Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Бакалаврська дипломна робота

на тему: Розробка Telegram-боту для завантаження TikTok і Youtube відео з використанням PyTube, Aiogram, RapidApi

Виконав: студент 4 курсу
групи 4ПІ-20Б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Банарь Дмитро Андрійович
(прізвище та ініціали)
Керівник: к.т.н., доц. каф. ПЗ Черноволик Г.О.
(прізвище та ініціали)
Рецензент: к.т.н., доц. каф. ОТ Томчук М.А.
(прізвище та ініціали)

Допущено до захисту

Зав. кафедри ПЗ _____
«10» червня 2024 р.

ВНТУ – 2024

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
О. Н. Романюк
«12 » березня 2024 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Банарю Дмитру Андрійовичу

1. Тема роботи – розробка Telegram-боту для завантаження TikTok і Youtube відео з використанням PyTube, Aiogram, RapidApi.

Керівник роботи: Черноволик Галина Олександрівна, к.т.н., доц. кафедри ПЗ, затверджені наказом вищого навчального закладу від 11 березня 2024 р. №80.

2. Строк подання студентом роботи 3 червня 2024.

3. Вихідні дані до роботи: базові алгоритми для обробки даних – алгоритм асинхронної обробка мережевих HTTP-запитів, алгоритм парсингу тексту, алгоритм аналізу проведених конвертацій; використані бібліотеки – aiogram, pytube, moviepy, shutil, requests, sqlite; середовище розробки – PyCharm; мова програмування – Python; API, що використані у ході розробки – RapidAPI; мінімальний рівень API – Python 3.12.0; вихідні дані – відеоряд, з обраної платформи без нанесених водяних знаків.

4. Зміст текстової частини: вступ; дослідження розвитку Telegram-ботів та формулювання завдань для аналізу; аналіз стану Telegram-ботів та їх актуальність; порівняльний аналіз аналогів; аналіз методів розв'язання задачі; постановка задач бакалаврської дипломної роботи; розробка структури та алгоритмів програмного продукту; визначення інструментів та аналіз вхідних даних системи; розробка інтерфейсу програмного застосунку; розробка моделі системи; розробка алгоритмів роботи системи; розробка програмних модулів реалізації Telegram-боту для завантаження TikTok і YouTube відео без водяного знаку; варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення; розробка модуля для завантаження відео з обраних платформ; розробка методів для взаємодії користувача з базою даних; тестування реалізованої системи; розробка інструкції користувача; список

використаних джерел; висновки.

5. Перелік ілюстративного матеріалу: титульний слайд, актуальність
мета, об'єкт та предмет дослідження, завдання на бакалаврську дипломну р
та практична цінність роботи, наукова новизна отриманих результат
практична цінність, порівняння аналогів (ч. 1-3), засоби для реалізації за
метод обробки відео-ресурсів (ч. 1-3), метод пошуку відео, блок-схема м
для обробки відео-ресурсів, UML-діаграма діяльності Telegram-боту, зага
алгоритм роботи програми та алгоритм для запису користувача в базу д
алгоритм визначення типу посилання, блок-схема функції для завантаж
відео з платформи TikTok, блок-схема функції додавання користувача у
даних, блок-схема функції створення таблиці з операціями, функц
розробленого боту (ч. 1-6), висновки (ч. 1-2), апробація та публікація резул
бакалаврської дипломної роботи.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завданн прийня
1-4	Черноволик Г.О., к.н.т., доцент кафедри ПЗ	13.03.24	03.06.24

7. Дата видачі завдання 13 березня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Приміт
1	Аналіз стану Telegram-ботів та їх актуальності	14.03.24 – 19.03.24	вик
2	Аналіз аналогів та проблематики розроблюваного застосунку	20.03.24 – 25.03.24	вик
3	Розробка модуля для взаємодії користувача з базою даних	26.03.24 – 10.05.24	вик
4	Розробка модуля для завантаження відеоряду з обраної платформи	11.05.24 – 23.05.24	вик
5	Оформлення матеріалів до захисту БДР	23.05.24 – 30.05.24	вик

Студент _____ **Д.А. Банар**
(підпис) (ініціали та пр

Керівник бакалаврської дипломної роботи _____ **Г. О. Чернов**
(підпис) (ініціали та пр

АНОТАЦІЯ

Бакалаврська дипломна робота складається з 113 сторінок формату А4, на яких є 42 рисунки, 4 таблиці, список використаної літератури містить 21 найменування.

Бакалаврська дипломна робота спрямована на розробку Telegram-бота, який дозволяє завантажувати відео з TikTok та YouTube без водяного знаку. Робота включає розробку бази даних, інтерфейсу взаємодії з користувачем та функціонал для керування викликами функцій бота.

Проаналізовано стан Telegram-ботів та обґрунтовано актуальність та інноваційність системи.

Реалізовано систему для обробки посилань на відео-ресурси та отримання відеорядів без водяного знаку.

Процес розробки включав аналіз вимог, вибір бібліотек та тестування. Використано Python та SQLite для роботи з базою даних, Aiogram для інтеграції з Telegram.

Ключові слова: бот, Telegram, актуальність, TikTok, Youtube, відеоряд, python, aiogram, юзабіліті, зручність.

ABSTRACT

The bachelor's thesis consists of 113 A4 pages, including 42 figures, 4 tables, and a reference list containing 21 sources.

The bachelor's thesis aims to develop a Telegram bot enabling video downloads from TikTok and YouTube without watermarks. The work encompasses database development, user interface design, and functional management of bot features.

A system is implemented for processing video resource links and obtaining watermark-free video sequences.

The development process includes requirement analysis, library selection, and testing.

Python and SQLite are utilized for database operations, while Aiogram facilitates integration with Telegram.

Keywords: bot, Telegram, relevance, TikTok, YouTube, video sequence, Python, Aiogram, usability, convenience.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Чат-боти стають популярні, як інструмент для автоматизації обслуговування клієнтів, що дозволяє швидко та ефективно відповідати на їх запитання та запити. Також вони можуть автоматично збирати відгуки від клієнтів після покупок чи надання послуг [1].

Завантаження відео з популярних платформ TikTok та YouTube без водяного знаку в останні роки стало популярною практикою. Завдяки зростанню інтересу до цих платформ та широкому поширенню мобільних пристроїв, користувачі все більше цікавляться можливістю отримати відео без додаткових елементів. У цьому контексті, розробка системи, яка б дозволяла зручно та ефективно завантажувати відео без водяного знаку, є актуальною та перспективною.

TikTok – це платформа, де користувачі можуть створювати та обмінюватися короткими відеокліпами. Цей додаток для соціальних мереж є дуже популярним у всьому світі завдяки великій кількості підписників [2].

YouTube – це сервіс для завантаження та обміну відео. Люди створюють відеоматеріали та діляться ними, отримуючи значну кількість переглядів [3].

Реалізація проекту включає в себе обробку посилань на відео-ресурси та отримання відео без водяного знаку відповідно до обраного сервісу завантаження. Процес розробки складається з детального аналізу вимог, обгрунтування та вибору відповідних бібліотек для роботи з базою даних та Telegram API. Результатом проєкту є функціональний Telegram-бот, який здатний швидко та зручно завантажувати відео з TikTok та YouTube без водяного знаку.

Ця робота не лише спростить процес завантаження відео для користувачів, але й надасть можливість досліджувати і використовувати різноманітні техніки роботи з базами даних та розробки ботів для месенджеру Telegram. Крім того, враховуючи актуальність теми, результати проєкту можуть бути корисні для

широкого кола користувачів, які зацікавлені в отриманні відео без додаткових маркерів.

Багато існуючих аналогів для завантаження відео на TikTok або YouTube не мають всіх потрібних функцій, і часто вони не інтегровані між собою. Конкурентноздатні аналоги часто вимагають плати за повний функціонал, що не завжди влаштовує користувачів. З популярністю TikTok і YouTube росте і попит на подібні сервіси, що робить актуальною розробку власного Telegram-боту з повним функціоналом.

Зв’язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Мета бакалаврської дипломної роботи полягає у підвищенні якості завантаження відео-контенту з платформ TikTok та YouTube за рахунок розробки і використання спеціалізованого Telegram-бота з використанням PyTube, Aiogram, RapidApi, що дозволяє завантажувати відео без водяних знаків.

У бакалаврській дипломній роботі необхідно виконати такі завдання:

1) розробити методи роботи системи;
2) визначити загальний алгоритм системи;
3) розробити загальну структуру бази даних за допомогою SQLite3;
4) розробити функіонал для взаємодії користувача з Telegram-ботом;
5) реалізувати функціонал для адміністрування кількості викликів з боку користувача;
6) здійснити тестування загальної системи згідно поставлених задач.

Об'єктом дослідження є процес розробки функціоналу та інтерфейсу для Telegram-боту для завантаження TikTok та YouTube без водяного знаку.

Предметом дослідження є алгоритми і засоби реалізації основного функціоналу і бази даних.

Методи дослідження. У процесі досліджень використовувались методи: емпіричні методи для збору та аналізу даних з реальних випадків використання Telegram, TikTok, та YouTube для розробки та тестування функціоналу бота, визначення його ефективності та придатності, технології мережевого взаємодії для інтеграції бота з Telegram та взаємодії з іншими сервісами через API, основи програмування на Python для імплементації функціоналу бота та роботи з бібліотеками PyTube, Aiogram, RapidAPI.

Новизна отриманих результатів:

1. Подальшого розвитку отримав метод обробки відео-ресурсів у Telegram-боті, який, на відміну від існуючих, базується на використанні бібліотеки MoviePy, що дозволяє забезпечити швидке та ефективне кліпування відеорядів без накладання водяного знаку та збільшити продуктивність роботи бота.

2. Подальшого розвитку отримав метод пошуку відео, який, на відміну від існуючих, базується на використанні ефективних алгоритмів для формування HTTP-запитів та обробки відповідей від YouTube API та Rapid API, використовує вбудовані функції PyTube для зручної взаємодії з YouTube API та функції бібліотеки requests для Rapid API, що дозволяє більш повно використовувати можливості YouTube каналу.

Практична цінність бакалаврської дипломної роботи полягає в наданні якісного і безкоштовного інструменту для роботи з відео-ресурсами. Програмні модулі, розроблені в рамках бакалаврської дипломної роботи, можуть бути використані для створення подібних ботів для інших сервісів або функцій.

Особистий внесок здобувача. Усі наукові результати, викладені у бакалаврській дипломній роботі, отримані особисто автором. У друкованій праці [4], опублікованій у співавторстві, автору належить розробка алгоритму завантаження контенту з онлайн платформ.

Апробація матеріалів бакалаврської дипломної роботи. Результати бакалаврської дипломної роботи доповідалися на LIII Науково-технічній

конференції факультету інформаційних технологій та комп'ютерної інженерії (2024).

Публікації. За тематикою дослідження опубліковано 1 наукову працю у збірнику матеріалів LIII Науково-технічної конференції факультету інформаційних технологій та комп'ютерної інженерії (2024) [4].

У першому розділі бакалаврської дипломної роботи розглянуто аналоги для завантаження відео з популярних платформ, таких як TikTok та Youtube, а також порівняння з існуючими рішеннями. Визначені завдання проєкту, включаючи вимоги до бази даних та функціоналу боту.

Другий розділ присвячений розробці програмних модулів для Telegram-боту. Тут описані алгоритми та моделі, використані для створення боту, інструменти програмування та технології, що використовуються.

Третій розділ містить деталі реалізації бази даних для зберігання відео без водяних знаків, включаючи використані мови програмування та бібліотеки. Описано структуру бази даних, її таблиці та зв'язки між ними.

У четвертому розділі проведено тестування функціоналу боту та бази даних, що забезпечить коректну роботу системи. Також надано інструкцію користувача з використання Telegram-боту для завантаження відео з TikTok та Youtube без водяних знаків.

Аналіз. У пояснювальній записці до бакалаврської дипломної роботи було описано 4 розділи та використано 21 літературне джерело.

1 ДОСЛІДЖЕННЯ РОЗВИТКУ TELEGRAM-БОТІВ ТА ФОРМУЛЮВАННЯ ЗАВДАНЬ РОБОТИ

1.1 Аналіз стану Telegram-ботів та їх актуальність

У сучасному цифровому ландшафті Telegram-боти стають все більш важливим інструментом для різноманітних завдань та взаємодії з користувачами. Для бізнесу, освіти, розваг та інших галузей вони відкривають широкі можливості для автоматизації процесів, залучення аудиторії та створення інтерактивних сервісів. У цьому аналізі буде розглянуто стан розвитку Telegram-ботів, їхні функціональні можливості, актуальні тренди та перспективи використання.

Telegram – це впливовий та універсальний месенджер, що надає відмінні можливості для спілкування та організації робочих процесів. Цей сервіс поєднує швидкість, надійність, конфіденційність та різноманітність функцій, гарантуючи користувачам неперевершений досвід [5].

На сьогоднішній день Telegram налічує понад 500 мільйонів активних користувачів по всьому світу. Це платформа, що має широкий функціонал для створення ботів, що працюють у різних сферах діяльності. Кожен з них може мати свою унікальну функціональність, від відправлення коротких повідомлень до автоматизації оплати послуг чи навіть проведення конкурсів та голосувань.

Однією з головних переваг Telegram-ботів є їхні широкі функціональні можливості. Вони можуть бути програмовані для виконання різних завдань, таких як:

1) відправлення повідомлень за розкладом;
2) отримання та відправлення інформації з баз даних;
3) створення анкет та опитувань;
4) оповіщення користувачів про новини та події;
5) запуск голосувань та організація конкурсів;
6) онлайн-продаж товарів та послуг.

Крім того, Telegram постійно оновлює та розширює свої можливості для розробників. Нові API та інструменти дозволяють створювати більш складні та функціональні боти, забезпечуючи їхню актуальність та конкурентоспроможність.

З урахуванням стрімкого прогресу технологій та зростання користувацької бази Telegram, використання його ботів для бізнесу відкриває широкі перспективи. Вони можуть стати невід'ємним інструментом для підприємств будь-якого розміру, від невеликих стартапів до великих корпорацій. Боти також надають можливості для автоматизації та покращення взаємодії з клієнтами в різних секторах.

Чат-боти використовують штучний інтелект для спілкування з користувачами, що дозволяє їм надавати актуальний контент та відповіді на запитання [6]. Чат-боти можуть працювати на основі набору інструкцій або використовувати машинне навчання. Функціонал чат-бота, який працює на основі інструкцій, має свої обмеження. В основному такі боти призначені для відповіді на конкретні, фіксовані питання. Таким чином, коли користувач задає запитання, яке не відповідає певним шаблонам, бот може не мати відповіді.

На відміну від цього, чат-боти, які використовують машинне навчання, працюють більш гнучко. Вони можуть розуміти не тільки конкретні запитання, а й загальний контекст розмови. Тому користувач може поставити питання різними способами, і бот все одно зможе надати релевантну відповідь.

Проте, із зростанням популярності Telegram-ботів, відбувається і збільшення конкуренції. Для ефективного використання бота важливо продумано планувати його функціонал та взаємодію з користувачами.

Отже, аналізуючи Telegram-ботів бачимо, що вони стають потужним інструментом для взаємодії з аудиторією, автоматизації процесів та створення інтерактивних сервісів. Це важливий інструмент як для особистого, так і для бізнес-використання. Проте для досягнення успіху важливо ретельно розробити стратегію використання бота, враховуючи потреби та вимоги аудиторії.

1.2 Порівняльний аналіз аналогів

Популярність Telegram в Україні стабільно зростає, і це не в останню чергу завдяки широкому спектру телеграм-ботів різної тематики та напрямків. Наприклад, доступні боти, які допомагають користувачам вирішувати фінансові питання, завантажувати музику та інше [7].

Вони стають не лише корисним інструментом для бізнесу, розваг, навчання та інших сфер, але й невід'ємною частиною повсякденного інтернет-взаємодії. Боти забезпечують автоматизацію багатьох процесів та полегшують спілкування з користувачами.

Створення програмних засобів для Telegram-боту, який дозволяє завантажувати TikTok та Youtube відео без водяного знаку, може значно покращити користувацький досвід і принести численні переваги. Це не лише зробить процес споживання контенту більш зручним і приємним, а також спростить обробку того ж контенту. Серед найбільш відомих аналогів розрізняють наступні:

- TokMateBot;
- SaveFrom.net;
- Ssstik.io;
- Tmate;
- Live Photos.

Першим аналогом на розгляді виступає прямий конкурент у Telegram, а саме TokMateBot (див. рисунок. 1.1). Телеграм-бот для швидкого завантаження відео з TikTok без водяного знаку з підтримкою режиму Inline [8]. Бот також є розробленим саме для Telegram та створений на основі двох мов програмування, а саме Python та Shell.

Shell – це більше, ніж просто командний інтерпретатор для взаємодії користувача з операційною системою. Це потужна мова програмування з операторами умовного розгалуження, циклами, змінними та іншими

конструкціями, які дозволяють створювати скрипти та автоматизувати операції в системі [9].

Також це інструмент, який не просто виконує команди, а й допомагає створювати складні скрипти з умовами та повтореннями. Це дуже важливо для автоматизації рутинних операцій у системі

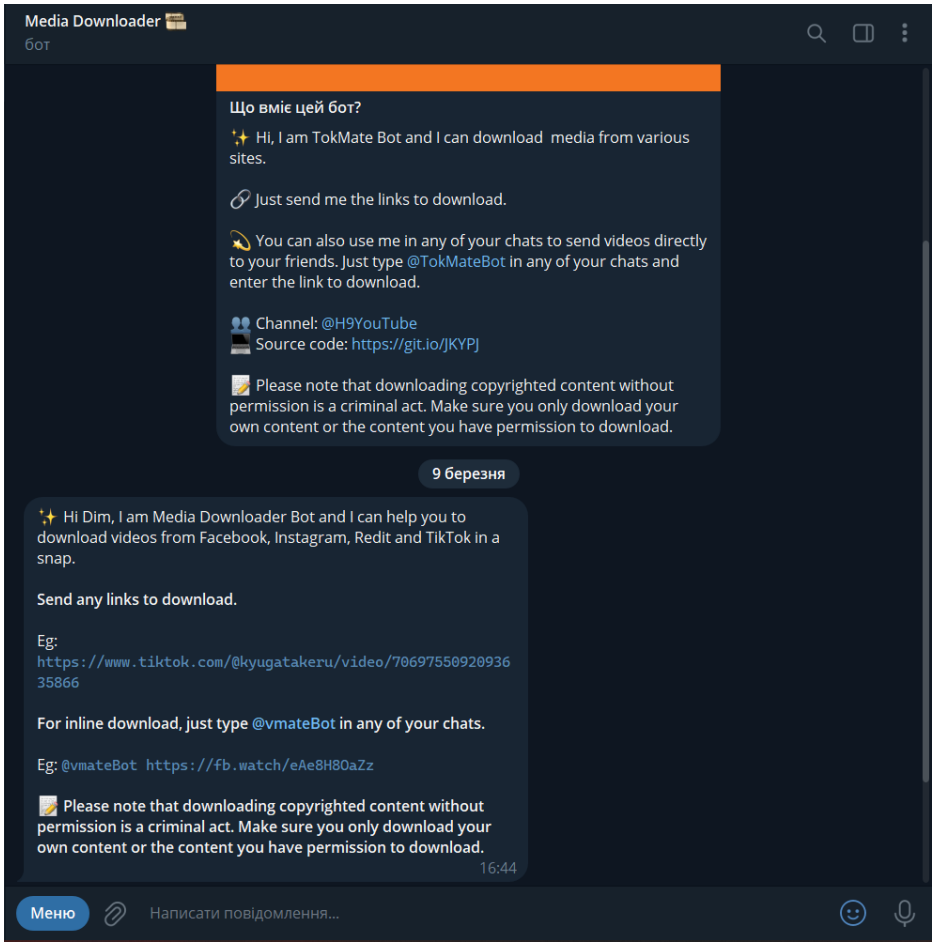


Рисунок 1.1 – Інтерфейс боту «TokMateBot»

Наступним аналогом розглянемо WEB застосунок «SaveFrom.net» (див. рисунок 1.2), онлайн-інструмент, який дозволяє швидко завантажувати відео та музичні файли. Вам не потрібно встановлювати додаткове програмне забезпечення або шукати інші альтернативи в мережі [10].

Наведений сервіс має дуже широкий функціонал і включає в себе завантаження відео та аудіоресурси з різних платформ, проте є у цього аналогу і

мінуси. Через його розміщеність у WEB-браузерах з нестабільним інтернетом у користувача навіть не буде можливості скористатися сервісом. Також значним мінусом можна вважати відсутності можливості завантаження TikTok відео без водяного знаку, що може вирішити розроблений в бакалаврській дипломній роботі Ttledram-бот.

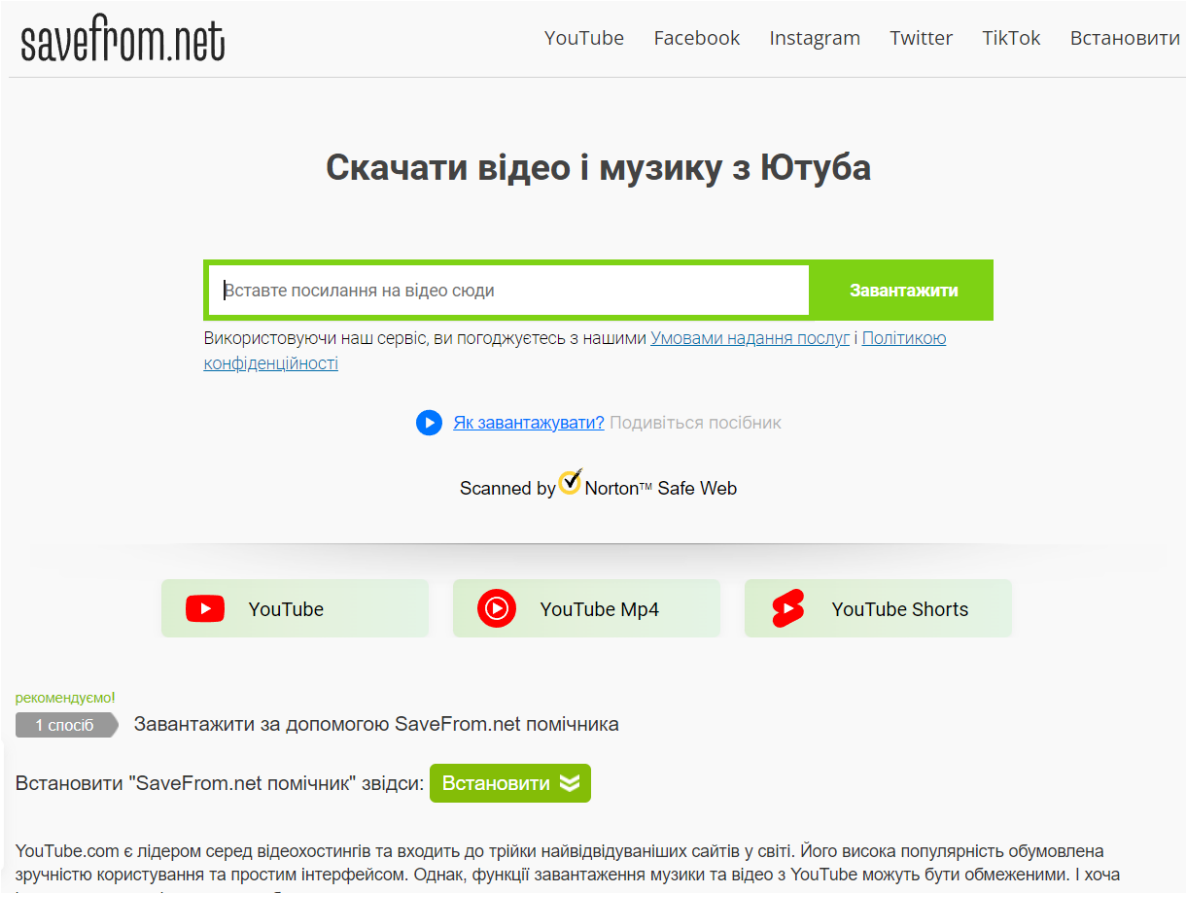


Рисунок 1.2 – Сторінка WEB-сервісу «SaveFrom.net»

Переходячи до наступного аналогу «Ssstik.io» (див. рисунок 1.3), слід зазначити, що цей аналог є також WEB-застосунком для завантаження відео лише з TikTok, який все ж таки може зберегти відео без водяного знаку [11].

Також цей сайт дозволяє обрати якість майбутнього відео у форматі MP4 та HD роздільною здатністю, не турбуючись про водяний знак. Для того, щоб дізнатися, як використовувати цей інструмент для видалення водяного знака TikTok, достатньо слідувати простим інструкціям на тому ж сайті.

І це я вважаю можна занести до мінусів, тобто недоліків тому що інтерфейс повинен відповідати нормам «юзабіліті» та на інтуітивному рівні повідомляти користувача куди потрібно натиснути для виконання того за чим користувач звернувся. Також вагомим недоліком можна вважати відсутність української локалізації, а також достатньо таки вузький функціонал, що дозволить відповісти лише на незначну частину запитів, які будуть пов'язані лише з Tik Tok.

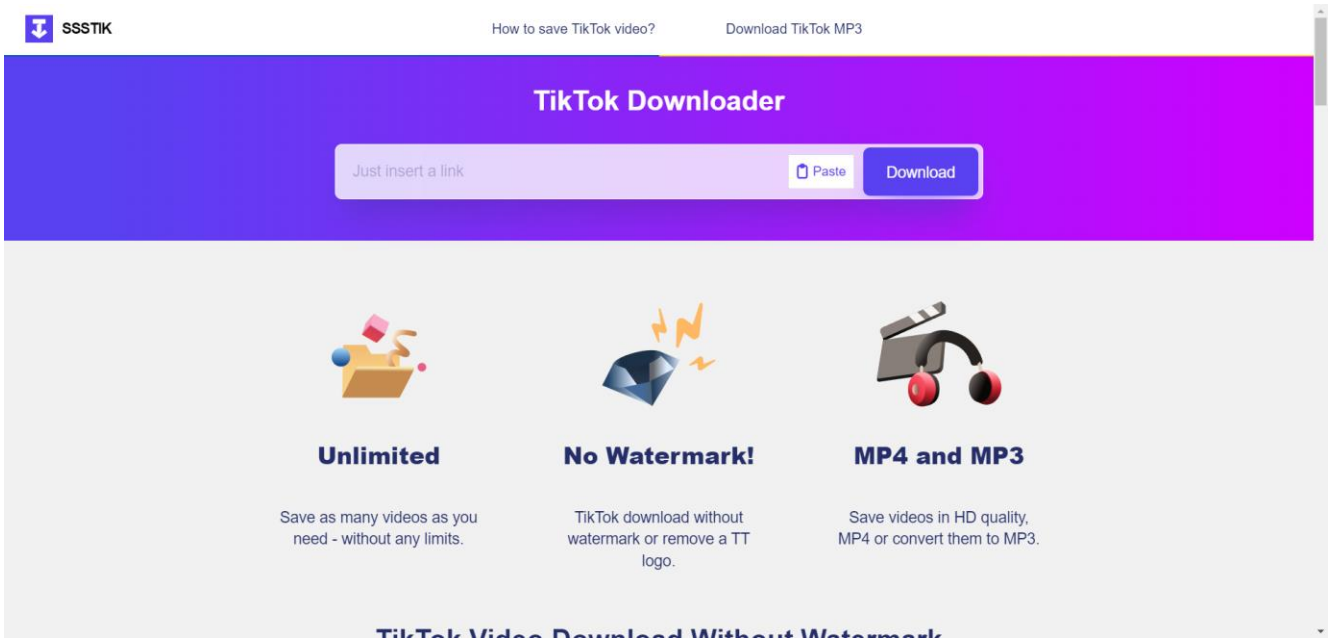


Рисунок 1.3 – Сторінка WEB-застосунку «Ssstik.io»

Tmate (див. рисунок 1.4) – це WEB-інструмент для завантаження відео без водяних знаків із TikTok використовуючи технологію PWA[12].

Цей сайт допомагає користувачеві отримати відео TikTok у найвищій якості у форматі mp4. Все, що потрібно зробити – вставити пряме посилання на ваше відео TikTok у текстове поле на нашому веб-сайті та натиснути кнопку "Завантажити".

Це зручний спосіб отримати відео з TikTok без водяних знаків, що дозволяє зберегти вміст у найкращій якості для подальшого використання.

PWA (Progressive Web Application) технологія дозволяє веб-сайтам працювати як нативні додатки на мобільних пристроях, що робить використання Tmate зручним і ефективним для користувачів.

Однією з ключових переваг PWA є можливість використання сервісного робочого працівника (Service Worker). Це скрипт, який працює в фоновому режимі і дозволяє PWA зберігати важливі дані локально на пристрої користувача. Це означає, що PWA може функціонувати навіть у випадку втрати зв'язку з Інтернетом, наприклад, можливість перегляду раніше завантажених сторінок або даних.

