості сесії, складності завдань; вибір формату взаємодії (напр., не пропонувати введення тексту під час бігу).
Мікрофон	Рівень навколишнього шуму	Уникнення аудіо-завдань у гучному середовищі; перевага текстовим сповіщенням.
Системний час	Час доби	Адаптація інтенсивності навчання (наприклад, легші завдання ввечері або вночі).
Тип сенсора або джерело даних	Збирані дані / Визначуваний фактор	Приклад адаптації DQL-агентом
Патерни взаємодії	Рівень втоми/енергії користувача (як мінімум, на основі швидкості реакцій, помилок)	Пропозиція коротших перерв; зміна типу контенту на менш вимогливий; рекомендація відкласти навчання.
Календар/Розклад (з дозволу)	Поточна діяльність (робота, зустріч, відпочинок)	Інтеграція навчання: короткі завдання під час запланованих перерв на роботі, довші сесії під час вільного часу для відпочинку.

Таким чином, ключовою перевагою розроблюваної системи є її здатність глибоко адаптуватися до умов, в яких перебуває користувач. Це передбачає використання сенсорів мобільного пристрою для визначення контексту. Наприклад, якщо система визначає, що користувач знаходиться в гучному середовищі, вона не буде пропонувати аудіо-завдання, а натомість може надіслати коротке текстове сповіщення. Така гнучкість

дозволяє системі не просто реагувати на поточні умови, а знаходити оптимальні «навчальні вікна» у повсякденному розкладі користувача.

Ключова мета такої глибокої адаптації – перетворити процес навчання з окремої, запланованої діяльності на органічну частину життя людини. Рішення, прийняті DQL-агентом щодо формату та інтенсивності, стають основою для циклу зворотного зв'язку. Ефективність кожної взаємодії аналізується, що дозволяє системі постійно вдосконалювати свої стратегії адаптації. Тож, система не лише навчає, але й навчається сама, оптимізуючи взаємодію на основі унікального стилю життя користувача.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ЕЛЕМЕНТІВ СИСТЕМИ АДАПТИВНОГО ФОНОВОГО ВИВЧЕННЯ МОВ

3.1 Обґрунтування вибору технологій

3.1.1 Загальна архітектура та технології

Уся подальша архітектура спроєктована з урахуванням ключових поставленії задач для реалізації цього проєкту, а саме:

– забезпечення адаптивності, як візуального інтерфейсу, так і процесу навчання;

– підтримка фонової роботи вебзастосунку;

– можливість інтеграції різнорідних компонентів, написаних на оптимальних для реалізації задач мовах програмування.

Основою для досягнення цих цілей став вибір мікросервісної архітектури, яка дозволяє реалізувати систему у вигляді набору незалежних служб, що взаємодіють між собою через чітко визначені програмні інтерфейси (API). Такий підхід дозволяє отримати наступні переваги при реалізації:

– кожен мікросервіс може бути реалізований з використанням мови програмування, фреймворку та бази даних, що найкраще відповідають його специфічним задачам. Це дозволило обрати «Python» (далі пайтон) для реалізації інтелектуального агента, «.NET» для фонових задач та «NodeJS» з «SvelteKit» для обробки API-запитів та повідомлень в месседж брокеру «Apache Kafka» [17] (далі кафка);

– незалежне масштабування, яке є необхідним через те що навантаження на різні частини системи може суттєво відрізнятися, а мікросервісна архітектура дозволяє масштабувати окремі служби незалежно одна від одної, оптимізуючи, таким чином, використання ресурсів;

– підвищена відмовостійкість завдяки ізоляції різних сервісів, таким чином, збій в одному з них не призведе до повної відмови всієї системи. Інші компоненти зможуть продовжувати свою роботу, що є критично важливим для забезпечення стабільного користувацького досвіду;

– гнучкість розгортання та оновлення завдяки тому, що кожен сервіс може оновлюватися та розгортатися незалежно, що значно прискорює цикл розробки та дозволяє оперативно впроваджувати нові функції чи виправляти помилки без необхідності перебудови всього застосунку.

Для забезпечення надійної та асинхронної комунікації між цими незалежними сервісами було обрано вже зазначену вище кафку. Вона виконує роль центральної нервової системи застосунку, реалізуючи патерн «видавець-підписник». Сервіси-продюсери (наприклад, сервіс обробки даних) публікують події у відповідні топіки кафки, а сервіси-консюмери (наприклад, сервіс сповіщень) підписуються на ці топіки та реагують на події асинхронно. Такий підхід повністю усуває необхідність у прямих синхронних викликах між ними, що підвищує загальну надійність та продуктивність системи.

Для управління інфраструктурними компонентами, зокрема, для розгортання та адміністрування кафки, використовується система контейнеризації «Docker» [19] (далі докер). Вона дозволяє інкапсулювати сервіс та всі його залежності у контейнери, які в свою чергу, гарантують ідентичність оточення на етапах розробки, що значно полегшує тестування та експлуатацію, спрощуючи процес розгортання та адміністрування системи.

Клієнтська частина системи, з якою безпосередньо взаємодіє користувач, реалізована як односторінковий застосунок «Single Page Application» (SPA) з використанням фреймворку «Svelte» та його надбудови «SvelteKit» [20]. Svelte був обраний через його простоту використання, оптимізацію, як програмну, так і SEO та динамічні компоненти у купі з можливістю програмування на мові «TypeScript» (далі TS) та

використовувати кастомні таблиці стилей. SvelteKit, у свою чергу, надає потужні інструменти для маршрутизації, рендерингу на стороні сервера та управління станом, що дозволило створити швидкий та інтерактивний вебзастосунок з простою моделью спілкування між бекенд і фронтенд частинами застосунку. Серверна частина SvelteKit працює на платформі NodeJS, яка також виконує роль API-шлюзу, обробляючи запити від клієнта та координуючи їх взаємодію з внутрішніми мікросервісами через кафку.

3.1.2 Обґрунтування вибору технологій для ключових підсистем

Для розробки інтелектуального агента, що є ядром адаптивності системи, було обрано мову програмування пайтон. Цей вибір обґрунтований кількома ключовими факторами.

По-перше, Python фактично є стандартом у галузі штучного інтелекту, машинного навчання та аналізу даних. Він має надзвичайно багату екосистему спеціалізованих бібліотек, таких як «TensorFlow», «PyTorch», «Scikit-learn», «TensorBoard» та «Pandas», які суттєво спрощують реалізацію складних алгоритмів. Для цього застосунку це дозволило ефективно реалізувати та тестувати алгоритми навчання з підкріпленням по типу «Deep Q-Learning» (далі DQL) та «Deep Deterministic Policy Gradient» (далі DDPG), які слугують для динамічної адаптації розкладу повторень.

По-друге, пайтон є потужним інструментом для обробки природної мови (NLP) або парсингу вебу, що було використано для інтеграції із зовнішніми лінгвістичними сервісами, такими як Cambridge Dictionary для отримання визначень та прикладів вживання слів, і DeepL API для їх перекладу. пайтон-сервіс відповідає за збір, аналіз та обробку даних про взаємодію користувача з системою, на основі яких інтелектуальний агент приймає рішення щодо подальшої стратегії навчання.

Підсистема, що відповідає за реалізацію механізму інтервальних повторень та надсилання фонових сповіщень, розроблена на платформі

.NET з використанням мови C# та патрену «Model View Controller» (далі MVC). Цей вибір зумовлений високою продуктивністю, надійністю та мультипоточністю платформи .NET, що робить її ідеальним кандидатом для створення довготривалих фонових служб (background services). Для взаємодії з операційною системою Windows з метою показу нативних спливаючих сповіщень використовується інтеграція з «Universal Windows Platform» (UWP) та «Windows Runtime» (WinRT), де WinRT API – це сучасна платформа додатків та програмний інтерфейс від Microsoft, що надає стандартизований доступ до функцій операційної системи. Це дозволяє .NET-застосунку надсилати користувачам інтерактивні картки, або ж «Content-Rich» сповіщення, як до цього посилається документація про сповіщення, для повторення у визначений системою час, навіть коли основний вебзастосунок неактивний. Такий підхід є ключовою вимогою для реалізації концепції фонового навчання, оскільки забезпечує проактивну взаємодію з користувачем без необхідності його постійної присутності в застосунку.

Для збереження даних про прогрес користувачів, їхні відповіді та налаштування було обрано вбудовану систему управління базами даних SQLite. Вона є легкою, серверною, файловою СУБД, що не потребує окремого процесу адміністрування та ідеально підходить для локального зберігання даних застосунку. Взаємодія з базою даних SQLite з боку .NET-сервісу здійснюється за допомогою технології Entity Framework Core (EF Core). Це сучасний, об'єктно-орієнтований мапер (ORM), який дозволяє розробникам працювати з базою даних, використовуючи C#, і абстрагуватися від написання SQL-запитів вручну. Такий підхід прискорює розробку, підвищує надійність коду та спрощує роботу зі схемою даних.

