

МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І
СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА
на тему:

РОЗРОБКА ТЕЛЕГРАМ-БОТА ДЛЯ СПОЖИВАЧІВ ПОСЛУГ
ВІДЕОІНДУСТРІЇ

Здобувача вищої освіти першого (бакалаврського) рівня,
4 курсу, групи КБ-19-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології (за галузями)»


(підпис)

Усачика Дениса Костянтиновича
(прізвище, ім'я, по батькові)

Науковий керівник


(підпис)

Редич Олександр
Володимирович, к.е.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Керівник освітньо-
професійної програми


(підпис)

Федотова - Півень Ірина
Миколаївна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Допущено до захисту за рішенням кафедри
протокол від «26» травня 2023 № 13

Завідувач кафедри


(підпис)

Філоненко Михайло Миколайович,
к.ф.-м.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене
звання)

Ірпінь – 2023

Реєстрація № 14 05.06.2023 
 (дата) (підпис відповідальної особи)

Результати перевірки до захисту 29.05.2023 
 (до захисту, на доопрацювання) (дата) (підпис)

Результати захисту 5 (відмінно) 14.06.2023 
 (оцінка) (дата) (підпис)

Голова комісії:  В.М. Франчук
 (підпис) (ініціали, прізвище)

Члени комісії:  М.М. Філоненко
 (підпис) (ініціали, прізвище)

 О.В. Редич
 (підпис) (ініціали, прізвище)

 Є.В. Котух
 (підпис) (ініціали, прізвище)

БЛАНК ОЦІНЮВАННЯ ВИПУСКНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії, секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	29		14.06.2023 Франчук В.М. Філоненко М.М.
2. Дотримання вимог щодо оформлення	від 0 до 10	9		Редич О.В.
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	4		Котух Є.В. Суховерхівка Л.П.
4. Рецензент	від 0 до 5	5		
РАЗОМ БАЛІВ	від 0 до 50	47		

Оцінювання кваліфікаційної роботи науковим керівником - 47 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
94	A	5 (відмінно)	14.06.2023	Франчук В.М. Філоненко М.М. Редич О.В. Котух Є.В. Суховерхівка Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	
50-59	E	3 (задовільно)
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій

Кафедра комп'ютерних та інформаційних технологій і систем

Галузь знань: 12 «Інформаційні технології»

Спеціальність: 122 «Комп'ютерні науки»

Освітня програма: «Інформаційні управляючі системи та технології (за галузями)»

Затверджую

Завідувач кафедри

М.М. Філоненко

« 20 » 11 2022 р.

Завдання на кваліфікаційну роботу здобувачу вищої освіти

Усачику Денису Костянтиновичу

1. Тема дипломної роботи

«Розробка телеграм-бота для споживачів послуг відеоіндустрії»

Затверджена наказом ректора від «15» листопада 2022 № 1205 (зі змінами наказ від «04» квітня 2023 № 361)

2. Строк здачі закінченої роботи: «20» травня 2023

3. Мета роботи: розробка телеграм-бота для споживачів послуг відеоіндустрії

Об'єкт дослідження: технології та фреймворки створення ботів

Предмет дослідження: мова Python та pyTelegramBotAPI як платформа створення телеграм-бота

Орієнтовна база дослідження: установи і підприємства відеоіндустрії

4. Вихідні дані до роботи: Існуючі реалізації Веб-орієнтованих систем надання послуг відеоіндустрії

5. Перелік графічного матеріалу: рисунки та графіки, формули, що пояснюють суть застосованих методів та засобів. Структурні схеми та таблиці, графіки результатів моделювання.

6. Зміст випускної кваліфікаційної (дипломної) роботи (перелік питань за кожним розділом) :

РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ АСПЕКТИ СТВОРЕННЯ ТЕЛЕГРАМ-БОТІВ

1.1. Сутність телеграм-ботів

1.2. Аналіз існуючих рішень

1.3. Огляд інформаційних технологій для розробки телеграм-ботів

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ТЕЛЕГРАМ-БОТА

2.1. Проектування алгоритму роботи бота

2.2. Розробка телеграм-бота у середовищі розробки

2.3. Тестування телеграм-бота та оцінка ефективності телеграм-бота

РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ ТЕЛЕГРАМ-БОТА

3.1. Порівняння розробленого бота з існуючими рішеннями

3.2. Ідеї щодо додаткового функціоналу та рекомендації щодо подальшої телеграм-бота

