Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня
магістра
(назва освітньо-кваліфікаційного рівня)

студента	*Ярощука Олексія Романовича*
	(ПІБ)
академічної групи	*121М-22-2*
	(шифр)
спеціальності	*121 Інженерія програмного забезпечення*
	(код і назва спеціальності)
освітньої програми	*«121 Інженерія програмного забезпечення»*
	(назва освітньої програми)
на тему:	*Розробка та обґрунтування телеграм бота – вивчення англійської мови із застосуванням Python*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
розділів кваліфікаційної роботи				
спеціальний	*проф. Мещеряков Л.І.*			

Рецензент				

Нормоконтролер	*доцент Гуліна І.Г.*			

Дніпро
2023

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)
_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)
« » _____ 20 23 Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

студенту *121М-22-2* *Ярощуку Олексію Романовичу*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Розробка та обґрунтування телеграм бота – вивчення англійської мови із застосуванням Python*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 09.10.2023 р. № 1224 -с

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – розробка та обґрунтування телеграм бота, спрямованого на ефективне вивчення англійської мови.

Предмет досліджень – сучасні підходи та технології, що застосовуються для вивчення іноземних мов та їхнє практичне впровадження в розробку телеграм бота.

Мета НДР – бот буде надавати користувачам інтерактивні завдання, вправи та ресурси для ефективного самостійного вивчення англійської мови.

Вихідні дані для проведення роботи – відкриті джерела, літературні джерела та онлайн-ресурси з метою аналізу існуючих підходів до вивчення іноземних мов та програмування на мові Python.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень полягає в розробці та застосуванні сучасних методів реалізації програмного забезпечення для удосконалення існуючих засобів зміни стану ресурсів тестування за рахунок збільшення ступеня автоматизації та безпеки використання ПЗ з прямим доступом до бази даних.

Практична цінність результатів полягає в інтеграції передових методів навчання та технологій в розроблений телеграм бот, що дозволяє виокремити його серед існуючих підходів до вивчення англійської мови. Такий підхід робить навчання більш

інтерактивним, персоналізованим та ефективним, сприяючи покращенню результативності користувачів у процесі освоєння мовних навичок.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Вимоги до роботи включають функціональний та інтерактивний бот, що застосовує мову програмування Python та забезпечує індивідуалізовані завдання для ефективного вивчення англійської мови. Результати виконання роботи повинні відзначатися новизною, інноваціями в навчанні та покращенням результативності користувачів, підкреслюючи переваги розробленого бота серед існуючих методів вивчення мови.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець**
Вивчити існуючі месенджери та технології розробки ботів	05.09.2023-24.09.2023
Розробити загальну архітектуру системи, вибрати апаратне та програмне забезпечення	25.09.2023-11.10.2023
Розробити конфігурацію апаратного та програмного забезпечення, розробити та реалізувати прототип системи, провести тестування працездатності	12.10.2023-07.11.2023

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект від реалізації результатів роботи очікується позитивним завдяки тому що бот може зменшити витрати на індивідуальні курси англійської мови, забезпечуючи доступність вивчення для більшого кола користувачів.

Соціальний ефект від реалізації результатів роботи очікується позитивним завдяки тому що бот сприяє глобальному навчанню, забезпечуючи можливість навчання англійської мови широкому колу людей, незалежно від їхнього фізичного місцезнаходження чи фінансових можливостей.

Завдання видав _____ *Мещеряков Л.І.*
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Стешенко А.А.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *05.09.2023 р.*

Термін подання кваліфікаційної роботи до ЕК *09.12.2023*

РЕФЕРАТ

Пояснювальна записка: 63 с., 17 рис., 8 дод., 17 джерел.

Об'єкт розробки: розробка та обґрунтування телеграм бота, спрямованого на ефективне вивчення англійської мови.

Предмет дослідження: сучасні підходи та технології, що застосовуються для вивчення іноземних мов та їхнє практичне впровадження в розробку телеграм бота.

Мета кваліфікаційної роботи: бот буде надавати користувачам інтерактивні завдання, вправи та ресурси для ефективного самостійного вивчення англійської мови.

Методи дослідження. Для вирішення визначених задач було використано різноманітні методи, а саме: аналіз літературних джерел, дослідження відкритих джерел, експериментальний підхід, аналіз даних користувачів, експертна оцінка, програмування, аналіз взаємодії з користувачем.

Наукова новизна полягає в новому підході до вивчення англійської мови через розробку та застосування телеграм бота, який використовує мову програмування Python. Цей підхід надає інтерактивні та індивідуалізовані методи навчання, сприяючи ефективному процесу освоєння мовних навичок.

Практичне значення роботи. Інтеграція передових методів навчання та технологій в розроблений телеграм бот, що дозволяє виокремити його серед існуючих підходів до вивчення англійської мови. Такий підхід робить навчання більш інтерактивним, персоналізованим та ефективним, сприяючи покращенню результативності користувачів у процесі освоєння мовних навичок.

Список ключових слів: телеграм бот, вивчення мов, python, інтерактивні завдання, ефективність, освітні технології, інноваційний підхід, індивідуалізація навчального процесу, соціальна доступність.

ABSTRACT

Explanatory note: 63 pages, 17 pictures, 8 appendices, 17 sources.

Object of research: development and substantiation of a Telegram bot aimed at effective English language learning.

Subject of research: modern approaches and technologies used for learning foreign languages and their practical implementation in the development of a Telegram bot.

Purpose of Master's thesis: the bot will provide users with interactive tasks, exercises and resources for effective self-study of English.

Research methods. Various methods were used to solve the identified problems, namely: analysis of literary sources, research of open sources, experimental approach, analysis of user data, expert evaluation, programming, analysis of interaction with the user.

Originality of research is a new approach to learning English through the development and application of a Telegram bot that uses the Python programming language. This approach provides interactive and individualized learning methods, contributing to the effective process of learning language skills.

Practical value of the results. The integration of advanced teaching methods and technologies into the developed Telegram bot, which makes it stand out among existing approaches to learning English. This approach makes learning more interactive, personalized and effective, contributing to the improvement of users' effectiveness in the process of learning language skills.

Keywords: Telegram bot, language learning, python, interactive tasks, efficiency, educational technologies, innovative approach, individualization of the educational process, social accessibility.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

API – Application Programming Interface – Інтерфейс програмування програми;

HTTPS – Hyper Text Transfer Protocol Secure – Розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки;

BOT – Robot – Віртуальний робот чи штучний інтелект;

REST – Representational State Transfer – Передача репрезентативного стану;

DB – Data Base – База даних;

SQL – Structured Query Language – Мова структурованих запитів;

ПК – персональний комп’ютер;

ОС – операційна система;

ПЗ – програмне забезпечення;

СУБД – система управління базою даних;

ЗМІСТ

с.

ВСТУП

Актуальність дослідження. Нині вже важко уявити собі життя без інтернету. У ньому щомиті відбуваються мільйони подій, починаючи від простого пошуку інформації, спілкування та перегляду новин, закінчуючи глобальними подіями на кшталт аукціонів або запровадження державних інтернет послуг. В даний час важко знайти кампанію, яка не має свою групу в соціальних мережах або власний сайт. Але в цій роботі нас цікавитиме спілкування, яке з реального світу поступово переходить у віртуальне завдяки доступності месенджерів та простоті їх використання.

Актуальність випускної кваліфікаційної роботи зумовлена високою популярністю месенджерів та засобів автоматизації як чат-боти серед користувачів мережі Інтернет. Чат-боти дозволяють спростити щоденні рутинні завдання, такі як отримання інформації про погоду, пробки, останні новини та інші. Головною перевагою щодо класичних додатків є можливість поєднання всіх можливостей на платформі одного месенджера.

Також у сучасному світі володіння іноземними мовами є надзвичайно важливою навичкою, що відкриває безліч можливостей для освіти, кар'єри та культурного обміну. Ця робота присвячена створенню робота здатного навчити англійської мови.

Мета дослідження: бот буде надавати користувачам інтерактивні завдання, вправи та ресурси для ефективного самостійного вивчення англійської мови.

Завдання дослідження: Для досягнення зазначеної цілі в дослідженні були визначені та вирішені наступні завдання:

1. Аналіз потреб користувачів

Визначення основних вимог та очікувань цільової аудиторії щодо навчання англійської мови через телеграм бот.

2. Вивчення існуючих підходів

Огляд та аналіз існуючих методик та платформ для вивчення мов, зокрема через телеграм ботів, з метою визначення найбільш перспективних рішень.

3. Формулювання вимог

Розробка конкретних вимог до функціоналу бота, враховуючи вивчені підходи та потреби користувачів.

4. Розробка та тестування бота

Створення прототипу бота та проведення тестування для перевірки функціональності та взаємодії з користувачами.

5. Оптимізація та покращення

Визначення слабких сторін бота та його функціоналу, а також внесення відповідних покращень для забезпечення оптимальної ефективності.

6. Експертна оцінка

Залучення експертів у галузі навчання та програмування для отримання обгрунтованих рекомендацій та оцінки результатів.

7. Тестування з користувачами

Проведення тестувань з реальними користувачами для оцінки ефективності та отримання фідбеку.

8. Аналіз результатів

Обробка та аналіз отриманих даних для визначення ступеня досягнення поставленої мети та виокремлення можливих напрямків подальшого розвитку.

Об’єкт дослідження: розробка та обґрунтування телеграм бота, спрямованого на ефективне вивчення англійської мови.

Методи дослідження. Для вирішення визначених задач було використано різноманітні методи, а саме: аналіз літературних джерел, дослідження відкритих джерел, експериментальний підхід, аналіз даних користувачів, експертна оцінка, програмування, аналіз взаємодії з користувачем.

Наукова новизна полягає в новому підході до вивчення англійської мови через розробку та застосування телеграм бота, який використовує мову програмування Python. Цей підхід надає інтерактивні та індивідуалізовані методи навчання, сприяючи ефективному процесу освоєння мовних навичок.

Практичне значення. Інтеграція передових методів навчання та технологій в розроблений телеграм бот, що дозволяє виокремити його серед

існуючих підходів до вивчення англійської мови. Такий підхід робить навчання більш інтерактивним, персоналізованим та ефективним, сприяючи покращенню результативності користувачів у процесі освоєння мовних навичок.

Особистий внесок автора:

1. Розробка концепції: Сформулювання та визначення концепції телеграм бота для вивчення англійської мови з урахуванням інтерактивності та індивідуалізації навчання.
2. Програмування бота: Реалізація функціоналу бота, використовуючи мову програмування Python, з урахуванням встановлених вимог та специфікацій.
3. Експериментальні дослідження: Проведення пілотних експериментів та тестувань для визначення ефективності та придатності бота для вивчення мови.
4. Аналіз результатів: Обробка та аналіз отриманих даних з метою формулювання висновків та рекомендацій щодо подальшого вдосконалення та розвитку проекту.
5. Взаємодія з експертами: Комунікація та залучення експертів для здобуття обгрунтованих оцінок та рекомендацій щодо вдосконалення роботи.

Структура і обсяг роботи. Робота складається з вступу, трьох розділів і висновків. Містить 63 сторінки, в тому числі 44 сторінки тексту основної частини з 17 рисунками, списку використаних джерел з 17 найменуваннями на 2 сторінках, 8 додатків на 17 сторінках.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ. ПОСТАНОВКА ЗАВДАННЯ

1.1 Месенджери

Свій розвиток інтернет-сервіси для спілкування почали з чатів, потім месенджери, потім соціальні мережі, але нещодавно месенджери знову очолили список найперспективніших сервісів. Це підтверджено дослідженнями компаній "We Are Social" та "Hootsuite" у 2021 році: кількість користувачів месенджерів у світі збільшилася на 13%. Водночас середня кількість месенджерів, які використовуються одним користувачем, збільшилася за 2021 рік із 3 до 4 штук. Дослідження компанії «Білайн» показали, що месенджери використовуються переважно дорослими, так на частку осіб до 18 років припадає 7,7% від загальної кількості аудиторії, користувачі віком від 18 до 25 років становлять 9,8%, від 25 до 35 років 32,3%, від 35 до 45 років – 26,8%. При цьому 13,1% користувачів месенджерів перебувають у віці від 45 до 55 років, а 7,6% від 55 до 64 років. І лише 2,7% аудиторії месенджерів перебуває у віці старше 64 років.

Причина повторної хвилі популярності месенджерів — самоізоляція, яка змусила більше людей використовувати месенджери для спілкування з близькими та колегами, що сильно вплинуло на зростання трафіку всередині додатків, за рік він збільшився більш ніж у чотири рази.

Telegram є програмою створеною мовою програмування C++. Клієнтські програми можуть бути встановлені на різні платформи: Windows, Linux, MacOS, IOS, Android також існує веб-версія. Користувачі можуть обмінюватися повідомленнями та різними файлами. Для роботи програма використовує серверну частину із закритим кодом, який розташований на серверах, які знаходяться в різних кінцях планети.

Головною особливістю Telegram вважається спеціально розроблений протокол шифрування MTProto та можливість створення секретних чатів.