Таким чином, комбінація пайтону для інтелектуальних завдань та .NET для надійних фонових процесів і роботи з даними, а також Svelte для фронтенду дозволяє максимально ефективно використовувати сильні сторони кожної технології для побудови потужної та гнучкої системи.

3.2 Опис ключових реалізованих компонентів

3.2.1 Побудова детальної архітектури системи

Таким чином, базуючись на описаних у 3.1 технологіях можна побудувати наступну архітектуру:

а) налаштування ядра сервера(ів);

1) операційна система – «Windows» 10;

2) «SvelteKit» з використанням «NodeJS» 18.x;

3) «Python» 3.11 – обробка даних та інтеграція із зовнішніми API;

4) «C# .NET» 9.0 – для обробки карток та відповідей користувача на них;

5) «MVC» – для спрощення процесу розробки та можливої імплементації візуального інтерфейсу для управління збереженною інформацією;

б) стек технологій;

1) фронтенд;

– «Svelte»;

– «WebSocket» – для оновлень у реальному часі;

2) бекенд;

– «NodeJS» – API та інтеграція з «Apache Kafka»;

– «SvelteKit» – для обробки повідомлень від кафки;

– «Python» – парсинг даних, зовнішні API (DeepL) або парсинг вебу (Cambridge Dictionary), інтелектуальні агенти з використанням «Experience Replay» та інші необхідні алгоритми;

– C# .NET MVC – обробка карток;

– «Entity Framework Core» (далі EF Core) – зберігання відповідей користувачів;

3) алгоритми;

– «Deep Q-Learning» (DQL) – для адаптивного розрахунку інтервалів з фіксованими проміжками часу (..., 1, 3, 7 днів, ...);

– «Deep Deterministic Policy Gradient» (DDPG) – для адаптивного розрахунку інтервалів з будь-якими (але всеж таки наближеними до результатів виконання алгоритмів для інтелектуального повторення) проміжками часу (від встановленого мінімуму (1 хвилина) до максимуму (30 днів));

– «Experience Replay» – використовуєть під час роботи інтелектуального агенту для покращення їх ефективності;

– алгоритми для інтервального повторення та «Bayesian Knowledge Tracing» (BKT) – використовуються для покращення роботи інтелектуального агенту через розрахунки ймовірності моделювання знання певних карток;

в) база даних;

1) локальна «SQLite» з «EF Core» – для зберігання відповідей користувача;

г) потокова передача подій;

1) «Apache Kafka» – комунікація між сервісами;

д) планування відправок повідомлень;

1) «UWP» у купі з «WinRT» – планування завдань для системних сповіщень (тостів);

е) моніторинг;

1) «Confluent Control Center» – показники стану та продуктивності системи, а також відстеження подій від кафки.

Модель розгортання (рисунок 3.1) – показує, як і де фізично чи віртуально розміщуються компоненти системи. У цьому проєкті це означає, що майже всі частини (фронтенд, сервіси, база даних) працюють локально на одному комп'ютері. Важливою деталлю є використання докер для

запуску кафки, що дозволяє ізолювати цей сервіс від основної системи. Таким чином, вона включає в себе:

– фронтенд через локально розміщену статична збірку Svelte;

– внутрішні служби є локально розгорнутими;

– база даних – локальний Sqlite;

– потокова передача подій через докеризовану кафку;

– моніторинг проходить через центр управління Confluent збирає метрики та дашборди для візуалізації.

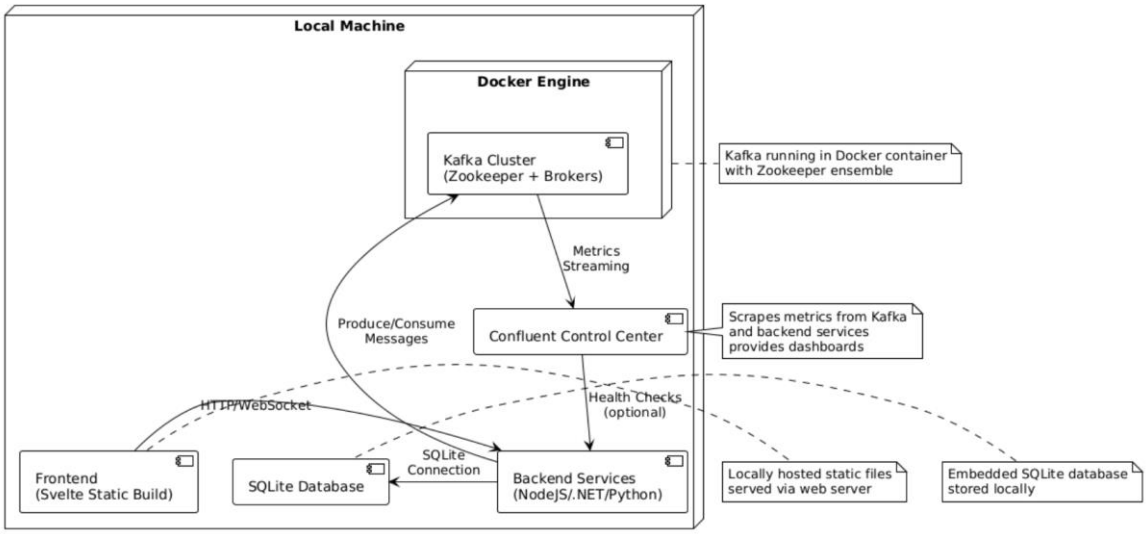

Рисунок 3.1 – Модель розгортання

Комунікаційна модель застосунку (рисунок 3.2), являє собою опис того, які протоколи та технології використовуються для обміну даними між різними частинами застосунку. Тут система використовує REST для прямого зв'язку «клієнт-сервер», кафка для надійної асинхронної взаємодії між внутрішніми сервісами та WebSockets для миттєвої відправки оновлень на фронтенд. Таким чином, модель включає в себе:

– «REST» – для зв'язку фронтенду з бекендом та для зв'язку з бекендом .NET для запуску/зупинки процесу навчання (зв'язок JSON);

– використання кафки для обробки всіх асинхронних подій за межами шлюзу API;

– використання WebSockets для оновлення фронтенду в режимі реального часу.

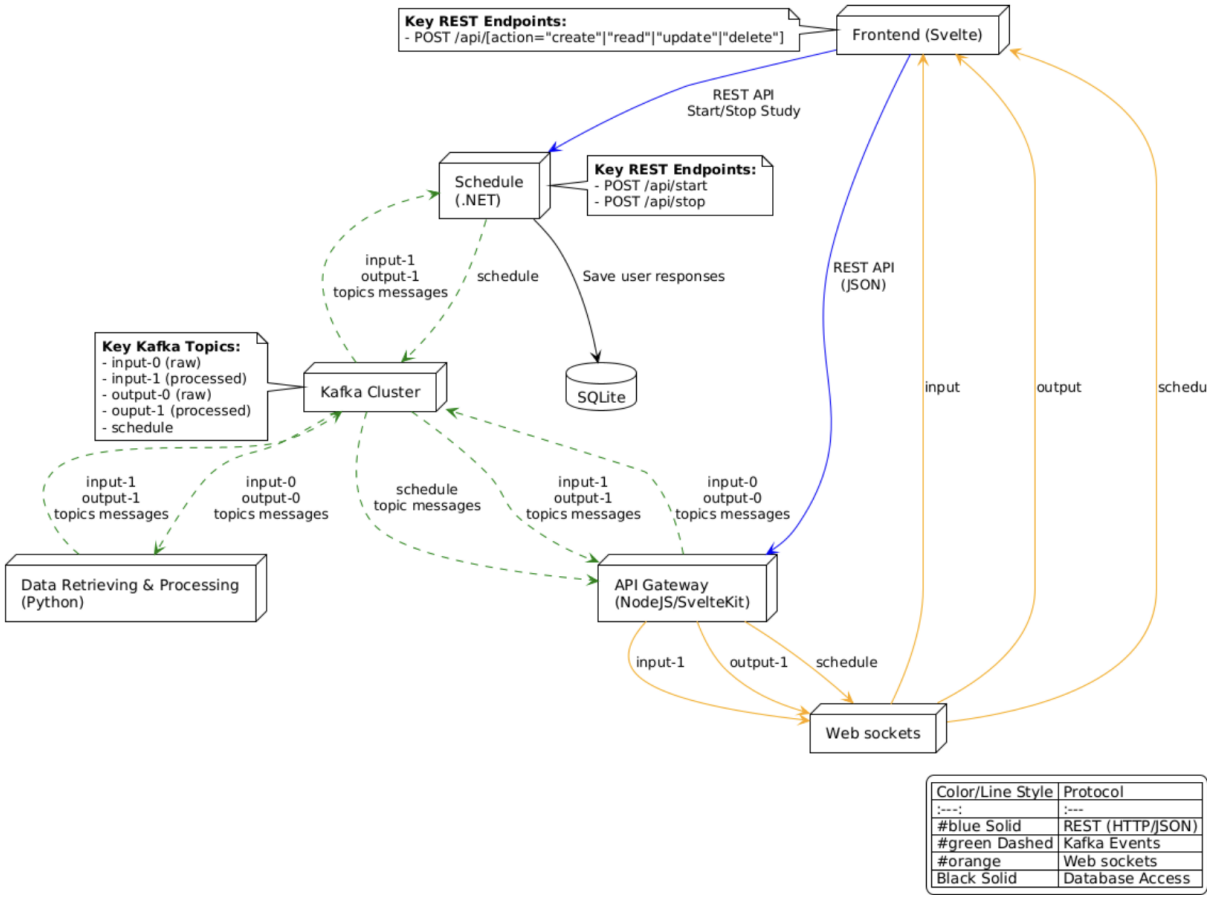

Рисунок 3.2 – Комунікаційна модель застосунку

Інтеграція із зовнішніми сервісами:

– «Cambridge Dictionary» – отримання та парсинг визначень та прикладів;

– «DeepL» API – сервіс перекладу.

