КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

Кафедра теорії та технології програмування

Кваліфікаційна робота

на здобуття ступеня бакалавра

за спеціальністю 122 «Комп'ютерні науки»
на тему:

АСИНХРОННИЙ ТЕЛЕГРАМ-БОТ З ПОШУКУ ТА СКАЧУВАННЯ КНИГ

Виконав студент 5-го курсу
заочної форми навчання
Вадим ГАЛЕНОК

(підпис)

Науковий керівник:
кандидат фіз.-мат. наук
Андрій КРИВОЛАП

(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент

(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування

«___»_____2022 р.,

протокол № _____

Завідувач кафедри
Микола НІКІТЧЕНКО

(підпис)

Київ-2022

РЕФЕРАТ

Обсяг роботи 37 сторінок, 14 ілюстрацій, 8 таблиць, 22 джерел посилань.

ТЕЛЕГРАМ-БОТ, ЧАТ-БОТ, МЕСЕНДЖЕР, ІНТЕРФЕЙС, ПОШУК КНИГ, АСИНХРОННІСТЬ, PYTHON.

Об’єктом роботи є процес пошуку та скачування книг за допомогою Телеграм-боту.

Предметом роботи є програмний засіб для пошуку та скачування книг.

Метою роботи є створення асинхронного Телеграм-боту з пошуку та скачування книг.

Методи розроблення: аналіз, узагальнення, опис, розробка програмного продукту. Інструменти розроблення: мова програмування Python, асинхронна бібліотека aiogram 2.14, мова структурованих запитів SQL.

Результати роботи: виконано загальний огляд чат-ботів для месенджерів, проаналізовано використання Телеграм-ботів як ефективного засобу отримання інформації, розроблено програмний продукт асинхронний Телеграм-бот, який дозволяє за запитом користувача пошук та скачування книг.

Програмний продукт Асинхронний Телеграм-бот може застосовуватися будь-якою людиною для пошуку та скачування електронних книг. Згідно зі статистикою, найчисленніша група людей, які активно користуються як інтернетом, так і можливостями електронних пристроїв та різноманітних додатків, це школярі та студенти.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

AI (Artificial Intelligence) – Штучний інтелект

ML (Machine Learning) – Машинне навчання

SQL (Structured Query Language) – Мова структурованих запитів

ВСТУП

Оцінка сучасного стану об’єкта розробки. Інтернет у сучасних умовах – це універсальне середовище для спілкування, розваг та навчання. В даний час у світі існує велика кількість засобів, форм і способів спілкування, і чимала частина з них так чи інакше пов'язана із сучасними технічними можливостями, які, зокрема, представлені за допомогою глобальної комп'ютерної мережі.

Свій розвиток інтернет-сервіси для спілкування розпочали з чатів, потім месенджери, потім соціальні мережі, але з недавніх пір месенджери знову очолили список найперспективніших сервісів. Причина повторної хвилі популярності месенджерів – зміни в галузі мобільного Інтернету: вища швидкість, набагато нижчі ціни, ніж раніше, широке поширення смартфонів.

Прогрес цієї технології такий, що такі компанії, як Facebook, запустили API-інтерфейси, які дозволяють брендам адаптувати та використовувати ботів у своїх месенджерах для спілкування зі своїми клієнтами [12]. Але є ще одна область, де чат-боти можуть мати величезний потенціал – це освіта.

Електронні книги у суспільстві давно стали заміною паперовим виданням. На сьогодні вони є майже у кожного користувача в телефоні, планшеті та електронній книзі. Це може бути не просто книга, а й підручник з предмета, довідник чи навіть енциклопедія.

За тематикою чат-ботів існує дуже мало літератури серед українських науковців.

Актуальність роботи та підстави для її виконання. Сучасний ритм життя надзвичайно динамічний, не останньою передумовою цього став стрімкий розвиток інформаційних технологій. В даний час різноманітні гаджети і додатки націлені на те, щоб максимально спростити життя звичайних користувачів у вирішенні повсякденних завдань, а також забезпечити досить швидкий доступ до будь-якої інформації, що їх цікавить.

Актуальність дослідження зумовлена високою популярністю месенджерів та засобів автоматизації таких як чат-боти серед користувачів мережі Інтернет.

Чат-боти дозволяють спростити щоденні рутинні завдання, такі як отримання інформації про погоду, останні новини та інші. Головною перевагою щодо класичних додатків є можливість поєднання всіх можливостей на платформі одного месенджера.

Мета й завдання роботи. Метою роботи є створення асинхронного Телеграм-боту з пошуку та скачування книг. Для досягнення цієї мети поставлено наступні завдання:

Об'єкт, методи й засоби розроблення. Об’єктом роботи є процес пошуку та скачування книг за допомогою Телеграм-боту. Методи розроблення: аналіз, узагальнення, опис, розробка програмного продукту. Інструменти розроблення: середовище розробки Eclipse з розширенням PyDev, мова програмування Python.

Можливі сфери застосування. Програмний продукт Асинхронний Телеграм-бот може застосовуватися будь-якою людиною для пошуку та скачування електронних книг. Згідно зі статистикою, найчисленніша група людей, які активно користуються як інтернетом, так і можливостями електронних пристроїв та різноманітних додатків, це школярі та студенти.

1 МЕТОДОЛОГІЧНІ ОСНОВИ ПОНЯТТЯ ЧАТ-БОТ ДЛЯ МЕСЕНДЖЕРІВ

1.1 Поняття чат-бот та його види

Чат-бот – це програмне забезпечення або комп’ютерна програма, яка імітує людську розмову за допомогою текстової або голосової взаємодії [15].

По суті, чат-бот – це комп’ютерна програма або віртуальний помічник, який дає змогу людям взаємодіяти з технологіями. Коли перші боти були розроблені в 1960-х роках, існували лише текстові боти, звідси й назва «чат-бот».

З тих пір технології розвивалися, і існують не тільки текстові боти, але й голосові боти, які здатні обробляти людську мову. Однак через оригінальний термін текстові боти та голосові боти називаються просто «чат-ботами» [21].

Чат-боти можуть полегшити користувачам пошук потрібної інформації, відповідаючи на їхні запитання та запити – за допомогою введення тексту, аудіо без участі людини.

Технологія чат-ботів сьогодні існує майже всюди, від розумних динаміків вдома до програм для обміну повідомленнями на робочому місці. Останні чат-боти з штучним інтелектом часто називають «віртуальними помічниками» або «віртуальними агентами». Вони можуть використовувати аудіовхід, як-от Apple Siri, Google Assistant і Amazon Alexa, або взаємодіяти з користувачами за допомогою SMS-повідомлень. У будь-якому випадку, є можливість задавати питання про те, що потрібно, у розмовній формі, а чат-бот може допомогти уточнити пошук за допомогою відповідей та наступних запитань [4].

В основному на ринку існує два типи чат-ботів – зі штучним інтелектом або без нього. Чат-боти без штучного інтелекту засновані на наборі правил і заздалегідь визначених команд. Вони можуть відповідати лише на конкретні завдання. Чат-боти на основі машинного навчання та штучного інтелекту здатні розуміти складні запити та поступово покращувати якість відповідей.

Користувачі все частіше використовують віртуальних помічників чат-ботів для виконання простих завдань.

Оскільки чат-боти все ще є відносно новою бізнес-технологією, не існує їх чіткої категоризації, кількості різних типів чат-ботів та їх назв.

Деякі поширені типи чат-ботів наступні [8]:

- Скриптові або чат-боти швидкої відповіді. Вони діють як ієрархічне дерево рішень. Ці боти взаємодіють з користувачами за допомогою попередньо визначених запитань, які прогресують, поки чат-бот не відповість на запитання користувача.
- Подібним до скриптового бота є чат-бот на основі меню, який вимагає від користувачів робити вибір із попередньо визначеного списку або меню, щоб надати боту глибше розуміння того, що потрібно користувачу.
- Чат-боти на основі розпізнавання ключових слів. Вони намагаються прислухатися до того, що вводить користувач, і відповідно реагувати, використовуючи ключові слова з відповідей користувачів. Цей бот поєднує ключові слова та штучний інтелект, що налаштовуються, щоб належним чином реагувати. На жаль, ці чат-боти борються з повторюваним використанням ключових слів або зайвими запитаннями.
- Гібридні чат-боти. Ці чат-боти поєднують елементи ботів на основі меню та розпізнавання ключових слів. Користувачі можуть обрати напрямок відповіді на свої запитання або скористатися меню чат-бота, щоб вибрати, якщо розпізнавання ключових слів неефективне.
- Контекстні чат-боти. Ці чат-боти складніші за інші та вимагають зосередженості на даних. Вони використовують AI та ML, щоб запам’ятати розмови та взаємодію користувачів, і використовують ці спогади для зростання та покращення з часом. Замість того, щоб покладатися на ключові слова, ці боти використовують те, що запитують користувачі і як вони це запитують, щоб надати відповіді та самовдосконалюватися.
- Голосові чат-боти. Чат-боти з підтримкою голосу використовують усний діалог користувачів як вхід, який підказує відповіді або творчі завдання.

Розробники можуть створювати ці чат-боти, використовуючи API перетворення тексту в мовлення та розпізнавання голосу. Приклади включають Amazon Alexa і Siri від Apple.

1.2 Застосування чат-ботів

Чат-боти використовуються в програмах обміну миттєвими повідомленнями та онлайн-інтерактивних іграх протягом багатьох років і лише нещодавно перейшли в продажі та послуги B2C та B2B [11].

Застосування елементів штучного інтелекту у сфері електронної комерції та послуг є якщо і не головним, то одним з основних трендів сучасного е-маркетингу. Серед них найбільш потрібні програмовані модулі, що дозволяють взаємодіяти з користувачами в режимі реального часу. Вони відомі під назвою «чат-боти» і є алгоритмами, що реалізуються в рамках месенджерів (Facebook, Telegram, Skype, Viber, WhatsApp та ін.) [13].

Організації можуть використовувати чат-боти наступними способами [15]:

- Інтернет-магазини. У таких середовищах відділи продажів можуть використовувати чат-боти, щоб відповісти на нескладні запитання про продукт або надати корисну інформацію, яку споживачі зможуть шукати пізніше, включаючи ціну доставки та наявність.
- Обслуговування клієнтів. Відділи обслуговування також можуть використовувати чат-боти, щоб допомогти агентам обслуговування відповідати на повторювані запити. Наприклад, представник служби може дати чат-боту номер замовлення та запитати, коли замовлення було відправлено. Як правило, чат-бот передає дзвінок або текстове повідомлення агенту з обслуговування людей, коли розмова стає занадто складною.
- Віртуальні помічники. Чат-боти також можуть виконувати роль віртуальних помічників. Apple, Amazon, Google і Microsoft мають різні форми віртуальних помічників. Додатки, такі як Siri від Apple і Cortana від Microsoft, або

продукти, як Amazon Echo з Alexa або Google Home, відіграють роль особистого чат-бота.

