Київський національний університет імені Тараса Шевченка

Навчально-науковий інститут філології

Кафедра української мови та прикладної лінгвістики

Чат-бот вправ з морфології української мови

Кваліфікаційна робота

освітнього ступеня «бакалавр»

студентки IV курсу освітньої програми

035.10 Прикладна(комп'ютерна лінгвістика)

та англійська мова

Шимків Каріни Миколаївни

Науковий керівник:

доц. Костіков Микола Павлович

«Допущено до захисту»
Протокол засідання
кафедри української мови та прикладної лінгвістики
від «06» 06 2024 року № 15

завідувач кафедри _____к.філол.н., доц. Сергій РІЗНИК

КИЇВ – 2024

ЗМІСТ

Анотація

Бакалаврська робота присвячена розробці та дослідженню чат-бота для вивчення морфології української мови, актуальність якого зумовлена необхідністю створення ефективних інструментів для покращення мовних навичок. Об'єктом дослідження є процес навчання морфології української мови. Предметом дослідження є використання чат-бота для навчання та вдосконалення морфології української мови.

Метою дослідження є розробка та впровадження чат-бота, який сприяє покращенню навичок української мови через інтерактивні тести та завдання. Завдання дослідження включають аналіз існуючих рішень у цій галузі, визначення оптимальних методів інтеграції мовних технологій, розробку чат-бота. Методологічну основу дослідження становить комплексний підхід, який поєднує теоретичний аналіз наукових джерел з емпіричними дослідженнями, спрямованими на оцінку ефективності впровадження чат-бота в навчальний процес.

Структура роботи складається з кількох розділів, що охоплюють теоретичний аналіз, методологічний підхід та розробку.

Наукова новизна дослідження полягає у впровадженні інноваційного підходу до навчання української мови через інтерактивні технології, що забезпечує гнучкість і адаптивність навчального процесу. Деякі положення по розробці диплому були опубліковані на наукові конференції з ІТ [1], що дало ще більший поштовх для вдосконалення ідеї. Розроблений чат-бот включає можливість проведення тестів на наголос і частини мови та відмінювання, надання зворотного зв'язку та пояснень, що сприяє глибшому засвоєнню матеріалу.

Результати дослідження підтверджують ефективність використання чат-ботів у навчальному процесі. Застосування інтерактивних завдань і миттєвого зворотного зв'язку значно підвищує мотивацію та успішність

учнів. Розроблений чат-бот може також бути ефективним інструментом для самостійного вивчення української мови.

Ключові слова: чат-бот, морфологія, українська мова, навчання, штучний інтелект, інтерактивні завдання.

Summary

This bachelor's thesis is dedicated to the development and research of a chatbot designed for learning the morphology of the Ukrainian language. The relevance of this study stems from the need to create effective tools for improving language skills. The object of the study is the process of learning Ukrainian morphology, while the subject is the use of a chatbot for teaching and enhancing knowledge of Ukrainian morphology.

The aim of the research is to develop and implement a chatbot that enhances Ukrainian language skills through interactive tests and exercises. The research tasks include analyzing existing solutions in this field, identifying optimal methods for integrating linguistic technologies, and developing the chatbot. The methodological foundation of the study is a comprehensive approach that combines theoretical analysis of scientific sources with empirical research aimed at evaluating the effectiveness of implementing the chatbot in the educational process.

The scientific novelty of the research lies in the implementation of an innovative approach to learning the Ukrainian language through interactive technologies, ensuring flexibility and adaptability in the learning process. Some findings from the development of the diploma were published at scientific IT conferences [1], which further propelled the enhancement of the idea. The developed chatbot includes the ability to conduct stress and part-of-speech tests and declensions, providing feedback and explanations, thereby facilitating deeper material comprehension.

The research results confirm the effectiveness of using chatbots in the educational process. The use of interactive tasks and immediate feedback significantly boosts student motivation and success. The developed chatbot can also serve as an effective tool for self-study of the Ukrainian language.

Keywords: Chatbot, morphology, Ukrainian language, education, artificial intelligence, interactive tasks.

Вступ

В умовах сучасного розвитку інформаційних технологій особливу актуальність набуває використання цифрових інструментів для навчання. Чат-боти, як інтерактивні системи, здатні значно покращити процес вивчення мов, надаючи всім охочим можливість отримувати негайні відповіді на питання та практикувати мовні навички в інтерактивному середовищі. «Чат-боти забезпечують безперервне навчання, надаючи студентам можливість отримувати негайні відповіді на їхні запитання, що підвищує їх мотивацію та залученість до навчального процесу» [2]. Морфологія української мови є однією з найскладніших і водночас важливих складових мовної компетенції, тому використання чат-ботів для її вивчення має велике практичне значення.

В чат-боті можна виконувати вправи, аналізувати помилки та отримувати зворотній зв'язок, а для того, щоб ним користуватись необхідний лише встановлений месенджер.

Практичне значення полягає в запровадження чат-ботів для вивчення морфології української мови, а це може сприяти підвищенню якості мовної освіти, зробити її більш доступною та індивідуалізованою. «Інтерактивні можливості чат-ботів сприяють покращенню комунікативних навичок студентів, дозволяючи їм практикувати мову в реальних ситуаціях» [3]. Використання таких інструментів дозволяє створити умови для постійної практики, що є важливим аспектом ефективного засвоєння мовного матеріалу.

У науковій літературі існує кілька основних підходів до використання чат-ботів у навчанні. Дослідження показують, що інтерактивність та адаптивність чат-ботів дозволяють ефективно залучати студентів до навчального процесу, покращувати їхні знання та навички. Зокрема, такі

інструменти вже успішно використовуються для вивчення іноземних мов, що підтверджує їхню ефективність.

Мета дослідження полягає у розробці та впровадженні чат-бота для вивчення морфології української мови на базі месенджера Telegram.

Завдання дослідження:

1. Провести аналіз існуючих чат-ботів для вивчення мов.
2. Розробити концепцію чат-бота, орієнтованого на вивчення морфології української мови.
3. Реалізувати чат-бота та інтегрувати його у навчальний процес.
4. Оцінити ефективність використання чат-бота через емпіричні дослідження.

Об'єктом дослідження є процес навчання морфології української мови. Предметом дослідження є використання чат-бота для навчання та вдосконалення морфології української мови.

Методологічною основою дослідження є комплексний підхід, що поєднує теоретичний аналіз наукових джерел з емпіричними дослідженнями, спрямованими на оцінку ефективності впровадження чат-бота у навчальний процес.

Методи вирішення поставлених завдань:

Для досягнення поставлених завдань будуть використані такі методи:

- Аналіз наукової літератури та існуючих чат-ботів для вивчення мов.
- Розробка та програмування чат-бота.
- Використання чат-бота, його тестування.

Завдяки використанню цих методів буде проведено всебічний аналіз впливу чат-бота на процес навчання морфології української мови та визначено його практичну цінність.

РОЗДІЛ 1 Теоретичне обгрунтування вибору теми дослідження

1.1. Історія появи чат-ботів

Перед початком ознайомлення з методами розробки необхідно мати хоча б базове розуміння того, що таке чат-бот. Оксфордський словник англійської мови визначає чат-бот як "комп’ютерну програму, яка може вести розмову з людиною, зазвичай через Інтернет". Це визначення в цілому описує функціональність таких програм, проте не охоплює всю їх складність і різноманітність застосувань.

Чат-боти, або розмовні агенти, стали невід'ємною частиною сучасних інформаційних технологій, сприяючи автоматизації обслуговування клієнтів, покращенню користувацького досвіду та ефективнішій комунікації. Їх розвиток почався з простих програм, здатних відповідати на обмежену кількість заздалегідь підготовлених питань, і еволюціонував до складних систем, що використовують штучний інтелект та машинне навчання для проведення глибших і змістовніших діалогів.

Як зазначають Сміт та Андерсон (2020), «чат-боти стають дедалі складнішими завдяки інтеграції штучного інтелекту, що дозволяє їм вести більш природні та змістовні розмови з користувачами» [4].

Для повного розуміння того, як і для чого створюються чат-боти, необхідно розглянути їх історію розвитку, ключові технології, що лежать в їх основі, а також приклади їх успішного використання в різних галузях. Це дозволить не лише усвідомити поточний стан технології, але й передбачити її майбутні напрямки розвитку.

«Англійський словник був використаний не випадково для визначення поняття "чат-бот", адже перша програма цього типу під назвою ELIZA, розроблена в Америці, могла обробляти природну англійську мову» [5]. Перший чат-бот був створений Джозефом Вайзенбаумом у 1966 році та

використовував методологію "matching and substitution" для моделювання розмови. ELIZA була запрограмована на імітацію терапевтичної розмови в стилі Роджеріанської терапії, яка полягає в тому, що терапевт ставить запитання пацієнту, перефразовуючи його висловлювання. Наприклад, відповідь на "У мене болить голова" могла бути "Чому ти кажеш, що у тебе болить голова?", а відповіддю на "Моя мати мене ненавидить" було "Хто ще у вашій родині вас ненавидить?".

Наступним відомим проектом був PARRY, чат-бот, розроблений Кеннетом Колбі через шість років після створення ELIZA. PARRY імітував пацієнта з шизофренією, використовуючи схожу методологію до ELIZA, і зміг пройти тогочасний тест Тьюрінга та тест на параноїдальний розлад.

Як зазначено у дослідженнях, «PARRY був одним із перших прикладів застосування комп'ютерних програм для імітації людських розмов та психологічних станів, що було значним досягненням у сфері штучного інтелекту і психіатрії» [6]. «»

Подальшими значущими проектами стали «Dr. Sbaitso та ALICE, які цікаві перш за все завдяки використанню штучного інтелекту в чат-ботах. Dr. Sbaitso був створений у 1992 році та імітував розмову з психологом, використовуючи елементи штучного інтелекту. З цим ботом досі можна поспілкуватися, ввівши його назву в пошуку, але замість складної взаємодії з користувачем, на більшість повідомлень бот відповідатиме запитанням "Why do you feel that way?"»[7].

ALICE, створена у 1995 році, мала більш розвинений штучний інтелект та імітувала розмову з реальною людиною через Інтернет. ALICE використовувала технологію AIML (Artificial Intelligence Markup Language), що дозволяло їй вести більш складні та змістовні розмови з користувачами.

Наступною важливою групою чат-ботів стали Siri, Alexa, Cortana та Google Assistant. Ці боти, розроблені після 2010 року, мають схожий функціонал та спільну мету – полегшення взаємодії користувачів з певними платформами. Вони використовують новітні технології для реалізації бажаних функцій. Ці боти можуть обробляти запити в різних форматах, включаючи голосові (аудіо), текстові та фото, та відповідати користувачам, а також виконувати певні дії. Наприклад, користувач iPhone може вигукнути "Привіт, Siri" для активації бота, а потім дати команду "Зателефонуй [ім'я контакту]", щоб зателефонувати людині зі свого списку контактів.

Сучасні чат-боти продовжують розвиватися, використовуючи штучний інтелект та машинне навчання для покращення взаємодії з користувачами та розширення функціональних можливостей. Це дозволяє їм не тільки відповідати на запитання, але й виконувати складніші завдання, адаптуватися до потреб користувачів та покращувати користувацький досвід.

1.2. Типи чат-ботів

Чат-бот – це тип програмного забезпечення для штучного інтелекту, яке може імітувати розмову (або чат) із користувачем природною мовою за допомогою програм обміну повідомленнями, веб-сайтів, мобільних додатків або телефонів. «Чатових ботів часто описують як один з найдосконаліших та перспективних способів організації діалогової взаємодії людини з комп’ютером. Однак з технічної точки зору чат-боти представляють лише природну еволюцію системи відповідей на питання, що використовують обробку природними мовами (NLP). Формулювання відповідей на питання природною мовою є одним із найбільш типових прикладів обробки природної мови, що використовується в різних

кінцевих цілях підприємства» [29].

«На цей момент існує велика класифікація чат-ботів: по платформі впровадження, технології розробки, способу спілкування з користувачами і функціональності» [3].

Але найпоширенішими залишається поділ з точки зору реалізації:

1. «Бізнес класифікація яка полягає у сферах застосування чат-ботів» [8].
2. Технічна класифікація полягає в основі створення чат-боту.

За технічною класифікацією чат-боти поділяються на прості, розумні, які є з підтримкою штучного інтелекту та гібридні.