Передумови та конфігурація середовища розробника:

– встановлення NodeJS 18.x, Python 3.11, .NET 9.0 SDK.

– «Docker Desktop» – для локальної оркестрації сервісів.

3.2.2 Інтерфейс вебзастосунку

Для пошуку, перекладу та контролю карток за наданою користувачем інформацією, створено компонент «Textbox» (рисунок 3.3). У данному випадку наступне речення не має ніяких співпадінь у словниках тож отримано тільки переклад.

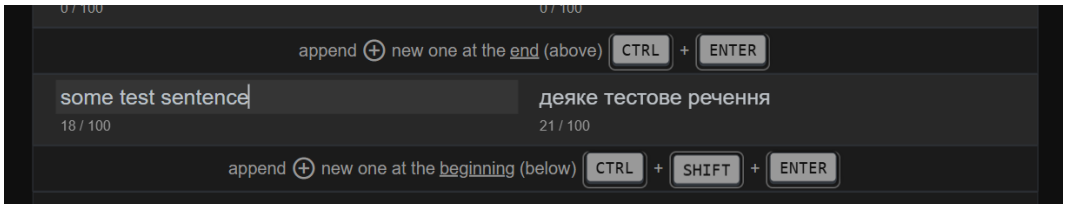

Рисунок 3.3 – Компонент «Textbox»

У наступному прикладі (рисунок 3.4) вже можна побачити, що для слова «word» були знайдені співпадіння тож отримано відповідну інформацію разом із перекладом.

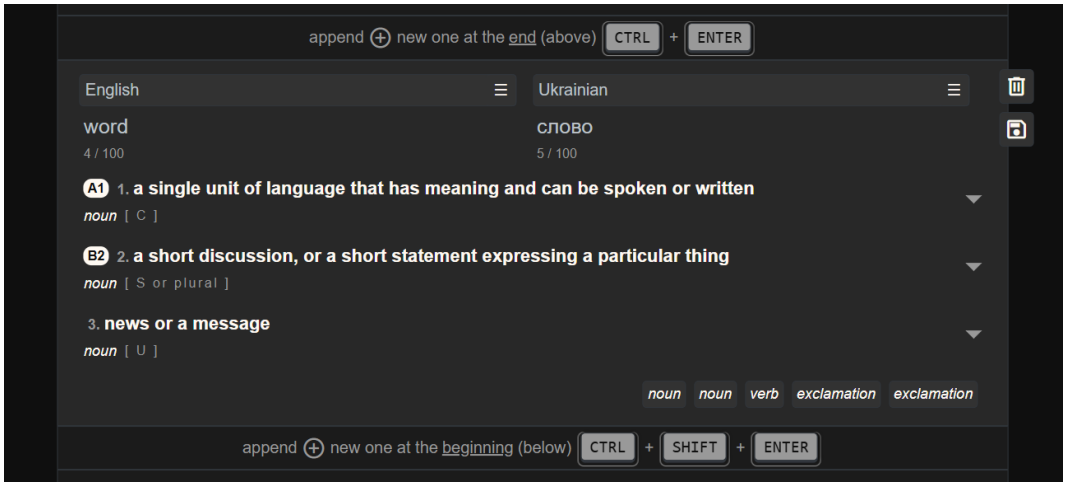

Рисунок 3.4 – Компонент «Textbox» з додатковою інформацією зі словника

Для інтерактивності та більшої наглядності були додані такі деталі як анімація натискання кнопок у відповідних інформаційних панелях, якщо

поля відповідаюсь певним вимогам (рисунок 3.5) та операція додавання нової картки може бути виконано то як можна побачити при натисканні «CTRL» + «SHIFT» відповідні кнопки теж будуть візуально натискатися (рисунок 3.6).

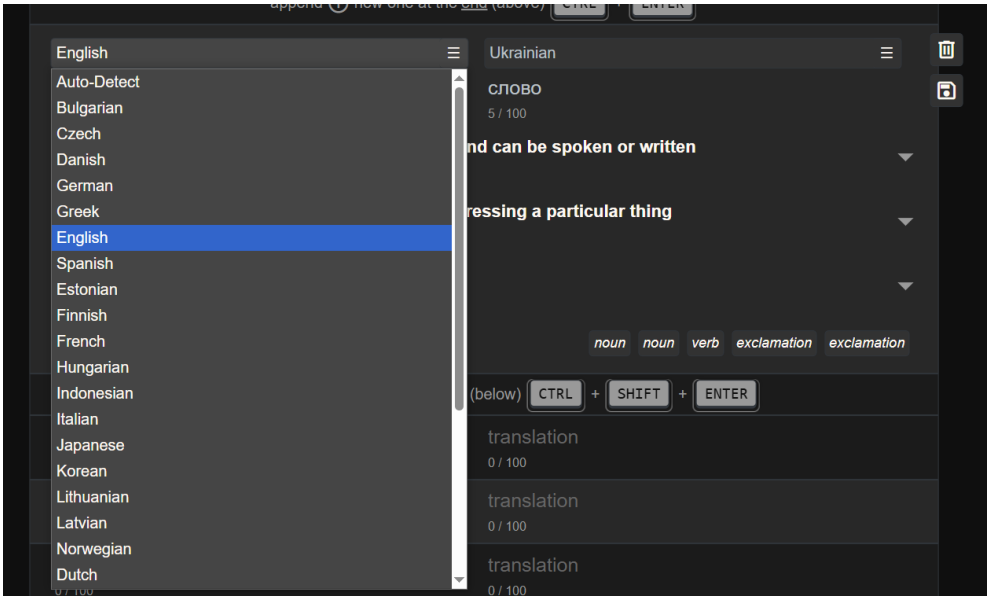

Рисунок 3.5 – Перелік доступних мов

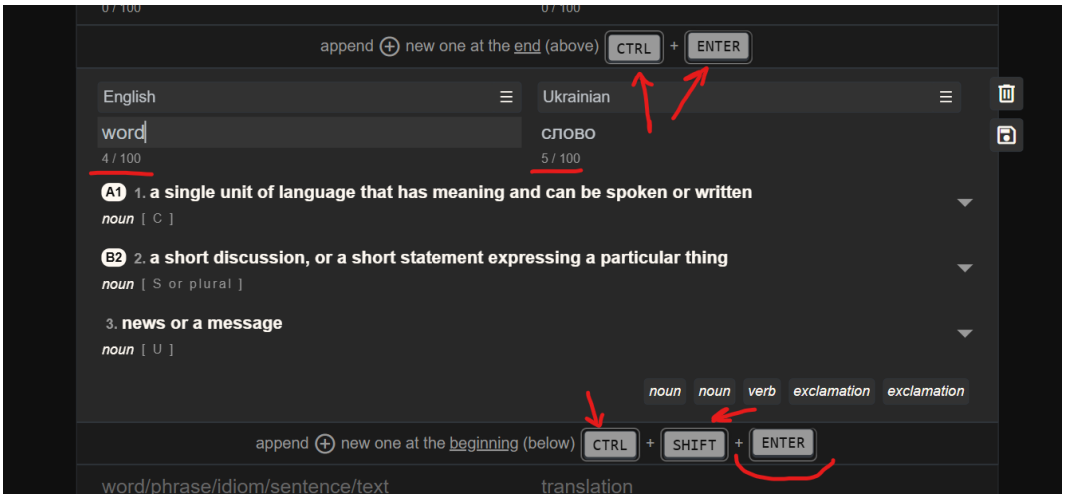

Рисунок 3.6 – Інтерактивні елементи

Для покращення юзер експіріенсу також додані відповідні зміни у кольорах деяких кнопок при наведенні на них (рисунок 3.7).

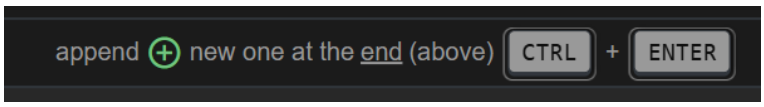

Рисунок 3.7 – Додані кнопки для управління картками

Для того щоб забезпечити гнучке управління навчальним процесом, а також представити користувачеві деталізовані картки слів створено компонент «Schedule» (рисунок 3.8), який є основою для створення та управління списком елементів застосунку, а також надає вичерпну інформацію, яка включає переклад, визначення, вимову, граматичні особливості, рівень складності та контекстуальні приклади.