Крім швидкості та простоти замовлення, чат-бот має ще кілька переваг для користувачів [2]:

- Знеособлене замовлення. Наприклад, багато компаній продають через Instagram активніше, ніж через сайт. Стрічка та сторіз виступають у ролі вітрини, а щоб зробити замовлення, клієнту потрібно написати в директ, месенджер або зателефонувати продавцеві. Але не всі люблять зайвий раз спілкуватися голосом – набагато простіше замовити в чаті та чекати на доставку, особливо для молоді. Тому чат-бот – це ще й перевага перед конкурентами, які не можуть зробити замовлення, не поспілкувавшись із продавцем.
- Асинхронність. Бот моментально видає потрібну інформацію, але не вимагає такої ж швидкої відповіді від клієнта. Це зручно – можна взяти паузу та повернутися до обговорення або замовлення пізніше.
- Поділ для клієнтів із різних міст. За допомогою чат-бота можна пропонувати клієнтам з різних міст різне меню, ціни та можливості доставки. І робити це тут також зручніше, ніж на сайті.

1.3 Огляд існуючих месенджерів

Боти можуть спілкуватися як за допомогою голосу, так і через текст, і їх можна розгорнути на веб-сайтах, у програмах і каналах обміну повідомленнями, таких як Telegram, Facebook Messenger, Twitter або Whatsapp.

Telegram (рис. 1.1) — популярний додаток для обміну повідомленнями, який широко використовується, оскільки він пропонує деякі покращені функції конфіденційності та шифрування, а також підтримку функцій чату для великих груп. Він також не пов’язаний з іншими платформами соціальних мереж (наприклад, Facebook Messenger і WhatsApp належать Facebook), що робить сервіс більш привабливим [16].

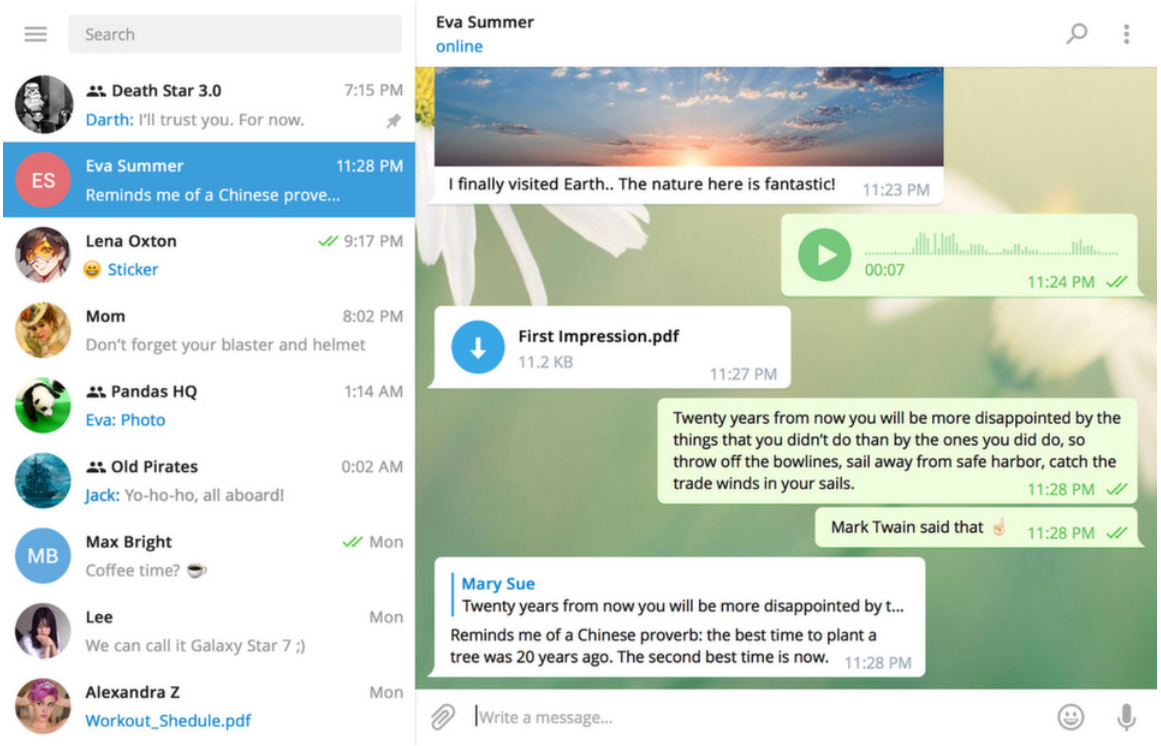

Рисунок 1.1 – Інтерфейс Telegram

Додаток є багатоплатформним, версії програми доступні для iOS, Android, Windows, Mac і Linux. Також можливо отримати доступ до Telegram з веб-браузера.

Telegram був заснований підприємцем у соціальних мережах Павлом Дуровим, цей сервіс безкоштовний.

Ось деякі плюси використання Telegram:

- Наскрізне шифрування: під час використання режиму секретного чату спілкування повністю шифрується від початку до кінця, що робить спілкування максимально безпечним. Також можна отримати наскрізне шифрування в таких програмах, як WhatsApp і Signal.
- Повідомлення, що самознищуються: секретні повідомлення можна налаштувати на самознищення через певний період часу, що робить їх ще більш безпечними. Це схоже на те, що ви можна робити в таких програмах, як Snapchat, Instagram та Facebook Messenger.
- Великі розміри файлів: Telegram підтримує вкладення файлів розміром до 2 ГБ. Це одна з областей, де Telegram перевершує практично всі інші

програми для обміну повідомленнями. Більшість додатків набагато більш обмежені — наприклад, WhatsApp становить лише 16 МБ.

Facebook Messenger (рис. 1.2) – це мобільний додаток, який дозволяє спілкуватися в чаті, голосі та відео між веб-сайтом соціальних мереж і смартфонами. (Конкретні можливості відрізняються залежно від пристрою та географічного розташування користувача.) Messenger доступний для пристроїв iOS, Android, Windows 10 і Blackberry і може підключатися через Wi-Fi або мобільний тарифний план [12].

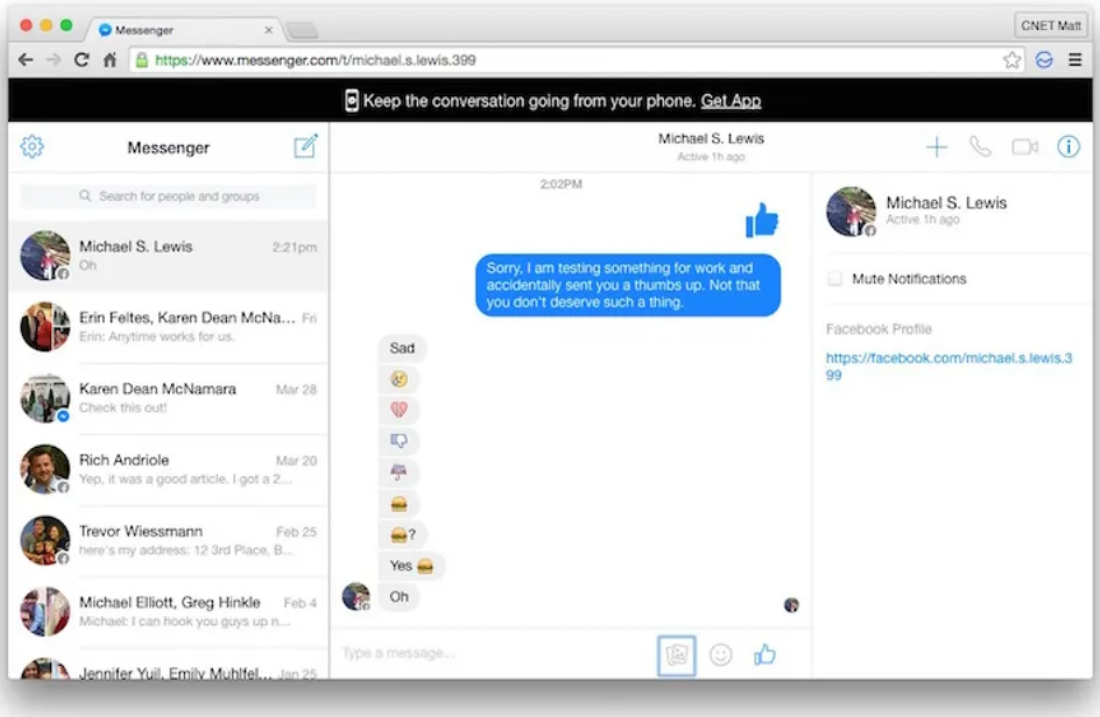

Рисунок 1.2 – Інтерфейс Facebook Messenger

У системах Android додаток також інтегрує SMS-повідомлення, щоб користувачам не доводилося перемикатися між інтерфейсами зв’язку для різних контактів. Після того, як користувач вибере Messenger як SMS-клієнт за замовчуванням, через інтерфейс стануть доступні текстові повідомлення та чат у Facebook. SMS-повідомлення та ланцюжки мають фіолетовий колір, щоб відрізняти їх від повідомлень Facebook із синім кольором.

На конференції розробників F8 у Facebook у 2016 році компанія повідомила, що впроваджує чат-ботів для спілкування. Facebook також оголосив, що платформа Messenger стане вільно доступною для розробки ботів. Користувачі можуть взаємодіяти з бізнес-ботом, як і з будь-яким іншим контактом, замість того, щоб завантажувати окремі програми для кожної компанії та перемикатися між додатками для спілкування. У Messenger користувач може перевірити спортивні результати чи погоду, зробити покупку, отримати квитки в кіно, забронювати столик у ресторані чи авіаквиток – серед багатьох інших можливостей.

Twitter (рис. 1.3) – соціальна мережа, запущена в 2006 році, безсумнівно, є однією з найпопулярніших платформ соціальних медіа, доступних сьогодні, зі 100 мільйонами активних користувачів щодня і 500 мільйонами твітів, що надсилаються щодня [18].