Простий бот – це бот, який відповідає на запитання на основі заздалегідь встановленого вибору інтегрованих відповідей. «Простих ботів також називаються ботами дерева прийняття рішень, як випливає з назви, вони використовують ряд визначених правил та нагадують текстову систему IVR (система попередньо записаних голосових повідомлень, що виконує функцію маршрутизації дзвінків всередині контакт-центру)» [9].

Такі боти не роблять жодних висновків з попередніх взаємодій та найкраще підходять для прямолінійних діалогів.

Простий бот зазвичай будується на базі кнопок, тому він ідеально підходить для опитування, підтримки продажів і практично для будь-якого простого завдання автоматизації процесів, де сценарії спілкування чітко визначені.

«Натомість *розумні чат-боти* з підтримкою штучного інтелекту створені для імітації майже людської взаємодії з клієнтами» [10]. Для того, щоб вести вільні розмови і розуміти намір, мову та почуття розумні чатботи використовують технології обробки природних мов (NLP, NLU тощо). Natural Language Processing (NLP) – це область штучного інтелекту, яка дозволяє комп’ютерам аналізувати та розуміти людську мову.

Найбільша відмінність від простого чат-бота полягає у використанні моделей машинного навчання, що значно збільшує функціональність бота, оскільки він здатний ідентифікувати сотні різних запитань, написаних людиною.

Бот запрограмований на самостійне навчання, оскільки він вводиться в нові діалоги та слова. Фактично, коли чат-бот отримує нові голосові або текстові повідомлення, кількість запитів, на які він може відповісти, і точність кожної відповіді, яку він дає, збільшується.

Гібридний чат-бот – це поєднання боту побудованого на базі штучного інтелекту та простого [8]. Зазвичай такі боти говорять: «Я бот, який може допомогти вам у розв’язанні таких питань; будь ласка, натисніть відповідну кнопку X, Y, Z або введіть своє запитання у поле нижче ». Особливість цих ботів полягає у тому, що вони використовують і дерева прийняття рішень і технології обробки природних мов.

Для прикладу:

ELIZA та PARRY - це прості чат-боти, які використовують методологію "matching and substitution", або ж скриптові чат-боти, що працюють з певним набором команд за заданим сценарієм.

Dr. Sbaitso та ALICE - це чат-боти з використанням штучного інтелекту (AI chatbots). Взаємодія з такими ботами більше нагадує спілкування з реальною людиною, ніж з програмою. Варто також зазначити, що можливості глибокого навчання дозволяють таким чат-ботам з часом ставати точнішими у своїх відповідях, створюючи мережу відповідних реакцій через взаємодію з користувачами.

Siri, Alexa, Cortana та Google Assistant - це віртуальні помічники (virtual agents), які в більшості випадків поєднують у собі елементи обох типів. Вони мають вбудований штучний інтелект, а також певний список команд з заготовленими відповідями або діями на них.

1.3. Сфери застосування різних типів чат-ботів

1) Звичайні чат-боти є найбільш розповсюдженими сьогодні та використовуються у багатьох галузях, замінюючи або доповнюючи певні сервіси. Наприклад, ви можете придбати квитки на потяг у касі на вокзалі, що вимагає знати розклад потягів і підлаштовувати свій графік під час купівлі. Іншим, більш зручним способом є купівля квитків онлайн на сайті УкрЗалізниці, хоча це теж має свої недоліки. Наприклад, вам доведеться кожного дня перевіряти наявність потрібного потягу. Зате, за допомогою чат-бота RailwayBot, можна запустити моніторинг потрібного маршруту і при появі відповідного квитка одразу його придбати. Таким чином, звичайні боти використовуються для створення нових сервісів або доповнення вже існуючих.

2) Чат-боти з штучним інтелектом (AI chatbots) використовують розуміння природної мови для розпізнавання потреб користувача. Вони застосовують передові інструменти штучного інтелекту для визначення цілей користувача. Завдяки машинному та глибокому навчанню, ці чат-боти постійно розвивають детальну базу знань із запитань і відповідей, що базується на взаємодії з користувачами. Це покращує їх здатність точно передбачати потреби користувачів та адекватно реагувати з часом. Сьогодні ця технологія є невід'ємною частиною всіх онлайн-консультантів та систем підтримки.

3) Віртуальні помічники, такі як Siri, Alexa, Cortana та Google Assistant, завжди були приємним доповненням до нових гаджетів. Компанії та їх розробники змагаються у створенні максимально швидких, зручних та корисних віртуальних асистентів, щоб зробити свої продукти більш привабливими для кінцевих користувачів. Ці віртуальні помічники поєднують у собі елементи простих чат-ботів і ШІ, обробляючи як текстові, так і голосові запити. Вони допомагають користувачам

виконувати різноманітні завдання, такі як управління календарем, здійснення дзвінків, пошук інформації в Інтернеті та багато іншого.

«Крім основних сфер застосування, чат-боти стають важливими інструментами в інших галузях. Наприклад, у сфері охорони здоров'я» [11] чат-боти допомагають пацієнтам знайти необхідну інформацію, записатися на прийом до лікаря, нагадують про прийом ліків. «У фінансовій сфері чат-боти можуть допомагати з управлінням фінансами» [12], надавати консультації щодо інвестицій та здійснювати транзакції. «У сфері освіти чат-боти допомагають студентам знаходити матеріали для навчання, відповідати на запитання та організовувати розклад занять» [13].

Чат-боти продовжують розвиватися та інтегруватися в різні аспекти нашого життя, роблячи його зручнішим та ефективнішим.

1.4. Аналіз існуючих чат-ботів для вивчення української мови у Telegram

1.4.1. Аналіз чат-бота «Баба Катря»

Чат-бот «Баба Катря» [14] є інструментом для вивчення української мови, доступним у месенджері Telegram. Цей бот орієнтований на покращення мовних навичок користувачів, зокрема на тренування правильного наголосу у словах. Основна мета «Баби Катрі» – зробити процес навчання простим, інтерактивним і цікавим.

Функціональність:

1. Вибір режиму навчання:

- Після початку листування (рис.1), чат-бот запитує, як користувач хоче навчатися, пропонуючи два варіанти: «Тренувати наголоси» або «На

курс до внучок». Це дозволяє користувачеві обрати найбільш підходящий для нього метод навчання.

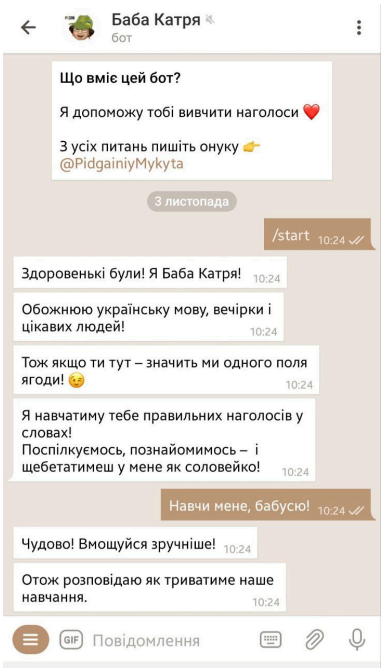


рис.1 - початок листування з ботом

2. Тренування наголосів:

- У режимі тренування наголосів (рис.2) бот пропонує користувачу два варіанти наголосу для певного слова. Користувач повинен обрати правильний варіант. Наприклад, уподОбання чи уподОбАння? Правильна відповідь – уподОбання. Якщо користувач вибирає правильний варіант, бот може надати додаткове пояснення для кращого запам'ятовування.

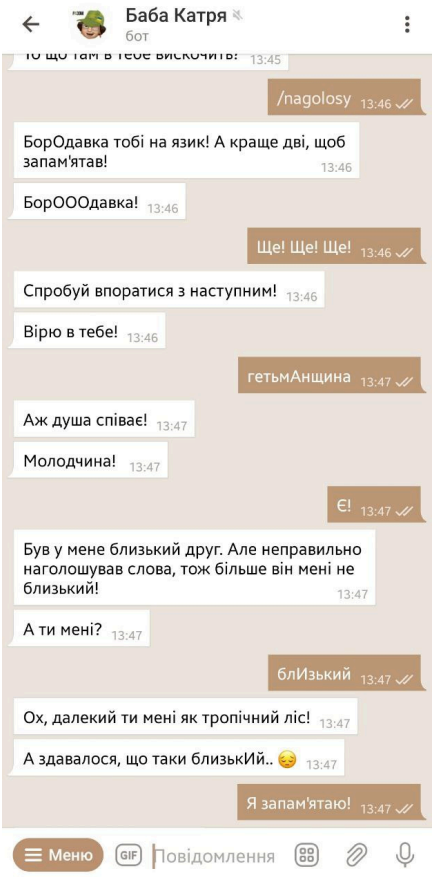


рис.2 - режим тренування

3. Інтерактивні реакції:

- Чат-бот «Баба Катря» реагує по-різному на правильні та неправильні відповіді, що додає елемент інтерактивності. Така особливість робить навчання більш цікавим і мотивує користувачів продовжувати тренування.

4. Елементи гумору та персоналізації:

- Чат-бот використовує стиль спілкування, який нагадує розмову з реальною людиною, зокрема з бабусею. Це робить взаємодію більш приємною та неформальною, що може сприяти кращому засвоєнню матеріалу.

Переваги:

- Простота використання: Інтерфейс чат-бота зрозумілий та інтуїтивний, що робить його доступним для широкого кола користувачів.

- Миттєвий зворотний зв'язок: Користувачі отримують негайну реакцію на свої відповіді, що сприяє ефективному засвоєнню матеріалу.

- Інтерактивність: Різні реакції на правильні та неправильні відповіді роблять процес навчання динамічним і цікавим.

- Персоналізований підхід: Використання стилю спілкування, який нагадує розмову з бабусею, робить взаємодію більш теплою та привабливою.

Недоліки:

- Обмежений обсяг матеріалу: Чат-бот спеціалізується переважно на тренуванні наголосів, що може не задовольнити потреби користувачів, які хочуть вивчати інші аспекти української мови.

- Залежність від настрійного фактору: Використання елементів гумору та персоналізації може бути не всім до вподоби, особливо тим, хто віддає перевагу більш формальному стилю навчання.

Висновок:

Чат-бот «Баба Катря» є ефективним інструментом для тренування наголосів у словах української мови. Його сильні сторони включають простоту використання, миттєвий зворотний зв'язок, інтерактивність та персоналізований підхід. Проте обмежений обсяг матеріалу та залежність від стилю спілкування можуть бути потенційними недоліками для деяких користувачів. В цілому, «Баба Катря» є цінним ресурсом для тих, хто прагне покращити свої навички української мови в невимушеній і цікавій формі.

1.4.2. Опис чат-бота «Твоя Мова»

Чат-бот «Твоя Мова» [15] спрямований на допомогу користувачам у вивченні української мови. Бот надає різноманітні інтерактивні завдання та

вправи для покращення знань з граматики, лексики, орфографії та інших аспектів української мови.

Особливості:

- Інтерактивні вправи: Чат-бот пропонує користувачам різні види вправ, які охоплюють основні аспекти української мови (рис.3). Це можуть бути завдання на запам'ятовування слів, вправи з граматики та орфографії, а також тестові запитання для перевірки знань.

- Тестування знань: Регулярні тести допомагають користувачам перевіряти свій прогрес та виявляти слабкі місця в знаннях. Чат-бот надає зворотний зв'язок по результатах тестів, що сприяє покращенню процесу навчання.

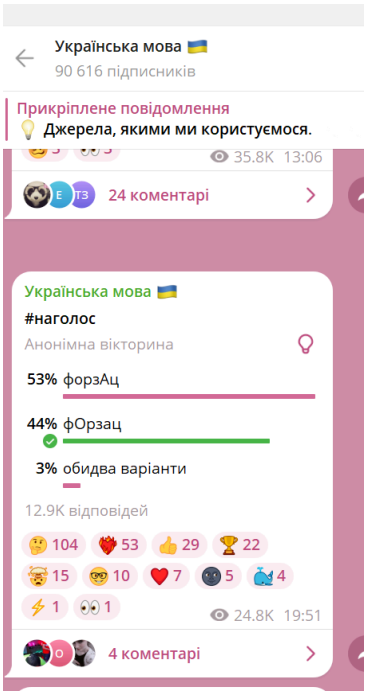


рис.3

Структура навчання:

- Щоденні завдання: Користувачі отримують щоденні завдання для практики, що допомагає підтримувати регулярність та послідовність у навчанні.