Секретні чати потрібні в тих випадках, коли надіслане повідомлення має бути максимально захищене і навіть бути видалено через деякий час, наприклад,

при передачі будь-яких паролів або важливих файлів, що містять інтелектуальну власність. У цих чатах реалізовано «Закінчене шифрування», яке гарантує неможливість перехоплення повідомлень, навіть якщо вас зламають, бо секретні чати прив'язані до самих пристроїв. Повідомлення з таких чатів не піддаються пересиланню та не залишають слідів на сервері Telegram. Також існують деякі особливості у цих чатів, а саме:

У тому випадку якщо ваш партнер або друг, з яким ви спілкувалися в секретному чаті змінить режим чату на звичайний, то повідомлення почнуть шифруватися стандартним методом шифрування і це відбудеться з усіма повідомленнями в чаті, які були не видалені.

Якщо ваш співрозмовник зробить фото екрану, на якому буде секретний чат, то повідомлення про фото прийде на ваш пристрій.

Коли ви вийдете з облікового запису, але забудете закрити секретний чат він автоматично вийде без можливості відновлення.

На даний момент Telegram може працювати на багатьох пристроях: телефонах, комп'ютерах і навіть доступний у веб-версії. Так само він не оминув і Linux так як початкові користувачі були зі сфери IT і більше користувалися Linux для своєї роботи. Зараз Telegram перекладено багатьма мовами: українською, французькою, німецькою, італійською, англійською, польською і т.д.

Якими перевагами володіє Telegram над іншими месенджерами:

- Має Portable та веб-версію;
- відкритість - використання відкритого протоколу MTProto та API, безкоштовних для всіх;
- швидкість відправлення повідомлень швидше за інших месенджерів;
- обмеження на розмір файлів, що відправляються, до 2 ГБ;
- в Telegram майже відсутня реклама;
- всі повідомлення будуть зашифровані, а повідомлення із секретних чатів видалені без слідів на сервері Telegram;

1.1.1 Роль у комунікації

Роль месенджерів у комунікаційному процесі надзвичайно важлива, особливо в контексті вивчення іноземних мов. Ці платформи стали не лише засобом обміну повідомленнями, але й важливим інструментом для побудови глобальних комунікаційних мереж та створення віртуальних спільнот.

Месенджери забезпечують доступ до текстових, аудіо- та відеовикликів, сприяючи ефективній мовній взаємодії. Це розширює можливості навчання, дозволяючи користувачам навчатися мові за допомогою різних форматів і взаємодіяти з носіями мови.

Крім того, месенджери об'єднують групи за інтересами, створюючи віртуальні спільноти для обміну знаннями. Вони також відкривають можливості для впровадження мовних ботів, які можуть надавати освітні послуги та інтерактивні завдання для вивчення мов.

Роль месенджерів у комунікації визначається їхніми можливостями у створенні віртуальних спільнот та наданні інструментів для інтерактивного та ефективного вивчення іноземних мов.

1.1.2 Особливості платформ

Особливості платформ месенджерів є важливою складовою їхньої функціональності та визначають їхню ефективність у різних контекстах вивчення іноземних мов. Почнемо з інтерфейсу та зручності використання, які є важливими для користувачів. Простий та інтуїтивно зрозумілий інтерфейс дозволяє легко навігувати та використовувати функції месенджера.

Можливості інтеграції визначають, наскільки месенджер може взаємодіяти з іншими платформами. Це може включати інтеграцію з освітніми ресурсами, що дозволяє розширити спектр доступних навчальних матеріалів та інструментів. Функціональність та відкриті API визначають рівень гнучкості та можливостей розробників для створення власних додатків та ботів для вивчення мов.

Мультимедійні можливості є ключовим елементом, оскільки вони дозволяють використовувати різні формати контенту, такі як аудіо та відео, для покращення якості навчання. Забезпечення безпеки та конфіденційності інформації користувачів є також важливою особливістю, яка визначає можливість безпечного вивчення мов в цьому середовищі.

Організація віртуальних спільнот грає ключову роль у створенні зручного середовища для обміну знаннями та досвідом вивчення іноземних мов. Мовні боти, надавання освітніх послуг та інтерактивних завдань, додають елемент індивідуалізації та забезпечують ефективний підхід до вивчення мови для кожного користувача.

Особливості платформ месенджерів визначають їхню потужність у плані вивчення іноземних мов, створюючи унікальне та інтерактивне середовище для користувачів.

1.1.3 Можливості в освіті

Можливості месенджерів в освіті виявляються в ряді ключових аспектів, що допомагають в удосконаленні навчання іноземних мов. По-перше, розглянемо їхню роль у комунікації. Месенджери створюють відмінне віртуальне середовище для спілкування, обміну думками та навчання на різних мовних рівнях.

Можливість використання текстових, аудіо- та відеоповідомлень надає користувачам різноманітні форми мовної взаємодії. Це сприяє розвитку всіх навичок: читання, письма, розуміння на слух та усного мовлення. Месенджери відкривають перед користувачами можливість отримати навички письмового спілкування через текстові повідомлення та сприяють активній участі в чат-групах.

Однією з ключових особливостей месенджерів в освіті є їхній потенціал для створення віртуальних спільнот та груп інтересів. Це дозволяє користувачам об'єднати зусилля та вивчати мову в колективі, обмінюючись досвідом та навчальними матеріалами.

Месенджери також стають ідеальним майданчиком для використання мовних ботів. Ці інтерактивні програми можуть надавати користувачам завдання, тести та вправи, персоналізовані під їхні потреби та рівень вивчення.

Можливості месенджерів у сфері освіти роблять їх ефективним інструментом для вивчення іноземних мов, допомагаючи користувачам розвивати комунікативні навички, спілкуватися та вивчати мову у віртуальному навчальному середовищі.

1.1.4 Тенденції та перспективи

Тенденції та перспективи розвитку месенджерів у сфері вивчення іноземних мов стають ключовими в аспекті розширення можливостей користувачів та покращення навчального процесу. Зараз спостерігається збільшення популярності використання месенджерів у якості освітнього інструмента. Це пов'язано зі зростанням зацікавленості у навчанні мов шляхом інтерактивних та доступних засобів комунікації.

Однією з важливих тенденцій є посилення використання штучного інтелекту та машинного навчання у месенджерах для створення більш інтелектуальних та індивідуалізованих підходів до вивчення мов. Це включає в себе розробку ботів, які аналізують навчальні досягнення користувачів та пропонують персоналізовані матеріали та завдання.

Зараз важливою є експансія мультимедійних можливостей месенджерів для покращення процесу вивчення. Використання аудіо- та відеоконтенту дозволяє створювати іммерсивне навчання та полегшує розвиток навичок вимови та слухання.

Ще однією тенденцією є розширення можливостей інтерактивності месенджерів у сфері тестування та оцінювання знань. Використання ботів для проведення тестів та отримання зворотного зв'язку дозволяє ефективно відстежувати прогрес користувачів.

Загалом, перспективи розвитку месенджерів у вивченні мов включають подальше удосконалення інтерактивності, впровадження інтелектуальних

технологій та розширення можливостей мультимедіа з метою забезпечення більш ефективного та захоплюючого навчання.

1.2 Чат-боти

Чат-боти - це програма в основі якої ІІ, яка здатна приймати від користувача різну інформацію: повідомлення, команди, файли тощо, після обробити їх і видати результат.

В основному виділяють чат-ботів двох типів:

1) Декларативні чат-боти, орієнтовані на завдання

Задіяні заздалегідь прописані інструкції та використовуються переважно для обслуговування та підтримки у форматі 24\7.

2) Предиктивні чат-боти на основі даних, що працюють у режимі діалогу

Залучають машинне навчання і є складнішою формою декларативних чат-ботів. Вони здатні враховувати контекст надісланого користувачем повідомлення та надіслати більш персоналізовану відповідь.

Чат-боти можуть застосовуватися для різних завдань: прості відповіді на запитання по командам, різні тести, відправка файлів, перевірка файлів на віруси, побудова маршрутів по карті, виклик таксі і т.п.

1.3 Приклади чат-ботів

1.3.1 Skeddy Bot

Наприклад візьмемо простого чат-бота "Skeddy"

Цей чат-бот був створений для людей, які часто щось забувають. Ось уявіть собі ситуацію, коли ви знаходитесь на роботі і тут різко згадуєте що вдома закінчилося молоко, але через наряджену роботу часто заходячи ввечері в магазин ви просто забували про це і вранці не могли випити чашку кави з молоком. Тут на допомогу приходить чат-бот «Skeddy».

З його допомогою ви можете створити собі замітку або нагадування, прив'язане до часу. Чому це так зручно? Тому що месенджери в наш час дуже популярні і є у більшості, а також зручності сприяє те, як саме задаються нагадування. Вам потрібно тільки знайти цього чат-бота і натиснути кнопку старт. Після цього можна просто писати йому звичайні фрази, наприклад, «Купити молоко о 18 годині» або «Вимкнути плиту через 35 хвилин» і це зручно, коли під рукою немає таймера, а вбудований таймер у телефоні занадто тихий.

Так само можна використовувати справжній розклад, який нагадуватиме вам зробити що-небудь щомісяця чи тиждень і т. п. у певний час потрібно лише запровадити: «Передати показання лічильників кожен перший день місяця о 10 ранку».

Приклади повідомлень показані на рисунку 1.

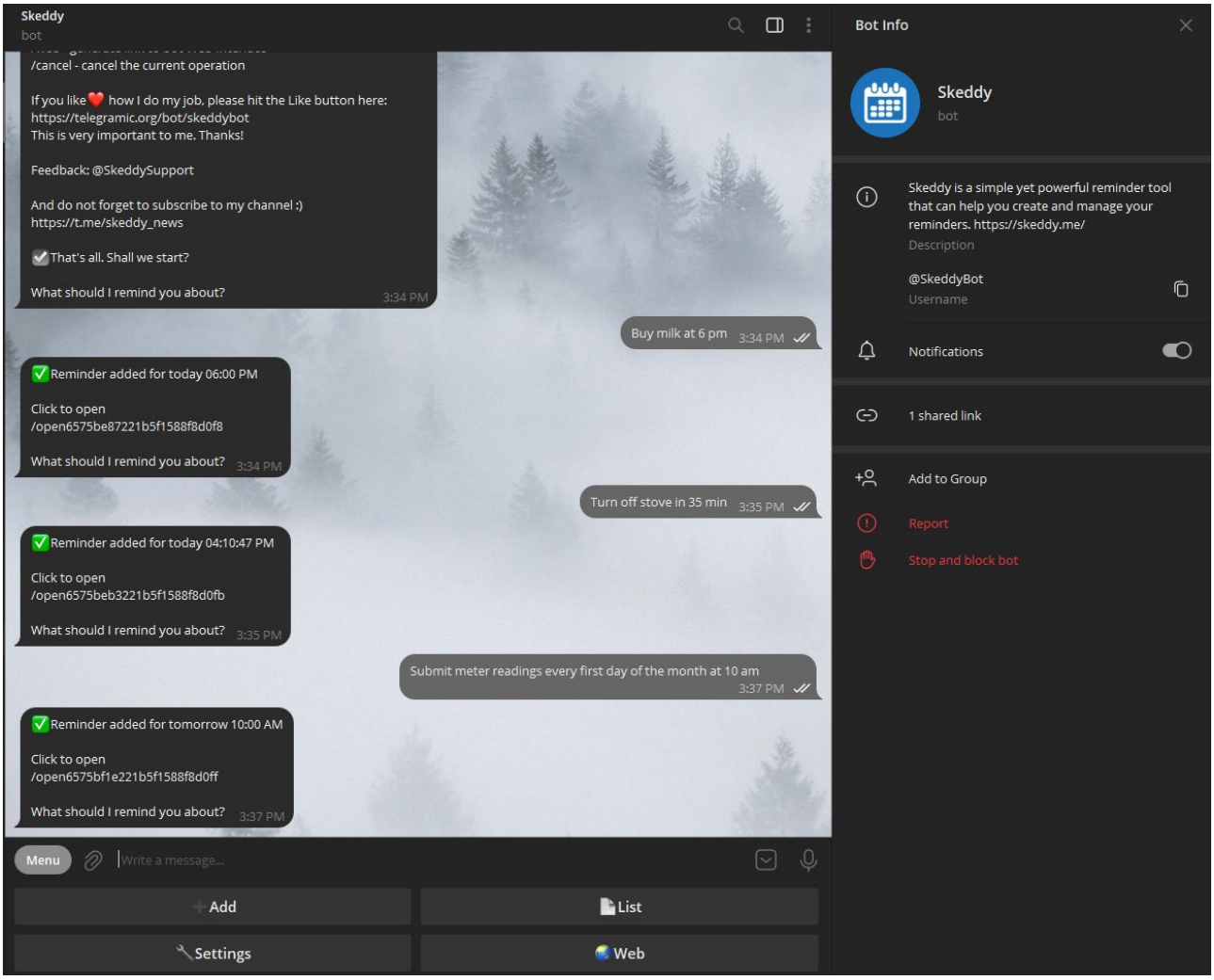

Рисунок 1.1 - приклад створення нотаток у чат-боті "Skeddy"

1.3.2 WikiBot

WikiBot — це потужний та інтелектуальний телеграм-бот, спеціально створений для максимального зручного вивчення та використання інформації. Отримайте доступ до безмежної всесвітньої бази знань, використовуючи цей бот як вашого особистого наукового помічника.