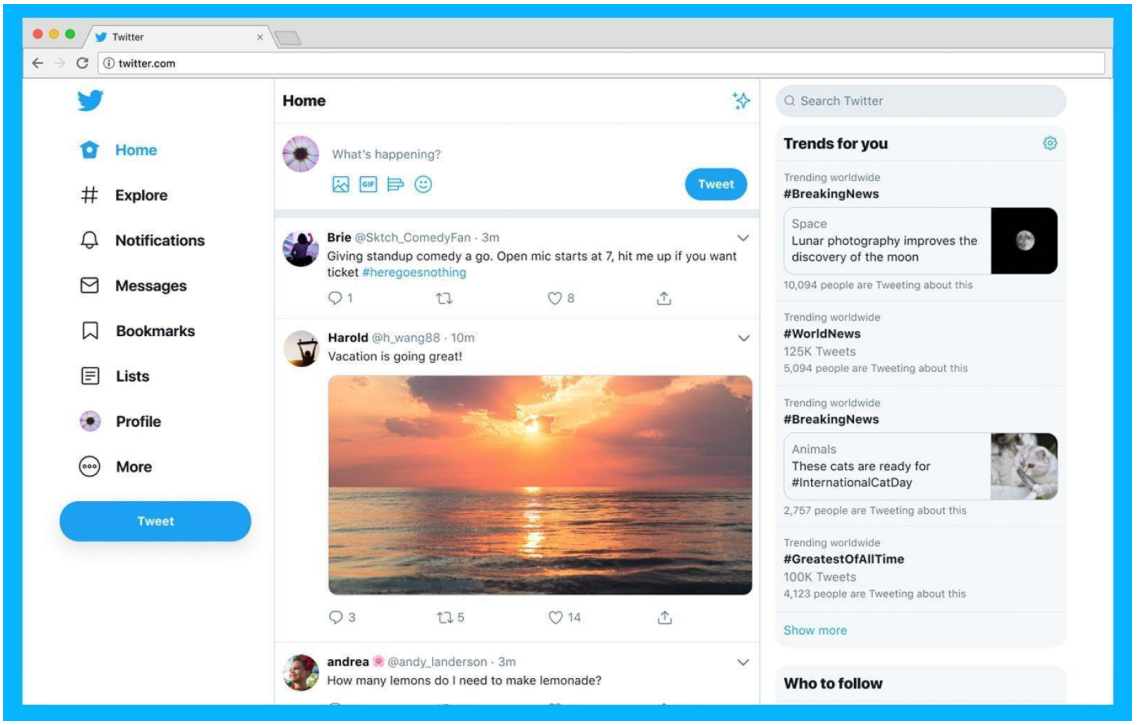

Рисунок 1.3 – Інтерфейс Twitter

Twitter можна використовувати, щоб отримувати новини, стежити за знаменитостями або залишатися на зв’язку з друзями.

Twitter може бути дуже корисною платформою для збільшення кількості читачів і надання аудиторії цінного контенту ще до того, як вони стануть клієнтами. Обмеження символів також може допомогти створити швидку та привабливу рекламу, як-от заклик до вебінару, який проводить компанія, або безкоштовну електронну книгу.

Twitter – це безкоштовний інструмент для обміну повідомленнями в соціальних мережах, який дозволяє користувачам залишатися на зв'язку за допомогою коротких текстових повідомлень довжиною до 140 символів. Соціальна мережа Twitter заснована на відповіді на питання «Що ти робиш?». Потім публікуються думки, спостереження, а також події протягом дня.

Оновлення цих повідомлень публікується на сторінці профілю користувача в Twitter за допомогою текстових повідомлень SMS, веб-сайту Twitter, миттєвих повідомлень, RSS, електронної пошти або через інші програми та соціальні сайти, такі як Facebook.

Зареєстровані користувачі Twitter можуть читати та публікувати твіти, але ті, хто не зареєстрований, можуть їх лише читати.

Сервіс обробляє 1,6 мільйона пошукових запитів на день. У 2013 році Twitter був одним із десяти найбільш відвідуваних веб-сайтів і був описаний як «SMS Інтернету». У травні 2019 року у Twitter було понад 126 мільйонів користувачів.

Користувачі також можуть групувати повідомлення за темами, використовуючи хештеги, які є словами або фразами, перед якими стоїть знак «#». Аналогічно, знак «@», за яким слідує ім'я користувача, використовується для згадки або відповіді іншим користувачам. Все це завжди в 140 символах.

Бот Twitter – це обліковий запис, запрограмований на виконання дій, таких як надсилання твітів у запланований час або підписка на облікові записи. Ці боти створюються та управляються через Twitter API.

Автоматизуючи певні завдання, боти Twitter можуть допомогти за менший час забезпечити активну присутність на одній з платформ соціальних мереж, що найбільш широко використовуються. За допомогою ботів Twitter можливо

запланувати публікацію твітів, коли користувач не в мережі. Є можливість автоматизувати відповіді нових передплатників, підписатися на акаунти або відписатися від них і багато іншого.

Запущений у 2009 році, WhatsApp (рис. 1.4) – це безкоштовна багатоплатформна програма для обміну повідомленнями, яка дозволяє користувачам здійснювати відео- та голосові дзвінки, надсилати текстові повідомлення, ділитися своїм статусом тощо лише за допомогою з'єднання Wi-Fi. Привабливим цей додаток робить те, що він працює на різних операційних системах телефонів і комп'ютерів, тому є можливість продовжувати розмову в будь-який час і в будь-якому місці. Він також може використовувати переваги Wi-Fi і стільникових даних для здійснення індивідуальних або групових дзвінків, зменшуючи потребу в дорогих дзвінках [22].

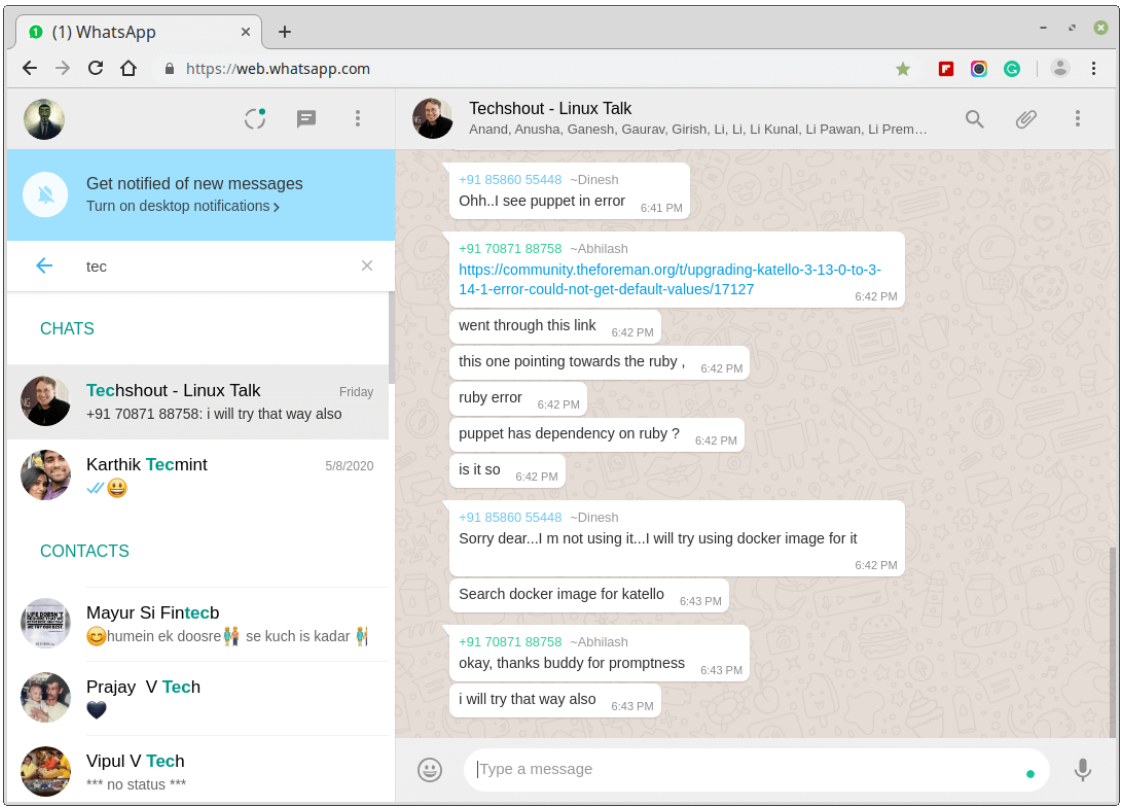

Рисунок 1.4 – Інтерфейс WhatsApp

WhatsApp використовує стільникове з'єднання або з'єднання Wi-Fi телефону, щоб полегшити обмін повідомленнями та голосові дзвінки практично будь-кому на планеті, поодинці чи в групі. Додаток дозволяє здійснювати дзвінки,

надсилати й отримувати повідомлення, а також ділитися документами, фотографіями та відео. WhatsApp є абсолютно безкоштовним — без комісій чи підписок — оскільки він використовує з'єднання 5G, 4G, 3G, 2G, EDGE або Wi-Fi замість голосових хвилин або текстового тарифного плану мобільного телефону. Якщо підключення через Wi-Fi, це також не вплине на тарифний план.

WhatsApp не залежить від платформи. Не потрібно володіти телефоном тієї ж марки, що й одержувач дзвінка, або бути на певній платформі — програма працює з телефонами iPhone і Android, а також на настільних або портативних комп'ютерах Mac або Windows, які можна використовувати для надсилання та отримання повідомлень, але не дзвонити. Як і в будь-якому іншому SMS-месенджері, можливо розпочати розмову з окремою людиною або групою та у відеочаті до восьми осіб. Версія для iOS також підтримує в програмі відтворення відео як з Instagram, так і з Facebook. Також можливо ділитися своїм місцезнаходженням, транслювати свій статус своїм контактам, ділитися контактами, налаштовувати фонові зображення та сповіщення, надсилати історію чату електронною поштою, використовувати камеру для зйомки фотографій та відео з програми та одночасно транслювати повідомлення кільком контактам.

Бот WhatsApp — це програма або програмне забезпечення чат-бота, яке можна використовувати спеціально в популярній програмі для обміну зашифрованими повідомленнями WhatsApp. Бот WhatsApp можна використовувати для діалогового маркетингу, продажів, рекламних акцій та може допомогти керувати запитами підтримки від клієнтів практично без людського контролю.

2 ТЕЛЕГРАМ-БОТ ЯК ЕФЕКТИВНИЙ ЗАСІБ ОТРИМАННЯ ІНФОРМАЦІЇ

2.1 Поняття Телеграм-бот

Завдяки швидкому росту компанії та вимогам щодо конфіденційності та безпеки Telegram став важливою платформою для бізнесу. Все більше компаній використовують платформи соціальних мереж для надання послуг підтримки клієнтів. Завдяки розширеній базі користувачів Telegram став важливою платформою [1].

