Київський національний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка засобу автоматизації пошуку літератури в бібліотеці на основі SQL SELECT»

Студента 4 курсу, 13 групи,

спеціальності
122 «Комп'ютерні науки»

Колесник Дмитро
Володимирович

_____
підпис студента

Науковий керівник
кандидат економічних наук, доцент

_____
підпис керівника

Шклярський
Сергій
Михайлович

Гарант освітньої програми
кандидат технічних наук, доцент

_____
підпис керівника

Демідов Павло
Георгійович

Київ 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую

Зав. кафедри _____ Пурський О.І.
«20» грудня 2020р.

Завдання
на випускний кваліфікаційний проект студенту

Колеснику Дмитру Володимировичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)

«Розробка засобу автоматизації пошуку літератури в бібліотеці на основі SQL SELECT»
Затверджена наказом ректора від *«15» грудня 2020 р. № 4111*
2. Строк здачі студентом закінченої роботи *31 травня 2021 року*
3. Цільова установка та вихідні дані до роботи
Мета роботи: розробка програмного засобу у вигляді чат-боту для прискорення пошуку літератури в бібліотеці КНТЕУ.
Об'єкт дослідження: процес проектування та створення чат-боту .
Предмет дослідження: засоби створення системи пошуку літератури в бібліотеці КНТЕУ в Telegram.
4. Перелік графічного матеріалу_____
_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.

2	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.
3	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.

6. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. Сучасні технології в розробці чат-ботів

1.1. Дослідження ринку чат-ботів

1.2. Порівняння аналогів та визначення проблем

Висновки до розділу

РОЗДІЛ 2. Постановка задач та методи їх дослідження

2.1. Мета та задачі

2.2. Методи реалізації

2.3. Моделювання використання чат– бота

Висновки до розділу

РОЗДІЛ 3. Розробка чат-бота

3.1. Розробка логічної структури чат-бота

3.2. Опис функціонування чат-бота

Висновки до розділу

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

Додатки

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	01.10.2020	01.10.2020
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	15.12.2020	15.12.2020
3	*Вступ*	03.02.2021	03.02.2021
4	*РОЗДІЛ 1. Сучасні технології в розробці чат-ботів*	26.02.2021	26.02.2021
5	*РОЗДІЛ 2. Постановка задачі та методи дослідження*	06.04.2021	06.04.2021
6	*РОЗДІЛ 3 Розробка чат-боту*	12.05.2020	12.05.2020
7	*Висновки*	15.05.2021	15.05.2021
8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	20.05.2021	20.05.2021
9	*Попередній захист випускної кваліфікаційної роботи*	26.05.2021	26.05.2021
11	*Виправлення зауважень, зовнішнє*	27.05.2021	

	рецензування випускної кваліфікаційної роботи		
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	31.05.2021	
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «15» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)
Шклярський С.М._____
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г._____
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник
Колесник Д.В.
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи (проекту)

Випускна кваліфікаційна робота студента Колесника Д.В. присвячена актуальній темі проектування чат-ботів для мобільних пристроїв, зокрема ботів, які підтримують взаємодію користувача з віддаленим інформаційним ресурсом.

В якості об'єкта дослідження було вибрано процеси проєктування чат-боту для пошуку книжок в бібліотечному фонді.

В ході розробки студент провів достатньо якісний аналіз існуючих підходів до вирішення проблеми, запропонував та обґрунтував власну розробку, що базується на сучасних досягненнях в області комп'ютерних наук.

Виконана розробка має практичний інтерес, відповідає вимогам Вищої школи до робіт кваліфікаційного рівня "Бакалавр", а студент Колесник Д.В. показав рівень що є достатнім для позитивної оцінки роботи.

Керівник випускної кваліфікаційної роботи_____ доц. Шклярський С.М.

13. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота (проект) студента _____
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____ Демідов П.Г.
(підпис, прізвище, ініціали)

Завідувач кафедри _____ Пурський О.І.
(підпис, прізвище, ініціали)
«_____»_____2021 р.

Анотація

У випускній кваліфікаційній роботі здійснено комплексну розробку моделей та інформаційного засобу автоматизації пошуку літератури в бібліотеці на основі SQL SELECT з метою підвищення швидкості пошуку літератури в бібліотеці КНТЕУ. Теоретично обґрунтовано основні положення формування та створення чат-боту для пошуку літератури. Розроблено метод пошуку літератури в базі даних, надання інформації через месенджер. Створено книжкового чат-бота для месенджера Telegram.

Ключові слова: чат-бот, пошукова система, пошук літератури.

Anotation

The graduation qualification work is devoted to development of model and information SQL SELECT-based library search automation tool in order to increase the speed of searching for literature in the library of KNTEU. The main provisions of the formation and creation of a chatbot for searching literature are theoretically substantiated. A method for searching literature in a database and providing information through a messenger has been developed. Created a book chat bot for Telegram messenger.

Keywords: chatbot, search engine, literature search.

ЗМІСТ

ВСТУП.....7
РОЗДІЛ 1. Сучасні технології в розробці чат-ботів.....8
1.1Дослідження ринку чат-ботів.....9
1.2 Порівняння аналогів та визначення проблем.....13
Висновки по розділу.....15
РОЗДІЛ 2. Постановка задачі та методи їх дослідження.....16
2.1 Мета та задачі.....16
2.2 Методи реалізації.....18
2.3 Моделювання використання чат-бота.....21
Висновок по розділу.....24
РОЗДІЛ 3. Розробка чат-бота.....25
3.1 Розробка логічної структури чат-бота.....25
3.2 Опис функціоналу чат-бота.....33
Висновок по розділу.....37
ВИСНОВОК.....38
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....39
Додаток А.....40

ВСТУП

Актуальність. Чат-бота можна кваліфікувати, як особистого помічника, який спілкується з користувачем за допомогою повідомлень і має багато специфічних функцій. Чат-бота можна використовувати, як для спілкування так і для розваг, від надання медичних консультацій до замовлення товарів та послуг за допомогою спеціалізованих прикладних рішень, від розпізнавання емоцій до рішення складних консалтингових задач в службах підтримки клієнтоорієнтованих інформаційних системах.

Месенджери перетворились із засобів для спілкування в засоби отримання інформації чи потужний інструмент в руках маркетингу. Велику частку в цьому відіграли боти. На даний момент є безліч варіацій, від ботів для отримання новин до ботів для замовлення їжі. На такий корисний інструмент звісно з'являється попит у власників бізнесу. В Україні вони не так поширені як за кордоном, тому являються перспективною нішею для розробників. В наш час більшість сервісів підходять під автоматизацію роботи за допомогою ботів, багатьом компаніям вигідно мати замість постійного працівника автономну систему, яка буде швидко, дешево та ефективно виконувати поставленні задачі. Бібліотека як раз підходить під можливість автоматизації. В час коли у кожного є смартфон і можливість доступу до інтернету є набагато зручнішим перегляд книжок не виходячи з дому, особливо коли це проста система з різноманітними можливостями.

На сьогоднішній день месенджер Telegram має великий попит, кількість користувачів дішла до відмітки 500 мільйонів і зростає з дуже швидким темпом, тому це є перспективною платформою для створення чат-бота.

Мета. Розробка програмного засобу у вигляді чат-боту для швидкого пошуку літератури в бібліотеці КНТЕУ.

Предмет дослідження. Засоби створення системи пошуку літератури в бібліотеці КНТЕУ в Telegram

Об'єкт дослідження. Процес проектування та створення чат-боту.

РОЗДІЛ 1. Сучасні технології в розробці чат-ботів

1.1Дослідження ринку чат-ботів

Чат-бот це програмний додаток , що використовується для проведення онлайн-чат розмови за допомогою тексту, замість забезпечення прямого контакту з людиною. Створені для переконливої імітації поведінки людини як партнера для розмов, системи чат-ботів, як правило, вимагають постійного налаштування та тестування, і багато хто на виробництві залишаються нездатними адекватно розмовляти або проходити стандартний тест Тьюрінга . Термін "ChatterBot" був спочатку введений Майклом Молдіном (творцем першого Verbot) в 1994 році для опису цих розмовних програм[1].

Чат-боти використовуються в діалогових системах для різних цілей, включаючи обслуговування клієнтів, маршрутизацію запитів або збір інформації. Хоча деякі додатки чат-ботів використовують великі процеси класифікації слів, складний штучний інтелект, інші просто сканують загальні ключові слова та генерують відповіді, використовуючи загальні фрази, отримані з підключеної бібліотеки чи бази даних.

Чат-боти багатьох компаній працюють на додатках для обміну повідомленнями або просто через SMS. Вони використовуються для обслуговування клієнтів роздрібну торгівлю, продажів та маркетингу.

У 2016 році Facebook Messenger дозволив розробникам розміщувати чат-боти на своїй платформі. За перші шість місяців було створено 30000 ботів для Messenger, які до вересня 2017 року зросли до 100000[1].

Боти зазвичай виступають як один із контактів користувача, але іноді можуть виступати учасниками групового чату. Багато банків, страхові компанії , медіа - компанії, електронно комерційні компанії, авіакомпанії, готельні мережі, продавці, працівники охорони здоров'я, державні установи та ресторанні мережі використовували чат-ботів для надання відповіді на прості запитання, збільшення залучення клієнтів, для просування і запропонування додаткових

способів замовлення їхніх послуг. Для компаній він стає каналом просування або частиною рекламної компанії, як для великих брендів так і для новачків.

На сьогоднішній день чат-боти стали невід'ємною частиною комунікаційних засобів, вони стали широко використовуватись в більшості сферах життєдіяльності людини. Завдяки масовості і обширності інструменту віртуальної комунікації з'явились можливості дізнаватись про спеціальні послуги онлайн, отримувати актуальні новини та вигідних пропозицій послуг, здійснювати покупки та продаж товарі.

Створення чат-бота стало не дуже складною процедурою особливо коли з'явились конструткори такі, як «Manybot, Botbot, Bottap, Botmaker, та інші». За допомогою таких кострукторів створення займає набагато менше часу, та для цього не потрібно знати мови програмування, але за допомогою конструкторів не можливо створити більш гнучку програму, задати унікальну функцію, налаштувати його ідеально під певну задачу. Тому навіть створення досить простого бота заточеного під певну задачу потребує написання коду.

Судячи по поточному стану ринка, чат боти, особливо для месенджерів дуже перспективне направлення, яке в даний час переживає бурний ріст. Цей зріст пов'язаний з тим, що користувачів месенджерів перевищило кількість користувачів соціальних мереж.

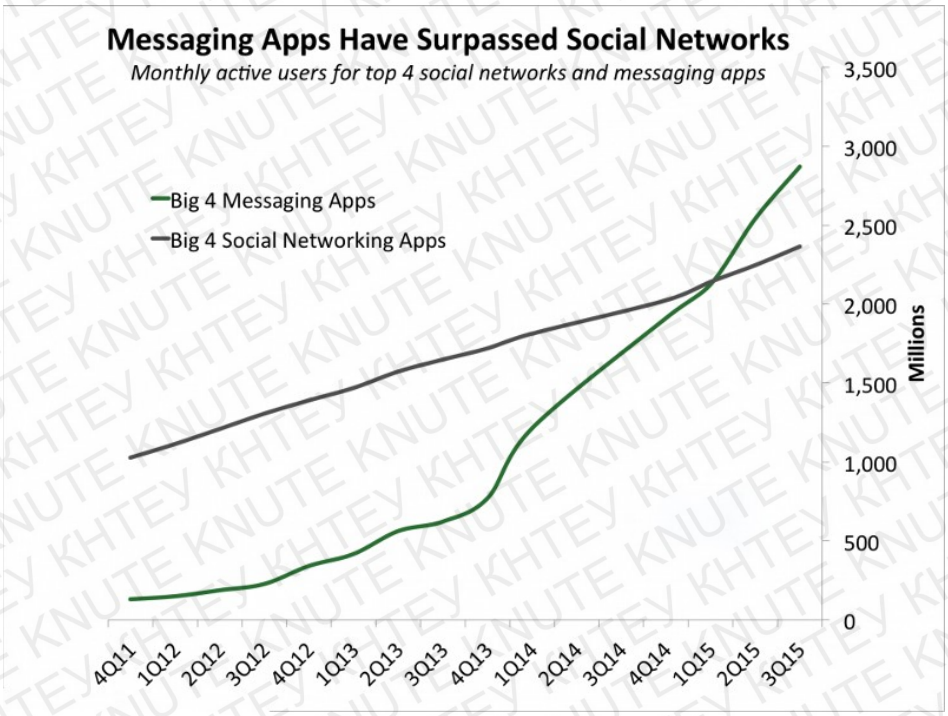

Рисунок 1.1 – Тенденція ринку чат-ботів

В найближчому майбутньому, чат-боти матимуть все більше значення. Для прикладу вони можуть повністю замінити пошукові двигуні і соціальні мережі. Використовування ботів значно спрощує взаємодію з сервісами, надаючи універсальний інтерфейс [2].