Швидкий та ефективний пошук. Забудьте про безкінечні години пошуків — WikiBot забезпечить вас швидким та точним знаходженням потрібної інформації в межах Вікіпедії.

Актуальна та достовірна інформація. Використовуйте перевірені статті та дані з різних галузей знань. Бот завжди працює з актуальними матеріалами.

Мовна універсальність. Незалежно від того, на якій мові користувач працю, WikiBot забезпечить інформацією, розуміючи багато мовний спектр.

Повідомлення та оновлення. Користувач може отримувати свіжі новини та актуалізації про нові статті та дані, щоб бути в курсі всіх змін у світі знань.

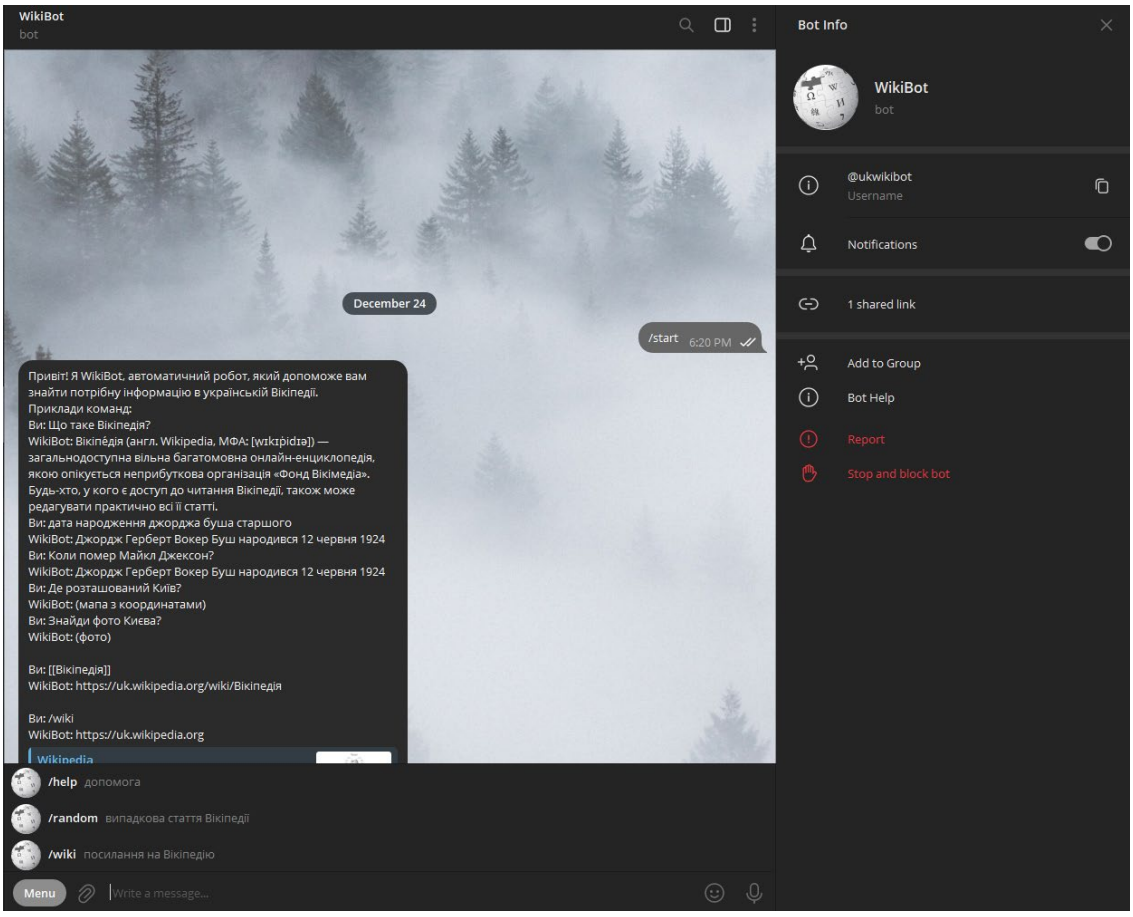

Рисунок 1.2 – WikiBot

1.3.3 Coin Market Cap Bot

CoinMarketCapBot створений, щоб стати вірним помічником у світі криптовалют та фінансових ринків. Цей телеграм-бот, розроблений CoinMarketCap, призначений для того, щоб забезпечити користувачів актуальною та точною інформацією про криптовалютні активи.

Отримайте найновіші дані щодо цін, обсягу торгів, змін вартості та ринкової капіталізації в режимі реального часу. Користувачі можуть отримати повну інформацію про будь-яку криптовалюту, включаючи історію цін, технічні характеристики та інші ключові параметри.

Плюс, створюйте власний портфель, встановлюйте межі цінових сповіщень та отримуйте найважливіші оновлення прямо у вашому Телеграмі. CoinMarketCapBot пропонує охоплення тисяч криптовалют з усього світу, а також індексів та інших ключових фінансових ринків.

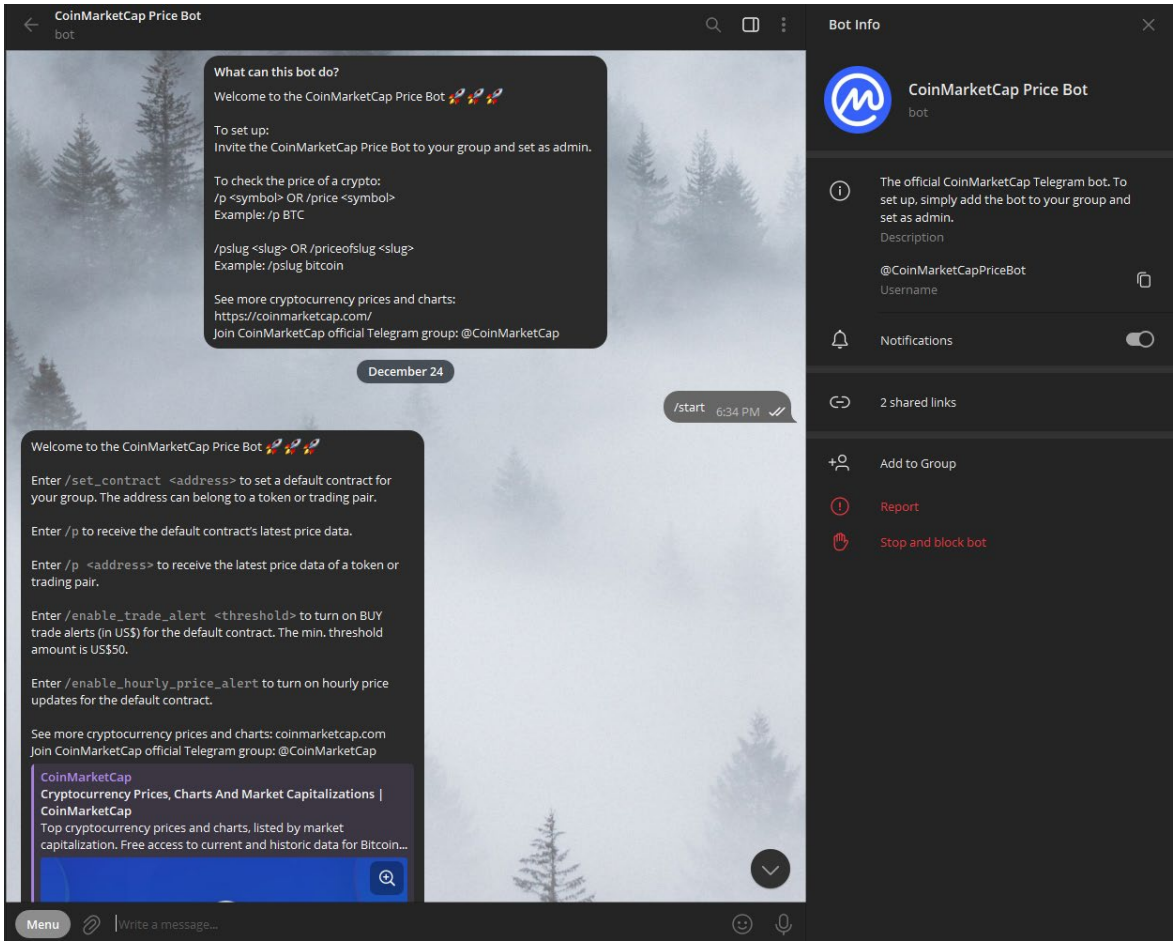

Рисунок 1.3 - Coin Market Cap Bot

1.3.4 QuizBot

Ідеальний супутник для тих, хто бажає насолоджуватися цікавими та захоплюючими вікторинами. Розроблений з урахуванням потреб користувачів, цей телеграм-бот пропонує унікальні та захоплюючі вікторини, які допомагають розширити знання в різних галузях.

Відчуйте задоволення від розв'язання цікавих головоломок та завдань, які розвивають ваш розум. З великим розмаїттям категорій вікторин ви можете обирати тему, яка вас цікавить - наука, історія, розваги та багато інших.

Викликайте друзів на вікторини, порівнюйте результати та беріть участь у захопливих змаганнях. Регулярні оновлення та нові питання завжди тримають вас на грані нових вражень та знань.

QuizBot - це ідеальний спосіб розважитися та вивчити щось нове одночасно. Починайте вікторини прямо зараз та долучайтеся до спільноти розумних гравців!

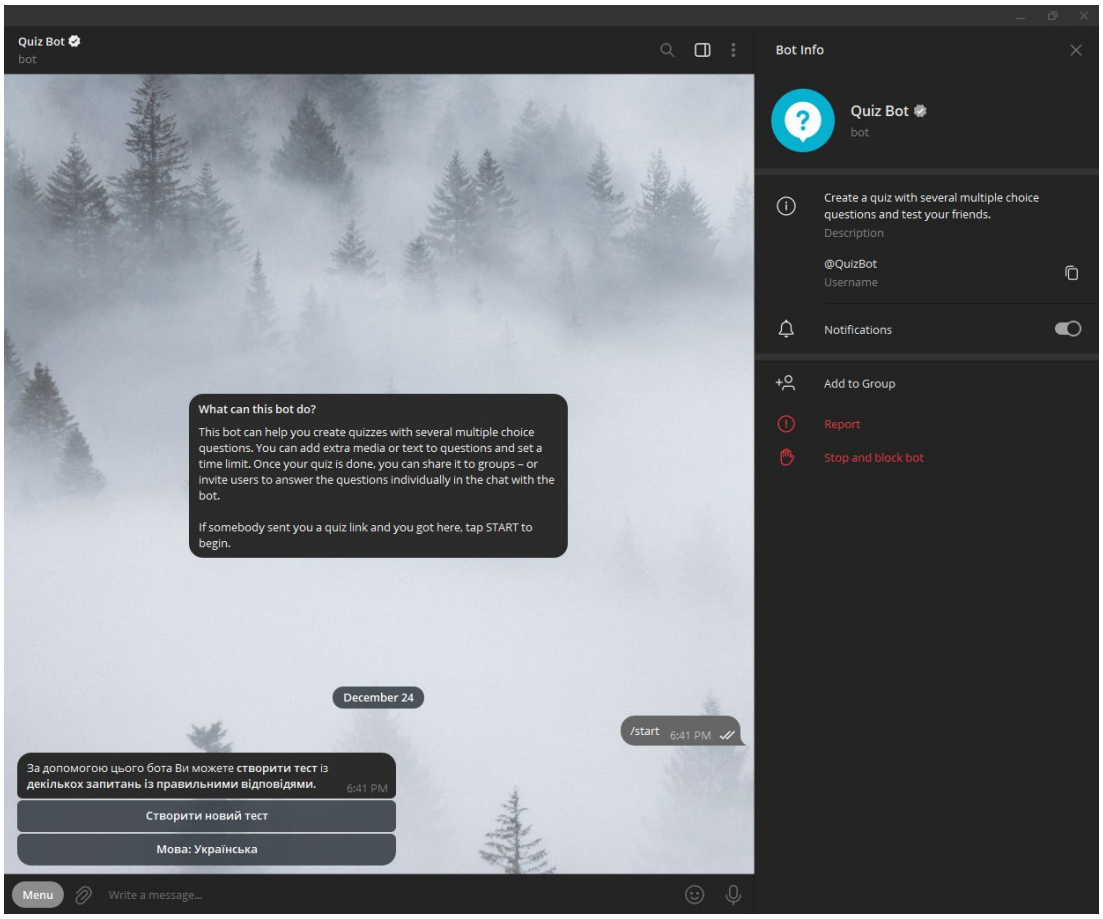

Рисунок 1.4 - QuizBot

1.3.5 Git Hub Bot

GitHubBot - важливий союзник у розробці, забезпечуючи негайні та зручні сповіщення про всі події в репозиторіях користувача на GitHub. Цей телеграм-бот розроблено з урахуванням потреб користувачів, з метою автоматизації сповіщень та полегшення відстеження всіх важливих змін у програмному забезпеченні.