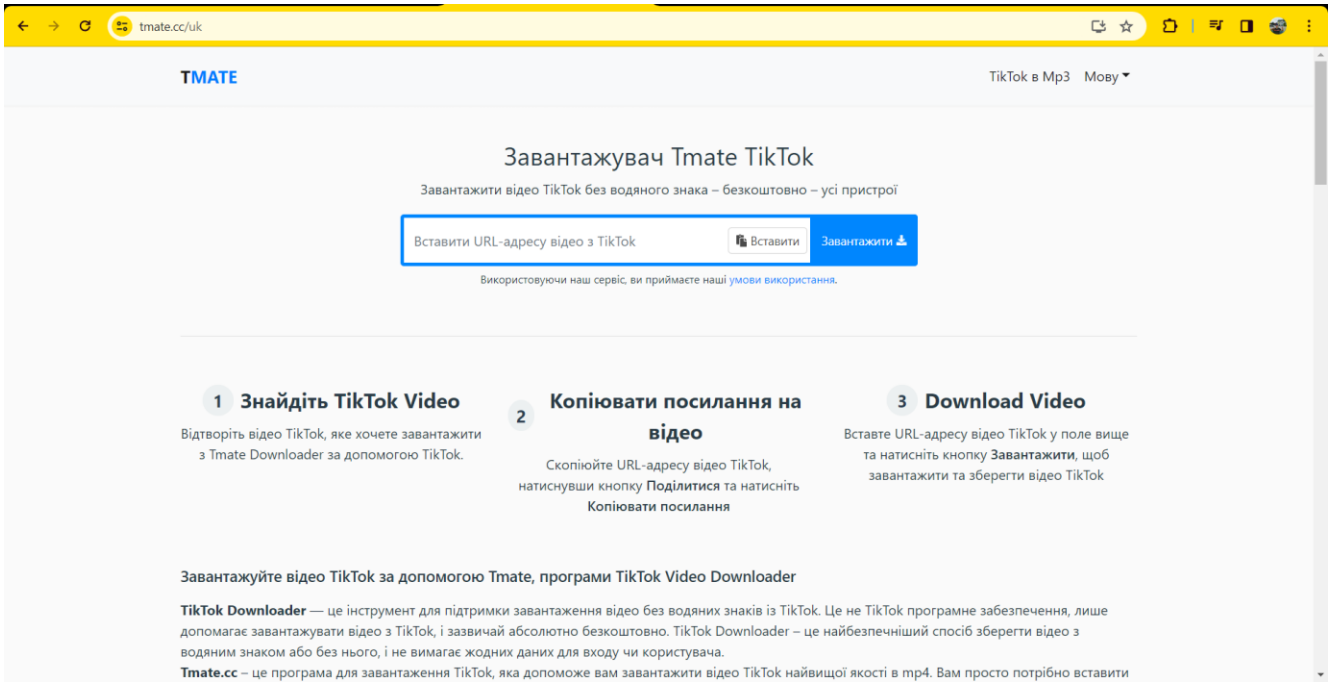


Рисунок 1.4 – Приклад роботи додатку Tmate

Останній аналог, що буде розглянуто – Live Photos (див. рисунок 1.5). Після зафіксування Live Photo, ви можете вибрати іншу ключову мить зі зйомки, що дозволяє вам зберегти саме те, що ви хочете [13].

Додатково, можемо застосовувати різноманітні цікаві ефекти та редагувати Live Photo безпосередньо на пристрої. Це включає різні фільтри, корекцію кольору та інші інструменти для покращення зображення. А також

після завантаження Tik Tok користувач також зможе видалити водяний знак, проте цей програмний застосунок має величезний недолік, а якщо бути точним цілий ряд недоліків.

По-перше, цей аналог не підтримує роботу на різних платформах, що становить його головний недолік. До того ж, обмежена функціональність також є важливим недоліком цього продукту.

Після того, як користувач відредагує Live Photo, він можете легко поділитися ним зі своєю сім'єю та друзями через різноманітні способи, такі як повідомлення, електронна пошта або соціальні мережі. Це дозволяє користувачам зручно ділитися цими спеціальними моментами, роблячи спільний перегляд ще цікавішим та захоплюючим. [13].

Рисунок 1.5 – Приклад роботи додатку

Розробка програмних засобів Telegram-боту для завантаження TikTok та Youtube відео без водяного знаку вимагає поєднання досвіду у розробці програмного забезпечення, розуміння принципів роботи API платформ TikTok та YouTube, а також вміння ефективно взаємодіяти з Telegram API. Цей процес передбачає створення бота, який може швидко та надійно обробляти запити користувачів для завантаження відео з вказаних платформ.

Для реалізації цієї функціональності спочатку потрібно зрозуміти, як працюють API TikTok та YouTube для завантаження відео. Далі необхідно реалізувати логіку бота, яка буде відповідати на команди користувачів у Telegram і виконувати відповідні запити до API цих платформ.

Результати порівняння аналогів зведено в табл. 1.1.

Таблиця 1.1 – Порівняльні характеристика онлайн платформ

Критерій	TokMate Bot	SaveFrom.net	Ssstik.io	Tmate	Live Photos	Власний застосунок
Завантаження відео з Tik Tok	+	+	+	+	+	+
Завантаження відео з YouTube	–	+	–	–	–	+
Доступність функцій з нестабільним інтернетом	+	–	–	–	+	+
Можливість обирати якість відео	–	+	–	–	–	+
Кросплатформенність	+	+	+	+	–	+
Звітність, щодо використаних транзакцій	–	+	–	–	–	+
Загальна оцінка	50%	90%	20%	20%	20%	100%

Відповідно до таблиці порівняльних характеристик розробка власного Telegram-боту є доцільною. Отриманий продукт зможе покрити недоліки існуючих рішень та забезпечити новий досвід та можливості під час завантаження та обробки відео та аудіо контенту з платформ Tik Tok та YouTube.

Отже, зважаючи на потужний розвиток мобільних технологій та соціальних мереж, розробка програмних засобів для Telegram-боту, що дозволить завантажувати відео з найпопулярніших платформ, таких як TikTok та Youtube, без водяного знаку, відкриває перед користувачами широкі можливості отримання зручностей та комфорту під час споживання контенту. Розуміючи потенційні переваги цього програмного рішення та вимоги до його розробки, компанії можуть створювати інноваційні та привабливі Telegram-боти, що є перспективним напрямком для бізнесу.

1.3 Аналіз методів розв’язання задачі

У цьому розділі розглянемо найбільш ефективні методи розробки програмного засобу для створення Telegram-бота, який дозволяє завантажувати відео з платформ TikTok та Youtube без водяного знаку.

1. Створення бота для готової платформи Telegram.

Перший варіант вирішення цієї задачі – розробка Telegram-бота для готової платформи Telegram. Цей метод вважається швидким та зручним, оскільки платформа Telegram вже має необхідні інструменти для створення ботів. Такий підхід дозволить швидко розпочати роботу над функціоналом завантаження відео з TikTok та Youtube без водяного знаку. Бот може бути розроблений за допомогою мов програмування, таких як Python, та використанням відкритих API цих платформ.

2. Розробка окремого сервісу для інтеграції модуля.

Другий підхід – створення окремого сервісу для інтеграції модулю в існуючі системи. Це означає розробку самостійної платформи, яка буде забезпечувати функціонал завантаження відео без водяного знаку з TikTok та

Youtube. Такий сервіс може бути більш гнучким і пристосованим до конкретних потреб користувачів. Проте, для цього методу буде потрібно врахувати розробку внутрішньої системи авторизації та забезпечення безпеки даних, що може зайняти більше часу та ресурсів.

3. Використання готових сервісів для завантаження відео.

Третій варіант – використання вже існуючих сервісів для завантаження відео з платформ TikTok та Youtube. Існують сервіси, які пропонують функціонал без водяних знаків, інтеграція з якими може бути швидким та ефективним рішенням. Однак, варто врахувати, що використання сторонніх сервісів може обмежити функціонал та контроль над процесом завантаження.

Після уважного аналізу різних методів вирішено, що найбільш ефективним та швидким рішенням буде створення Telegram-бота для готової платформи Telegram. Цей метод має численні переваги:

1. Завдяки наявності готової платформи Telegram та можливості використання її API, розробка бота буде досить швидкою та ефективною.

2. Бот на платформі Telegram вже має свою аудиторію та інтерфейс, що зробить процес використання більш зручним для користувачів.

3. Платформа Telegram має різноманітні можливості для розширення функціоналу бота, який може бути корисним для майбутнього розвитку проекту.

4. Мінімальне навантаження на апаратне забезпечення під час використання готової платформи дозволяє уникнути значного навантаження на апаратне забезпечення та оптимізувати витрати на розробку.

Отже, обране рішення відповідає вимогам ефективності, швидкості та зручності використання для користувачів. Розробка Telegram-бота дозволить створити продукт, який буде швидко реалізований та зручно використовуваний для завантаження відео без водяного знаку з платформ TikTok та Youtube.

1.4 Постановка задач бакалаврської дипломної роботи

Інформаційні технології в сучасному світі відіграють важливу роль у різних сферах життя, включаючи комунікації, розваги та навчання. Однією з популярних платформ для спілкування та отримання різноманітної інформації є Telegram. У зв'язку з цим, створення Telegram-ботів стає актуальною задачею для вирішення різних завдань.

У бакалаврській дипломній роботі треба виконати такі задачі:

– розробити методи роботи системи;

– розробити алгоритм для зчитування посилань за допомогою обраних бібліотек та API;

– розробити алгоритм обробки помилок та коректівізувати відповідь бота на некоретно отримані дані;

– розробити зручний та простий у розумінні інтерфейс, систему кнопок для інформування користувача про функціонал боту;

– розробити автоматизовану базу даних для моніторингу використаних транзакцій;

– розробити функціонал, що дозволяє завантажувати відео-матеріали будь-якого розміру в незалежності від якості відео;

– провести тестування розробленого боту згідно поставлених задач.

1.5 Висновки

У першому розділі бакалаврської дипломної роботи було розглянуто стан Telegram-ботів та їх актуальність. Боти стають все більш важливим інструментом для різних сфер діяльності, від бізнесу до освіти та розваг, надаючи широкі можливості для автоматизації процесів та залучення аудиторії.

Проведено порівняльний аналіз аналогів Telegram-ботів, які дозволяють завантажувати відео з платформ TikTok та Youtube без водяного знаку. Перелічено відомі аналоги, такі як TokMateBot, SaveFrom.net, Ssstik.io, Tmate, Live Photos, зазначивши їхні переваги та недоліки. Було відзначено, що кращим

рішенням для цього проєкту буде розробка власного Telegram-бота для готової платформи Telegram з використанням її API та Python для програмування.

Проаналізовано методи розв'язання поставленої задачі розробки Telegram-боту. Обгрунтовано вибір кращого методу, який полягає у створенні бота для готової платформи Telegram. Цей вибір обумовлений швидкістю розробки, зручністю для користувачів та можливістю розширення функціоналу.

На основі проведеного аналізу було сформульовано завдання бакалаврської дипломної роботи, які необхідно виконати для успішної реалізації програмного засобу.

Таким чином, розробка програмного засобу Telegram-боту для завантаження TikTok та Youtube відео без водяного знаку має стратегічну важливість у сучасному цифровому середовищі, і виконання вищезазначених завдань допоможе досягти успішної реалізації цього проєкту.

2 РОЗРОБКА МЕТОДІВ, МОДЕЛІ ТА АЛГОРИТМІВ ПРОГРАМНОГО ПРОДУКТУ

2.1 Визначення інструментів та аналіз вхідних даних системи

Цей етап реалізації бакалаврської дипломної роботи визначає інструменти та методи, які будуть використовуватися для отримання та обробки даних, а також аналіз даних, що будуть оброблятися застосунком.

Розгляд вхідних даних дозволяє точно визначити потреби користувачів та визначити, які саме дані необхідно отримати та обробити для забезпечення оптимального функціоналу боту.

Для реалізації Telegram-боту для завантаження відео з TikTok та Youtube будуть використані PyTube та Aiogram.

PyTube – це бібліотека на мові програмування Python, яка забезпечує зручний доступ до YouTube API. Завдяки PyTube можна отримати різноманітні дані про відео на YouTube, такі як заголовки, посилання на відео, кількість переглядів, рейтинг, теги, а також інші метадані, які можуть бути корисні для подальшого аналізу чи обробки відео [14].

Aiogram – це фреймворк на мові програмування Python для створення Telegram-ботів. Забезпечує широкі можливості взаємодії з користувачами: керування повідомленнями, реакція на команди, створення інтерактивних клавіатур та інші функції. Цей інструмент є потужним для розробки Telegram-ботів з різним функціоналом, популярним серед розробників у галузі. [15].

Поєднання цих інструментів дозволить створити зручний та функціональний Telegram-бот для завантаження відео з вказаних платформ та максимально зручно використовувати вхідні дані.

Після визначення інструментів, які будуть використовуватися для розробки Telegram-боту, наступним кроком є аналіз вхідних даних системи. Оскільки бот призначений для завантаження відео з платформ TikTok та YouTube, вхідні дані можна розділити на дві основні частини: дані, отримані зі

сторінок відео на обох платформах, та дані, що стосуються взаємодії з користувачем через Telegram.

Почати слід з вхідних даних, що отримуються з платформи YouTube та обробляються за допомогою бібліотеки PyTube:

1. Заголовок відео – описує суть відео та використаний для створення опису в Telegram-повідомленні, що містить посилання на відео. Також є важливою складовою під час реалізації відповіді боту користувачеві.

2. Посилання на відео – необхідна складова для початку завантаження відеоряду та пошуку відео за допомогою URL-адреси у більшості методів застосунку, наприклад визначення типу відео.

3. Розмір відеоряду – змінна, що дозволяє реалізувати алгоритм обтину відео, для можливості обходження обмеження на файли, що може надсилати Telegram-бот.

4. Назва каналу – дані, що використовуються для формування відповіді користувачеві на результат завантаження відео.

Дані з Aiogram для взаємодії з користувачем:

1. Повідомлення від користувача – текстові повідомлення, які містять команди або запити від користувача, наприклад, запит на завантаження відео з конкретного джерела.

2. ID-користувача – дані необхідні для формування бази даних, які бот використовує під час початку роботи з застосунком.

3. Дата та час повідомлення – складова для формування звіту у базі даних, яка використовуєть та записується у базу під час кожної операції, що виконує користувач.

Вхідні дані для використання RapidAPI:

1. URL-відео – адреса відео, яка потім буде використана для завантаження в Telegram.

Після отримання цих даних, бот може обробити їх та використовувати для створення повідомлень для користувачів Telegram, які містять інформацію про

відео та посилання на них. Також, бот повинен розуміти команди користувачів та відповідати на них, ініціюючи процес завантаження відео з YouTube або TikTok.

Отримання та аналіз вхідних даних є ключовим етапом у розробці Telegram-боту для завантаження відео з платформ TikTok та YouTube. Використання бібліотек PyTube та Aiogram надає широкі можливості для отримання та обробки даних з цих платформ, а також для взаємодії з користувачами.

2.2 Аналіз функціоналу боту

Зв'язок між Telegram та ботом для завантаження відео з платформ, таких як YouTube, TikTok та YouTube Shorts, вирішує низку проблем для користувачів. Користувачам не доведеться витрачати час на ручне копіювання посилань на відео з різних платформ, оскільки бот автоматично завантажує відео та надсилає його у зручному форматі. Це рятує час та зусилля, дозволяючи швидко отримувати доступ до потрібного відео. Основні функції Telegram-боту:

1. Завантаження відео без водяних знаків – користувачі можуть швидко та легко завантажувати відео з TikTok та Youtube без водяних знаків чи зайвих елементів на відео.
2. Командний інтерфейс – інтерфейс, що дозволяє вводити спеціальні команди для виконання різноманітних завдань, наприклад, пошук та завантаження відео з обраних платформ.
3. Інтерактивне меню – бот має інтерактивне меню на основі кнопок, що спрощує взаємодію з ним. Користувачі можуть вибирати опції, не вводячи текстові команди, включаючи довідку та FAQ.
4. Звітність та статистика – після завантаження відео користувачем, з'являється можливість отримати звіт, який містить статистику про кількість завантажених відео та інші дані у зручному форматі.

5. SQLite3 використовується для створення та управління базою даних, де зберігається інформація про завантажені відео та статистика користувальницької активності.

Реалізована функціональність Telegram-бота надає користувачам зручний та швидкий спосіб завантаження відео з платформ TikTok та Youtube без водяних знаків. Бот відповідає на спеціальні команди від користувачів, що дозволяє виконувати різноманітні дії, такі як завантаження відео або відправлення посилань. Користувачі можуть легко знаходити відео за ключовими словами чи посиланнями, що значно спрощує процес пошуку.

Інтерактивне меню бота робить взаємодію з ним більш зручною, надаючи швидкий доступ до основних функцій, таких як пошук, завантаження та довідкова інформація. Командна логіка спрощує використання бота, а система збереження даних у форматі звіту (наприклад, за допомогою бази даних SQLite3) дозволяє вести облік кількості завантажених відео, дій користувачів та іншої статистики.

Також існує широкий простір для можливих подальших розширень функціоналу, наприклад можливість підписки на оновлення, що дозволить коористувачам підписатися на отримання сповіщень про нові відео від певних каналів. Це може бути зручно для тих, хто хоче слідкувати за певним контентом на платформах і одночасно завантажити його для подальшої роботи з готовим відеорядом.

Отже, використання PyTube та Aiogram у розробці дозволяє створити ефективний інструмент для швидкого доступу до медіа-контенту. Визначені інструменти сприяють створенню потужного бота, який не має зайвих обмежень та водяних знаків. Інтерактивне меню та командна логіка роблять взаємодію з ботом простою та зрозумілою для користувачів, а збереження статистики дозволяє аналізувати активність та популярність завантаженого вмісту.

У контексті загальної роботи над розробкою Telegram-боту для завантаження відео з платформ TikTok та YouTube, аналіз проблеми важливий

для розуміння та визначення ключових викликів та перешкод, з якими можна зіткнутися під час розробки бакалаврської дипломної роботи.

Цей підрозділ враховує чинники як позитивного, так і негативного впливу на реалізацію застосунку. Важливим є правильне узагальнення накопиченого фактичного матеріалу, групування та оброблення даних для кваліфікованого аналізу та обґрунтування пропозицій.

До факторів позитивного впливу можна віднести:

1. Популярність платформ, таких як TikTok та YouTube є одними з найпопулярніших медіа-платформ у світі, що забезпечує широкий потенціал аудиторії для Telegram-боту.

2. Інструменти розробки, що були визначені вище дозволяють впроваджувати потужний функціонал, що полегшує розробку та взаємодію з користувачами.

3. Зручність для користувачів – бот, який спрощує процес пошуку та завантаження відео, може викликати позитивний відгук у користувачів та підвищити їхню задоволеність від використання платформ.

До факторів негативного впливу можна віднести:

1. Технічні обмеження – обмеження або зміни в API YouTube та TikTok можуть вплинути на ефективність роботи боту та призвести до необхідності швидкої адаптації програмного забезпечення.

До таких обмежень також можна віднести обмеження від самого Telegram. Таким стало обмеження для розміру файлів, що можуть надсилати боти користувачам у розмірі 50 мб. Таку проблему було вирішено інноваційним методом поділу відео. Було прийнято рішення не надсилати користувачеві повне відео у разі великого розміру файлу, а розбивати його на певні частини по 50 мб і надсилати файл-архів, що буде містити всі частини одночасно.

2. Питання авторських прав, що має на увазі збір та поширення відео може порушити авторські права, що може призвести до правових проблем.

Це питання також не є невирішальним, так у цій роботі бот не зможе

завантажувати відеоряд, що є захищеним авторським правом. Під час спроби завантажити такий відео-ролик користувача буде повідомлено про помилку у завантаженні.

3. Існує велика конкуренція в області розробки Telegram-ботів, що може ускладнити просування та призвести до втрати аудиторії.

Проблема конкуренція є постійною та гострою у будь-якій сфері. Щоб протистояти конкуренції в розробці Telegram-ботів, важливо зосередитися на унікальній пропозиції цінності для користувачів. Унікальний функціонал, який інші боти не мають, чи покращений досвід взаємодії з користувачем. Також важливо зосередити увагу на конкретній ніші або аудиторії, щоб створити бота, який найкращим чином задовольняє їхні потреби та вимоги.

Аналіз цих факторів допомагає зрозуміти загальний контекст проекту та ідентифікувати основні виклики та можливості для подальшої роботи. Враховуючи ці аспекти, можна розробити стратегії для оптимізації роботи боту та мінімізації можливих ризиків.

2.3 Розробка методу обробки відео-ресурсів

У сучасному цифровому середовищі, зростаюча популярність месенджерів, таких як Telegram, сприяє активному розвитку різноманітних ботів, які полегшують виконання різних завдань для користувачів.

У підрозділі буде розглянуто розробку методу обробки відео-ресурсів для Telegram-боту, що дозволяє користувачам зручно завантажувати великі відео-файли безпосередньо через месенджер.

Основною метою розробки цього методу є забезпечення стабільної та ефективної обробки ботом відео-ресурсів, що важать більше ніж 50 МБ. Це включає оптимізацію процесу завантаження відео, їх конвертацію в формат «mp4», обробку для надсилання таких відео по частинкам, а також забезпечення можливості перегляду відео безпосередньо в Telegram.

Описаний метод має враховувати різні технічні аспекти, такі як підтримка

різних форматів відео, обмеження розміру файлів та швидкість обробки, що вимагає ретельного підходу до розробки та тестування.

Описаний вище метод має таку структуру функціонування:

1. Після авторизації користувача, бот запитує у нього посилання на відео для завантаження. Користувач вводить посилання на відео з YouTube у чаті Telegram.

2. Бот отримує посилання та викликає асинхронний метод download_video з параметрами URL відео та повідомленням користувача.

3. Клас YouTube з бібліотеки pytube намагається завантажити відео за наданим посиланням. Якщо посилання є неправильним, викликається виняток RegexMatchError, і бот сповіщає користувача про помилку за допомогою методу message.answer.

4. Після успішного завантаження, назва відео обробляється функцією sanitize_filename для забезпечення коректності імені файлу. Відео зберігається у директорії, яка відповідає ID чату користувача, під назвою {message.chat.id}_{video_title}.mp4.

5. Метод перевіряє розмір файлу за допомогою атрибуту filesize об'єкта yt.streams.get_highest_resolution(). Якщо розмір файлу перевищує 50 МБ, бот сповіщає користувача про те, що відео занадто велике та буде розділене на частини, використовуючи метод message.answer.

6. Відео завантажується у найвищій якості за допомогою методу yt.streams.get_highest_resolution().download. Завантажене відео зберігається у вказаному шляху.

7. Бот додає запис про успішну конвертацію у базу даних за допомогою методу database.add_convertation, передаючи ідентифікатор користувача та статус 'Виконано'.

8. Метод виконує затримку на 3 секунди за допомогою функції await asyncio.sleep(3) для забезпечення стабільності процесу.

9. Якщо розмір файлу перевищує 50 МБ, відео розділяється на частини за

допомогою класу VideoFileClip з бібліотеки moviepy.

10. Відеофайл завантажується і відкривається для подальшої обробки, використовуючи методи бібліотеки moviepy.editor.

11. За допомогою атрибуту duration класу VideoFileClip визначається загальна тривалість відео в секундах.

12. Для поділу відео на частини встановлюється фіксована тривалість кожного сегмента у 180 секунд.

13. Обчислення кількості сегментів, на які буде розділене відео здійснюється за допомогою цілочисельного ділення загальної тривалості відео на тривалість одного сегмента з додаванням одиниці.

14. Для кожного сегмента працює визначення початкового (start_time) та кінцевого (end_time) часу сегмента.

15. Початковий час кожного сегмента обчислюється як start_time = i * chunk_duration, де i – номер сегмента, а chunk_duration – тривалість сегмента. Кінцевий час обчислюється як end_time = min((i + 1) * chunk_duration, duration), де duration – загальна тривалість відео.

16. Далі використовуючи метод subclip класу VideoFileClip, створюється підкліп відео з вказаними початковим та кінцевим часами. Підкліп зберігається у тимчасовий файл з унікальною назвою, яка включає номер сегмента, за допомогою методу write_videofile.

17. Надсилання підкліпу користувачеві за допомогою методу bot.send_video, супроводжуючи повідомлення інформацією про назву відео, канал та порядковий номер частини:

18. Тимчасовий файл підкліпу відкривається та надсилається користувачеві у вигляді відеоповідомлення з відповідним описом.

19. Після відправлення відеоповідомлення тимчасовий файл видаляється для звільнення дискового простору.

20. Якщо розмір відео не перевищує 50 МБ, воно надсилається користувачеві цілком за допомогою методу bot.send_video, разом з інформацією

про назву відео та канал.

21. Після відправлення відео, тимчасові файли видаляються для оптимізації використання дискового простору за допомогою функції os.remove.

22. У разі виникнення винятків, наприклад, RegexMatchError при неправильному посиланні на відео, бот сповіщає користувача про помилку за допомогою методу message.answer.

23. І на останок бот повертається у режим очікування наступного завдання від користувача, готовий обробляти нові запити на завантаження відео.

Алгоритм роботи цього методу наведено на рисунку 2.6 наведено першу частину.

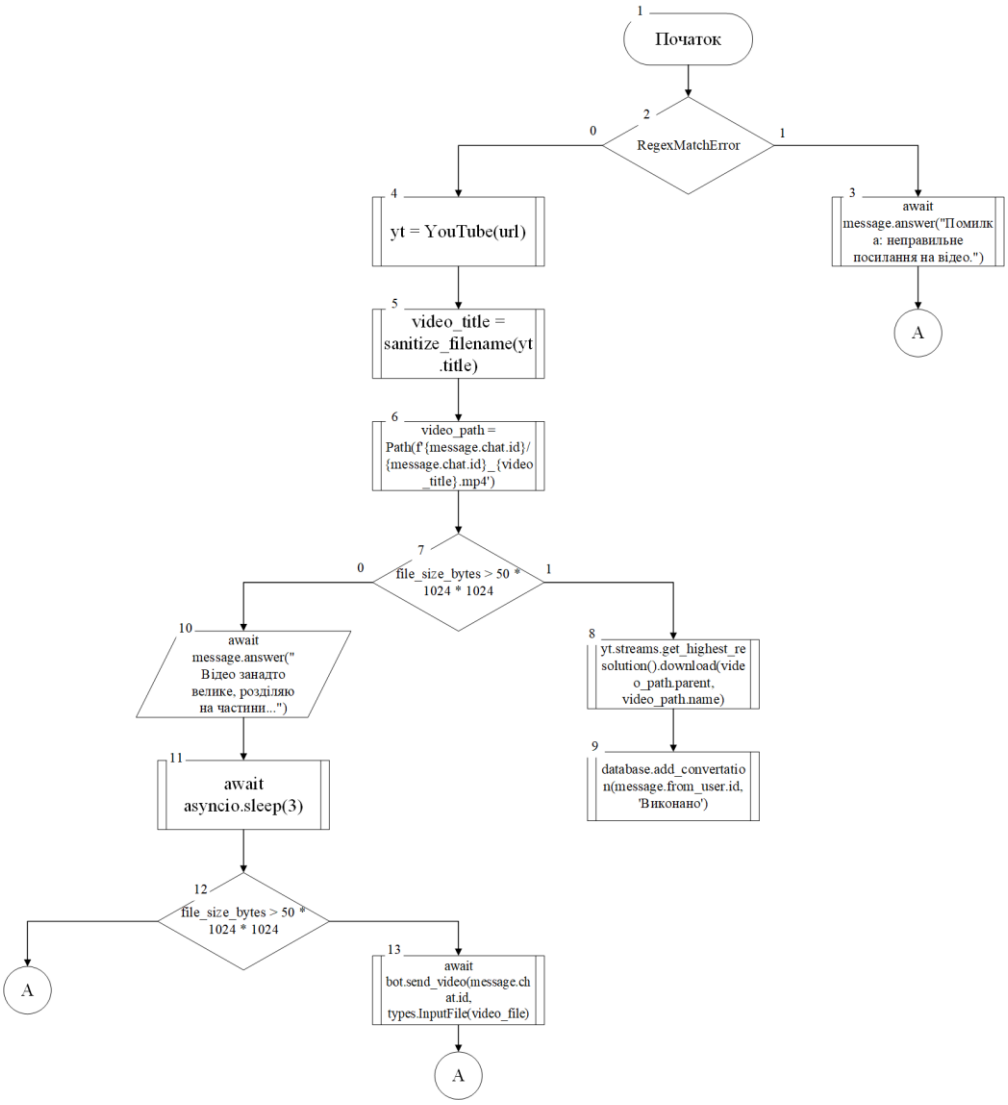


Рисунок 2.6 – Метод обробки відео-ресурсів (частина 1)

На рисунку 2.7 зображено продовження блок-схеми для методу обробки відео-ресурсів.