- Миттєвий зворотний зв'язок: Після виконання завдання користувач одразу отримує зворотний зв'язок (рис.4), що допомагає швидко зрозуміти помилки та виправити їх. Біля кожного завдання є покликання на правило згідно завдання.

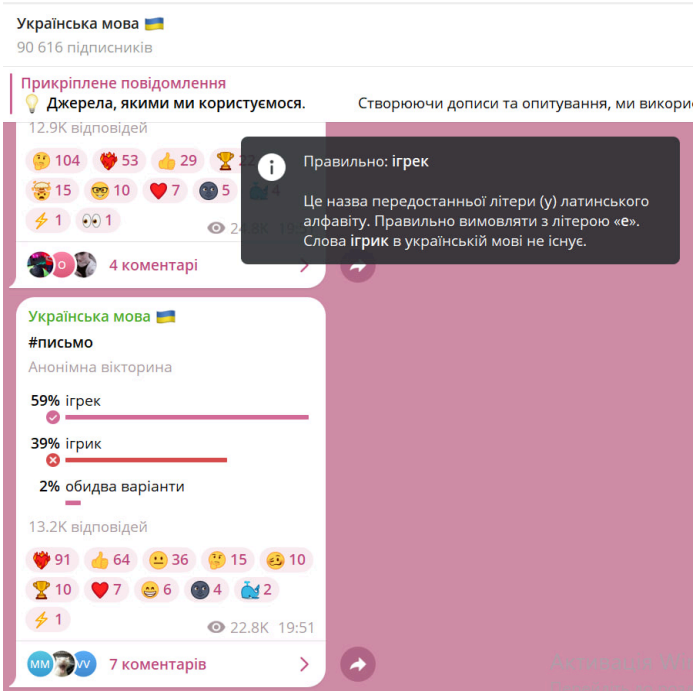


рис.4

Додаткові функції:

- Культурні та історичні контексти: Час від часу бот надає цікаві факти про українську культуру, історію та традиції, що робить процес навчання більш цікавим та різноманітним.

- Поради та рекомендації: Чат-бот надає корисні поради щодо ефективного вивчення мови та рекомендує додаткові ресурси для самостійного навчання.

Висновок:

Чат-бот «Твоя Мова» є ефективним інструментом для вивчення української мови, який поєднує в собі різноманітні інтерактивні завдання, тестування знань та індивідуальний підхід до кожного користувача. Його

особливості та структура навчання роблять процес засвоєння матеріалу цікавим та продуктивним, що сприяє більш глибокому та систематичному вивченню української мови.

Висновок до Розділу 1

В цьому розділі було розглянуто теоретичне обґрунтування вибору теми дослідження, яке охоплює історію появи чат-ботів, їх типи та сфери застосування, а також аналіз існуючих чат-ботів для вивчення української мови.

З огляду на історичний контекст, ми бачимо, що розвиток чат-ботів пройшов довгий шлях від простих програм, таких як ELIZA та PARRY, до сучасних систем, що використовують штучний інтелект та машинне навчання. Ці технології значно розширили функціональність чат-ботів, перетворивши їх на потужні інструменти для автоматизації сервісів, покращення користувацького досвіду та ефективнішої комунікації.

Було також висвітлено різні типи чат-ботів: прості, інтелектуальні з підтримкою штучного інтелекту та гібридні. «Розуміння цієї класифікації є важливим для визначення того, який тип чат-бота найкраще підходить для конкретних завдань та потреб» [16].

Сфери застосування чат-ботів охоплюють широкий спектр галузей, від обслуговування клієнтів до освіти, охорони здоров'я та фінансових послуг. Це свідчить про їхню універсальність та важливість у сучасному світі.

Аналіз існуючих чат-ботів для вивчення української мови у Telegram, таких як «Баба Катря» та «Твоя Мова», показав, що ці боти є ефективними інструментами для покращення мовних навичок користувачів. Вони використовують різні методи інтерактивного навчання, надають миттєвий зворотний зв'язок та забезпечують персоналізований підхід до кожного

користувача. Водночас, було виявлено певні недоліки, такі як обмежений обсяг матеріалу та залежність від стилю спілкування.

Загалом, розуміння історії розвитку, типів та сфер застосування чат-ботів, а також аналіз існуючих рішень для вивчення української мови, дозволяє зробити висновок про необхідність подальшого розвитку та вдосконалення чат-ботів. Це створює перспективи для розробки нових, більш функціональних та ефективних інструментів, які можуть бути корисними в різних сферах життя та діяльності.

РОЗДІЛ 2 Практична реалізація чат-бота

2.1. Огляд плану чат-бота

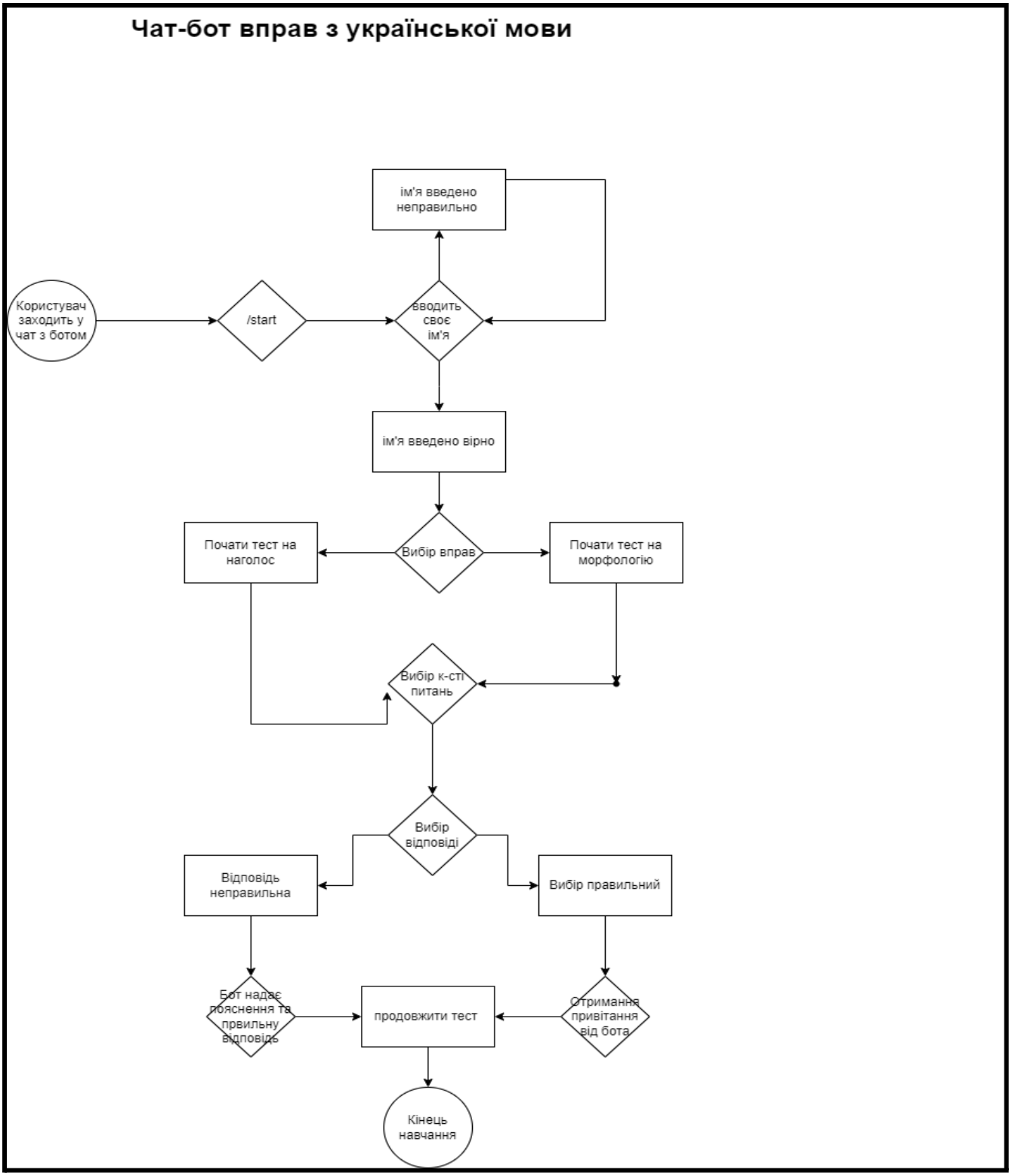
Перед реалізацію коду було створено UserFlow, тобто блоксхеми, які відображають шлях користувача або поряд дій, які користувач робить при використанні продукту. Розробка такої схеми допомагає зробити чат-бот більш інтуїтивним та простішим у використанні для користувача.

Перед початком створення User Flow важливо зрозуміти, які основні цілі переслідує користувач, взаємодіючи з чат-ботом. Наприклад:

- Почати тест на наголос.
- Почати тест на частини мови.
- Отримати зворотний зв'язок на відповідь.

Процес взаємодії користувача з ботом можна розділити на основні етапи:

1. Привітання і запит імені.
2. Вибір типу тесту.
3. Вибір кількості запитань.
4. Проходження тесту.
5. Отримання результатів і зворотного зв'язку.


Чат-бот вправ з української мови
ім'я введено неправильно
Користувач заходить у чат з ботом
/start
вводить своє ім'я
ім'я введено вірно
Почати тест на наголос
Вибір вправ
Почати тест на морфологію
Вибір к-сті питань
Вибір відповіді
Відповідь неправильна
Вибір правильний
Бот надає пояснення та првильну відповідь
продовжити тест
Отримання привітання від бота
Кінець навчання

2.2.1. Вибір мови програмування

У сучасному світі програмування, де кожна мова має свої особливості та застосування, вибір найкращої мови для конкретного проєкту стає вирішальним кроком для його успіху. Розглядаючи мову програмування для створення чат-бота, зіштовхнулися з декількома популярними варіантами, кожен з яких має свої сильні та слабкі сторони. Зокрема, для нашого проекту ми розглядали Python, JavaScript, Java, C++, та Rust.

Python є однією з найпопулярніших мов програмування завдяки своїй простоті та читабельності. Вона широко використовується в наукових дослідженнях, машинному навчанні та веб-розробці. Python має велику кількість бібліотек та активну спільноту, що робить її дуже привабливою для новачків та досвідчених розробників. Однак, Python має свої недоліки: вона відносно повільна у виконанні та має проблеми з багатопоточністю, що може бути критичним у контексті високопродуктивних чат-ботів.

JavaScript, як основна мова веб-розробки, забезпечує універсальність у створенні динамічного контенту на веб-сторінках. Вона підтримується всіма браузерами і широко використовується для створення інтерактивних веб-додатків. Однак, JavaScript має проблеми з безпекою і відносно повільну швидкість виконання, що може обмежувати її використання для складних систем з високими вимогами до безпеки та продуктивності.

Java відома своєю портативністю, висока продуктивністю та багатопоточністю. Вона широко використовується для корпоративних додатків та мобільних додатків на платформі Android. Проте, Java може бути складною в освоєнні і має високу споживаність ресурсів, що може створювати проблеми для менш ресурсомістких систем, таких як чат-боти.

C++ забезпечує високий рівень контролю над апаратними ресурсами і високу продуктивність. Вона широко використовується в системному програмуванні та ігровій індустрії. Однак, C++ є складною у використанні і може мати потенційні проблеми з безпекою пам'яті, що може стати великою перешкодою для безпечних систем.

Rust виділяється серед цих мов завдяки своїм унікальним перевагам, які роблять її ідеальним вибором для розробки нашого чат-бота. Однією з найважливіших характеристик Rust є її безпека. Rust забезпечує сувору безпеку пам'яті завдяки системі перевірки запозичень і власності. Це знижує ризик багатьох поширених помилок, таких як звернення до неініціалізованої пам'яті або витоки пам'яті. «У порівнянні з C++ та іншими низькорівневими мовами, Rust автоматично запобігає багатьом типам помилок, що робить її надійнішою для критичних застосувань» [17].

Крім того, Rust відома своєю швидкістю. «Вона компілюється до машинного коду та ефективно управляє пам'яттю, забезпечуючи продуктивність, порівнянну з C++. Це дозволяє створювати високопродуктивні програми, які можуть обробляти великі обсяги даних та взаємодіяти з користувачами в режимі реального часу» [18].