Рисунок 1.2 – Класифікація чат-ботів

По функціоналу можна виділити наступні види чат-ботів

- Чат-бот для продажу – консультує покупців і допомагає підібрати потрібний товар, повідомляє про стан замовлення, розповідає про акції та знижки.
- Лід-бот – збирає данні відвідувачів сайту, пропонує записатись на демонстрацію продукту.
- Транзакційний-бот – виконує різноманітні транзакції: розміщення замовлення, резервування, грошові переводи.
- Бот-інформатор – відповідає на запити, надає інформацію про варіанти авіа-перельотів, цінах, та інших.

- Чат-бот для підтримки – допомагає в питаннях використовування продукту чи послуги.

- Бот-асистент – інтегрується з іншими платформами і допомагає користувачу вирішити відразу декілька задач, такі як пошук в Google, встановлення нагадувань, відбір новин.

Чат-боти допомагають компаніям спілкуватись з своїми клієнтами. Вони дають користувачам можливості швидко отримати потрібні інструкції, не очікуючи відповіді оператора. Чат-боти заощаджують час працівників і беруть на себе прості комунікації, а користувачі отримують відповіді на свої запити швидше[3].

Основні переваги використання чат-ботів:

- Допомагають примножити об’єм продажу. Вони можуть запропонувати потрібний товар, боті з штучним інтелектом консультують покупця і запам’ятовують їхні вподобання.

- Допомагають скорочувати витрати. Чат-боти дають можливість компаніям обслуговувати більше клієнтів, і при цьому скорочувати витрати на персонал. За даними дослідження Acquire, 80% запитів вирішуються чат-ботами без участі людини.

- Працюють 24/7. У чат-ботів немає неробочих годин і вихідних днів.

- Працюють швидше співробітників. У користувачів є можливість миттєво отримати консультацію без очікування оператора.

- Автоматизують прості комунікації. Чат-боти вирішують прості і однотипні завдання, залишаючи співробітникам тільки складні і цікаві.

- Збирають данні. Чат-боти можуть задавати питання і генерувати ліди.

Таким чином компанія отримує більше даних про замовників, що дозволяє спілкуватися з ними на новому рівні - заздалегідь знаючи, що їх цікавить, і які у них є завдання і потреби.

1.2 Порівняння аналогів та визначення проблем

Для створення якісного продукту потрібно ознайомитись з ринком чат-ботів для того щоб виділити основні функції та можливості. Кожен бот який відповідає інформацією можна назвати інформаційним, тобто бот який надає інформацію на запити. Чат-бот який розробляється має вузьке і актуальне направлення, особливо в наш час коли публічні місця являються загрозую. За осонву було взяту просту механіку бібліотеки.

Книжкових ботів на просторі месенджера досить багато, кожен з яких має багато різних функцій. Для порівняння та визначення напрямку розробки проекту було вибрано список самих популярних книжкових чат ботів.

Bookinator (@bookinator_bot) простий бот який замінює додаток для читання книги, своя домашня бібліотека в месенджері, для його роботи достатньо завантажити файл книги та за допомогою команд перемикатись між сторінками та керувати своєю бібліотекою, зображений на рисунку 1.2.

Рисунок 1.3 – Чат-бот Bookinator

Аудиокниги (@aud_books_bot) бот за допомогою якого можна знаходити та прослуховувати аудіокниги, Для того щоб знайти те що вам потрібно, достатньо ввести назву книги або автора, зображений на рисунку (1.3).

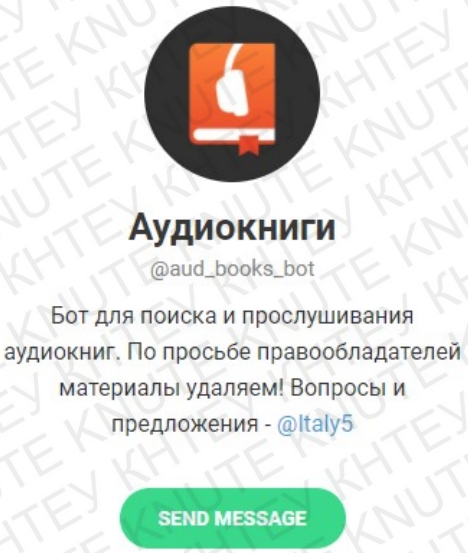

Рисунок 1.4 – Чат-бот Аудіокнига

Flibustafreebookbot (@flibustafreebookbot) бот за допомогою якого можна швидко знайти книгу по її назві, також є можливість завантажити її в декількох форматах, зображений на рисунку 1.4.

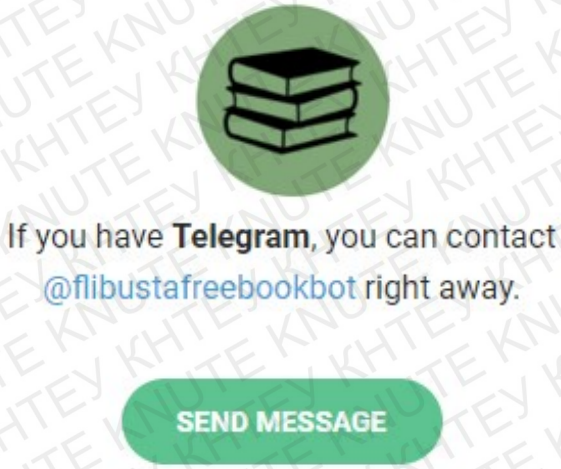

Рисунок 1.5 – Чат-бот Flibustafreebookbot

AlignedBooks (@DualBookBot) Бот за допомогою якого можна вчити англійську за допомогою метода паралельного читання. В його базі вже є певна кількість книг яких можна вибрати для читання, зображений на рисунку 1.5.

Рисунок 1.6 – Чат-бот AlignedBooks

Ці боти займають топ 5 книжкових ботів по версії «Telegram Blog»[4]. Крім цих ботів існують інші, деякі мають схожі функції, деякі мають свої унікальні. Всі вони знаходяться у вільному доступі.

Зробивши аналіз по представленим ботам, був вибраний основний концепт напряму для реалізації проекту, це має бути простий для користування та перспективний в розширенні чат-бот з можливостями швидкого пошуку книжок по автору чи по назві книги. Також в ньому повинна бути зручна база даних з швидким додаванням нових книг, для збільшення асортименту. Чат-бот підійде любому користувачу у якого в наявності месенджер Telegram та любов до читання книг.

Висновки по розділу: Чат-боти – це дуже перспективне направлення. Їх активне використання в месенджерах і в якості «цифрових асистентів» в смартфонах з великою ймовірністю призведе до популярізації UX-парадигми messaging-as-an-interface. Уже зараз ключові вендори (Facebook, Apple, Google) займаються розробкою чат-ботів персональних помічників.

Створення комерційного чат-бота (наприклад, для розвантаження онлайн-підтримки чи консультантів, при відповідях на найбільш популярні запитання) не потребує важких технологій, достатньо буде базових технологій оброки мови.

РОЗДІЛ 2. Постановка задачі та методи їх дослідження

2.1 Мета та задачі

Мета роботи – розробити книжкового чат-бота в месенджері Telegram. Судячи з аналізу самих популярних книжкових ботів, було вибрано певний список функцій які буде мати чат бот:

- база даних з можливістю простого внесення нових книг;
- команда для пошуку книги по назві;
- команда для пошуку книги по назві автора;
- команда яка видає користувачу випадкову книгу з бази даних.

Цей бот матиме широкий простір для розширення свого функціоналу. Ці можливості являються необмеженими. Невеликий список можливих вдосконалень:

- глибокий пошук книги по таким параметрам як - дата написання, рядки з книги, жанр, мова, пошук з перекладом;
- можливість вибору мови бота та мови пошуку книг;
- додавання можливості завантаження книг з бази даних;
- створення історії прочитаних книг та бібліотеки книг на майбутнє;
- рекомендації користувачу по прочитаним книгам;
- додавання рейтингової системи оцінювання книг для формування топів найкращих книг.

Існуючі аналоги дали зрозуміти, що дана розробка буде актуальною для повсякденного використання та в майбутньому зможе замінити класичний похід до бібліотеки.

При збільшені трафіку чат-бота, можливе підключення монетизації, за допомогою якої можна отримати кошти в залежності від популярність та кількості користувачів.

Як було вище зазначено, проект розробляється для месенджера Telegram. Telegram Messenger – це безкоштовна кросплатформенна програма для смартфонів, планшетів та ПК, яка дозволяє обмінюватися текстовими повідомленнями й медіафайлами. Являється аналогією таких програм як Facebook Messenger, WhatsApp, Viber. Для того щоб зареєструватись, достатньо завантажити додаток, чи зайти через браузер в веб клієнт Telegram, та ввести свій мобільний номер, на який прийде перевірочний код. Спілкуватись можна як з конкретним користувачем, так і в організованому груповому чаті, який може створити кожен з користувачів. Також в месенджері присутні різноманітні групи з різним контентом, наприклад: новини, цікаві історії, навчальні статті, та інші.

За допомогою Telegram ви можете надсилати повідомлення, фотографії, відео та файли будь-якого типу (doc, zip, mp3 тощо). В месенджері є можливість писати на контакти телефону та знаходити людей за їхніми іменами користувачів. Як результат, Telegram схожий на SMS та електронну пошту - і він може подбати про всі особисті або ділові потреби щодо обміну повідомленнями. Також є підтримка наскрізних зашифрованих голосових та відеодзвінків , а також голосових чатів в групах для тисяч учасників[5].

Для месенджера був створений протокол MTProto, що передбачає використання декількох протоколів шифрування. Під час авторизації і аутентифікації використовуються алгоритми RSA-2048, DH-2048 для шифрування, під час передачі повідомлень протоколу в мережу вони шифруються AES з ключем, відомим клієнту і серверу. Також застосовуються криптографічні хеш-алгоритми SHA-1 і MD5[6].

Завдяки захищеності месенджера і захисту від перехоплення, додаток став популярним в Гонконзі під час протестів. Спочатку про мітинг домовлялися через Facebook, але цю інформацію легко дістали.

2.2 Методи реалізації

Для реалізації проекту було вибрано мову програмування Python, тому що реалізація такого функціоналу досить складна і не підвладна конструкторам чат-ботів.

Python - це високорівнева мова програмування загального призначення, яка використовується в тому числі і для розробки веб-додатків. Мова орієнтована на підвищення продуктивності розробника і читання коду.

Python підтримує кілька парадигм програмування: структурну, об'єктно-орієнтовану, функціональну, імперативну і аспектно-орієнтовану. У мові присутня динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень і зручні високорівневі структури даних. Програмний код на Python організовується у функції та класи, які можуть об'єднуватися в модулі, а вони в свою чергу можуть бути об'єднані в пакети. Python зазвичай використовується як інтерпретуємий, але може бути скомпільовано в байт-код Java і в MSIL (в рамках платформ .NET)[7].

В цій мові програмування низький поріг входження – це значить, що для написання готових продуктів потрібно затратити менше часу на навчання чім в C++ або Java, а код лаконічний і простий для розуміння і читання. Написані проекти за рахунок простоти коду в подальшому його супровід стає легше і приємніший[8]. З точки зору написання проекту, ця мова програмування є продуктивнішою і вона зменшує затрати ресурсів які вимірюються в людино-годинах.

Також ця мова програмування була вибрана із за наявності зручної бібліотеки pyTelegramBotAPI. Ця бібліотека заміняє код на ключові функції за допомогою яких достатньо описувати реакцію бота на різні дії користувача.

Для написання коду використовується середовище програмування JetBrains PyCharm Community Edition. Це інтегроване середовище розробки для мови програмування Python. Надає засоби для аналізу коду, графічний зневаджувач, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django.

Рисунок 2.1 – JetBrains PyCharm

З використанням PyCharm на багато прискорюється напис коду за рахунок систем резервованих блоків та конструкцій, також дуже зручне автозаповнення. До всіх системних команд є анотації.

Для зберігання всієї інформацію про книги, які є в наявності чат-бота, використовується база даних SQL. SQL – це мова структурних запитів, яка використовується для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними базами даними, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування (C або Java), SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних.