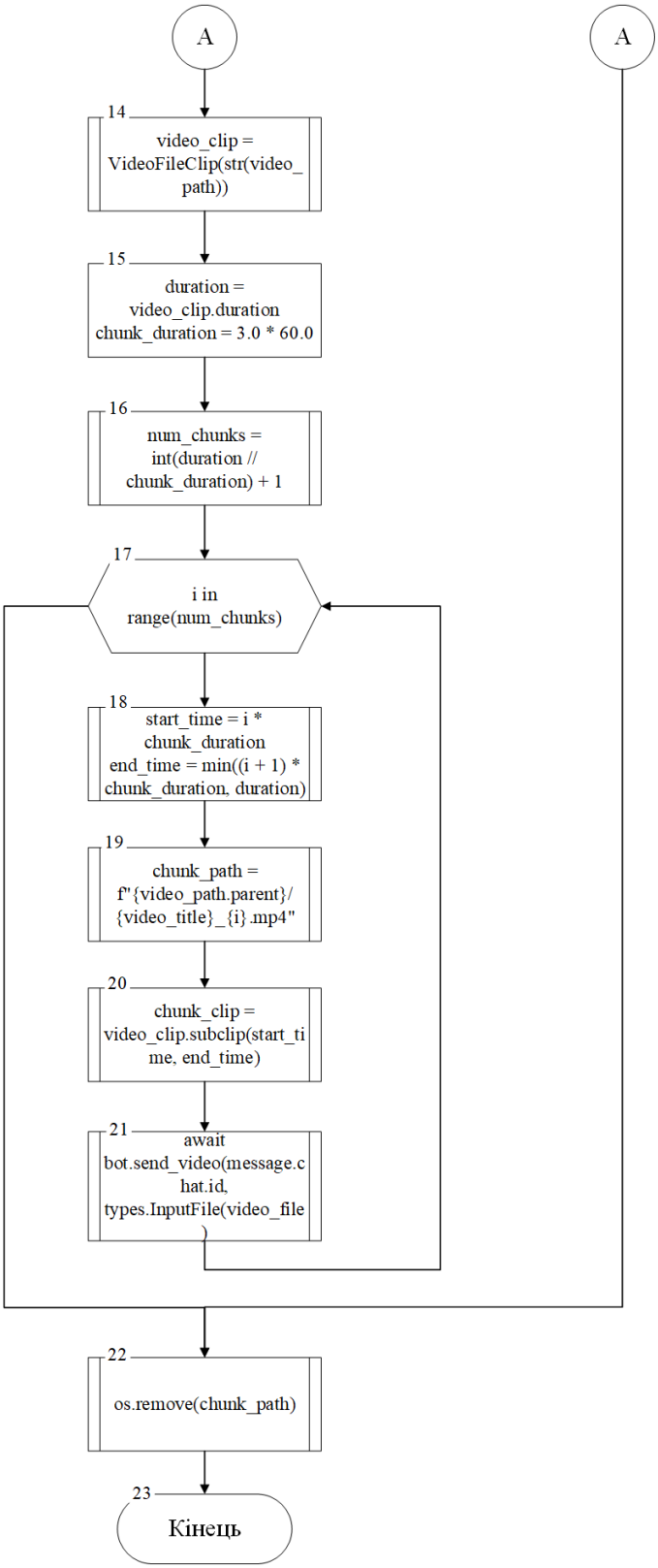


Рисунок 2.7 – Метод обробки відео-ресурсів (частина 2)

Отже у підрозділі було описано та візуалізовано роботу методу для обробки відео-ресурсів. В результаті опису було складено 23 пункти та створено блок-схему, що вміщає свій вміст на рисунках 2.6 та 2.7, а також у повному розмірі відображає функцію, яка відповідає за реалізацію цього методу.

2.4 Розробка методу пошуку відео

У цьому підрозділі буде розглянуто розробку методу пошуку відео-ресурсів на платформах, що користувач обере у процесі користування ботом. Загалом функціонал цього методу можна описати наступним чином:

Пошук відео, якщо бот визначив, що воно було надіслано з платформи YouTube:

1. Коли користувач надсилає посилання на відео з YouTube, спочатку проводиться перевірка посилання. Якщо воно дійсне, викликається функція process_youtube(url, message).

2. У функції process_youtube, спочатку викликається download_video(url, message) для завантаження відео з YouTube.

3. Для цього використовується бібліотека PyTube, яка взаємодіє з YouTube API, використовуючи ефективні алгоритми для формування HTTP-запитів.

4. HTTP-запити формуються згідно з документацією YouTube API та надсилаються до серверів YouTube для отримання відповіді з інформацією про відео.

Пошук відео, якщо бот визначив, що воно було надіслано з платформи TikTok:

1. Коли користувач надсилає посилання на відео з TikTok, спочатку проводиться перевірка посилання. Якщо посилання є дійсним, викликається функція process_tiktok(url, message).

2. У функції process_tiktok, викликається download_tiktok_video(link, message) для завантаження відео з TikTok.

3. В цьому випадку також використовується HTTP-запит, але до серверів TikTok через Rapid API. Формування запитів також відбувається відповідно до документації TikTok API та Rapid API.

Для пошуку відео програма взаємодіє з відповідними API платформ YouTube та TikTok, формуючи HTTP-запити з необхідними параметрами для пошуку відео. YouTube Data API та TikTok API оброблюють ці запити та повертають результат у форматі JSON або XML, який містить інформацію про знайдені відео.

Ці дані можна використовувати для подальшої обробки, такої як завантаження відео. Отже, використання API дозволяє програмі взаємодіяти з ресурсами платформ YouTube та TikTok та отримувати необхідну інформацію для подальшого використання.

2.5 Розробка моделі і алгоритмів

Для кращого розуміння роботи функціональних модулів Telegram-боту було вирішено розробити UML діаграму діяльності.

UML-діаграма – це графічне зображення, що використовується для візуалізації, спрощення та розуміння структури та поведінки системи [16].

Засоби UML, зокрема діаграми діяльності, стають надзвичайно корисними при розробці програмних продуктів, таких як Telegram-бот для завантаження відео з TikTok та Youtube без водяного знаку. Ці діаграми дозволяють візуалізувати послідовність дій, які користувач може виконувати при взаємодії з ботом (див. рисунок 2.6).

Діаграма відобразить послідовність операцій, які виконує бот у відповідь на різні вхідні сигнали або події. Такий підхід спрощує розуміння та аналіз функціональності бота, допомагаючи виявити можливі точки оптимізації та вдосконалення. Діаграма може включати такі елементи, як дії, рішення, умови та переходи між станами, що утворюють повну картину роботи бота з точки зору функціональності.

Також спрощують розуміння та аналіз логіки програми, допомагаючи забезпечити ефективну та інтуїтивно зрозумілу інтеракцію з користувачем. Такий підхід сприяє покращенню якості та функціональності програмного продукту, а також зменшує ризик помилок та непорозумінь під час розробки.

Користувач починає взаємодію з ботом, введуючи команду "/start". Після цього дії користувача реєструються у базі даних бота. Вони вітаються з базовим меню, що містить такі кнопки як "Help", "FAQ", "YouTube", "TikTok".

Ці кнопки дозволяють користувачу легко та швидко навігуватися між можливостями бота. Наприклад, "Help" надає короткий опис можливостей бота та інструкції по використанню. "FAQ" дає відповіді на найбільш часто задавані питання. "YouTube" та "TikTok" відкривають доступ до відповідних сервісів для завантаження відео. "Stat" відображає статистику користування ботом, наприклад, кількість транзакцій і т.д. .

Крім використання кнопок, користувач може вводити команди безпосередньо шляхом введення "/". Це дає змогу зручно та швидко викликати певні функції бота. Наприклад, користувач може викликати "Help" за допомогою "/help".

Загальна структура взаємодії з ботом через кнопки та команди "/start" і "/", надає простий та ефективний інструмент для користувачів. Кнопки з базовим меню дозволяють швидко переходити до потрібної функціональності, забезпечуючи легку навігацію. Водночас, можливість викликати функції за допомогою команд "/help", "/stat", тощо, надає додаткову гнучкість для більш швидкого доступу до необхідних опцій.

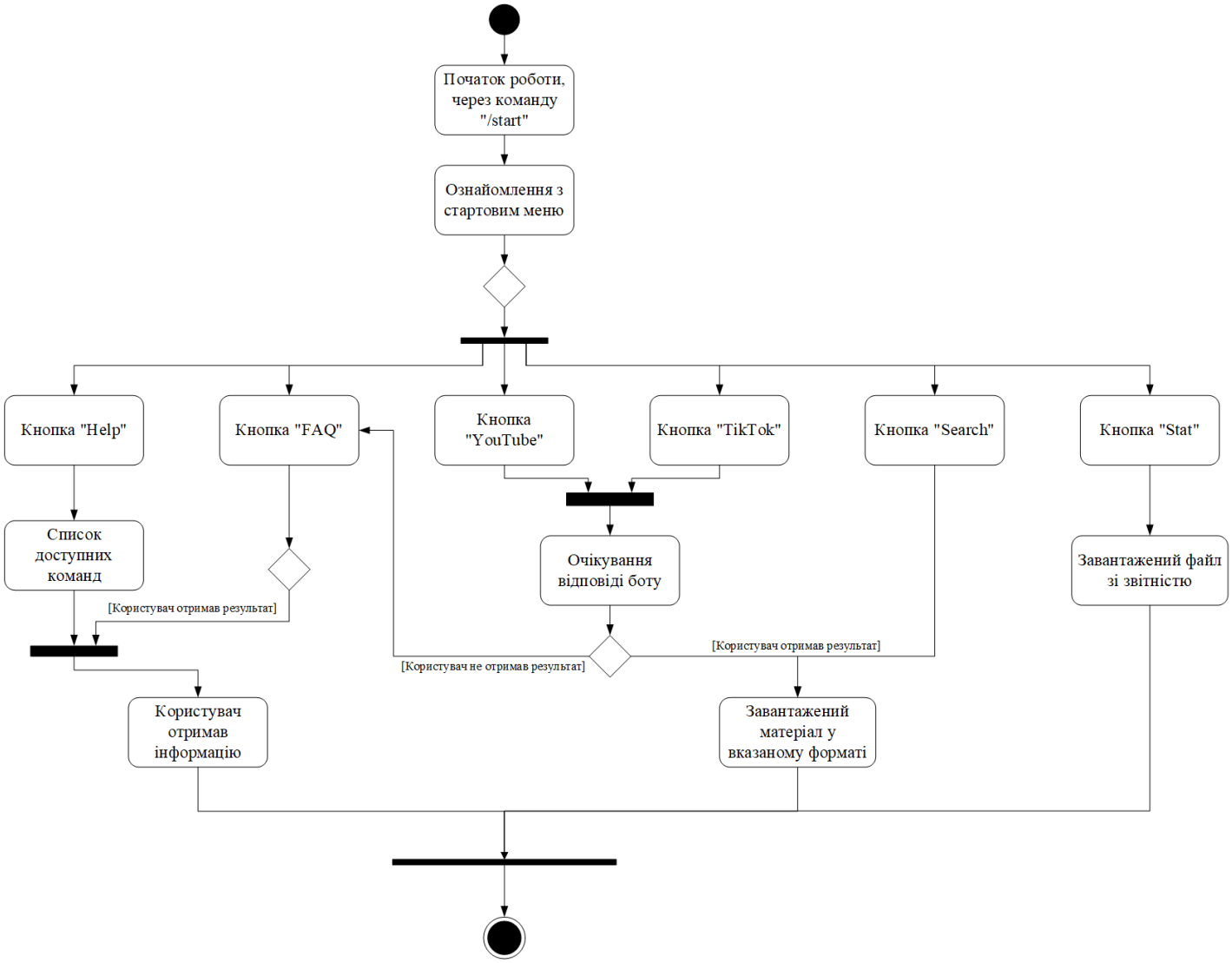


Рисунок 2.6 – Діаграма діяльності Telegram-боту для завантаження відео

Таким чином після створення і використання UML-діаграми допомагає не лише уявити послідовність подій в боті для завантаження відео, а й зрозуміти як взаємодіють між собою різні частини системи, що дозволяє здійснити ефективний процес розробки та забезпечити зручний функціонал для користувачів.

Для розробки Telegram боту, для завантаження TikTok та YouTube відео без водяного знаку, необхідно візуалізувати загальний алгоритм, згідно якого буде працювати бот (див. рисунок 2.7).

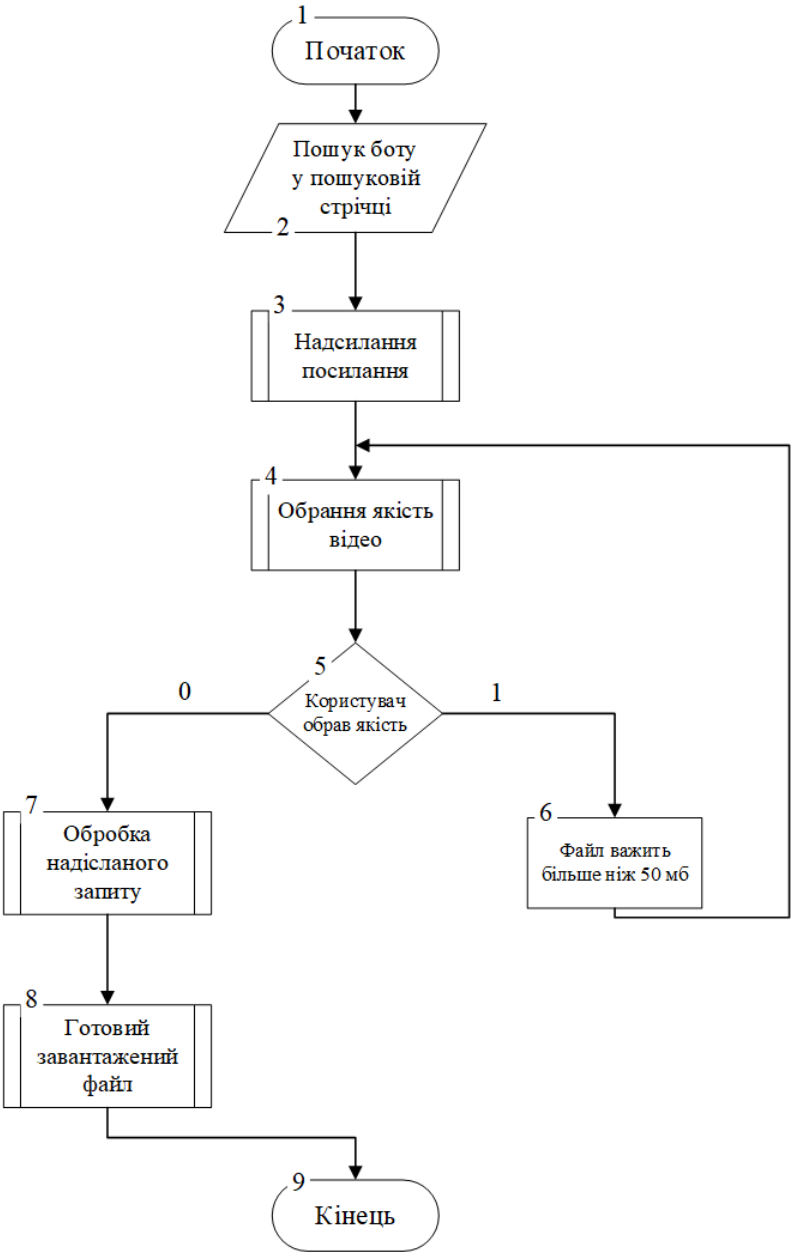


Рисунок 2.7 – Загальний алгоритм роботи програми

Згідно цього алгоритму, першим ділом, користувач знаходить бота у пошуковій стрічці Telegram. Після входу в систему, запускається модуль запису користувача у базу даних, а також модуль для розпізнавання запитів користувача, що у свою чергу перенаправляє отримані від користувача дані до відповідних модулів, які слідуючи їхньому унікальному алгоритму виконують поставлену задачу.

Почати детальніший розгляд алгоритмів слід з одного алгоритму

взаємодії користувача і бази даних. Коли користувач викликає команду /start, бот надсилає вітальне повідомлення та відкриває доступ до основного меню (див. рисунок 2.8). На цьому етапі і починається взаємодія, а саме після введення цієї команди користувач автоматично заноситься до бази даних і буде мати свій персональний ID-номер.

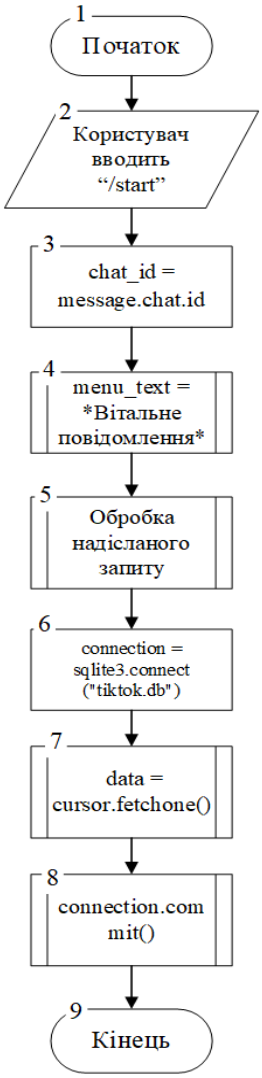


Рисунок 2.8 – Алгоритм для запису користувача в базу даних

Після цього, користувач може вибрати опції "YouTube" або "TikTok" у меню. Після вибору опції, бот повідомляє користувача про можливість надсилання посилання на відео.

Коли користувач надсилає посилання на відео, бот отримує його та розпізнає платформу. Після завершення завантаження, бот відправляє відео користувачеві разом із назвою відео та каналу у Markdown форматі (див. рисунок 2.9).

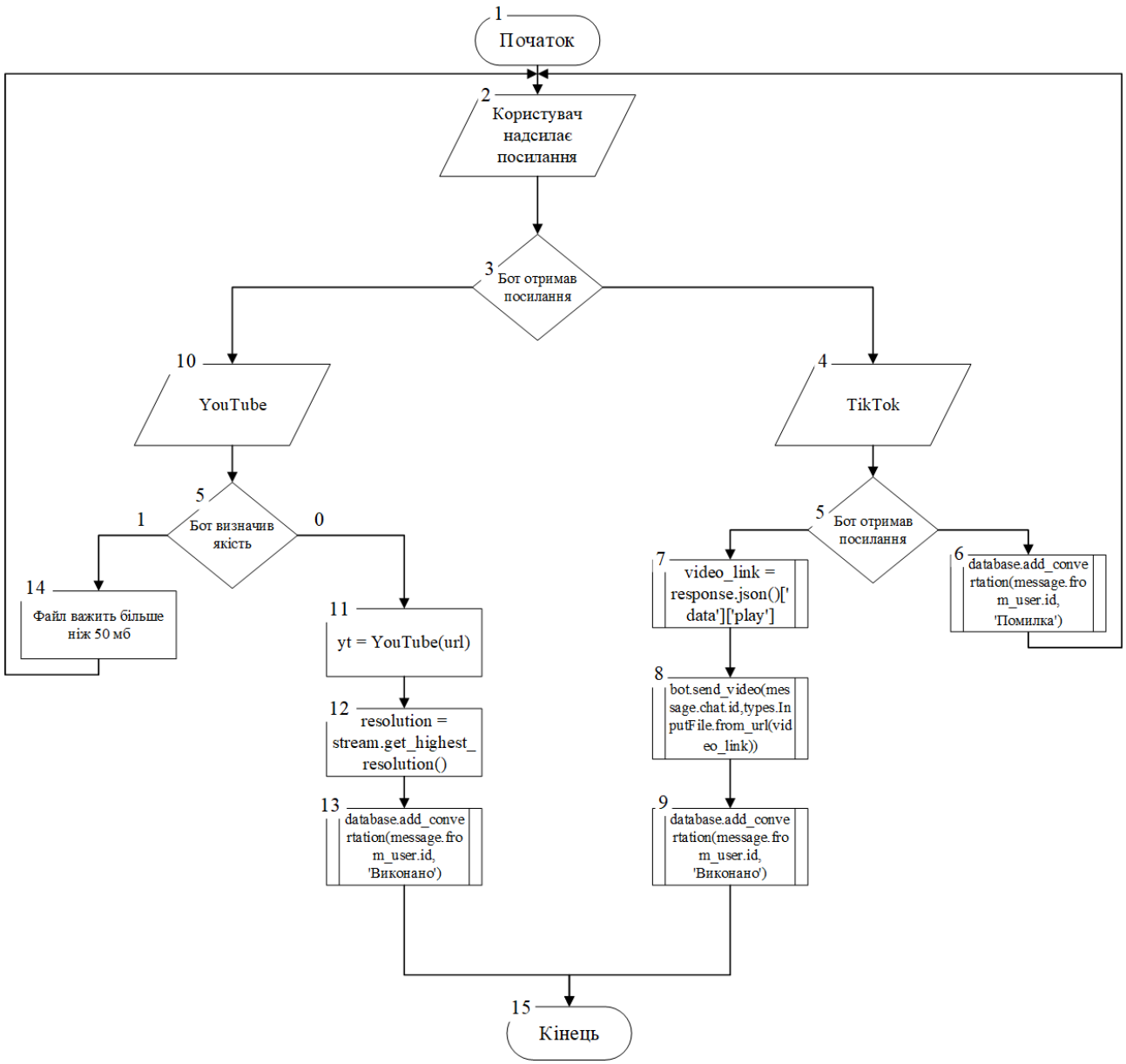


Рисунок 2.9 – Алгоритм визначення типу посилання

У разі, якщо це відео з TikTok, бот починає процес отримання відео з TikTok. Після отримання відео, він відправляє його користувачеві. Після відправлення відео, бот записує дані про користувача у базу даних.

Крім того, користувач може вибрати опцію "FAQ" для отримання інформації про те, як використовувати бота. Після завершення обробки запиту, бот також виконує запис про виконану операцію у базу даних, вказуючи, що користувач використав опцію YouTube чи TikTok.

2.6 Висновки

У другому розділі було виконано детальний аналіз вхідних та вихідних даних програмного продукту. На його основі була розроблена модель роботи за допомогою UML діаграми, яка відображає основні процеси та взаємозв'язки в програмі. Крім того, в цьому розділі було створено основний алгоритм роботи Telegram-боту, який визначає послідовність дій для обробки користувацьких запитів.

Також були розроблені два додаткові алгоритми: перший для взаємодії з базою даних, що включає в себе процес зберігання та отримання інформації з неї; і другий – для визначення типу посилання, який допоможе в розпізнаванні та подальшій обробці коректних посилань у боті.

3. РОЗРОБКА ПРОГРАМНИХ МОДУЛІВ РЕАЛІЗАЦІЇ TELEGRAM-БОТУ ДЛЯ ЗАВАНТАЖЕННЯ TIKTOK ТА YOUTUBE ВІДЕО БЕЗ ВОДЯНОГО ЗНАКУ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення

Зважаючи на вибір мови програмування Python та використання бібліотек Aiogram, PyTube, MoviePy і Rapid API для реалізації програмного забезпечення, можна провести порівняння з іншими можливими варіантами.

Python є високорівневою мовою програмування, яка відома своєю простотою, чистотою синтаксису та широким спектром бібліотек, які роблять його потужним інструментом для реалізації різноманітних проектів [17]. Його зрозумілий синтаксис дозволяє швидко розробляти функціонал без великої кількості вихідного коду.

Для розробки бота, який взаємодіє з Telegram API, було обрано Python через його багату екосистему бібліотек. Наприклад, для роботи з API Telegram використовується бібліотека Aiogram. Вона надає зручний інтерфейс для створення ботів та обробки повідомлень, подій та команд від користувачів.

Java – це мова програмування, що відома своєю платформенною незалежністю, що означає, що програми, написані на Java, можуть запускатись на будь-якій платформі, що підтримує відповідну віртуальну машину Java (JVM) [18].

Проте, для цього проєкту Java не є найкращим варіантом, оскільки створення бота для роботи з Telegram API вимагає швидкості розробки та взаємодії з API, що може бути затримано Java з великою кількістю бібліотек та додаткових налаштувань. Також, у порівнянні з Python, Java має більше формальності та вимагає більшої кількості коду для досягнення тих самих результатів.

JavaScript – це мова програмування, яка зазвичай використовується для динамічного та інтерактивного веб-змісту, такий як анімація, валідація форм, інтерфейси користувача та багато іншого [19]. Вона дозволяє створювати живу та змінну веб-сторінку, але для розробки бота для Telegram це не є оптимальний вибір. JavaScript, як інструмент веб-розробки, має свої сильні сторони у взаємодії з HTML та CSS, але робота з Telegram API та бібліотекою Aiogram краще вирішується за допомогою Python. Python має більше спеціалізованих інструментів для роботи з API та розробки ботів, що робить його більш придатним для цієї задачі.

Щодо C++, ця мова програмування відома своєю високою швидкодією та ефективністю, і часто використовується для розробки системних додатків, вбудованих систем та ігор [20]. Проте, для створення бота для Telegram це також не найкращий вибір. Розробка та підтримка коду на C++ може бути більшою витратою часу, оскільки ця мова вимагає більше формальності та має складніше синтаксичне використання порівняно з Python. Крім того, використання бібліотек для роботи з Telegram API та Aiogram більш зручне та ефективне на Python, що робить його оптимальним вибором для швидкої реалізації бота з багатофункціональними можливостями.

Aiogram – це бібліотека для взаємодії з Telegram API у Python. Вона дозволяє розробникам створювати потужних ботів для Telegram з великим функціоналом. Aiogram має простий і зрозумілий інтерфейс, що полегшує створення команд, обробників подій, відповідей на повідомлення та інше. Ця бібліотека також підтримує асинхронне програмування, що дозволяє боту працювати ефективно в реальному часі.

PyTube – це Python-бібліотека для роботи з YouTube API та завантаження відео з YouTube. Вона дозволяє отримувати інформацію про відео, таку як назва, опис, теги, а також завантажувати відео у високій якості. PyTube є популярним інструментом серед розробників, які працюють з відео на YouTube, завдяки його простоті використання та широкому функціоналу.

Rapid API – це платформа, яка надає доступ до різних API з різних сервісів. Вона пропонує широкий вибір API для різних потреб, від соціальних мереж до фінансових сервісів. З допомогою Rapid API розробники можуть легко і швидко інтегрувати різні функції та дані в свої додатки, використовуючи зручний інтерфейс. Ця платформа спрощує роботу з API, дозволяючи швидше реалізувати різноманітні можливості в програмному забезпеченні.

Отже, обрані технології для розробки програмних модулів для Telegram-боту з завантаження YouTube та TikTok без водяного знаку.

Обрані технології для розробки програмного модуля для Telegram-боту, який здатний завантажувати відео з YouTube та TikTok без водяного знаку, є важливим кроком у забезпеченні потужного функціоналу. Aiogram дозволяє легко створювати та керувати ботами для Telegram, маючи зрозумілий інтерфейс та підтримку асинхронного програмування для ефективної роботи в реальному часі.

PyTube є відмінним інструментом для роботи з YouTube API, надаючи можливість отримання та завантаження відео у високій якості без водяного знаку. Rapid API відкриває доступ до різних API з різних сервісів, що спрощує інтеграцію функціоналу у додатки.

Ці технології разом створюють потужний набір інструментів для розробки програмного модуля, який здатний забезпечити користувачів зручною та ефективною роботою з ботом, включаючи завантаження відео з популярних платформ без водяного знаку. У таблицю 3.1 було занесено основні критерії, за якими здійснювалося порівняння мов програмування, а також описано кожен пункт відповідно його значенню після безпосередньо таблиці.

Таблиця 3.1 – Порівняння мов програмування

Критерій	Python	C++	Java	JavaScript
Актуальність	+	–	+	+
Об’єктно-орієнтованість	+	+	+	–
Багатозадачність	+	+	+	–
Узагальнене програмування	+	+	+	+
Розподіленість	+	–	+	+
Легкість в освоєнні	+	+/–	–	+
Влучність для створення бота	+	–	+/–	–

Актуальність – вказує на значимість та сучасність мови програмування на сьогоднішній день, здатність втримати свою релевантність та актуальність у контексті швидко змінюючого індустріального та технологічного ландшафту.

Об'єктно-орієнтована мова програмування – базується на принципах об'єктно-орієнтованого програмування, де програма структурується навколо об'єктів, що взаємодіють між собою.

Багатозадачність – можливість операційної системи або середовища програмування обробляти декілька завдань одночасно, що дозволяє ефективно використовувати ресурси системи.

Узагальнене програмування відноситься до здатності мови програмування працювати з різноманітними типами даних та алгоритмами, що спрощує розробку різноманітних програм, включаючи ботів. Мова з високим рівнем узагальненого програмування може бути ефективно використана для широкого спектру завдань без значних модифікацій та адаптацій.

Розподіленість вказує на здатність мови програмування обробляти розподілені обчислення та комунікацію між вузлами в мережі. Для ботів, які можуть взаємодіяти з користувачами через різні пристрої та сервіси, важливо, щоб мова програмування мала засоби для ефективного керування розподіленими ресурсами та обміном даними.

Легкість в освоєнні оцінює, наскільки просто новачкам можна оволодіти мовою програмування та розпочати розробку програм. Мови з простим синтаксисом та широким спектром документації та навчальних матеріалів зазвичай вважаються легкими для освоєння.

Влучність для створення бота оцінюється на те, наскільки добре підходить конкретна мова програмування для створення чат-ботів та автоматизованих веб-програм. Це включає в себе наявність відповідних бібліотек та фреймворків, які полегшують розробку ботів, а також підтримку для взаємодії з популярними месенджерами та платформами.

В результаті порівняння мов C++, Python, JavaScript та Java було зроблено висновок, що мова Python є оптимальною для реалізації застосунку для бакалаврської дипломної роботи.

3.2 Розробка модуля для завантаження відео з обраних платформ

Основною функцією Telegram-боту є можливість завантаження відео з платформ TikTok та Youtube без водяного знаку. Реалізація цієї функції включає в себе використання різних API та бібліотек, які дозволяють отримувати відео з відповідних платформ та надсилати його користувачам Telegram.

Процес завантаження відео з TikTok через API і відправки його користувачу у чат Telegram представляє собою асинхронний метод для завантаження відео. Вона побудована на основі бібліотеки requests, що підтверджує використання методу requests.get(), щоб отримати відповідь від сервера TikTok. Проте, через те, що цей метод блокуючий, він може призвести до затримок у роботі програми, особливо якщо є багато запитів.

Спочатку, коли користувач надсилає посилання на відео з TikTok, бот надсилає повідомлення у чат користувача про те, що запит на завантаження відео обробляється: await message.answer("Будь ласка, зачекайте, обробляється ваш запит!").

Після цього, бот виконує запит до API сервісу TikTok за допомогою GET-запиту на tiktok-video-no-watermark2.p.rapidapi.com. У цьому запиті вказуються параметри, такі як посилання на відео та бажана якість (hd).

На рисунку 3.1 наведено блок-схему функції для завантаження відео з платформи TikTok.

