МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
(повне найменування назва факультету)

кафедра інформаційних систем
(повна назва кафедри)

ДИПЛОМНА РОБОТА

на тему:

Застосування штучного інтелекту для аналізу повідомлень

Виконав: студент групи ПМі-42
спеціальності 122 – комп'ютерні науки

Морозов В. А.
(підпис) (прізвище та ініціали)

Керівник доц. Бернакевич І. Є.
(підпис) (прізвище та ініціали)

Рецензент
(підпис) (прізвище та ініціали)

Львів – 2023

ЗМІСТ

Вступ 4

Розділ 1. Постановка задачі 5

1.1 Функціональні вимоги до Telegram бота 5

1.2 Нефункціональні вимоги до Telegram бота 6

1.3 Use-case діаграма 7

Розділ 2. Використані технології 8

2.1 Підключення до Telegram API 8

2.2 Реєстрація бота через BotFather 9

2.3 Бібліотека Telethon 10

2.4 Бібліотека SpaCy 10

Розділ 3. Основний функціонал чат-бота 13

3.1 Робота зі списками 13

3.2 Тригери 15

3.3 Згадка всіх учасників чату 18

3.4 Розіграш титулу «котик дня» 18

3.5 Інший функціонал 20

Розділ 4. Застосування алгоритму аналізу тексту 22

4.1 Робота зі списками 22

4.1.1 Створення нового списку 22

4.1.2 Додавання пунктів до списків 23

4.1.3 Видалення пунктів списку 24

4.2 Тригери 24

4.3 Розіграш титулу «котик дня» 26

4.3.1 Реєстрація в розіграші 27

4.3.2 Результат розіграшу 27
4.3.3 Рейтинг переможців 28
4.4 Додатковий функціонал 29
Висновки 31
Список використаних джерел 32

ВСТУП

Кожна людина завжди хоче бути на зв’язку зі своїми друзями, знайомими, родичами або ж просто ділитись своїми думками. Саме тому на сьогоднішній день соціальні мережі та месенджери мають величезну популярність. Кількість користувачів цих додатків стрімко росла впродовж останніх років. І щоб привернути увагу саме до свого застосунку, розробники вимушені вигадувати різноманітні особливості. Від підтримки голосових повідомлень до ігор, що можна грати з друзями прямо в чаті, від різноманітних стікерів до чат-ботів. Все це допомагає кожному додатку вирізнятись на фоні інших.

Користуючись месенджером Telegram, можна одразу помітити велике різноманіття чат-ботів, які виконують різний функціонал: пошук музики, ігри, підтримка розмови, оформлення стікерів, збереження цитат, тощо. Проте серед них існує доволі мало ботів з українським інтерфейсом. Більшість з них мають російське походження, що на сьогоднішній день є негативною рисою для українських користувачів.

Також, все більшої популярності набувають чат-боти, такі як ChatGPT. Їхня можливість відповідати на запити користувачів та підтримувати видимість живого співрозмовника дуже приваблюють найрізноманітніших користувачів. Такі боти можуть виконувати багато різних завдань, хоча кожен з них буде більш спеціалізованим в чомусь одному.

Зважаючи на все вищеперераховане, було вирішено створити чат-бота, що міг би розширити функціонал звичайного чату в Telegram та стати помічником українських користувачів цього месенджера. При цьому, він зможе розуміти не лише команди, а я звичайні запити, написані реченнями.

РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ

Необхідно створити застосунок, що реалізує Telegram чат-бота, який надає доступ до розширених можливостей користування даним месенджером. При цьому, вимагається, щоб користування даним ботом було можливе з застосуванням природної мови. Тобто фактичним завданням цієї роботи є створення застосунку, що за допомогою алгоритмів машинного навчання здатен розрізняти та аналізувати природне мовлення в повідомленнях та виконувати відповідні завдання.

Завдання, які має виконувати чат-бот наведено в наступному підрозділі.

1.1 Функціональні вимоги до Telegram бота

– Створення списків, які може редагувати кожен учасник групового чату:

Редагування списків всіма учасниками дозволяє одразу декільком користувачам одночасно пропонувати свої ідеї без страху, що вони загубляться серед інших повідомлень. Це корисно для вирішення, які речі необхідно взяти в похід, або ж утворити порядок виступів.

– Реакція на сталі вирази, визначені користувачем:

Створення тригерів дозволяє користувачу зберігати інформацію за певною фразою або ж створювати веселі відповіді-реакції. Збереження інформації таким чином дозволяє зменшити кількість закріплених повідомлень та спростити її пошук, адже потрібно лише написати повідомлення.

– Можливість згадки кожного учасника групового чату:

В групових чатах можуть виникати ситуації, коли необхідно зібрати всіх для обговорення, вирішення якоїсь проблеми, донесення важливої інформації, тощо. Проте, в месенджері Telegram не реалізований функціонал для таких випадків.

– Щоденний розіграш титулу:

Для створення кращої атмосфери в груповому чаті, один раз на день можна обирати серед учасників одного, якому присвоюється титул «котик дня».

– Зворотній зв’язок з розробником:

Зворотній зв’язок дозволяє знаходити небажану поведінку і ділитись своїми ідеями покращення бота.

1.2 Нефункціональні вимоги до Telegram бота

– Українська мова інтерфейсу:

Забезпечення того, що всі повідомлення чат-бота будуть українською мовою, оскільки цільовою аудиторією є українські користувачі месенджера.

– Зрозумілість опису функціоналу:

Забезпечення достатньої кількості опису функціоналу, що дозволяє користувачам зрозуміти, що виконує та чи інша команда.

– Зручність використання:

Використання функціоналу має бути доволі простим для того, щоб користувачам було зручно ним користуватись. Також користування має бути інтуїтивно зрозумілим.

– Відсутність затримки відповіді

– Можливість використання команд:

Забезпечення використання всього функціоналу чат-бота за допомогою команд. Команда – переважно одне слово, написане латинськими літерами, перед яким стоїть символ `/`

– Можливість використання текстових запитів (природна мова):

Забезпечення використання всього функціоналу чат-бота за допомогою запитів від користувачів, які виглядають як звичайне речення.

1.3 Use-case діаграма

Для кращого представлення поставлених завдань було створено use-case діаграму. Як видно з рисунку 1.1, функціонал чат-бота складається з чотирох категорій та однієї функції. Кожна з категорій в свою чергу поділяється на 3 основні функції відповідної категорії.

За допомогою цього логічного поділу, можна створити модульний поділ застосунку, що допоможе проводити виправлення функціоналу однієї категорії не впливаючи на функціонал інших.

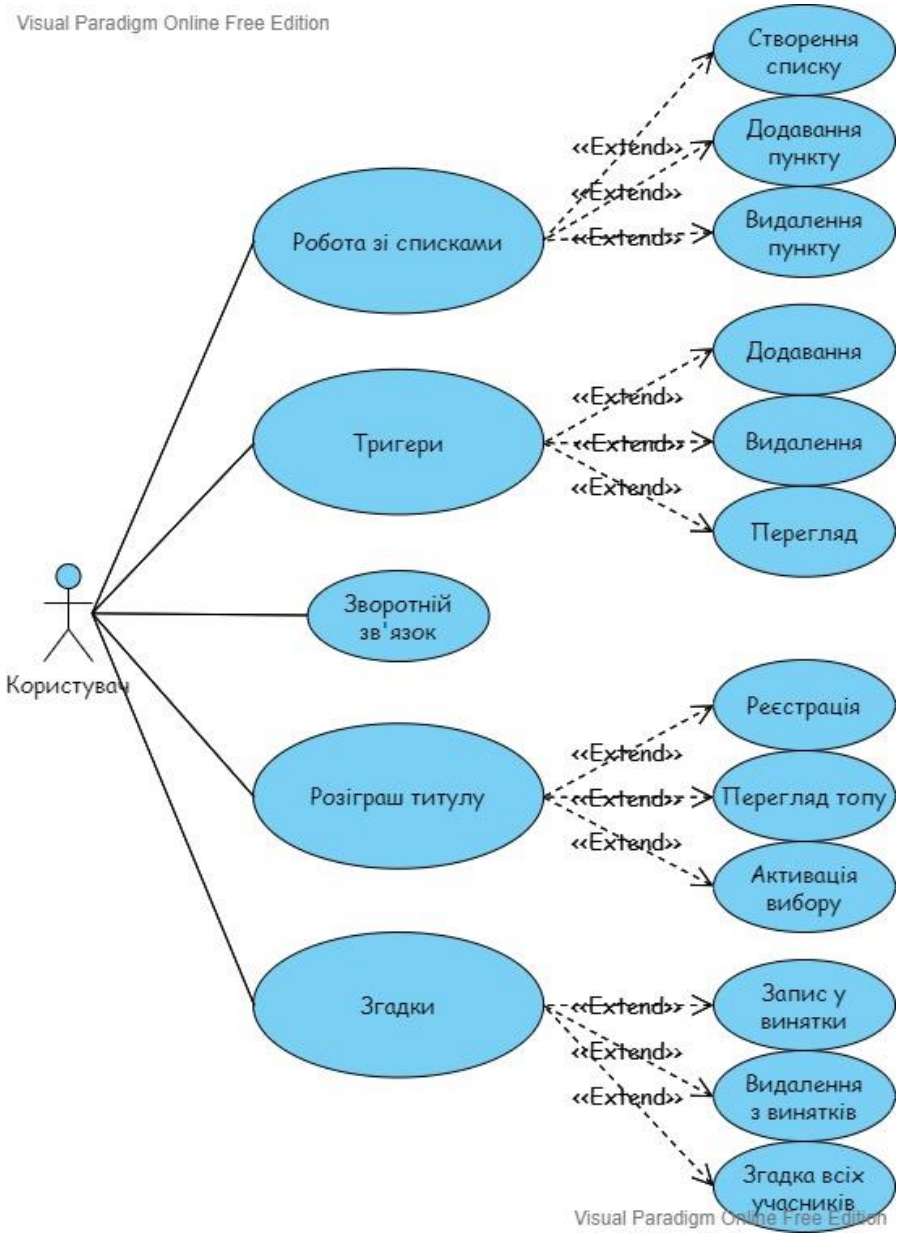

Рисунок 1.1 – Use-case діаграма застосунку

РОЗДІЛ 2. ВИКОРИСТАНІ ТЕХНОЛОГІЇ

Для створення власного Telegram бота необхідно пройти декілька обов'язкових етапів: отримання токену, реєстрація бота, створення та хостинг програми[2].