Перша версія SQL була розроблена на початку 1970-х років у IBM. Ця версія мала назву SEQUEL і була призначена для обробки та пошуку даних, що містилися в реляційній базі даних IBM, System R. Мова SQL надалі була стандартизована Американськими Держстандартами (ANSI) в 1986. На початку

SQL була запланована як мова запитів і управління даними, а пізніші модифікації SQL створені продавцями системи управління базами даних, які додали процедурні конструкції, control-of-flow команд і темпоральні розширення мов. З випуском стандарту SQL:1999 такі розширення були формально запозичені як частина мови SQL через Persistent Stored Modules (SQL/PSM)[9].

Переваги використання SQL:

1. Незалежність від конкретної СУБД .Не зважаючи на наявність діалектів і відмінностей в синтаксисі, більшість текстів SQL-запитів, що містять, DDL і DML, можуть бути досить легко перенесені з однієї СУБД в іншу. Існують системи, розробники яких спочатку орієнтувалися на застосування щонайменше кількох СУБД (наприклад: система електронного документообігу Documentum може працювати як з Oracle, так і з Microsoft SQL Server та IBM DB2). Природно, що при застосуванні деяких специфічних для реалізації можливостей, такого рівня перенесення дуже важко досягти.

2. Наявність стандартів. Наявність стандартів і наборів тестів для виявлення сумісності та відповідності конкретній реалізації SQL загальноприйнятому стандарту тільки сприяє «стабілізації» мови. Щоправда, слід звернути увагу на той факт, що сам по собі стандарт місцями занадто формалізований і має завеликі розміри, наприклад, Core-частина стандарту SQL:2003 містить понад 1300 сторінок тексту.

3. Декларативність. За допомогою SQL програміст описує лише дані, які потрібно витягнути або модифікувати. Яким саме чином це зробити, вирішує СУБД безпосередньо при обробці SQL-запиту. Не слід вважати, що це повністю універсальний принцип — програміст описує набір даних для вибірки або модифікації, проте йому корисно уявляти, як СУБД інтерпретуватиме текст його запиту. Такі моменти стають особливо критичними при роботі з великими базами даних та зі складними запитами — чим складніше сконструйований запит, тим більше варіантів виконання він припускає. Ці варіанти можуть дуже відрізнятися за швидкістю виконання та використаними ресурсами, хоча результат (набір даних) має бути однаковим.

В проекті для створення бази даних використовується вбудована бібліотека «sqlite3» яка при першому запуску програми створює файл бази даних. І дозволяє писати запити безпосередньо в скриті.

2.3 Моделювання використання чат-бота

Моделювання полегшує вивчення об'єкта з метою його створення, подальшого перетворення і розвитку. Воно використовується для дослідження існуючої системи, коли реальний експеримент проводити недоцільно через значні фінансові і трудові витрат, а також при необхідності проведення аналізу проектованої системи, тобто яка ще фізично не існує в даній організації.

Модель повинна враховувати якомога більше число факторів. Однак реалізувати таке положення складно особливо в слабкоструктурованих системах. Тому найчастіше прагнуть створювати моделі досить простих елементів, з урахуванням їх мікро- і макрозв’язків. Це дозволяє отримувати адекватні результат.

Для представлення усіх процесів, взаємодії користувача з чат-ботом та можливих варіантів розвитку подій всередині самої системи, необхідно побудувати контекстну діаграму IDEF0 та її декомпозицію, а також діаграму прецедентів.

Для початку необхідно побудувати контекстну діаграму IDEF0. Головним процесом є створення книжкового чат-боту. На вході маємо API telegram для доступу до створення чат-боту, а також технічне завдання для його розробки. Механізмами виступають мова програмування python, а також мова структурних запитів SQL. Процеси створення регулюються документацією telegram та бібліотеки pyTelegramBotAPI.

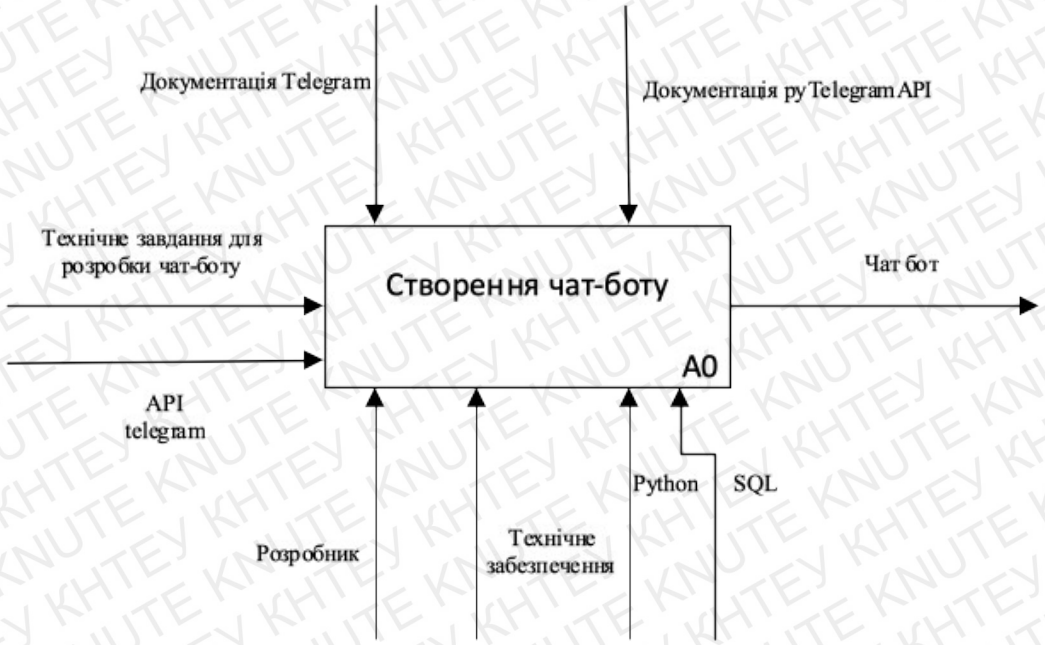

Рисунок 2.2 - Діаграма IDEF0

Для повного опису системи цієї діаграми не достатньо, тому що це контекстна діаграма з загальним описом. Щоб побачити повний опис потрібно зробити декомпозицію, за допомогою чого можна більше ознайомитись з структурою системи.

Блок створення програми був розбитий на дві дії: отримання доступу до telegram, створення алгоритму.

Перший етапу формування даних:

- Вхідні данні: технічне завдання для розробника.
- Вихідні данні: отримання зв’язку з telegram.
- Управління: документація Telegram.
- Механізми: розробник, технічне забезпечення, python.

Другий етап формування даних:

- Вхідні данні: отримання зв’язку з telegram, API telegram.
- Вихідні дані: чат-бот.
- Управління: документація Telegram, документація pyTelegramBotAPI.
- Механізми: розробник, технічне забезпечення, python, SQL.

Діаграма другого рівня представлена на рисунку 2.3

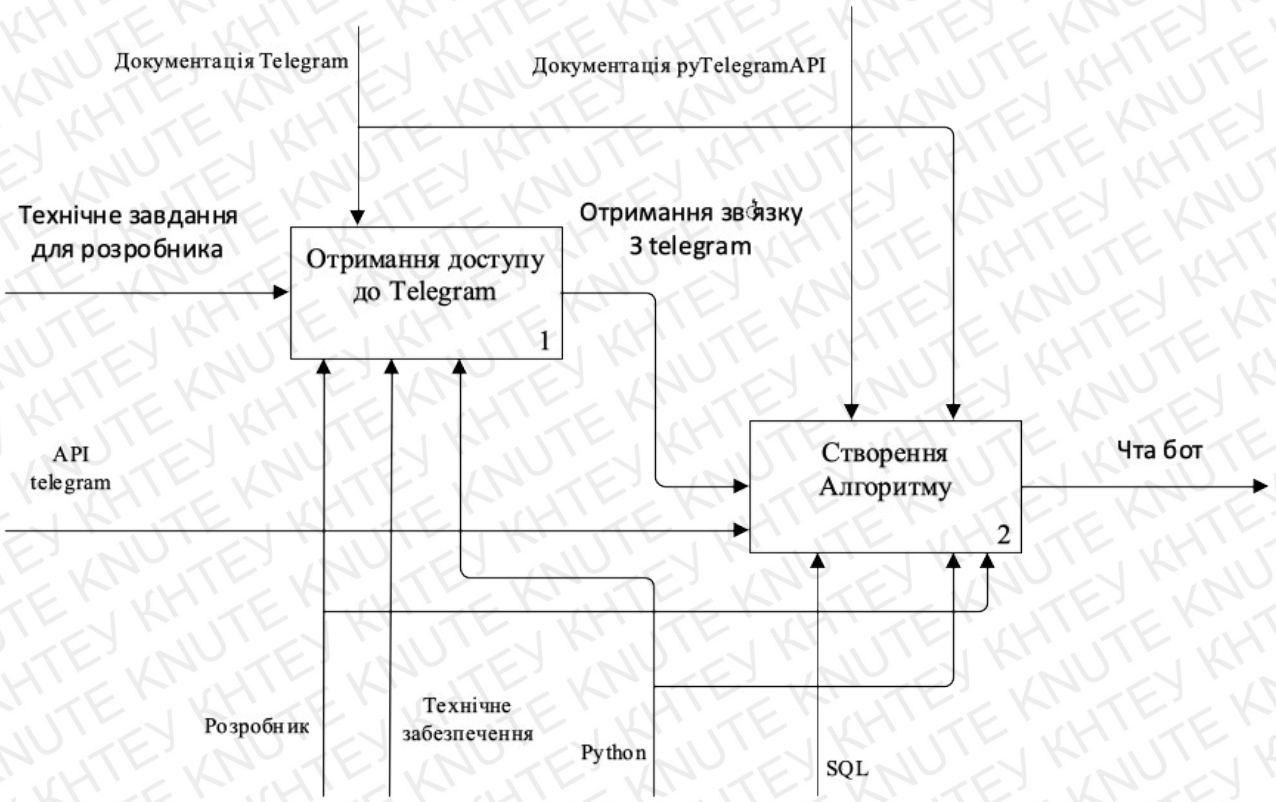

Рисунок 2.3 - Діаграма діаграма другого рівня

Для позначення різних частин систем програмного забезпечення використовується конструкція UML. Вона може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм.

Для розробки діаграми прецедентів були визначені такі актори:

- User – користувач, він може користуватись всіма наданими йому командами через бота і використовувати отриману інформацію.
- Admin – розробник, повністю може маніпулювати чат-ботом, змінюючи любі його параметри, команди, та вміст БД.

Після визначення всіх акторів, потрібно сформувати список можливих взаємодій з системою. Вони будуть визначені на основи проведення аналітки існуючих ботів на ринку.

Список взаємодій з чат-ботом:

- Telegram
- Редагування вмісту БД
- Додавання нових книг в БД

- Пошук по назві книги
- Пошук по назві автора
- Отримання випадкової книги
- Отримання випадкової книги

На основі сформованих даних про акторів та всі можливі варіанти взаємодії з чат-ботом, була сформована діаграма прецедентів, яка представлена на рисунку 2.4.