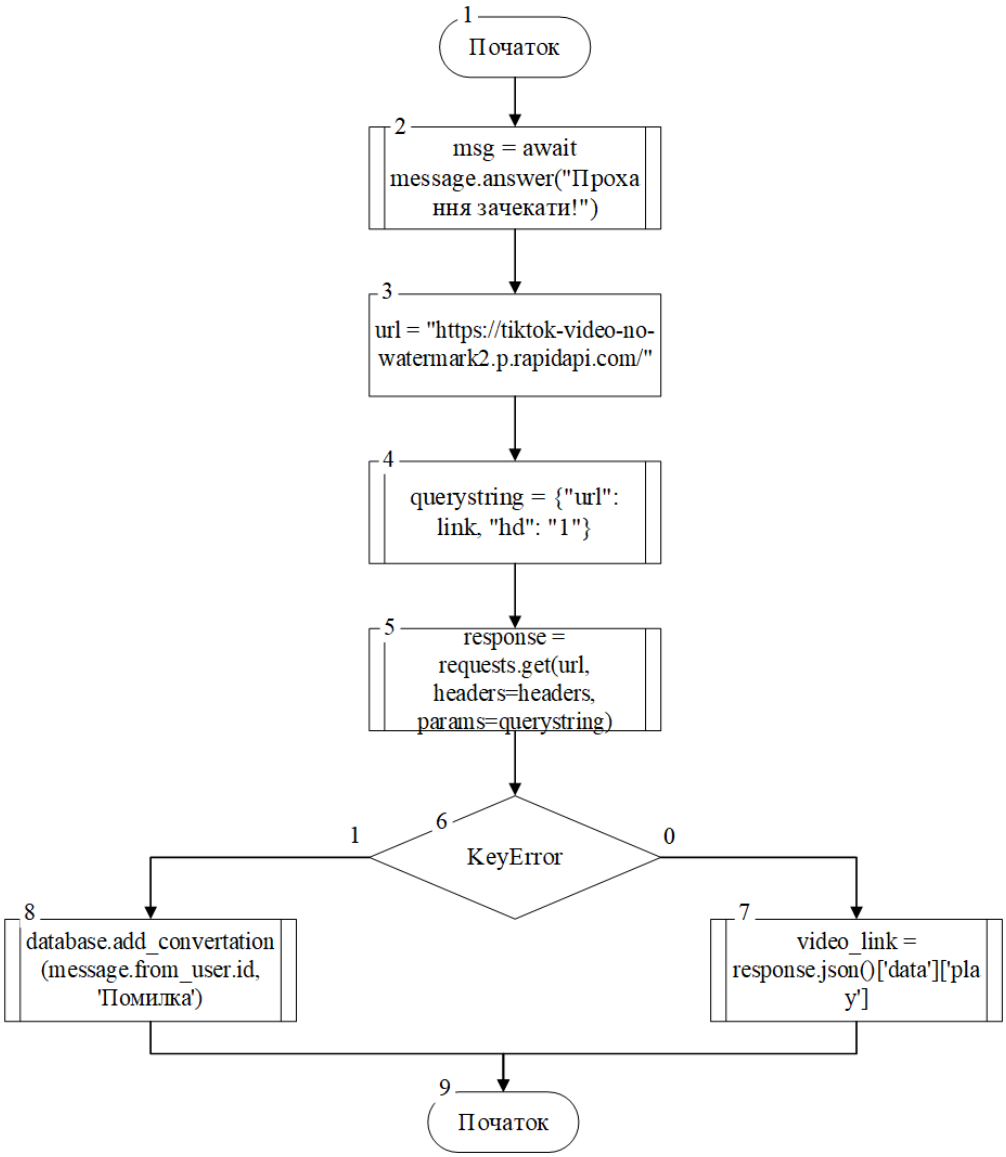


Рисунок 3.1 – Блок-схема функції для завантаження відео з платформи TikTok

Після успішного отримання відповіді від сервера TikTok у форматі JSON, бот витягує посилання на саме відео: video_link = response.json()['data']['play']. З отриманим посиланням на відео бот відправляє відео у чат користувача за допомогою bot.send_video(message.chat.id, types.InputFile.from_url(video_link)). Таким чином, користувач отримує відео без водяного знаку прямо у чаті Telegram.

Нарешті, коли відправлення відео успішно завершується, бот оновлює статус в базі даних, щоб відзначити, що операція завантаження відео була виконана: database.add_convertation(message.from_user.id, 'Виконано').

У випадку з YouTube, бот використовує бібліотеку pytube, яка дозволяє отримувати відео з YouTube за посиланням користувача. Після завантаження відео, бот також відправляє його у чат користувача, забезпечуючи можливість перегляду без водяного знаку.

У коді є функція async def download_video(url, message). Ця функція розроблена для завантаження відео з YouTube та його поділу на частини, якщо розмір перевищує 50 МБ. Починає вона з отримання URL відео з YouTube, визначення найвищої роздільної здатності, і завантаження відео у форматі MP4. Далі, вона перевіряє розмір завантаженого відео. Якщо розмір менше або дорівнює 50 МБ, відео надсилається без обрізки.

Якщо розмір перевищує 50 МБ, відео обрізається на частини за 5 хвилин (300 секунд) кожна. Після цього частини об'єднуються у RAR-архів, який надсилається користувачеві. Також для економій пам'яті та простору на диску, у кінці після надсилання відео видаляються тимчасові файли: архів, частини відео та вихідне відео.

Ця функція дозволяє ефективно завантажувати великі відеофайли з YouTube, забезпечуючи користувачам можливість перегляду та обміну їх, навіть якщо їх розмір перевищує обмеження.

Функція починається з створення об'єкта yt за допомогою посилання на відео з YouTube, яке користувач передає: yt = YouTube(url). Потім бот фільтрує

доступні потоки і обирає прогресивний потік з розширенням mp4: stream = yt.streams.filter(progressive=True, file_extension="mp4").

Після цього вибирається найвища доступна роздільна здатність для відео: resolution = stream.get_highest_resolution(). Обране відео завантажується з обраною роздільною здатністю у вказану директорію: resolution.download(video_path.parent, video_path.name).

Після завантаження відео, виконується перевірка розміру файлу, щоб визначити, чи необхідно розділити відео на частини. Розмір відео перевіряється таким чином: video_size = video_path.stat().st_size / (1024 * 1024), де розмір файлу переводиться у мегабайти.

Якщо розмір відео менше або дорівнює 50 МБ, воно надсилається у чат користувача без поділу на частини. Це виконується за допомогою методу await message.answer_video(types.InputFile(str(video_path))).

Якщо ж розмір перевищує 50 МБ, відео обрізається на частини. Обрізка відео здійснюється у циклі, де кожна частина відео має розмір приблизно 50 МБ. Кожна частина обрізається та зберігається у окремому файлі.

Після обрізки відео на частини, вони об'єднуються у ZIP-архів, який надсилається користувачеві. Для цього використовується метод shutil.make_archive(f'{video_path.parent}/{video_title}_parts', 'zip', f'{video_path.parent}').

На завершення, видаляються тимчасові файли: частини відео та вихідне відео. Це здійснюється за допомогою методів os.remove.

3.3 Розробка модуля взаємодії користувача з базою даних

При створенні програмного засобу для Telegram-боту, який взаємодіє з базою даних, важливо розробити ефективні методи взаємодії користувача з інформацією. Це включає в себе збереження даних, що вводяться користувачем через бота, такі як ім'я, контактні дані тощо. Крім того, витягнута інформація повинна бути доступна для подальшого використання, наприклад, для

формування звітів або статистики про користувачів та їх дії. Важливо також забезпечити безпеку даних, щоб інформація користувачів була захищена від несанкціонованого доступу.

При першому відкритті бота користувач додається до бази даних з його Telegram ID та ім'ям користувача (якщо воно доступне). Це зроблене для економії часу та зручності для користувачів (див. рисунок 3.2).

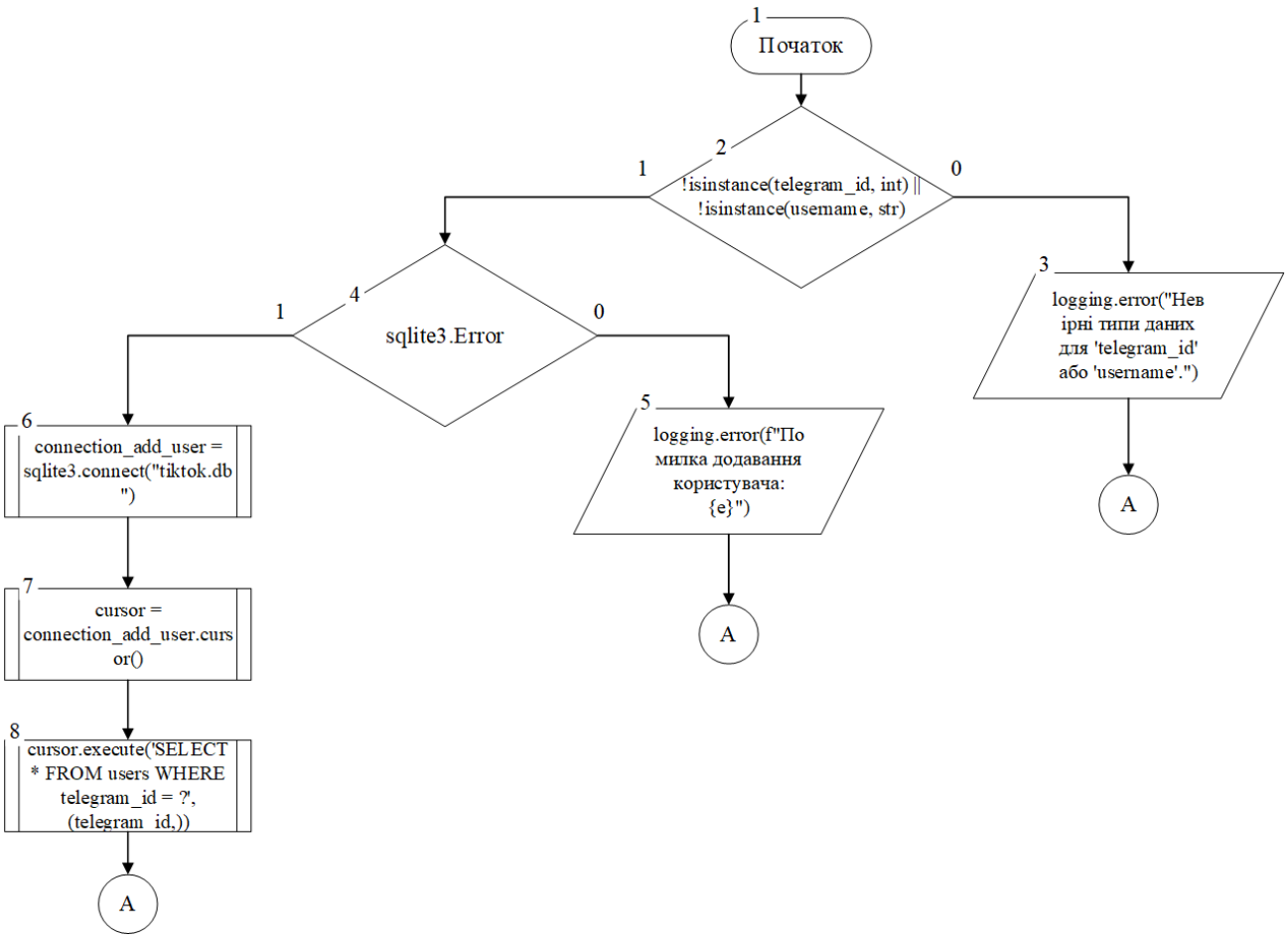


Рисунок 3.2 – Функція «add_user» (частина 1)

Функція add_user в базі даних перевіряє наявність користувача за його Telegram ID. Якщо користувач не знайдений, він додається до бази даних разом з датою реєстрації. Це зроблено для зручності самого користувача і для подальшого використання у функції з отриманням звідності про проведені операції з ботом (див. рисунок 3.3).

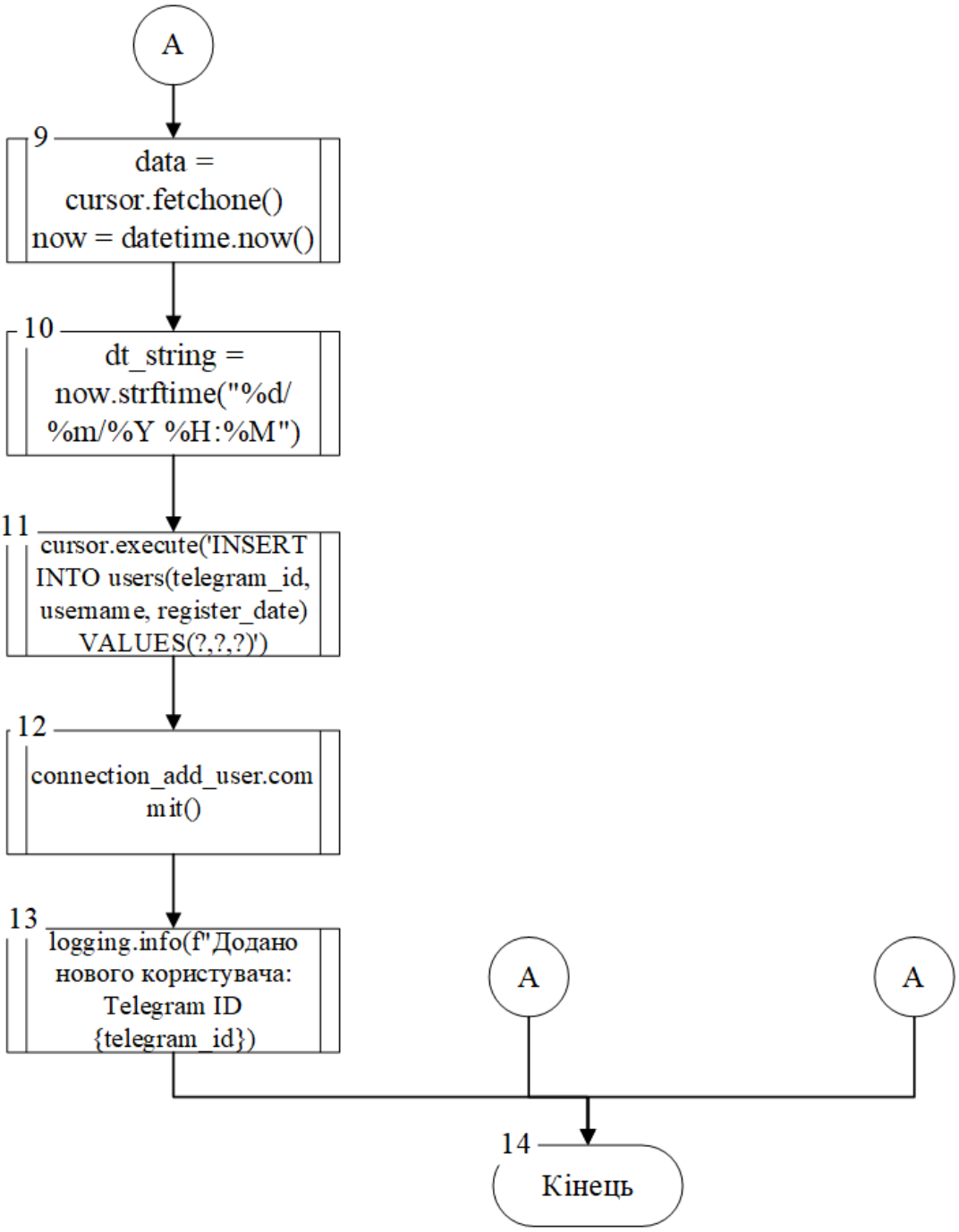


Рисунок 3.3 – Функція «add_user» (частина 2)

Функція add_convertation додає запис про конвертацію в базу даних SQLite. Ця функція використовується для збереження інформації про виконані конвертації відео з YouTube або TikTok. Після завершення завантаження відео з відповідного джерела, ця функція викликається для додавання запису про конвертацію. Вона додає новий запис в таблицю convertations, що містить

інформацію про конвертацію відео: telegram_id, дату та час реєстрації конвертації, а також статус конвертації ('YouTube' або 'TikTok'). Після вставки запису функція закриває з'єднання з базою даних.

Функція get_convertations використовується для отримання списку виконаних конвертацій з бази даних. Вона витягує всі записи з таблиці convertations і формує текстовий звіт, який потім може бути відправлений користувачеві.

На рисунку 3.4 наведено блок-схему для функції create_table_convertations.

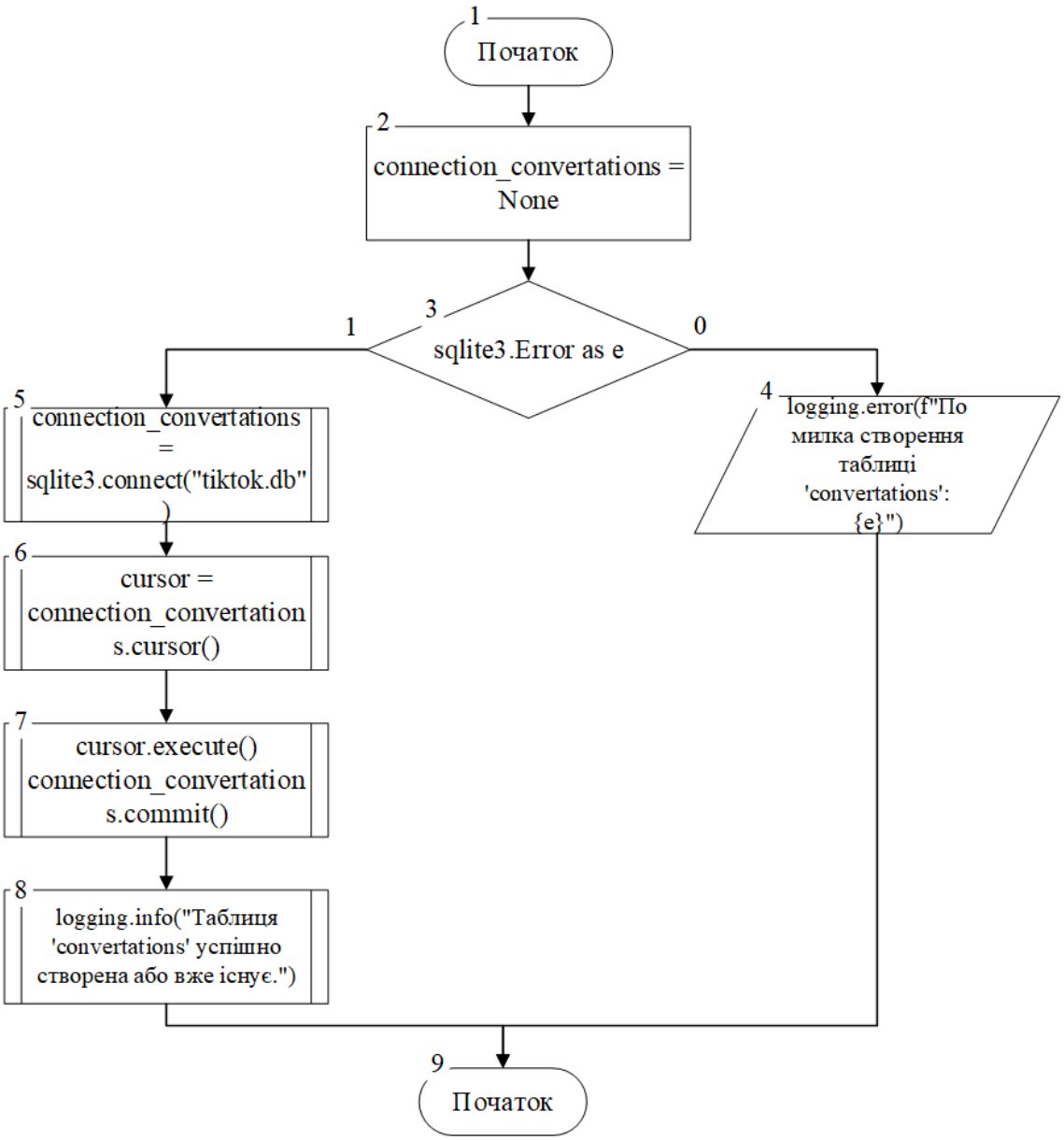


Рисунок 3.4 – Блок-схема функції «create_table_convertations»

Функція create_table_convertations в базі даних створює таблицю convertations, яка використовується для збереження інформації про виконані конвертації відео.

Функція create_table_users створює таблицю users, де зберігаються дані про користувачів, які використовують бота.

Всі ці функції грають ключову роль у веденні статистики та відстеженні активності користувачів, що використовують бот для конвертації відео з YouTube та TikTok. Ця інформація корисна для аналізу популярності різних джерел відео серед користувачів та для вдосконалення функціоналу бота.

3.4 Висновки

У третьому розділі було обґрунтовано вибір мови програмування та методів для розробки системи до Telegram-боту. Проаналізувавши існуючі мови програмування було прийнято рішення використовувати Python для реалізації поставленого завдання. Також було наведено приклади з реалізованими функціями, як у основному так і у коді бази даних. Реалізовані методи і реалізують функціонал, який необхідний для зручного використання боту у Telegram.

4 ТЕСТУВАННЯ БОТУ

4.1 Тестування реалізованої системи

Тестування Telegram-боту є процесом, спрямованим на перевірку його функціональності, надійності та коректності реакцій на вхідні дані та команди від користувачів [21]. Цей процес включає кілька ключових етапів, кожен з яких відіграє важливу роль у забезпеченні якості роботи бота.

На початковому етапі проводиться аналіз вимог до бота, що дозволяє зрозуміти його функціональність та очікувані можливості. Після цього розробляються тестові сценарії, які визначають послідовність дій для перевірки різних аспектів бота. Такі сценарії включають в себе введення різних команд, взаємодію з різними функціями та перевірку відповідей бота на ці дії.

Далі відбувається етап розробки тестових даних, які використовуються для симуляції взаємодії з ботом. Це може включати в себе різні типи вхідних даних, такі як текстові повідомлення, команди, файли тощо, щоб перевірити реакцію бота на різні ситуації.

Після підготовки тестових даних проводиться безпосереднє виконання тестів, де введені дані перевіряються на відповідність очікуваним результатам. В цей час тестувальник аналізує реакцію бота на вхідні дані, перевіряє коректність відповідей, а також можливі помилки в роботі.

Після виконання тестів здійснюється аналіз результатів. Виявлені помилки та недоліки документуються для подальшої виправлення розробниками. Також оцінюється загальна відповідність бота вимогам та його готовність до релізу або вдосконалення.

На рисунку 4.1 наведено початкову функцію, яку було прийнято протестувати, так як з неї починається взагалі будь-яка взаємодія з ботом, а саме відповідь на команду «/start».

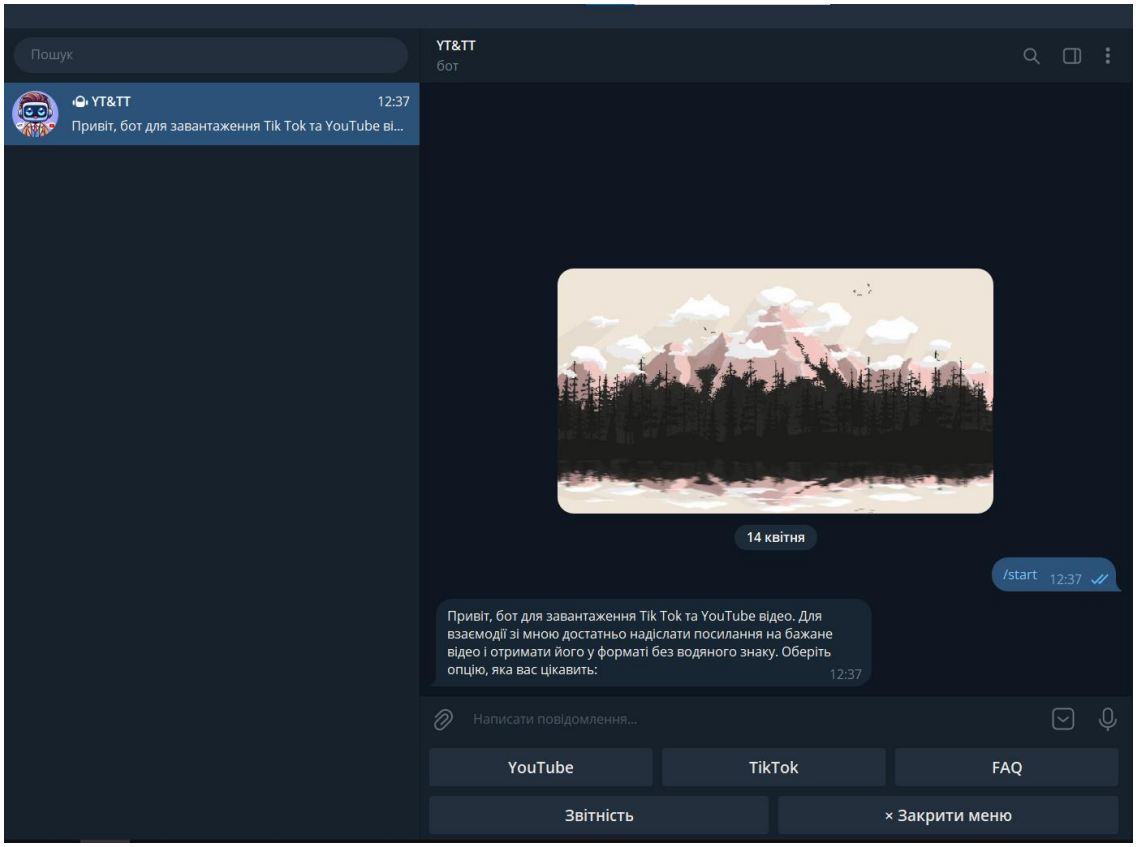


Рисунок 4.1 – Тестування відповіді на команду початку

Після введення команди бачимо, що бот адекватно відповів на повідомлення, а також видав стартове меню, яке додатково можна викликати командою «/help». Команда працює аналогічно, як команда «/start», проте різниця між командами проявляється у фіксації реєстрування користувача. Як вже було вище зазначено у боті реалізована автоматична фіксація користувачів у базу даних. На рисунку 4.2 наведено відповідь програми з консолі про успішну фіксацію користувача у базі даних.

Наступним кроком тестування було прийнято рішення перевірити реакцію бота на неправильну команду або хибне повідомлення, результат тестування цієї перевірки зображено на рисунку 4.3.

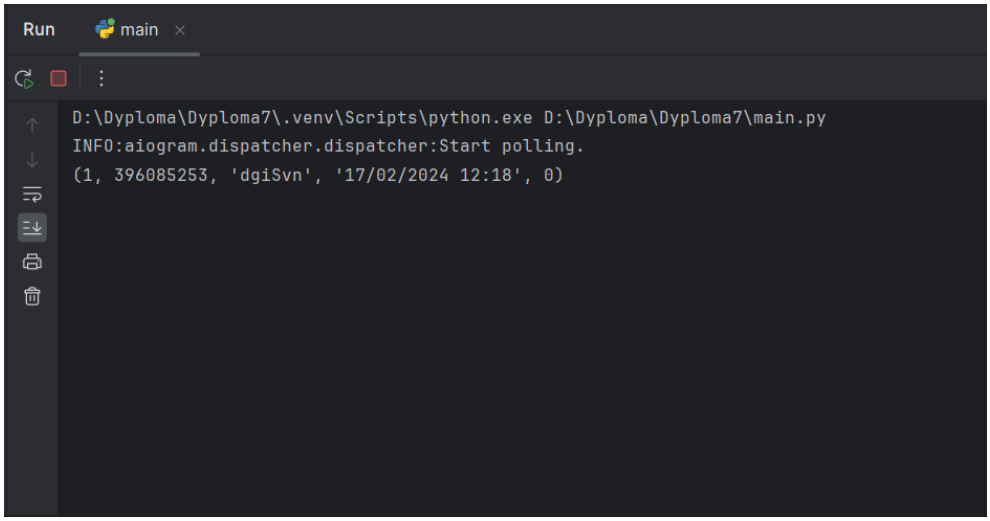


Рисунок 4.2 – Реєстрація користувача у базі даних за Telegram-тегом

Слід зазначити, при наявності такого користувача у базі даних фіксація не буде проведена для уникнення непотрібного заповнення бази даних, що може призвести до певних втрат даних, некоректних відображень або навіть збоїв у роботі певних функції Telegram-боту.