Ще одна важлива перевага Rust — її сильна і розвинута система типізації. Rust має потужну систему типізації, яка допомагає уникати багатьох типів помилок на етапі компіляції. «Наприклад, використання Rust для реалізації POS-тегування (Part of Speech Tagging) демонструє, як її система типізації може забезпечити точність та надійність» [30]. Завдяки цьому можна створити чат-бота, який використовує дерева відповідальності зі скінченним автоматом, де стани бота описані у State enum у файлі main.rs:

```
pub enum State {
    #[default]
    Start,
    ReceiveFullName,
    RecieveGameChoice,
    StressedWordsQuizRecieveAmountOfQuestions,
    StressedWordsQuiz {
        quiz: quiz::Quiz,
        question_number: usize,
        score: usize,
    },
}
```

Наприклад, у коді чат-бота Rust дозволяє легко працювати з різними станами користувача, як у наступному прикладі:

```
const GREETING_TEXT: &str = "Привіт! Я -- морфологічний бот. Я допоможу тобі вивчити українську мову! Давай познайомимося! Як тебе звати?";
```

```
async fn start(bot: Bot, dialogue: QuizDialogue, msg: Message) -> HandlerResult {
    bot.send_message(msg.chat.id, GREETING_TEXT).await?;
    dialogue.update(State::ReceiveFullName).await?;
    Ok(())
}
```

У цьому прикладі бот починає взаємодію з користувачем, відправляючи привітальне повідомлення і змінюючи стан користувача на State::ReceiveFullName. Це дозволяє обробляти наступні дії користувача у відповідному контексті. Такі переходи між станами роблять логіку програми більш зрозумілою та керованою, що є великою перевагою Rust.

Таким чином, Rust є оптимальним вибором для розробки нашого чат-бота завдяки своїй безпеці, швидкості та розвиненій системі типізації. Ці характеристики забезпечують надійну, продуктивну та зручну у використанні платформу для створення складних і високоефективних програмних рішень [30].

2.2.2. Месенджер Telegram

Інтерактивні комунікаційні платформи стають дедалі популярнішими, важливим кроком у розробці чат-бота є вибір оптимальної платформи для його реалізації. Розглянемо, чому Telegram є найкращим вибором для нашого чат-бота, враховуючи кількість користувачів, простоту використання та доступність Bot API.[19]

«Показовою перевагою є звісно кількість користувачів застосунку. Telegram є однією з найбільш швидкозростаючих платформ для обміну повідомленнями у світі, з більш ніж 700 мільйонами активних користувачів на місяць станом на 2023 рік. Хоча ця кількість менша порівняно з гігантами, такими як WhatsApp (понад 2 мільярди користувачів) та Facebook Messenger (понад 1,3 мільярда користувачів), Telegram продовжує набирати популярність завдяки своїм унікальним функціям та зосередженості на безпеці.» [20]

Інтерфейс Telegram відомий своєю інтуїтивною простотою, що робить його доступним для користувачів будь-якого рівня технічної підготовки. Користувачі можуть легко адаптуватися до платформи завдяки її зручному дизайну та простому управлінню чатами і ботами. Ця простота використання є важливою для забезпечення активної взаємодії користувачів з чат-ботом.

Однією з найважливіших переваг Telegram є його розвинена Bot API, яка є однією з найкращих на ринку. Вона надає розробникам потужні інструменти для створення багатофункціональних ботів. Основні можливості Telegram Bot API включають:

1. Вебхуки: Підтримка вебхуків дозволяє швидко отримувати оновлення та події, що робить інтеграцію з іншими сервісами ефективною і швидкою.
2. Підтримка мультимедіа: Боти можуть обробляти текстові повідомлення, фото, відео, документи та інші типи контенту, що робить взаємодію з користувачами багатогранною.
3. Кастомізація інтерфейсу: Можливість створення кастомних клавіатур, меню та кнопок покращує взаємодію користувача з ботом, роблячи її більш інтерактивною та зручною.

4. Аутентифікація користувачів: Підтримка OAuth2.0 для безпечної аутентифікації користувачів, що забезпечує високий рівень безпеки даних.

Порівняння з іншими платформами:

Хоча інші платформи також мають свої переваги, Telegram виділяється завдяки своїй потужній Bot API та фокусі на безпеці. WhatsApp, наприклад, має більшу кількість користувачів, але її Bot API є значно обмеженішою у функціональності. Facebook Messenger має достатньо розвинену Bot API, але її інтеграція може бути складнішою через обмеження доступу та політики конфіденційності.

Telegram відомий своїм акцентом на безпеку та конфіденційність. Він використовує наскрізне шифрування для секретних чатів, що забезпечує високий рівень безпеки для користувачів. Це є важливим фактором для багатьох користувачів, які цінують приватність своїх комунікацій.

Telegram доступний у багатьох країнах і підтримує численні мови, що робить його привабливим для міжнародної аудиторії. Це дозволяє нашому чат-боту охопити широку аудиторію і забезпечити багатомовну підтримку.

Отже, Telegram є оптимальною платформою для розробки нашого чат-бота завдяки своїй великій користувацькій базі, зручності використання та найкращій у своєму класі Bot API. Його потужні інструменти для розробників, висока безпека та глобальна доступність роблять його ідеальним вибором для створення функціонального та ефективного чат-бота. Вибір Telegram дозволяє нам забезпечити високу якість взаємодії з користувачами, надійну безпеку та широкі можливості для кастомізації бота.

2.2.3. Вибір бібліотеки

Для розробки чат-бота у Telegram ми обрали бібліотеку Teloxide, яка є однією з найпотужніших та найзручніших бібліотек для створення ботів на Rust. Teloxide пропонує сучасний та зручний API, який спрощує процес розробки ботів, забезпечуючи широкий спектр функцій та інструментів.

«Teloxide – це асинхронний фреймворк для створення Telegram ботів на Rust, який розробляється спільнотою та підтримується на GitHub»[21]. Основні особливості та переваги Teloxide включають:

1. Асинхронність:

- Teloxide побудований на основі асинхронного програмування, що дозволяє ефективно обробляти велику кількість одночасних запитів, забезпечуючи високу продуктивність та швидкість роботи.

2. Модульність:

- Бібліотека модульна, що дозволяє розробникам вибирати лише ті компоненти, які необхідні для конкретного проекту. Це спрощує процес налаштування та знижує загальну складність коду.

3. Підтримка діалогів:

- Teloxide підтримує створення складних діалогових взаємодій з користувачами через діалогові сценарії. Це дозволяє легко реалізовувати складні логіки чат-ботів, такі як квізи, опитування та інші інтерактивні сценарії.

4. Документація та приклади:

- На GitHub доступна детальна документація та численні приклади використання бібліотеки, що робить її легкою у вивченні та освоєнні для розробників різного рівня.

5. Зручність використання:

- Teloxide пропонує зручний API для взаємодії з Telegram Bot API. Наприклад, створення команд, обробка повідомлень та інших дій стають простими та інтуїтивно зрозумілими завдяки добре продуманій архітектурі бібліотеки.

Для нашого проекту, де чат-бот використовує дерева відповідальності зі скінченним автоматом, Teloxide є особливо зручним вибором. Бібліотека дозволяє ефективно управляти станами користувачів та переходами між ними завдяки своїй підтримці діалогів та станів. Це можна легко реалізувати через State enum у файлі main.rs:

```
pub enum State {
    #[default]
    Start,
    ReceiveFullName,
    RecieveGameChoice,
    StressedWordsQuizRecieveAmountOfQuestions,
    StressedWordsQuiz {
        quiz: quiz::Quiz,
        question_number: usize,
        score: usize,
    },
```

```
}
```

Завдяки Teloxide, створення та обробка діалогів стають простими та зрозумілими. Наприклад, налаштування гілок для обробки різних станів:

```
Dispatcher::builder(bot, Update::filter_message()
    .enter_dialogue::<Message, ErasedStorage<State>, State>()
    .branch(dptree::case![State::Start].endpoint(start))

.branch(dptree::case![State::ReceiveFullName].endpoint(receive_full_name))
```

У цьому прикладі кожна гілка відповідає конкретному стану і обробляє наступні дії користувача через відповідний endpoint.

У підсумку, Teloxide є відмінним вибором для створення чат-бота у Telegram на Rust завдяки своїй асинхронній природі, модульності, підтримці діалогів та зручності використання. Бібліотека надає всі необхідні інструменти для реалізації складних логік і сценаріїв взаємодії з користувачами, що робить процес розробки ефективним та приємним.

2.2.4. Вибір LLM (Large Language Model)

В епоху стрімкого розвитку штучного інтелекту, Large Language Models (LLMs) стали ключовими інструментами для розробки інтелектуальних систем взаємодії, таких як чат-боти. Існує кілька передових моделей LLM, кожна з яких має свої унікальні особливості та переваги. У цьому підрозділі ми розглянемо доступні моделі, їхні характеристики, вартість і якість, а також обґрунтуємо вибір GPT-3.5 від OpenAI для нашого проекту.

Аналіз альтернативних моделей:

Claude 3:

Claude 3, розроблена компанією Anthropic, [22] орієнтована на забезпечення безпечних і етичних відповідей. Ця модель особливо цінна для застосувань, де важливо уникнути шкідливих або неточних відповідей. За словами Anthropic, Claude 3 використовує спеціальні алгоритми для зменшення ризиків, пов'язаних із генерацією тексту. Однак, вартість доступу до цієї моделі може варіюватися залежно від обраного плану, що робить її менш доступною для невеликих проектів або стартапів .

Gemini:

Розроблена Google DeepMind, [23] модель Gemini інтегрується з екосистемою Google, що забезпечує глибоку інтеграцію з іншими продуктами та сервісами Google. Висока якість відповідей та гнучкість у налаштуванні роблять Gemini потужним інструментом для різних завдань обробки природної мови. Проте, вартість використання Gemini також залежить від обраного плану, що може бути обмежуючим фактором для деяких користувачів .

Llama 3:

Відкрита модель Llama 3 [24], розроблена Meta (Facebook), призначена для академічних і дослідницьких проектів. Її основна перевага — безкоштовний доступ для дослідницьких та особистих проектів, що робить її привабливою для дослідників та ентузіастів. Однак, для досягнення високої продуктивності та точності ця модель може вимагати додаткового налаштування та оптимізації, що може бути викликом для менш досвідчених розробників .

GPT-3.5 та GPT-4:

Моделі GPT, розроблені OpenAI [25], є одними з найбільш потужних і широко використовуваних у світі. GPT-3.5 забезпечує високу якість генерації тексту, що робить її придатною для широкого спектра завдань обробки природної мови. GPT-4 [26] є покращеною версією, яка пропонує ще вищу точність і кращу якість відповідей у складних завданнях. Проте, вартість використання GPT-4 значно вища, що робить її менш доступною для деяких користувачів .

Порівняння характеристик

табл.1

Модель	**Якість відповідей**	**Вартість**	**Доступність API**	**Інтеграція**
Claude 3	Висока	Залежить від плану	Відкрита	Фокус на безпеку
GPT-3.5	Висока	Нижча порівняно з GPT-4	Відкрита	Широка підтримка
GPT-4	Дуже висока	Висока	Відкрита	Широка підтримка
Gemini	Висока	Залежить від плану	Відкрита	Інтеграція з Google

Llama 3	Висока	Безкоштовна	Відкрита	Потребує налаштування

Вибір GPT-3.5 для нашого проєкту базується на кількох ключових факторах, які забезпечують відмінне співвідношення ціни та якості, легкість використання і широкі можливості інтеграції.

Ціна/якість:

GPT-3.5 забезпечує високий рівень продуктивності за значно нижчою вартістю порівняно з GPT-4. Це робить її доступною для широкого кола користувачів, включаючи стартапи та малі підприємства, які потребують потужних інструментів за доступною ціною.

Легкість використання:

OpenAI надає добре документовані інструменти та API для інтеграції GPT-3.5. Це полегшує процес налаштування і використання моделі, забезпечуючи швидкий старт проекту. Додатково, велика спільнота розробників і наявність численних прикладів та ресурсів роблять інтеграцію простою та ефективною.

Доступність:

GPT-3.5 доступна через OpenAI API, що дозволяє легко інтегрувати модель у різні додатки та сервіси. Це важливо для забезпечення безперебійної роботи чат-бота і швидкого вирішення можливих проблем.

Широка підтримка:

GPT-3.5 широко використовується в багатьох проектах, що забезпечує її стабільність та надійність. Вона здатна виконувати широкий спектр завдань, включаючи генерацію тексту, обробку природної мови, переклад, аналіз тональності та багато іншого.