3.3. Використання телеграм-бота в реальних умовах, потенційні користи можлива комерційна вигода

7. Календарний план виконання роботи

№ п/п	Назва етапів випускної кваліфікаційної (дипломної) роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури ДР	До 15.11.2022 р.	« 14 » 11
2	I розділ (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	26. 03. 2023 р.	« 25 » 03
3	II розділ (аналіз експериментальної частини про і методики досліджень)	20. 04.2023 р.	« 20 » 04
4	III розділ (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	12. 05.2023 р.	« 12 » 05
5	Висновки	18. 05.2023 р.	« 18 » 05
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	20.05.2023 р.	« 20 » 05
7	Отримання відгуку керівника	03.06.2023 р.	« 03 » 06
8	Отримання зовнішньої рецензії	05.06.2023р.	« 05 » 06

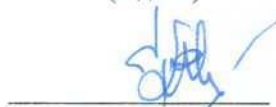
Дата видачі завдання « 20 » листопада 2022 р.

Керівник дипломної роботи


(підпис)

О.В. Редич

Керівник освітньо-професійної програми


(підпис)

І.М. Федотова-Півоварська

Завдання до виконання отримав здобувач вищої освіти


(підпис)

Д.К. Усачик

АНОТАЦІЯ

«Розробка телеграм-бота для споживачів послуг відеоіндустрії»

Мета роботи полягає в розробці телеграм-бота для споживачів послуг відеоіндустрії, який дозволить зберігати обрані користувачем фільми та надавати їх рейтинг, надавати рекомендаційні добірки фільмів за жанром, схожістю.

Завдання роботи є провести дослідження технологій створення мережових ботів та створити телеграм-бота для споживачів відеоіндустрії, який буде зберігати фільми користувачів, виводити список рекомендацій фільмів, робити добірки за різними критеріями.

Предметом дослідження мова Python та pyTelegramBotAPI як платформа створення телеграм-бота.

Об'єктом дослідження є технології та фреймворки які допомагають при створення телеграм-ботів.

Робота має 3 розділи: теоретико-методологічні аспекти створення телеграм-ботів, проектування та розробка телеграм-бота, перспективи розвитку телеграм-бота.

В першому розділі проводиться огляд та аналіз предметної сфери, необхідні інструменти та обґрунтування вибору фреймворків та засобів для створення телеграм-ботів

В другому розділі проводиться розробка алгоритму телеграм-бота з описом його функціонування, а також саме кодування та тестування.

У третьому розділі проводиться порівняння телеграм-бота з існуючими рішеннями, а також рекомендація для подальшого розвитку.

Кваліфікаційна робота містить 56 сторінок, 1 таблиця, 15 рисунків, список використаних джерел 40 найменувань, 2 додатки

Ключові слова: Чат-бот, Telegram, відеоіндустрія, Python, IMDb, OpenAI.

Рік виконання кваліфікаційної роботи: 2023

Рік захисту роботи: 2023

ANNOTATION

«Development of a Telegram bot for consumers in the video industry»

The goal of this work is to develop a Telegram bot for consumers in the video industry that allows users to save their favorite movies, provide ratings, and offer recommended film selections based on genre and similarity.

The objectives of this work are to research technologies for creating networked bots, and to create a Telegram bot for consumers in the video industry that can store user films, generate lists of recommended movies, and create curated selections based on various criteria.

The subject of research is the Python language and pyTelegramBotAPI as a platform for creating a Telegram bot.

The object of research is technologies and frameworks that help in creating Telegram bots.

The work consists of three sections: theoretical and methodological aspects of creating Telegram bots, design and development of the Telegram bot, and prospects for the bot's future development.

The first section provides an overview and analysis of the subject area, necessary tools, and justification for the choice of frameworks and resources for creating Telegram bots.

The second section focuses on the development of the Telegram bot algorithm, including its functioning, coding, and testing.

The third section compares the Telegram bot with existing solutions and provides recommendations for future development.

This thesis consists of 56 pages, 1 table, 15 figures, a reference list of 40 sources, and 2 appendix.

Keywords: Chatbot, Telegram, video industry, Python, IMDb, OpenAI.