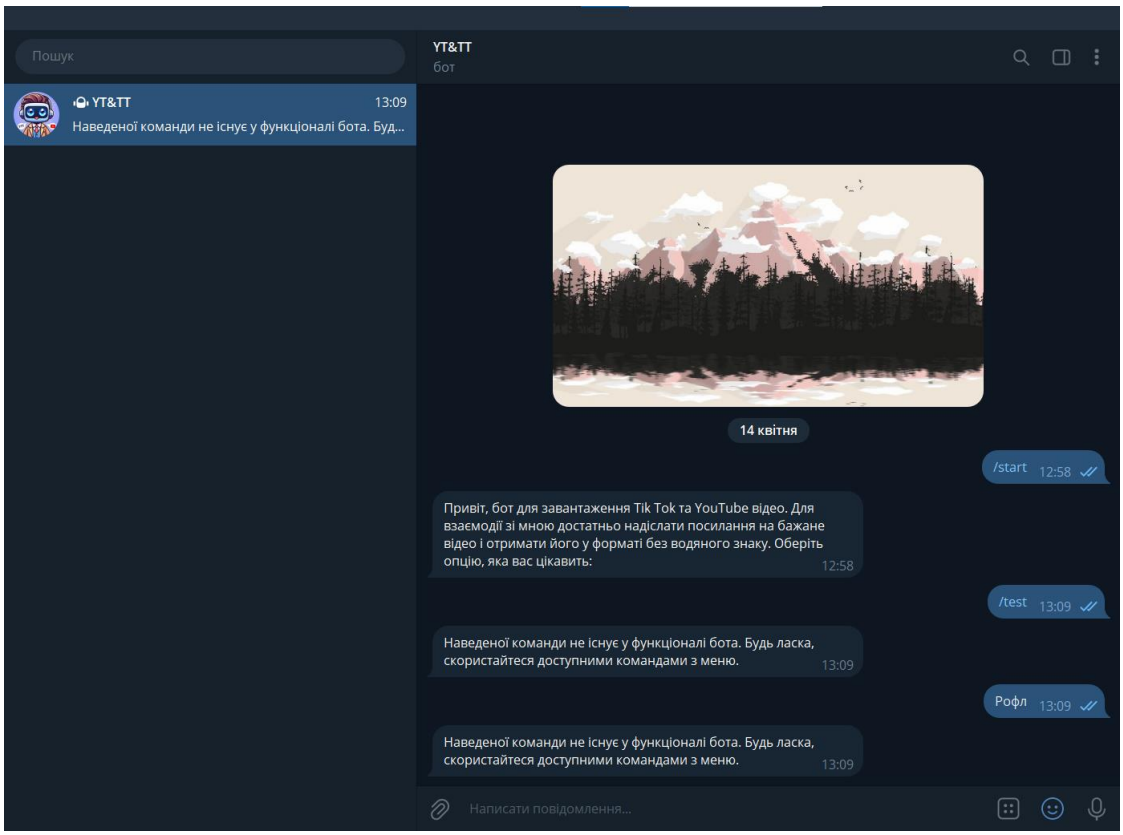


Рисунок 4.3 – Відповідь бота на некоректні дані

Наступним було прийнято рішення перевірити функціонал команди «/help». На рисунку 4.4 наведено скріншот відповіді боту на команду.

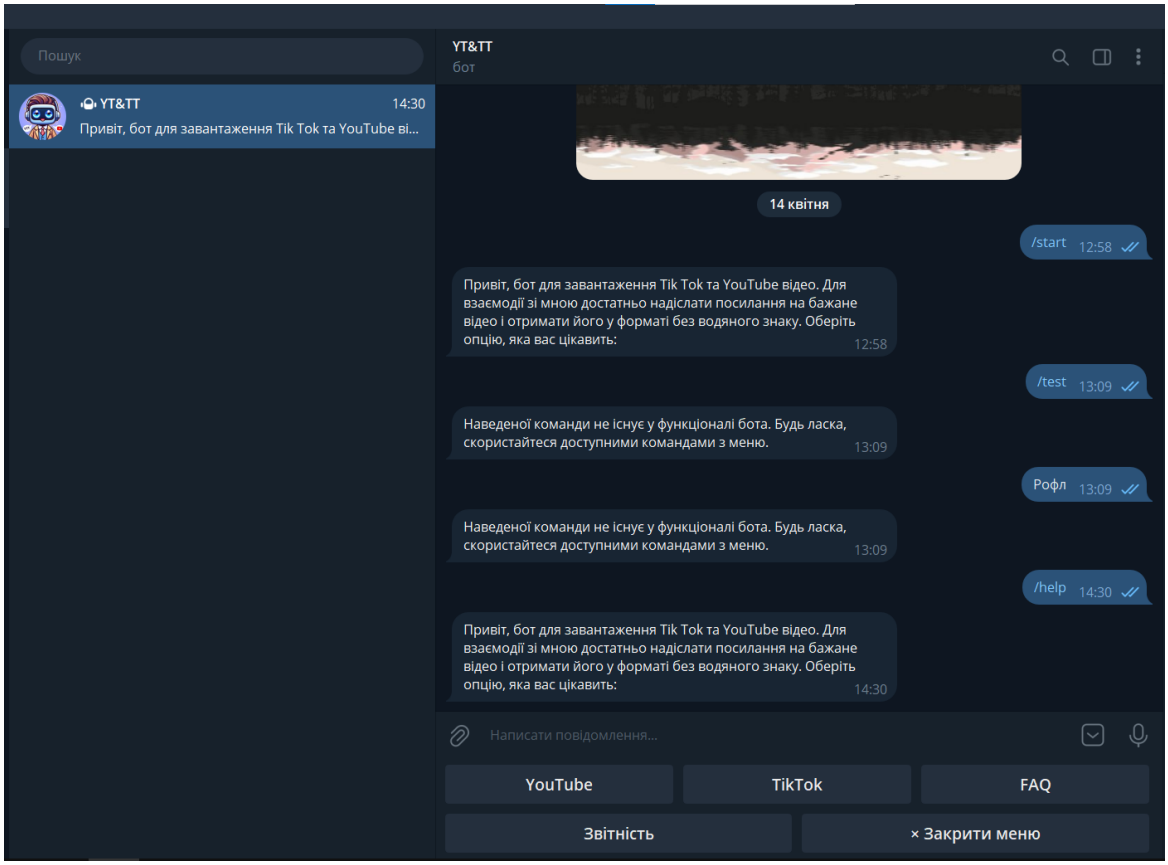


Рисунок 4.4 – Відповідь на команду «/help»

Після перевірки на коректнусть відповіді на неправильні дані можна перейти до тестування окремих команд, таких як «/yt» та «/tt». Вони націлені на більш професійних користувачів. Команди являють собою скорочення від платформ, з яких користувач хоче завантажити відео. Результатом введення цих команд повинна бути відповідь від бота про необхідність надіслати посилання на потрібну платформу, після чого буде здійснено завантаження. На рисунку 4.5 наведено результати тестування обох команд.

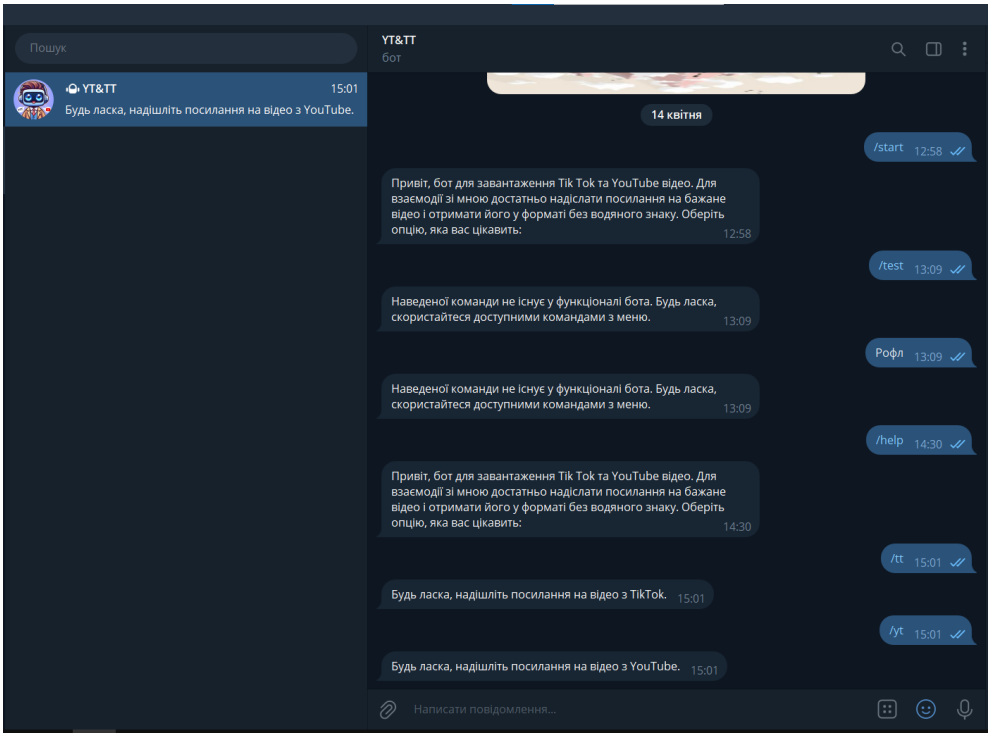


Рисунок 4.5 – Результат успішного введення обох команд

На рисунку 4.6 наведено результат завантаження відео з платформи TikTok, а на рисунку 4.7 наведено результат тестування, проте вже з платформи YouTube.

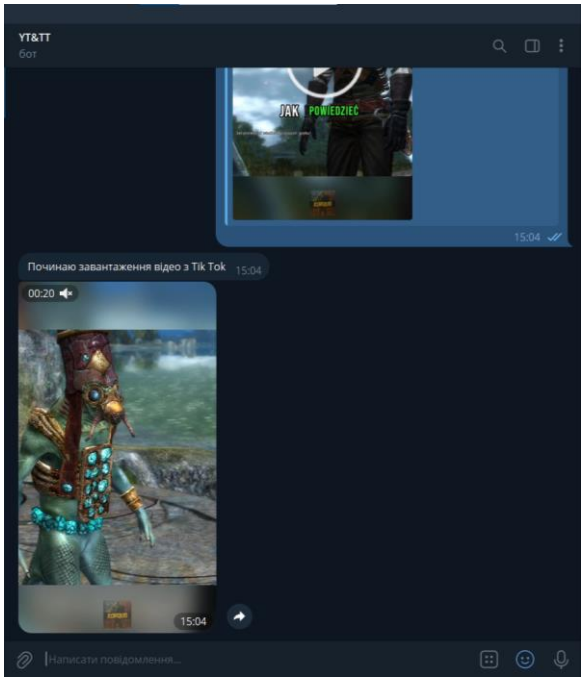


Рисунок 4.6 – Результат завантаження TikTok відео

Видно, що після введення посилання було надіслано інформаційне повідомлення про початок обробки запиту. Через деякий час було надіслано готове завантажене відео вже без водяного знаку сервісу TikTok.

Рисунок 4.7 – Неправильний результат завантаження відео з YouTube

На рисунку 4.7 наведено приклад помилкового завантаження відео. Це означає, що надіслане відео захищене правами автора і для завантаження потребує входження у аккаунт YouTube, що у свою чергу є правомірним захистом для завантаження відео-контенту.

На рисунку 4.8 наведено успішний варіант завантаження відео з YouTube, а саме YouTube Shorts. Такий тип відеофайлу є дуже схожим на TikTok відео, проте його вага не може бути вищою за 50 мб, що не дозволить користувачеві

порушити ліміт і отримати помилку пов'язану з перевищенням ліміту ваги файлу, який може надсилати Telegram-бот.

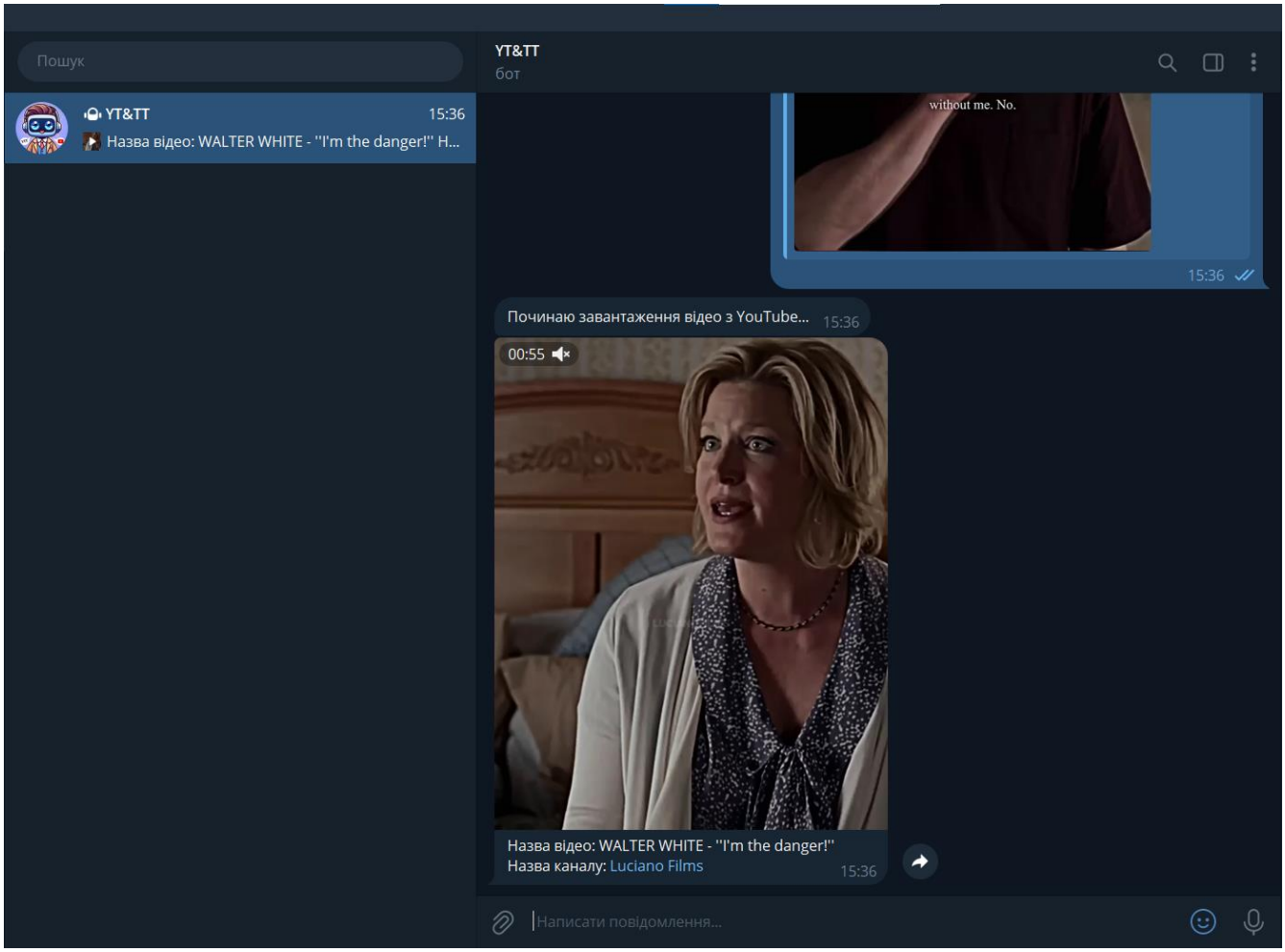


Рисунок 4.8 – Результат завантаження YouTube Shorts

Наступним кроком у тестуванні боту було прийнято рішення протестувати можливість завантаження звичайного відео з YouTube платформи.

Як було вище зазначено, що можливий ще один тип помилки, при якому користувач не може отримати відео, яке було надіслано боту. На рисунку 4.9 наведено саме цю помилку, яка пов'язана з перебільшенням ліміту ваги файлу, так як у безкоштовній версії бота можливість надсилати ботом файли важчі ніж 50 мб заборонено.

На рисунку 4.10 наведено результат успішного завантаження просто відео з YouTube, проте поки що через команду.

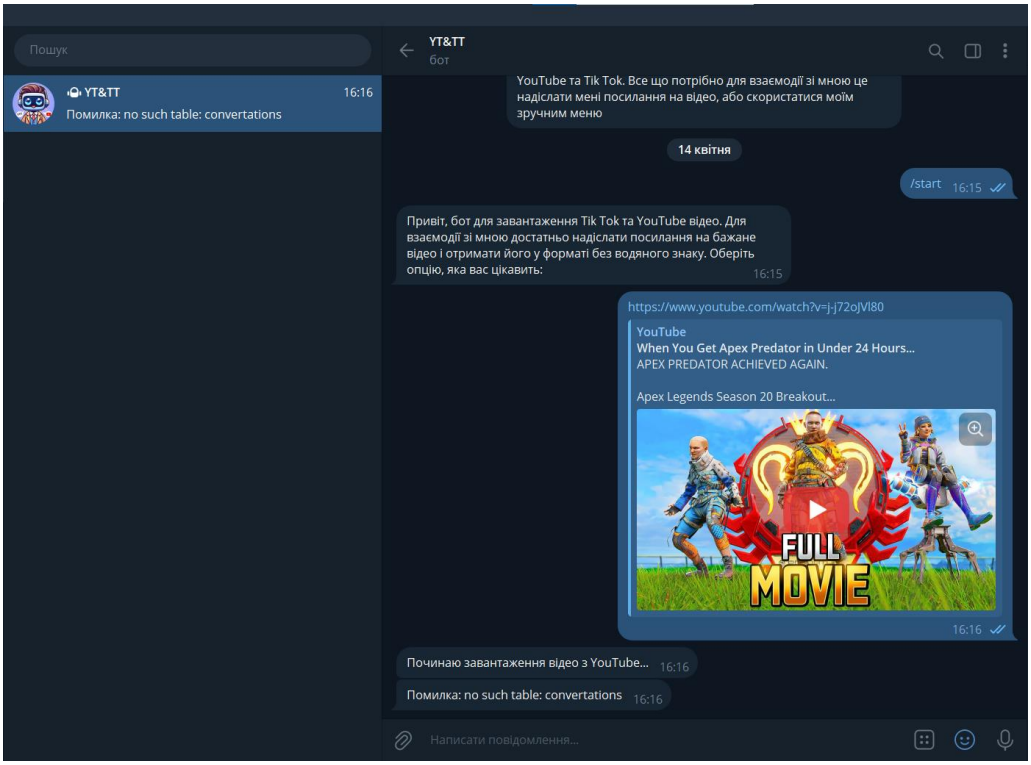


Рисунок 4.9 – Результат неуспішного завантаження відео

Наразі бачимо помилку, що на пряму пов'язана з великим розміром відео, що унеможливлює отраминня такого відео. Це пов'язано з обмеженнями зі сторони Telegram.

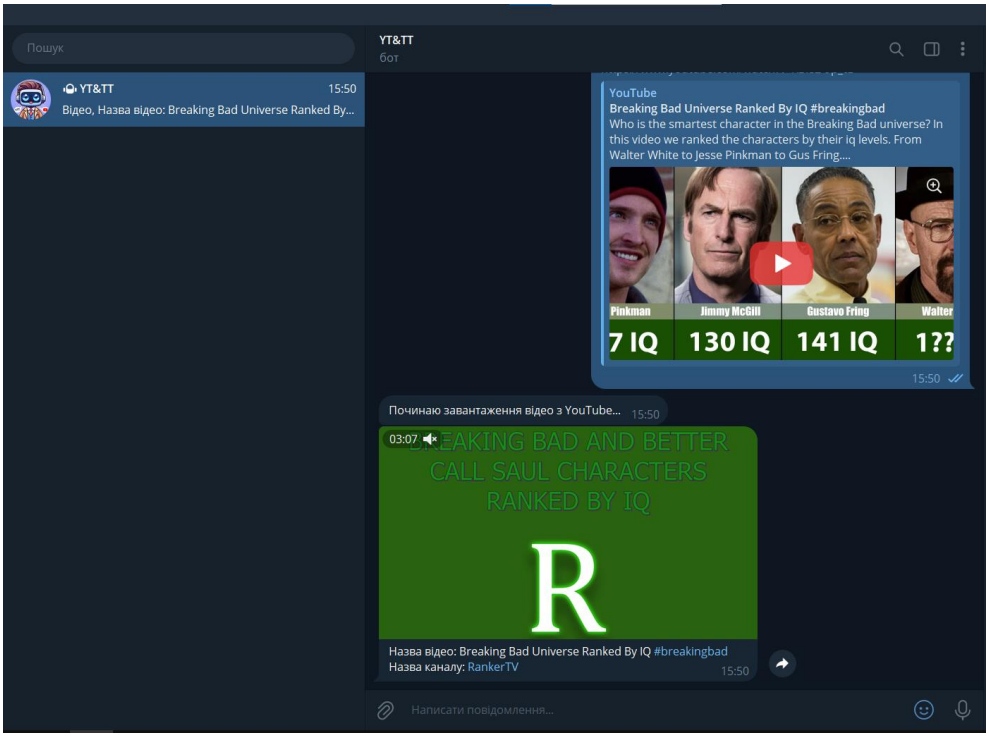


Рисунок 4.10 – Коректне завантаження відео

Далі слід протестувати розроблене інтерактивне меню на функціональність на рисунку 4.11 буде протестовано обидві функції для завантаження відео.

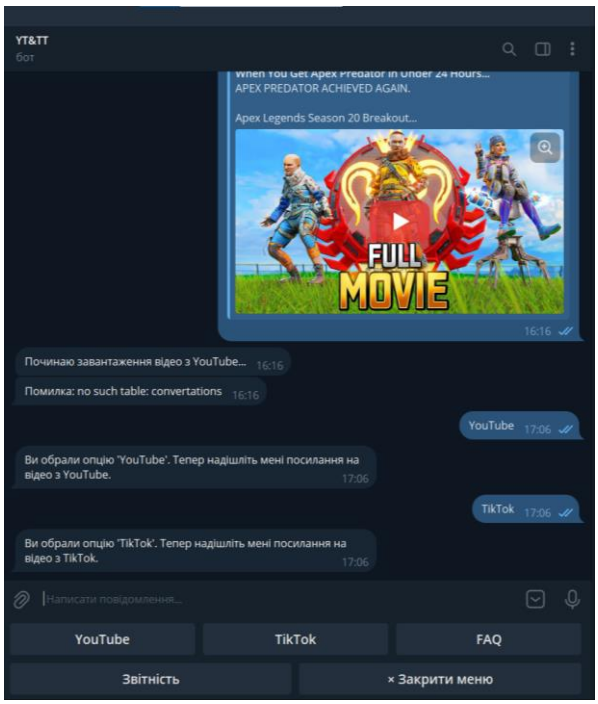


Рисунок 4.11 – Результат тестування кнопок «TikTok» та «YouTube»

У випадку певних помилок або складнощів користувач завжди може викликати FAQ меню де він зможе побачити відповіді на найбільш розповсюджені запитання, що можуть виникати у користувача (див. рис. 4.12).

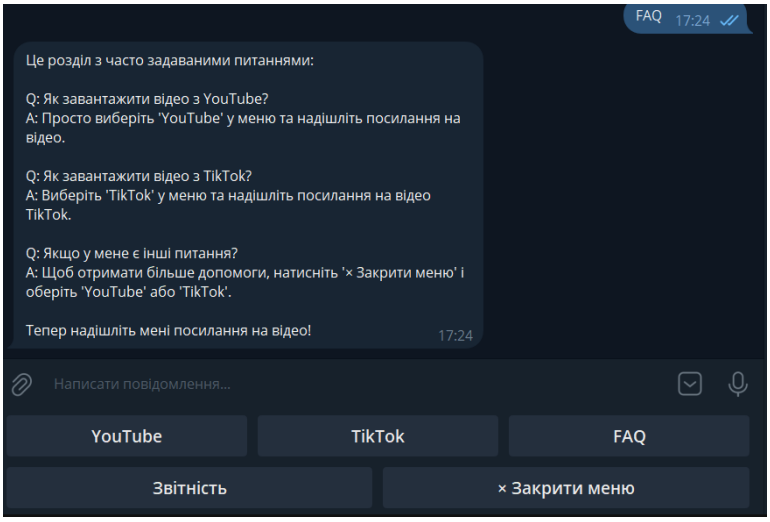


Рисунок 4.12 – Результат тестування кнопки «FAQ»

Остаточним функціональним тестом можна вважати функцію для надання користувачеві файлу-звітності. На рисунку 4.13 наведено результат виклику цієї функції через команду «/report», а також через кнопку звітність.

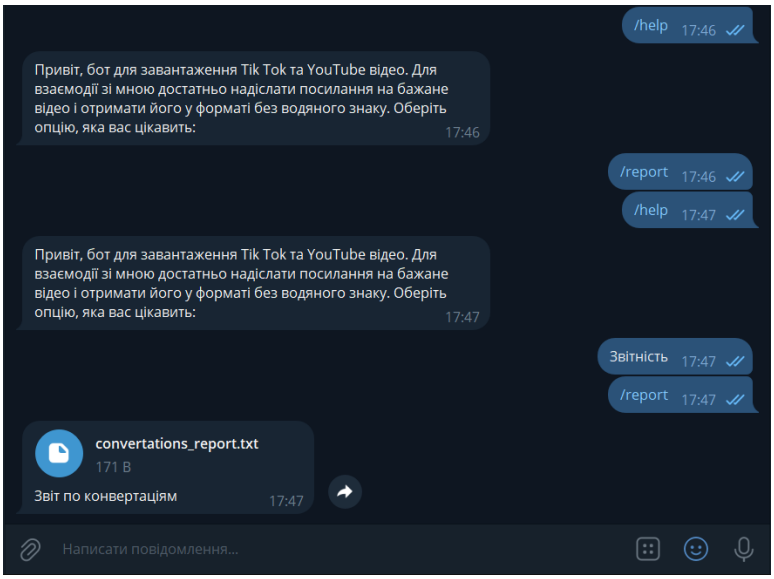


Рисунок 4.13 – Результат тестування функції отримання звітності

Отже у цьому підрозділі було протестовано весь функціонал, що наявний у боті. Усі реалізовані функції відповідають усім поставленим вимогам, а команди і кнопки безперебійно виконують поставлені задачі.

4.2 Розробка інструкції користувача

Розробляючи інструкцію користувача для боту необхідно врахувати, що інструкція повинна бути написана легкою та зрозумілою мовою, уникати технічних термінів або пояснювати їх у простих словах. Також неменш важливим є додавання зображень або скріншотів з метою полегшання розуміння інструкції за допомогою зорової пам'яті.

Отже, для початку користуванням ботом його спочатку потрібно знайти, користувач відкриває Telegram та пише у пошуку назву бота або за тегом «@BanDimBot» (див. рисунок 4.14), після чого обрати саме потрібного бота і можна перейти до розгляду інтерфейсу у самому чаті.

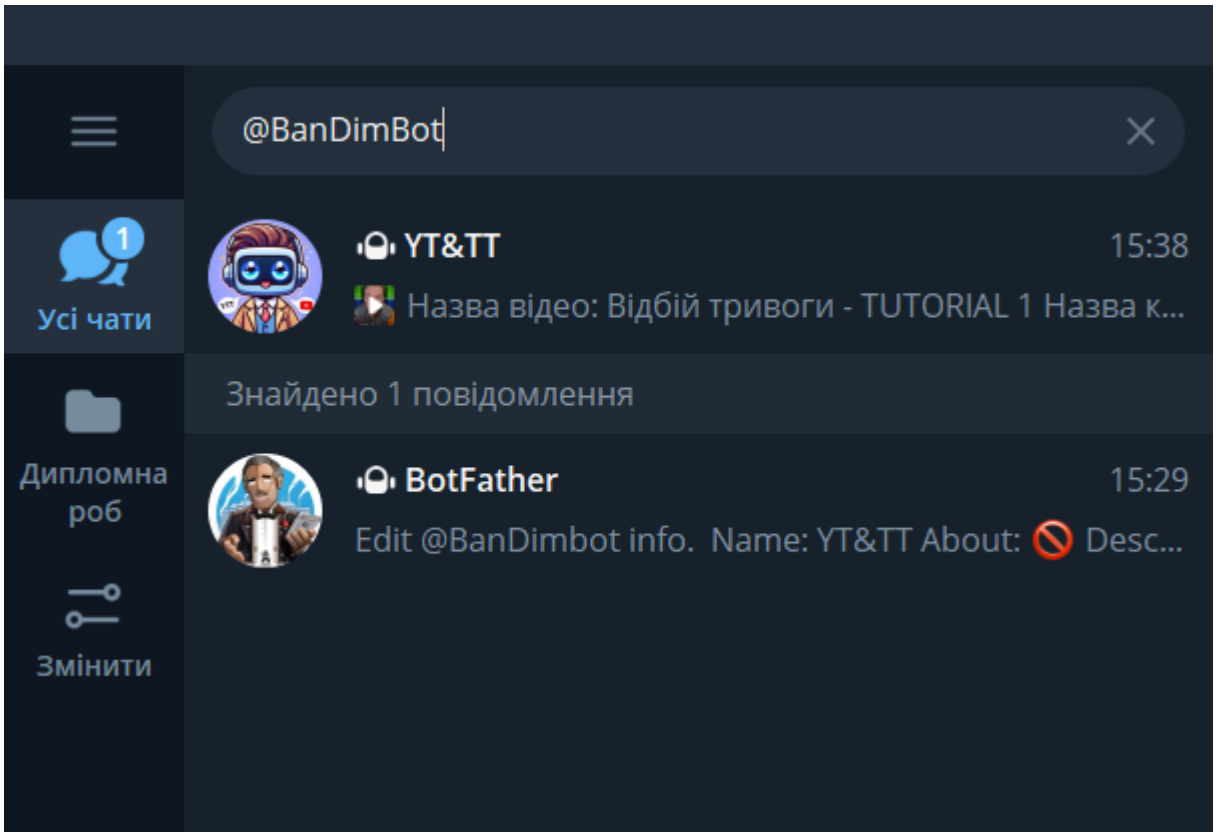


Рисунок 4.14 – Відображення боту у графі пошуку

Після натискання на іконку бота (див. рисунок 4.15) користувач буде перенаправлений у чатове вікно, де відбувається взаємодія з ботом. Це вікно має звичайний інтерфейс, який знайомий більшості користувачів, схожий на той, що використовується у всіх інших месенджерах.

Один з основних відмінностей цього чату від звичайних розмов – це те, що відразу після входу в чат, автоматично надсилається команда «/start» (див. рисунок 4.16), що починає роботу бота. Це означає, що користувач не повинен самостійно вводити цю команду, все відбувається автоматично для його зручності.