Приклади застосування:

Багато великих компаній та стартапів успішно інтегрували GPT-3.5 у свої продукти, що демонструє її практичну цінність. Вона використовується у різних галузях, від обслуговування клієнтів до створення контенту, що підтверджує її універсальність і потужність.

Висновок**:**

Обираючи модель GPT-3.5 для нашого чат-бота, ми отримуємо відмінне співвідношення ціни та якості, легкість використання та широкі можливості інтеграції. Це дозволяє нам створити функціональний, ефективний і надійний чат-бот, забезпечуючи високий рівень взаємодії та задоволення.

2.2.5. Корпуси

У проєкт були залучені два корпуси. Корпус для частин мови — «Золотий стандарт — це розмічений корпус української мови, що містить тексти, які були вручну анотовані для забезпечення високої точності. Цей корпус використовується для розробки та оцінки моделей обробки природної мови (NLP).

Вміст корпусу включає різноманітні тексти, які анотовані з використанням системи Universal Dependencies (UD)»[28].

А також корпус для наголосів — Ukrainian Word Stress Dictionary. «Цей корпус є словником наголосів українських слів, який використовується для аналізу та розробки моделей обробки природної мови. Він містить інформацію про правильне наголошення слів» [29].

Обидва корпуси є важливими ресурсами для розробки чат-бота, що займається навчанням морфології української мови. Вони забезпечують дані для тренування моделей та створення інтерактивних навчальних завдань.

Спершу ніж працювати з корпусом в нашому проєкті, був написаний код для зчитування файлу, ось фрагмент коду:

```
pub fn new(file: File) -> Self {
    let conllu_doc: Vec<PartsSentence> = rs_conllu::parse_file(file)
        .filter(|sentence| sentence.is_ok())
            // We can unwrap safely here because we've already filtered out the errors
        .map(|sentence| sentence.unwrap())
        .map(|sentence| PartsSentence::new(sentence))
        .collect();
    Self {
        sentenses: conllu_doc,
    }
```

}

Використано CoNLL-U (Conference on Computational Natural Language Learning) — це стандартний формат для анотованих текстових корпусів, який використовується для подання граматичної та морфологічної інформації про кожне слово в реченні. Формат CoNLL-U є зручним для обробки та аналізу текстів, що робить його дуже корисним для розробки чат-ботів, особливо тих, які спрямовані на навчання мови та обробку природної мови (NLP).

Було написано фрагмент для генерації питань для користувача з використанням зчитуванних розмічених речень з корпусу:

```
fn generate_question_out_of_sentence(sentence: &rs_conllu::Sentence) -> quiz::Question {
    let words_to_be_asked_about = sentence
        .tokens
        .iter()
        .filter(|t| t.upos != Some(rs_conllu::UPOS::PUNCT))
        .collect::<Vec<_>>();
    let random_word = words_to_be_asked_about
        .choose(&mut rand::thread_rng())
        .unwrap();
    let text_sentence = sentence
        .meta
        .iter()
```

```
    .find(|m| m.starts_with("text = "))
    .map(|m| m.replace("text = ", ""))
    .expect("Original 'text' metadata field not found on the sentence");

let correct_answer = match random_word.upos {
    Some(rs_conllu::UPOS::ADJ) => "прикметник",
    Some(rs_conllu::UPOS::ADV) => "прислівник",
    Some(rs_conllu::UPOS::INTJ) => "вигук",
    Some(rs_conllu::UPOS::NOUN) => "іменник",
    Some(rs_conllu::UPOS::PROPN) => "власний іменник",
    Some(rs_conllu::UPOS::VERB) => "дієслово",
    Some(rs_conllu::UPOS::PRON) => "займенник",
    Some(rs_conllu::UPOS::ADP) => "прийменник",
    Some(rs_conllu::UPOS::CCONJ) => "сполучник",
    Some(rs_conllu::UPOS::SCONJ) => "підрядний сполучник",
    Some(rs_conllu::UPOS::AUX) => "допоміжне дієслово",
    Some(rs_conllu::UPOS::DET) => "детермінатив",
    Some(rs_conllu::UPOS::NUM) => "числівник",
    Some(rs_conllu::UPOS::PART) => "частка",
    Some(rs_conllu::UPOS::X) => "інше",
    Some(rs_conllu::UPOS::SYM) => "символ",
```

```
        Some(rs_conllu::UPOS::PUNCT) => "пунктуація",
        None => "інше",
    };
    let possible_answers = vec![
        "прикметник", "прислівник", "вигук", "іменник", "власний іменник",
            "дієслово", "займенник", "прийменник", "сполучник", "підрядний
сполучник",
        "допоміжне дієслово", "детермінатив", "числівник", "частка"
    ];
    let incorrect_answer = possible_answers
        .iter()
        .filter(|a| a != &&correct_answer)
        .choose(&mut rand::thread_rng())
        .unwrap();
    let answers = {
        let mut shuffled_answers = vec![
            quiz::Answer::new(correct_answer.to_string(), true),
            quiz::Answer::new(incorrect_answer.to_string(), false),
        ];
        shuffled_answers.shuffle(&mut rand::thread_rng());
        shuffled_answers
```

```
};

let text_sentence = text_sentence
    .split(" ")
    .map(|word| {
        let no_sep = word
            .chars()
            .filter(|c| c.is_alphanumeric())
            .map(|c| c.to_string())
            .collect::<Vec<_>>()
            .join("");
        if no_sep == random_word.form {
            return format!("<b><u>{}</u></b>", word);
        }
        return word.to_string();
    })
    .collect::<Vec<String>>()
    .join(" ");

let question_text = format!(
```

```
            "У реченні:\n\"{}\"\n\nЯкою частиною мови є підкреслене слово \"{}\"?",
        text_sentence, random_word.form
    );
    quiz::Question::new(question_text, answers)
}
```

Функція *generate_question_out_of_sentence* генерує питання для користувача на основі розміченого речення у форматі CoNLL-U. Вона вибирає випадкове слово з речення і формує питання про його частину мови. Парсинг CoNLL-U забезпечує нашому чат-боту доступ до детальної лінгвістичної інформації, що дозволяє створювати більш точні та корисні навчальні вправи для користувачів. Це сприяє глибшому розумінню мови та поліпшенню мовних навичок у інтерактивній формі.

1. Код фільтрує всі токени речення, виключаючи пунктуацію, і випадково вибирає одне слово для питання.
2. Знаходить метадані з оригінальним текстом речення.
3. За допомогою match визначається правильна частина мови для вибраного слова.
4. Генерується список можливих відповідей, серед яких вибирається одна правильна і одна неправильна. Потім ці відповіді перемішуються.
5. Речення розбивається на слова, і вибране слово підкреслюється та виділяється жирним шрифтом.
6. На завершення функція створює текст-питання, яке включає форматоване речення з підкресленим словом, і повертає об'єкт quiz::Question з текстом питання та можливими відповідями.

Отже, ця функція демонструє, як зчитувати розмічені дані з корпусу, аналізувати їх та генерувати навчальні питання для користувачів. Вона використовує рандомізацію для вибору слів, формує питання, надає правильну відповідь та кілька варіантів неправильних відповідей, забезпечуючи інтерактивний та ефективний процес навчання.

2.2. Реалізація чат-бота

Для створення Telegram боту необхідно знайти та запустити BotFather. *BotFather* – це бот, який допомагає створити нового бота, керує уже існуючими та допомагає їх налаштувати. Після запуску, BotFather пропонує використати певні команди, але необхідно обрати саме /newbot. Надалі все відбувається дуже просто:

1. Необхідно написати унікальне ім’я нового бота. Адже існування чат-ботів з однаковими іменами – неможливе. Якщо бот з такою назвою вже існує, то BotFather запропонує вигадати інше ім’я.

2. Написати назву бота із закінченням bot у кінці.

3. Отримати маркер доступу (token) до HTTP API.

Маркер доступу – це унікальний ключ, що складається із набору унікальних цифр та букв, і дає можливість керувати ботом та його функціями. Під час створення програмного забезпечення, розробник використовує цей ключ для ідентифікації бота. Для безпеки бота, маркер доступу, бажано не розповсюджувати.

Після створення бота, розробник отримує певний список команд, для кращого налаштування бота:

setname – встановити нове ім’я;

setdesctiption – змінити опис боту;

setabouttext – змінити інформацію щодо бота;

setuserpic – змінити інформацію у профілі бота;

setcommands – змінити список команд;

deletebot – видалити бота;

token – згенерувати новий токен авторизації;

revoke – анулювати токен;

Ці дії допоможуть створити та налаштувати бота. При виконанні усіх перелічених дій *BotFather* відправить повідомлення про успіх створення Telegram бота. Але для його навчання на виконання будь-яких функцій потрібно написати відповідний програмний код

Для реалізації всіх можливостей чат-бота, необхідно також отримати унікальний код власника бота, та код створеного каналу, для отримання файлів від студентів. Це потрібно для того, щоб надісланні дані отримувала лише певна людина, яка являється власником бота та каналу. Щоб отримати ці коди, потрібно скористатись додатковим Telegram ботом. Всі ці дані записуються кодом в окремому файлі, в якому також додається з’єднання із базою даних.

Початком першого етапу реалізації є створення бота в системі месенджера Телеграм та отримання API токену. Даний процес зображений на рис. 5.

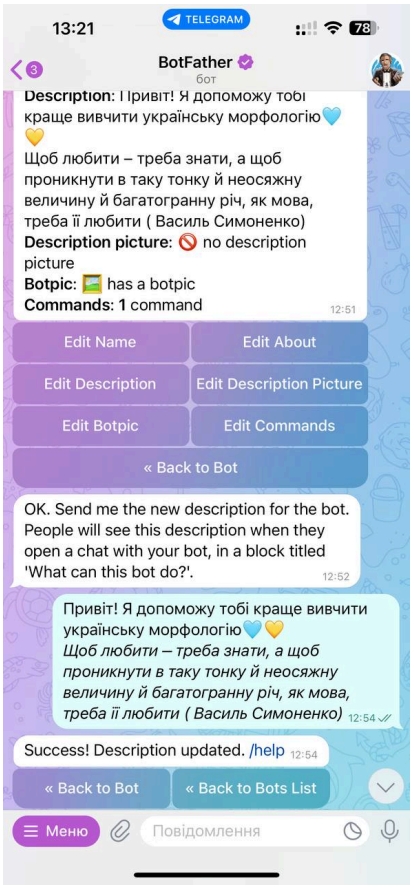


Рис. 5 – Отримання API токену для майбутнього бота

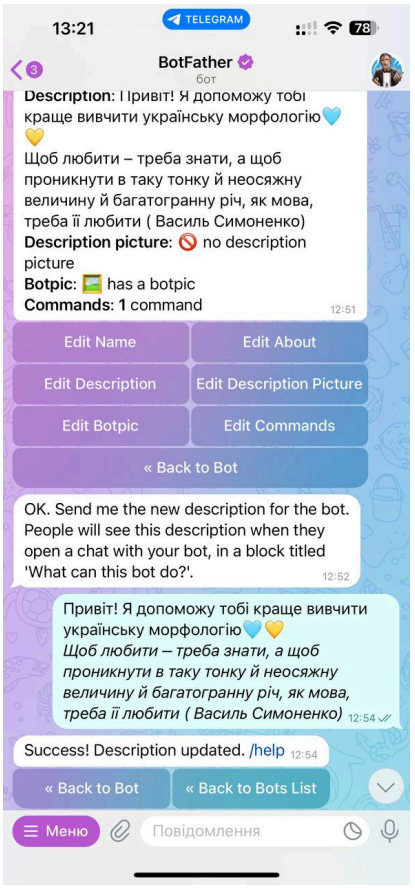


Рис.6 – Створення опису бота

Наступним кроком є редагування бота за допомогою сервісів телеграм месенджеру. На рис. 6-8. зображені процес додавання опису бота, який буде відображатися у користувача одразу після відриття бота, а також процес присвоєння боту власного розробленого логотипу та створення стандартних команд.