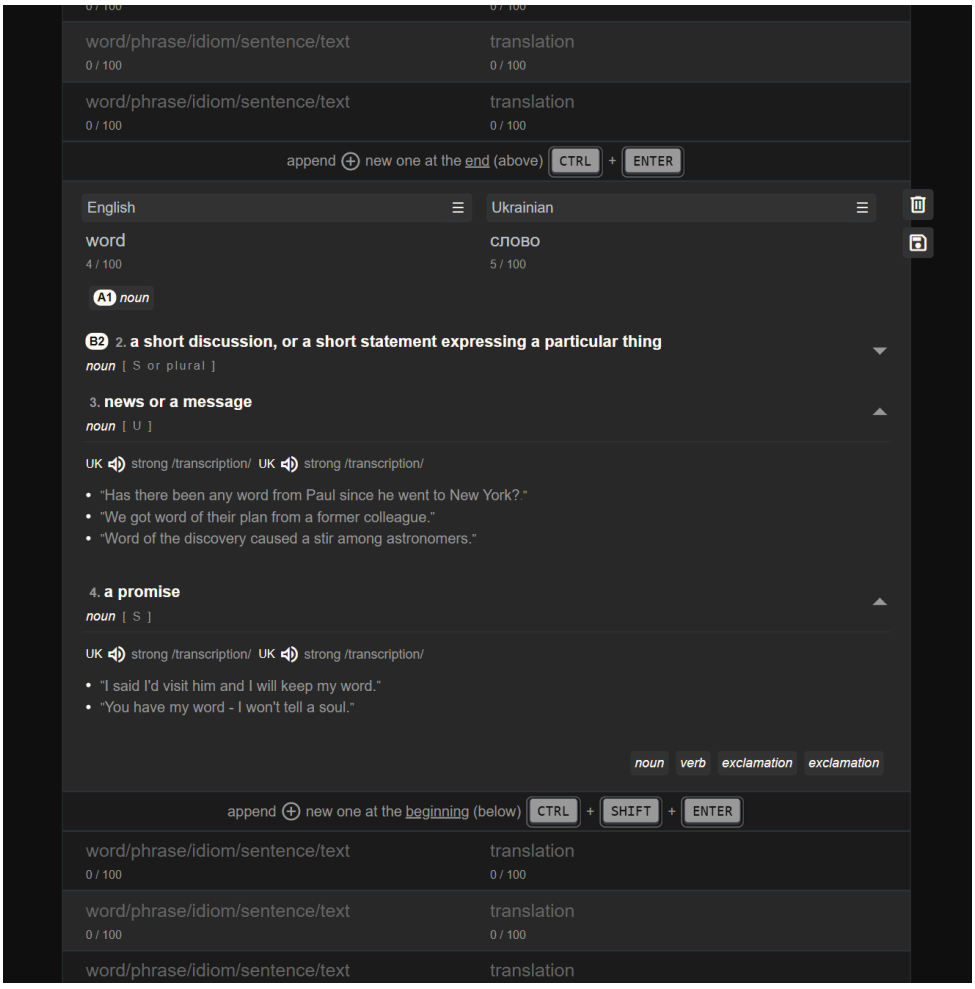

Рисунок 3.8 – Компонент «Schedule» та розгорнути подробиці по слову «word»

Такий підхід сприяє не механічному запам'ятовуванню, а реальному розумінню та засвоєнню іноземних слів. Елементи для вивчення розташовані у вигляді вертикальної стрічки. Користувач може прокручувати її вгору та вниз, бачачи картки, які він додав. Вище і нижче активної картки «word» знаходяться порожні блоки з плейсхолдером «word/phrase/idiom/sentence/text» та порожній переклад. Це шаблони, які показують, куди можна додати новий навчальний елемент. Таким чином, компонент «Schedule» – це не просто статичний список, а динамічний робочий простір, де користувач формує свій персональний розклад вивчення.

Далі потрібен зручний та компактний спосіб перемикатися між частинами мови для кожної з карток, таким чином, створено компонент «Shortcuts» (рисунки 3.9, 3.10, 3.11). В його межах корстувач може швидко пролистувати перелік усіх частин мови та бачити лише три детальні записи одночасно.

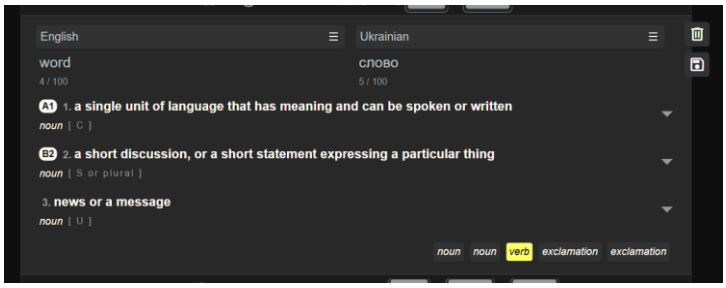

Рисунок 3.9 – Компонент «Shortcuts» на початку переліку

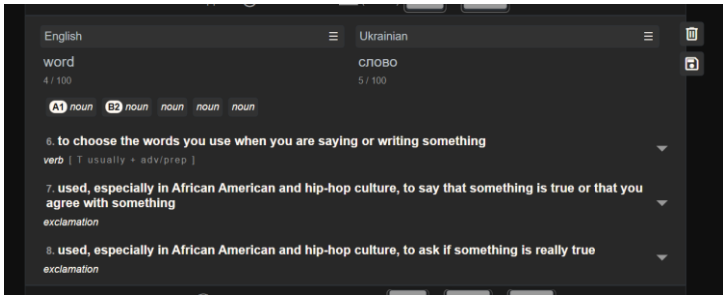

Рисунок 3.10 – Компонент «Shortcuts» в кінці переліку

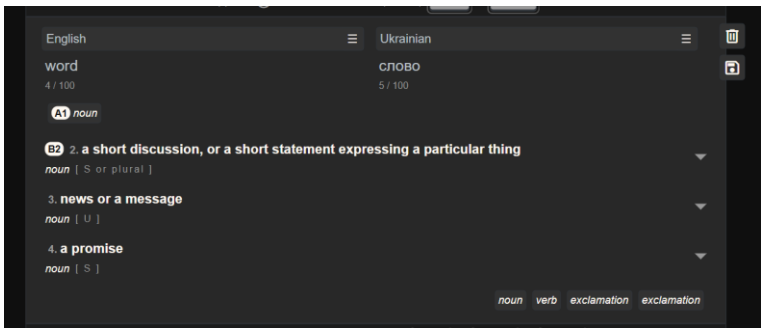

Рисунок 3.11 – Компонент «Shortcuts» в середині переліку

Був використаний бекенд, який здатний працювати з «rich-content» «system toast» повідомленнями та визуалом системи віндовс, таким чином, було створено наступну картку (рисунок 3.12), яка надсилається користувачу під час процесу навчання. Програмний код реалізації, поданий у лістингу 3.1.

Лістинг 3.1 – Програмний код тестовогосистемного «rich-content» тосту

```
    var word = "word";
    var definition = "a short discussion, or a short statement expressing a particular thing";
    var example = "We need to have a word (= to talk) about something.";

    new ToastContentBuilder()
        .AddHeader("WordCard", $"{word.ApplyCase(LetterCasing.Sentence)}", "action=openApp")

.AddText($"{definition.ApplyCase(LetterCasing.Sentence)}", AdaptiveTextStyle.Body)

.AddText($"\"{example.ApplyCase(LetterCasing.Sentence)}\"", AdaptiveTextStyle.Caption)

        //Buttons with arguments
        .AddButton(new ToastButton()
            .SetContent("Mark as Known")
            .AddArgument("action", "known")
            .AddArgument("word", word)
```

Продовження лістингу 3.1

```
            .SetBackgroundActivation())

        .AddButton(new ToastButton()
            .SetContent("Forgot")
            .AddArgument("action", "forgot")
            .AddArgument("word", word)
            .SetBackgroundActivation())

        .AddButton(new ToastButton()
            .SetContent("Skip for Now")
            .AddArgument("action", "skip")
            .AddArgument("word", word)
            .SetBackgroundActivation())

        .SetToastScenario(ToastScenario.Reminder)     //   Keep
visible longer
        .Show(toast =>
        {
            toast.ExpirationTime                                =
DateTime.Now.AddSeconds(15);
            toast.Tag = $"WordCard_{word}";
            toast.Group = "Study";
        });
```

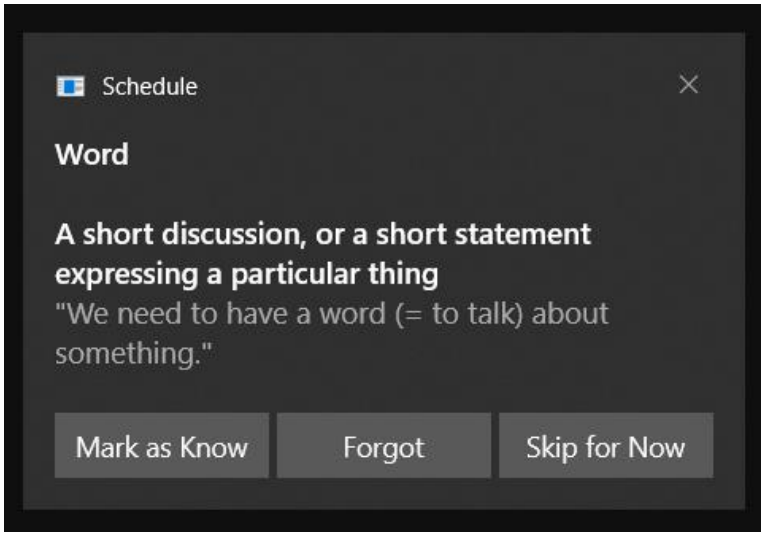

Рисунок 3.12 – Приклад системного «Toast notification»

Таким чином, було представлено основні реалізовані елементи інтерфейсу, приклади та результати використання можна побачити у 3.3.