Сповіщення в реальному часі. GitHubBot інформує користувача миттєво про будь-яку дію в їхніх репозиторіях, надаючи повний огляд подій - від нових комітів до пул-реквестів та виправлень багів.

Стеження за багатьма репозиторіями. Користувач може зручно відстежувати кілька проектів одночасно, налаштовуючи індивідуальні сповіщення для кожного репозиторію.

Деталізовані зміни. За допомогою GitHubBot користувач отримує легкий доступ до перегляду змін в коді, нововведень та інших актуальних подій у їхніх проектах.

GitHubBot - це невід'ємна частина розробницького процесу, яка допомагає користувачу залишатися інформованим та контролювати важливі аспекти їхнього програмного забезпечення.

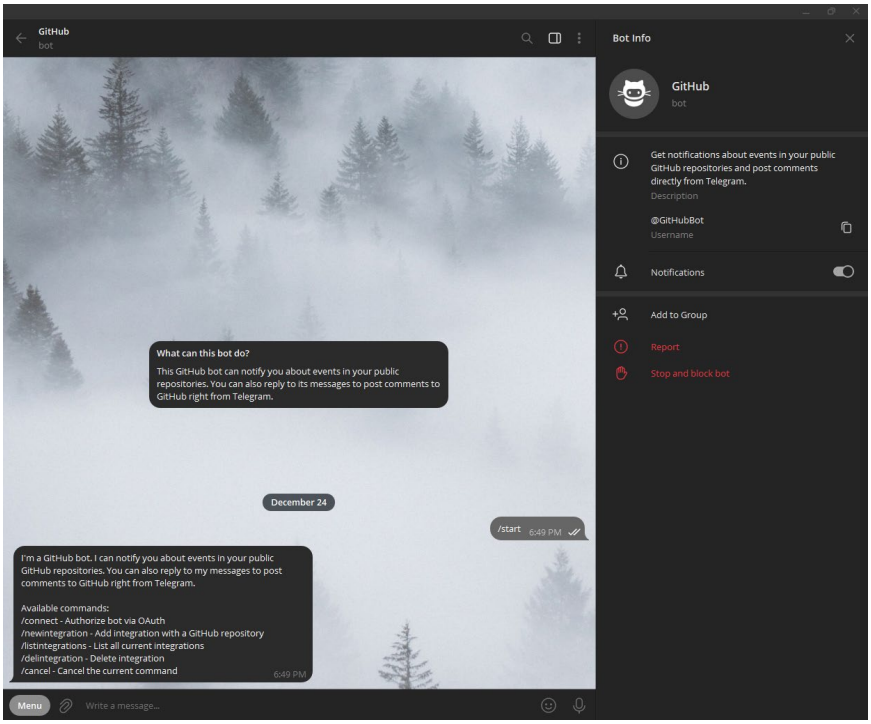

Рисунок 1.5 – GitHubBot

1.3.6 Gif Bot

Відмінний співбесідник у світі гіфок! Цей телеграм-бот призначений для швидкого та легкого пошуку та відправлення найсмішніших, захоплюючих та виразних гіфок прямо в чаті.

Швидкий пошук. Користувачі можуть отримати миттєвий доступ до безлічі гіфок на різні теми, просто введіть ключове слово, і бот знайде ідеальну гіфку.

Забавні та виразні. Незалежно від того, чи шукаєте щось смішне або виразне для вираження почуттів, @gif має величезний вибір, задовольняючи всі потреби користувачів.

Легка відправка. Обирайте кращі гіфки та надсилайте їх прямо у чаті або групі, роблячи ваші розмови більш живими та веселими.

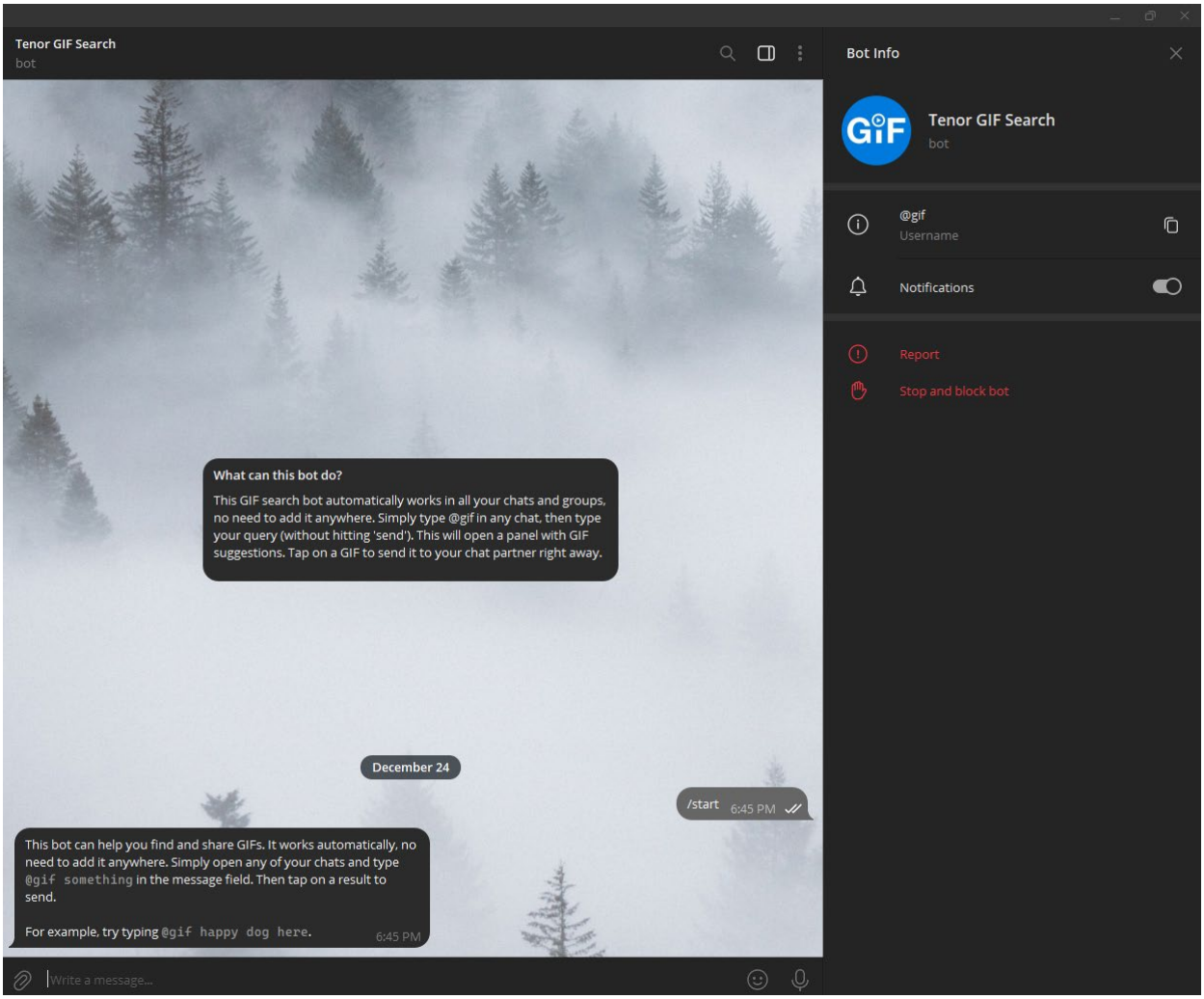

Рисунок 1.6 – Gif Bot

1.4 Telegram Bot API

Bot API дає можливість підключити робота до системи Telegram. Telegram-боти - це звичайні облікові записи, але для їх налаштування не потрібний додатковий номер телефону. Керувати цими ботами можна за допомогою HTTPS - запитів. Але при цьому існує два методи отримання даних відправлення даних.

Long Pooling та Webhook – дозволяють запросити оновлення від бота.

• Перший метод постійно через певний проміжок часу опитує сервер Telegram про наявність оновлень.

• Другий метод сервер сам оповіщає програму оновлення

Щоб керувати ботом за допомогою Telegram Bot API потрібно отримати спеціальний токен який є унікальним для кожного бота. Його можна отримати, зареєструвавши свого нового чат-бота в особливому чат-боті BotFather. Повну документацію можна знайти у списку джерел.

Приклад діаграми Telegram Bot API представлений рисунку 1.2

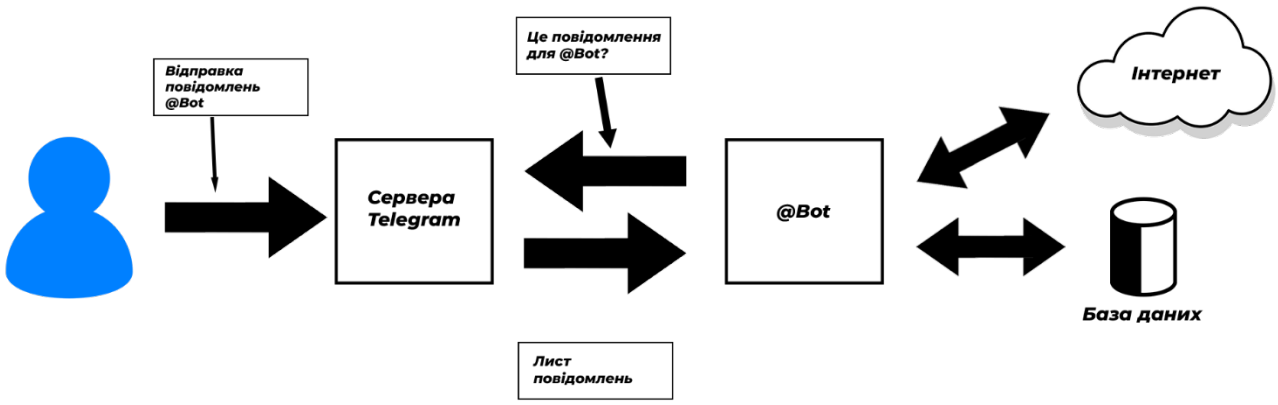

Рисунок 1.7 - Приклади діаграми Telegram Bot API

В даний час Telegram Bot API реалізований на різних платформах та мовах.

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА СЕРЕДОВИЩА РОЗРОБКИ

Під час пошуку відповідних рішень для реалізації чат-бота було зроблено висновок, що для розробки бота потрібно знати хоча б одну мову серверного програмування, наприклад Python, Ruby, Node.JS або PHP. Потрібно було визначитися яку саме мову вивчати та використовувати. Також важливо вміти працювати з REST (Representational State Transfer) API (Application Programming Interface), який надають месенджери, а саме Telegram Bot API.

2.1 Мова програмування Python

Як мову програмування було обрано мову Python.

Python - це високорівнева загальнопризначена мова програмування, яка володіє динамічною строгою типізацією та автоматичним управлінням пам'яттю. Основна мета мови - полегшити роботу розробника, зробити код більш читабельним та високоякісним, а також забезпечити переносимість написаних програм. Вона повністю орієнтована на об'єкт, де все є об'єктами, а виділення блоків коду відбувається за допомогою відступів. Мінімалістичний синтаксис ядра мови дозволяє рідко звертатися до документації. Python відомий як інтерпретована мова, і використовується, зокрема, для написання скриптів. Проте, в порівнянні з компільованими мовами, такими як C або C++, він може мати меншу швидкість та використовувати більше пам'яті.

Python є мультипарадигмовою мовою програмування, яка підтримує імперативне, процедурне, структурне, об'єктно-орієнтоване програмування, метапрограмування, функціональне програмування та асинхронне програмування. Динамічна типізація дозволяє вирішувати завдання загального програмування. Часткову підтримку аспектно-орієнтованого програмування надають декоратори, а повна підтримка реалізується за допомогою додаткових фреймворків. Контрактне та логічне програмування можна реалізувати через бібліотеки чи розширення. Основні архітектурні особливості включають динамічну типізацію, автоматичне управління пам'яттю, повну інтроспекцію,

механізм обробки винятків, підтримку багатопотокових обчислень з глобальним блокуванням інтерпретатора (GIL), а також високорівневі структури даних. Розділення програм на модулі, які можуть об'єднуватися в пакети, є підтримуваною функцією.

Основним інтерпретатором Python є CPython, який підтримує більшість платформ та є стандартом де-факто. Його вільна ліцензія дозволяє використання в будь-яких застосунках, включаючи пропрієтарні. CPython компілює вихідний код у високорівневий байт-код, який виконується в стековій віртуальній машині. Ще три реалізації мови включають Jython (для JVM), IronPython (для CLR/.NET) і PyPy. PyPy, написаний на підмножині мови Python (RPython), створений як альтернатива CPython для підвищення швидкості виконання програм за рахунок JIT-компіляції. Підтримка версії Python 2 припинилася в 2020 році, а активно розробляється версія Python 3 через пропозиції PEP (Python Enhancement Proposal), що визначають нововведення, вносять коригування згідно зі зворотнім зв'язком від спільноти та документують кінцеві рішення.