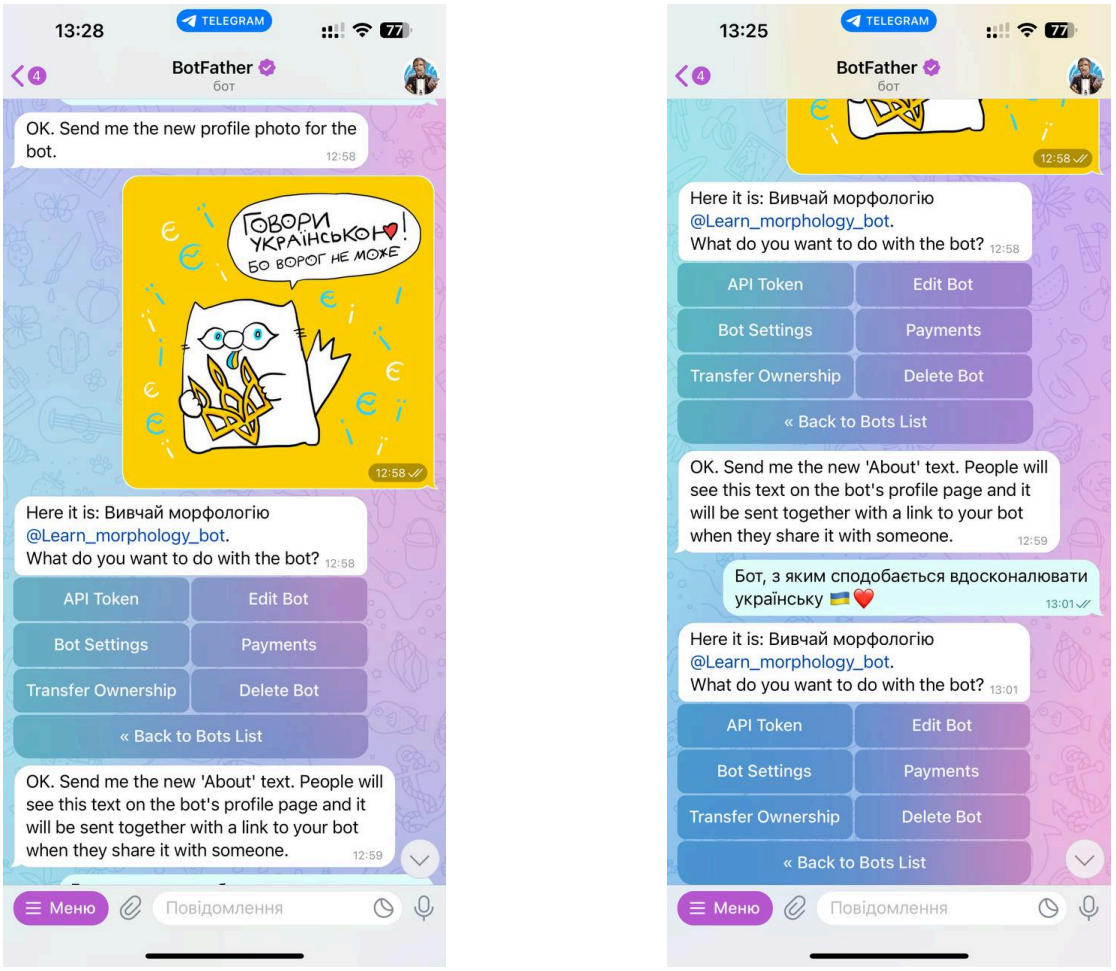


Рис. 7 – Присвоєння боту опису та створеного власного логотипу

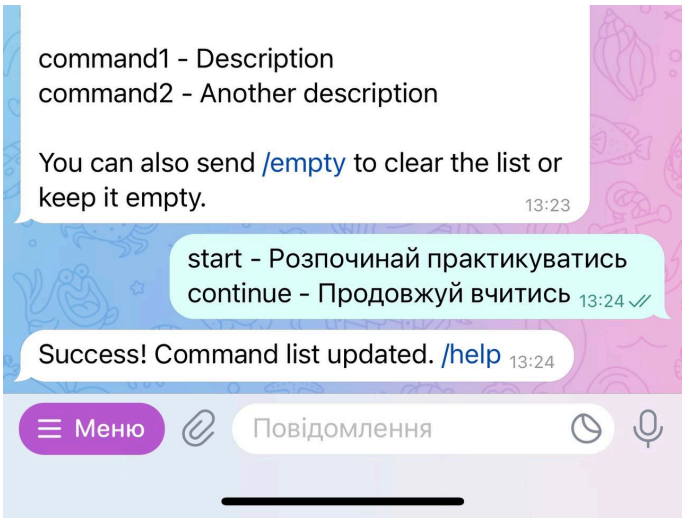


Рис. 8 – Створення стандартних команд чат-бота

2.3. Реалізація коду

Код реалізує чат-бот для вивчення української мови в Telegram. Основними завданнями бота є проведення тестів на знання наголосів та частин мови, надання прикладів речень та пояснення правильних відповідей. Для генерації відповідей використовується API ChatGPT.

Структуру складається з кількох основних модулів і файлів. Кожен файл відповідає за певну функціональність бота. Нижче наведено аналіз кожної частини коду та опис того, як вони взаємодіють між собою для забезпечення роботи бота.

ai_helper.rs

Файл *ai_helper.rs* містить логіку для взаємодії з API ChatGPT. Він відповідає за генерацію прикладів речень та пояснень відповідей на запитання користувачів. Основні компоненти цього файлу:

1. Структура QuizHelper: містить поля *personality* (тип особистості) і *chat_gpt* (екземпляр API ChatGPT).
2. Конструктор: ініціалізує новий екземпляр *QuizHelper* з наданими параметрами *chat_gpt* та *personality*.
3. Методи:

- *generate_example_for_stress_question*: генерує приклад речення для питання про наголос.
- *generate_reply_to_wrong_stress_answer*: генерує пояснення до неправильної відповіді на питання про наголос.
- *generate_reply_to_wrong_parts_answer:* генерує пояснення до неправильної відповіді на питання про частини мови.

4. Перерахування Personality: містить три варіанти особистості: *Shevchenko*, *Lesya, Franko,* і метод *get_personality*, який повертає ім'я особистості як рядок.

mod.rs

Файл *quiz/mod.rs* є основним файлом для визначення типів, які використовуються в квізах. Тут визначені структури для запитань, відповідей та самого квізу. Основні компоненти:

1. Модулі:
- *ai_helper*
- *parts*
- *stress*

2. Структури:
- *Quiz*: містить список запитань, поточне питання та поточний рахунок.
- *Question*: містить текст питання та список відповідей.
- *Answer*: містить текст відповіді та позначку, чи є відповідь правильною.

stress.rs

Файл *stress.rs* відповідає за зчитування і генерацію завдань і відповідей для тесту з наголосів. Основні компоненти:

1. Структура StressWords: містить список слів з наголосами.
2. Методи:
- *new*: зчитує слова з файлу.
- *get_random_word*: повертає випадкове слово для тесту.

3. Структура StressWord: містить слово з наголосом і без наголосу.
4. Методи:

- *new*: створює нове слово.
- *get_word_without_stress:* отримує слово без наголосу.
- *generate_question*: генерує питання для квізу з наголосами.

parts.rs

Файл *parts.rs* відповідає за зчитування і генерацію завдань і відповідей для тесту з частинами мови. Основні компоненти:

1. Структура PartsSentences: містить список речень з розміченими частинами мови.
2. Методи:

- *new*: зчитує речення з файлу.
- *get_random_sentence*: повертає випадкове речення для тесту.

3. Структура PartsSentence: містить розмічене речення.
4. Методи:

- *new*: створює нове розмічене речення.
- *generate_question*: генерує питання для квізу з частинами мови.

declension.rs

Файл *declension.rs* відповідає за зчитування і генерацію завдань і відповідей для тесту з відмінювання іменників. Основні компоненти

1. Структура Declension містить список іменників та їхні форми.
2. Методи:

- *new*: зчитує дані з JSON-файлу та фільтрує слова, залишаючи тільки іменники.

- *generate_question_out_of_noun:* генерує питання на основі випадково обраного відмінка та числа (однини або множини).

3. Структура *JsonWord* містить методи для конвертації JSON-об'єктів у структуру Noun.

4. Методи

- *to_noun:* конвертує JSON-об'єкт у Noun, якщо цей об'єкт є іменником.

5. Структура Noun представляє іменник і його форми.

6. Методи

- *generate_question_out_of_noun*: генерує питання, яке включає:

1. Вибір називного відмінка як базової форми слова.
2. Випадковий вибір відмінка та числа.
3. Пошук правильної відповіді.
4. Генерація можливих варіантів відповіді (за виключенням правильної відповіді та називного відмінка).

Основні кроки для створення питання:

1. Вибирається називний відмінок як базова форма слова.
2. Обирається випадковий відмінок та множина/однина.
3. Знаходиться правильна відповідь.
4. Генеруються всі можливі варіанти відповіді, які фільтруються для виключення правильної відповіді та називного відмінка.

main.rs

Файл *main.rs* є основним файлом проєкту, де оголошуються стани користувача та основні функції для обробки повідомлень. Основні компоненти:

1. Модуль quiz: містить підмодулі *ai_helper*, *parts* та *stress*.
2. Типи та константи:

- *QuizDialogue*
- *HandlerResult*
- *State*
- *GREETING_TEXT*
- *STRESSED_WORDS_GAME*
- *PARTS_OF_SPEECH_GAME*

3. Функція main: ініціалізує бота та встановлює з'єднання з базою даних.
4. Функції обробки повідомлень:

- *start*: надсилає привітання користувачу.
- *receive_full_name*: обробляє введене ім'я користувача.
- *receive_game_choice*: обробляє вибір гри користувачем.
- *receive_amount_of_questions*: обробляє кількість питань для тесту з наголосами.
- *stressed_quiz*: проводить тест з наголосами.
- *receive_amount_of_questions_parts_of_speech*: обробляє кількість питань для тесту з частинами мови.
- *parts_of_speech_quiz*: проводить тест з частинами мови.

Взаємодія між модулями

1. Ініціалізація: У функції *main* відбувається ініціалізація всіх необхідних компонентів, включаючи бота, з'єднання з базою даних, завантаження словників та ініціалізація ChatGPT.
2. Запити користувачів: Коли користувач взаємодіє з ботом, відправляючи повідомлення, бот обробляє їх за допомогою відповідних функцій обробки повідомлень.

3. Використання ChatGPT: Для генерації прикладів речень та пояснень до відповідей використовуються методи з *ai_helper.rs*, які взаємодіють з API ChatGPT.

4. Генерація питань: Для генерації питань використовуються методи з *stress.rs* та *parts.rs*, які зчитують дані з файлів та генерують відповідні питання.

2.4. Інтеграція з API ChatGPT

Бот використовує API ChatGPT для генерації прикладів речень та пояснень відповідей на запитання користувачів.

Код, який відповідає за реалізацію основної логіки чат-бота можна переглянути в репозиторії GitHub [44].

Структура QuizHelper

Структура *QuizHelper* створює об'єкт, який керує взаємодією між користувачем і ChatGPT. Вона містить два основні поля: *personality і chat_gpt*. Поле *personality* визначає тип особистості, яку бот використовує для надання відповідей, роблячи взаємодію з ботом більш цікавою та різноманітною. Поле *chat_gpt* є екземпляром API ChatGPT, який відповідає за генерацію тексту на основі заданих запитів.

Конструктор new

Конструктор *new* ініціалізує новий екземпляр *QuizHelper* з наданими параметрами *chat_gpt* та *personality.* Це дозволяє налаштувати бота відповідно до специфічних потреб користувачів, забезпечуючи його індивідуальність та гнучкість у взаємодії.

Метод generate_example_for_stress_question

Метод *generate_example_for_stress_question* приймає запитання типу *Question*, яке включає слово з наголосом. Він генерує речення з використанням цього слова, не вказуючи наголос, і використовує API ChatGPT для створення такого речення. Отриманий текст повертається користувачеві. Це допомагає учням краще розуміти контекст використання слів у реченнях, сприяючи ефективнішому засвоєнню матеріалу.

Метод generate_reply_to_wrong_stress_answer

Метод *generate_reply_to_wrong_stress_answer* обробляє запитання, на яке користувач відповів неправильно. Він знаходить правильну та неправильну відповідь у списку відповідей, а потім генерує пояснення, чому правильний наголос саме такий, використовуючи API ChatGPT. Пояснення повертається користувачеві, допомагаючи зрозуміти помилку і правильно засвоїти матеріал.