2.1 Підключення до Telegram API

Для того, щоб програма мала можливість взаємодіяти з повідомленнями в месенджері Telegram, необхідно підключитися до його API. Це можна доволі швидко зробити на сайті Telegram. Зайшовши на сайт, потрібно відкрити свій акаунт, ввівши зареєстрований в мережі номер телефону, та отриманий на додаток код підтвердження. Після успішного входу, можна обрати пункт «API development tools», де сайт дозволить створити або налаштувати свій додаток для підключення.

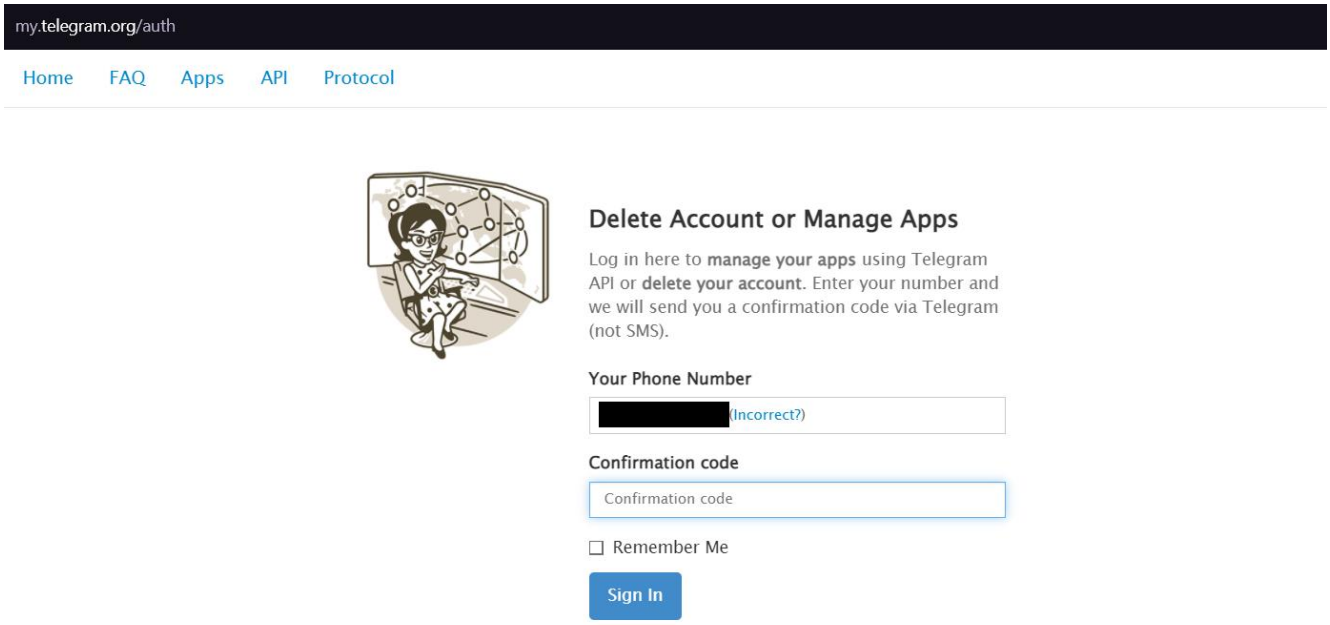

Рисунок 2.1 – Вхід на сайт Telegram

Для того, щоб програмно підключитись до API та мати змогу програмно читати повідомлення та відповідати на них, на сайті потрібно взяти секретні ключі: api_id та api_hash. Ці ключі потрібно тримати в секреті, щоб ніхто інший не міг користуватись вашим підключенням.

2.2 Реєстрація бота через BotFather

Наступним кроком для створення свого Telegram чат-бота є налаштування його вигляду в офіційному боті від розробників Telegram – BotFather. Цей бот допомагає користувачам задати зображення, яке буде використовуватись як аватар майбутнього бота, дати йому псевдонім, ім’я та опис. Також тут можна керувати й іншими аспектами бота.

Але найголовніше на цьому етапі – отримати токен бота – ключ, який дозволяє підключати код з програмною логікою до створеного бот-акаунта. Як і попередні ключі, цей потрібно тримати в секреті, адже будь-хто з його допомогою може використовувати вашого бота.

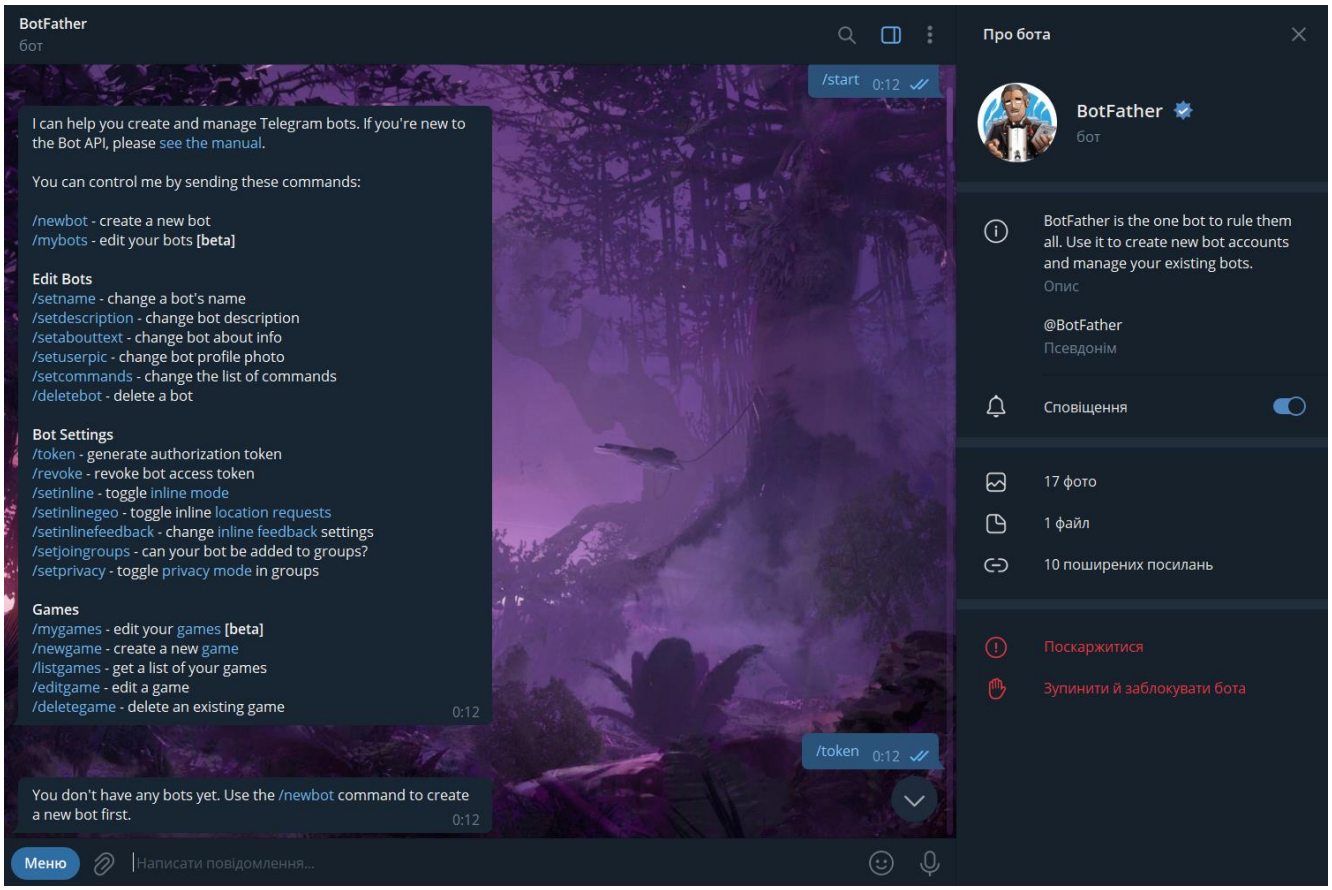

Рисунок 2.2 – BotFather

2.3 Бібліотека Telethon

Отримавши три згадані раніше ключі, можна скористатись одною з багатьох різних бібліотек для написання програми Telegram бота. Для цього бота було обрано python бібліотеку Telethon[3]. Простота написання коду з допомогою цієї бібліотеки та можливість запускати python файли на різних платформах були вирішальними факторами при виборі бібліотеки.

Можливість реагувати на певні повідомлення надається через декоратор @client.on, в параметри якого потрібно передати шаблон очікуваного повідомлення. Одним з параметрів повідомлення є очікуваний текст, який можна задати з допомогою regex виразів. Також можна фільтрувати повідомлення за групою, звідки вони надіслані, або ж за тим хто їх автор.

Самі ж функції, які обробляють подію отримання повідомлення, повинні отримувати один аргумент – повідомлення, відфільтроване декоратором. З цього аргументу можна дізнатись хто його відправник, в якому чаті його було написано, який в ньому текст та багато чого іншого. Також в нього є функції, які дозволяють без зайвих кроків відповідати на отримане повідомлення.