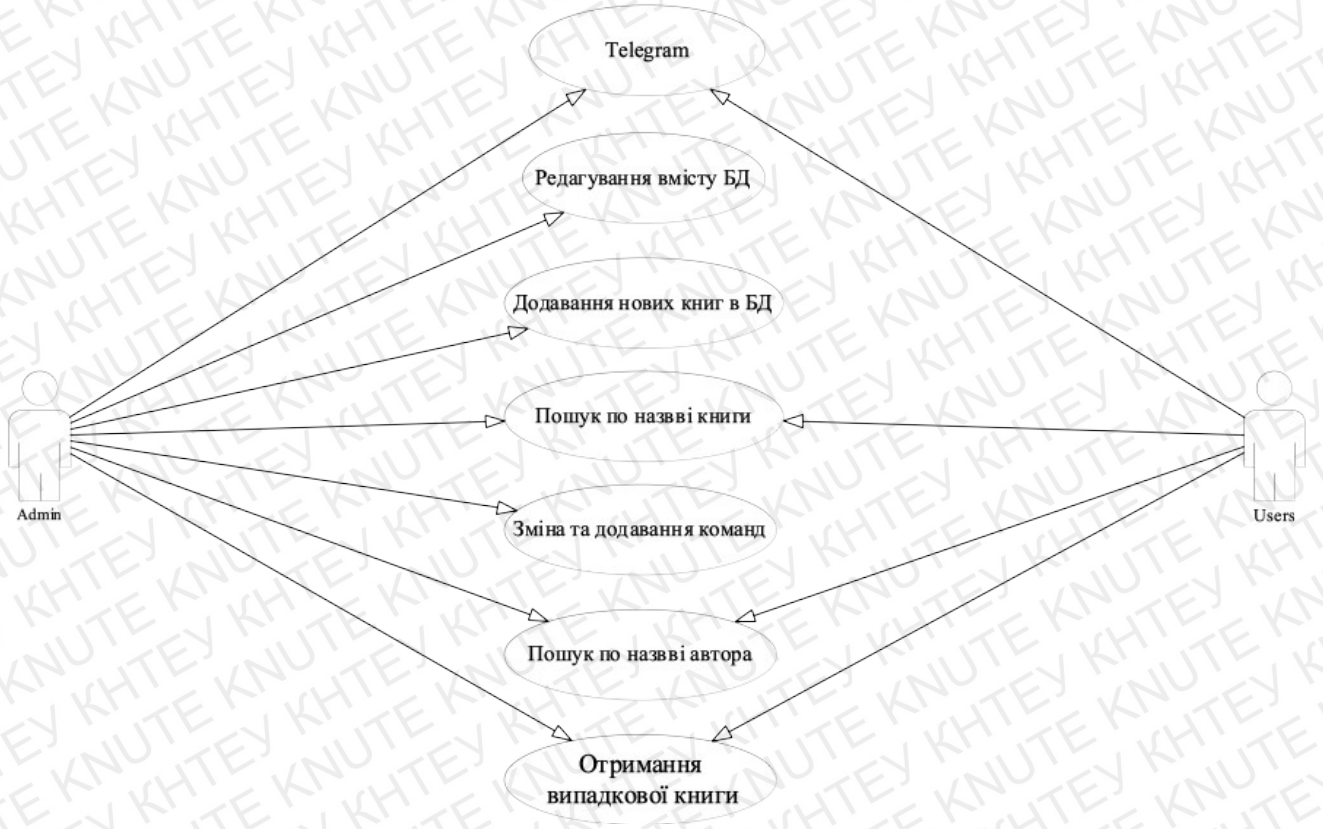

Рисунок 2.4 – Діаграма варіантів використання чат-боту

Висновок по розділу: Якісний аналіз і проектування є необхідними умовами успішного розроблення нетипових інформаційних систем. Саме на початкових стадіях визначаються фундаментальні аспекти моделювання системи і рішення, від яких значною мірою залежить успіх проекту. Зазначена послідовність розроблення моделей є каркасом, який може бути нарощений залежно від типу та призначення інформаційної системи. Послідовне і якісне розроблення кожної з моделей є чинником успішного завершення проектування ІС.

РОЗДІЛ 3. Розробка чат-бота

3.1 Розробка логічної структури чат-бота

Даний проект являється досить простим у використанні чат-ботом, для відображення його логічної структури була збудована блок-схема, яка зображена на рисунку 3.1.

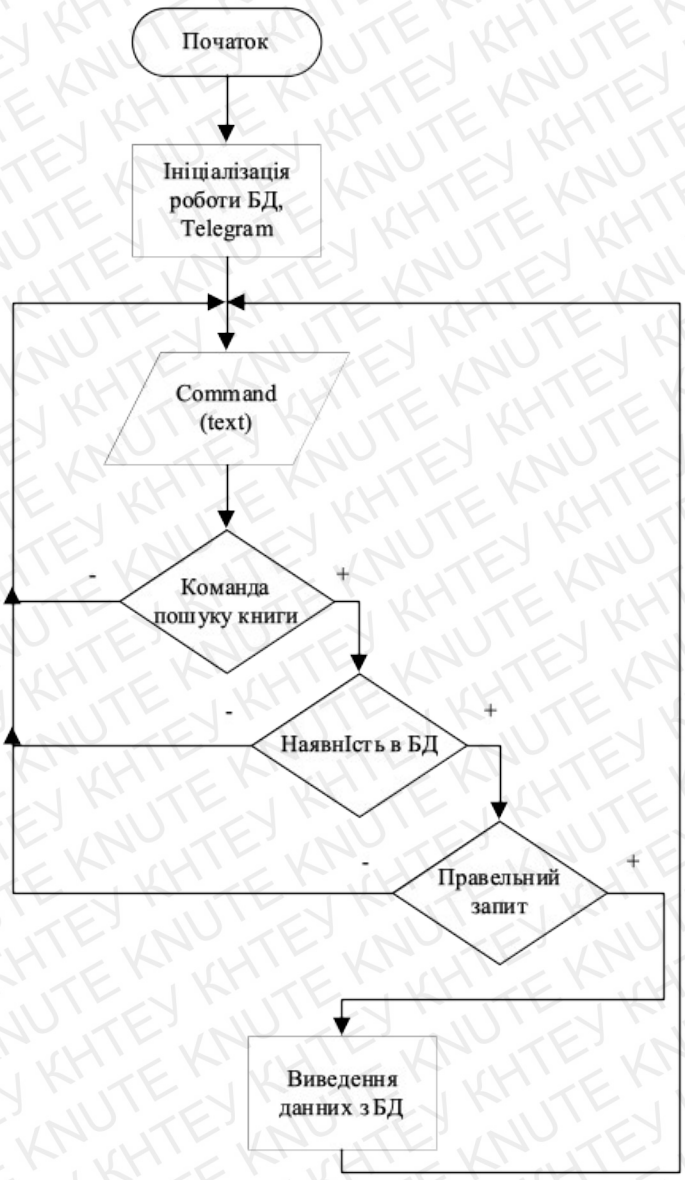

Рисунок 3.1 – логічна блок-схема

Для того, щоб створити бота в telegram та отримати доступ до його API ключа, потрібно звернутись до «бота папи» BotFather. Для початку роботи в поле

для повідомлення потрібно вписати команду /start. Після привітання, виводиться список всіх команд для того, щоб створити та налаштувати свого чат-бота.

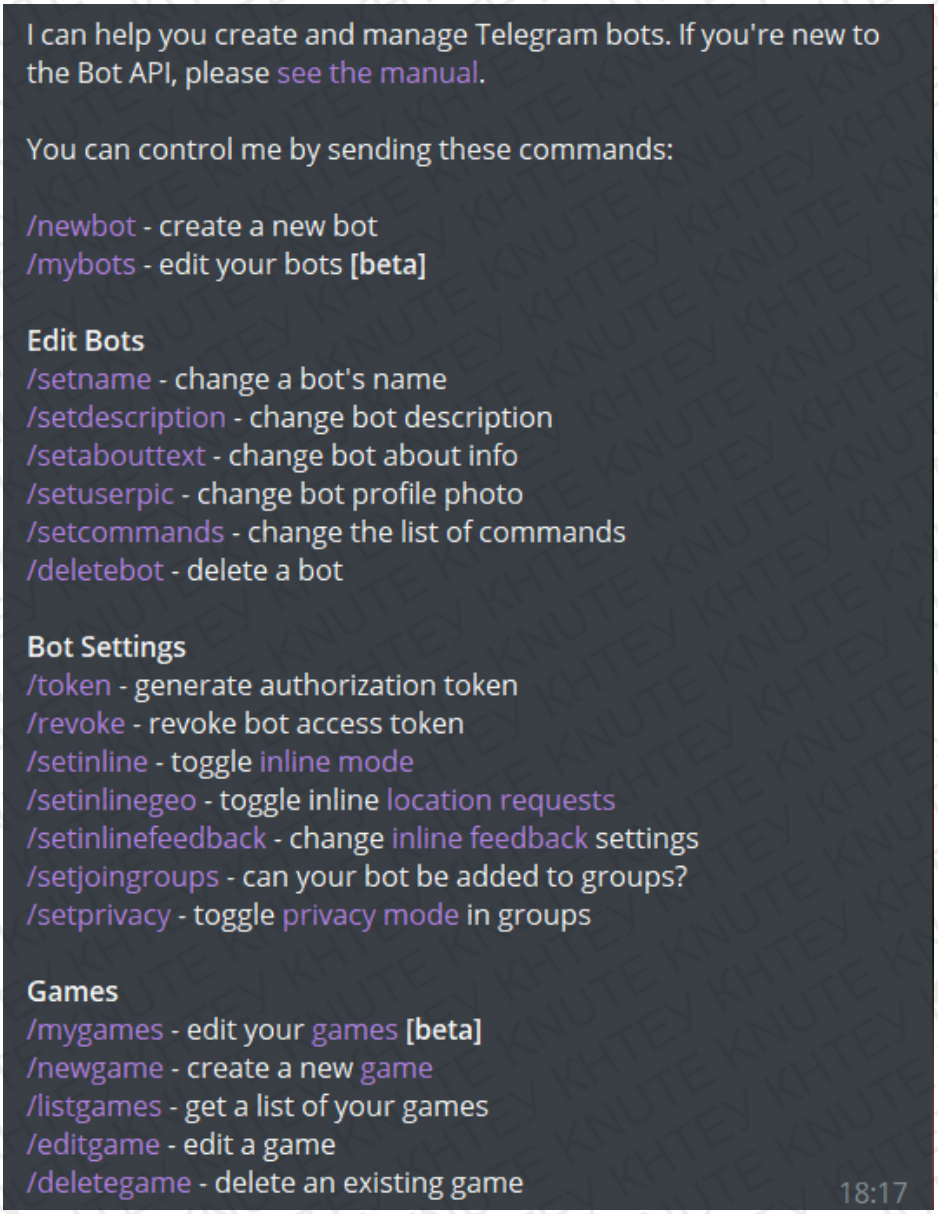

Рисунок 3.2 – Список команд BotFather

Для створення бота прописуємо команду /newbot. Потім іде процедура вибору назви та імені бота. Ім’я бота повинно бути унікальним та мати закінчення «bot».

Після успішного вибору імені BotFather відправляє коротку інформацію про новоствореного бот: посилання через яке можна вийти на чат з ботом та токен для підключення скрипту до бота[10].

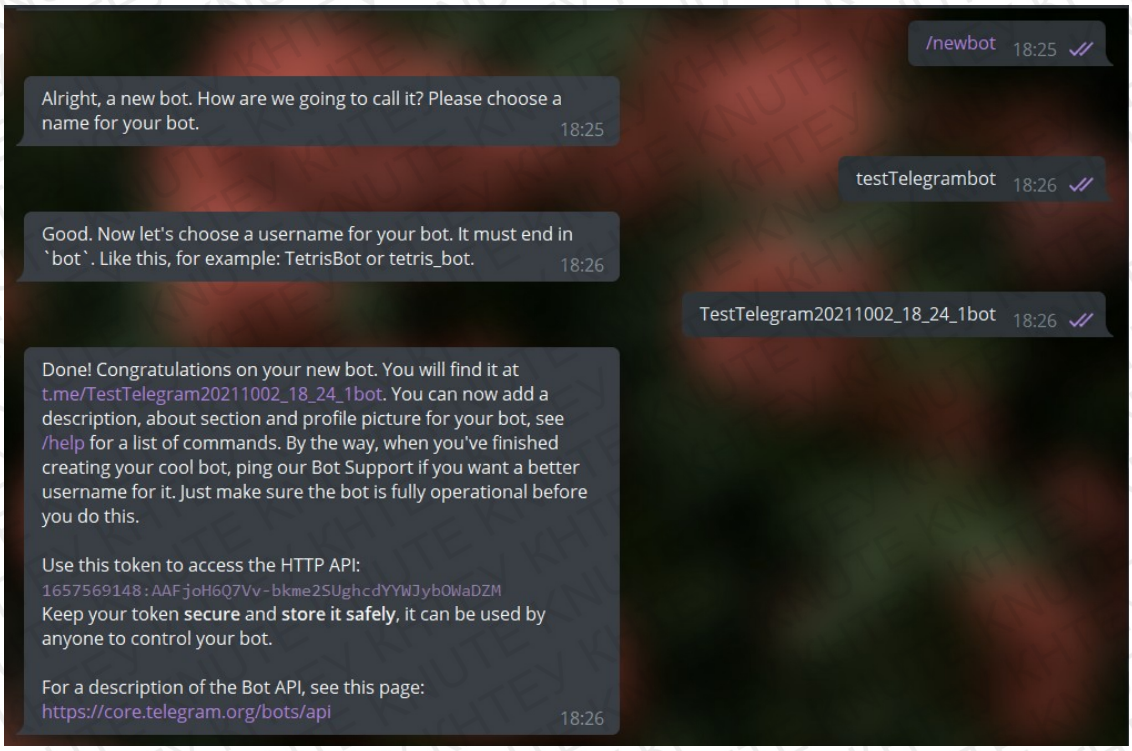

Рисунок 3.3 – Вибір імені бота

В BotFather є список команд для «кастомізації» бота. Список команд для цього:

- /setname – зміна назви бота;
- /setdescription – зміна опису бота;
- /setabouttext – зміна інформації про бота;
- /setuserpic – зміна фотографії профілю;
- /setcommands – змінити лист команд бота;
- /deletebot – видалити бота;

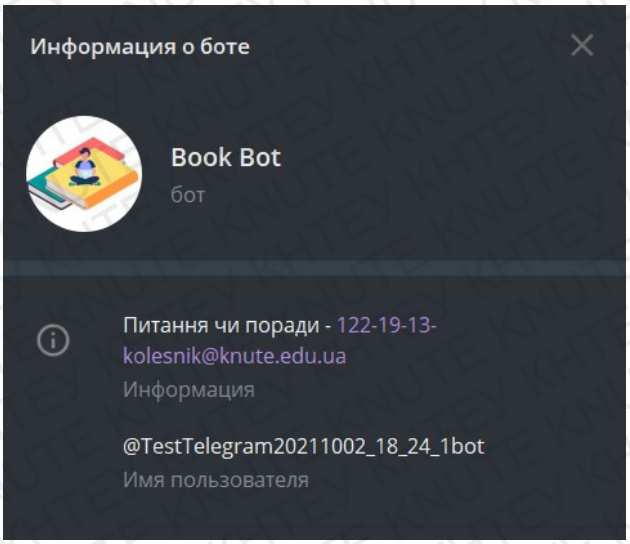

Рисунок 3.4 – Назва та опис чат-бота

Наступний крок – створення скрипту, який буде працювати з чат-ботом через його токен. Представлена ієрархія проекту у вигляді папок, файлів на мові програмування Python.