Метод generate_reply_to_wrong_parts_answer

Метод *generate_reply_to_wrong_parts_answer* обробляє запитання про частини мови, на яке користувач відповів неправильно. Він знаходить правильну та неправильну відповідь у списку відповідей і генерує пояснення, чому правильна частина мови саме така, і в чому була помилка користувача. Це пояснення також генерується за допомогою API ChatGPT і повертається користувачеві, допомагаючи краще зрозуміти граматичні правила та виправити свої помилки.

Перерахування Personality

Перерахування *Personality* містить три варіанти особистості: *Shevchenko*, *Lesya* та *Franko*. Метод *get_personality* повертає ім'я відповідної особистості у вигляді рядка, додаючи індивідуальність до відповідей бота. Це робить взаємодію з ботом більш цікавою та персоналізованою, залучаючи користувачів до процесу навчання.

Отже, використання ChatGPT у цьому боті значно підвищує його функціональність та ефективність. Завдяки інтеграції з API ChatGPT, бот здатен генерувати точні та релевантні відповіді, надавати пояснення до помилок користувачів та створювати приклади речень для кращого розуміння мовних правил. Це дозволяє робити навчальний процес інтерактивним, цікавим та індивідуалізованим. Застосування ChatGPT сприяє більш глибокому засвоєнню матеріалу користувачами, забезпечуючи миттєвий зворотний зв'язок і підтримку в режимі реального часу.

2.5. Опис роботи чат-бота

Morphology UA Test Bot - це чат-бот у Telegram, який розроблений для допомоги користувачам у вивченні української мови. Бот надає інтерактивні завдання і квізи, спрямовані на покращення знань з морфології та акцентології української мови.

Основні функції та можливості бота:

Привітання та початкове знайомство:

Користувач починає взаємодію з ботом шляхом надсилання повідомлення. Бот відповідає привітанням і запитує ім'я користувача, що створює персоналізований підхід до навчання (рис.9).

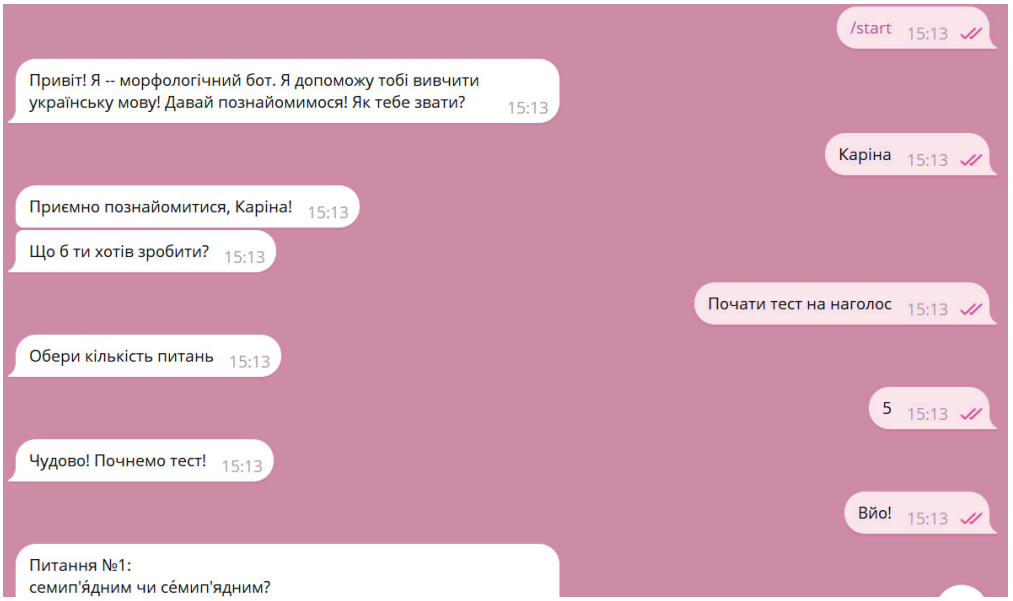


рис.9

Приклад: "Привіт! Я - морфологічний бот. Я допоможу тобі вивчити українську мову! Давай познайомимося! Як тебе звати?"

Вибір режиму навчання:

Бот пропонує користувачам різні режими навчання, такі як тренування наголосів, граматики, лексики тощо. Користувач може обрати потрібний режим за допомогою інтерактивних кнопок або команд (рис.10).

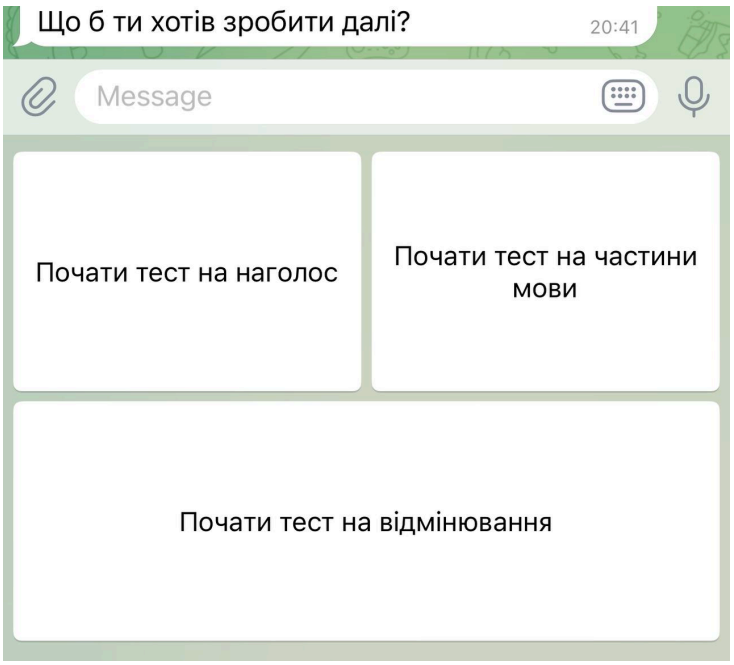


рис.10

Приклад команд: "Тренувати наголоси", "Тренувати морфологію", "Тренувати відмінювання".

Інтерактивні завдання та квізи:

Після вибору теми запитань та кількості завдань (рис.11), бот надає завдання, де користувач повинен вибрати правильний варіант наголосу, форми слова або відповісти на питання. Завдання представлені у форматі тестів з кількома варіантами відповідей. Вправи побудовані так, щоб користувач зміг у певному контексті відповісти на запитання бота (рис.12).

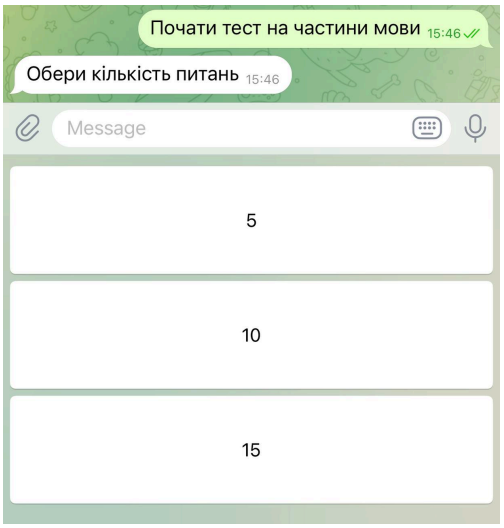


рис.11 - вибір кількості запитань

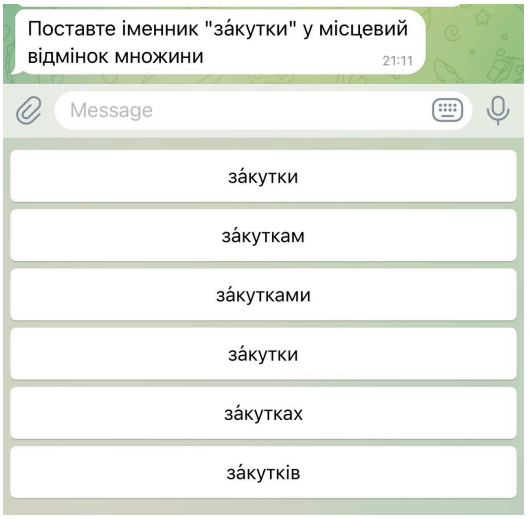


рис.12 - вибір варіантів відповіді

Миттєвий зворотний зв'язок:

Після відповіді на запитання бот надає миттєвий зворотний зв'язок, повідомляючи, чи була відповідь правильною або ні, і надає пояснення або додаткову інформацію.

Приклад реакції: "Неправильно! На мою думку, мій шанований учень, ти допустив невелику помилку. Слово "апдейт" у цьому реченні є іменником, а не дієсловом. Іменник вказує на конкретний предмет, явище або ідею, у цьому випадку на оновлення чи зміни. Дієслово, натомість, вказує на дію або стан.”

Персоналізація та прогрес:

Бот зберігає інформацію про прогрес користувача, що дозволяє адаптувати наступні завдання до його рівня знань та успіхів. Це сприяє більш ефективному навчанню та мотивації користувачів (рис.13).

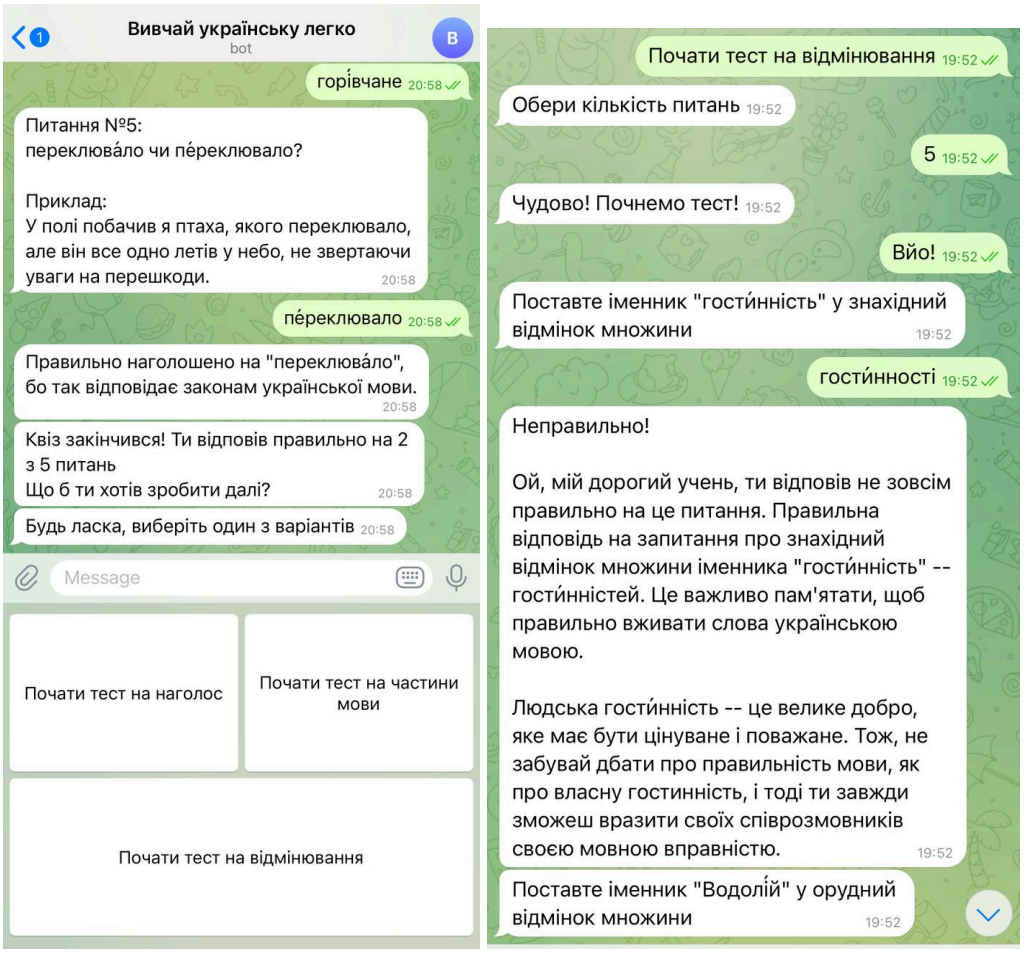


рис.13

Чат-бот у відкритому доступі в телеграмі, кожен може спробувати інтерактивне практикування української мови [45].

Висновки до Розділу 2

У процесі розробки чат-бота для вивчення української мови ми зосередилися на створенні ефективної та зручної системи, яка допомагає

користувачам практикувати вправи з української мови. Використання мови програмування Rust забезпечило високу продуктивність та надійність роботи бота, а інтеграція з платформою Telegram дозволила створити зручний інтерфейс для взаємодії з користувачами.