В прикладі на рис. 2.3 можна побачити код, який змусить бота відповідати на будь-яке повідомлення, що закінчується набором символів ‘hello’ ігноруючи регістр. Відповіддю буде просте повідомлення ‘Hey!’

```
@client.on(events.NewMessage(pattern='(?i).*Hello'))
async def handler(event):
    await event.reply('Hey!')
```

Рисунок 2.3 – Приклад обробника з документації бібліотеки[3]

2.4 Бібліотека SpaCy

Для того, щоб чат-бот мав можливість аналізувати повідомлення, необхідно створити власний алгоритм машинного навчання, або ж використати відповідну бібліотеку, що дає доступ до вже навчених моделей.

Бібліотека SpaCy[4] дає можливість аналізувати текст за різними показниками, що дозволяє програмі “розуміти” текстові повідомлення користувачів.

Для того, щоб користуватися аналізом української мови, необхідно завантажити відповідну мовну модель у програмі, що буде використовувати можливості NLP (Natural Language Processing). Найперше для цього потрібно переконатися, що, окрім власне бібліотеки SpaCy, також завантажено і потрібну мовну модель. Для реалізації чат-бота було використано середню модель української мови, створеної на новинах.

Рисунок 2.4 – Створення мовної моделі шляхом завантаження

Після завантаження такої моделі, можна починати аналізувати тексти.

Конкретно ця модель підтримує наступні дії з повідомленнями:

- поділ повідомлення на окремі речення
- аналіз залежності слів в реченні
- визначення частини мови слів
- визначення ролі в реченні слів
- приведення слів до їх лем
- розпізнавання деяких іменованих об’єктів

Після того, як модель проаналізує повідомлення та створить всі необхідні токени з інформацією про них, можна продовжувати роботу з цією інформацією та отримувати з неї необхідну інформацію.

Для того, щоб визначати певні шаблонні конструкції в реченнях, можна використовувати клас Matcher з цієї бібліотеки. Він дозволяє створювати різні можливі шаблони для знаходження в тексті. Від звичайного пошуку за допомогою

регулярних виразів, його особливістю є врахування всіх атрибутів, знайдених під час аналізу тексту за допомогою NLP.

Поєднуючи можливості класу Matcher та подальший аналіз знайденого тексту за допомогою наявної інформації про слова та їх взаємозв’язки, можна програмно читати текст повідомлень та знаходити певні шаблонні фрази, що можуть бути командами для чат-бота.

При знаходженні таких фраз, можна додатково обробляти інформацію, для того, щоб, наприклад, знаходити заголовки до списків, що необхідно створити. Знайти такі заголовки доволі просто, оскільки після початкового аналізу тексту головному слову в заголовку призначається параметр dep (роль слова в реченні) = flat:title. Після знаходження такого слова можна обрати лише слова, що є залежними від нього та побудувати з них заголовок.

РОЗДІЛ 3. ОСНОВНИЙ ФУНКЦІОНАЛ ЧАТ-БОТА

3.1 Робота зі списками

Застосунок реалізує функціонал для комфортного створення списків всіма учасниками групи. Для того, щоб працювати зі списками для користувачів визначено декілька команд:

– /list <назва>
– /add <текст>
– /addmany <текст1>
  <текст2>…
– /delrow <номер>

Команда /list <назва> дозволяє користувачу створити новий список з обраною назвою. До редагування списку може доєднатись кожен з учасників чату з допомогою інших команд.

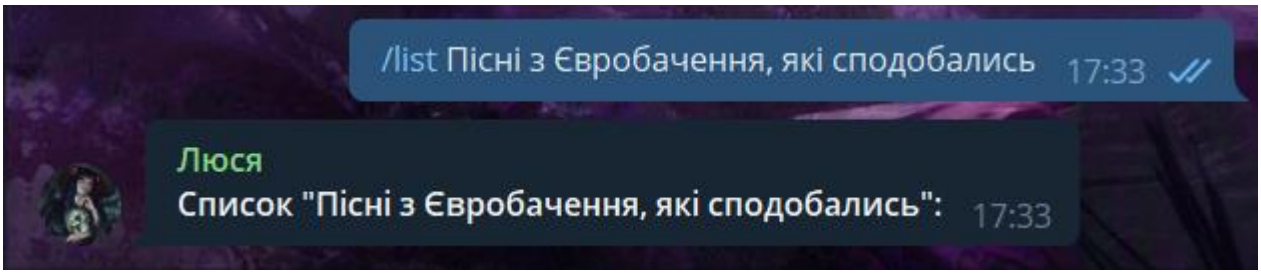

Рисунок 3.1 – Створення списку

Команди /add та /addmany дозволяють додавати до створеного ботом списку нові пункти. Команда /add додає весь текст після неї до списку одним пунктом. Тоді як /addmany дозволяє додати одразу декілька пунктів, розділених в повідомленні користувача новими рядками. Обидві команди необхідно надсилати у відповідь на повідомлення з утвореним списком.

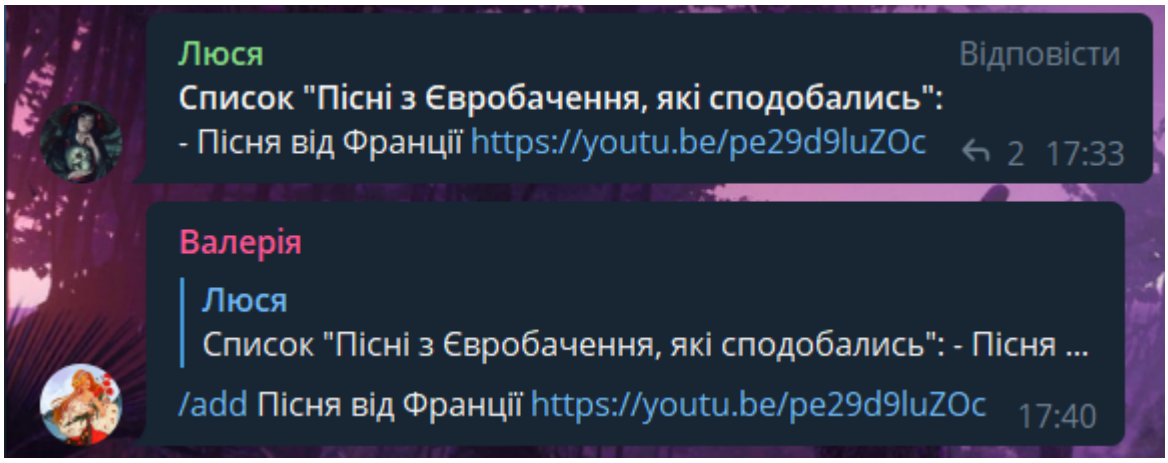

Рисунок 3.2 – Додавання пункту в список

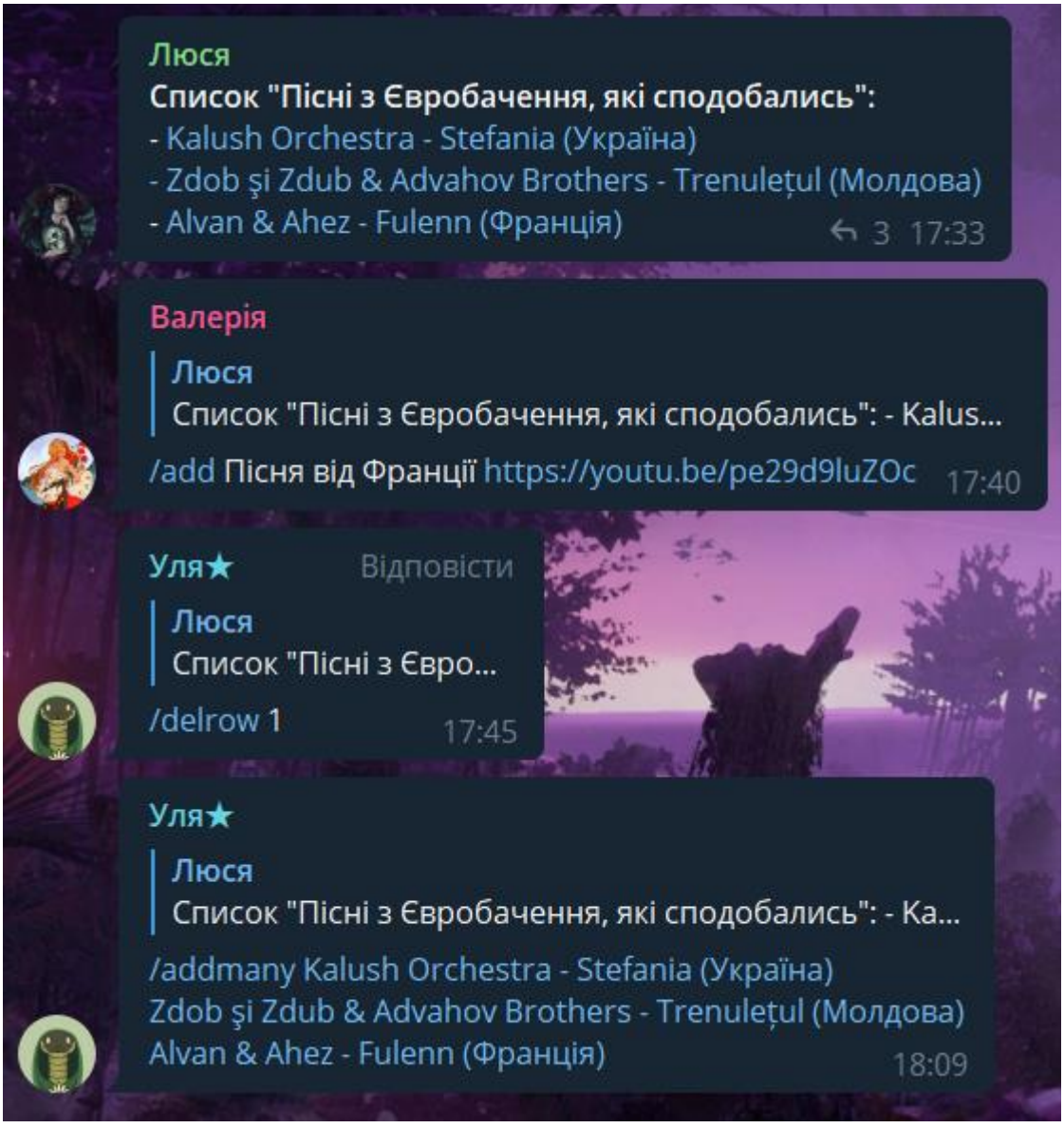

Рисунок 3.3 – Додавання декількох пунктів

Команда /delrow дозволяє користувачам викидати пункти зі списку. Для цього необхідно надіслати повідомлення команди, задавши номер рядка, який необхідно видалити, у відповідь на повідомлення зі списком. Як можна побачити з порівняння стану одного і того ж списку з рисунків 3.2 та 3.3, видалення рядків списку доступне для кожного учасника чату, а не лише для того, хто додав цей пункт до списку.