Telegram – це служба обміну повідомленнями, яка дуже схожа на WhatsApp, але з додатковими перевагами шифрування даних, безпеки та конфіденційності. В Telegram якщо новий контакт підписується на бізнес-канал, компанія не отримує доступу до жодної з контактних або особистих даних користувачів [5].

У той же час користувачі Telegram можуть шукати назви компаній і підписатися на компанії через область відкриття. Ця домовленість працює таким чином, що клієнти можуть зв’язатися з компаніями, якщо вони хочуть, але компанії не можуть зв’язатися з клієнтами без їхнього дозволу.

Рівні Telegram – це об’єкти, з якими користувачі можуть взаємодіяти в Telegram. В останній версії Telegram є чотири типи однорангових користувачів [6]:

- Профіль користувача.

Профіль користувача – це особистий обліковий запис, який зазвичай реєструється під номером телефону користувача. Такий одноранговий тип використовується для особистих повідомлень, але іноді може виконувати роль адміністратора в каналах і групах.

- Група Telegram.

Група Telegram – це група користувачів і ботів, які називаються її учасниками. Група Telegram може мати максимум 200000 учасників, і всі учасники мають право публікувати повідомлення в групі. Адміністратор групи має

спеціальні привілеї та призначає право власності своїм учасникам, щоб дозволяти публікації, пов’язані з відео та зображеннями.

- Телеграм-канал.

Телеграм-канал – це підмножина груп Telegram, це спеціальна група, яка може мати необмежену кількість передплатників, але лише адміністратори та боти мають право публікувати повідомлення в каналі, усі інші користувачі каналу не можуть публікувати.

- Телеграм-бот.

Телеграм-бот – це особливий тип користувачів, який є не людиною, а комп’ютерною програмою, яка може обслуговувати компанії чи бренди з багатьма функціями, такими як надсилання інформації, нагадування, відтворення мелодій, замовлення тощо. Бот може опублікувати повідомлення в групі або каналі.

Звичайні користувачі можуть стежити за будь-яким телеграм-ботом. З усіх чотирьох однорангових типів бот Telegram має багато функцій, які дійсно корисні для бізнесу. Telegram надає API для створення ботів для соціальних взаємодій, продуктивності, ігор та послуг електронної комерції на платформі [17].

Крім цього, боти Telegram також можуть надавати підтримку клієнтам або збирати потенційних клієнтів, підключаючи їх до CRM, системи продажу квитків або платформи обміну повідомленнями. Оскільки щодня з’являються нові варіанти використання в бізнесі для використання ботів, боти стають все більш популярними для роботи в середовищі Telegram. Також велика вдячність API telegram, за допомогою яких ви легко створюєте телеграм-бота, рекламуєте його, щоб залучити більше користувачів для вашої ділової взаємодії.

Плюси використання Телеграм-ботів [7]:

- Безкоштовна платформа.

Telegram є безкоштовним, незалежно від кількості повідомлень і мети використання (для професійного чи особистого користування). Оскільки Telegram безкоштовний як для користувачів, так і для бізнесу, він дозволяє компаніям залучати потенційних клієнтів, ефективно використовуючи його. Настільки, що створення телеграм-бота також безкоштовне. Компанії можуть використовувати

цю безкоштовну платформу для створення своєї клієнтської бази, надсилаючи інформаційні бюлетені, надаючи підтримку клієнтів або взаємодіючи з ними.

- Краще залучення.

Якщо клієнтська база активно використовує Telegram, це чудова можливість залучити клієнтів, оскільки Telegram дозволяє користувачам надсилати аудіо, відео та зображення. Компанії можуть створити маркетингову кампанію, щоб залучити своїх клієнтів різними способами. Окрім маркетингу, деяким компаніям потрібно надавати підтримку клієнтів цілодобово, тому підприємства можуть створити чат-бот для Telegram або створити команду підтримки клієнтів, щоб майже миттєво відповідати на запити клієнтів. Це також дозволяє краще взаємодіяти з клієнтами.

- Безпечний.

Враховуючи глобальні тенденції щодо випадків злому на кожній платформі соціальних мереж, Telegram вважається досить безпечним, головним чином, тому що повідомлення надсилаються через платформу в зашифрованому вигляді. Це є достатньою причиною для використання його в бізнес-цілях, оскільки він захистить всі дані компанії та клієнтів. Telegram надає безпечне та краще місце для взаємодії щодо продуктів і послуг.

- Доступність.

Месенджер Telegram охоплює всі основні платформи, такі як телефони Android, iOS, Windows із настільними програмами для Mac, Linux та Windows. Крім того, він також має веб-версію, яка дозволить націлити потенційних клієнтів у дуже широкому масштабі. По-друге, не потрібно турбуватися, що клієнти не зможуть отримати доступ до цього месенджера. За допомогою бота Telegram можливо дозволити своїм клієнтам швидко отримати інформацію, і вони можуть використовувати її з будь-якого місця та в будь-який час без будь-яких проблем. Простота доступності збільшує можливість знайти потенційних покупців і підвищує коефіцієнт конверсії.

Телеграм-бот може допомогти збирати потенційних клієнтів, надавати підтримку клієнтам, надсилати інформаційні бюлетені наявним клієнтам,

показувати портфоліо, запускати маркетингові кампанії або автоматизувати деякі взаємодії [3].

Telegram – це не лише спосіб охопити потенційних клієнтів, націлити їх і підтримувати зв'язок із ними, але й потужний інструмент для автоматизації служб підтримки клієнтів та оптимізації багатьох процесів, щоб звести до мінімуму ручну роботу.

2.2 Класифікація Телеграм-ботів

Telegram – це процвітаюча програма для обміну повідомленнями, яка складається з усіх таких функцій, як Instagram, Whatsapp, Messenger тощо. Він пропонує більше, ніж просто обмін повідомленнями, голосові та відеодзвінки та обмін файлами. Telegram можна використовувати для інших цілей, таких як вивчення нової мови, отримання новин та сповіщень про останні оновлення, ігри, отримання нагадувань, перегляд фільмів, ігри, прослуховування пісень, продуктивність тощо. Він складається з усіх цих функцій мобільних додатків.

Існують різні типи ботів Telegram залежно від їх функціональності. Класифікація ботів Telegram на основі їх функцій [20]:

- Ігрові боти в Telegram пропонують своїм користувачам ігри HTML5, щоб грати самому або змагатися один з одним у групах та чатах. Ігри HTML5 – це сторонні ігрові програми, які працюють всередині програми Telegram. З моменту запуску ігрової платформи на основі бота 3 жовтня 2016 року Telegram став однією з найкращих платформ для гри з друзями в прості ігри.
- Боти для підвищення продуктивності в Telegram працюють як доповнення до програмних інструментів, які допомагають зробити користувачів більш організованими, менше зволікати і залишатися більш зосередженими у виконанні щоденних завдань. Коли бот підключений до Telegram, він працює, як і будь-які інші програми для продуктивності, і користувачам не потрібно буде завантажувати додаткові програми. Продуктивні боти бувають різних типів. Деякі

працюють як QR-сканери, які можливо використовувати в магазинах, деякі повідомляють новини та оновлення, а інші дозволяють проводити конференції.

- Навчальні боти Telegram – це ті боти, які допомагають покращити навчання та зосередитися. Серед різних типів знайдених ботів Telegram деякі навчальні боти є інтерактивними, тоді як інші роблять переклади, дають поради на незрозумілі теми, надають онлайн навчальні матеріали.
- Розважальні боти Telegram. Дозволяють дивитися фільми або слухати пісні.
- Службові боти Telegram. Для використання замість допоміжних програм (прогноз погоди, нагадування та ін.).

Однією з єдиних причин, чому Telegram опинився на першому місці, є його функції безпеки. Багато людей перейшли з WhatsApp на Telegram, оскільки у першого було багато проблем із хакерством.

Розглянемо деякі Телеграм-боти з пошуку та скачування книг:

- Аудіокниги (@audioknigi_channel). Мова: російська. Наявність рекламних постів та посилань на ютуб-канали. Кількість передплатників висока – 57 862 особи. Вказано час прослуховування файлів.
- Сила знань (@ReadKnigi) – telegram канал з аудіокнигами про найважливіше, що допоможе людині у її саморозвитку. Мова: російська. Посилання на сучасні твори, відомі у певних колах, авторів. Велике поле для всіх, хто любить шукати сенс життя і прагне до досягнення власних ідеалів. Вказано час на прослуховування аудіофайлів, що в середньому не перевищують 40 хвилин. Актуальна цифра передплатників – 68 676 осіб.
- Затишна бібліотека (@knijky) – на постах нові книги, учасники каналу обмінюються безкоштовними файлами. Мова: російська. Посилання на канали, авторські звернення до передплатників. Кількість передплатників – 19 262 особи. Завантажити книги в телеграм каналі Затишна бібліотека можна наступних форматів: FB2, EPUB та DOC.

2.3 Технологія реєстрації Телеграм-бота

Перш ніж починати розробку, бота необхідно зареєструвати і отримати його унікальний ID, що є одночасно і токеном [16].

Для початку потрібно зареєструватися в Telegram-месенджері. Далі, в мобільній, десктопній або web-версії месенджера необхідно відкрити спілкування з ботом @BotFather (рис. 2.1) або за посиланням https://telegram.me/botfather.

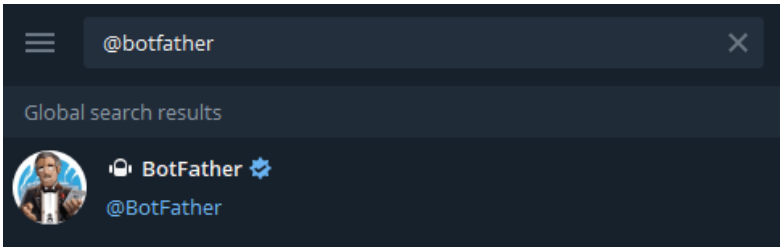

Рисунок 2.1 – Пошук бота @BotFather

Після початку спілкування з цим ботом, натиснувши /start, отримується відповідь від бота з зазначенням його можливостей. Потрібно обрати створення нового робота – /newbot (рис. 2.2).