Стандартна бібліотека Python включає різноманітні переносимі функції, починаючи від роботи з текстом і закінчуючи розробкою мережевих додатків. Додаткові можливості, такі як математичне моделювання, робота з обладнанням, написання веб-додатків чи ігор, реалізуються завдяки великій кількості сторонніх бібліотек, а сам інтерпретатор Python може інтегруватися в проекти, написані на мовах, таких як C або C++. Науковці та розробники широко використовують Python в аналізі даних, машинному навчанні, DevOps, веб-розробці та інших галузях, завдяки його читабельності, простому синтаксису та можливості вивчення програмування без обов'язкової компіляції, що полегшує відладку та експериментування. Python також використовується у великих компаніях, таких як Google та Facebook.

Python має структурований і добре читаний код і має високу продуктивність програмних рішень. На Python завдяки широкому вибору вбудованих бібліотек

та корисних функцій можна написати більшість програм від найпростіших скриптів до систем управління.

Через простоту коду, сувору динамічну типізацію, автоматичне управління пам'яттю Python має низький порог входження і навіть не найдосвідченіший програміст з легкістю зможе написати потрібну йому не велику програму. майбутньому буде набагато простіше коригувати та доопрацьовувати програму

Однією з переваг Python є те, що він реалізований на різних платформах та операційних системах.

Мінусом Python є низька швидкість виконання програм, оскільки вона буде інтерпретована, проте даний мінус можна не враховувати при написанні програм для яких швидкість виконання не буде вважатися критичною.

Оскільки ми будемо використовувати мову програмування Python як фреймворк для Telegram Bot API, ми візьмемо Aiogram.

2.1.1. Framework Aiogram

AIogram — це асинхронний фреймворк для роботи з Telegram API на мові програмування Python. Він спрощує розробку чат-ботів для платформи Telegram, надаючи зручний та ефективний інтерфейс для взаємодії з Telegram API.

Основними особливостями AIogram є:

1. Асинхронність (Asynchronous)

AIogram базується на асинхронному програмуванні з використанням ключових слов async та await. Це дозволяє створювати швидкі та ефективні боти, особливо в умовах великого навантаження.

2. Простий та зручний API

AIogram надає простий та лаконічний API для взаємодії з Telegram. Завдяки цьому, розробникам легше створювати та розширювати функціональність своїх ботів.

3. Обробка подій

Фреймворк надає зручний механізм обробки подій, таких як отримання повідомлень, клавіш відповіді та інші. Розробники можуть легко визначити, які дії повинні відбутися при різних подіях.

4. Робота з клавіатурою

AIogram дозволяє створювати різноманітні клавіатури для взаємодії з користувачами, включаючи текстові, візуальні та інші типи клавіш.

5. Взаємодія з мультимедіа

Фреймворк підтримує обробку мультимедійних файлів, таких як фотографії, відео, аудіо та документи, що робить його ідеальним для створення ботів, які обробляють різні типи контенту.

6. Робота з базою даних

AIogram може легко інтегруватися з різними системами управління базами даних (наприклад, SQLite, PostgreSQL) для зберігання та отримання даних.

7. Міжнародна мовна підтримка

Фреймворк дозволяє створювати боти, які підтримують різні мови та локалізацію для більш ефективної комунікації з користувачами з різних країн.

8. Розширюваність та модульність

AIogram сприяє розширенню та модульності. Розробники можуть використовувати різні плагіни та розширення для розширення функціональності свого бота.

9. Документація та спільнота

AIogram має докладну документацію, що спрощує вивчення та використання фреймворку. Крім того, існує активна спільнота розробників, яка може надавати підтримку та поради.

AIogram є одним з популярних інструментів для створення Telegram-ботів в екосистемі Python і використовується для широкого спектру завдань, включаючи взаємодію з користувачами, обробку замовлень, автоматизацію та багато іншого.

Цей фреймворк дозволяє просто використовувати Bot API, не витрачаючи величезні ресурси часу на вивченні що дозволяє сконцентрувати свою увагу на розробці чат-бота.

Щоб почати використовувати фреймворк aiogram, потрібно скачати і встановити його командою меденжера пакетів PIP:

“$ pip install -U aiogram”

Після цього в самій програмі робота підключити такі модулі:

“from aiogram import Bot, Dispatcher, executor, types”

Потім потрібно ініціалізувати отримані від “BotFather” токен, “Bot” і “Dispatcher”

І після всіх цих дій можна приступати до програмування власного чат-бота.

2.2 База даних SQLite

SQLite - це вбудована, легка та самостійна система управління базами даних (СУБД), яка не потребує окремого серверу чи конфігурації.

Основні особливості та характеристики бази даних SQLite:

1. Тип системи.

SQLite є СУБД типу файлової системи, що означає, що вся база даних зберігається в одному файлі на диску. Це полегшує розгортання та обслуговування, оскільки не потрібен окремий сервер для роботи з базою даних.

2. Легкість Використання

SQLite використовується в основному для вбудованих додатків та мобільних додатків, оскільки вона легка, проста у використанні та не вимагає значних ресурсів.

3. Типи даних

Підтримуються стандартні типи даних, такі як INTEGER, TEXT, REAL, BLOB. Через динамічну типізацію, колонки в базі даних можуть зберігати дані будь-якого типу.

4. Транзакції

SQLite підтримує транзакційний підхід до обробки даних, де можна об'єднувати кілька операцій в одну транзакцію для забезпечення цілісності даних.

5. Крос-платформенність

База даних SQLite є крос-платформенною і може працювати на різних операційних системах, включаючи Windows, macOS та різні дистрибутиви Linux.

6. Низький обсяг пам'яті

SQLite оптимізована для роботи з обмеженими ресурсами, що дозволяє їй працювати на пристроях з обмеженим обсягом пам'яті.

7. Вбудовані функції

SQLite має вбудовані функції для роботи з рядками, датами, числами та іншими типами даних, що робить її потужною для різних сценаріїв.

8. Велика спільнота та підтримка

SQLite має велику та активну спільноту розробників, що забезпечує підтримку, оновлення та розвиток системи.

9. Безпека

Забезпечує можливість використання паролів для захисту бази даних та додаткові можливості шифрування.

10. Інструментарій для адміністрування

Є різноманітні інструменти для адміністрування та роботи з базою даних SQLite, включаючи командний рядок та графічні інтерфейси.

Саме те що в SQLite існує великий набір різних інструментів для реалізацій на відміну від мережевих систем, а також є пряме звернення до файлів, від чого кількість таблиць і самих баз обмежується тільки жорстким диском користувача і до всього вище сказаного база даних відразу вбудована в вигляді модуля у мові програмування Python спонукало мене використовувати SQLite у своїй реалізації чат-бота.

Як пізніше з'ясувалося через свою компактність та простоту SQLite ідеально підходить для створення невеликих проектів для демонстрації можливостей.

Існує лише один мінус: максимальний розмір однієї бази даних становить 2ГБ, проте це заважає дуже рідко.

Сьогодні більшість розробників використовують SQLite

Типи даних, що підтримуються:

- NULL: значення NULL.

- INTEGER: ціле число зі знаком, що зберігається в 1, 2, 3, 4, 6 або 8 байтах.

- REAL: число з плаваючою комою, що зберігається у 8-байтовому форматі IEEE.

- TEXT: текстовий рядок, закодований UTF-8, UTF-16BE або UTF-16LE.

- BLOB: тип даних, що зберігаються у тій самій формі, в якій вони були отримані.

Переваги:

- Файлова: вся база даних зберігається в одному файлі, що полегшує рух.

- Стандартизований: SQLite використовує SQL.

Недоліки:

- Відсутність управління користувача: інші більш великі бази даних дозволяють користувачам управляти зв'язками у великій кількості таблиць відповідно до привілеїв. SQLite такої функції немає.

- Неможливість додаткового налаштування: SQLite не може підвищити продуктивність.

SQLite є популярним вибором для додатків, де потрібна легка та надійна база даних без складних налаштувань.

2.3 Середовище розробки PyCharm

PyCharm - це інтегроване середовище розробки (IDE) для мови програмування Python, розроблене компанією JetBrains. Це потужний інструмент, який надає розширені можливості для розробки, відладки та тестування Python-програм. Ось детальний огляд середовища розробки PyCharm:

1. Редактор коду

- PyCharm надає потужний редактор коду зі значними можливостями виділення синтаксису, автоматичним завершенням коду, підсвічуванням синтаксичних помилок і т.д.
- Підтримка роботи з різними видами файлів, такими як Python-скрипти, файли Markdown, XML, JSON тощо.

2. Відлагодження

- Інтегрована система відлагодження дозволяє вам ефективно відлагоджувати свій код.
- Підтримка відлагодження в реальному часі та можливість встановлення точок зупинки.

3. Управління залежностями та віртуальні середовища

- PyCharm інтегрується з інструментами управління залежностями, такими як pip та conda.
- Підтримка віртуальних середовищ, що дозволяє ізолювати проекти та їх залежності.

4. Автоматичне завершення коду та підказки

- PyCharm пропонує автоматичне завершення коду для швидкої роботи та уникнення помилок.
- Розумне автозавершення та підказки на основі контексту та типів даних.

5. Аналіз коду

- Інструменти аналізу коду для виявлення помилок, покращення стилю коду та виявлення потенційних проблем.
- Підтримка віджетів величезної кількості стандартів коду та інструментів аналізу.

6. Підтримка версійного контролю

- Інтеграція з різними системами контролю версій, такими як Git, Mercurial, Subversion тощо.
- Зручний інтерфейс для роботи з гілками, злиттям змін, переглядом історії тощо.

7. Підтримка роботи з базами даних
- Інтеграція з різними базами даних, включаючи SQLite, MySQL, PostgreSQL тощо.
- Можливість виконання SQL-запитів та взаємодії з базами даних безпосередньо з IDE.

8. Можливості тестування
- Інструменти для написання, виконання та аналізу тестів, включаючи підтримку фреймворків, таких як pytest та unittest.

9. Інтеграція із фреймворками
- Підтримка роботи з популярними веб-фреймворками, такими як Django, Flask, Pyramid тощо.

10. Підтримка іншомовних окремостей
- Можливість роботи з мовами програмування, такими як JavaScript, HTML, CSS, що використовуються в проектах.

11. Розширюваність та плагіни
- PyCharm дозволяє розширювати свої можливості за допомогою плагінів та налаштувань.

12. Оптимізація робочого процесу
- Різні функції та скорочення клавіш для прискорення робочого процесу.

Це середовище розробки було обрано завдяки ранньому досвіду роботи з PyCharm, що має допомогти у більш швидкій та якісній реалізації поставлених завдань. Так само кампанія JetBrains судячи зі звітів за рік набирає популярності, розширюючи свою аудиторію та штат співробітників.

2.4 Висновок

Вибір мови програмування, фреймворку, системи управління базами даних і інтегрованого середовища розробки є ключовим етапом в розробці будь-якого програмного продукту. В моєму випадку, вибір здійснено на користь Python як мови програмування, фреймворку aiogram для роботи з Telegram API, системи управління базами даних SQLite та інтегрованого середовища розробки

PyCharm. Кожен з цих інструментів відіграває важливу роль і взаємодіє з іншими для створення ефективного та надійного програмного забезпечення.

Python обраний через його читабельність, простий синтаксис та широкий спектр бібліотек. Він володіє великою спільнотою розробників, активною підтримкою та великою кількістю різноманітних інструментів, що полегшують розробку.

Aiogram вибраний для реалізації бота у Telegram через його потужність та гнучкість. Він надає зручний API для взаємодії з Telegram, підтримує асинхронні запити, має багатофункціональний модуль для роботи з різними типами повідомлень.

SQLite обрана через її легкість використання та вбудовані можливості. Для невеликих проектів, де потрібно зберігати обмежені об'єми даних, вона є ідеальним вибором. Вона не потребує окремого сервера та має високий рівень надійності.

PyCharm обрано завдяки його розширеним можливостям відлагодження, відзначеної системою підказок та розширеною підтримкою роботи з Python. Воно надає ефективний інтерфейс для роботи з проектами будь-якої складності, забезпечуючи комфортне середовище для розробки.

Загальний висновок полягає в тому, що обрані інструменти спільно створюють зручне та продуктивне середовище для розробки програмного забезпечення. Python забезпечує читабельний та ефективний код, aiogram дозволяє легко взаємодіяти з Telegram API, SQLite дозволяє зручно зберігати дані, а PyCharm полегшує робочий процес та надає засоби для швидкої розробки та відлагодження.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ, ЗАПУСК І ТЕСТУВАННЯ ЧАТ-БОТА

3.1 Реєстрація чат-бота для Telegram

Для початку розробки необхідно пройти реєстрацію у спеціального чат-бота “BotFather”.

Потрібно використовувати команду /newbot щоб зареєструвати нового чат-бота. Потім BotFather запросить написати вас ім'я та ім'я користувача, а потім згенерує унікальний токен авторизації для нашого нового чат-бота.

Унікальний токен авторизації має вигляд “6702976447:AAEUD6IZg36U5RcLl_rixJPpXoeML_nLdCE” і має бути захищений вами. Якщо ваш токен був скомпрометований або ви втратили його існує можливість створити новий токен ввівши команду “/token”.

Також у чат-бота присутні постреєстраційні налаштування описані в таблиці 3.1.