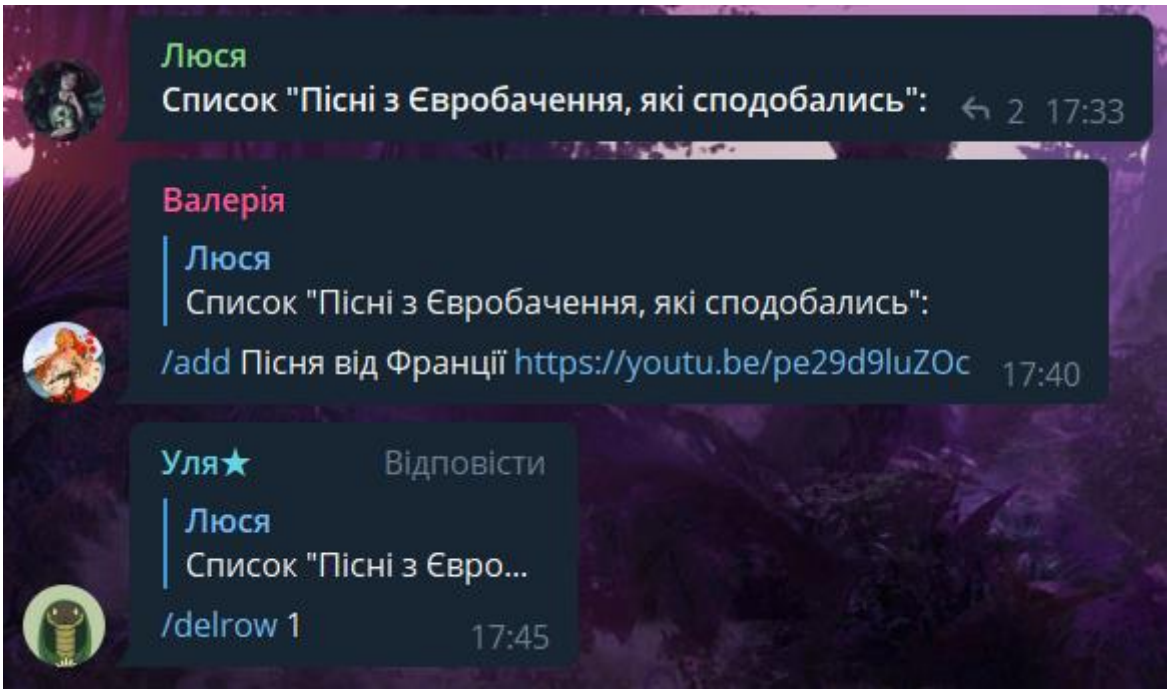

Рисунок 3.4 – Видалення зі списку

3.2 Тригери

Окрім спрощення роботи зі списками, застосунок надає можливість користувачам додавати в чат тригери. Це фіксовані відповіді бота на певні повідомлення в чаті. Повідомлення, на які необхідно надсилати реакції, в кожному чаті мають бути унікальними, тобто не можна додати два різних тригери на одну і ту ж фразу. Також вони не чутливі до регістру.

Для тригерів визначено наступні команди:

- +trigger <текст>
- -trigger <текст>
- /triggers

– /triggers_clear

Команда +trigger <текст> дозволяє користувачу додати тригер на вказаний текст. Цю команду необхідно надіслати в відповідь на інше повідомлення, яке і стане відповіддю бота на майбутні повідомлення в чаті з вказаним раніше текстом.

Рисунок 3.5 – Додавання тригеру

На противагу, команда -trigger <текст> видаляє з бази даних тригер на вказаний текст, якщо такий тригер існував.

У відповідь на команду /triggers користувач отримає список всіх активних тригерів чату. Навпроти кожного пункту списку буде позначено, який тип повідомлення буде відповіддю на вказаний тригер.

Також реалізована команда /triggers_clear для швидкого очищення всіх тригерів чату.

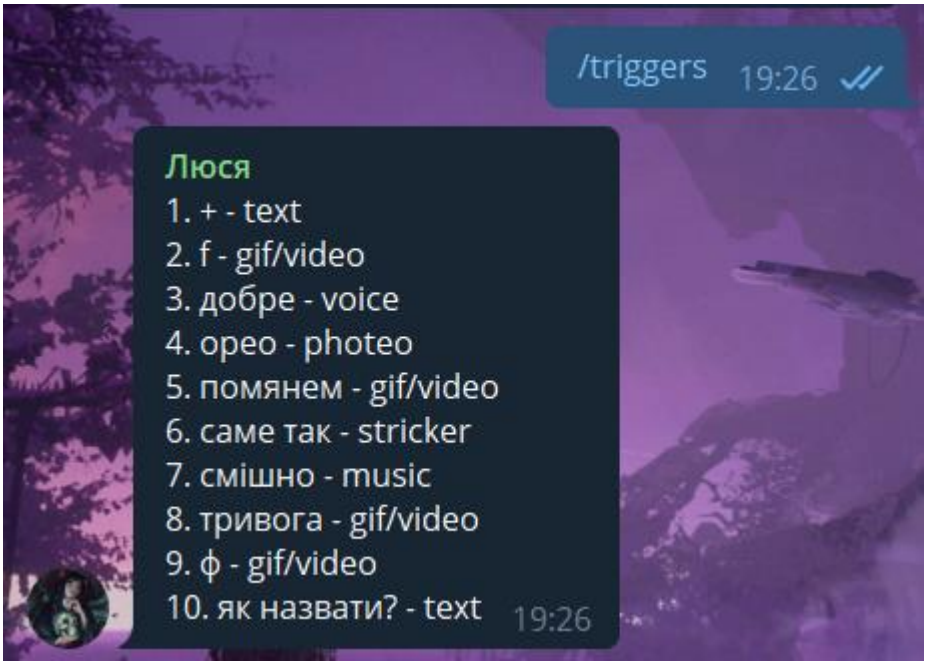

Рисунок 3.6 – Список тригерів чату

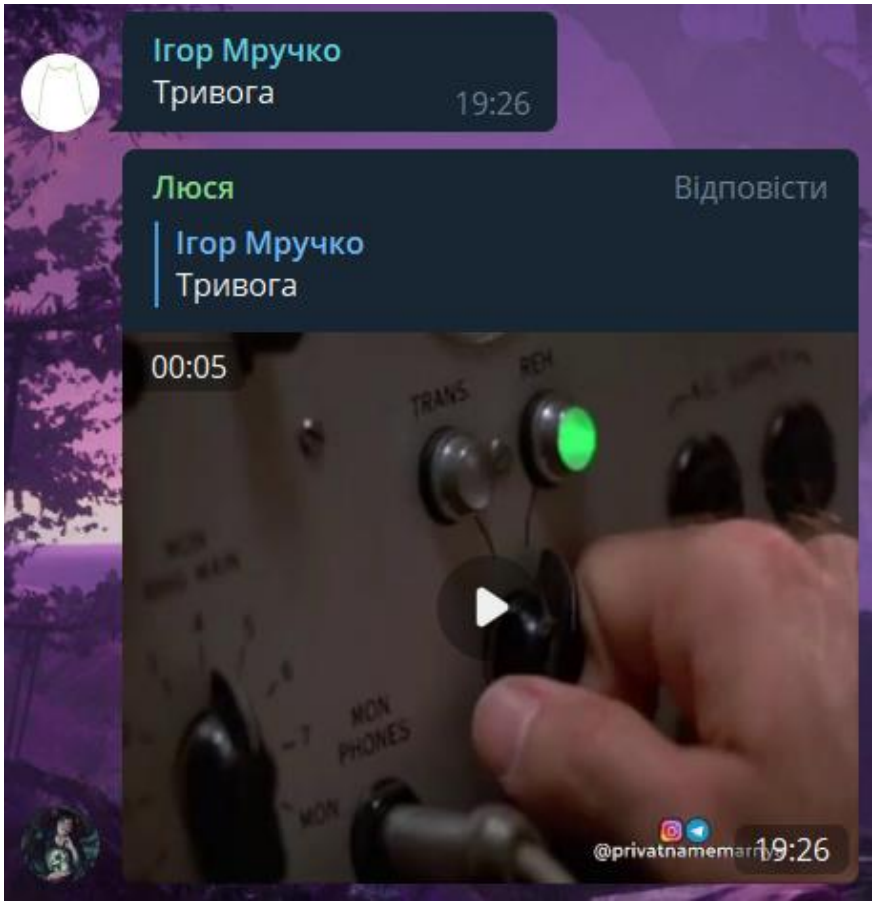

Рисунок 3.7 – Використання тригеру

3.3 Згадка всіх учасників чату

Для сповіщення всіх учасників чату реалізований набір команд:

- @all
- /addmetoall
- /removemefromall

Згадка кожного учасника чату активується, якщо в будь-якому з повідомлень групового чату, в якому присутній бот, зустрічається @all і біля нього стоять лише знаки пунктуації та/чи пробіли. В такій ситуації бот декількома повідомленнями, до п’яти людей в повідомленні, сповістить усіх учасників, крім тих, хто додав себе в винятки, інших ботів чату та користувача, на повідомлення якого відреагував бот.