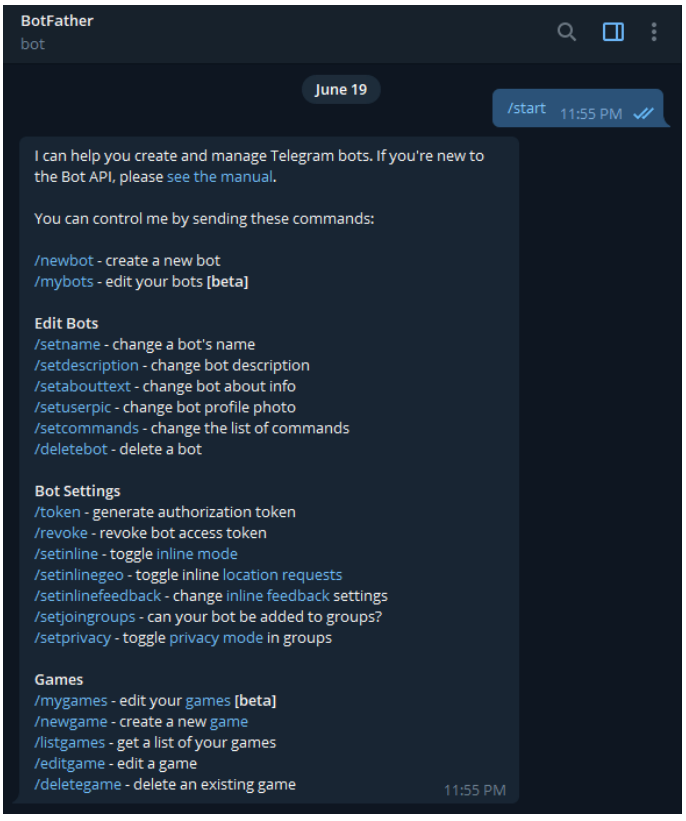

Рисунок 2.2 – Обрання команди

BotFather запропонує вибрати ім'я для майбутнього бота (рис. 2.3). Воно повинне бути унікальним, тому що в іншому випадку реєстрація бота буде відхилена. Наприкінці назви обов'язково потрібно вказати приставку _bot.

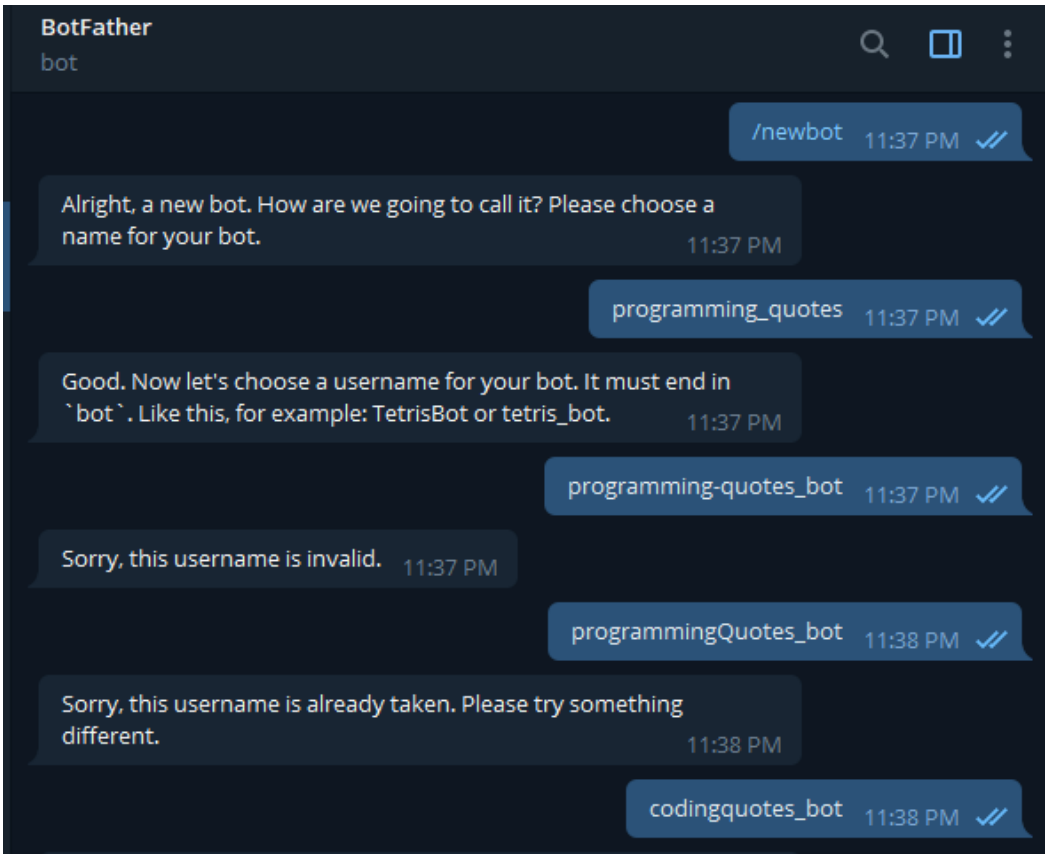

Рисунок 2.3 – Реєстрація імені бота

Після цього BotFather надсилає нам спеціальний токен (рис. 2.4).

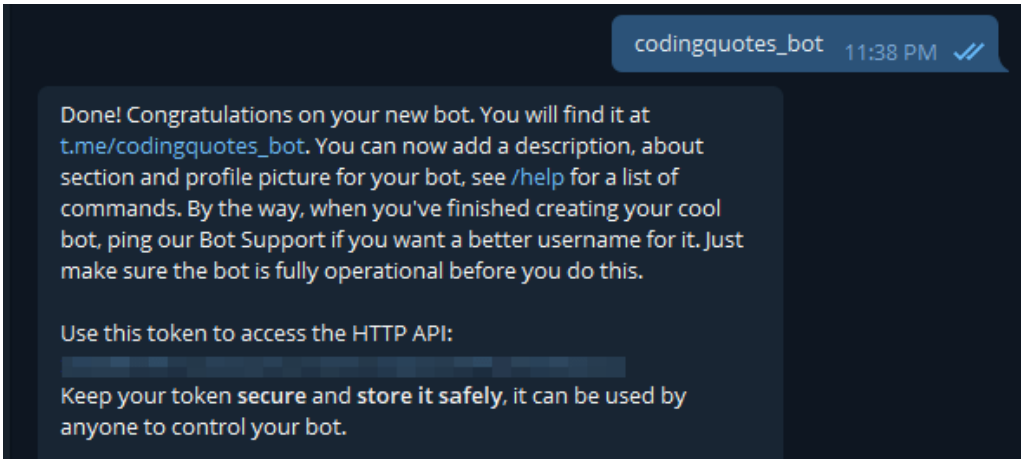

Рисунок 2.4 – Завершення реєстрації. Отримання токена та посилання

Цей токен знадобиться при налаштуванні бота. Токен має бути збережений. Саме він є єдиним ключем для взаємодії із ботом.

Також отримуємо посилання на бот. У цьому випадку це t.me/asdkldjqjwdjdjwdwdbot.

Після реєстрації можна переходити безпосередньо до створення самого бота.

3 ТЕХНОЛОГІЯ РОЗРОБКИ ТА СТРУКТУРА ТЕЛЕГРАМ-БОТУ

3.1 Вибір мови програмування

Python — це інтерпретована, об’єктно-орієнтована мова програмування високого рівня. Оскільки вона є універсальною, має широкий спектр застосувань від веб-розробки, створення графічного інтерфейсу робочого столу до наукових і математичних обчислень [14].

Основна мова для ботів – Python. Але писати бота можна будь-якою мовою програмування. Python обирають тому, що в його екосистемі є багато відкритих бібліотек для ботів, причому бібліотек на основі штучного інтелекту. Другий момент, чому його обирають – це відносна легкість у вивченні.

Найпопулярніші мови, якими пишуть ботів, виглядають наступним чином:

1. Python;
2. PHP;
3. Node.js;
4. Go;
5. C#;
6. Ruby;
7. Java;
8. Rust;
9. C++;
10. JavaScript.

Python популярна завдяки своєму простому та відносно зрозумілому синтаксису. Читабельність його синтаксису підвищує продуктивність, оскільки дозволяє більше зосередитися на проблемі, а не на структуруванні коду.

Python – це мова комп’ютерного програмування, яка часто використовується для створення веб-сайтів і програмного забезпечення, автоматизації завдань і аналізу даних. Python є мовою загального призначення, тобто її можна використовувати для створення різноманітних програм і не спеціалізується на

будь-яких конкретних проблемах. Ця універсальність разом із зручністю для початківців зробили її однією з найбільш використовуваних мов програмування сьогодні. Опитування, проведене аналітичною фірмою RedMonk, показало, що це друга за популярністю мова програмування серед розробників у 2021 році.

Python популярний з кількох причин. Ось більш глибокий погляд на те, що робить його настільки універсальним і простим у використанні для розробників:

Python має простий синтаксис, який імітує природну мову, тому його легше читати та розуміти. Це дозволяє швидше створювати проекти та швидше їх покращувати.

Python універсальний. Python можна використовувати для багатьох різних завдань, від веб-розробки до машинного навчання.

Python зручний для початківців, що робить його популярним для програмістів початкового рівня.

Це відкритий вихідний код, що означає, що його можна безкоштовно використовувати та поширювати навіть у комерційних цілях.

Архів модулів і бібліотек Python — пакети коду, які сторонні користувачі створили для розширення можливостей Python — величезний і зростає.

Python має велику та активну спільноту, яка вносить свій внесок у пул модулів і бібліотек Python, а також є корисним ресурсом для інших програмістів. Велика спільнота підтримки означає, що якщо розробники стикаються з каменем спотикання, знайти рішення відносно легко; хтось обов'язково стикався з такою ж проблемою раніше [14].

Python зазвичай використовується для розробки веб-сайтів і програмного забезпечення, автоматизації завдань, аналізу даних і візуалізації даних.

Python став основним продуктом науки про дані, дозволяючи аналітикам даних та іншим фахівцям використовувати цю мову для проведення складних статистичних обчислень, створення візуалізації даних, створення алгоритмів машинного навчання, маніпулювання та аналізу даних, а також виконання інших завдань, пов’язаних з даними.

Aiogram — це досить проста і повністю асинхронна бібліотека для Telegram Bot API, написана на Python 3.7 з asyncio і aiohttp. Це допомагає зробити ботів швидшими та простішими [9].

Для того, щоб зробити код чистішим і читабельнішим, aiogram розширює можливості стандартних об'єктів Telegram. Наприклад, замість bot.send_message(...) можна написати message.answer(...) або message.reply(...). В останніх двох випадках не потрібно підставляти chat_id, мається на увазі, що він такий самий, як і у вихідному повідомленні.

Різниця між answer і reply проста: перший метод просто відправляє повідомлення у той же чат, другий робить «відповідь» на повідомлення з message.

У aiogram механізм кінцевих автоматів реалізований набагато краще, ніж у програмі TelegramBotAPI. У фреймворк вже вбудована підтримка різних бекенд для зберігання станів між перезапусками бота, крім цього можна зберігати довільні дані.