3.2.3 Інтелектуальний агент

ІА є центральним компонентом розроблюваної системи адаптивного фонового вивчення іноземних мов, відповідальним за динамічну персоналізацію навчального процесу. Як було детально обґрунтовано у розділі 2.4 даної роботи, для реалізації функції прогнозування оптимальних неперервних інтервалів повторення навчального матеріалу було обрано алгоритм DDPG. Цей вибір зумовлений його доведеною ефективністю у задачах з неперервним простором дій.

Алгоритм DDPG працює як актор-критик. В основі його архітектури лежать дві взаємодіючі нейронні мережі: актор (actor) та критик (critic). Актор відповідає за формування політики поведінки агента, тобто, за вибір конкретної дії на основі поточного стану середовища. Критик, у свою чергу, оцінює обрану актором дію, прогнозуючи її довгострокову цінність (Q-значення). DDPG є «off-policy» алгоритмом, що означає, що він може навчатися на даних, зібраних за допомогою політики, відмінної від тієї, що оптимізується, що підвищує ефективність використання досвіду.

Для забезпечення стабільності процесу навчання, що є типовою проблемою для алгоритмів глибокого навчання з підкріпленням, DDPG використовує два важливі механізми:

– буфер досвіду (Experience Replay Buffer) – це структура даних, де зберігаються попередні взаємодії агента з середовищем у вигляді кортежів (поточний стан, виконана дія, отримана винагорода, наступний стан). Під час навчання агент не використовує послідовні дані безпосередньо, а випадковим чином вибирає міні-батчі з цього буфера. Такий підхід значно стабілізує процес навчання та підвищує ефективність використання зібраних даних;

– цільові мережі (Target Networks) – є копіями основних мереж актора та критика, але їхні ваги оновлюються значно повільніше. Вони використовуються для подальшої стабілізації навчання DDPG.

Для ефективного прогнозування персоналізованих інтервалів повторень, модель DDPG-агента визначається через специфікацію простору станів, простору дій та функції винагороди:

– простір станів (State Space), відповідно до таблиці 3.1;

– простір дій (Action Space) – це прогнозований оптимальний інтервал до наступного повторення навчального елемента. Цей інтервал є неперервним значенням, що може вимірюватися у днях, годинах або хвилинах; Для забезпечення реалістичності прогнозів, необхідно визначити практичні межі для інтервалів (наприклад, від 1 хвилини до 180 днів);

– штрафна функція винагороди. Замість традиційної функції винагороди, що прагне максимізувати позитивні результати (наприклад, +1 за правильну відповідь), у даній роботі пропонується використовувати штрафну функцію винагороди. В такій функції ідеальним станом є нульова винагорода (що відповідає бездоганному пригадуванню матеріалу в оптимально розрахований момент), а будь-які відхилення від цього ідеалу призводять до негативних штрафів. Такий підхід дозволяє інтуїтивніше інтерпретувати «витрати», пов'язані з неоптимальним плануванням інтервалів повторення.

Структура штрафів передбачає наступні основні компоненти:

– ідеальний стан (ввинагорода $\approx$ 0), коли користувач правильно пригадує навчальний елемент, коли його запитують через інтервал, прогнозований DDPG-агентом, і цей інтервал відповідає «ідеальній» ймовірності пригадування (наприклад, 90%, згідно з FSRS);

– штраф за неправильне пригадування (або ж забування), коли призначається великий негативний штраф (наприклад, –1.0), якщо користувач забуває елемент. Величина штрафу може додатково масштабуватися залежно від того, наскільки простроченим був елемент (на основі прогнозу BKT/FSRS щодо падіння ймовірності пригадування нижче критичного рівня);

– штраф за занадто раннє/неефективне пригадування – призначається менший негативний штраф, якщо користувач успішно пригадує елемент, але запропонований інтервал був значно коротшим, ніж міг би бути для підтримки цільової ймовірності пригадування (згідно з FSRS/BKT). Це штрафує неефективне планування, яке не максимізує довжину інтервалу. Штраф може бути пропорційним тому, наскільки рано було зроблено повторення відносно оптимального інтервалу, запропонованого FSRS;

– опціональний штраф за зусилля/час користувача. Може бути введений невеликий постійний негативний штраф за кожне повторення, щоб стимулювати агента до вибору довших інтервалів, за умови збереження високого рівня пригадування.

Така структура штрафної функції має важливе значення, оскільки вона неявно навчає DDPG-агента ключовим принципам ефективного запам'ятовування, а не просто максимізації кількості правильних відповідей. Традиційні алгоритми інтервального повторення, такі як SM2 або FSRS, базуються на поступовому збільшенні інтервалів у міру зміцнення пам'яті, прагнучи досягти цільової ймовірності пригадування. Проста функція винагороди (наприклад, +1 за правильну відповідь) може призвести до того, що RL-агент навчиться показувати елементи дуже часто, що є неефективним для довгострокового запам'ятовування.

Таблиця 3.1 – Компоненти простору станів DDPG-агента

Компонент стану	Нормалізований діапазон	Джерело/Обґрунтування
1	2	3
Ймовірність пригадування (BKT)	[0, 1]	Вихід моделі BKT; оцінка поточного рівня знання елемента.
Стабільність пам'яті (S) (FSRS)	[0, 365]	Вихід моделі FSRS; характеристика швидкості забування.

Продовження таблиці 3.1

1	2	3
Складність елемента (D) (FSRS)	[1, 10]	Вихід моделі FSRS; внутрішня складність елемента.
Час з останнього повторення (нормалізований)	[0, 1]	Час, що минув з моменту останнього повторення елемента.
Історія попередніх інтервалів (агрегована)	[0, 1]	Кількість успішних/неуспішних спроб, середній попередній інтервал.
Поточна активність користувача (категорійна)	[0, N_activities – 1]	Класифікована активність (ходьба, стан спокою тощо) з сенсорів.
Рівень навколишнього шуму (нормалізований)	[0, 1]	Нормалізований рівень шуму з мікрофона.
К-сть елементів у поточній сесії (норм.)	[0, 1]	Кількість елементів, повторених у поточній навчальній сесії.
Час у поточній сесії (норм.)	[0, 1]	Загальний час, проведений користувачем у поточній сесії.
Кількість елементів у сесії	[0, 50]	Кількість елементів на під час тренувальної сесії

Одними з важливіших показників на рисунку 3.13 є «ep_rew_mean», який відображає середню винагороду за епізод, через реалізовану, показну, штрафну функцію винагороди. А також, «ep_len_mean», яка зображує середню довжину епізоду, яка є важливою через те, що ІА повинен розраховувати інтервали наперед, а не вчитися на незалежних (одиничних) повтореннях.