За допомогою команд /removemefromall та /addmetoall користувач може додати себе в список винятків загальних згадок, або ж видалити себе з винятків та знову отримувати загальні сповіщення, відповідно.

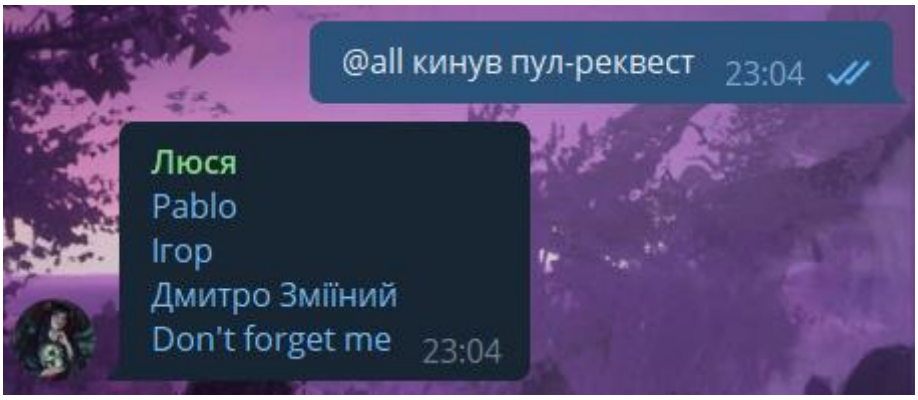

Рисунок 3.8 – Згадка учасників

3.4 Розіграш титулу «котик дня»

Наступна структурна одиниця функціоналу – це вибір «котика» дня. Один раз на день в кожному з групових чатів можна запустити розіграш, та обрати одного «котика». Також для кожного чату ведеться окремий рейтинг, який можна переглянути командою.

Реалізовані наступні команди для взаємодії:

- /reg
- /kotyk

– /kotyktop
– /kotykdel <id>

На початку, жоден з учасників чату не бере участі у розіграші титулу котика. Для цього охочим необхідно зареєструватися командою /reg. Після того, як двоє чи більше людей зареєструвались, будь-який учасник чату може використати команду /kotyk. Це запустить механізм вибору переможця та сповістить про це його. Для того щоб обрати переможця, чат-бот використовує звичайній вибір випадкового учасника із списку зареєстрованих.

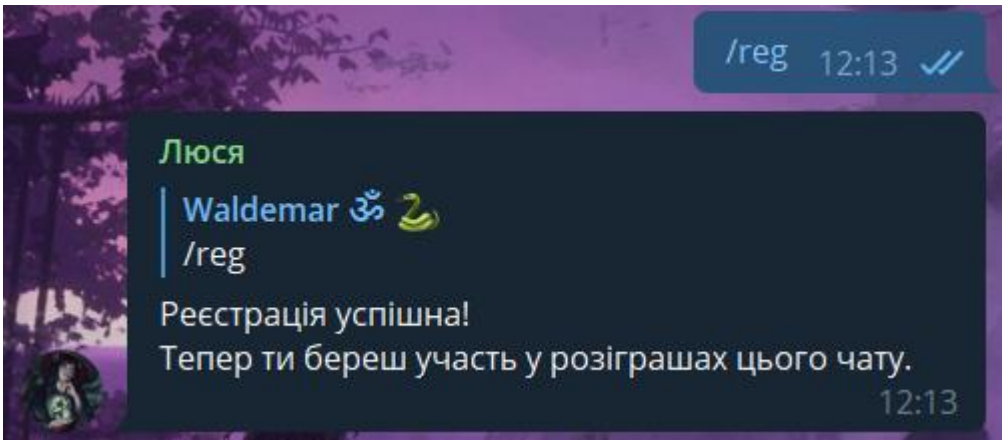

Рисунок 3.9 – Реєстрація учасника

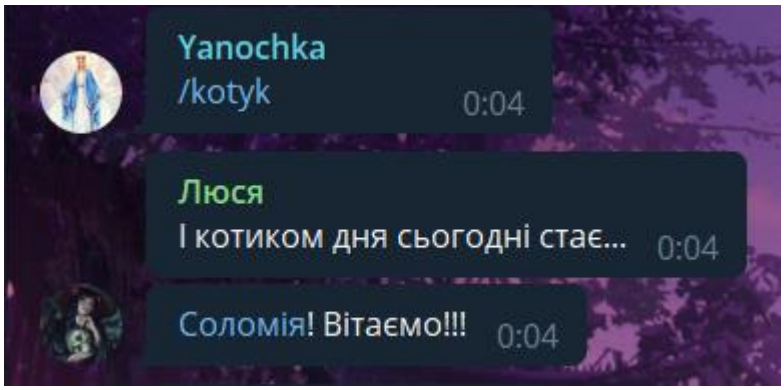

Рисунок 3.10 – Запуск розіграшу та його результати

За допомогою команди /kotyktop можна викликати локальний рейтинг, в якому можна побачити, який топ-10 котиків чату, скільки разів вони отримували цей титул та скільки всього учасників бере участь у розіграші.

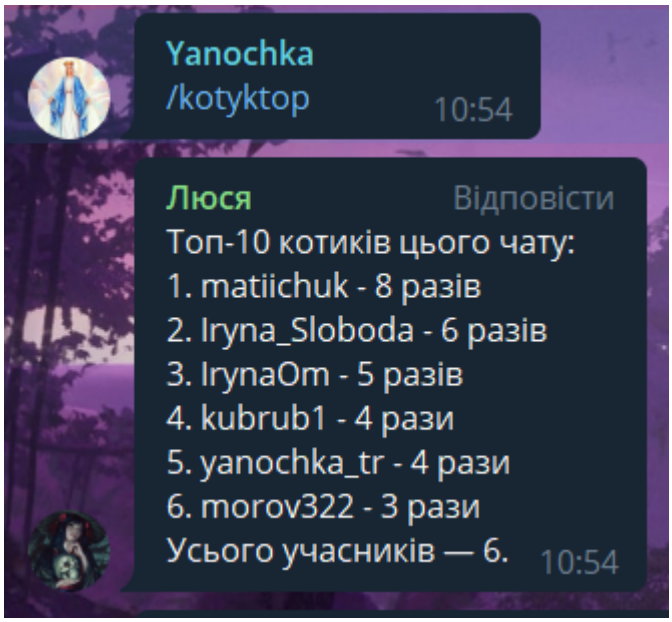

Рисунок 3.11 – Виклик топу

На противагу до загальнодоступних команд, /kotykdel можуть використовувати лише адміністратори чату. З його допомогою можна видалити користувачів з розіграшу, вказавши після команди необхідний id. Це можна використовувати як для очищення кандидатів від учасників, що покинули груповий чат, так і для видалення зі списку людей, які з тих чи інших причин вже не бажають брати участь у розіграші.

3.5 Інший функціонал

В цьому розділі описано функціонал, який не згруповано за спільним напрямком.

Для того, щоб отримати id учасників чату, адміністратор може використати команду /participants та отримати список всіх учасників з їх id в особистих повідомленнях з ботом.

Також, для створення кращої атмосфери в чаті, бот буде реагувати на приєднання нових учасників. Він буде вітати кожного новоприбулого учасника по імені. Таким чином кожен новий учасник буде тепло прийнятим в новому груповому чаті.

Якщо користувач знайшов помилку в роботі бота або ж має пропозицію щодо його покращення, він може надіслати свою думку розробнику завдяки команді /suggest. У відповідь на повідомлення з цією командою бот надішле подяку за ідею покращення. Також, користувач отримає відповідь, чи його ідея була реалізованою чи відхиленою.

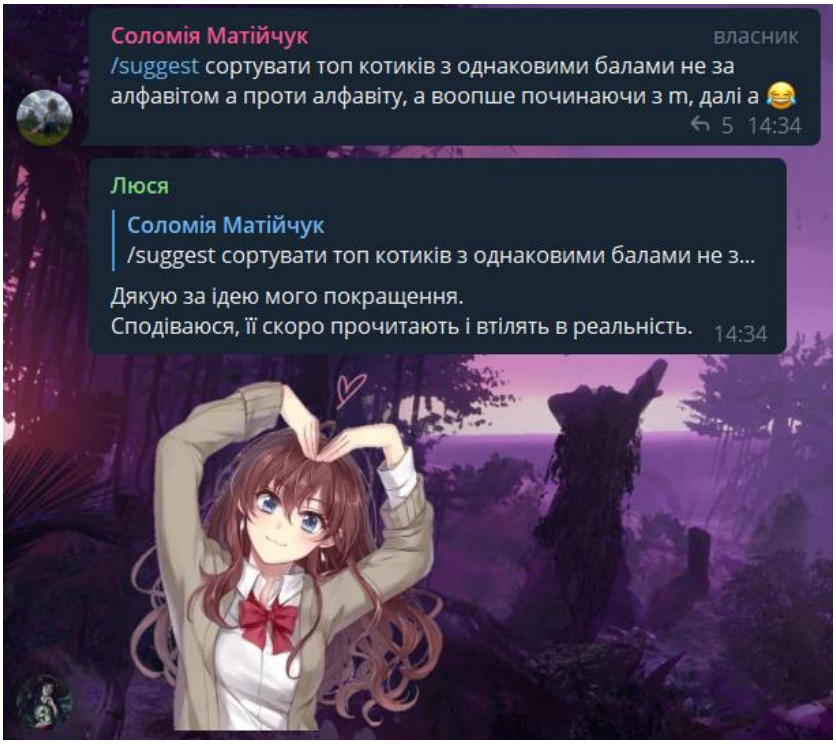

Рисунок 3.12 – Ідея для розвитку бота

РОЗДІЛ 4. ЗАСТОСУВАННЯ АЛГОРИТМУ АНАЛІЗУ ТЕКСТУ

Після розробки основної частини функціоналу чат-бота та впевненості, що він функціонує правильно, потрібно було приступити до вдосконалення бота. Оскільки однією з головних вимог було те, що чат-бот може реагувати не лише на команди, які були описані в попередньому розділі, але й на живе мовлення, то наступним кроком розробки було введення до застосунку модулю, що дозволяє аналізувати повідомлення.