Year of completion: 2023

Year of thesis defense: 2023

ЗМІСТ

ВСТУП	3
РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ АСПЕКТИ СТВОРЕННЯ ТЕЛЕГРАМ-БОТІВ	6
1.1. Сутність телеграм-ботів	6
1.2. Аналіз існуючих рішень	10
1.3. Огляд інформаційних технологій та інструментів для розробки телеграм-ботів	12
РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ТЕЛЕГРАМ-БОТА	19
2.1. Проектування архітектури та алгоритму роботи бота	19
2.2. Розробка телеграм-бота у середовищі розробки	23
2.3. Тестування телеграм-бота та оцінка ефективності телеграм-бота	34
РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ ТЕЛЕГРАМ-БОТА	47
3.1. Порівняння розробленого бота з існуючими рішеннями	47
3.2. Ідеї щодо додаткового функціоналу та рекомендації щодо подальшої розробки телеграм-бота	47
3.3. Використання телеграм-бота в реальних умовах, потенційні користувачі та можлива комерційна вигода	49
ВИСНОВКИ	51
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	53
ДОДАТКИ	57
Додаток А	57
Додаток Б	58

ВСТУП

В сучасному світі інтернет-технології та соціальні мережі відіграють значну роль у нашому житті, забезпечуючи зручні та швидкі засоби комунікації та інформаційного обміну. Особливо широко поширені месенджери, які дозволяють користувачам обмінюватися повідомленнями, фотографіями та відео зі своїми друзями, родичами та колегами.

Одним із найпопулярніших месенджерів є Telegram, який має понад 700 мільйонів активних користувачів щомісяця по всьому світу (за даним за червень 2022 року). Зі зростанням популярності месенджерів з'явилась необхідність у розробці чат-ботів, що дозволяють користувачам отримувати інформацію та взаємодіяти зі сторонніми сервісами, не виходячи з месенджера.

Однією з перспективних галузей, в яких можуть бути використані чат-ботами, є відео та кіноіндустрія. У зв'язку з пандемією COVID-19 та стрімким зростанням онлайн-трансляцій, діяльність у цій сфері стала значно актуальнішою.

Актуальність теми. Відеоіндустрія останні роки дуже швидко зростає, тому що все більшій кількості людей подобається диванний спосіб життя та витратити більше часу на перегляд відео в онлайн режимі. З мінливістю споживачів змінюється і технологія перегляду відео: з'являються нові сервіси, які дозволяють перегляд відео без реклами та низькою ціною. Це вимагає нових підходів до взаємодії зі споживачами відеоіндустрії та створення ефективних інструментів зберігання та аналізу даних.

Чат-боти в Телеграмі дозволяють споживачам відеоіндустрії отримувати більш зручний та простий спосіб отримання інформації про фільми, що може позитивно впливати на їх взаємодію з відеосервісами та сприяти збільшенню витрат на відео. Розробка телеграм-бота для зберігання та аналізу даних про фільми може допомогти покращити якість обслуговування та забезпечити зручні умови для споживачів кіно та серіалів.

Мета і завдання дослідження. Це розробка телеграм-бота для споживачів послуг відеоіндустрії, який здатний зберігати вибрані користувачем фільми та надавати їх рейтинги на основі оцінок IMDb, а також рекомендувати нові фільми та серіали користувачу. Провести дослідження технологій створення мережових ботів та обґрунтувати вибір фреймворку для реалізації завдання.

Головна мета роботи полягає в розробці та створенні інструменту взаємодії між споживачами відеоіндустрії та відеосервісами. Основна ідея полягає в тому, щоб забезпечити зручний та ефективний спосіб зберігання та аналізу даних про фільми, що дозволить споживачам відеоіндустрії швидко та легко знаходити необхідну інформацію про відеоконтент, який їх цікавить.

Крім того, мета роботи полягає в розробці алгоритмів аналізу інформації про фільми, що дозволить користувачам отримувати рекомендації фільмів. Такий аналіз може допомогти споживачам відеоіндустрії зробити більш об'єктивний вибір при перегляді фільмів.

Стан проблеми. Телеграм-ботів зі збереженням фільмів та рекомендаціями за жанрами та схожістю доволі мало. Більшість з ботів спеціалізується на якомусь конкретному напрямку, та майже немає таких які містили б у собі увесь необхідний функціонал. Багато таких ботів вже не працюють та зняті з хостингу. Тому, прийнято рішення про розробку даного телеграм-бота.

Об'єктом дослідження є технології та фреймворки які допомагають при створенні телеграм-ботів.

Предметом дослідження є процес розробки та імплементації телеграм-бота, а також мова Python та pyTelegramBotAPI як платформа створення телеграм-бота.

База досліджень: інтернет-ресурси, мобільні додатки, вебсайти та веб-технології

Практична значущість отриманих результатів: розроблений базовий функціонал може використовуватись медіаресурсами про фільми або

стрімінговими сервісами для покращення взаємодії з користувачами та для покращення користувацького досвіду.