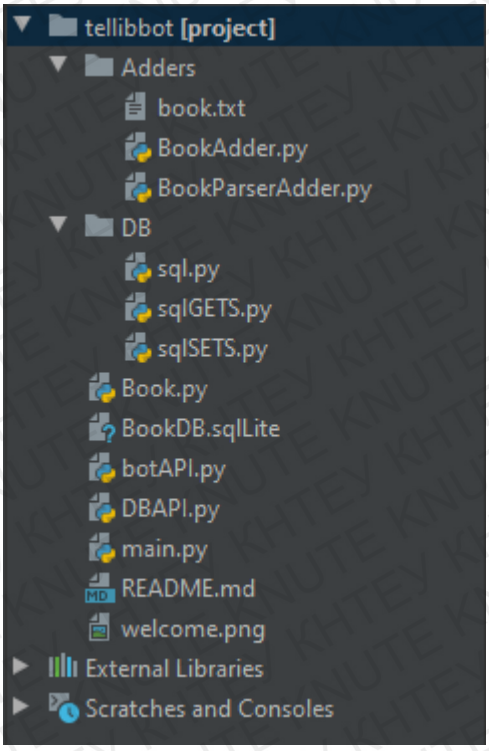

Рисунок 3.5 – Структура проекту

При старті роботи потрібно завантажити та імпортувати бібліотеку pyTelegramBotAPI для спрощеного написання коду. Після потрібно підключити бота до коду через його токен:

```
import telebot
```

```
token = '1657569148:AAFjoH6Q7Vv-bkme2SUghcdYYWJybOWaDZM'
bookBot = telebot.TeleBot(token)
```

В файлі botAPI.py описані всі функціональні команди взаємодії з користувачем такі як:

- /start
- /help
- /book
- /author
- /random

При введені команди /start відправляється повідомлення привітання при старті розмови в якому іде посилання на команду для перегляду можливостей бота. Фрагмент коду:

```
@bookBot.message_handler(commands=['start'])
    def start_bot(message):
    img = open('welcome.png', 'rb')
    bookBot.send_photo(message.chat.id, img, "Вітаємо вас в *BookBot*
♂\n\n"
            "  ----------  \n\n"
        "Бот з перспективною базою книг  \n\n"
        "  Для перегляду команд натисни /help", parse_mode='Markdown')
```

Для цього привітання використовується картинка, яка відправляється разом з текстом повідомлення. В ієрархії файл називається «welcome.png».

При введені команди /help відбувається відправлення короткого опису де прописані всі команди та правила їх застосування. Фрагмент коду:

```
@bookBot.message_handler(commands=['help'])
    def help_bot(message):
    bookBot.send_message(message.chat.id, "Як мною користуватись?\n"
```

"Все дуже просто, вибире спосіб пошуку та напиши назву книги або автора для пошуку\n\n"

"Список доступних команд для мене: \n\n"

"/start - перша команда з якої почнеться наше спілкування\n"

"/book *НАЗВА КНИГИ* - пошук книжки по її назві\n"

"/author *ІМ'Я АВТОРА* - пошук книжки по автору\n"

"/random - випадкова книжка", parse_mode='Markdown')

При виборі одного з варіантів пошуку підключається скрипт для роботи з базою даних SQL. В ієрархії назва файлу DBAPI.py. Цей скрипт описує всі основні функції пов’язані з базою даних.

def add_to_sql() – за допомогою цієї функції додаються в створену базу даних книжки по запланованому шаблоні:

- id – числова змінна яка позначає нумерацію книги в базі даних.
- name – поле яке зберігає назву книги.
- author – поле яке зберігає ім’я автора книги.
- description – поле в якому короткий опис книги.
- tags – тег для книги.
- links – посилання на саму книгу.

take_random_book() – функція за допомогою якої можна отримати випадкову книгу з бази даних. Код представлений нижче:

```
def take_random_book(count):

    books_ = []
    data_base = create_connection(DB)
    info = read_query(data_base, read_books)
    for i in range(count):
    rnd = random.randrange(0, len(info))
```

```
            book = Book(info[rnd][0], info[rnd][1], info[rnd][2], info[rnd][3],
            info[rnd][4], info[rnd][5])
        books_.append(book)
        return books_
```

find_name() – функція через яку виконується пошук через базу даних, якщо при визові вказати в аргументі «0» то виконуватиметься пошук по назві книги, якщо «1» то по назві автора. Також ця функція приймає текст, який приходить від бота.

В папці під назвою DB, написані заготовленні функції для створення бази даних, запити для додавання нової інформації, запити для отримання даних з бази даних. В цій папці сформовано універсальний набір команд для роботи з базою даних. В них прописані функції створення бази даних, підключення до неї основного коду, витягування інформації з неї. Такий набір функцій можна використовувати, як шаблон для написання програм в роботі яких використовується база даних SQL.

В папці Adders міститься скрипт який використовується для додавання нової літератури в базу даних. Файл BookAdders.py використовується як функція, яку можна підключити до основного коду та виконувати додавання через чат-бота. Файл BookParserAdder.py робить додавання в базу даних за допомогою звичайного txt файлу book.txt. В цей файл, в певному порядку, записуються данні про книгу. Кожна нова інформація пишеться з нового рядку і в тому порядку, як прописано в функції def add_to_sql().

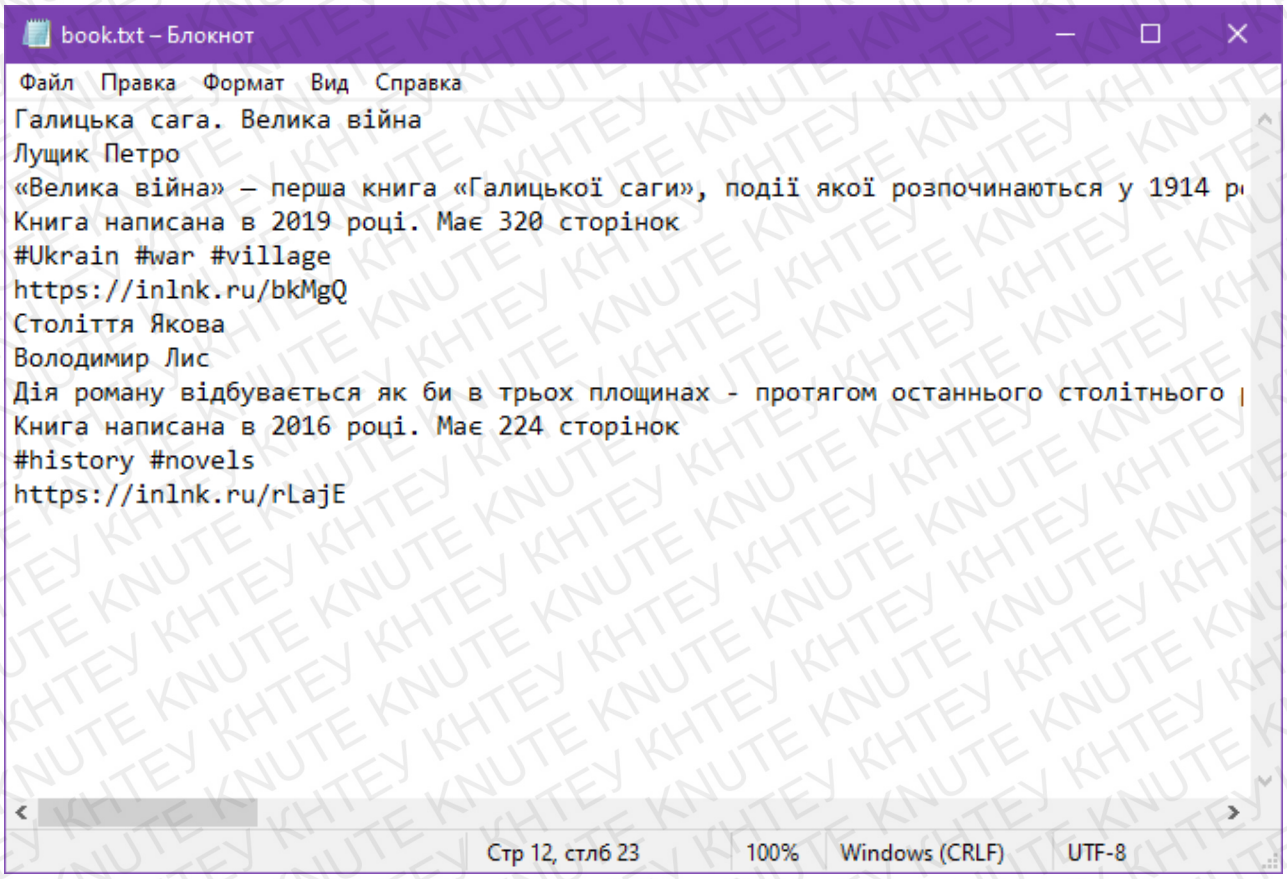

Рисунок 3.6 – Заповнення інформації для внесення в БД

Після того, як потрібну інформацію внесли до текстового файлу, окремим потоком від основного коду запускається файл BookParserAdder.py, після чого нова інформація додається в базу даних.

Файл book.py створений клас книжки для ініціалізації нової книги в базі даних як окремого об’єкта, який має свої унікальні властивості. Також додатково в цьому файлі прописується функція для відображення інформації в боті show_info(). Змінюючи функцію, можна змінити порядок та зовнішній вигляд інформації яка відправляється користувачу.

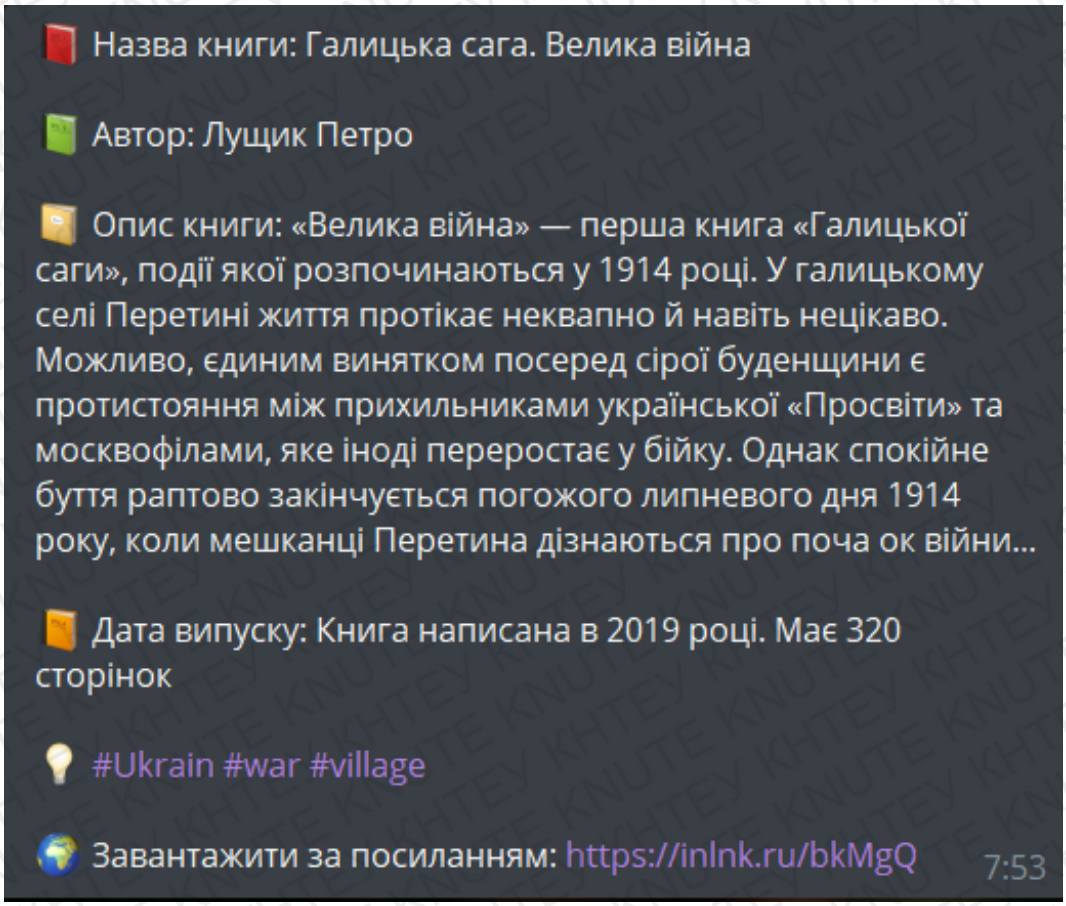

Рисунок 3.7 – Варіант показу інформації

Файл BookDB.sqlLite являється файлом бази даних в якому зберігаються всі книги. Всі маніпуляції з файлом виконуються через скрипт. Якщо файл не існує в папці проекту то він створюється автоматично. Переглянути чи редагувати вміст файлу, можна через програму «DB Browser for SQLite». Ця програма є дуже простою у використанні і має досить простий інтерфейс.