4.1 Робота зі списками

Для вдосконалення цієї частини функціоналу потрібно визначити три основні можливості того, що може хотіти користувач:

- Створити список
- Додати до списку новий пункт
- Видали певний пункт із списку

4.1.1 Створення нового списку

Для створення нового списку потрібно проаналізувати повідомлення: чи хоче користувач створити новий список, чи просто говорить про списки, чи взагалі ніде їх не згадує. При другому та третьому випадках нічого робити не потрібно. Однак якщо користувач має бажання, щоб бот створив новий список, то потрібно з цього ж повідомлення зрозуміти, який заголовок має бути в цього списку.

Для цього можна скористатися властивістю, що після аналізу мовною моделлю головному слову назви (заголовку) призначається роль в реченні – flat:title. Знайшовши головне слово назви можна також знайти всі залежні від нього слова та зібрати їх в початковій послідовності, в якій вони знаходились в повідомленні.

Таким чином застосунок зчитує назву, яку необхідно надати списку, створює запис в базі даних про новий список та надсилає у відповідь користувачу повідомлення, в якому знаходиться заголовок пустого списку.

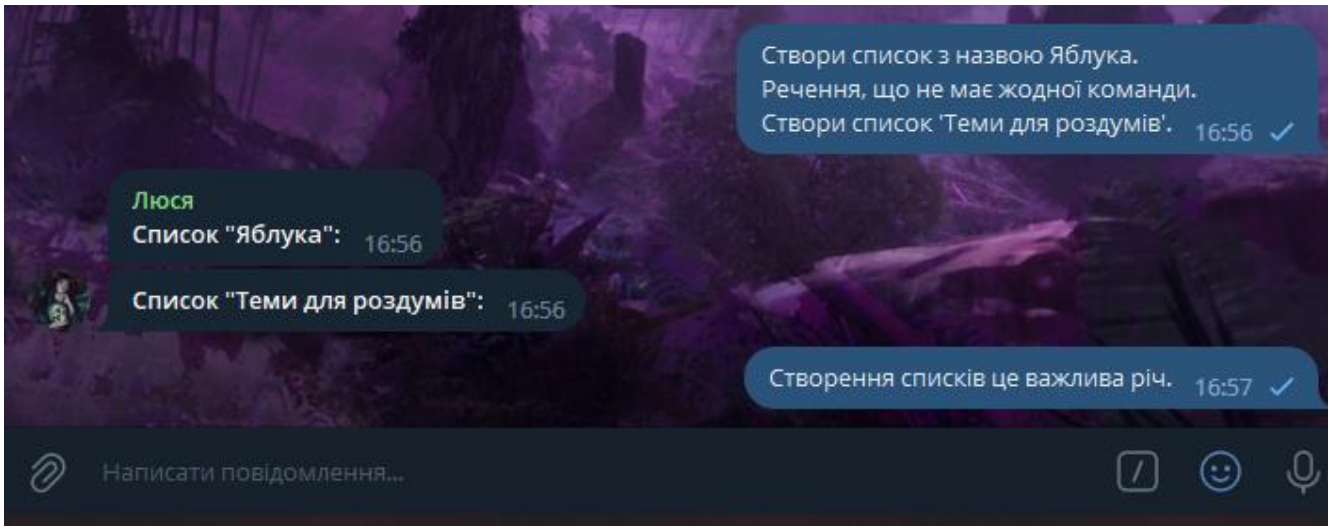

Рисунок 4.1 – Створення списку за допомогою речень та відсутність реакції на звичайне повідомлення, в якому є ключові слова

При цьому, кожне повідомлення розбивається на окремі речення. Тому можливі випадки, коли в одному повідомленні є декілька команд, які будуть виконані. Також, оскільки кожне таке речення аналізується окремо, то не всі речення обов’язково мусять виконувати команди.

4.1.2 Додавання пунктів до списків

Після того, як користувач створив список, він може додавати до нього пункти. Для цього йому все ще необхідно надсилати повідомлення у відповідь на вже створений ботом список. У випадку, якщо алгоритм сприйме це як команду додавання пункту, він спробує знайти заголовок (як і для створення списку) та відтворить головне слово заголовку разом зі всіма залежними від нього словами.

Також назву пункту, що необхідно додати, можна знайти за допомогою залежності слів у реченні. Назва пункту завжди буде залежною від слова ‘пункт’, якщо таке зустрічається в тексті.

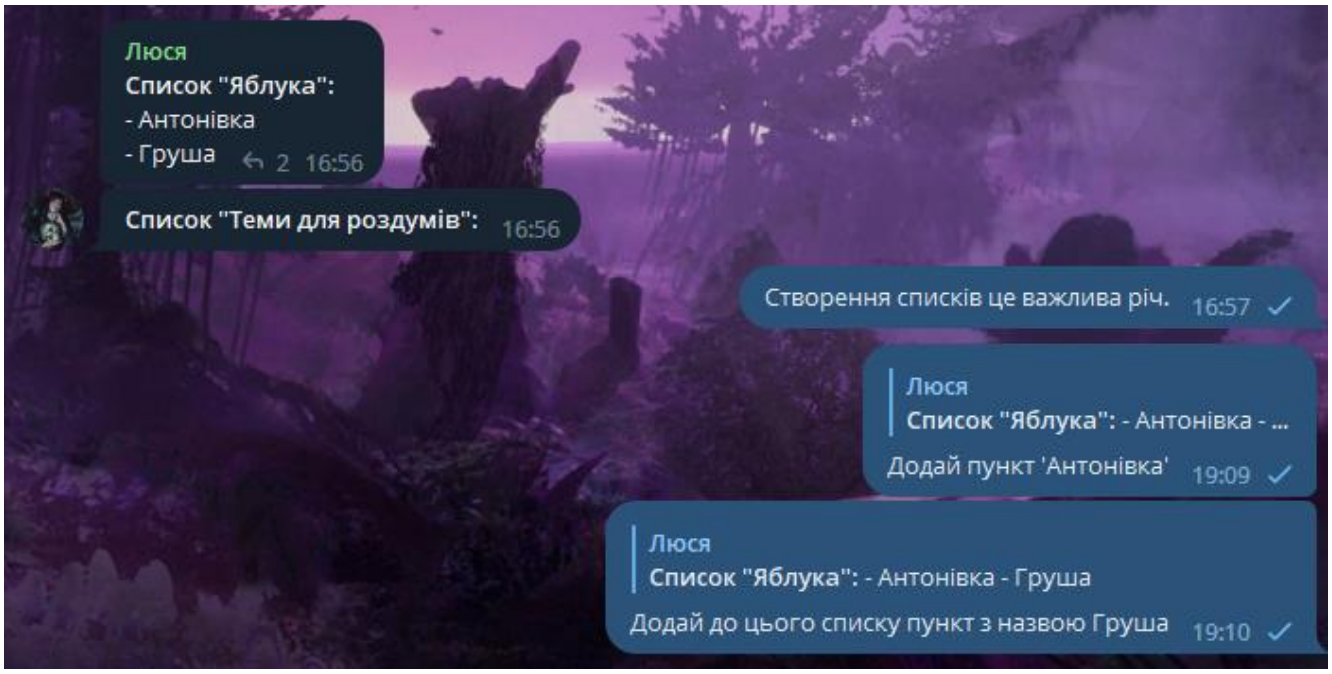

Рисунок 4.2 – Додавання до списку нових пунктів

4.1.3 Видалення пунктів списку

Оскільки користувач може з деяких причин, наприклад, таких як помилкове додавання пункту або ж виконання пункту зі списку щоденних справ, хотіти видали певний рядок зі списку, необхідно також надати таку можливість у формі запиту, а не команди.

Як і в випадку з додаванням до списку, користувачу потрібно відповісти на повідомлення з створеним раніше списком, який має необхідний рядок. Для того, щоб зрозуміти, що користувач хоче видалити певний рядок зі списку, потрібно щоб в його повідомленні було слово, лемою якого є ‘видалити’. Також потрібно щоб в повідомленні було числове представлення рядка, що необхідно видалити.

Також, як і в застосуванні командою, залишається можливість видалення пункту списку, який додав раніше інший користувач.

4.2 Тригери

Так само як і для створення списку, при аналізі повідомлення, в якому користувач просить створити новий тригер, буде знайдено слово, роль в реченні якого є flat:title. Оскільки назва, або ж заголовок, до тригеру є необхідний для його

виклику, то це є обов’язковою умовою для виклику команди створення нового тригеру.