Мова структурованих запитів (SQL) — це стандартизована мова програмування, яка використовується для управління реляційними базами даних і виконання різних операцій над даними в них. Спочатку створений у 1970-х роках, SQL регулярно використовують не лише адміністратори баз даних, а й розробники, що пишуть сценарії інтеграції даних, і аналітики даних, які прагнуть налаштувати й запустити аналітичні запити [19].

SQL використовується для наступного:

- зміна структури таблиць та індексів бази даних;
- додавання, оновлення та видалення рядків даних;
- отримання підмножин інформації з систем управління реляційними базами даних (RDBMS) – цю інформацію можна використовувати для обробки транзакцій, аналітичних додатків та інших програм, які потребують зв’язку з реляційною базою даних.

Запити SQL та інші операції мають форму команд, записаних у вигляді операторів, і об’єднані в програми, які дозволяють користувачам додавати, змінювати або отримувати дані з таблиць бази даних.

Таблиця є найпростішою одиницею бази даних і складається з рядків і стовпців даних. Одна таблиця містить записи, і кожен запис зберігається в рядку таблиці. Таблиці є найбільш використовуваним типом об’єктів бази даних або структур, які містять або посилаються на дані в реляційній базі даних. Інші типи об'єктів бази даних включають наступне [10]:

- Подання – це логічні представлення даних, зібрані з однієї або кількох таблиць бази даних.
- Індекси – це таблиці пошуку, які допомагають прискорити функції пошуку в базі даних.
- Звіти складаються з даних, отриманих з однієї або кількох таблиць, як правило, підмножини цих даних, які вибираються на основі критеріїв пошуку.

Кожен стовпець таблиці відповідає категорії даних, наприклад, імені або адреси клієнта, тоді як кожен рядок містить значення даних для стовпця, що перетинається.

Реляційні бази даних складаються з таблиць, які пов’язані одна з одною. Наприклад, база даних SQL, яка використовується для обслуговування клієнтів, може мати одну таблицю для імен та адрес клієнтів, а також інші таблиці, які містять інформацію про конкретні покупки, коди продуктів і контакти клієнтів. Таблиця, яка використовується для відстеження контактів клієнтів, зазвичай використовує унікальний ідентифікатор клієнта, який називається ключем або первинним ключем, щоб посилатися на запис клієнта в окремій таблиці, яка використовується для зберігання даних клієнтів, таких як ім’я та контактна інформація.

3.2 Графічний інтерфейс користувача Телеграм-боту

Розроблений Телеграм-бот виконує за запитом користувача пошук книг за автором (рис.3.1) чи серією книг (рис.3.2).

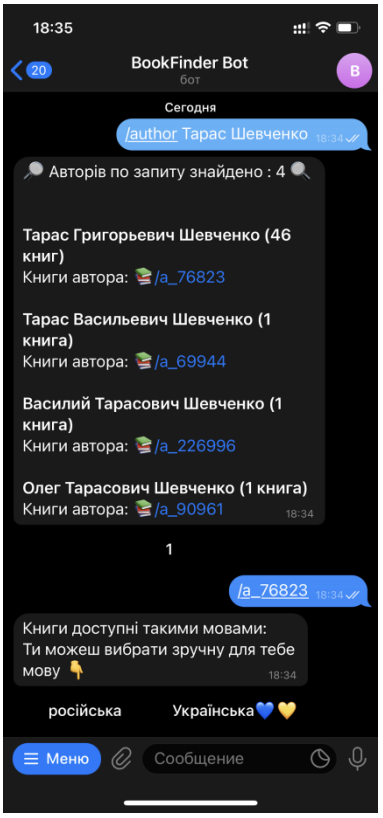

Рисунок 3.1 – Пошук за автором

Обрана база книг Flibusta, тому що вона має відкритий доступ API. Розробка є універсальною, тому можливо використовувати будь-яку базу. Додано обмеження пошуку книг, авторів і серій в сам бот, щоб бот не був заблокований за порушення авторських прав. Якщо додати бот у групу – обмежень немає.

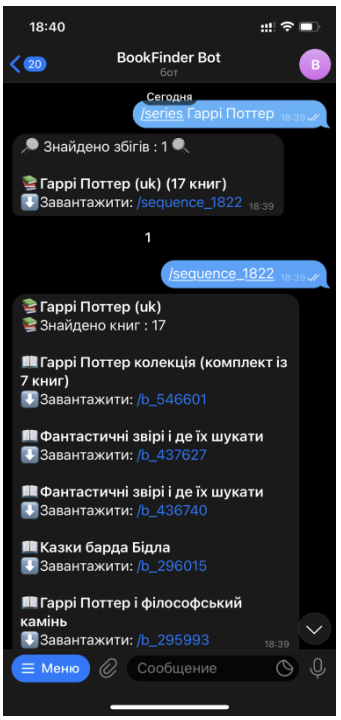

Рисунок 3.2 – Пошук за серією книг

Можливий вибір мови книг (за замовчуванням англійська, українська та російська мови), якщо здійснюється пошук за автором.

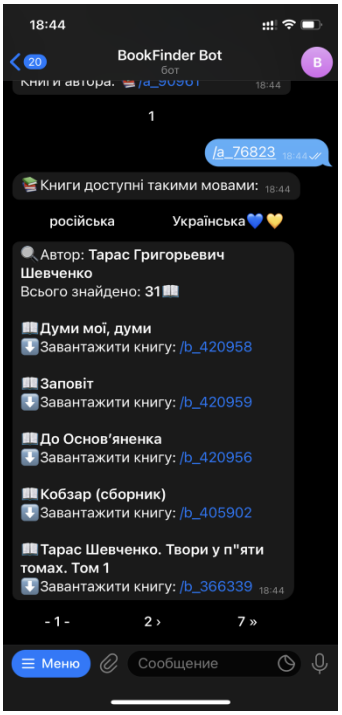

Рисунок 3.3 – Вибір мови книги

Безпосередньо введене повідомлення без команди – видасть доступні варіанти для вибору з інлайн кнопками (письменники, книги або серії) (рис.3.4).

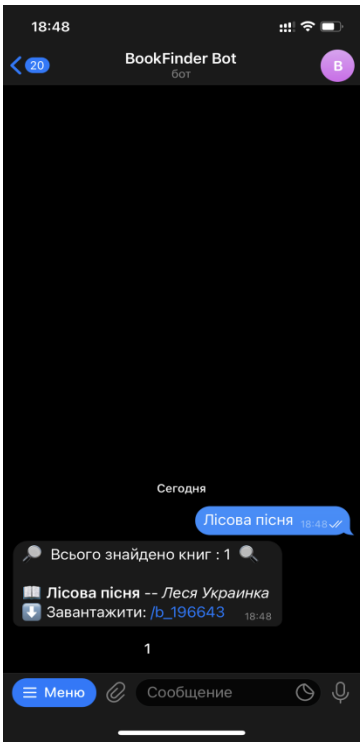

Рисунок 3.4 – Повідомлення без команди

Після того, як необхідна книга знайдена, користувач вводить команду і запускає процес скачування (рис. 3.5).

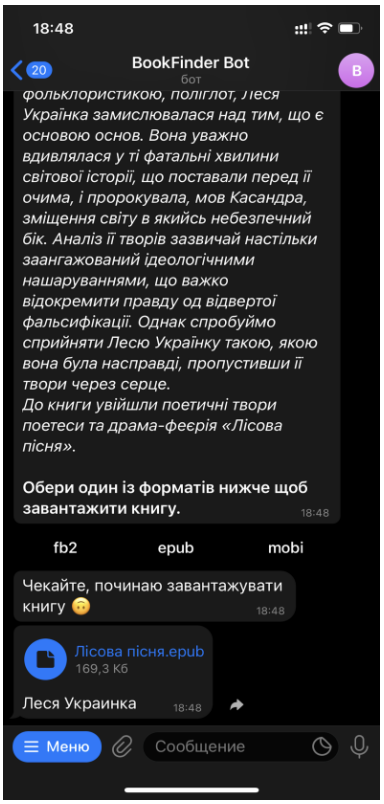

Рисунок 3.5 – Скачування книги

За замовчуванням стоять формати (fb2/epub/mobi) якщо їх немає, надає доступні формати (підтримуються всі доступні формати).

Встановлено тротлінг (throttling.py) (Додаток В), щоб користувачі не надсилали спам до боту. Також встановлені обмеження, щоб не було інших ботів у групі (antispam.py) (Додаток В).

База книг величезна, але на жаль не всі книги доступні для скачування, через деякі обмеження, цей бот дозволяє їх обійти, наприклад процес реєстрації на сайті. Функція async def get_tempfile(url) потрібна для того, щоб скачувати тимчасовий файл, так як за допомогою методів aiogram неможливо скачувати деякі файли, тому потрібна авторизація на сайті (general.py). Авторизация на сайті проходить за допомогою функції async def get(url) (Додаток В).

Дані, отримані від користувачів записуються до таблиць SQL (db_commands.py) (Додаток В). З них адміністратор отримує статистичні дані.

3.3 Структура Телеграм-боту

Програма Асинхронний Телеграм-бот для пошуку з пошуку та скачування книг включає в себе наступні папки:

- filters – містить файли з фільтрами;
- handlers – містить файли обробки запитів;
- keyboards – містить файли обробки інлайн команд;
- middlewares – містить файли антиспаму та тротлінгу;
- screenshots – містить скріншоти програми;
- utils – містить файли бази даних, генерації сторінок, парсингу, набору команд бота тощо.

Файлова структура представлена на рис. 3.6.

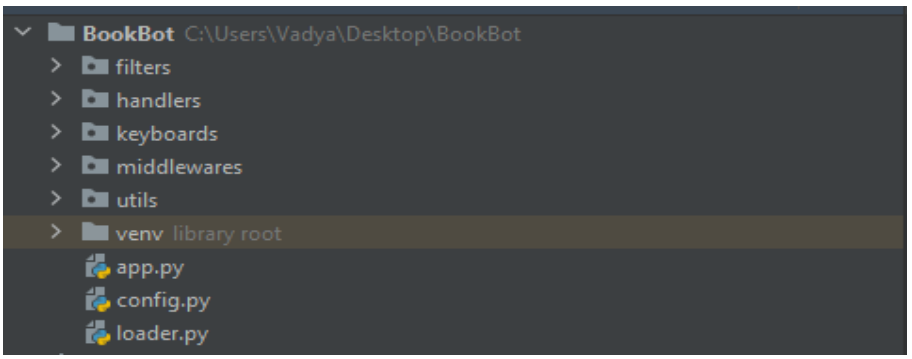

Рисунок 3.6 – Файлова структура Телеграм-боту

База даних складається з наступних таблиць:

- user – таблиця для зберігання користувачів (табл. 3.1);
- books – таблиця для зберігання рейтингу завантажених книг (табл. 3.2);
- authors – таблиця для для зберігання рейтингу авторів завантажених книг (табл. 3.3);

- book_pages – таблиця для зберігання сторінок із результатами запитів та назвами запитів (табл. 3.4);
- author_books_pages – книги за обраним автором (табл. 3.5);
- series_books_pages – книги за обраною серією (табл. 3.6);
- author_pages – таблиця для зберігання сторінок з авторами (табл. 3.7);
- series_pages – таблиця для зберігання сторінок з книжковими серіями за запитом (табл. 3.8).