Особистий внесок автора: розроблено телеграм-бота з унікальним функціоналом з використанням методів штучного інтелекту рекомендацій фільмів за схожістю, а також збереження фільмів. Такого функціоналу в існуючих рішеннях немає.

Наукова новизна проєкту. Новизна дипломної роботи полягає в розробці телеграм-бота для споживачів послуг відеоіндустрії з можливістю збереження обраних користувачем фільмів та отримання рекомендації на основі штучного інтелекту. Даний телеграм-бот буде мати наступні переваги порівняно з іншими аналогічними рішеннями:

- Зручність використання – телеграм-бот дозволить користувачам зберігати свої улюблені фільми в одному місці та отримувати їх рейтинг на основі оцінок IMDb без необхідності виходити з месенджера.
- Висока точність рейтингу – оцінки IMDb вважаються одними з найбільш авторитетних та об'єктивних, тому рейтинг фільмів, який буде надаватися телеграм-ботом, буде вірогідно відображати дійсну популярність та якість фільмів.
- Швидкий доступ до оцінок – телеграм-бот буде швидко та безпосередньо отримувати оцінки фільмів з вебсайту IMDb, що забезпечить користувачам швидкий та точний доступ до необхідної інформації.
- Універсальність – телеграм-бот є універсальним рішенням, яке може використовуватися на будь-якому пристрої з підтримкою телеграм-клієнта, що робить його доступним та зручним для широкого кола користувачів.
- Штучний інтелект – телеграм-бот рекомендує добірки фільмів за допомогою алгоритмів штучного інтелекту, він здатний знайти фільми за схожістю та жанром, які введе користувач.

РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ АСПЕКТИ СТВОРЕННЯ ТЕЛЕГРАМ-БОТІВ

1.1. Сутність телеграм-ботів

«**Телеграм-боти** — це спеціальні облікові записи, для налаштування яких не потрібен додатковий номер телефону. Ці облікові записи служать інтерфейсом для коду, що виконується десь на вашому сервері» [28]. Вони реалізовані за допомогою спеціальних бот-платформ, що надають розширений набір функцій для створення і керування ботами.

Різниця між ботами та користувачами:

- Боти не мають статусів «онлайн» — але вони мають спеціальну позначку «бот», яка відрізняє користувача від бота.
- У ботів є обмежене хмарне сховище, деякі старі повідомлення можуть бути видалені сервером незабаром після їх обробки.
- Бот не може перший почати спілкуватись з користувачем. Користувач повинен надіслати боту повідомлення, щоб почати працювати з ним. Люди можуть шукати ім'я користувача вашого бота або розпочати чат за унікальним посиланням(яке обов'язково містить слово «bot»).
- За замовчуванням боти, додані до груп, бачать лише відповідні повідомлення в чаті.

Сутність телеграм-ботів полягає в автоматизації різних завдань та наданні користувачам швидкого доступу до інформації та сервісів. Вони можуть виконувати різні функції, включаючи:

- *Відповідь на запити користувачів:* Телеграм-боти можуть приймати текстові повідомлення від користувачів та надавати їм запрограмовані відповіді. Це може бути відповідь на запитання, пояснення інформації, посилання на ресурси тощо.

- *Розсилання повідомлень:* Боти можуть надсилати повідомлення користувачам за певними правилами та розкладом. Це може бути корисно для розсилання новин, оновлень, розкладу подій тощо.
- *Збір інформації:* Боти можуть запитувати користувачів про певні дані або проводити опитування, щоб зібрати необхідну інформацію. Наприклад, бот може створити опитування про задоволеність певним продуктом або послугою. Це може бути корисним для отримання статистики.
- *Інтеграція зі зовнішніми сервісами:* Телеграм-боти можуть підключатися до зовнішніх сервісів та API для виконання певних завдань. Наприклад, бот може використовувати відкритий API вебресурсу IMDb для отримання бази даних про фільми.
- *Налаштування та персоналізація:* Телеграм-боти можуть надати користувачам можливість налаштувати певні параметри та отримати персоналізовані повідомлення або рекомендації. Наприклад, бот може запитувати користувачів щодо їхніх вподобань та пропонувати відповідні матеріали.
- *Організація робочих процесів:* Боти можуть бути використані для автоматизації робочих процесів та спрощення комунікації в команді або організації. Вони можуть надавати доступ до спільних документів, виконувати завдання з управління проєктами, нагадувати про терміни, проводити опитування серед колег тощо.
- *Фінансові операції:* Телеграм-боти можуть бути інтегровані з фінансовими сервісами, що дозволяє користувачам здійснювати платежі, переказувати кошти, перевіряти баланс тощо безпосередньо через месенджер.
- *Розваги та ігри:* Боти можуть надавати користувачам розваги, такі як вікторини, гра у шахи або розповідь анекдотів тощо. Це дозволяє забезпечити користувачам розваги безпосередньо у месенджері.