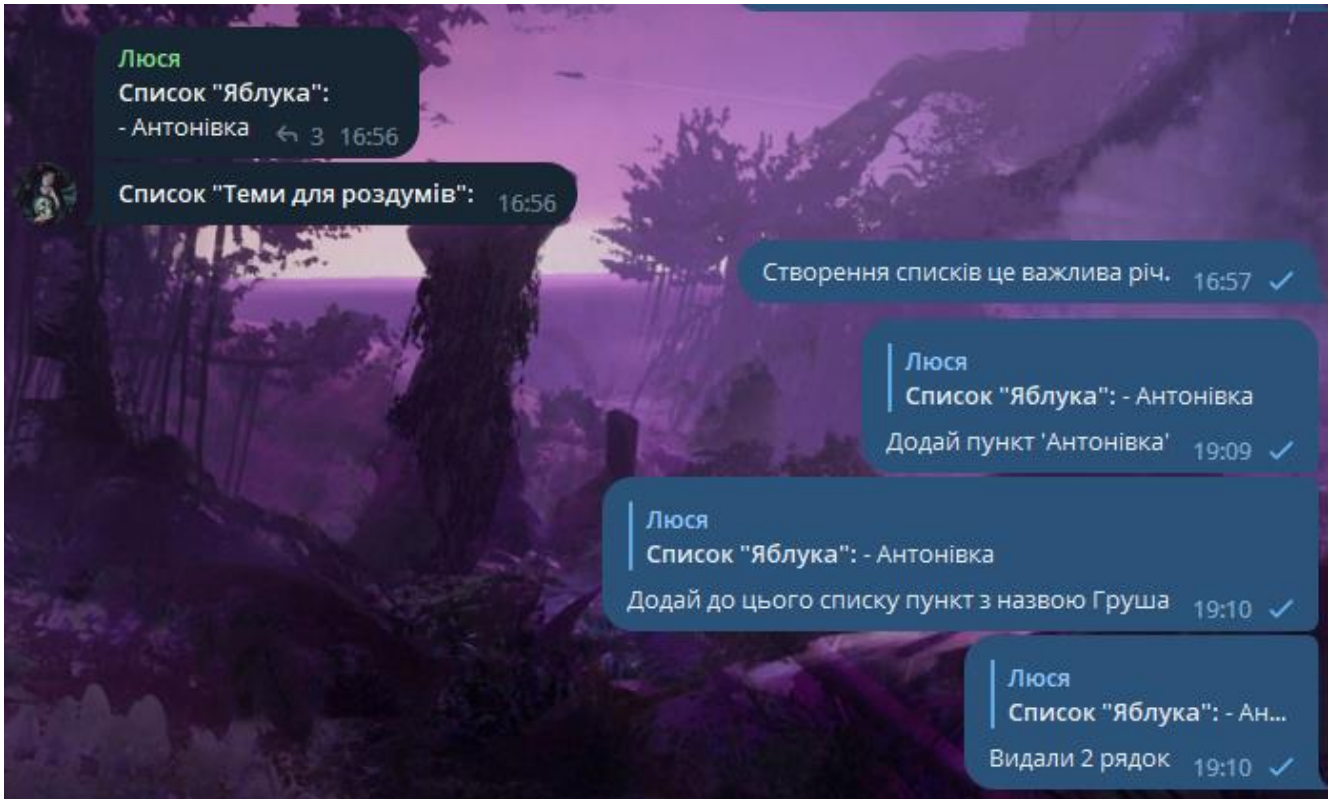

Рисунок 4.3 – Видалення помилково доданого пункту

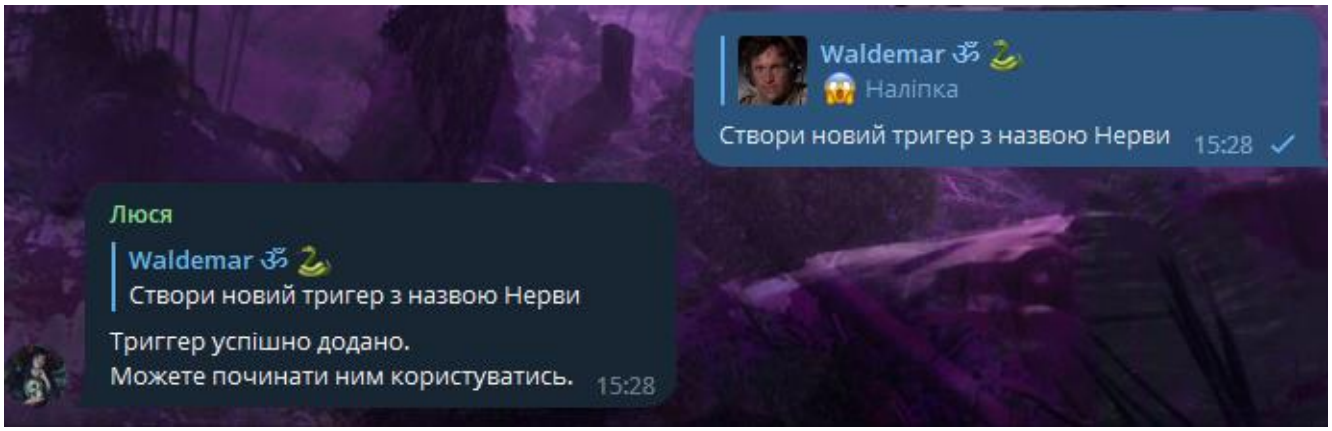

Рисунок 4.4 – Створення нового триггера

Знайшовши у повідомленні з проханням додати чи створити новий тригер головне слово назви, тобто фрази для якої викликатиметься цей тригер, алгоритм вибере всі залежні від головного слова слова та реконструює їх в окреме речення. Це речення і буде заголовком / тригером.

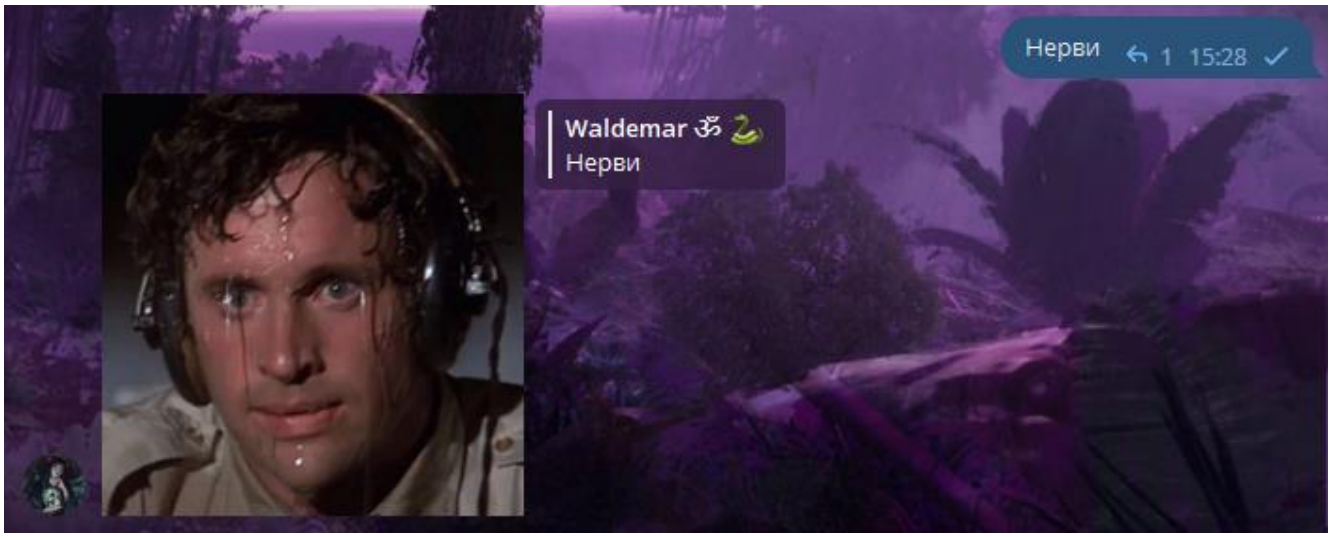

Рисунок 4.5 – Перевірка роботи триггера

Також користувачі чату можуть мати потребу у перегляді всіх тригерів, доступних в цьому чаті. Знайшовши шаблони, якими може користуватись учасник чату, можна вивести правила, за якими повідомлення буде сприйматися як команда до друку всіх доступних тригерів та видів повідомлень, що буде надіслано у відповідь на них.

Рисунок 4.6 – Виведення списку всіх триггерів

4.3 Розіграш титулу «котик дня»

Для вдосконалення цієї категорії функціоналу потрібно визначити наступні можливі запити користувачів:

- Реєстрація в розіграші
- Отримання інформації про котика дня
- Перегляд рейтингової таблиці

4.3.1 Реєстрація в розіграші

Для того, щоб зрозуміти, що користувач бажає зареєструватися в розіграші, алгоритму необхідно знайти в повідомленні слова з такими лемами: «зареєструвати», «я», «розіграші», «котик».

Також, мають збережені залежності слів у повідомленні: слово «мене» (лема якого «я») має бути залежним від «зареєструй» (лема це «зареєструвати»). При цьому згадка розіграшу чи котика лише підсилює впевненість у тому, що це повідомлення має бути інтерпретоване як команда.

Оскільки база даних для зареєстрованих користувачів одна на весь застосунок, то не має різниці чи користувач реєструвався звичайною командою, чи за допомогою звичного для нього повідомлення. При повторній спробі зареєструватись виведеться відповідне повідомлення.