Таблиця 3.1

Таблиця user

Назва поля	Тип даних
user_id	serial primary key
full_name	varchar(255)
telegram_id	bigint

Таблиця 3.2

Таблиця books

Назва поля	Тип даних
book_id	serial primary key
book_name	varchar(255)
link	varchar(255)
downloaded	bigint

Таблиця 3.3

Таблиця authors

Назва поля	Тип даних
author_id	serial primary key
author_name	varchar(255)
link	varchar(255)
queries	bigint

Таблиця 3.4

Таблиця book_pages

Назва поля	Тип даних
pages_id	serial primary key
request_name	varchar(255)
pages	text

Таблиця 3.5

Таблиця author_books_pages

Назва поля	Тип даних
pages_id	serial primary key
request_name	varchar(255)
author_name	varchar(255)
count_books	int
pages	text

Таблиця 3.6

Таблиця series_books_pages

Назва поля	Тип даних
pages_id	serial primary key
request_name	varchar(255)
series_name	varchar(255)
series_author	varchar(255)
series_genres	varchar(255)
pages	text

Таблиця 3.7

Таблиця author_pages

Назва поля	Тип даних
pages_id	serial primary key
request_name	varchar(255)
pages	text

Таблиця 3.8

Таблиця series_pages

Назва поля	Тип даних
pages_id	serial primary key
request_name	varchar(255)
pages	text

З даних таблиць бази даних адміністратор отримує статистику за запитами користувачів.

ВИСНОВКИ

Чат-боти – один із багатьох перспективних напрямів інформаційних технологій. Вони були задумані як новий інтерфейс, покликаний замінити або доповнити програми або відвідування веб-сайту, дозволяючи користувачам просто взаємодіяти зі службою через чат.

В результаті проведення дослідження:

- виконано загальний огляд чат-ботів для месенджерів;
- проаналізовано використання Телеграм-ботів як ефективного засобу отримання інформації;
- розроблено програмний продукт асинхронний Телеграм-бот, який дозволяє за запитом користувача пошук та скачування книг.

Розробка асинхронного Телеграм-боту з пошуку та скачування книг була виконана за допомогою мови програмування Python, асинхронної бібліотеки aiogram 2.14, мови структурованих запитів SQL.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бас Р. В. Розробка telegram бота з автоматизацією його функцій : дипломна робота магістра за спеціальністю „126 — інформаційні системи та технології“ / Р. В. Бас. — Тернопіль : ТНТУ, 2020. — 68 с.

2. Бондаренко О. О. Технологія використання чат-ботів для забезпечення якості дистанційного навчання / О. О. Бондаренко. – Дніпро. [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: http://elibrary.kdpu.edu.ua/xmlui/handle/123456789/6058.

3. Голуб Т.В, Наумик В. В Технології та іновації сучасного світу: Навчальний посібник /. Запоріжжя: Національний університет «Запорізька політехніка», 2018. – 240 с.

4. Даниленко Дарина. Як чат-боти стають новими медіами? [Електронний ресурс] // Інтернет свобода. – 2020. – Режим доступу до ресурсу: https://netfreedom.org.ua/article/column-yak-chatboti-stayut-novimi-media-i-chomu-zhurnalistam-ye-sens-iz-nimi-rozibratisya?fbclid=IwAR0JUl4eWw_gkMs0bQ8MlGAyux76X8e1g9GGAMBPaGYvrxXrMbnCxt30S6A.

5. Комса Карина. 12 Telegram-каналів для тих, хто любить читати, але не завжди має на це час. [Електронний ресурс] // ТОВ «Фьючер Медіа». – 2017. – Режим доступу до ресурсу: https://mind.ua/publications/20177809-12-telegram-kanaliv-dlya-tih-hto-lyubit-chitati-ale-ne-zavzhdi-mae-na-ce-chas.

6. Кіншаков Е. В. Інформаційний чат-бот для соціальної мережі Telegram з використанням Google Assistant та Google Search API [Електронний ресурс] / Е. В. Кіншаков // Сумський державний університет. – 2019. – Режим доступу до ресурсу: http://essuir.sumdu.edu.ua/handle/123456789/73577.

7. Коляструк, Б., і Ю. Антонов. «ОСОБЛИВОСТІ РОЗРОБКИ TELEGRAM БОТА ДЛЯ ІНФОРМАЦІЙНОЇ ПІДТРИМКИ НАВЧАЛЬНОГО ПРОЦЕСУ». Матеріали конференцій МЦНД, Жовтень 2020, с. 67-69.

8. Abdul-Kader, S., & Woods, J. (2015). Survey on Chatbot Design Techniques in Speech Conversation Systems. International Journal of Advanced Computer Science and Applications, 6(7).

9. Aiogram documentation. [Електронний ресурс]. – Режим доступу: https://docs.aiogram.dev/en/latest/.

10. Craig Dickson. How to Create and Manipulate SQL Databases with Python. – 2020. – Режим доступу до ресурсу: https://www.freecodecamp.org/news/connect-python-with-sql/.

11. Dale R. The Return of the Chatbots [Електронний ресурс] / Dale // Natural Language Engineering. – 2016. – Режим доступу до ресурсу: https://doi.org/10.1017/S1351324916000243.

12. Facebook messenger. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.facebook.com/messenger/.

13. Hung, V., Elvir, M., Gonzalez, A., & Demara, R. (2009). Towards a method for evaluating naturalness in conversational dialog systems. 2009 IEEE International Conference on Systems, Man and Cybernetics.

14. Python. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.python.org/.

15. Shevat A. Designing Bots: Creating Conversational Experiences. Sebastopol [Електронний ресурс]. – 2018. – Режим доступу: https://www.accenture.com/t20180509T102140Z__w__/us-en/_acnmedia/PDF77/AccentureResearch-Conversational-AI-Platforms.pdf.

16. Telegram. [Електронний ресурс]. – Режим доступу до ресурсу: https://web.telegram.org/.

17. Telegram APIs. [Електроннийресурс]. – Режим доступу: https://core.telegram.org/.

18. Twitter. [Електронний ресурс]. – Режим доступу до ресурсу: https://twitter.com/.

19. Usman Malik. Introduction to Python SQL Libraries. [Електронний ресурс]. – Режим доступу: https://realpython.com/python-sql-libraries/.

20. Veglis A. Chatbots on the Rise: A New Narrative in Journalism [Електронний ресурс] / A. Veglis, T. Maniou // Redfame Publishing. – 2019. – Режим доступу до ресурсу: http://redfame.com/journal/index.php/smc/article/view/3986.

21. Vitaly K. Chatbots: a Worthy Replacement for Apps or Temporary Obsession? [Електронний ресурс]. Cleveroad. – 2016. – Режим доступу: https://www.cleveroad.com/blog/chatbots-a-worthy-replacement-forapps-or-temporary-obsession.

22. WhatsApp. [Електронний ресурс]. – Режим доступу до ресурсу: https://web.whatsapp.com/.

ДОДАТОК А

Інструкція користувача

- повідомлення без команди – видасть доступні варіанти для вибору з інлайн кнопками (письменники, книги або серії);
- /author ім'я автора – пошук за ПІБ автора;
- /series назва книжкової серії – пошук за назвою серії;
- /help – коротка довідка;
- /rating_a – рейтинг за авторами (кількість запитів);
- /rating_b – рейтинг по книгах (у завантажених книг);
- /create_group – інструкція зі створення особистої групи та додаванням бота до панелі адміністратора;
- /report – поскаржитися на спам/користувача в групі (повідомлення надсилається адміністратору).

ДОДАТОК Б

Інструкція адміністратора

- /rating_book – загальна кількість завантажених книг;
- /rating_user – загальна кількість користувачів, які запускали бот;
- /log-file – отримати файл із логгами;
- /rating_u – топ 10 користувачів за активністю.

ДОДАТОК В

Лістинг

Файл general.py

```
import tempfile

import aiohttp
import fake_useragent
from bs4 import BeautifulSoup

import config

user = fake_useragent.UserAgent().random
url_to_register = 'http://flibusta.is/user'
headers = {'user-agent': user}
data = {
    'name': config.CITE_LOGIN,
    'pass': config.CITE_PASS,
    'form_id': 'user_login',
}


async def get(url):
    async with aiohttp.ClientSession() as session:
         async with session.post(url_to_register, headers=headers, data=data):
# Авторизация на сайте
            async with session.get(url, headers=headers) as response:
                res = await response.text()
                soup = BeautifulSoup(res, 'lxml')
                return soup


async def get_tempfile(url):
    async with aiohttp.ClientSession() as session:
        async with session.post(url_to_register, headers=headers, data=data):
            async with session.get(url, headers=headers) as response:
                if response.status == 200:
                    fp = tempfile.TemporaryFile()
                    fp.write(await response.read())
                    fp.seek(0)
                    return fp
```

```
async def get_without_register(url):
    async with aiohttp.ClientSession() as session:
        async with session.get(url, headers=headers) as response:
            res = await response.text()
            soup = BeautifulSoup(res, 'lxml')
            return soup
```

Файл antispam.py

```
from aiogram import types
from aiogram.dispatcher.middlewares import BaseMiddleware


class CheckTypeMessage(BaseMiddleware):

    async def on_pre_process_update(self, update: types.Update, data: dict):
        if update.message:
            if not update.message.text:
                await update.message.delete()
                return
```