Таблиця 3.1 - Доступні команди для зміни чат-ботів

Команда	Опис
/setname	Дозволяє змінити ім'я вашого чат-бота
/setdescription	Встановлює опис чат-бота, який бачить користувач при першому відкритті вашого бота
/setabouttext	Додає текст у поле "Про чат-бот"
/setuserpic	Додає аватар вашому чат-боту
/setcommands	Дозволяє додати список доступних команд
/deletebot	Видаляє вказаного чат-бота

Після виконання налаштувань за допомогою чат-бота “BotFather” можна розпочати розробку програмної частини чат-бота.

3.2. Опис функціональності чат-бота

Насамперед чат-бот повинен буде допомогти людям, які тільки почали вивчати англійську мову і їм критично не вистачає слів для комфортного продовження вивчення.

Для зручності використання та автоматичного розширення було прийнято рішення використовувати інтерфейси з вбудованою клавіатурою. Планується реалізувати наступні меню:

- Головне меню
- Меню налаштувань
- Меню вибору блоку для вивчення
- Меню вибору правильного слова
- Меню підтвердження
- Меню завершення блоку

Чат-бот при першому запуску користувачем пропонує ввести команду “/start” після чого користувачеві буде показано меню, в якому він повинен вибрати тему для вивчення і підтвердити початок вивчення.

У разі успішного завершення тесту без жодної помилки користувач отримає повідомлення про успішне завершення теми або повернеться в меню вибору теми.

3.3 Опис прецедентів

На рисунку 3.1 представлена діаграма прецедентів

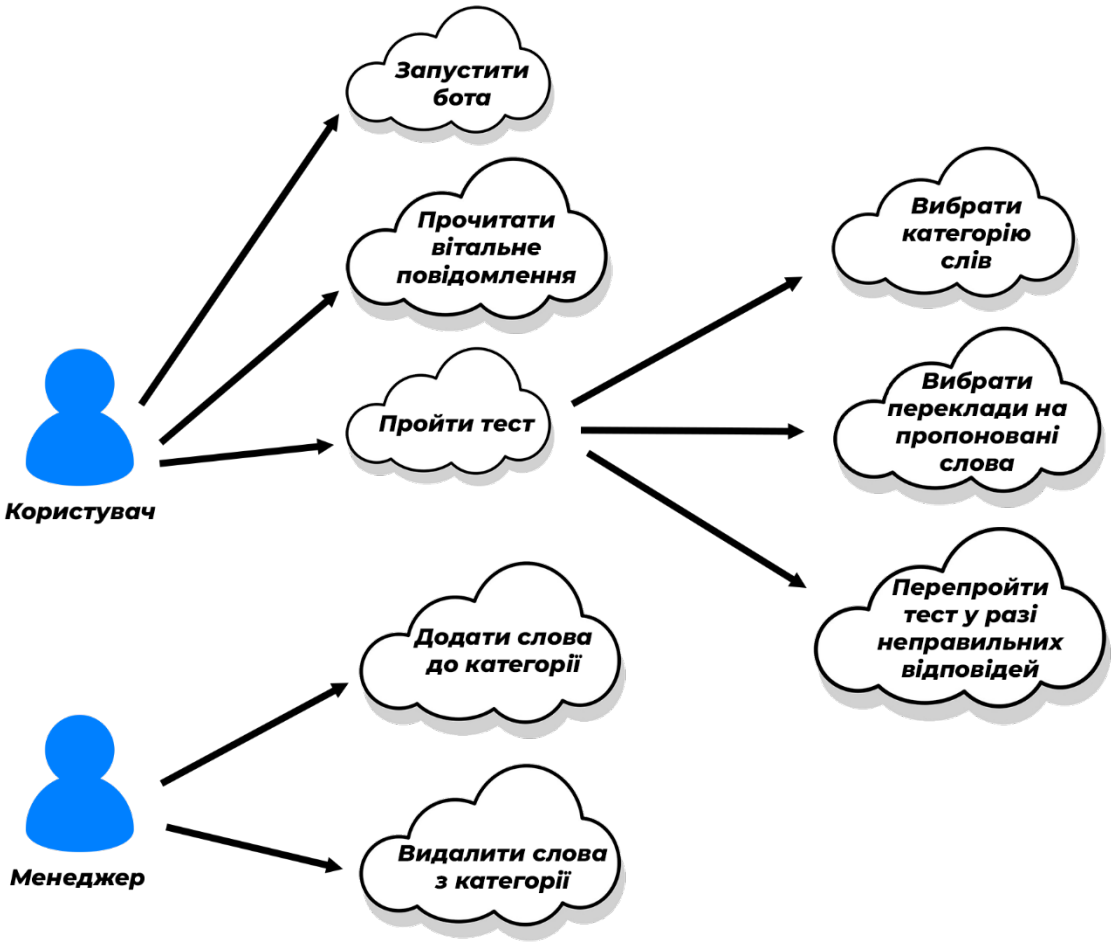

Рисунок 3.1 - Діаграма прецедентів

Текстовий опис прецедентів:

Прецедент “Запустити бота” активується командою “/start” і дозволяє користувачам ознайомитися з представленим чат-ботом та вибрати подальші дії.

Прецедент “Пройти тест” включає кілька прецедентів і дозволяє користувачеві успішно завершити тест або повернутися назад до вибору теми для проходження.

Прецедент “Вибрати категорію слів” під час запуску надсилає запит до бази даних і виводить користувачу на вибір список доступних тем для вивчення на даний момент.

Прецедент “Вибрати переклади на пропоновані слова” запускається після вибору теми та пропонує користувачеві англійське слово та кілька варіантів перекладу. Якщо користувач вибирає всі переклади вірно, бот надсилає

повідомлення про успішне завершення теми та записує інформацію до бази даних.

Прецедент “ Перепройти тест у разі неправильних відповідей” якщо користувач припускається помилки при проходженні тесту його повертає в меню вибору теми для проходження.

3.4 Опис серверної частини

3.4.1 База даних

Для реалізації поставленого завдання знадобилася база даних (БД). У проекті використовується БД SQLite. База даних має 6 таблиць.

Діаграма представлена на рисунку 3.2

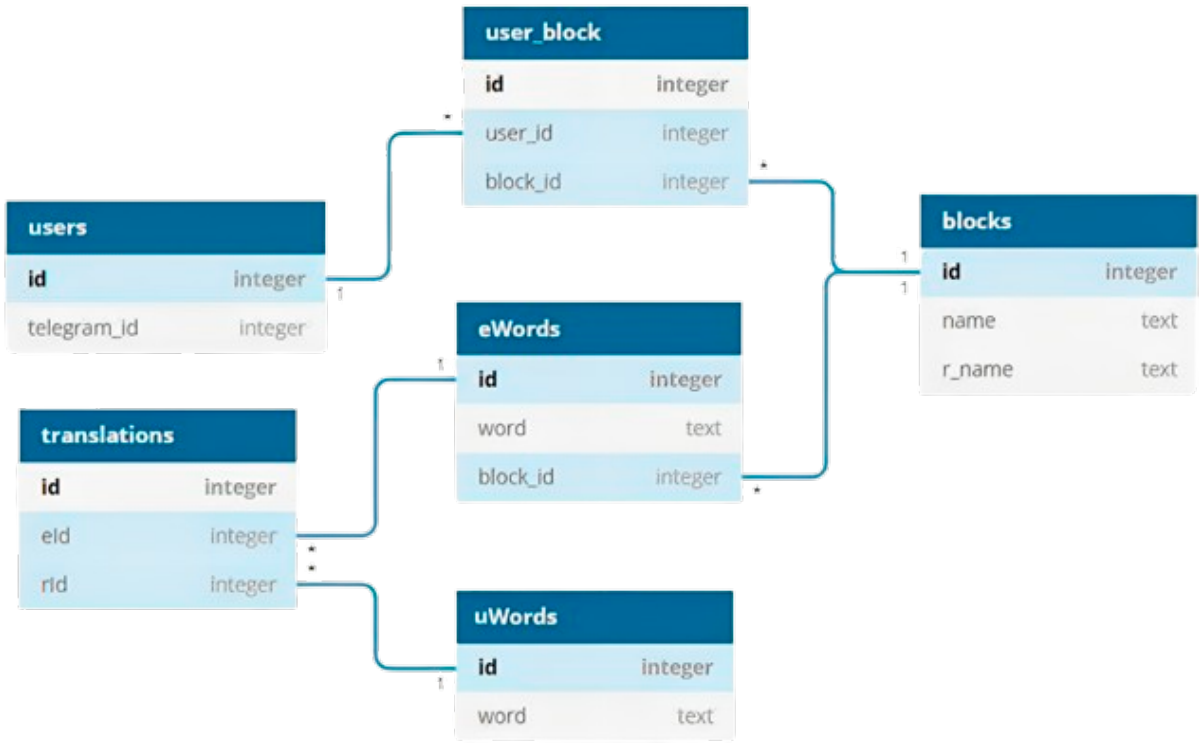

Рисунок 3.2 – Діаграма бази даних

Кожна таблиця містить у собі таку інформацію:

Blocks - Містить у собі номери та назви блоків для вивчення

EWords - Містить у собі англійські слова та номер прив'язаного блоку

UWords - Містить у собі українські слова

Translations - Містить зв'язок між унікальними ідентифікаторами Англійських та Українських слів

User_Block - Містить у собі інформацію про пройдені блоки користувачем

Users - Містить у собі інформацію про користувачів, що розпочали роботу з чат-ботом

Для створення та заповнення бази даних було використане програмне забезпечення “DB Brower for SQLite”. Таким чином, для заповнення або виправлення таблиць месенджеру не потрібно буде вивчати SQL запити і це підвищить доступність бота.

3.4.2. Опис серверної частини чат-бота

Від серверної частини чат-бота потрібно вирішити такі завдання:

- Отримувати та обробляти повідомлення від користувача
- Отримувати інформацію з бази даних виконуючи запити з необхідними параметрами
- Складати та надсилати відповідь у вигляді тексту користувачеві

Кнопки inline-клавіатури та їх зворотна інформація дозволяють чат-боту отримати команду для подальших дій. Після того як користувач натисне кнопку на inline-клавіатурі, серверна частина порівняє отриману відповідь з варіантами, закладеними в програмі і видасть результат.

У документації Telegram Bot API сказано, що кожна клавіатура повинна мати один обов'язковий параметр - ім'я кнопки (text), і шість необов'язкових - посилання (url), зворотні дані (callback_data), можливість вбудованого запиту (switch_inline_query), можливість виведення клавіатури з іншого чату (switch_inline_query_current_chat), виклик опису запущеної гри (callback_game) та кнопка з можливістю купівлі (pay).

Приклад inline - клавіатури представлений рисунку 3.3

Рисунок 3.3 – Приклад inline – клавіатури

Для початку реалізуємо функцію прийому початкового повідомлення та надсилання відповіді:

```
async def start_message
```

Після цього створимо функцію обробки натискання кнопки «Вибір блоку» яка створить нам нове меню з усіма блоками, запросивши інформацію про існуючі та доступні на даний момент блоки:

```
async def process_callback_choose_block
```

Потім реалізуємо обробку кнопок блоків у якій звертатимемося до бази даних для отримання інформації про кількість слів у блоці:

```
async def process_callback_blocks
```

Після почнемо роботу над функцією обробної кнопки варіанта відповіді, яка так само буде запитувати один правильний і 3 неправильні варіанти відповіді з бази даних:

```
async def process_callback_check
```

Наприкінці потрібно додати обробку кнопки «Що пройдено» в якій буде звернення до бази даних за інформацією про пройдені блоки користувачем.

```
async def process_callback_done
```

Таким чином було реалізовано серверну частину чат-бота з необхідними функціями.

3.5 Запуск та тестування

При введенні команди "/start" користувачеві пропонується почати вибір блоку або перейти в налаштування (рисунок 3.4).

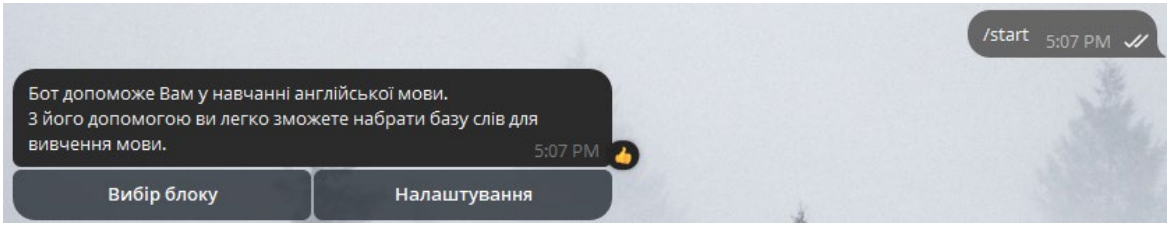

Рисунок 3.4 – Головне меню

Далі було реалізовано меню вибору блоку вивчення. Воно дозволяє вибрати цікаву для нас тему для вивчення слів. Серед кнопок клавіатури можна знайти всі доступні теми (рисунок 3.5).