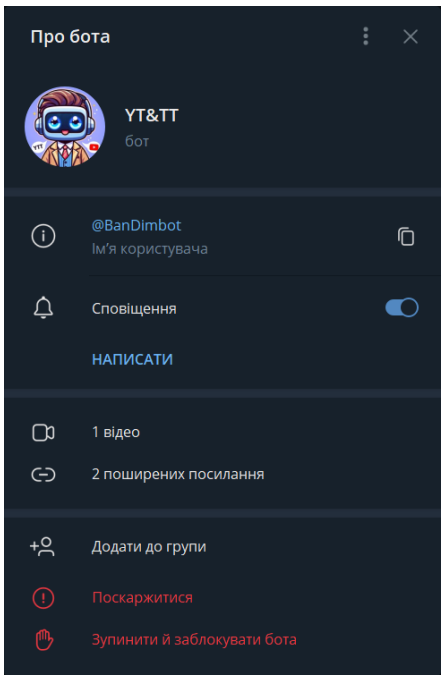


Рисунок 4.15 – Ознайомче вікно бота

Також автоматична авторизація у майбутньому дозволить скористатися функцією звітності, яка формує файл із інформацією про всі проведені операції користувача.

Коли бот починає свою роботу після отримання команди «/start», користувач може отримати доступ до різноманітних функцій та можливостей, що пропонуються ботом.

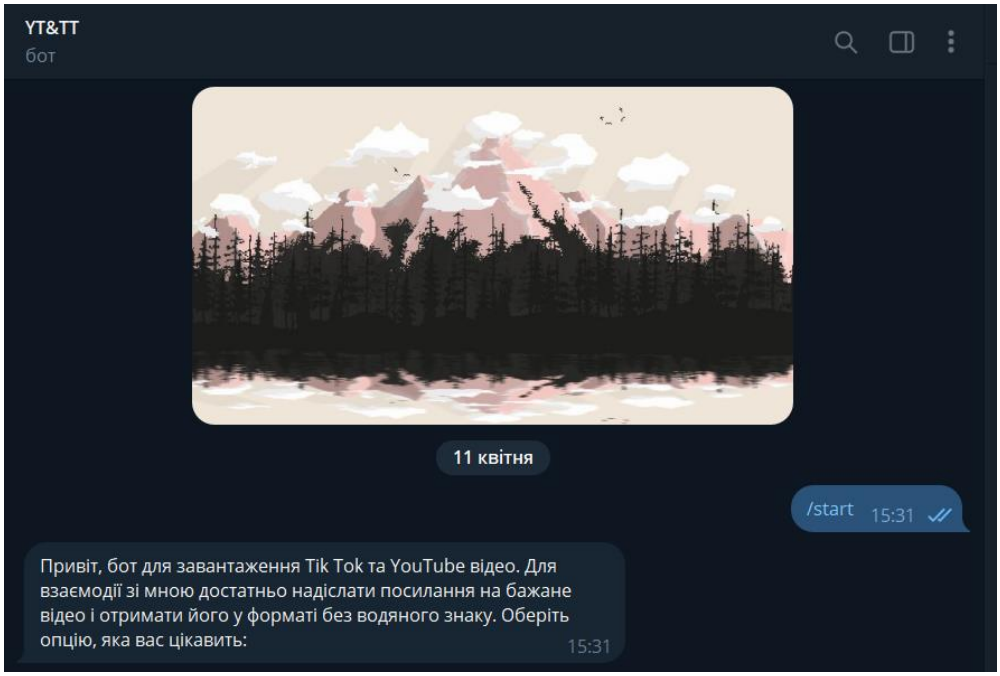


Рисунок 4.16 – Відповідь бота на початок роботи з користувачем

Інтерфейс також дозволяє користувачеві не прописувати команди повністю в ручну і надає змогу через кнопкове меню викликати окремі функції боту. На рисунку 4.17 наведено вигляд меню.

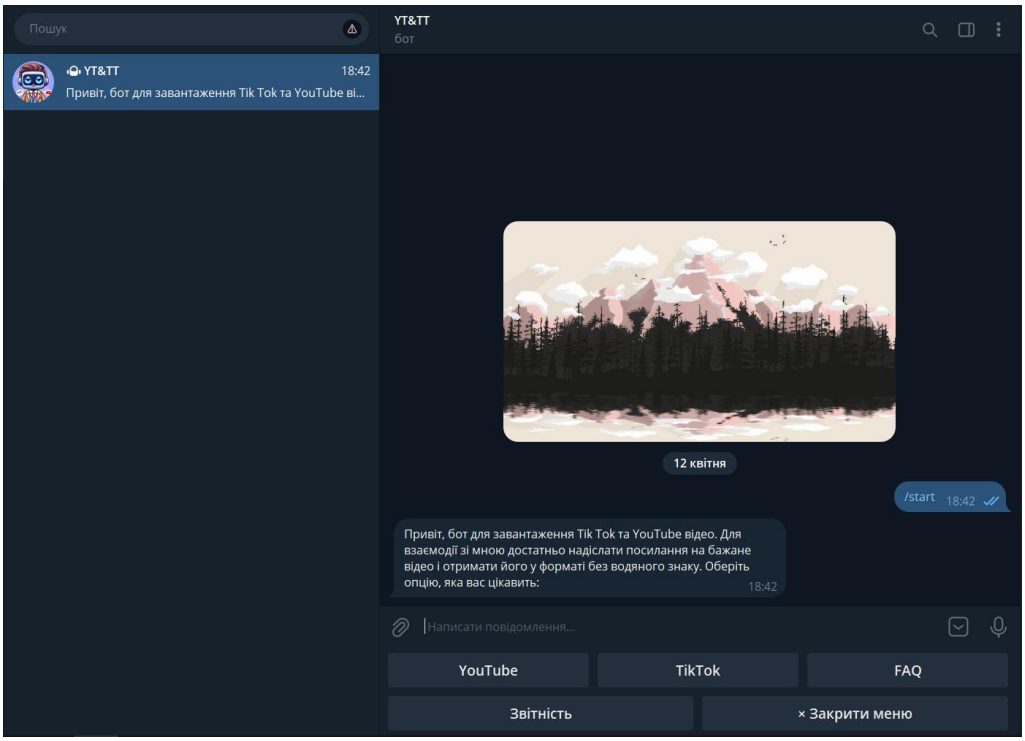


Рисунок 4.17 – Навігаційне меню, що допомагає орієнтуватися у боті

Окрім цього, користувач може в будь-який момент використовувати різні команди чату для взаємодії з ботом. Наприклад, команда «/help» може допомогти отримати список доступних команд і їх призначення.

На рисунку 4.18 наведено FAQ відповідь для користувача. Вона містить певний фіксований набір порад у випадках, які можуть виникнути у користувача.

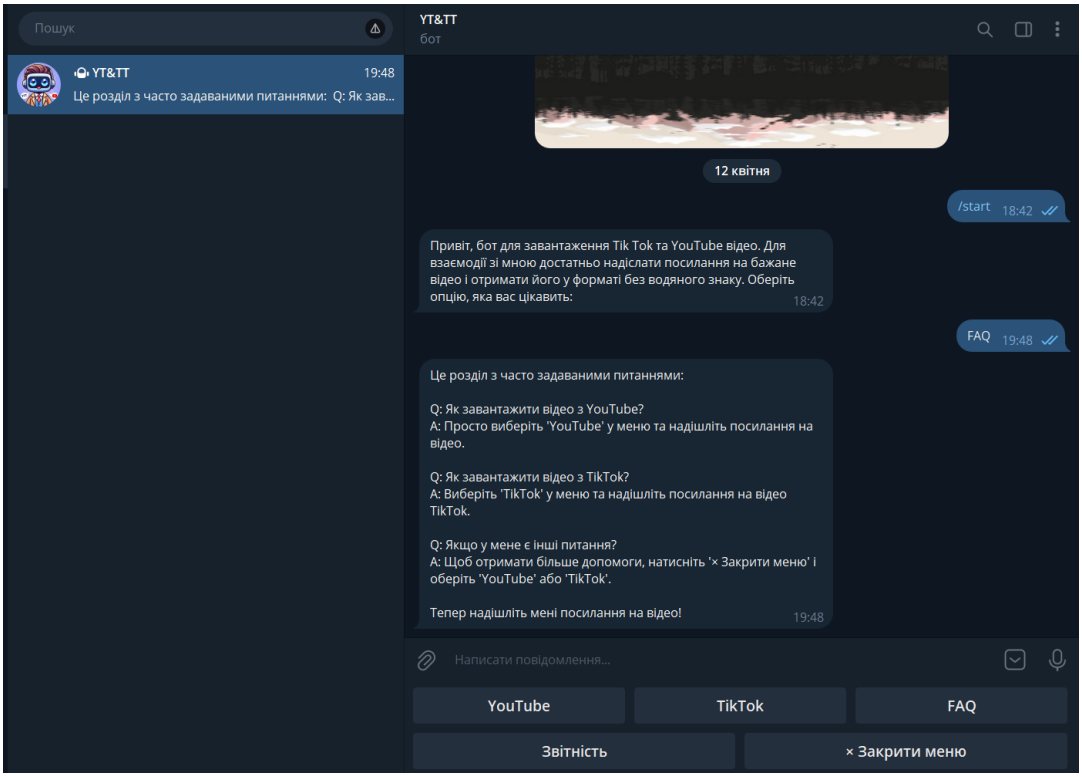


Рисунок 4.18 – Відповідь бота на FAQ запит

Отже, у підрозділі було описано та сформовано інструкцію користувача та продемонстровано окремі елементи інтерфейсу за допомогою скріншотів. Такий підхід дозволяє значно полегшити процес використання боту для користувачів будь-якого рівня. Чіткі і зрозумілі інструкції разом із візуальними демонстраціями сприяють підвищенню продуктивності та ефективності користувачів, зменшуючи можливість помилок та покращуючи їх загальний досвід використання Telegram-боту.

4.3 Формування технічних вимог

Формування технічних вимог передбачає визначення конфігурацій для запуску програмного продукту [22].

Перш за все, правильно підібрана конфігурація може забезпечити оптимальну швидкість та продуктивність роботи програми. Наприклад, визначення необхідних ресурсів пам'яті та потужності процесора дозволяє уникнути затримок у виконанні завдань та забезпечити швидку відповідь на запити користувачів. Особливо це стосується машини, на якій буде відбуватися хостинг боту.

Крім того, правильно сконфігурована програма може бути більш надійною та стійкою до виникнення помилок або аварій. Наприклад, налаштування параметрів безпеки та мережевого з'єднання може зменшити ймовірність вразливостей та забезпечити захист від зловмисників.

Деталі щодо мінімальної та рекомендованої конфігурації серверного комп’ютера можна знайти в таблицях 4.1 та 4.2.

Таблиця 4.1 – Мінімальна конфігурація:

Тип процесора	Будь-який сучасний процесор, який підтримує Python 3.7+
Об’єм оперативної пам’яті	2 ГБ
Місце на жорсткому диску	10 ГБ
Операційна система	Будь-яка операційна система, на якій можна запустити Python 3.7+
Швидкість інтернету	Мінімум 1 Мбіт/с для взаємодії з мережею

При цьому типі застосунку великого значення набуває місце на жорсткому диску. Цей показник має прямий вплив на обсяг відеоматеріалів, який користувач може зберігати під час використання функціоналу боту.

Таблиця 4.2 – Рекомендована конфігурація:

Тип процесора	Intel Core i5
Об’єм оперативної пам’яті	4 ГБ або більше
Місце на жорсткому диску	20 ГБ або більше
Операційна система	Windows 10, macOS
Швидкість інтернету	Мінімум 5 Мбіт/с для швидкого завантаження великих відеофайлів і відправки їх користувачам

1) Бот використовує Python для взаємодії з API TikTok і завантаження відео з YouTube. Це не вимагає дуже потужного процесора, тому будь-який сучасний процесор, який підтримує Python 3.7+, буде достатнім.

2) Для бота, який обробляє текстові повідомлення і відправляє відео, мінімальний обсяг оперативної пам’яті може бути 2 ГБ. Але рекомендується мати 4 ГБ або більше для плавної роботи.

3) Мінімальне місце на диску – це близько 10 ГБ для встановлення і запуску бота. Для зберігання відправлених файлів і тимчасових даних рекомендується 5 ГБ або більше.

4) Бот може працювати на будь-якій операційній системі, на якій можна запустити Python 3.7+.

5) Мінімальна швидкість інтернету повинна бути не менше 1 Мбіт/с для взаємодії з мережею. Але для завантаження великих відеофайлів і їх відправки користувачам рекомендується мати швидкість не менше 5 Мбіт/с.

Ці вимоги встановлені з урахуванням того, що бот виконує відносно прості операції з обробки відео і взаємодії з мережею. Їх можна вважати досить мінімальними для ефективної роботи бота, але рекомендовані параметри можуть забезпечити плавну роботу при великій кількості запитів і великих відеофайлах.

4.4 Висновки

У четвертому розділі було проведено тестування програмних модулів Telegram-боту для завантаження TikTok та YouTube відео. Було проаналізовано та створено мінімальні та рекомендовані системні конфігурації.

Було розроблено інструкцію користувача, що у супроводі з візуальними прикладами, роблять використання боту легким для користувачів будь-якого користувача. Всі модулі відповідають поставленій задачі і чітко виконують їх.

ВИСНОВКИ

У бакалаврській дипломній роботі було проаналізовано Telegram-боти та їх набираючу популярність в Україні. Вони стають потужним інструментом для взаємодії з аудиторією, автоматизації процесів та створення інтерактивних сервісів.

Результати аналізу підтвердили високий потенціал Telegram-ботів у сферах завантаження відео-контенту, збереження документації та юзабіліті для користувача, що підкреслює їх значущість як важливого інструменту в сучасному цифровому середовищі.

Бакалаврська дипломна робота оформлена згідно методичних вказівок[21].

У бакалаврській дипломній роботі було виконано наступні завдання:

1) розроблено методи роботи системи;
2) розроблено загальний алгоритм системи;
3) розроблено загальну структуру бази даних за допомогою SQLite3;
4) розроблено функіонал для взаємодії користувача з Telegram-ботом;
5) реалізовано функціонал для адміністрування кількості викликів з боку користувача;
6) здійснено тестування загальної системи згідно поставлених задач.

У першому розділі бакалаврської дипломної роботи було розглянуто стан Telegram-ботів та їх актуальність. Проведено порівняльний аналіз аналогів Telegram-ботів, які дозволяють завантажувати відео з платформ TikTok та Youtube без водяного знаку. Перелічено відомі аналоги зазначивши їхні переваги та недоліки. Було відзначено, що кращим рішенням для цього проєкту буде розробка власного Telegram-бота для готової платформи Telegram з використанням її API та Python для програмування.

У другому розділі було виконано детальний аналіз вхідних даних програмного продукту. На його основі була розроблена модель роботи за допомогою UML діаграми, яка відображає основні процеси та взаємозв'язки в

програмі. Також у розділі було створено основний алгоритм роботи Telegram-боту, який визначає послідовність дій для обробки користувацьких запитів.

У третьому розділі було обґрунтовано вибір мови програмування та методів для розробки системи до Telegram-боту. Проаналізувавши існуючі мови програмування було прийнято рішення використовувати Python для реалізації поставленого завдання. Також було наведено приклади з реалізованими функціями, як у основному так і у коді бази даних. Реалізовані методи і реалізують функціонал, який необхідний для зручного використання боту у Telegram.

Додатково у третьому розділі було проведено візуалізацію деяких функцій, що були описані у попередніх підрозділах. Це було зроблено за допомогою створення чотирьох блок-схем, які повністю візуалізували описані функції. Ці блок-схеми створено з метою кращого розуміння логіки та послідовності операцій, які відбуваються в програмі. Вони допомагають у визначенні основних кроків та взаємозв'язків між ними, що сприяє зрозумінню роботи програми для розробників та інших зацікавлених сторін.

У четвертому розділі було проведено тестування програмних модулів Telegram-боту для завантаження відео з TikTok та YouTube. Під час тестування було здійснено аналіз та встановлено мінімальні та рекомендовані системні конфігурації для коректної роботи боту. У результаті тестування було підтверджено, що всі модулі відповідають поставленій задачі і функціонують бездоганно.

Крім того, було розроблено інструкцію користувача з покроковим описом роботи розробленого боту. Ця інструкція дозволяє користувачам з легкістю взаємодіяти з ботом та виконувати необхідні дії для завантаження відео з TikTok та YouTube.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Чат-бот базові поняття [Електронний ресурс] – Режим доступу до ресурсу: https://sendpulse.ua/knowledge-base/chatbot/telegram/create-telegram-chatbot.

2. Тік Ток і що таке тік ток? [Електронний ресурс] – Режим доступу до ресурсу: https://termin.in.ua/tiktok/.

3. YouTube – що це таке, визначення та поняття [Електронний ресурс] – Режим доступу до ресурсу: https://uk.economy-pedia.com/11030595-youtube.

4. Банарь Д. А. Дослідження алгоритмів для швидкого та зручного завантаження контенту з онлайн платформ/ Д. А. Банарь, Г. О. Черноволик // Матеріали LII Науково-технічної конференції підрозділів Вінницького національного технічного університету, Вінниця, 2024. [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2024/paper/view/20871.

5. Месенджер Telegram [Електронний ресурс] – Режим доступу до ресурсу: https://bizmag.com.ua/telegram/.

6. Розвиток чат-ботів та їх устрій [Електронний ресурс] – Режим доступу до ресурсу: https://sendpulse.ua/support/glossary/chatbot.

7. Найпопулярніші телеграм боти [Електронний ресурс] – Режим доступу до ресурсу: https://psm7.com/uk/news/top-40-boty-telegram-v-ukraine.html.

8. TokMateBot – це?[Електронний ресурс] – Режим доступу до ресурсу: https://github.com/hemantapkh/TokMate?tab=readme-ov-file.

9. Що таке командна оболонка (shell)? [Електронний ресурс] – Режим доступу до ресурсу: https://acode.com.ua/shell-in-linux/.

10. SaveFrom.net сервіс для завантаження відео [Електронний ресурс] – Режим доступу до ресурсу: http://surl.li/qqlpo.

11. Ssstik.io та його функціонал [Електронний ресурс] – Режим доступу до ресурсу: https://ssstik.io/en.

12. Tmate Tik Tok downloader [Електронний ресурс] – Режим доступу до ресурсу: https://tmate.cc/uk.

13. Live Photos iphone [Електронний ресурс] – Режим доступу до ресурсу: https://support.apple.com/uk-ua/104966.

14. PyTube бібліотека Python [Електронний ресурс] – Режим доступу до ресурсу: https://pytube.io/en/latest/.

15. Aiogram – що це за фреємворк? [Електронний ресурс] – Режим доступу до ресурсу: https://drukarnia.com.ua/articles/telegram-boti-na-python-oglyad-p-yati-naikrashikh-freimvorkiv-bibliotek-L7UA7.

16. UML для чого потрібні діаграми процесів [Електронний ресурс] – Режим доступу до ресурсу:https://evergreens.com.ua/ua/articles/uml-diagrams.html.

17. Python, як мова програмування [Електронний ресурс] – Режим доступу до ресурсу:https://freehost.com.ua/ukr/faq/wiki/chto-takoe-jazik-programmirovanija-python/.

18. Java, спільнота програмістів [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/articles/how-to-learn-java/

19. JavaScript, як мова програмування веб-застосунків [Електронний ресурс] – Режим доступу до ресурсу: https://cases.media/article/sho-take-javascript

20. C++ – перевірений варіант програміста [Електронний ресурс] – Режим доступу до ресурсу: https://acode.com.ua/uroki-po-cpp/

21. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів студентів спеціальності 121 "Інженерія програмного забезпечення" / Уклад. О. Н. Романюк, Г. О. Черноволик – Вінниця: ВНТУ, 2022. – 52с.

ДОДАТКИ

Додаток А – Технічне завдання

73

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
д.т.н., проф. О. Н. Романюк
" 12 " березня 2024 р.

Технічне завдання
на бакалаврську дипломну роботу
«Розробка Telegram-боту для завантаження TikTok і Youtube відео з використанням PyTube, Aiogram, RapidApi»
за спеціальністю 121 – Інженерія програмного забезпечення

Керівник бакалаврської дипломної роботи:
к.т.н., доц. каф. ПЗ Г.О. Черноволик
" 12 " березня 2024 р.

Виконав:
студент гр. 4ПІ-20Б Д.А. Банарь
" 12 " березня 2024 р.

Вінниця – 2024 року

1. Найменування та галузь застосування

Бакалаврська дипломна робота: «Розробка Telegram-боту для завантаження TikTok і Youtube відео з використанням PyTube, Aiogram, RapidApi».

Галузь застосування – інженерія ботів.

2. Підстава для розробки.

Підставою для виконання бакалаврської дипломної роботи (БДР) є індивідуальне завдання на БДР та наказ №80 від «11» березня 2024 р. ректора ВНТУ про закріплення тем БДР.

3. Мета та призначення розробки.

Мета бакалаврської дипломної роботи полягає у підвищенні якості завантаження відео-контенту з платформ TikTok та YouTube за рахунок розробки і використання спеціалізованого Telegram-бота з використанням PyTube, Aiogram, RapidApi, що дозволяє завантажувати відео без водяних знаків.

Призначення роботи – розробка Telegram-боту, що зможе завантажувати відео з TikTok, YouTube Shorts, а також невеликі відео з безпосередньо самого YouTube.

4. Вихідні дані для проведення НДР

Вихідні дані для розробки є індивідуальне завдання на бакалаврську дипломну роботу на розробку Telegram-боту з визначеним функціоналом.

5. Технічні вимоги

Вхідні дані – посилання на відео-ряд, що користувач прагне отримати; вихідні дані – готовий відео-ряд з видаленими водяними знаками визначеної платформи.

6. Конструктивні вимоги.

Бот повинен відповідати всім стандартам якості, включаючи юзабіліті інтерфейсу з подальшим клієнтно-орієнтованим обслуговуванням. Графічна та текстова документація повинна відповідати ДСТУ.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

– пояснювальна записка до бакалаврської дипломної роботи;

– технічне завдання;

– лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

Для забезпечення якості, сумісності та безпеки програмного забезпечення при розробці програмних засобів було дотримано уніфікації та державних стандартів України (ДСТУ).

9. Стадії та етапи розробки:

№	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи
1	Аналіз стану Telegram-ботів та їх актуальності	14.03.24 – 19.03.24
2	Аналіз аналогів та проблематики розроблюваного застосунку	20.03.24 – 25.03.24
3	Розробка модуля для взаємодії користувача з базою даних	26.03.24 – 10.05.24
4	Розробка модуля для завантаження відеоряду з обраної платформи	11.05.24 – 23.05.24
5	Оформлення матеріалів до захисту БДР	23.05.24 – 30.05.24

10. Порядок контролю та прийняття.

Виконання етапів бакалаврської дипломної роботи контролюється керівником згідно з графіком виконання роботи. Прийняття бакалаврської дипломної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком.

Додаток Б – Протокол перевірки бакалаврської дипломної роботи на наявність текстових запозичень

77

ПРОТОКОЛ
ПЕРЕВІРКИ БАКАЛАВРСЬКОЇ ДИПЛОМНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: розробка Telegram-боту для завантаження TikTok і Youtube відео з використанням PyTube, Aiogram, RapidApi

Тип роботи: БДР

Підрозділ : кафедра програмного забезпечення, ФІТКІ

Науковий керівник: к.т.н., доц. кафедри ПЗ Черноволик Галина Олександрівна

Оригінальність	97,8 %
Схожість	2,2 %

Аналіз звіту подібності

■ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку Черноволик Г. О.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Автор роботи Банарь Д.А.

Керівник роботи Черноволик Г. О.

Додаток В – Лістинг програми

```
import os
import asyncio
import logging
import sys
from aiogram import Bot, Dispatcher, types
from aiogram.utils.exceptions import NetworkError
from pytube import YouTube
from pathlib import Path
import requests
from moviepy.editor import VideoFileClip
import database
from pytube.exceptions import RegexMatchError
from aiogram.utils.markdown import escape_md



TOKEN = "7131757882:AAE14usWNLIirV3PZOEXzW-llOtRGEQ16qE"
bot = Bot(token=TOKEN)
dp = Dispatcher(bot)
logging.basicConfig(level=logging.INFO, stream=sys.stdout)

# Функція для завантаження відео з TikTok
async def download_tiktok_video(link, message):
    """
        Ця функція виконує завантаження відео з TikTok. Вона використовує API,
        що прибирає водяний знак з відео, та надсилає його користувачеві.

        Параметри:
        - link: посилання на відео TikTok
```

- message: об'єкт повідомлення, яке містить дані про користувача, що надсилає запит

```
    """
    msg = await message.answer("Будь ласка, зачекайте, обробляється ваш запит!")
    url = "https://tiktok-video-no-watermark2.p.rapidapi.com/"
    querystring = {"url": link, "hd": "1"}
    headers = {
        "X-RapidAPI-Key":
"5ba57866cbmshfb5cd3ea69e5122p19232fjsna8c0da5682d6",
        "X-RapidAPI-Host": "tiktok-video-no-watermark2.p.rapidapi.com"
    }
    response = requests.get(url, headers=headers, params=querystring)
    try:
        video_link = response.json()['data']['play']
    except KeyError:
        await msg.edit_text('Недійсне посилання. Будь ласка, надішліть дійсне
посилання на відео TikTok!')
        database.add_convertation(message.from_user.id, 'Помилка')
        return

    await msg.edit_text("Починаю завантаження відео з Tik Tok")
    await bot.send_video(message.chat.id, types.InputFile.from_url(video_link))
    database.add_convertation(message.from_user.id, 'Виконано')


async def download_video(url, message):
    """
    Функція отримує посилання на відео з YouTube та об'єкт повідомлення,
```

який містить дані про користувача, що надсилає запит.

Вона спочатку перевіряє, чи не порожнє посилання, а потім використовує бібліотеку pytube для завантаження відео.

Якщо відео велике (понад 50 МБ), воно розбивається на частини та надсилається користувачеві окремими частинами.

Після завершення процесу завантаження відео, функція надсилає отримане відео користувачеві у чаті.