Файл throttling.py

```
import asyncio
from typing import Union

from aiogram import types, Dispatcher
from aiogram.dispatcher import DEFAULT_RATE_LIMIT
from aiogram.dispatcher.handler import CancelHandler, current_handler
from aiogram.dispatcher.middlewares import BaseMiddleware
from aiogram.utils.exceptions import Throttled


class ThrottlingMiddleware(BaseMiddleware):

    def __init__(self, limit=DEFAULT_RATE_LIMIT, key_prefix='antiflood_'):
        self.rate_limit = limit
        self.prefix = key_prefix
```

```
            super(ThrottlingMiddleware, self).__init__()

                async def throttle(self, target: Union[types.Message,
types.CallbackQuery]):
            handler = current_handler.get()
            dispatcher = Dispatcher.get_current()
            if not handler:
                return
            limit = getattr(handler, 'throttling_rate_limit', self.rate_limit)
                            key = getattr(handler, 'throttling_key',
f"{self.prefix}_{handler.__name__}")

            try:
                await dispatcher.throttle(key, rate=limit)
            except Throttled as t:
                await self.target_throttled(target, t, dispatcher, key)
                raise CancelHandler()

        @staticmethod
                async def target_throttled(target: Union[types.Message,
types.CallbackQuery],
                                           throttled: Throttled, dispatcher: Dispatcher,
key: str):
             msg = target.message if isinstance(target, types.CallbackQuery) else
target
            delta = throttled.rate - throttled.delta

            await asyncio.sleep(delta)

            if throttled.exceeded_count == 2 and key == 'downloading':
                   await msg.reply('⚠ Занадто часті команди! Давай не так швидко
⚠\n'
                                   '<b>Я вже почав завантажувати для тебе книгу</b>
👐\n')
                return
            elif throttled.exceeded_count == 2 :
                await msg.reply('Занадто часто! Давай не так швидко')
                return
            elif throttled.exceeded_count == 3:
                    await msg.reply(f'⚠ Зачекай ще{round(delta, 3)} секунд перед
наступним запитом!')
                return
```

```
    async def on_process_message(self, message, data):
        await self.throttle(message)

    async def on_process_callback_query(self, call, data):
        await self.throttle(call)
```

Файл db_commands.py

```
from typing import Union

import asyncpg
from asyncpg import Connection
from asyncpg.pool import Pool

import config


class Database:
    def __init__(self):
        self.pool: Union[Pool, None] = None

    async def create(self):
        self.pool = await asyncpg.create_pool(
            user=config.DB_USER,
            password=config.DB_PASS,
            host=config.DB_HOST,
            database=config.DB_NAME
        )

    async def execute(self, command, *args,
                      fetch: bool = False,
                      fetchval: bool = False,
                      fetchrow: bool = False,
                      execute: bool = False):
        async with self.pool.acquire() as connection:
            connection: Connection
            async with connection.transaction():
                if fetch:
```

```
                        result = await connection.fetch(command, *args)  # - все
записи списком
                    elif fetchval:
                        result = await connection.fetchval(command, *args)  # -
одна запись
                    elif fetchrow:
                        result = await connection.fetchrow(command, *args)  # -
первая строка
                    elif execute:
                        result = await connection.execute(command, *args)
                    return result

        async def create_tables(self):

            users = '''
            CREATE TABLE IF NOT EXISTS users (
            user_id SERIAL PRIMARY KEY,
            full_name VARCHAR(255) NOT NULL,
            telegram_id BIGINT NOT NULL UNIQUE
            )
            '''
            books = '''
            CREATE TABLE IF NOT EXISTS books (
            book_id SERIAL PRIMARY KEY,
            book_name VARCHAR(255) NOT NULL,
            link VARCHAR(255) NOT NULL UNIQUE,
            downloaded BIGINT NOT NULL,
            author VARCHAR(255),
            formats VARCHAR(255),
            description TEXT
            )
            '''
            authors = '''
            CREATE TABLE IF NOT EXISTS authors (
            author_id SERIAL PRIMARY KEY,
            author_name VARCHAR(255) NOT NULL,
            link VARCHAR(255) NOT NULL UNIQUE,
            queries BIGINT NOT NULL
            )
            '''
            book_pages = '''
            CREATE TABLE IF NOT EXISTS book_pages (
```

```
pages_id SERIAL PRIMARY KEY,
request_name VARCHAR(255) NOT NULL UNIQUE,
pages text[]
)
'''
author_pages = '''
CREATE TABLE IF NOT EXISTS author_pages (
pages_id SERIAL PRIMARY KEY,
request_name VARCHAR(255) NOT NULL UNIQUE,
pages text[]
)
'''
series_pages = '''
CREATE TABLE IF NOT EXISTS series_pages (
pages_id SERIAL PRIMARY KEY,
request_name VARCHAR(255) NOT NULL UNIQUE,
pages text[]
)
'''
author_books_pages = '''
CREATE TABLE IF NOT EXISTS author_book_pages (
pages_id SERIAL PRIMARY KEY,
request_name VARCHAR(255) NOT NULL UNIQUE,
author_name VARCHAR(255),
count_books INT,
pages text[]
)
'''
series_books_pages = '''
CREATE TABLE IF NOT EXISTS series_book_pages (
pages_id SERIAL PRIMARY KEY,
request_name VARCHAR(255) NOT NULL UNIQUE,
series_name VARCHAR(255),
series_author VARCHAR(255),
series_genres VARCHAR(255),
pages text[]
)
'''

book_formats = '''
CREATE TABLE IF NOT EXISTS book_formats (
format_id  SERIAL PRIMARY KEY,
```

```
            book_id  INT NOT NULL UNIQUE,
            fb2 VARCHAR(255),
            epub VARCHAR(255),
            mobi VARCHAR(255),
            download VARCHAR(255),
            FOREIGN KEY(book_id) REFERENCES books(book_id)
            )
            '''
                list_tables = [users, books, authors, book_pages, author_pages,
series_pages,
                               author_books_pages, series_books_pages, book_formats]

            for table in list_tables:
                await self.execute(table, execute=True)

        async def add_user(self, user: str, telegram_id: int):
              sql = f"INSERT INTO users(full_name, telegram_id) VALUES ('{user}',
{telegram_id}) ON CONFLICT  DO NOTHING"
            await self.execute(sql, execute=True)


         async def insert_book(self, book: str, link: str, author: str, formats:
str, description: str):
            sql = f'''
                   INSERT INTO books(book_name, link, downloaded, author, formats,
description)
                          VALUES('{book}', '{link}', 1, '{author}', '{formats}',
'{description}')
                ON CONFLICT (link) DO UPDATE SET
                author = '{author}',
                formats = '{formats}',
                description = '{description}'
                '''
            await self.execute(sql, execute=True)


        async def select_file_id(self, link, format):
            sql = f'''
                SELECT {format} FROM book_formats
                JOIN books USING(book_id)
                WHERE link = '{link}'
            '''
```

```
            return await self.execute(sql, fetchval=True)

        async def insert_file_id(self, link, format, file_id):
            sql = f'''
                INSERT INTO book_formats (book_id, {format}) VALUES
                ((SELECT book_id FROM books WHERE link = '{link}'), '{file_id}')
                ON CONFLICT (book_id) DO UPDATE
                SET {format} = '{file_id}'
            '''
            return await self.execute(sql, fetchval=True)


        async def update_count_downloaded(self, link: str):
            sql = f'''
                UPDATE books SET
                 downloaded = 1 + downloaded
                WHERE link = '{link}'
            '''
            await self.execute(sql, execute=True)


        async def select_book(self, link):
            sql = f'''
                   SELECT book_name, author, formats, description FROM BOOKS WHERE
link = '{link}'
                     '''
            res = await self.execute(sql, fetchrow=True)
            return res if res else None


        async def rating_author(self, author: str, link: str):
                sql = f'''INSERT INTO authors(author_name, link, queries) VALUES
('{author}', '{link}', 1)
                        ON CONFLICT (link) DO UPDATE
                          SET queries = 1 + (SELECT queries from authors where link
= '{link}')'''
            await self.execute(sql, execute=True)

        async def select_count_values(self, table_name):
                count = await self.execute(f'SELECT count(*) FROM {table_name}',
fetchval=True)
            return count
```

```
        async def rating_top_10_values(self, table):
            query = 'downloaded' if table == 'book' else 'queries'
              top = await self.execute(f'SELECT * FROM {table}s ORDER BY {query}
DESC LIMIT 10', fetch=True)
            top_dict = {}
            for elem in top:
                link = elem.get('link')
                link = link[1:].replace('/', '_', 1)
                top_dict[link] = elem.get(f'{table}_name')

            return top_dict

        async def delete_table_pages(self):
                  await self.execute(f'DROP TABLE book_pages, author_book_pages,
series_pages', execute=True)


        async def add_new_pages(self, table_name, items, request_name):
            sql = f"""
            INSERT INTO {table_name}(request_name, pages)
            VALUES ('{request_name}', ARRAY[{items}]) ON CONFLICT  DO NOTHING
            """

            await self.execute(sql, execute=True)

              async def add_new_author_book_pages(self, items, request_name,
count_books, author):
                sql = f"""INSERT INTO author_book_pages(request_name, author_name,
count_books, pages)
                              VALUES ('{request_name}', '{author}', {count_books},
ARRAY[{items}])
                         ON CONFLICT DO NOTHING"""
            await self.execute(sql, execute=True)

              async def add_new_series_book_pages(self, items, request_name,
series_name, series_author, series_genres):
                sql = f"""INSERT INTO series_book_pages(request_name, series_name,
series_author, series_genres, pages)
                                       VALUES ('{request_name}', '{series_name}',
'{series_author[:250]}', '{series_genres}', ARRAY[{items}])
                         ON CONFLICT  DO NOTHING"""
```

```
            await self.execute(sql, execute=True)

        async def select_pages(self, request_name, table_name, *args):
            if args:
                  sql = f"SELECT request_name, {', '.join(args)} FROM {table_name}
WHERE request_name = '{request_name}'"
            else:
                    sql = f"SELECT request_name, pages FROM {table_name} WHERE
request_name = '{request_name}'"

            res = await self.execute(sql, fetch=True)
            if not res: return

            return self.get_args(res[0], table_name)

           async def update_author_pages(self, pages: list, request_name: str,
count_books: int):
            sql = f"""
                    UPDATE author_book_pages SET pages = ARRAY[{pages}],
                               count_books = {count_books} WHERE request_name =
'{request_name}'
                    """
            await self.execute(sql, execute=True)

            async def update_book_pages(self, request_name, pages, table_name,
column='pages'):
                sql = f"UPDATE {table_name} SET {column} = ARRAY[{pages}] WHERE
request_name='{request_name}'"
            await self.execute(sql, execute=True)

        @staticmethod
        def get_args(result, table_name):
            name = result.get('request_name')
              pages_lst = [list(map(lambda x: x.replace('\\n', '\n'), elem)) for
elem in result.get('pages')[0]]

            if table_name == 'author_book_pages':
                author_name = result.get('author_name')
                count_books = result.get('count_books')
                return name, pages_lst, author_name, count_books
            elif table_name == 'series_book_pages':
```

```
                                        series_info   =   [result.get('series_name'),
result.get('series_author'), result.get('series_genres')]
                    return name, pages_lst, series_info
                else:
                    return name, pages_lst
```