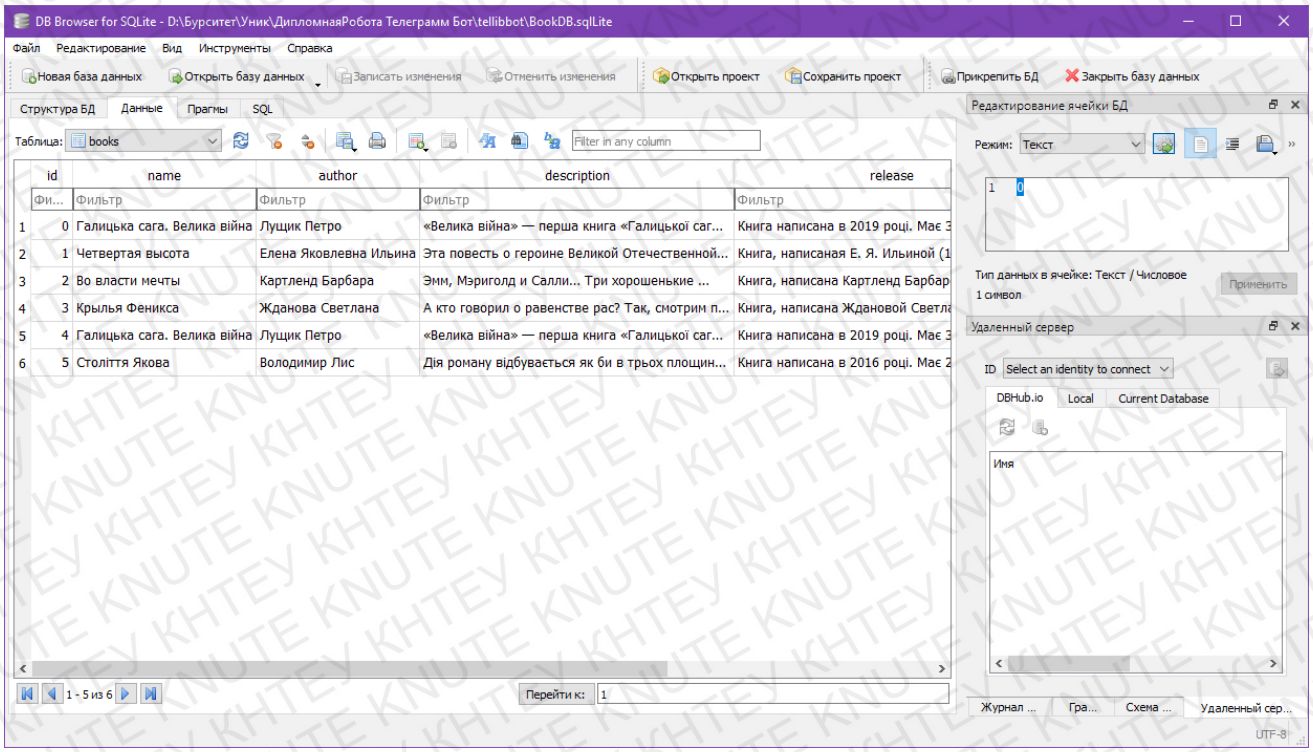

Рисунок 3.8 – Відображення вмісту бази даних

3.2 Опис функціоналу чат-бота

Для постійної роботи чат-бота потрібно перемістити програму на сервер, тоді у кожного користувача буде постійний доступ до чат-боту. На даний момент для його роботи потрібно вручну запускати скрипт через термінал.

Опис взаємодії користувача з книжковим чат ботом:

При переході на посилання чат-бота, відображається короткий опис можливостей.

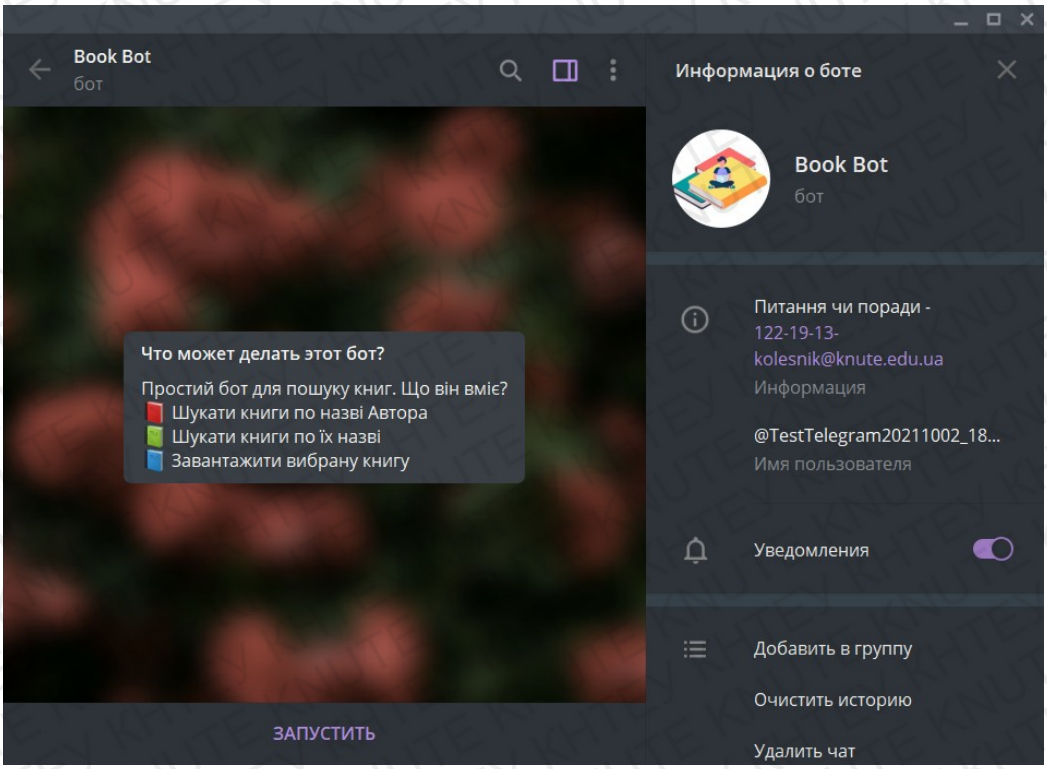

Рисунок 3.9 – Сторінка бота в Telegram

Для початку роботи потрібно натиснути кнопку «Запустить». В чаті автоматично пропишеться команда /start, на що чат-бот відправить повідомлення привітання.

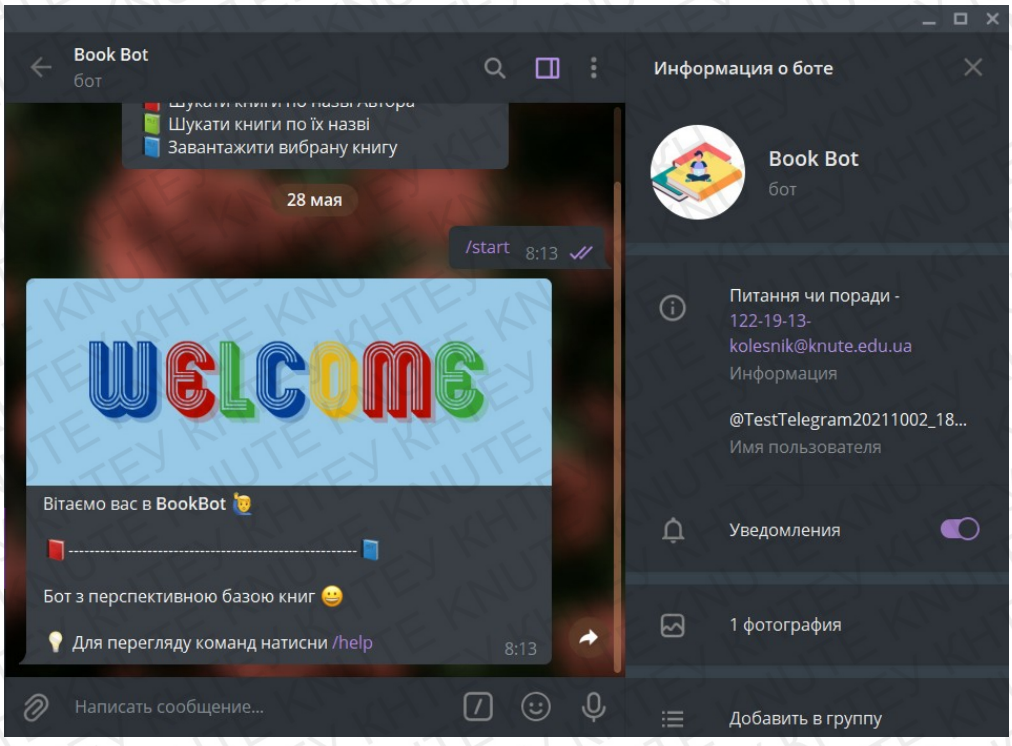

Рисунок 3.10 – Повідомлення привітання

В повідомленні користувачу пропонується скористатись командою /help, для відображення всіх команд чат-бота. Цю команду можна ввести в чат, або натиснуті в повідомлені.

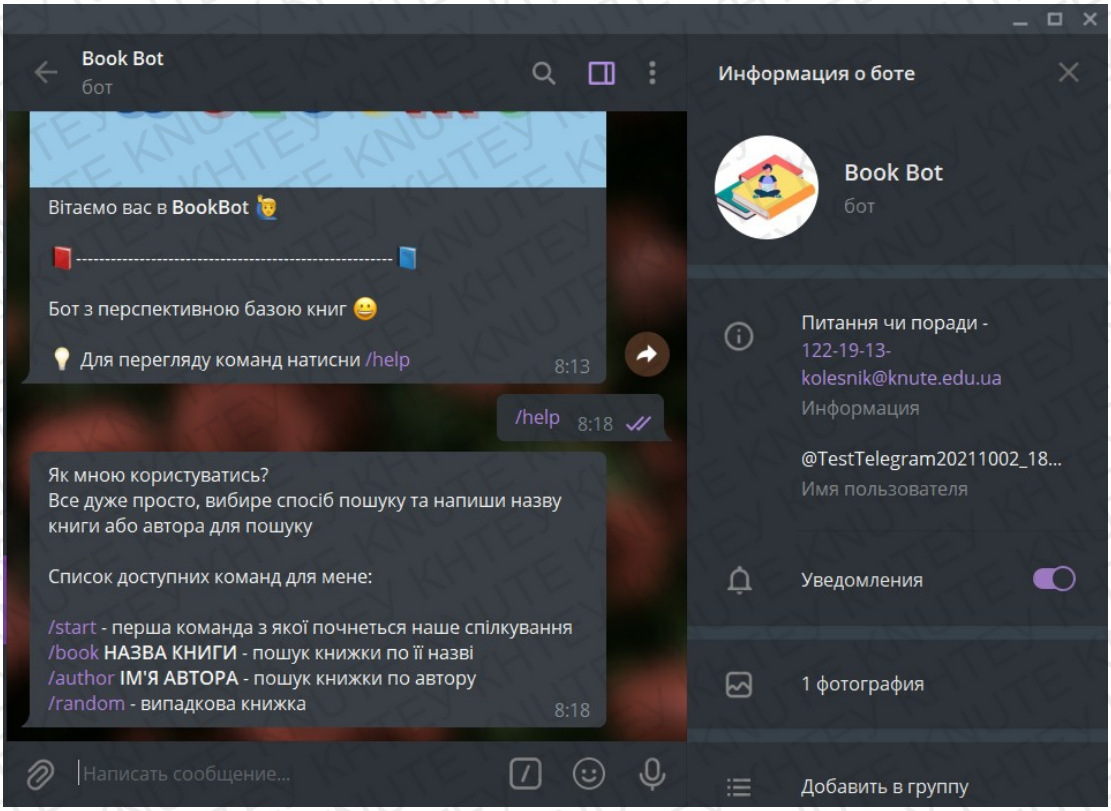

Рисунок 3.11 – Перелік всіх команд

Для того щоб отримати книгу по її назві потрібно написати саму команду та написати назву книги, такий самий спосіб пошуку по імені автора. Якщо користувач зробить щось не так, йому буде відправлено попереджувальне повідомлення.