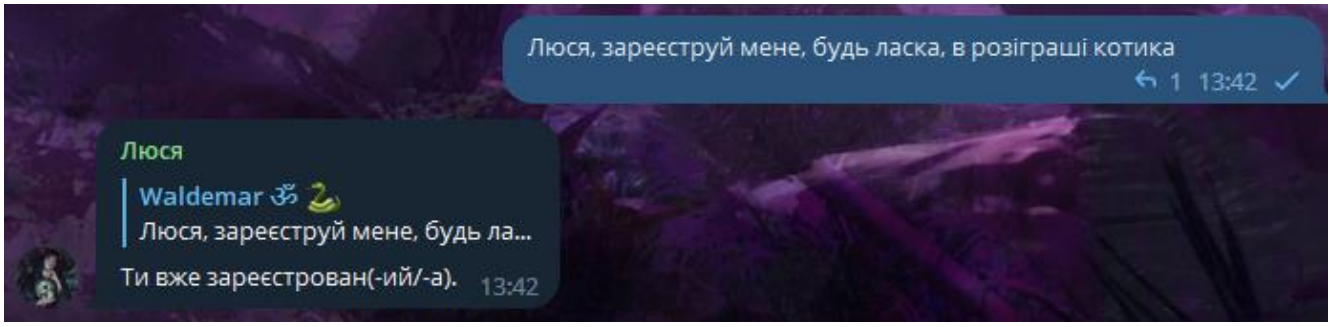

Рисунок 4.7 – Запит для реєстрації в розіграші

4.3.2 Результат розіграшу

Для того, щоб дізнатися який результат розіграшу титулу на сьогоднішній день в конкретному чаті, користувачу необхідно задати необхідний запит у вигляді повідомлення. Коли алгоритм інтерпретує повідомлення як команду до включення розіграшу, він або обирає нового переможця, якщо такого ще не було, або виводить інформацію про те, що титул уже було розіграно.

Як і у випадку з реєстрацією, через об’єднану базу даних неважливо яким саме методом викликали розіграш – результат буде однаковим для кожного чату в кожний окремий день.

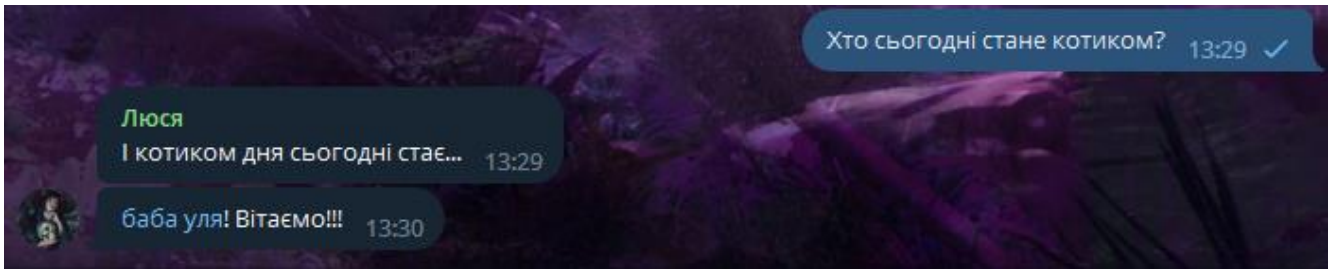

Рисунок 4.8 – Розіграш титулу

Рисунок 4.9 – Повідомлення про те, що переможця вже обрано

4.3.3 Рейтинг переможців

Останнім можливим запитом користувачів, що стосується розіграшу, може бути прохання показати рейтинг чату. При проханні показати/вивести рейтинг програма проаналізує, чи це не випадкове вживання ключових слів у розмові. Якщо ж це таки буде інтерпретовану як команду – у відповідь на запит буде виведено рейтинг попередніх розіграшів цього чату.

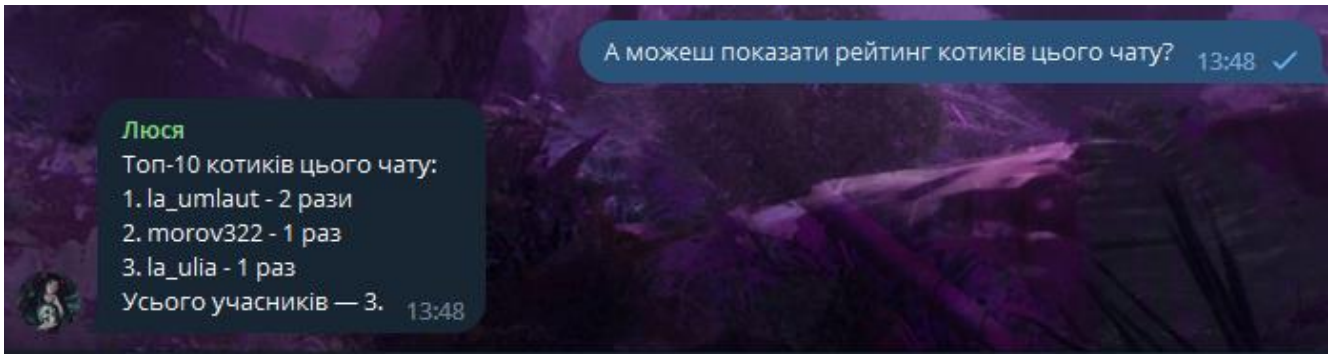

Рисунок 4.10 – Виведення рейтингу

4.4 Додатковий функціонал

Для створення відчуття живого співрозмовника до програми було додано можливість зміни стану настрою. В кожен момент часу чат-бот може перебувати в одному з п’яти настроїв, кожен з яких можна репрезентувати числом на шкалі від -2 до 2:

- Лють (-2)
- Роздратованість (-1)
- Нейтральність (0)
- Безтурботність (1)
- Піднесення (2)

Залежно від того, в якому стані настрою перебуває чат-бот, деякі частини його функціоналу можуть змінюватись. Передбачено, що від настрою мають залежати привітальні повідомлення, шанс на реакцію на окремі повідомлення, загальний настрій цих реакцій, реакція на вживання імені бота та її шанс і т.д.

На теперішній момент, оскільки генерації тексту до програми ще не додано, єдине на що впливає настрій зараз – це можливість чат-бота забрати собі титул «котик дня».

Кожен настрій має свій базовий шанс присвоєння титулу собі. Спочатку програма з допомогою випадкового значення буде перевіряти, чи цей шанс виконується. Якщо результат перевірки негативний, тоді застосовується звичний метод випадкового вибору одного зареєстрованого учасника.

Також залежно від настрою змінюється повідомлення, яким супроводжується присвоєння титулу чат-боту.

Зміна настрою відбувається через аналіз повідомлень, які може бачити чат-бот. За допомогою додаткового модуля бібліотеки SpaCy можна визначити який загальний настрій несе повідомлення. Цей показник знаходиться в межах [-1; 1], що інтерпретується як [негативне; позитивне] повідомлення. Для того, щоб визначити, чи зміниться настрій, використовується випадкова змінна, яка порівнюється з значенням показника загального настрою повідомлення.

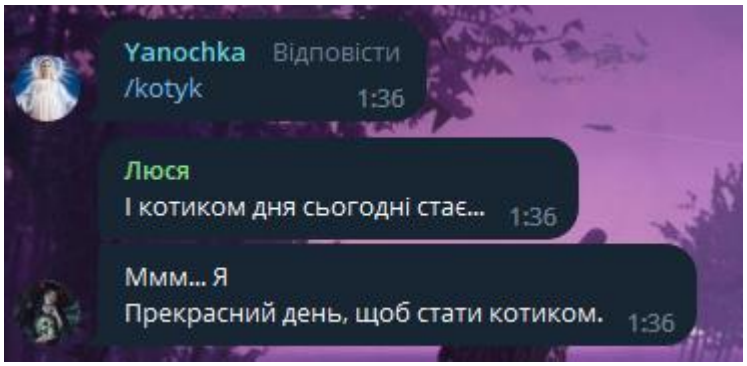

Рисунок 4.11 – Обрання чат-бота на титул «котик дня»

Якщо модуль випадкової змінної менший ніж модуль настрою повідомлення, то настрій чат-боту змінюється у відповідному напрямку: погіршується при негативному настрої повідомлення і покращується при позитивному. Така закономірність означає, що чим ближче настрій повідомлення до нейтрального (0), тим менший шанс зміни настрою чат-бота.

ВИСНОВКИ

Реалізувавши програму для чат-бота в месенджері Telegram було отримано застосунок, що може допомагати в групових чатах згаданого месенджера різним категоріям людей. Також він може частково виконувати розважальну роль.

За допомогою використання NLP було отримано можливість надати доступ до користування бота не лише з використанням чітко визначених наперед команд. Використання бібліотеки SpaCy дозволило аналізувати повідомлення для того, щоб при знаходженні в них бажання користувача скористатись однією з можливостей бота, застосунок сам виконував відповідну команду, а не чекав на знаходження користувачем відповідної команди.

За допомогою використання доповнення до згаданої бібліотеки, було отримано можливість аналізувати загальний настрій повідомлень. Це в свою чергу дало можливість створити умови для введення до застосунку системи зміни настрою, що наближує його до більш людяної поведінки при спілкуванні.

Хоча для створення повноцінного чат-бота окрім розуміння запитів користувача необхідно також мати можливість генерувати змістовні тексти у відповідь, створений для дипломної роботи застосунок ще не має такої можливості. Це, звісно, обмежує його можливості, але водночас залишає місце для вдосконалення застосунку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Functional vs Non Functional Requirements [Електронний ресурс]: Режим доступу: https://www.geeksforgeeks.org/functional-vs-non-functional-requirements.
2. Writing simple Telegram bot with Telethon [Електронний ресурс]: Режим доступу: https://bmwlog.pp.ua/writing-simple-telegram-bot-with-telethon/.
3. Telethon’s Documentation [Електронний ресурс]: Режим доступу: https://docs.telethon.dev/en/stable/index.html.
4. SpaCy Documentation [Електронний ресурс]: Режим доступу: https://spacy.io/api.