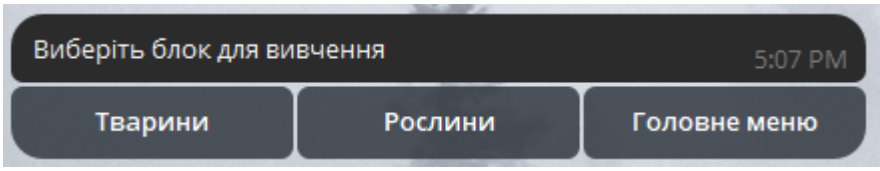

Рисунок 3.5 - Меню вибору блоку для вивчення

Після вибору блоку для вивчення піде його опис та підтвердження про початок проходження (рисунок 3.6).

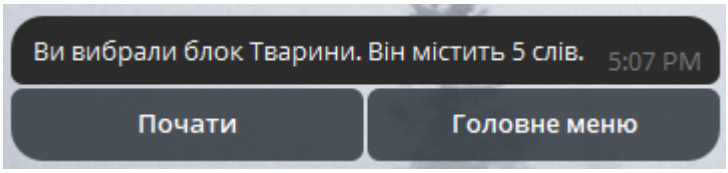

Рисунок 3.6 - Меню підтвердження

У разі натискання кнопки “Почати” буде створено нове меню, яке пропонує вам слово з блоку і 2 варіанти перекладу з яких вірний буде лише один (рисунок 3.7).

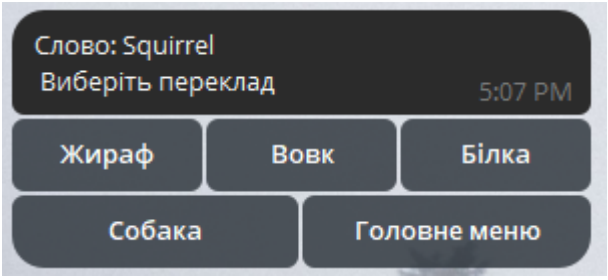

Рисунок 3.7 - Меню вибору правильного слова

Після проходження блоку з правильним 100% результатом буде отримано повідомлення про успішне завершення блоку з можливістю повернення до головного меню або вибору наступного блоку (рисунок 3.8).

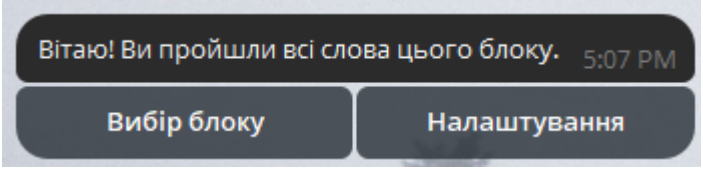

Рисунок 3.8 - Меню завершення блоку

При натисканні "Налаштування" буде отримано клавіатуру налаштувань (рисунок 3.9).

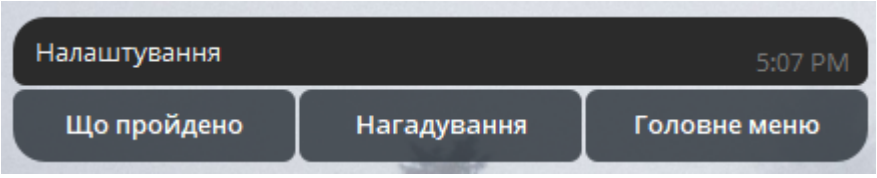

Рисунок 3.9 – Меню налаштувань

Після вибору варіанта "Що пройдено" можна отримати повідомлення про пройдені блоки та з можливістю повернутися до головного меню (рисунок 3.10).

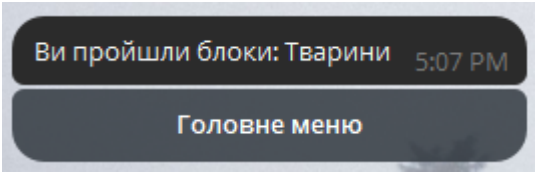

Рисунок 3.10 - Повідомлення про пройдені блоки

На цьому реалізація клієнтської частини чат-бота було завершено.

ВИСНОВКИ

Висновок до даної дипломної роботи є результатом комплексного дослідження та впровадження інноваційних підходів у вивченні англійської мови за допомогою телеграм бота, розробленого з використанням мови програмування Python. У ході дослідження було здійснено аналіз сучасних методик вивчення іноземних мов, визначено основні виклики та можливості у цій галузі.

Розглянуто і обгрунтовано вибір технологій та інструментів для реалізації проекту, зосереджуючись на використанні мови програмування Python та інтерактивних методик вивчення. Було обґрунтувано вибір конкретних засобів машинного навчання та відкритих джерел для підтримки ефективного навчання.

Практична реалізація телеграм бота підтвердила його успішність у підтримці індивідуального вивчення англійської мови. Користувачі можуть взаємодіяти з ботом, виконуючи різноманітні завдання та отримуючи індивідуалізовані рекомендації для поліпшення своїх мовних навичок. Такий інтерактивний підхід сприяє активному та ефективному вивченню мови.

Важливим внеском роботи є детальний аналіз та обґрунтування ефективності використання технологій у сфері освіти, що визначається не лише реалізацією бота, але й виявленням можливостей подальшого вдосконалення та розширення його функціональності. У цьому контексті висновки містять конкретні рекомендації для подальшого розвитку проекту, зокрема щодо вдосконалення інтерфейсу користувача, розширення завдань та використання додаткових педагогічних методик.

У майбутньому планується збільшення функціональності чат-бота, а саме:

• Нагадування про необхідність щоденного/тижневого проходження різних тем

• Можливість додавання слів користувача та тем

• Можливість вивчення граматики

Кінцевим результатом випускної кваліфікаційної роботи є реалізований чат-бот у месенджері Telegram, який допоможе у вивченні англійської мови шляхом нарощування вашої власної бази слів.

ПЕРЛІК ПОСИЛАНЬ

1. Що таке Телеграм: повний огляд месенджера і його переваги / URL: https://107.com.ua/blog/sho-take-telegram-povnii-ogliad-mesendjera-%D1%96-iogo-perevagi/ (дата звернення: 26.09.2023).

2. Оцінка ринку месенджерів / URL: https://ain.ua/2023/03/09/telegram-osnovnyj-mesendzher-opytuvannya/ (дата звернення: 14.09.2023).

3. Рейтинг найбезпечніших месенджерів. / URL: https://ain.ua/2023/05/11/rejtyng-najbezpechnishyh-mesendzheriv-telegram-na-5-misczi/ (дата звернення: 17.10.2023).

4. SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems / URL: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems (дата звернення: 04.09.2023).

5. Aiogram / URL: https://docs.aiogram.dev/en/latest/ (дата звернення: 26.08.2023).

6. PyCharm Community Edition / URL: https://blog.jetbrains.com/pycharm/2017/09/pycharm-community-edition-and-professional-edition-explained-licenses-and-more/ (дата звернення: 03.10.2023).

7. JetBrains Toolbox / URL: https://blog.jetbrains.com/blog/2016/03/09/jetbrains-toolbox-release-and-versioning-changes/ (дата звернення: 25.09.2023).

8. Bots: An introduction for developers / URL: https://core.telegram.org/bots#6-botfather (дата звернення: 22.09.2023).

9. Telegram APIs / URL: https://core.telegram.org/ (дата звернення: 25.08.2023).

10. Програмуваня по-українськи / URL: programming.in.ua (дата звернення: 21.09.2023).

11. MySQL Documentation / URL: https://dev.mysql.com/doc/ (дата звернення: 22.10.2023).

12. Дейт К. Дж. Введення в системи баз даних (8-е видання). – М.: Вільямс, 2005. – 1328с.

13. Scheduled Tasks Documentation / URL: https://www.aquaclusters.com/app/home/project/public/aquadatastudio/wikibook/Documentation21.0/page/Scheduled-Tasks/Scheduled-Tasks / (дата звернення: 02.11.2023).

14. Abhishek Singh. Master Python Using Version 3.11: Learn Python Like Never Before. 2023. – 385 c.

15. John Whitington. Python from the Very Beginning. 2023. – 238 c.

16. Mohit, Bhaskar N. Das. Learn Python in 7 Days. 2017. – 280 c.

17. Ana Bell. Learn Python. – Manning Publications Co. 2018. – 456 c.

ВІДГУК КЕРІВНИКА

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ «ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет інформаційних технологій
Кафедра програмного забезпечення комп'ютерних систем

ВІДГУК

Наукового керівника Мещерякова Леоніда Івановича, д.т.н., проф. каф. ПЗКС
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання, посада, місце роботи)

на магістерську роботу

студента Ярощук Олексія Романовича
(прізвище, ім'я, по батькові)

курсу II групи 121м-22-2
спеціальності 121 Інженерія програмного забезпечення
освітньої програми «121 Інженерія програмного забезпечення»
на тему Розробка та обґрунтування телеграм бота – вивчення англійської мови із застосуванням Python

Актуальність теми Ця дипломна робота належить до категорії високоактуальних досліджень в області вивчення іноземних мов та застосування сучасних технологій у процесі освіти. Автор обрав не лише популярну тему, але і вдало розробив та обґрунтував телеграм бота для вивчення англійської мови, використовуючи мову програмування Python..

Мета досліджень , визначена автором, була чітко викладена та конкретизована. Важливим етапом роботи було вивчення існуючих методик вивчення, аналіз та вибір необхідних технологій, а також ретельне обґрунтування концепції та функціональності розробленого бота.

Коротка характеристика розділів роботи У першому розділі проведено глибокий аналіз сучасних методик вивчення іноземних мов, висвітлено основні поняття та підходи у сфері чат-ботів. Особлива увага приділена постановці завдань для подальшого дослідження та розробки чат-бота для вивчення англійської мови. У другому розділі розглядається процес вибору необхідних технологій та середовища для розробки чат-бота. Автор детально аналізує

доступні технології, вибирає мову програмування Python та описує вибір інших ключових компонентів для ефективної реалізації поставлених завдань. Останній розділ включає в себе детальний опис процесу розробки чат-бота для вивчення англійської мови. Автор описує етапи від реалізації до запуску та тестування бота. Зосереджено увагу на практичному впровадженні та результативності розробленого рішення.

Практичне значення роботи роботи проявляється в реалізації та використанні розробленого телеграм бота для інтерактивного вивчення англійської мови. Відзначаю позитивний вплив бота на підтримку індивідуального процесу вивчення та надання користувачам інтерактивних завдань та вправ.

Зауваження та недоліки Розділ, присвячений аналізу та обґрунтуванню ефективності використання технологій у процесі освіти, може бути більш детальним. У розділі висновків слід додати конкретні рекомендації для подальшого вдосконалення бота або розширення його можливостей.

Висновки та оцінка Ця дипломна робота не лише відповідає усім встановленим критеріям, але й пропонує конкретні практичні рішення для покращення процесу вивчення іноземних мов. Під час виконання магістерської кваліфікаційної роботи студент Ярощук О.Р. постав кваліфікованим та впевненим спеціалістом, який знаходить оптимальні рішення у складних технічних питаннях. Вважаю, що магістерська кваліфікаційна робота заслуговує оцінку «добре», а Ярощук О. Р. – присвоєння кваліфікації «магістра» по спеціальності інженерії програмного забезпечення.

Науковий керівник Мещеряков Л.І., док. техн. наук, проф., проф. каф. ПЗКС

(прізвище, ім’я, по батькові, посада, місце роботи)

«___»_____20__ р. _____

(підпис)

РЕЦЕНЗІЯ
на магістерську роботу

студента Ярощука Олексія Романовича
(прізвище, ім'я, по батькові)

курсу II групи 121м-22-2
кафедри програмного забезпечення комп'ютерних систем
спеціальності 121 Інженерія програмного забезпечення
освітньої програми
Тема роботи Розробка та обґрунтування телеграм бота – вивчення англійської мови із застосуванням Python

Стисла характеристика розділів роботи У першому розділі докладно проаналізовано сучасні стратегії вивчення іноземних мов та висвітлені ключові підходи у сфері чат-ботів. Особлива увага була зосереджена на створенні чат-бота для вивчення англійської мови, що передбачає постановку конкретних завдань для подальших досліджень. Другий розділ детально розглядає вибір технологій та середовища для розробки чат-бота. Автор проводить аналіз доступних технологій, обирає мову програмування Python та розкриває важливі компоненти для ефективної реалізації завдань. У заключному розділі ретельно описано процес розробки чат-бота для вивчення англійської мови, з викладенням етапів від реалізації до запуску та тестування бота. Автор акцентує увагу на практичному впровадженні та результативності розробленого рішення.

Пропозиції, внесені студентом, рівень їх наукового обґрунтування У даній кваліфікаційній роботі студент висунув кілька пропозицій для вирішення поставлених завдань. Кожна із цих пропозицій була належно обґрунтована та підкріплена науковими даними.

Практичне значення роботи проявляється в конкретній реалізації та ефективному використанні створеного телеграм бота для інтерактивного освоєння англійської мови. Зауважується позитивний вплив бота на підтримку індивідуального навчального процесу, забезпечуючи користувачам доступ до інтерактивних завдань та вправ..