Параметри:

- url: посилання на відео YouTube

- message: об'єкт повідомлення, яке містить дані про користувача, що надсилає запит

```
"""
try:
    # Завантаження відео з YouTube
    yt = YouTube(url)
    video_title = sanitize_filename(yt.title)
    video_path = Path(f'{message.chat.id}/{message.chat.id}_{video_title}.mp4')

    # Перевірка розміру файлу
    file_size_bytes = yt.streams.get_highest_resolution().filesize
    if file_size_bytes > 50 * 1024 * 1024:  # Перевірка на великий файл
        await message.answer("Відео занадто велике, розділяю на частини...")
    yt.streams.get_highest_resolution().download(video_path.parent, video_path.name)
    database.add_convertation(message.from_user.id, 'Виконано')

    await asyncio.sleep(3)  # Затримка на 3 секунди
```

```
        # Розділення відео на частини
        if file_size_bytes > 50 * 1024 * 1024:  # Якщо файл великий, розділити на частини
            video_clip = VideoFileClip(str(video_path))
            duration = video_clip.duration
            chunk_duration = 2.3 * 60.0  # 10 хвилин на частину
            num_chunks = int(duration // chunk_duration) + 1
            for i in range(num_chunks):
                start_time = i * chunk_duration
                end_time = min((i + 1) * chunk_duration, duration)
                chunk_path = f"{video_path.parent}/{video_title}_{i}.mp4"
                chunk_clip = video_clip.subclip(start_time, end_time)
                chunk_clip.write_videofile(chunk_path)
                with open(chunk_path, 'rb') as video_file:
                    await bot.send_video(message.chat.id, types.InputFile(video_file),
                                         caption=f"Частина {i + 1} з {num_chunks}\nНазва відео: {yt.title}\nНазва каналу: [{yt.author}]({yt.channel_url})",
                                         parse_mode="Markdown")
                os.remove(chunk_path)
        else:
            # Відправлення цілого відео
            with open(video_path, 'rb') as video_file:
                await bot.send_video(message.chat.id, types.InputFile(video_file),
                                     caption=f"Назва відео: {yt.title}\nНазва каналу: [{yt.author}]({yt.channel_url})",
                                     parse_mode="Markdown")
        os.remove(video_path)
    except RegexMatchError:
```

```
        await message.answer("Помилка: неправильне посилання на відео.")


def sanitize_filename(filename):
    """
    Ця функція приймає назву файлу та повертає її замінивши неприпустимі символи на допустимі.

    Параметри:
    - filename: назва файлу, яку потрібно змінити

    Повертає:
    - нова назва файлу зі зміненими символами

    """
    return ''.join(char if char.isalnum() or char in ['-', '_'] else '_' for char in filename)

@dp.message_handler(commands=["start"])
async def start_message(message: types.Message):
    """
    Цей обробник відправляє користувачеві стартове повідомлення та відображає меню.

    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача, що відправив команду

    """
```

```
    chat_id = message.chat.id
    menu_text = (
        "Привіт, бот для завантаження Tik Tok та YouTube відео. "
        "Для взаємодії зі мною достатньо надіслати посилання на бажане відео і отримати його у форматі без водяного знаку. "
        "Оберіть опцію, яка вас цікавить:"
    )
    await bot.send_message(chat_id, menu_text, reply_markup=get_menu_keyboard())

    database.add_user(message.from_user.id, message.from_user.username)

@dp.message_handler(commands=["help"])
async def start_message(message: types.Message):
    """
    Цей обробник відправляє користувачеві стартове повідомлення та відображає меню.

    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача, що відправив команду

    """
    chat_id = message.chat.id
    menu_text = (
        "Привіт, бот для завантаження Tik Tok та YouTube відео. "
        "Для взаємодії зі мною достатньо надіслати посилання на бажане відео і отримати його у форматі без водяного знаку. "
        "Оберіть опцію, яка вас цікавить:"
```

```
    )
    await bot.send_message(chat_id, menu_text,
reply_markup=get_menu_keyboard())

def get_menu_keyboard():
    """
    Ця функція створює клавіатуру з опціями меню та повертає її.
    - клавіатура з опціями меню
    """
    keyboard = types.ReplyKeyboardMarkup(resize_keyboard=True)
    keyboard.add(
        types.KeyboardButton("YouTube"),
        types.KeyboardButton("TikTok"),
        types.KeyboardButton("FAQ"),
        types.KeyboardButton("× Закрити меню")
    )
    return keyboard

@dp.message_handler(lambda message: message.text == "YouTube",
content_types=types.ContentTypes.TEXT)
async def youtube_option(message: types.Message):
    # Обробник для опції YouTube
    """
    Цей обробник відображає повідомлення про обрану опцію YouTube.
    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача та обрану
опцію
    """
```

```
    await message.answer("Ви обрали опцію 'YouTube'. Тепер надішліть мені посилання на відео з YouTube.")

@dp.message_handler(lambda message: message.text == "TikTok", content_types=types.ContentTypes.TEXT)
async def tiktok_option(message: types.Message):
    # Обробник для опції TikTok
    await message.answer("Ви обрали опцію 'TikTok'. Тепер надішліть мені посилання на відео з TikTok.")

@dp.message_handler(lambda message: message.text == "FAQ", content_types=types.ContentTypes.TEXT)
async def faq_option(message: types.Message):
    """
    Цей обробник надсилає користувачеві часто задавані питання.

    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача та обрану опцію

    """
    faq_text = (
        "Це розділ з часто задаваними питаннями:\n\n"
        "Q: Як завантажити відео з YouTube?\n"
        "A: Просто виберіть 'YouTube' у меню та надішліть посилання на відео.\n\n"
        "Q: Як завантажити відео з TikTok?\n"
        "A: Виберіть 'TikTok' у меню та надішліть посилання на відео TikTok.\n\n"
        "Q: Якщо у мене є інші питання?\n"
```

```
        "А: Щоб отримати більше допомоги, натисніть '× Закрити меню' і оберіть 'YouTube' або 'TikTok'.\n\n"
        "Тепер надішліть мені посилання на відео!"
    )
    await bot.send_message(message.chat.id, faq_text, reply_markup=get_menu_keyboard())

@dp.message_handler(lambda message: message.text == "× Закрити меню", content_types=types.ContentTypes.TEXT)
async def close_menu(message: types.Message):
    # Обробник для закриття меню
    """
    Цей обробник закриває меню.
    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача та обрану опцію
    """
    await message.answer("Меню закрито.", reply_markup=types.ReplyKeyboardRemove())

@dp.message_handler(commands=["yt"])
async def yt_command(message: types.Message):
    # Обробник для команди /yt
    await message.answer("Будь ласка, надішліть посилання на відео з YouTube.")

@dp.message_handler(commands=["tt"])
async def tt_command(message: types.Message):
    # Обробник для команди /tt
    await message.answer("Будь ласка, надішліть посилання на відео з TikTok.")
```

```
@dp.message_handler(commands=["report"])
async def report_command(message: types.Message):
    # Обробник для команди /report
    """
    Цей обробник генерує звіт про конвертації та відправляє його користувачеві у вигляді документу.
    Параметри:
    - message: об'єкт повідомлення, яке містить дані про користувача, що відправив команду
    """
    convertations = database.get_convertations()
    filename = "convertations_report.txt"
    with open(filename, "w", encoding="utf-8") as file:
        file.write(convertations)
    with open(filename, "rb") as report_file:
        await bot.send_document(message.chat.id, report_file, caption="Звіт по конвертаціям")
    os.remove(filename)

@dp.message_handler(lambda message: message.text.startswith("http"), content_types=types.ContentTypes.TEXT)
async def handle_video_link(message: types.Message):
    # Обробник для обробки посилання на відео
    """
    Цей обробник відповідає за обробку посилання на відео, яке надсилає користувач.
    Параметри:
```

```
    - message: об'єкт повідомлення, яке містить дані про користувача та посилання на відео
    """
    chat_id = message.chat.id
    url = message.text.strip()

    if not url:
        await message.answer("Посилання порожнє. Будь ласка, надішліть посилання на відео.")
        return

    if 'youtu.be' in url or 'youtube.com' in url:
        await process_youtube(url, message)
    elif 'tiktok.com' in url or 'vm.tiktok.com' in url:
        await process_tiktok(url, message)
    else:
        await message.answer("Підтримуються лише посилання на YouTube та TikTok.")

async def process_youtube(url, message):
    """
    Ця функція відповідає за обробку посилання на відео з YouTube. Вона починає завантаження відео та додає запис про конвертацію до бази даних.
    Параметри:
    - url: посилання на відео YouTube
    - message: об'єкт повідомлення, яке містить дані про користувача
    """
    try:
        await message.answer("Починаю завантаження відео з YouTube...")
```

```
        await download_video(url, message)
        database.add_convertation(message.from_user.id, 'YouTube')
    except Exception as e:
        await message.answer(f"Помилка: {str(e)}")

async def process_tiktok(url, message):
    """
    Ця функція відповідає за обробку посилання на відео з TikTok. Вона починає завантаження відео та додає запис про конвертацію до бази даних.

    Параметри:
    - url: посилання на відео TikTok
    - message: об'єкт повідомлення, яке містить дані про користувача
    """
    chat_id = message.chat.id
    try:
        await message.answer("Починаю завантаження відео з TikTok...")
        await download_tiktok_video(url, message)
        database.add_convertation(message.from_user.id, 'TikTok')
    except Exception as e:
        await message.answer(f"Помилка: {str(e)}")

async def main() -> None:
    # Головна функція для запуску бота
    await dp.start_polling()

if __name__ == "__main__":
    asyncio.run(main())
```

Лістинг database.py

```
import sqlite3
from datetime import datetime
import logging

# Налаштування логування
logging.basicConfig(filename='database.log', level=logging.INFO,
                    format='%(asctime)s - %(levelname)s - %(message)s')

def create_table_convertations():
    connection_convertations = None
    try:
        connection_convertations = sqlite3.connect("tiktok.db")
        cursor = connection_convertations.cursor()
        cursor.execute(
            'CREATE TABLE IF NOT EXISTS convertations (id INTEGER PRIMARY
KEY, telegram_id BIGINT, register_date TEXT, status TEXT)'
        )
        connection_convertations.commit()
        logging.info("Таблиця 'convertations' успішно створена або вже існує.")
    except sqlite3.Error as e:
        logging.error(f"Помилка створення таблиці 'convertations': {e}")
    finally:
        if connection_convertations:
            connection_convertations.close()

def add_convertation(telegram_id, status):
    if not isinstance(telegram_id, int) or not isinstance(status, str):
        logging.error("Невірні типи даних для 'telegram_id' або 'status'.")
```

```
        return

    connection_add = None
    try:
        connection_add = sqlite3.connect("tiktok.db")
        cursor = connection_add.cursor()
        now = datetime.now()
        dt_string = now.strftime("%d/%m/%Y %H:%M")
        cursor.execute('INSERT INTO convertations(telegram_id, register_date, status)
VALUES(?,?,?)',
                       (telegram_id, dt_string, status))
        connection_add.commit()
        logging.info(f"Додано нову конвертацію: Telegram ID {telegram_id}, Статус
{status}.")
    except sqlite3.Error as e:
        logging.error(f"Помилка додавання конвертації: {e}")
    finally:
        if connection_add:
            connection_add.close()

def get_convertations():
    try:
        connection_get = sqlite3.connect("tiktok.db")
        cursor = connection_get.cursor()
        cursor.execute('SELECT * FROM convertations')
        data = cursor.fetchall()
        info = ''
        for i in data:
```

```
            if i[3].strip():  # Перевіряємо, чи статус не є пустим після видалення пробілів
                conversion_type = i[3].strip()
                info += f'ID: {i[0]}, Telegram ID: {i[1]}, Register Date: {i[2]}, Status: {conversion_type}\n'
            else:
                # Видаляємо порожні записи
                cursor.execute("DELETE FROM convertations WHERE id=?", (i[0],))
                connection_get.commit()
                logging.info(f"Видалено порожній запис з ID: {i[0]}")
        return info
    except sqlite3.Error as e:
        logging.error(f"Помилка отримання конвертацій: {e}")
        return "Помилка отримання даних"
    finally:
        connection_get.close()

def create_table_users():
    connection_users = None
    try:
        connection_users = sqlite3.connect("tiktok.db")
        cursor = connection_users.cursor()
        cursor.execute(
            'CREATE TABLE IF NOT EXISTS users (id INTEGER PRIMARY KEY, telegram_id BIGINT, username TEXT, register_date TEXT)'
        )
        connection_users.commit()
        logging.info("Таблиця 'users' успішно створена або вже існує.")
    except sqlite3.Error as e:
```

```
        logging.error(f"Помилка створення таблиці 'users': {e}")
    finally:
        if connection_users:
            connection_users.close()

def add_user(telegram_id, username):
    if not isinstance(telegram_id, int) or not isinstance(username, str):
        logging.error("Невірні типи даних для 'telegram_id' або 'username'.")
        return

    try:
        connection_add_user = sqlite3.connect("tiktok.db")
        cursor = connection_add_user.cursor()
        cursor.execute('SELECT * FROM users WHERE telegram_id = ?', (telegram_id,))
        data = cursor.fetchone()
        if data is not None:
            logging.info(f"Користувач з Telegram ID {telegram_id} вже існує.")
            return
        now = datetime.now()
        dt_string = now.strftime("%d/%m/%Y %H:%M")
        cursor.execute('INSERT INTO users(telegram_id, username, register_date) VALUES(?,?,?)',
                       (telegram_id, username, dt_string))
        connection_add_user.commit()
        logging.info(f"Додано нового користувача: Telegram ID {telegram_id}, Username {username}.")
    except sqlite3.Error as e:
        logging.error(f"Помилка додавання користувача: {e}")
```

```
    finally:
        connection_add_user.close()


def get_users():
    try:
        connection_get_users = sqlite3.connect("tiktok.db")
        cursor = connection_get_users.cursor()
        cursor.execute('SELECT * FROM users')
        data = cursor.fetchall()
        info = ''
        for i in data:
            info += f'Telegram ID: {i[1]}, Username: {i[2]}, Register Date: {i[3]}\n'
        return info
    except sqlite3.Error as e:
        logging.error(f"Помилка отримання користувачів: {e}")
        return "Помилка отримання даних"
    finally:
        connection_get_users.close()
```

Додаток Г – Ілюстративна частина

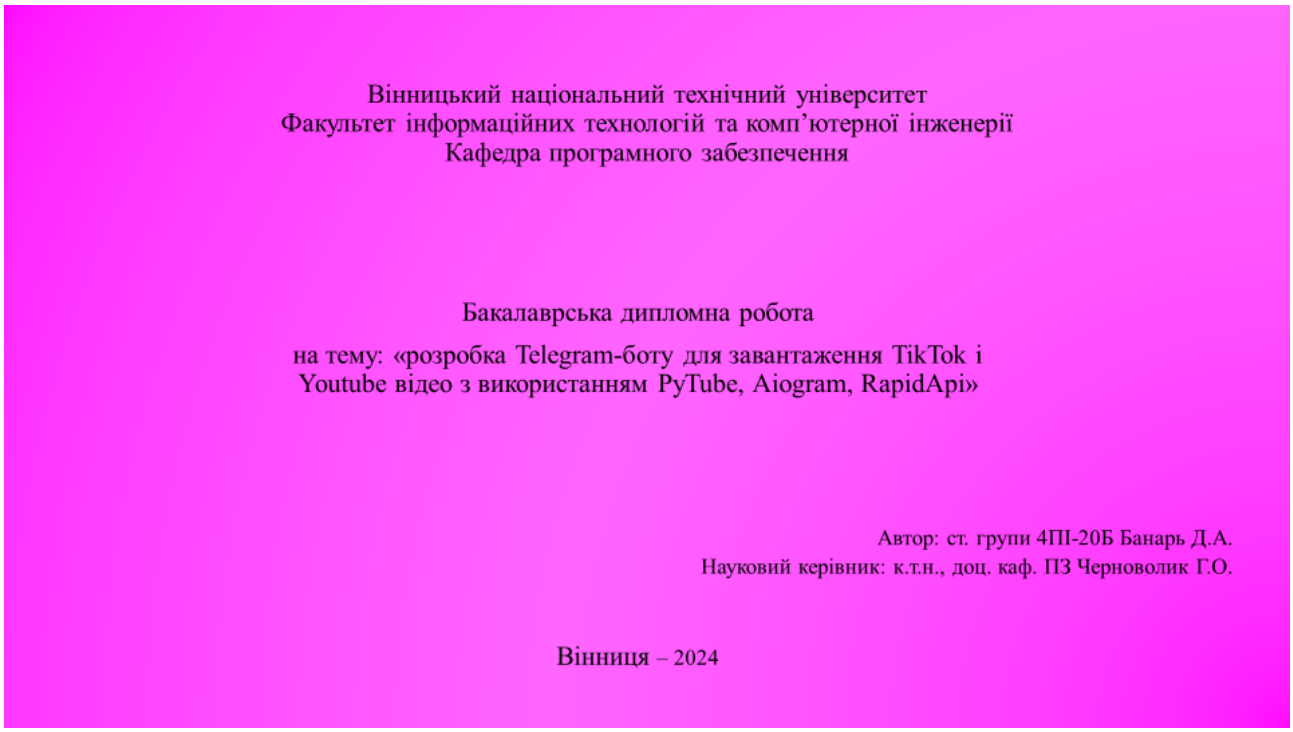


Рисунок Г.1 – Титульний слайд презентації

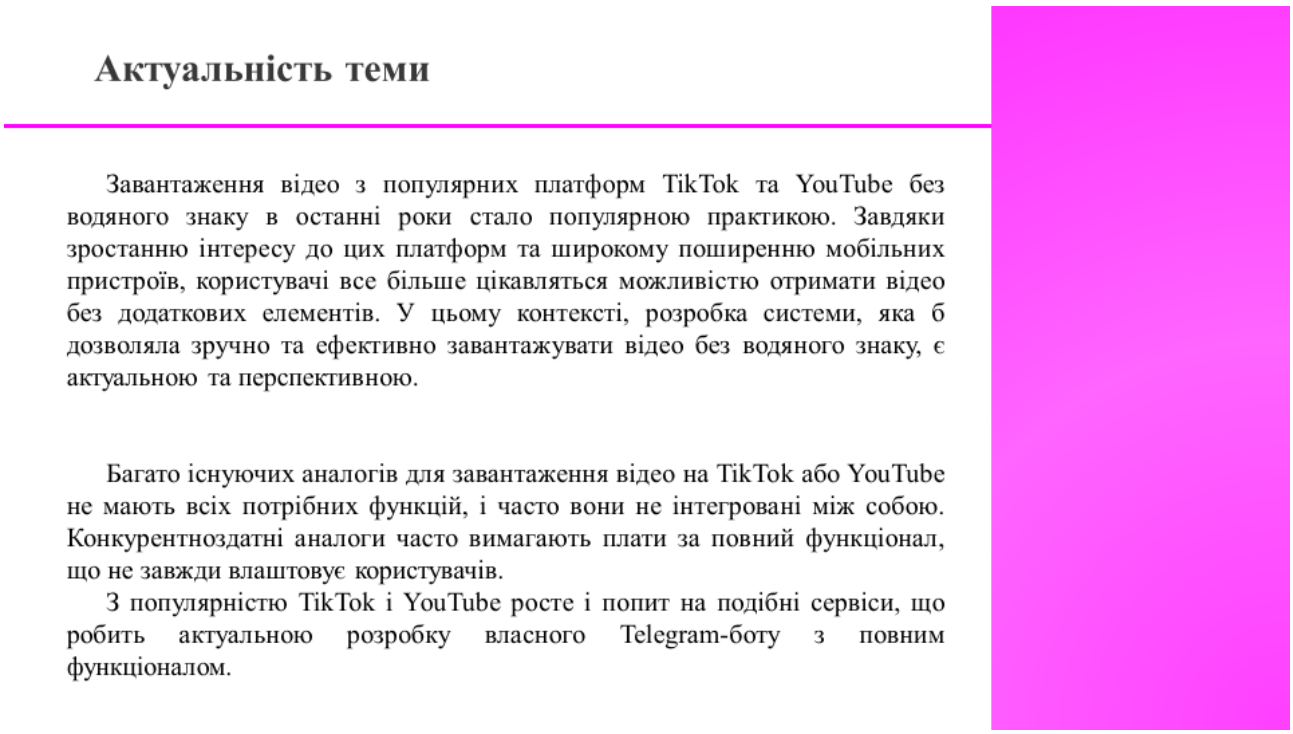


Рисунок Г.2 – Слайд «Актуальність теми»

Мета, об'єкт та предмет дослідження

1. **Мета** бакалаврської дипломної роботи полягає у підвищенні якості завантаження відео-контенту з платформ TikTok та YouTube за рахунок розробки і використання спеціалізованого Telegram-бота з використанням PyTube, Aiogram, RapidApi, що дозволяє завантажувати відео без водяних знаків.
2. **Об'єкт дослідження** є процес розробки функціоналу та інтерфейсу для Telegram-боту для завантаження TikTok та YouTube без водяного знаку.
3. **Предмет дослідження** є алгоритми і засоби реалізації основного функціоналу і бази даних.

Рисунок Г.3 – Слайд «Мета, об'єкт та предмет дослідження»

Завдання на бакалаврську дипломну роботу

1. розробити методи роботи системи;
2. визначити загальний алгоритм системи;
3. розробити загальну структуру бази даних за допомогою SQLite3;
4. розробити функіонал для взаємодії користувача з Telegram-ботом;
5. реалізувати функціонал для адміністрування кількості викликів з боку користувача;
6. здійснити тестування загальної системи згідно поставлених задач;

Рисунок Г.4 – Слайд «Завдання на бакалаврську дипломну роботу»

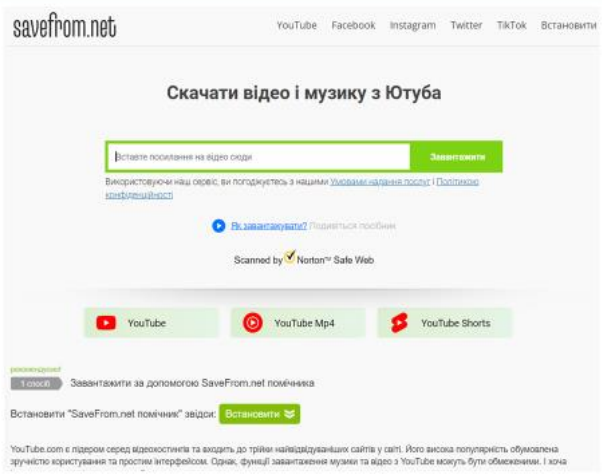
Наукова новизна отриманих результатів та практична цінність

1. Подальшого розвитку отримав метод обробки відео-ресурсів у Telegram-боті, який, на відміну від існуючих, базується на використанні бібліотеки MoviePy, що дозволяє забезпечити швидке та ефективне кліпування відеорядів без накладання водяного знаку та збільшити продуктивність роботи бота.
2. Подальшого розвитку отримав метод пошуку відео, який, на відміну від існуючих, базується на використанні ефективних алгоритмів для формування HTTP-запитів та обробки відповідей від YouTube API, використовує вбудовані функції PyTube для зручної взаємодії з YouTube API, що дозволяє більш повно використовувати можливості YouTube каналу.

Практична цінність бакалаврської дипломної роботи полягає в наданні якісного і безкоштовного інструменту для роботи з відео-ресурсами. Програмні модулі, розроблені в рамках бакалаврської дипломної роботи, можуть бути використані для створення подібних ботів для інших сервісів або функцій.

Рисунок Г.5 – Слайд «Наукова новизна отриманих результатів та практична цінність»

Порівняльний аналіз аналогів

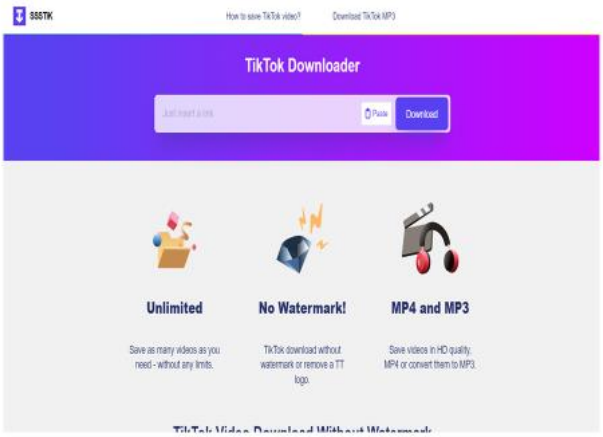


Сторінка WEB-сервісу «SaveFrom.net»

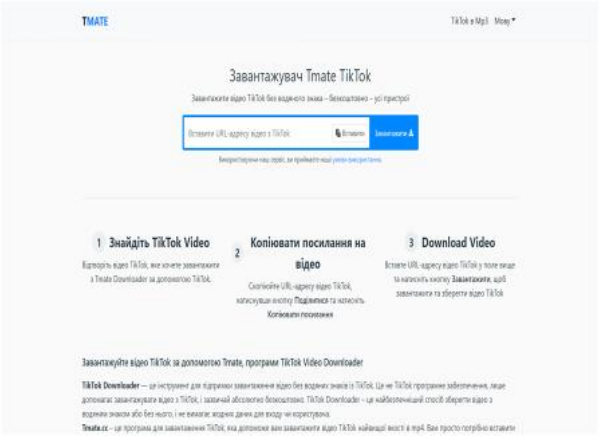
Інтерфейс боту «TokMateBot»

Рисунок Г.6 – Слайд «Порівняльний аналіз аналогів» (частина 1)

Порівняльний аналіз аналогів



Сторінка WEB-застосунку «Ssstik.io»



Сторінка WEB-застосунку «TMate»

Рисунок Г.7 – Слайд «Порівняльний аналіз аналогів» (частина 2)

Порівняльний аналіз аналогів

Критерій	TokMateBot	SaveFrom.net	Ssstik.io	Tmate	BanDimBot
Завантаження відео з Tik Tok	+	+	+	+	+
Завантаження відео з YouTube	–	+	–	–	+
Доступність функцій з нестабільним інтернетом	+	–	–	–	+
Можливість обирати якість відео	–	+/–	–	–	–
Кросплатформенність	+	+	+	+	+
Звітність, щодо використаних операцій	–	+	–	–	+
Загальна оцінка	50%	85%	20%	20%	95%

Рисунок Г.8 – Слайд «Порівняльний аналіз аналогів» (частина 3)

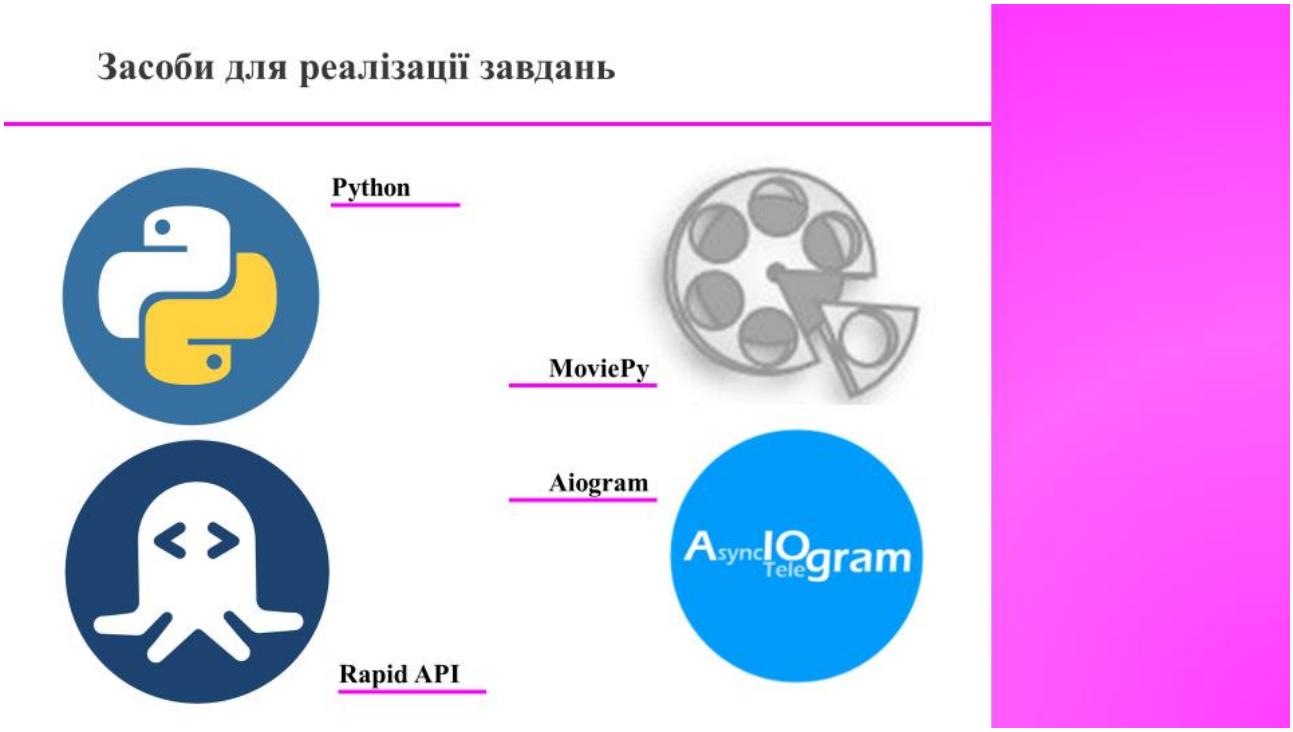


Рисунок Г.9 – Слайд «Засоби для реалізації завдань»

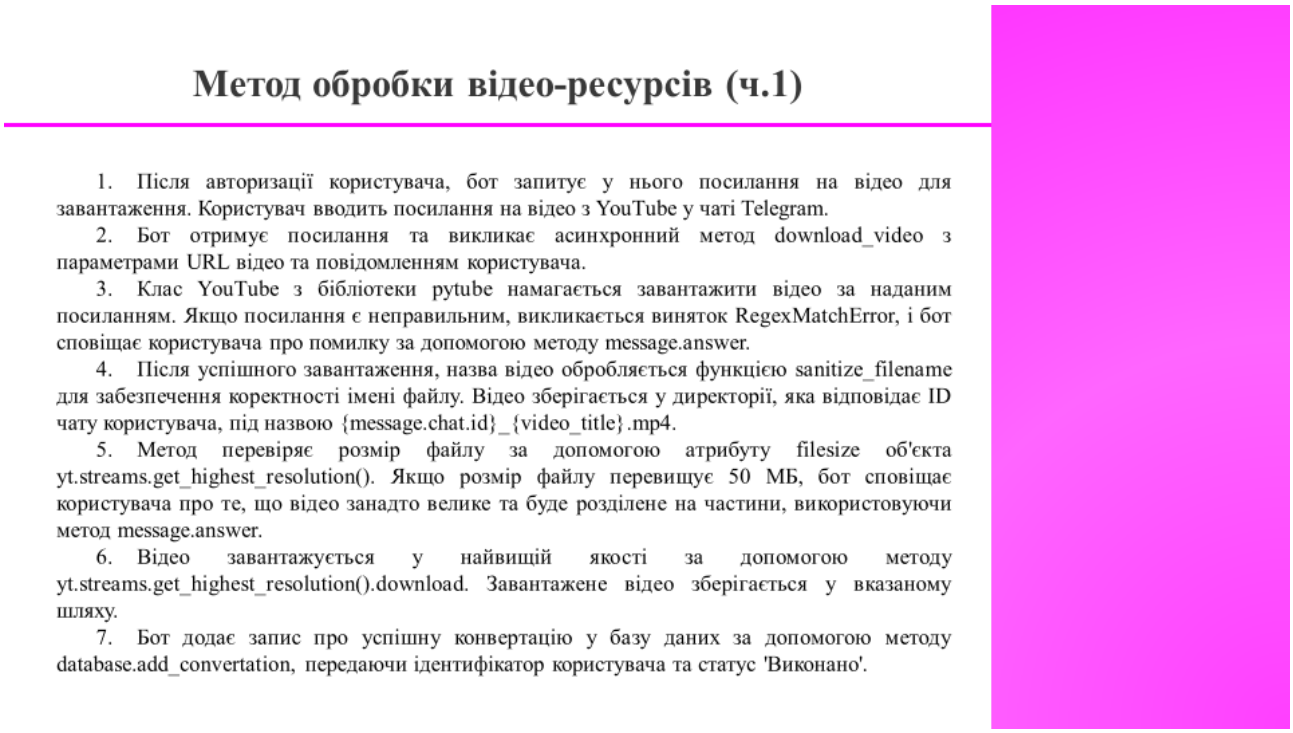


Рисунок Г.10 – Слайд «Метод обробки відео-ресурсів (ч.1)»

Метод обробки відео-ресурсів (ч.2)

8. Метод виконує затримку на 3 секунди за допомогою функції await asyncio.sleep(3) для забезпечення стабільності процесу.
9. Якщо розмір файлу перевищує 50 МБ, відео розділяється на частини за допомогою класу VideoFileClip з бібліотеки moviepy.
10. Відеофайл завантажується і відкривається для подальшої обробки, використовуючи методи бібліотеки moviepy.editor.
11. За допомогою атрибуту duration класу VideoFileClip визначається загальна тривалість відео в секундах.
12. Для поділу відео на частини встановлюється фіксована тривалість кожного сегмента у 180 секунд.
13. Обчислення кількості сегментів, на які буде розділене відео здійснюється за допомогою цілочисельного ділення загальної тривалості відео на тривалість одного сегмента з додаванням одиниці.
14. Для кожного сегмента працює визначення початкового (start_time) та кінцевого (end_time) часу сегмента.
15. Початковий час кожного сегмента обчислюється як start_time = i * chunk_duration, де i – номер сегмента, а chunk_duration – тривалість сегмента. Кінцевий час обчислюється як end_time = min((i + 1) * chunk_duration, duration), де duration – загальна тривалість відео.