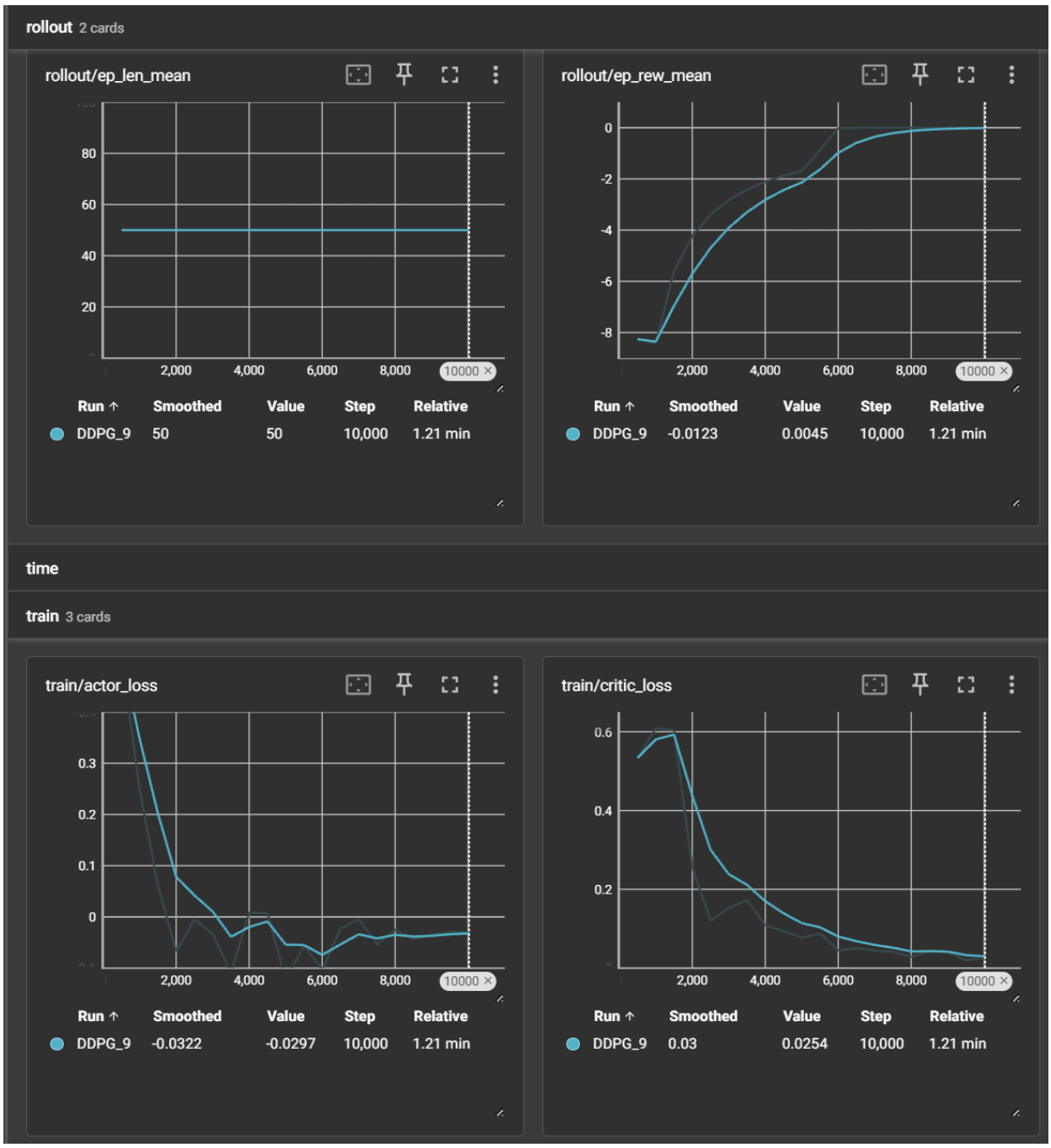

Рисунок 3.13 – Ключові метрики процесу тренування DDPG-агента

3.3 Аналіз працездатності та ефективності

Даний розділ присвячений аналізу поточного стану розробленого прототипу системи (наведена в додатку Б) адаптивного фонового вивчення іноземних мов. Важливо зазначити, що повна програмна реалізація інтелектуальних агентів (DDPG та DQL), їх тренування та інтеграція з

моделями FSRS/BKT на момент написання даної роботи ще тривають, як вказано у розділі 3.2.3 та висновках. Відповідно, аналіз зосереджується на функціональності вже реалізованих компонентів користувацького інтерфейсу та базових механізмів.

Аналіз прототипу показує реалізацію кількох ключових функціональних блоків, що закладають основу для майбутньої інтеграції інтелектуальних агентів:

– взаємодія з навчальними картками та їх відображення через компонент «Textbox» для введення користувачем слів або фраз та відображення їх перекладу (рисунки 3.3, 3.4). Система здатна отримувати та показувати додаткову інформацію, що надходить зі сторонніх сервісів, як це зазначено у вимогах та описі системи (рисунки 3.4, 3.14). Компонент «Schedule» (рисунки 3.15) демонструє можливість відображення списку навчальних карток. Ці елементи є фундаментальними, оскільки саме цими картками та пов'язаною з ними інформацією буде оперувати інтелектуальний агент при формуванні простору станів та виборі стратегії навчання;

– механізм фонових сповіщень для інтервального повторення через систему надсилання системних «Toast notification» з навчальним контентом (рисунки 3.14, 3.20, 3.21). Ці сповіщення містять інформацію для повторення, його визначення та приклад використання. Важливою є наявність інтерактивних кнопок («Mark as Known», «Forgot», «Skip for Now»), які дозволяють користувачеві надати зворотний зв'язок щодо свого рівня знання картки. Цей зворотний зв'язок також впливає на розрахунки винагороди для DDPG-агента та оновлюватися параметри моделей BKT/FSRS. Наявність динамічних лічильників часу до наступного сповіщення (рисунок 3.20) існує для подальшої реалізації адаптивних інтервалів, якими в майбутньому керуватиме DDPG-агент;

– інтерактивні елементи та налаштування спрямовані на покращення користувацького досвіду, таких як візуальна реакція кнопок на

натискання (рисунок 3.6), зміна їхнього вигляду при наведенні курсора (рисунки 3.7, 3.8, 3.9), а також кнопки для запуску та зупинки процесу навчання (рисунки 3.16, 3.17). Ці елементи, зокрема, кнопки керування навчанням, безпосередньо впливатимуть на активацію та деактивацію ІА.

Реалізований прототип, таким чином, створює необхідну інфраструктуру для взаємодії користувача з системою та для функціонування майбутнього інтелектуального агента. Хоча сам ІА, відповідальний за адаптивне планування, ще перебуває на стадії розробки, наявні компоненти користувацького інтерфейсу (відображення карток, отримання детальної інформації, механізм сповіщень з кнопками відповіді) забезпечують необхідні канали для збору даних (формування станів для агента, фіксація дій користувача та результатів взаємодії) та для надання користувачеві рекомендацій, згенерованих агентом (час та зміст навчальних сповіщень). Кнопки відповіді у сповіщеннях («Mark as Known», «Forgot», «Skip for Now») є методом для отримання результату взаємодії, на основі якого буде розраховуватися винагорода для DDPG-агента. Отже, хоча система та ІА ще формуються, її основа (користувацький інтерфейс та базові механізми) вже мають необхідні механізми для збору даних та спілкування з корисувачем.

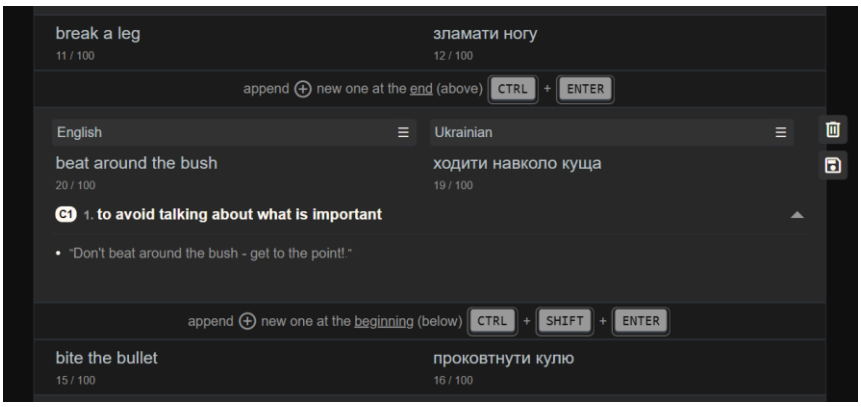

Рисунок 3.14 – Результати запиту до словника на прикладі ідіоми «beat around the bush»