Якість оформлення роботи Магістерська кваліфікаційна робота, яку подано на

рецензію, виконана у повному обсязі в установлений термін. Робота має чітку структуру та належну ілюстрацію. Вона висвітлює основний зміст проблеми, яку вирішує, та шляхи її розв'язання..

Недоліки в роботі Розділ, присвячений аналізу та обґрунтуванню ефективності використання технологій у процесі освіти, може бути розгорнутим з більшою деталізацією. В розділі висновків слід включити конкретні рекомендації для подальшого вдосконалення бота або розширення його можливостей.

Загальний висновок Магістерська кваліфікаційна робота виконана у
(підготовленість студента до самостійної роботи як спеціаліста)

відповідності з завданням із дотриманням всіх вимог.

Оцінка магістерської роботи Робота заслуговує оцінки «добре», а студент Ярощук О.Р. – присвоєння кваліфікації «магістра» з Інженерії програмного забезпечення.

Рецензент _____
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання, посада, місце роботи)

«___»_____20__ р. _____
(підпис)

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ОПТИЧНОМУ НОСІЇ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом_Ярощук_О.Р.doc	Пояснювальна записка до проекту. Документ Word.
Диплом_ Ярощук_О.Р.pdf	Пояснювальна записка до проекту в форматі PDF.
Програма	
Programm.rar	Архів, що містить коди програми для запуску проекту.
Презентація	
Презентація_Ярощук.ppt	Презентація для проекту.

ДОДАТОК Г. main.py

```
import logging
from bd import get_dict_blocks, get_blocks_list, get_blocks_list1,
get_rus_name_and_id, get_eng_words, set_user, \
    set_user_block, get_done_blocks
import config
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.utils import executor
from aiogram.contrib.fsm_storage.memory import MemoryStorage
from aiogram.contrib.middlewares.logging import LoggingMiddleware
from config import BOT_TOKEN
from messages import MESSAGES, get_block_message_start
import keyboard as kb
url
="https://api.telegram.org/bot1320775247:AAHT5VzNVr4tnriNowopkLnl3vBWLx4
oWHw/getUpdates"
logging.basicConfig(format=u'%(filename)+13s [LINE:%(lineno)-4s] %(levelname)-
8s [%(asctime)s]%(message)s',
level=logging.DEBUG)
bot = Bot(token=BOT_TOKEN)
dp = Dispatcher(bot, storage=MemoryStorage())
dp.middleware.setup(LoggingMiddleware())
GROUP_ID = -448197357
block_list = get_blocks_list()
block_list1 = get_blocks_list1()
# eng_words = []
count = 0
check = -1
async def update_text_blocks(message: types.Message,
dict):await message.edit_text(MESSAGES['blocks_message'],
reply_markup=kb.getMarkup(dict))
async def update_text_menu(message: types.Message):await
message.edit_text(MESSAGES['enter'],
reply_markup=kb.inline_kb_menu)
async def update_text_blocks_start(message:
types.Message, word_count, rus_name, block):await
message.edit_text(get_block_message_start(word_count, rus_name),
```

```
reply_markup=kb.getStartBlockMarkup(block))
async def update_text_words(message: types.Message):
    await message.edit_text("Слово: " + config.eng_words[count] + ".\n Виберіть
переклад", reply_markup=kb.getRusWordsMarkup(config.eng_words[count]))
async def update_text_final_words(message:
types.Message): await message.edit_text("Вітаю! Ви пройшли всі слова з цього
блоку.", reply_markup=kb.inline_kb_menu)
async def update_text_settings(message: types.Message): await
message.edit_text("Налаштування",
reply_markup=kb.getSettingsMarkup())
async def update_text_done_blocks(message:
                                  types.Message, blocks):
    result = "Ви пройшли блоки: "
    for s in blocks:
        result = result + s + ", "
    result = result[:len(result) - 2]
    await message.edit_text(result, reply_markup=kb.getMenuOnlyMarkup())

@dp.callback_query_handler(lambda c: c.data == 'choose_block')
async def process_callback_choose_block(callback_query: types.CallbackQuery):
    await bot.send_message(callback_query.from_user.id, 'Натиснута перша
кнопка!')
    await update_text_blocks(callback_query.message, get_dict_blocks())
    print(callback_query.from_user.id)
    await bot.answer_callback_query(callback_query.id)

@dp.callback_query_handler(lambda c: c.data == 'menu')
async def process_callback_menu(callback_query:
types.CallbackQuery):
    await update_text_menu(callback_query.message)
    await bot.answer_callback_query(callback_query.id)
@dp.callback_query_handler(lambda c: c.data in block_list)
async def process_callback_blocks(callback_query: types.CallbackQuery):
    print(callback_query.data)
    word_count, rus_name = get_rus_name_and_id(callback_query.data)
    await update_text_blocks_start(callback_query.message,
word_count, rus_name, callback_query.data)
    await bot.answer_callback_query(callback_query.id)
@dp.callback_query_handler(lambda c: c.data in block_list1)
```

```
async def process_callback_start_block(callback_query: types.CallbackQuery):
    global count
    count = 0
    config.eng_words = get_eng_words(callback_query.data)
    print(config.eng_words)
    await update_text_words(callback_query.message)
    await bot.answer_callback_query(callback_query.id)
@dp.callback_query_handler(lambda c: c.data == "0" or c.data == "1" or c.data ==
"2" or c.data == "3" or c.data == "10")
async def process_callback_check(callback_query: types.CallbackQuery):
    global count
    count = count + 1
    if count == len(config.eng_words):
        await update_text_final_words(callback_query.message)
        set_user_block(callback_query.from_user.id)
    elif callback_query.data == "10":
        await update_text_words(callback_query.message)
    else:
        await update_text_blocks(callback_query.message,
get_dict_blocks())
    await bot.answer_callback_query(callback_query.id)

@dp.callback_query_handler(lambda c: c.data == "settings")
async def process_callback_settings(callback_query:
types.CallbackQuery):
    await update_text_settings(callback_query.message)
    await bot.answer_callback_query(callback_query.id)

@dp.callback_query_handler(lambda c: c.data == "done")
async def process_callback_done(callback_query:
types.CallbackQuery):
    done_blocks = get_done_blocks(callback_query.from_user.id)
    await update_text_done_blocks(callback_query.message, done_blocks)
    await bot.answer_callback_query(callback_query.id)

@dp.message_handler(commands=['start'])
async def start_message(message: types.Message):
    set_user(message.from_user.id)
    await message.reply(MESSAGES['start'],
```

```
reply_markup=kb.inline_kb_menu, reply=False)

@dp.message_handler(commands=[MESSAGES['test_message']])
async def help_message(message: types.Message):
    await message.reply(MESSAGES['enter'])

if __name__ == '__main__':
    executor.start_polling(dp)
```

ДОДАТОК Д. messages.py

```
enter_message = 'Бот допоможе Вам у навчанні англійської мови. \n3 його допомогою ви легко зможете набрати базу слів для вивчення мови.' \

start_message = 'Привіт!\n' + enter_message
invalid_key_message = 'Ключ "{key}" не підходить.\n' + enter_message
state_reset_message = 'Стан успішно скинуто'
current_state_message = 'Поточний стан - "{current_state}", що задовольняє умову "один з {states}"'
blocks_message = 'Виберіть блок для вивчення'
blocks_message_start = 'Ви обрали блок ... Він містить ... слів'
blocks_message_words = 'Тут слово з БД'

MESSAGES = {
    'start': start_message,
    'enter': enter_message,
    'invalid_key': invalid_key_message,
    'state_reset': state_reset_message,
    'current_state': current_state_message,
    'blocks_message': blocks_message,
    'blocks_message_start': blocks_message_start,
    'blocks_message_words': blocks_message_words,
}

def get_block_message_start(word_count, rus_name):
    return "Ви обрали блок " + rus_name + ". Він містить " + str(word_count) + " слів."
```

ДОДАТОК Е. keyboard.py

```
import random
from aiogram.types import ReplyKeyboardRemove, \
ReplyKeyboardMarkup, KeyboardButton, \
InlineKeyboardMarkup, InlineKeyboardButton
from messages import MESSAGES
from bd import get_rus_right_words, get_other_rus_words


inline_btn_choose_block = InlineKeyboardButton('Вибір блоку',
callback_data='choose_block')
inline_btn_settings = InlineKeyboardButton('Налаштування',
callback_data='settings')
inline_btn_menu = InlineKeyboardButton('Головне меню', callback_data='menu')
inline_kb_menu = InlineKeyboardMarkup().add(inline_btn_choose_block,
inline_btn_settings)


def getMarkup(blockDict):
    print(blockDict)
    markup = InlineKeyboardMarkup()
    for key, value in blockDict.items():
        markup.add(InlineKeyboardButton(value, callback_data=key))
    markup.add(inline_btn_menu)
    return markup


def getStartBlockMarkup(block):
    markup = InlineKeyboardMarkup()
    block = block + '1'
    print(block)
    text = "Почати"
    markup.add(InlineKeyboardButton(text, callback_data=block),
inline_btn_menu)
    return markup


def getRusWordsMarkup(word):
```

```
    right_words = get_rus_right_words(word)
    other_words = get_other_rus_words(word)
    sum_words = [right_words[0], other_words[0], other_words[1], other_words[2]]
    random.shuffle(sum_words)
    answerId = sum_words.index(right_words[0])
    markup = InlineKeyboardMarkup(row_width=2)
    for idx, val in enumerate(sum_words):
        if idx == answerId:
            markup.add(InlineKeyboardButton(val, callback_data="10"))
        else:
            markup.add(InlineKeyboardButton(val, callback_data=str(idx)))
    markup.add(inline_btn_menu)
    return markup


def getSettingsMarkup():
    markup = InlineKeyboardMarkup()
    markup.add(InlineKeyboardButton("Що пройдено", callback_data="done"),
        InlineKeyboardButton("Нагадування", callback_data="remember"),
        inline_btn_menu)
    return markup


def getMenuOnlyMarkup():
    markup = InlineKeyboardMarkup()
    markup.add(inline_btn_menu)
    return markup
```

ДОДАТОК Ж. config.py

```
BOT_TOKEN = '6702976447:AAEUD6IZg36U5RcLl_rixJPpXoeML_nLdCE'

CHANNEL_NAME = '@English_Studing_Bot'

eng_words = []
```

ДОДАТОК 3. bd.py

```
import sqlite3
import logging
import random
import config


def get_dict_blocks():
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Name, R_Name from Blocks")
    result = {}
    for row in cur:
        result[row[0]] = row[1]
    con.close()
    return result


def get_blocks_list():
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Name from Blocks")
    result = []
    for row in cur:
        result.append(row[0])
    con.close()
    return result


def get_blocks_list1():
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Name from Blocks")
    result = []
    for row in cur:
        result.append(row[0] + "1")
    con.close()
    return result
```

```
def get_rus_name_and_id(name):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select R_Name from Blocks where Name=\"" + name + "\"")
    row = cur.fetchone()
    rus_name = row[0]
    cur.execute("select count(*) from EWords where Block_id=(select Id from Blocks
where Name=\"" + name + "\")")
    row = cur.fetchone()
    word_count = row[0]
    con.close()
    return word_count, rus_name


def get_eng_words(name1):
    name = name1.replace("1", "")
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Word from EWords where Block_id= (select Id from Blocks
where Name=\"" + name + "\")")
    result = []
    for row in cur:
        result.append(row[0])
    con.close()
    random.shuffle(result)
    return result


def get_rus_right_words(word):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Word from Translations inner join RWords on
Translations.RId = RWords.Id where EId = (select Id from EWords where Word =
\"" + word + "\")")
    result = []
    for row in cur:
        result.append(row[0])
```

```
    con.close()
    random.shuffle(result)
    return result


def get_other_rus_words(word):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Word from Translations inner join RWords on
Translations.RId = RWords.Id where EId != (select Id from EWords where Word =
\"" + word + "\")")
    result = []
    for row in cur:
        result.append(row[0])
    con.close()
    random.shuffle(result)
    return result


def get_name_block_by_word(word):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Name from Block where Id = (select Block_id from EWords
where Word = \"" + word + "\")")
    row = cur.fetchone()
    result = row[0]
    con.close()
    return result


def set_user(id):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    print(id)
    cur.execute("insert or ignore into Users(Telegram_id) values(" + str(id) + ")")
    con.commit()
    con.close()
```

```
def set_user_block(id):
    print(config.eng_words)
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("select Block_id from EWords where Word = \"" +
config.eng_words[0] + "\"")
    row = cur.fetchone()
    block_id = row[0]
    user_id = []
    block_completed = []
    cur.execute("select Block_id from User_Block where User_id = " + str(id))
    for row in cur:
        block_completed.append(row[0])

    if block_id not in block_completed:
        con.cursor().execute("insert into User_Block(User_id, Block_id) values(" +
str(id) + ", " + str(block_id) + ")")
    print("sssss", id, block_id)
    con.commit()

    con.close()


def get_done_blocks(id):
    con = sqlite3.connect("DataBase.db")
    cur = con.cursor()
    cur.execute("SELECT R_Name from Blocks inner join User_Block on
User_Block.Block_id = Blocks.Id where User_Block.User_id = " + str(id))
    result = []
    for row in cur:
        result.append(row[0])
    return result
```