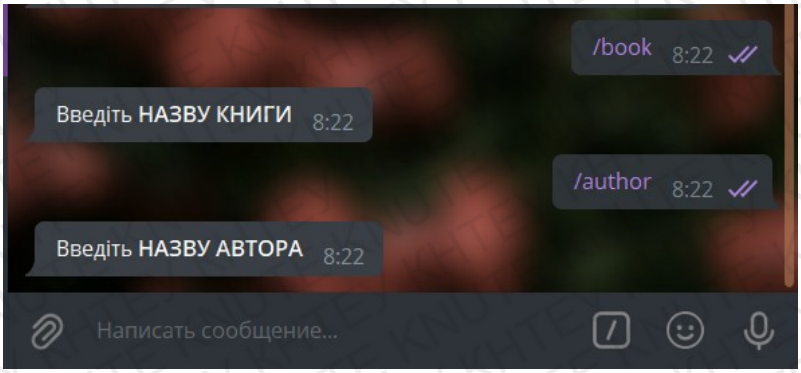

Рисунок 3.12 – Попереджувальне повідомлення

Якщо користувач зробить все правильно йому буде показаний результат пошуку.

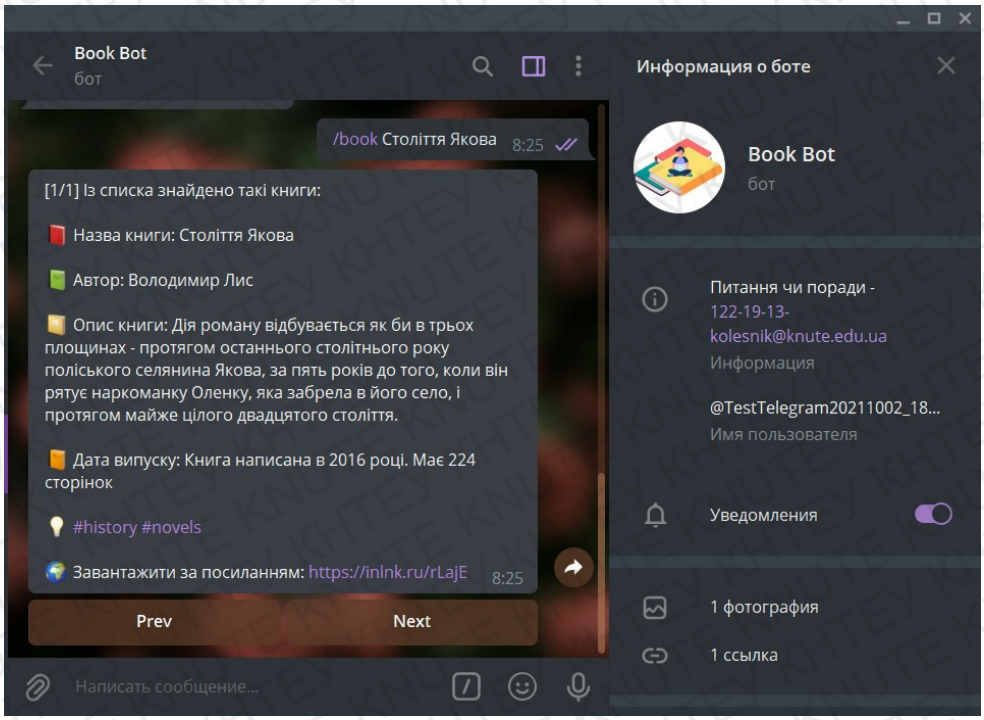

Рисунок 3.13 – Показ результатів пошуку по назві книги

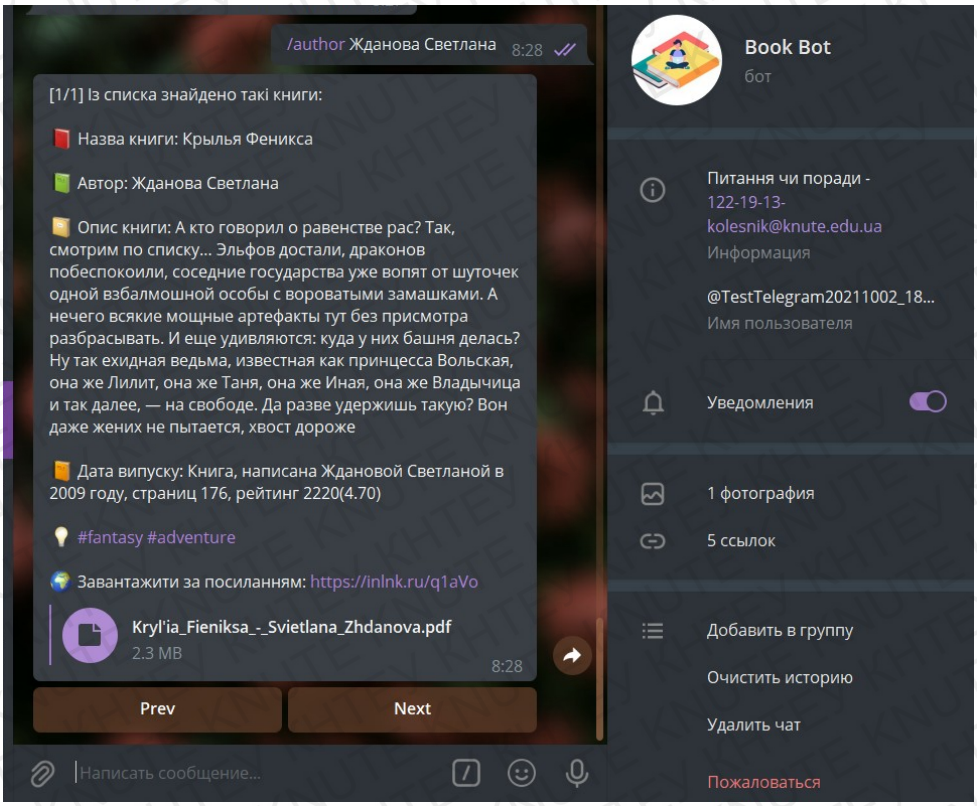

Рисунок 3.14 – Показ результатів пошуку по імені автора

Якщо на запит користувача буде знайдено декілька книг, він зможе перемикатись між варіантами за допомогою клавіш «Prev, Next». В повідомлені буде писатись кількість знайдених варіантів, та на якому варіанті в даний момент знаходиться користувач. При пошуковому запиті який відсутній в базі даних, користувачу буде відправлено повідомлення про відсутність такої книги чи автора.

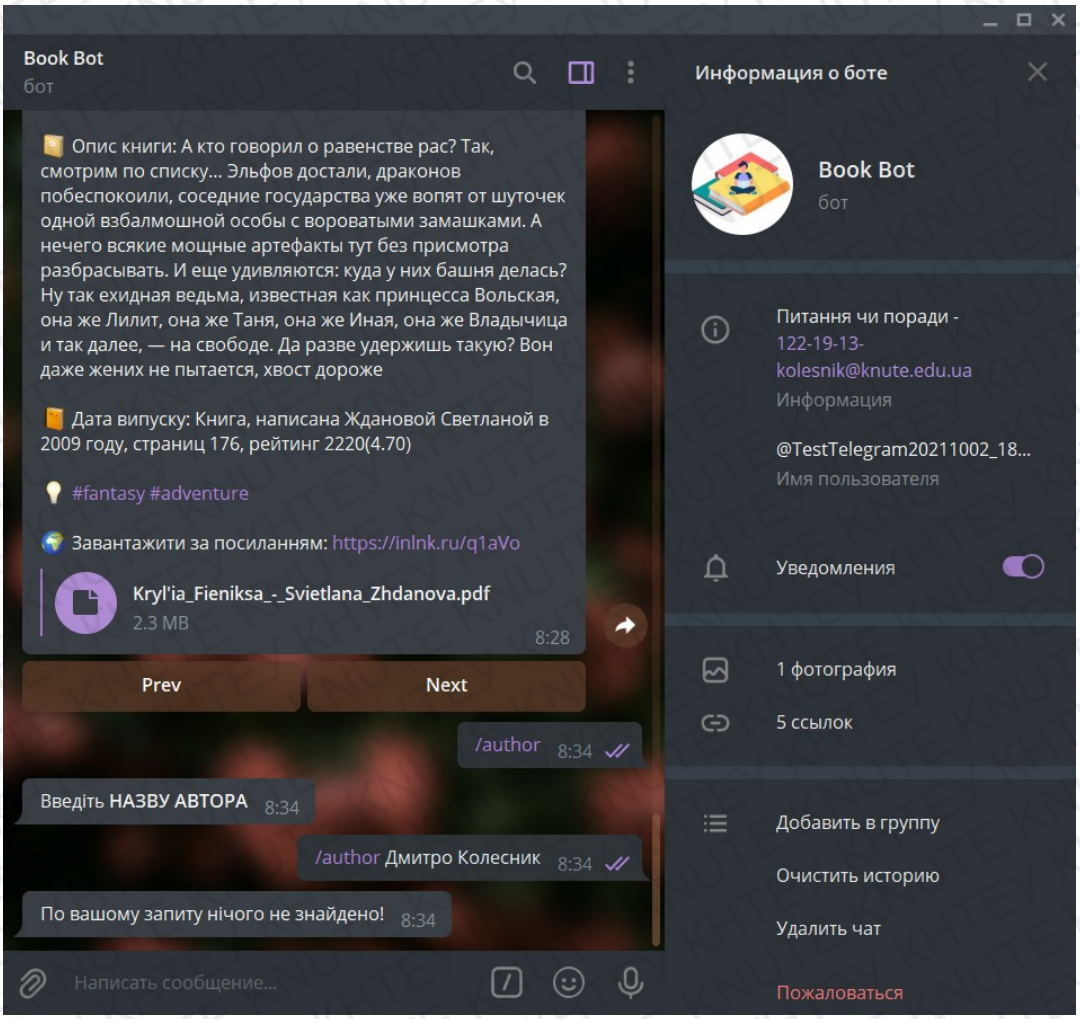

Рисунок 3.15 – Повідомлення про неіснуючий варіант пошуку

Проект є безкоштовний і знаходиться в вільному доступі, завантажити який можна через GitHub[11].

Висновок по розділу: Була розроблена пошукова система літератури в базі даних SQL та чат-бот за допомогою мови програмування Python. Представлені основні елементи коду та описані властивості функцій. Також була представлена взаємодія користувача з чат ботом, з усіма варіантами подій.

ВИСНОВОК

В ході роботи був проведений аналіз предметної області, а саме дослідження існуючих систем пошуку книги в месенджері за допомогою чат-бота. Було вибрано ключові елементи яким була наділення розробка система пошуку книг в базі даних SQL. Було обрано самі зручні методи взаємодії користувача з додатком.

Аналіз аналогів дав можливість порівняти книжкових ботів між собою. Завдяки даному аналізу був вибрані основні можливості чат-бота. Дана розробка буде безкоштовною і буде у вільному доступі для завантаження. Також чат-ботом зможуть користуватись всі верстви населення, у яких є смартфон, а також месенджер Telegram.