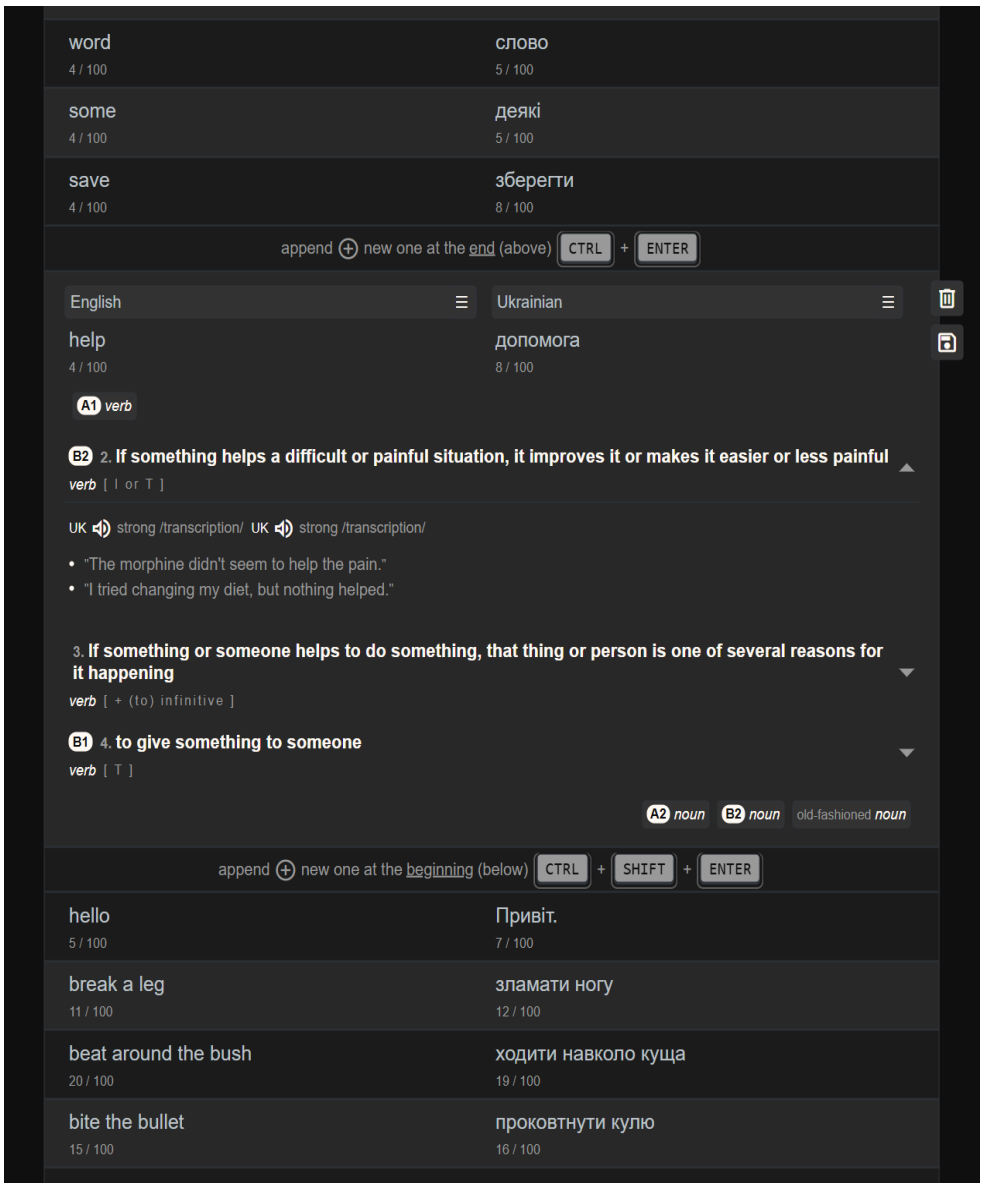

Рисунок 3.15 – Загальний вигляд компоненту «Schedule»

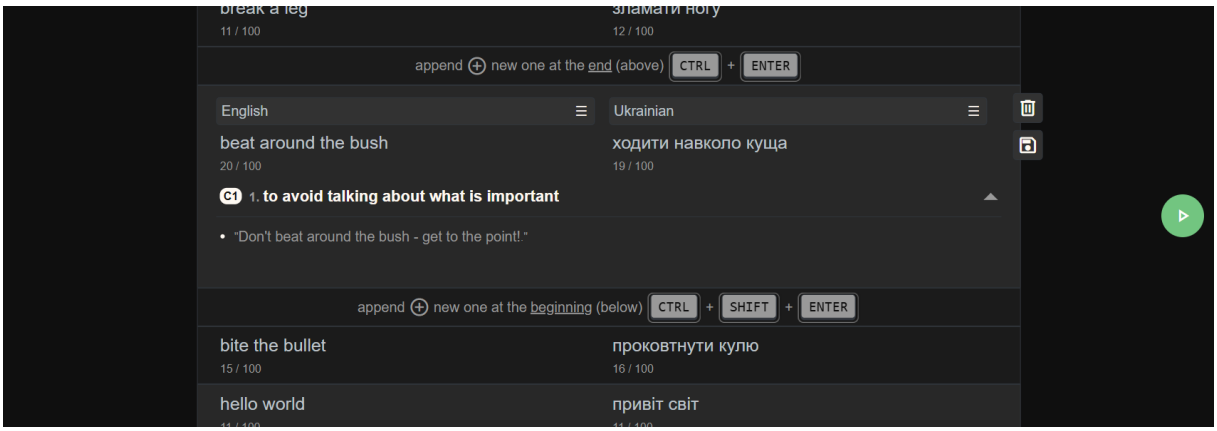

Рисунок 3.16 – Кнопка для початку процесу навчання

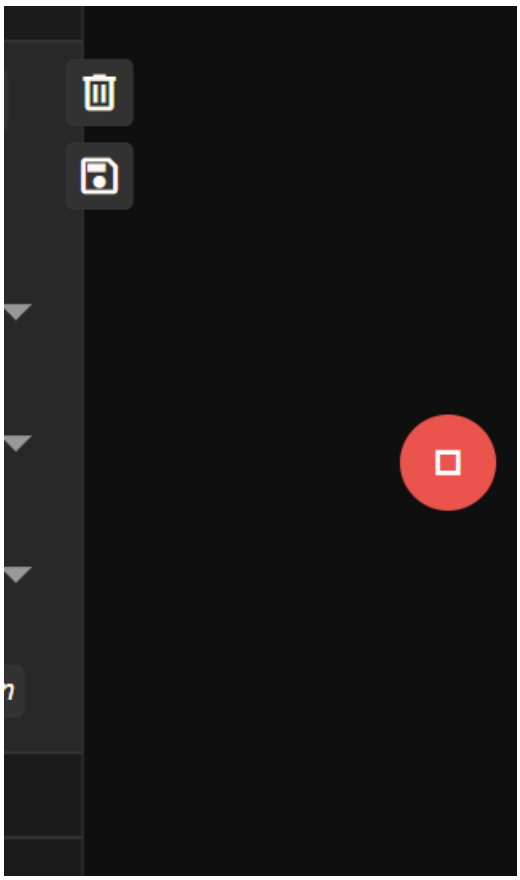

Рисунок 3.17 – Кнопка для призупинки процесу навчання

Для наступної перевірки (рисунки 3.18, 3.19) працездатності процесу навчання виставимо інтервал рівний 5ти секундам.

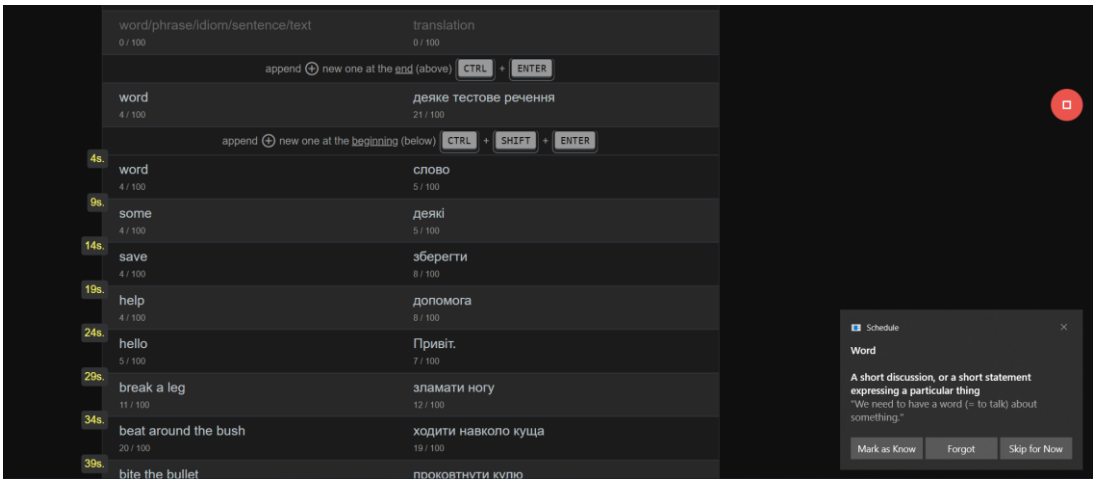

Рисунок 3.18 – Демонстація процесу вивчення зі спеціально заниженими інтервалами для тестування