Рисунок Г.11 – Слайд «Метод обробки відео-ресурсів (ч.2)»

Метод обробки відео-ресурсів (ч.3)

16. Далі використовуючи метод subclip класу VideoFileClip, створюється підкліп відео з вказаними початковим та кінцевим часами. Підкліп зберігається у тимчасовий файл з унікальною назвою, яка включає номер сегмента, за допомогою методу write_videofile.
17. Надсилання підкліпу користувачеві за допомогою методу bot.send_video, супроводжуючи повідомлення інформацією про назву відео, канал та порядковий номер частини:
18. Тимчасовий файл підкліпу відкривається та надсилається користувачеві у вигляді відеоповідомлення з відповідним описом.
19. Після відправлення відеоповідомлення тимчасовий файл видаляється для звільнення дискового простору.
20. Якщо розмір відео не перевищує 50 МБ, воно надсилається користувачеві цілком за допомогою методу bot.send_video, разом з інформацією про назву відео та канал.
21. Після відправлення відео, тимчасові файли видаляються для оптимізації використання дискового простору за допомогою функції os.remove.
22. У разі виникнення винятків, наприклад, RegexMatchError при неправильному посиланні на відео, бот сповіщає користувача про помилку за допомогою методу message.answer.
23. І на останок бот повертається у режим очікування наступного завдання від користувача, готовий обробляти нові запити на завантаження відео.

Рисунок Г.12 – Слайд «Метод обробки відео-ресурсів (ч.3)»

Метод пошуку відео

Пошук відео, якщо бот визначив, що воно було надіслано з платформи YouTube:

1. Коли користувач надсилає посилання на відео з YouTube, спочатку проводиться перевірка посилання. Якщо воно дійсне, викликається функція process_youtube(url, message).
2. У функції process_youtube, спочатку викликається download_video(url, message) для завантаження відео з YouTube.
3. Для цього використовується бібліотека PyTube, яка взаємодіє з YouTube API, використовуючи ефективні алгоритми для формування HTTP-запитів.
4. HTTP-запити формуються згідно з документацією YouTube API та надсилаються до серверів YouTube для отримання відповіді з інформацією про відео.

Пошук відео, якщо бот визначив, що воно було надіслано з платформи TikTok:

1. Коли користувач надсилає посилання на відео з TikTok, спочатку проводиться перевірка посилання. Якщо посилання є дійсним, викликається функція process_tiktok(url, message).
2. У функції process_tiktok, викликається download_tiktok_video(link, message) для завантаження відео з TikTok.
3. В цьому випадку також використовується HTTP-запит, але до серверів TikTok через Rapid API. Формування запитів також відбувається відповідно до документації TikTok API та Rapid API.

Рисунок Г.13 – Слайд «Метод пошуку відео»

Блок-схема методу для обробки відео-ресурсів

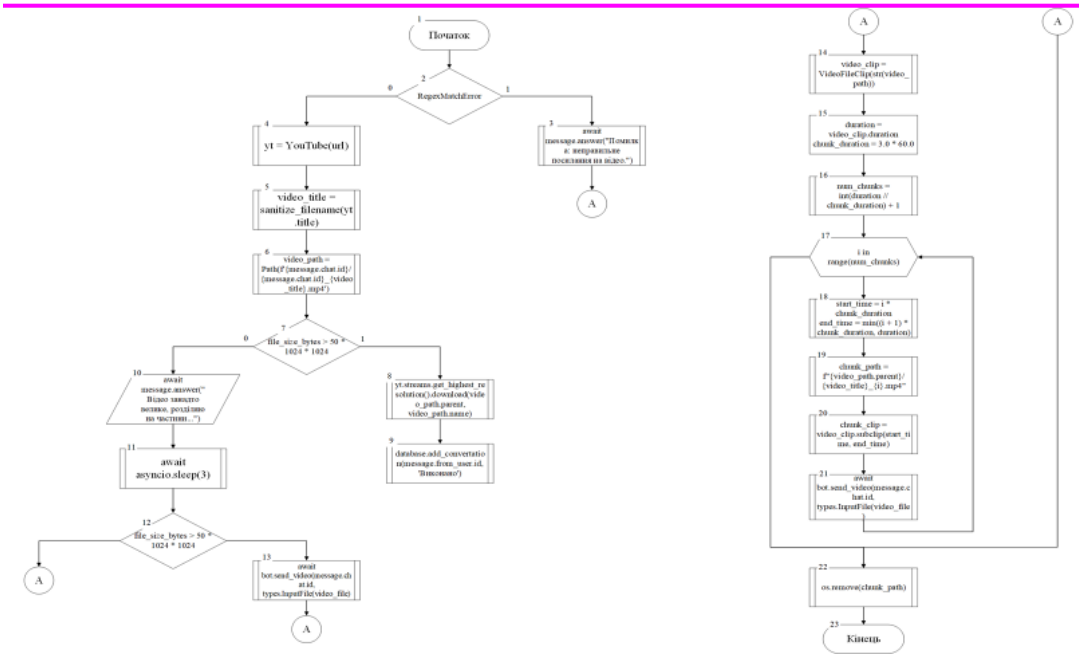


Рисунок Г.14 – Слайд «Блок-схема методу для обробки відео-ресурсів»

UML-діаграма діяльності Telegram-боту

Рисунок Г.15 – Слайд «UML-діаграма діяльності Telegram-боту»

Загальний алгоритм роботи програми та алгоритм для запису користувача в базу даних

Рисунок Г.16 – Слайд «Загальний алгоритм роботи програми та алгоритм для запису користувача в базу даних»

Алгоритм визначення типу посилання

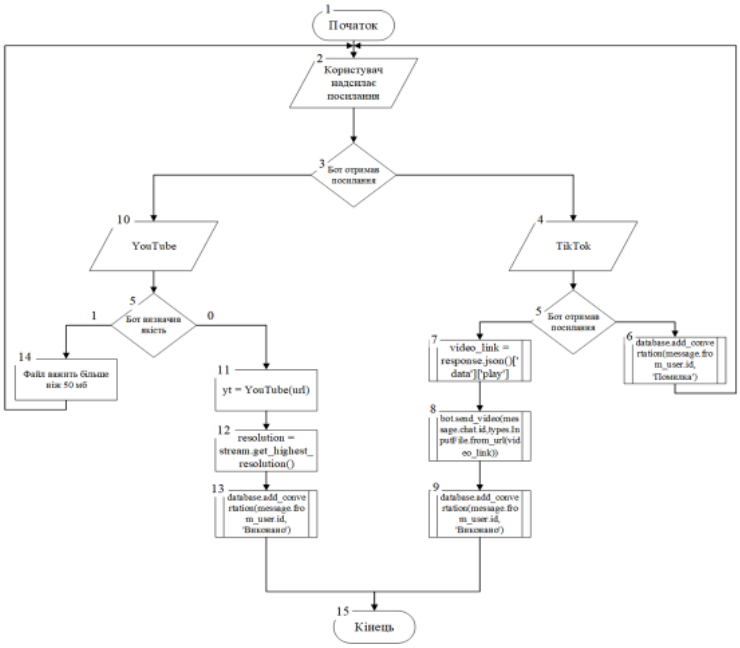
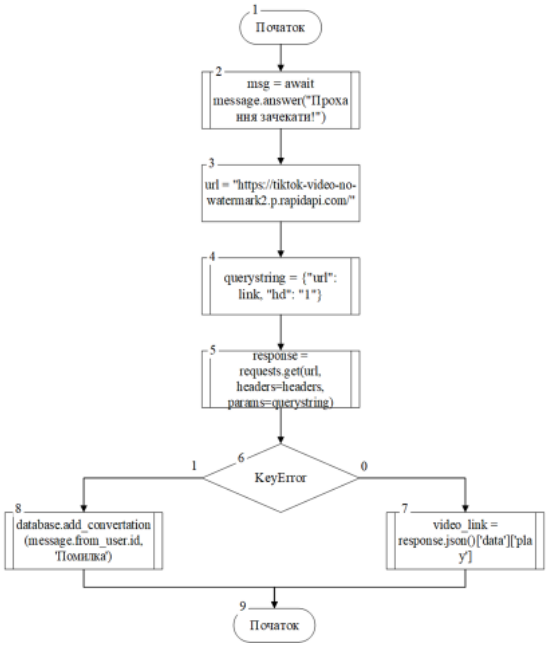


Рисунок Г.17 – Слайд «Алгоритм визначення типу посилання»

Блок-схема функції для завантаження відео з платформи TikTok



Основною функцією Telegram-боту є можливість завантаження відео з платформ TikTok та Youtube без водяного знаку. Реалізація цієї функції включає в себе використання різних API та бібліотек, які дозволяють отримувати відео з відповідних платформ та надсилати його користувачам Telegram.

Рисунок Г.18 – Слайд «Блок-схема функції для завантаження відео з платформи TikTok»

Блок-схема функції додавання користувача у базу даних

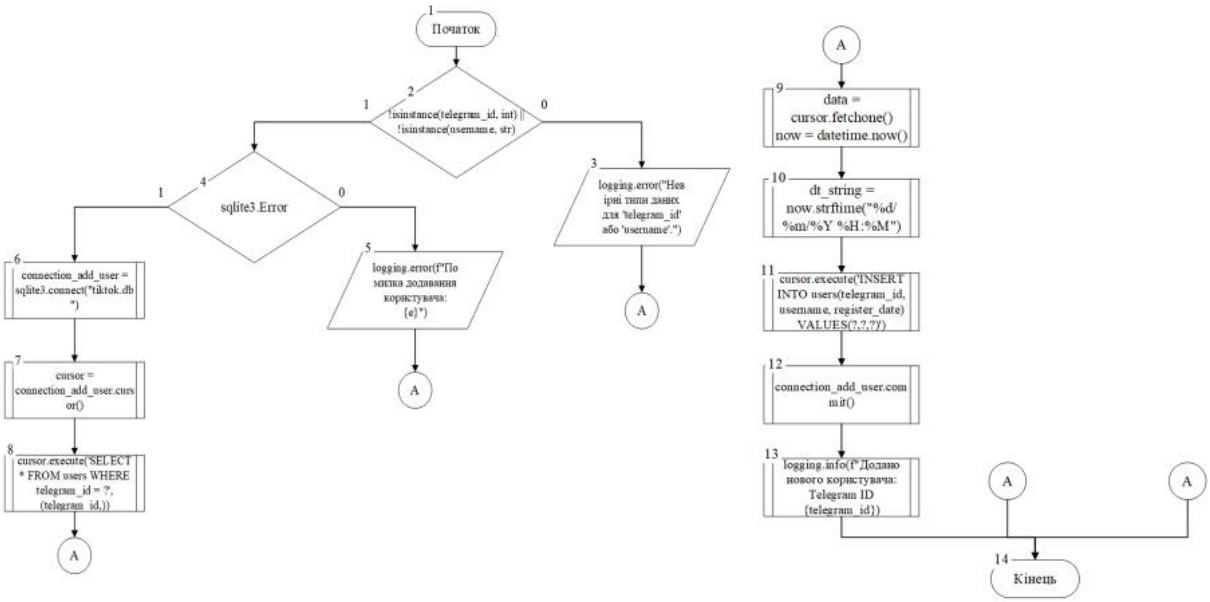
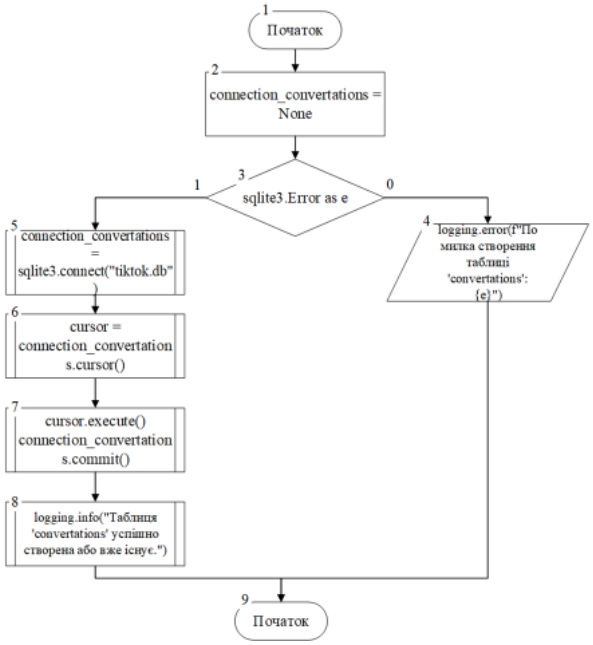


Рисунок Г.19 – Слайд «Блок-схема функції додавання користувача у базу даних»

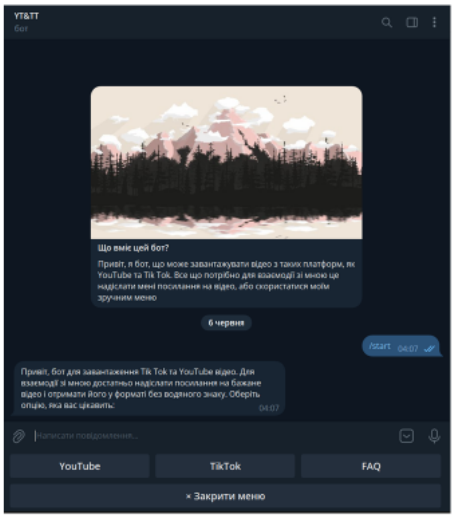
Блок-схема функції створення таблиці з операціями



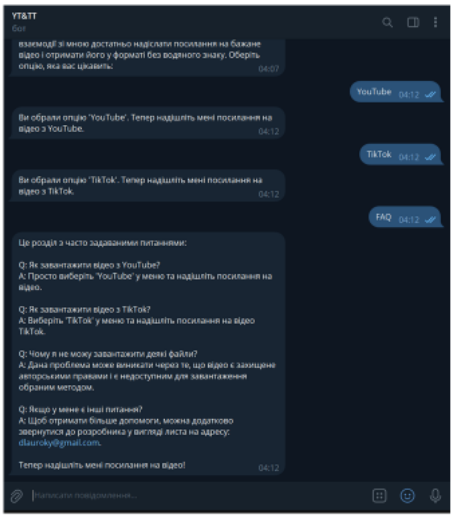
Функція create_table_convertations в базі даних створює таблицю convertations, яка використовується для збереження інформації про виконані конвертації відео.

Рисунок Г.20 – Слайд «Блок-схема функції створення таблиці з операціями»

Функціонал розробленого боту



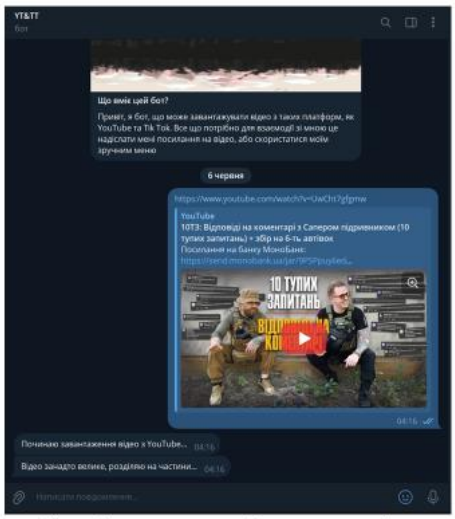
Відповідь на початок роботи з новим користувачем



Відповідь на базові кнопки IVR-меню

Рисунок Г.21 – Слайд «Функціонал розробленого боту (частина 1)»

Функціонал розробленого боту



Відповідь на початок роботи з довгим відео

Відповідь з першим фрагментом довгого відео

Рисунок Г.22 – Слайд «Функціонал розробленого боту (частина 2)»

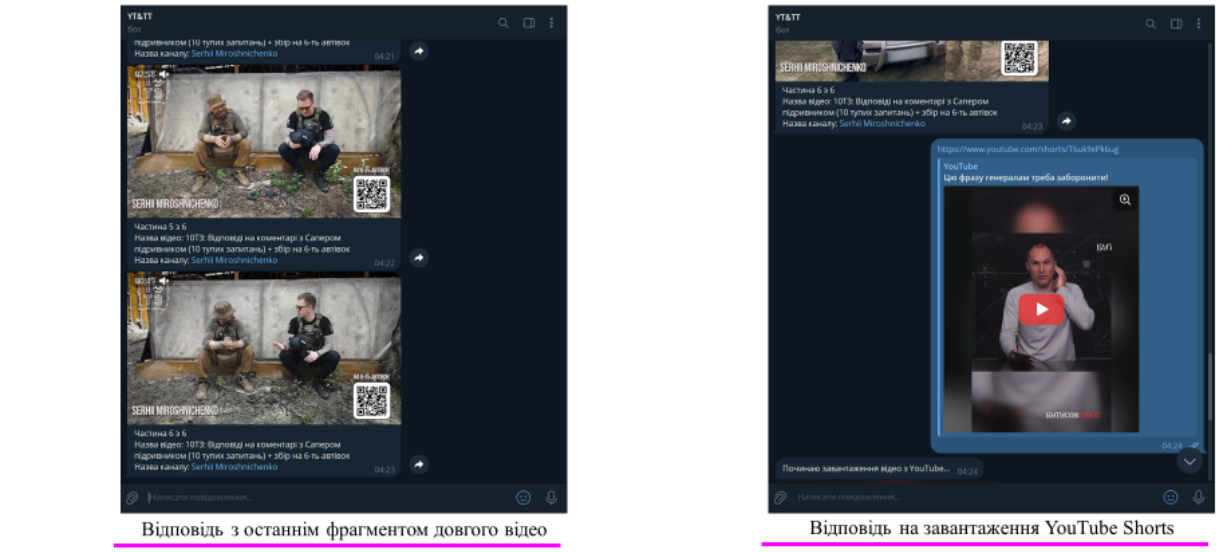


Рисунок Г.23 – Слайд «Функціонал розробленого боту (частина 3)»

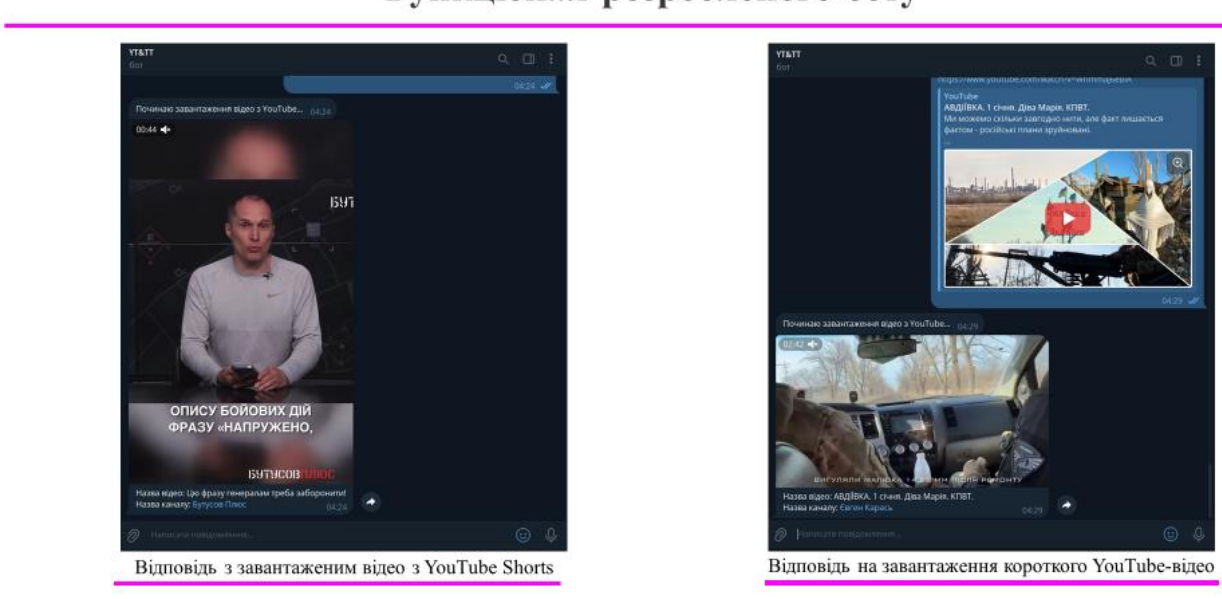
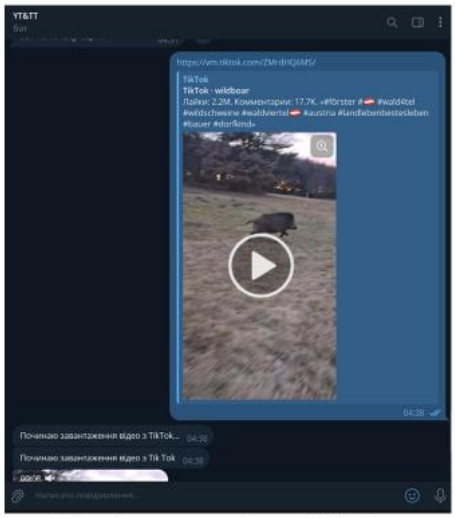
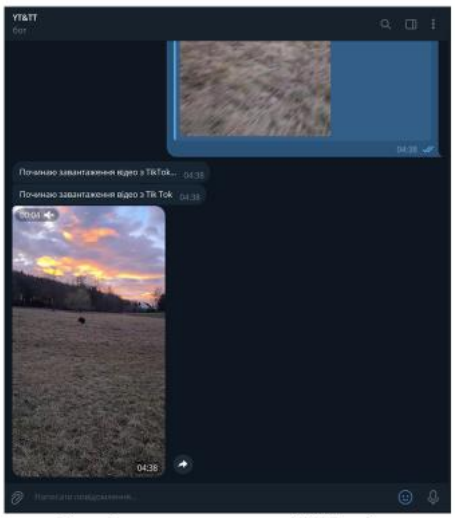


Рисунок Г.24 – Слайд «Функціонал розробленого боту (частина 4)»

Функціонал розробленого боту



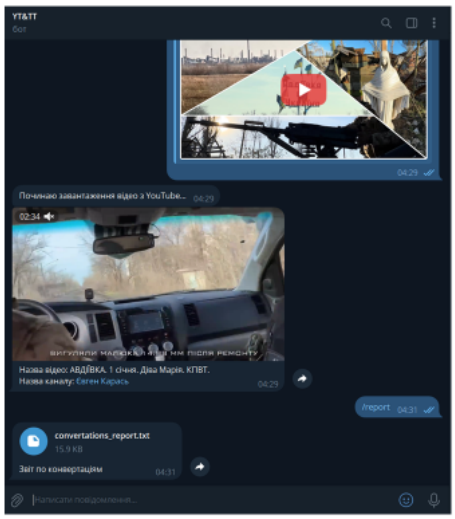
Відповідь-початок роботи з TikTok відео



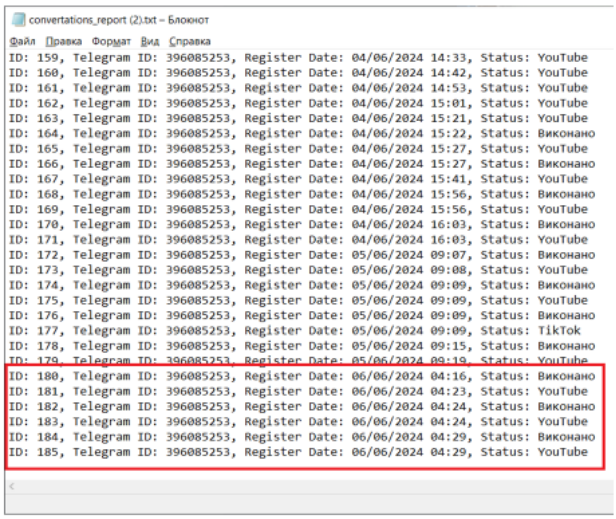
Відповідь на завантаження TikTok відео

Рисунок Г.25 – Слайд «Функціонал розробленого боту (частина 5)»

Функціонал розробленого боту



Видача файлу-звітності з бази даних



Вигляд звітності з бази даних

Рисунок Г.26 – Слайд «Функціонал розробленого боту (частина 6)»

Висновки

У бакалаврській дипломній роботі було проаналізовано Telegram-боти та їх набираючу популярність в Україні. Вони стають потужним інструментом для взаємодії з аудиторією, автоматизації процесів та створення інтерактивних сервісів.

Результати аналізу підтвердили високий потенціал Telegram-ботів у сферах завантаження відео-контенту, збереження документації та юзабіліті для користувача, що підкреслює їх значущість як важливого інструменту в сучасному цифровому середовищі.

Також бакалаврська дипломна робота оформлена згідно методичних вказівок.

У бакалаврській дипломній роботі було виконано наступні завдання:

1) розроблено методи роботи системи;
2) розроблено загальний алгоритм системи;
3) розроблено загальну структуру бази даних за допомогою SQLite3;
4) розроблено функціонал для взаємодії користувача з Telegram-ботом;
5) реалізовано функціонал для адміністрування кількості викликів з боку користувача;
6) здійснено тестування загальної системи згідно поставлених задач. .

Рисунок Г.27 – Слайд «Висновки (частина 1)»

Висновки

Було розглянуто стан Telegram-ботів та їх актуальність. Проведено порівняльний аналіз аналогів Telegram-ботів, які дозволяють завантажувати відео з платформ TikTok та Youtube без водяного знаку. Перелічено відомі аналоги зазначивши їхні переваги та недоліки. Було відзначено, що кращим рішенням для цього проєкту буде розробка власного Telegram-бота для готової платформи Telegram з використанням її API та Python для програмування.

Виконано детальний аналіз вхідних даних програмного продукту. На його основі була розроблена модель роботи за допомогою UML діаграми, яка відображає основні процеси та взаємозв'язки в програмі. Також у розділі було створено основний алгоритм роботи Telegram-боту, який визначає послідовність дій для обробки користувацьких запитів.

Обгрунтовано вибір мови програмування та методів для розробки системи до Telegram-боту. Проведено тестування програмних модулів Telegram-боту для завантаження відео з TikTok та YouTube. Під час тестування було здійснено аналіз та встановлено мінімальні та рекомендовані системні конфігурації для коректної роботи боту. У результаті тестування було підтверджено, що всі модулі відповідають поставленій задачі і функціонують бездоганно.

Крім того, було розроблено інструкцію користувача з покроковим описом роботи розробленого боту. Ця інструкція дозволяє користувачам з легкістю взаємодіяти з ботом та виконувати необхідні дії для завантаження відео з TikTok та YouTube.

Рисунок Г.28 – Слайд «Висновки (частина 2)»

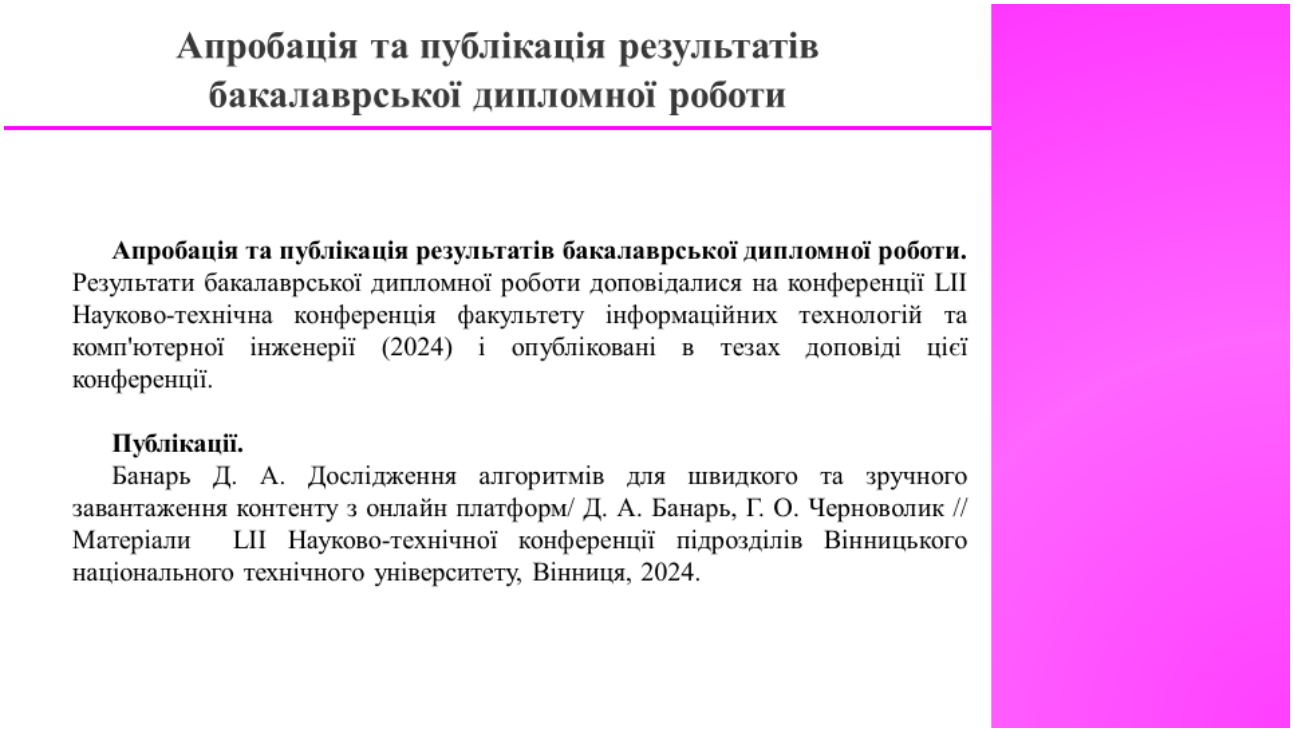


Рисунок Г.29 – Слайд «Апробація та публікація результатів бакалаврської дипломної роботи»