Основні розроблені функції проекту:

- додавання літератури в базу даних;
- пошук літератури по її назві;
- пошук літератури по назві автора;
- отримання випадкової літератури;

При створенні проекту були освоєні навички програмування чат-ботів використовуючи бібліотеку telebot та Telegram API. Створення, підключення, редагування бази даних використовуючи мову запитів SQL. Чат-бот реалізований за допомогою мови програмування Python/

Під час експлуатації розробки, були намічені напрями вдосконалення, а саме: додавання нових видів команд, вдосконалення системи пошуку, розширення бази даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ChatBot Wikipedia [Електронний ресурс] – Режим Доступу: https://en.wikipedia.org/wiki/Chatbot
2. Чат-боты: обзор и состояние технологий в отрасли [Електронний ресурс] – Режим Доступу: http://nlpx.net/archives/425
3. Чат-бот на сайте: зачем создавать, кому использовать и как настроить [Електронний ресурс] – Режим Доступу: https://blog.click.ru/growthhacking/chat-bot-na-sajte-zachem-sozdavat-komu-ispolzovat-i-kak-nastroit/
4. TELEGRAM BLOG [Електронний ресурс] – Режим Доступу: https://tgram.ru/blog
5. Telegram API Documentation [Електронний ресурс] – Режим Доступу: https://telegram.org/faq
6. Telegram Wikipedia [Електронний ресурс] – Режим Доступу: https://en.wikipedia.org/wiki/Telegram _(software)
7. Язык программирования Python [Електронний ресурс] – Режим Доступу: https://web-creator.ru/articles/python
8. Лутц М. Программирование на Python, том II, 4-е издание. – Пер. сангл. – СПб.: Символ-Плюс, 2011. – 992 с.
9. SQL Wikipedia [Електронний ресурс] – Режим Доступу: https://en.wikipedia.org/wiki/SQL
10. Покрокова інструкція: як створити телеграм-бота [Електронний ресурс] – Режим Доступу: https://dou.ua/lenta/articles/telegram-bot-ruby/
11. GitHub [Електронний ресурс] – Режим Доступу: https://github.com/D-Ulriken/tellibbot

Додаток А

Файл коду sql.py

```
from DB.sqlGETS import *
from DB.sqlSETS import *

data_base = create_connection("BookDB")

user_table = default_write_table("book",
                        id="id INTEGER PRIMARY KEY AUTOINCREMENT",
                        name="name TEXT NOT NULL",
                        author="author TEXT NOT NULL",
                        description="description TEXT NOT NULL",
                        release="release TEXT NOT NULL",
                        tags="tags TEXT NOT NULL")

write_query(data_base, user_table)
```

Файл коду sqlGETS.py

```
import sqlite3
from sqlite3 import Error


def read_query(connection, query):
    cursor = connection.cursor()
    result = None
    try:
        cursor.execute(query)
        result = cursor.fetchall()
        return result
    except Error as e:
        print(f"The error '{e}' occurred")



def default_table_select(name_table, **kwargs):
    value = ""
    for args in kwargs.values():
        value += args + ","
    value = value[:-1]
    select = """
    SELECT
        {value}
    FROM
        {name_table}
    """.format(name_table=name_table, value=value)
    return select
```

```
def default_table_from(**kwargs):
    value = ""
    for args in kwargs.values():
        value += args + ","
    value = value[:-1]
    select = """
    FROM
    {value}
    INNER JOIN users ON users.id = posts.user_id
    """.format(value=value)
    return select
```

Файл коду sqlSETS.py

```
    import sqlite3
import sys
from sqlite3 import Error
import os

DB = sys.path[1] + "\BookDB.sqlLite"

def create_connection(path=DB):
    connection = None
    try:
        connection = sqlite3.connect(path)
        print("Connection to SQLite DB successful")
    except Error as e:
        print(f"The error '{e}' occurred")
    return connection


def write_query(connection, query):
    cursor = connection.cursor()# object of db
    try:
        cursor.execute(query) # integrate values in db
        connection.commit()
        print("Query executed successfully")
    except Error as e:
        print(f"The error '{e}' occurred")


def default_write_table(name_table: str, **kwargs: str):
    fields = ""
    for args in kwargs.values():
        fields += args + ","
    fields = fields[:-1]
    create_users_table = """
    CREATE TABLE IF NOT EXISTS {name_table} (
```

```
    {args}
  );
  """.format(name_table=name_table, args=fields)
  return create_users_table


def default_add_info_table(name_table: str, fields: str, **kwargs: str):
  info = ""
  for args in kwargs.values():
    info += args + ","
  values = info[:-1]
  add_users = """
  INSERT INTO
   {name_table}({fields})
  VALUES
   {values};
  """.format(name_table=name_table, fields=fields, values=values)
  return add_users
```

Файл коду BookAdders.py

```
from DBAPI import *


def add_book():
  name = input("Enter name")
  author = input("Enter author")
  description = input("Enter description")
  release = input("release")
  tags = input("tags")
  links = input("link")
  book = Book(name, author, description, release, tags, links)
  books.append(book)
  add_to_sql()


add_book()
```

Файл коду BookParserAdders.py

```
from DBAPI import *

def parser_book():
  file = open("book.txt", "r", encoding="utf-8")
  counter = 0

  name = None
  author = None
  description = None
  release = None
```

```
    tags = None
    links = None

    for line in file.readlines():
        line = line.replace("\n", "")
        if counter == 0:
            name = line
        if counter == 1:
            author = line
        if counter == 2:
            description = line
        if counter == 3:
            release = line
        if counter == 4:
            tags = line
        if counter == 5:
            links = line
        counter += 1
        if counter == 6:
            book = Book(name, author, description, release, tags, links)
            books.append(book)
            counter = 0
    add_to_sql()

parser_book()
```

Файл коду Book.py

```
class Book:

    def __init__(self, name: str = "", author: str = "",
                 description: str = "", release: str = "", tags: str = "", links: str = ""):
        self.name = name
        self.author = author
        self.description = description
        self.release = release
        self.tags = tags
        self.links = links

    def show_info(self):
        return "    Назва книги: " + self.name + "\n\n" \
               + "    Автор: " + self.author + "\n\n" \
               + "    Опис книги: " + self.description + "\n\n" \
               + "    Дата випуску: " + self.release + "\n\n" \
               + "    " + self.tags + "\n\n" \
               + "    Завантажити за посиланням: " + self.links
```

Файл коду botAPI.py

```
import telebot
from telebot import types

import DBAPI as db

token = '1657569148:AAFjoH6Q7Vv-bkme2SUghcdYYWJybOWaDZM'

bookBot = telebot.TeleBot(token)

counter = 0

query_book_finder = ""


@bookBot.message_handler(commands=['start'])
def start_bot(message):
   img = open('welcome.png', 'rb')
   bookBot.send_photo(message.chat.id, img, "Вітаємо вас в *BookBot*     ♂\n\n"
                                 "   ----------------------------------------------------    \n\n"
                                 "Бот з перспективною базою книг     \n\n"
                                 "     Для перегляду команд натисни /help", parse_mode='Markdown')


@bookBot.message_handler(commands=['help'])
def help_bot(message):
   bookBot.send_message(message.chat.id, "Як мною користуватись?\n"
                                 "Все дуже просто, вибире спосіб пошуку та напиши назву книги або
автора для пошуку"
                                 "\n\n"
                                 "Список доступних команд для мене: \n\n"
                                 "/start - перша команда з якої почнеться наше спілкування\n"
                                 "/book *НАЗВА КНИГИ* - пошук книжки по її назві\n"
                                 "/author *ІМ'Я АВТОРА* - пошук книжки по автору\n"
                                 "/random - випадкова книжка", parse_mode='Markdown')


@bookBot.message_handler(commands=['book'])
def book_finder(message):

   global counter, books, query_book_finder

   book_name = str(message.text).replace("/book", "")
   book_name = book_name.lstrip()

   if book_name == "":
      bookBot.send_message(message.chat.id, "Введіть *НАЗВУ КНИГИ*",
parse_mode='Markdown')
      return

   query_book_finder = book_name
```

```
    books = db.find_name(book_name, 0)

    markup_inline = types.InlineKeyboardMarkup()
    prev = types.InlineKeyboardButton(text='Prev', callback_data='book_finder_prev')
    next = types.InlineKeyboardButton(text='Next', callback_data='book_finder_next')
    markup_inline.add(prev, next)

    if counter < 0:
        counter = len(books) - 1
    if counter > len(books) - 1:
        counter = 0

    if len(books) > 0:
        bookBot.send_message(message.chat.id, "[" + str(counter+1) + "/" + str(books.__len__()) + "] "
                             + "Із списка знайдено такі книги: \n\n"
                             + books[counter].show_info(), reply_markup=markup_inline)
    else:
        bookBot.send_message(message.chat.id, "По вашому запиту нічого не знайдено!")


@bookBot.message_handler(commands=['author'])
def author_finder(message):

    global counter, books, query_book_finder

    author_name = str(message.text).replace("/author", "")
    author_name = author_name.lstrip()

    if author_name == "":
        bookBot.send_message(message.chat.id, "Введіть *НАЗВУ АВТОРА*",
parse_mode='Markdown')
        return

    query_book_finder = author_name

    books = db.find_name(author_name, 1)

    markup_inline = types.InlineKeyboardMarkup()
    prev = types.InlineKeyboardButton(text='Prev', callback_data='book_finder_prev')
    next = types.InlineKeyboardButton(text='Next', callback_data='book_finder_next')
    markup_inline.add(prev, next)

    if counter < 0:
        counter = len(books) - 1
    if counter > len(books) - 1:
        counter = 0

    if len(books) > 0:
        bookBot.send_message(message.chat.id, "[" + str(counter + 1) + "/" + str(books.__len__()) + "]
```

```
"
                    + "Із списка знайдено такі книги: \n\n"
                    + books[counter].show_info(), reply_markup=markup_inline)
    else:
        bookBot.send_message(message.chat.id, "По вашому запиту нічого не знайдено!")


@bookBot.message_handler(commands=['random'])
def random_finder(message):
    for book in db.take_random_book(1):
        bookBot.send_message(message.chat.id, book.show_info(), parse_mode='Markdown')


@bookBot.callback_query_handler(func=lambda call: True)
def finder_counter(call):
    global counter
    if call.data == 'book_finder_next':
        counter += 1
        call.message.text = query_book_finder
        book_finder(call.message)
    if call.data == 'book_finder_prev':
        counter -= 1
        call.message.text = query_book_finder
        book_finder(call.message)


def start_bot():
    bookBot.polling(none_stop=True, interval=0)
```

Файл коду DBAPI.py

```
from DB.sqlGETS import *
from DB.sqlSETS import *
from Book import *
import random
import re

books = []

book_table = default_write_table("books",
                        id="id INTEGER PRIMARY KEY AUTOINCREMENT",
                        name="name TEXT NOT NULL",
                        author="author TEXT NOT NULL",
                        description="description TEXT NOT NULL",
                        release="release TEXT NOT NULL",
                        tags="tags TEXT NOT NULL",
                        links="links TEXT NOT NULL")

read_books = default_table_select("books",
                         name="books.name",
```

```
                    author="books.author",
                    description="books.description",
                    release="books.release",
                    tags="books.tags",
                    links="books.links")

def add_to_sql():
    global books
    data_base = create_connection(DB)
    for book in books:
        new_users = default_add_info_table("books",
                                "name, author, description, release, tags, links",
                                a="('{name}', '{author}', '{description}', '{release}', '{tags}', '{links}')"
                                .format(name=book.name, author=book.author,
                                        description=book.description,
                                        release=book.release, tags=book.tags, links=book.links))
        write_query(data_base, new_users)
    books = []

def take_random_book(count):
    books_ = []
    data_base = create_connection(DB)
    info = read_query(data_base, read_books)
    for i in range(count):
        rnd = random.randrange(0, len(info))
        book = Book(info[rnd][0], info[rnd][1], info[rnd][2], info[rnd][3], info[rnd][4], info[rnd][5])
        books_.append(book)
    return books_

def find_name(word, find: int = 0):
    """0 - name, 1 - author"""
    data_base = create_connection(DB)
    books_info = read_query(data_base, read_books)
    book = None
    books = []

    if find < 0 or find > 1:
        find = 0
    counter = 0
    for i in range(len(books_info)):
        result = re.match(r" + word.lower(), books_info[i][find].lower())

        if result is not None:
            counter += 1
            if find == 0:
                book = Book(books_info[i][0], books_info[i][1], books_info[i][2],
                            books_info[i][3], books_info[i][4], books_info[i][5])
            if find == 1:
                book = Book(books_info[i][0], books_info[i][1], books_info[i][2],
                            books_info[i][3], books_info[i][4], books_info[i][5])
```

```
        books.append(book)
    return books
```

Файл коду main.py

```
    import botAPI as bot
from DBAPI import *

if __name__ == '__main__':
    data_base = create_connection(DB)
    write_query(data_base, book_table)
    bot.start_bot()
```