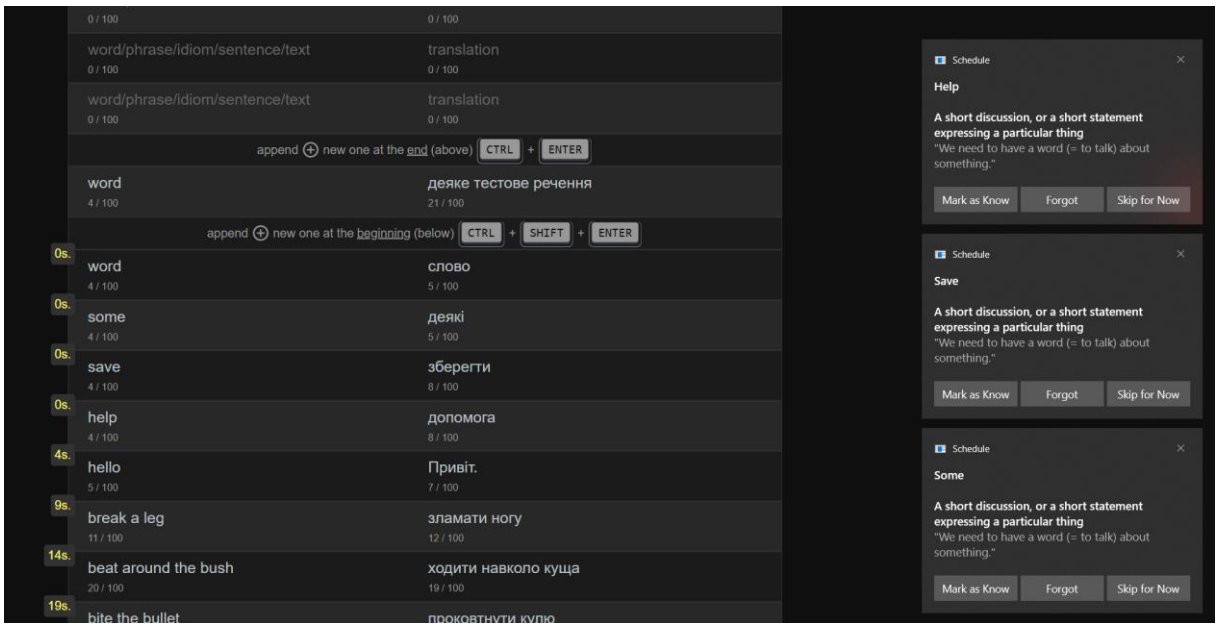

Рисунок 3.19 – Декілька карток надіслано за короткий проміжок часу

Загалом, поточний прототип закладає функціональну основу для реалізації системи адаптивного фонового вивчення мов. Реалізовані компоненти інтерфейсу та механізм сповіщень дозволяють збирати дані для навчання ІА та представляти результати їх роботи користувачеві. Повна оцінка ефективності системи буде можлива лише після завершення розробки та інтеграції інтелектуальних агентів DDPG та DQL, а також проведення всебічного користувацького тестування.

ВИСНОВКИ

Основна мета полягала у створенні програмного рішення, яке поєднує сучасні алгоритми інтервального повторення, моделі людської пам’яті та інтелектуальних агентів, здатних навчатися та адаптуватися до індивідуальних особливостей кожного користувача, зокрема, в режимі фонового навчання.

Необхідно зазначити, що на момент завершення кваліфікаційної роботи, повна програмна реалізація інтелектуальних агентів (DDPG та DQL, їх тренування, інтеграція з FSRS/BKT) та всебічний аналіз ефективності системи шляхом емпіричного тестування перебувають на етапі розробки. Таким чином, висновки щодо ефективності адаптивних алгоритмів є переважно теоретичними та базуються на аналізі проєктних рішень та частковій реалізації. Поточна робота підтверджує потенціал та структуру такої системи, але фактична ефективність ШІ-керованої адаптації залишається гіпотезою, що потребує перевірки. Ключовими наступними кроками є:

– завершення розробки та тренування DDPG та DQL агентів;

– повна імплементація інтеграції моделей FSRS та BKT у функцію винагороди DDPG-агента;

– реалізація гібридної моделі тренування (попереднє навчання глобальної моделі та індивідуальне доналаштування) для вирішення проблеми «холодного старту» та забезпечення глибокої персоналізації;

– повноцінна реалізація модуля контекстної обізнаності на основі даних з сенсорів мобільного пристрою (акселерометр, гіроскоп, мікрофон) для адаптації типу та способу подачі навчального контенту до поточних умов користувача;

– інтеграція розроблених ІА з існуючим UI та бекендом сповіщень;

– проведення комплексного тестування та оцінки впливу системи на ефективність навчання та залученість користувачів, порівняно з існуючими методами.

Подальші дослідження можуть включати вивчення більш складних архітектур ІА, розширення можливостей контекстної обізнаності, проведення масштабних користувацьких досліджень для збору якісного зворотного зв'язку, підтримку більшої кількості мов та типів навчального контенту. Окремої уваги заслуговує дослідження етичних аспектів, пов'язаних із збором та використанням персональних даних та даних із сенсорів пристрою. Успішна реалізація та валідація такої системи не лише надасть ефективний інструмент для вивчення мов, але й стане цінною платформою для подальших досліджень у сферах взаємодії людини з комп'ютером, застосування когнітивних моделей у ШІ та практичного розгортання передових RL-систем у реальних освітніх застосунках.

Представлена кваліфікаційна робота заклала міцне теоретичне та архітектурне підґрунтя для створення інноваційної системи адаптивного фонового вивчення іноземних мов. Частково реалізовані компоненти користувацького інтерфейсу та механізму фонових сповіщень підтвердили життєздатність ключових ідей проєкту. Запропонована система має значний потенціал для кардинального покращення процесу засвоєння мов, роблячи його більш персоналізованим, зручним та інтегрованим у повсякденне життя. Хоча для повної реалізації та підтвердження ефективності всіх інтелектуальних компонентів потрібна подальша розробка та всебічне тестування, проведена робота є важливим кроком на шляху до створення нового типу інтелектуальних інструментів для вивчення мов.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) Spaced Repetition Algorithm: A Three-Day Journey from Novice to Expert. *GitHub*. URL: https://github.com/open-spaced-repetition/fsrs4anki/wiki/Spaced-Repetition-Algorithm:-A-Three-Day-Journey-from-Novice-to-Expert (date of access: 11.06.2025).

2) Rea C. P., Modigliani. The Effect of Expanded versus Massed Practice on the Retention of Multiplication Facts and Spelling Lists. Department of Psychology, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6. 1985. URL: https://gwern.net/doc/psychology/spaced-repetition/1985-rea.pdf (date of access: 14.05.2025).

3) John J. Donovan, David J. Radosevich. The Effect of Expanded versus Massed Practice on the Retention of Multiplication Facts and Spelling Lists. University at Albany, State University of New York. 1999. URL: https://gwern.net/doc/psychology/spaced-repetition/1999-donovan.pdf (date of access: 14.05.2025).

4) Ye Junyao. Open dataset by the language learning application MaiMemo. 2022. DOI: https://doi.org/10.7910/DVN/VAGUL0 (дата звернення: 14.05.2025).

5) Ye J., Su J., Cao Y. A Stochastic Shortest Path Algorithm for Optimizing Spaced Repetition Scheduling. *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington DC USA. New York, NY, USA, 2022. URL: https://doi.org/10.1145/3534678.3539081 (date of access: 14.06.2025).

6) Optimizing Spaced Repetition Schedule by Capturing the Dynamics of Memory / J. Su et al. *IEEE Transactions on Knowledge and Data Engineering*. 2023. P. 1–13. URL: https://doi.org/10.1109/tkde.2023.3251721 (date of access: 14.06.2025).

7) P.A. Wozniak. Account of research leading to the SuperMemo method: 3.1. The approximate function of optimal intervals. University of Technology in

Poznan. 1985. URL: https://super-memory.com/english/ol/beginning.htm#Algorithm (date of access: 14.05.2025).

8) P.A. Wozniak. History of spaced repetition. URL: https://supermemo.guru/wiki/History_of_spaced_repetition (date of access: 14.05.2025).

9) P.A. Wozniak. Application of a computer to improve the results obtained in working with the SuperMemo method. University of Technology in Poznan. May 10, 1998. URL: https://super-memory.com/english/ol/sm2.htm (date of access: 14.05.2025).

10) SRS Benchmark. URL: https://github.com/open-spaced-repetition/srs-benchmark/tree/main (date of access: 14.05.2025).

11) Reinforcement Learning Agents. URL: https://ww2.mathworks.cn/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html (date of access: 14.05.2025).

12) Deep Deterministic Policy Gradient (DDPG) Agent. URL: https://ww2.mathworks.cn/help/reinforcement-learning/ug/ddpg-agents.html (date of access: 14.05.2025).

13) Lillicrap TP, Hunt JJ, Pritzel A, Heess N, et al. Continuous control with deep reinforcement learning. DOI: https://doi.org/10.48550/arXiv.1509.02971 (date of access: 14.05.2025).

14) Anirudhan Badrinath, Frederic Wang, Zachary Pardos. pyBKT: An Accessible Python Library of Bayesian Knowledge Tracing Models. May 2, 2021. DOI: https://doi.org/10.48550/arXiv.2105.00385 (date of access: 14.05.2025).

15) Cambridge Dictionary. Cambridge University Press & Assessment. URL: https://dictionary.cambridge.org/ (date of access: 14.05.2025).

16) DeepL API. URL: https://www.deepl.com/en/products/api (date of access: 14.05.2025).

17) Apache Kafka. URL: https://kafka.apache.org/documentation/ (date of access: 14.05.2025).

18) Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu. 2015. Human-level control through deep reinforcement learning. DOI: https://doi.org/10.1038/nature14236 (date of access: 11.06.2025).

19) Docker Inc. Docker Documentation. URL: https://docs.docker.com/ (date of access: 11.06.2025).

20) SvelteKit Documentation. URL: https://kit.svelte.dev/docs (date of access: 11.06.2025).