Основними завданнями, які були вирішені в рамках цього розділу, стали:

1. Розробка базової архітектури бота: створення структури програми, визначення основних модулів та функціональних частин бота.
2. Реалізація основних функцій: розробка алгоритмів для аналізу та обробки користувацьких запитів, генерація відповідей та забезпечення коректного функціонування бота.
3. Інтеграція з Telegram API: налаштування взаємодії з API для отримання та відправки повідомлень, а також для обробки подій.
4. Тестування та оптимізація: проведення тестів для виявлення та виправлення помилок, оптимізація коду для покращення продуктивності та надійності.

Результати показали, що обраний підхід до розробки є ефективним і дозволяє створити зручний та функціональний інструмент для вивчення української мови. Чат-бот здатен надавати користувачам зворотний зв'язок в реальному часі, що сприяє більш ефективному засвоєнню матеріалу та підвищенню мотивації до вивчення мови.

Таким чином, можна зробити висновок, що практична реалізація чат-бота для вивчення української мови є успішною і досягнуті результати відповідають поставленим цілям проекту.

Висновок

У першій частині дипломної роботи було розглянуто теоретичне обґрунтування вибору теми дослідження, яке охоплює історію розвитку чат-ботів, їх типи та сфери застосування, а також аналіз існуючих чат-ботів для вивчення української мови.

Розвиток чат-ботів пройшов довгий шлях від їхнього зародження до сучасних складних форм. Ранні приклади, такі як ELIZA та PARRY, заклали основу для розмовних агентів, хоча їхня функціональність була обмеженою. Завдяки досягненням у галузі штучного інтелекту та машинного навчання, сучасні чат-боти перетворилися на потужні інструменти, здатні автоматизувати сервіси, покращувати користувацький досвід та забезпечувати ефективну комунікацію. Ці технології значно розширили можливості чат-ботів, зробивши їх незамінними в різних сферах, включаючи обслуговування клієнтів, охорону здоров'я та освіту.

Аналіз існуючих чат-ботів для вивчення української мови у Telegram, таких як «Баба Катря» та «Твоя Мова», показав, що ці боти є ефективними інструментами для покращення мовних навичок користувачів. Вони використовують різні методи інтерактивного навчання, надають миттєвий зворотний зв'язок та забезпечують персоналізований підхід до кожного користувача. Водночас, було виявлено певні недоліки, такі як обмежений обсяг матеріалу та залежність від стилю спілкування.

Загалом, розуміння історії розвитку, типів та сфер застосування чат-ботів, а також аналіз існуючих рішень для вивчення української мови, дозволяє зробити висновок про необхідність подальшого розвитку та вдосконалення чат-ботів. Це створює перспективи для розробки нових, більш функціональних та ефективних інструментів, які можуть бути корисними в різних сферах життя та діяльності.

Перед початком реалізації коду було створено UserFlow, що є блок-схемою, яка відображає шлях користувача або порядок дій, які користувач робить при використанні продукту. Розробка такої схеми допомагає зробити чат-бот більш інтуїтивним і простим у використанні для користувача. Важливо зрозуміти основні цілі користувача, взаємодіючи з чат-ботом. Наприклад, користувач може бажати почати тест на наголос або частини мови, отримати зворотний зв'язок на відповідь. Процес взаємодії можна розділити на кілька основних етапів: привітання і запит імені, вибір типу тесту, вибір кількості запитань, проходження тесту, отримання результатів і зворотного зв'язку.

У сучасному світі програмування вибір мови для конкретного проєкту є вирішальним кроком. Розглядаючи мови програмування для створення чат-бота, були враховані кілька популярних варіантів: Python, JavaScript, Java, C++ та Rust. Вибір мови визначався її перевагами та недоліками щодо продуктивності, безпеки, простоти використання та відповідності вимогам проєкту.

Rust був обраний завдяки своїм унікальним перевагам, таким як безпека пам'яті, висока продуктивність та розвинена система типізації, що робить його ідеальним вибором для розробки чат-бота. Завдяки цим характеристикам Rust забезпечує надійну, продуктивну та зручну платформу для створення складних і ефективних програмних рішень.

Telegram був обраний як платформа для чат-бота завдяки своїй великій користувацькій базі, зручності використання та найкращій у своєму класі Bot API. Висока безпека та глобальна доступність Telegram роблять його оптимальним вибором для створення функціонального та ефективного чат-бота.

Для розробки чат-бота була обрана бібліотека Teloxide, яка є потужною та зручною для створення ботів на Rust. Teloxide забезпечує асинхронну обробку запитів, модульність та підтримку складних діалогів, що спрощує процес розробки ботів та забезпечує високу продуктивність і ефективність.

Для інтеграції з ботом було обрано модель GPT-3.5 від OpenAI. Цей вибір базувався на кількох ключових факторах, таких як висока якість відповідей, доступність API, широкі можливості інтеграції та відмінне співвідношення ціни та якості. GPT-3.5 забезпечує точні та релевантні відповіді, допомагаючи користувачам краще розуміти та засвоювати матеріал.

Для розробки чат-бота використовувалися два корпуси: «Золотий стандарт» та Ukrainian Word Stress Dictionary. Ці корпуси забезпечують дані для тренування моделей та створення інтерактивних навчальних завдань. Використання стандартного формату CoNLL-U забезпечує доступ до детальної лінгвістичної інформації, що дозволяє створювати точні та корисні навчальні вправи.

Реалізація чат-бота включала створення основних модулів для обробки запитів користувачів, генерації питань та відповідей, а також інтеграцію з API ChatGPT для генерації прикладів речень та пояснень відповідей. Використання ChatGPT значно підвищило функціональність та ефективність бота, роблячи процес навчання інтерактивним, цікавим та індивідуалізованим.

Розробка та впровадження чат-бота для вивчення української мови в Telegram є складним, але надзвичайно перспективним завданням. Використання сучасних технологій, таких як Rust, Teloxide та GPT-3.5, дозволяє створити функціональний, ефективний та зручний у використанні

інструмент для навчання. Завдяки інтеграції з потужними моделями та базами даних, цей чат-бот здатен забезпечити високу якість навчання та задоволення користувачів.

Список використаної літератури

1. Шимків К. М., Костіков М. П. Створення чат-бота для вивчення морфології української мови // Матер. X Міжнар. наук.-техн. Internet-конф. «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», 24 листоп. 2023 р. – К.: НУХТ, 2023. – С. 218–219.
2. Romero, C., & Ventura, S. (2019). Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6), 601-618.
3. Lee, C., & Wang, M. (2018). A Study on the Use of Chatbots for Customer Service: A Case of Taiwan's Banking Industry. *Journal of Management Information Systems*, 35(4), 131-150.
4. Smith, J., & Anderson, K. (2020). Chatbots in Modern Customer Service: Advancements and Challenges. *Journal of Artificial Intelligence Research*, 58(2), 123-135.
5. Berry, D. M. (2023). The Limits of Computation: Joseph Weizenbaum and the ELIZA Chatbot. *Weizenbaum Journal of the Digital Society*, 3(3). https://weizenbaum-institute.de/journals/digital-society.
6. Archive.org. "PARRY: Kenneth Mark Colby, Stanford University." https://archive.org/details/parry-stanford-university.
7. Educational Technology Journal. "Role of AI chatbots in education: systematic literature review." https://edtechjournal.org/ai-chatbots-in-education.
8. MDPI. (2024). Chatbot Design and Implementation: Towards an Operational Model for Chatbots. *Information*, 15(4), 226. https://doi.org/10.3390/info15040226.
9. Renaissancerachel.com. The Ultimate Guide to Chatbots: Design, Implementation, and Best Practices. https://renaissancerachel.com.
10. Five Different Types of Chatbot. [Електронний ресурс] // medium.com – 2019. – Режим доступу: https://medium.com/voiceui/five-different-types-of-chatbot17bb255b23b.

11. IBM Watson Health. (2023). The Role of AI in Healthcare: Improving Patient Outcomes. https://www.ibm.com/watson-health.
12. Accenture. (2022). The Future of Banking: Chatbots and AI in Financial Services. https://www.accenture.com/us-en/insights/financial-services/future-of-banking.
13. Educause. (2023). AI and Chatbots in Higher Education: Trends and Best Practices. https://www.educause.edu/research-and-publications.
14. Баба Катря Telegram Bot https://t.me/KatryaBot
15. Всеосвіта. https://vseosvita.ua/c/news/post/36130.
16. Твоя Мова Telegram Bot. https://t.me/tvoyamova.
17. Матвеева Н. Ю., Золотарюк А. В. (2018). Технологии создания и применения чат-ботов. *Научные записки молодых исследователей*, 1, 28-30.
18. IBM. (2023). Advantages of Rust for System Programming. *IBM Documentation*. https://www.ibm.com/docs.
19. Benchmarks Game. (2023). Programming Language Benchmarks. *Benchmarks Game*. https://benchmarksgame-team.pages.debian.net/benchmarksgame/.
20. Telegram. (2023). Telegram Bot API Documentation. https://core.telegram.org/bots/api.
21. The Verge. (2023). Telegram hits 700 million monthly active users. *The Verge*. https://www.theverge.com/2023/6/telegram-700-million-users.
22. Teloxide GitHub Repository. https://github.com/teloxide/teloxide.
23. Anthropic. (2023). Claude Documentation. *Anthropic*. https://www.anthropic.com/claude.
24. Google DeepMind. (2023). Gemini Model Information. *Google DeepMind*. https://www.deepmind.com/research/case-studies/gemini.
25. Meta. (2023). Llama 3 Documentation. *Meta*. https://ai.facebook.com/research/models/llama.
26. OpenAI. (2023). OpenAI API Documentation. *OpenAI*. https://beta.openai.com/docs/.

27. OpenAI. (2023). Introducing GPT-4. *OpenAI*. https://openai.com/research/gpt-4.
28. Universal Dependencies GitHub Repository. https://github.com/UniversalDependencies/UD_Ukrainian-IU/archive/dev.zip.
29. Ukrainian Word Stress Dictionary GitHub Repository. https://github.com/lang-uk/ukrainian-word-stress-dictionary.
30. Dale, R. (2016). The return of the chatbots. *Natural Language Engineering*, 22(5), 811-817.
31. Kopřiva, J., & Mareš, J. (2021). Building Chatbots with Rust: A Comprehensive Guide. *Journal of Computer Science and Technology*, 36(3), 567-582.
32. Petrov, P., & Ivanov, D. (2020). High-Performance Language Learning Bots Using Rust. *Proceedings of the International Conference on Computational Linguistics*, 112-125.
33. Lang-uk team. (2020). Ukrainian Word Stress Dictionary: A Computational Resource. *GitHub Repository*. https://github.com/lang-uk/ukrainian-word-stress-dictionary.
34. Blandy, J., Orendorff, J., & Tindall, L. F. S. (2021). Programming Rust: Fast, Safe Systems Development. *O'Reilly Media*. https://www.oreilly.com/library/view/programming-rust-2nd/9781492052586/.
35. Kaihlavirta, V. (2021). Mastering Rust. *Packt Publishing*. https://www.packtpub.com/product/mastering-rust/9781785885303.
36. Gupta, A. (2021). Natural Language Processing with Rust. *Manning Publications*. https://www.manning.com/books/natural-language-processing-with-rust.
37. Troutwine, B. L. (2021). Hands-On Concurrency with Rust. *Packt Publishing*. https://www.packtpub.com/product/hands-on-concurrency-with-rust/9781788399975.
38. Telegram Bot Development with BotFather. *Telegram*. https://telegram.org/blog/botfather.
39. Advanced Telegram Bot Programming. *Tech Press*.

40. Designing a Chatbot for Contemporary Education: https://www.mdpi.com/2078-2489/14/9/503
41. Онуфрієнко, С. В. (2020). Інформаційні технології в освіті: використання чат-ботів для навчання. Київ: Видавництво КНУ імені Тараса Шевченка.
42. Болюбаш, Я. М. (2018). Використання чат-ботів у викладанні іноземних мов. Наукові записки. Серія: Педагогічні науки, (180), 23-29.
43. Семенець, О. І. (2022). Чат-боти для вивчення української мови: аналіз ефективності. Мовознавство, (10), 67-74.
44. Посилання на репозиторій з кодом програми https://github.com/karinashymkiv/morphology_bot/tree/main
45. Посилання на чат-бот https://t.me/morphology_ua_test_bot