Для проєкту необхідно буде, щоб бот приймав текстові повідомлення від користувача та давати відповіді на запити. А також можна інтегрувати бот з зовнішніми сервісами для спрощення та оптимізації роботи з телеграм-ботом.

Основні види взаємодії користувача з телеграм-ботом:

- *Текстові повідомлення:* Користувач може відправляти текстові повідомлення боту, а той своєю чергою, може відповідати на них текстом, зображеннями, відео, стікерами, та іншими видами повідомлень.
- *Команди:* Користувач може викликати команди, вводячи спеціальний символ «/» перед командою. Бот реагує на команди, та виконує дії які були запрограмовані на конкретну команду.
- *Кнопки:* Бот може створювати клавіатуру з кнопками, які користувач може натискати для взаємодії. Кнопки можуть бути розташовані як в повідомленнях бота, так і в меню користувача.
- *Inline-відповіді:* Користувач може викликати бота у будь-якому чаті, використовуючи спеціальну форму написання команди «@НАЗВА_БОТА запит», і отримувати inline-відповіді від бота. Це дозволяє боту реагувати на запити користувачів навіть без прямого спілкування з ботом.

Щодо видів клавіатур, існує декілька типів клавіатур, які можуть бути використані в телеграм-ботах:

- *Звичайна клавіатура:* Це стандартна клавіатура з кнопками, які можна використовувати для вибору команди. Працює як заготовлений текст, який вводяться у чат при натисненні на кнопку.
- *Inline-клавіатура:* Інлайн-клавіатура розташовується під повідомленням бота або inline-відповіддю. Вона має кнопки, які користувач може вибирати, і кожна кнопка може мати власний текст або значок. Користувач може натискати на кнопки для взаємодії з ботом.

Боти створюються за допомогою спеціального «бота-батька», який так і називається «BotFather».

BotFather – офіційний бот-керівник для створення та керування телеграм-ботами в месенджері Telegram. Він надає користувачам можливість створити нового бота, налаштувати його параметри, отримати доступ до токenu, необхідного для використання API та інших функцій.

Токен є ключем доступу, який дозволяє взаємодіяти з ботом Telegram. Кожен токен унікальний і пов'язаний лише з одним конкретним ботом.

Токен потрібно зберігати у секреті та не розголошувати його, для уникнення несанкціонованого доступу. Тому у своєму проєкті я не буду демонструвати токен від свого телеграм-бота.

BotFather надає ряд можливостей, які дозволяють налаштувати параметри телеграм-бота:

- *Зміна імені та короткого імені користувача:* Коротке ім'я користувача це ідентифікатор бота за яким можна легко його знайти або додати у групу.
- *Налаштування фото профілю:* Можна встановити власне фото для візуального відображення сутності бота.
- *Встановлення опису бота:* Можна додати загальну інформацію, або пояснити функціональність бота. Зазвичай, в описі залишають посилання на технічну підтримку бота, або посилання на автора для зв'язку.
- *Налаштування команд:* Можна налаштувати список команд для взаємодії з ботом. Зазвичай, повинна бути стартова команда «/start» для початку роботи з ботом, але все залежить від потрібної функціональності бота.
- *Налаштування стартової панелі «Що вміє бот?»:* Це інформація яка виводиться при першому використанні бота, можна також вставити картинку. В цій панелі можна розписати функціонал бота та його можливості.

«Загалом, боти це невеличкі програми які працюють в месенджері Telegram. Користувачі взаємодіють із ботами через гнучкі інтерфейси, які можуть підтримувати будь-які завдання чи послуги» [2].

1.2. Аналіз існуючих рішень

За допомогою вебресурсів агрегаторів, було проаналізовано існуючі рішення, аналоги ботів на подібну тематику. З усіх переглянутих варіантів, я обрав три аналоги які підходять для аналізу (див. таблиця 1):

Таблиця 1 – Порівняння аналогів

Назва	Посилання	Основна задача	Можливості
BlaBlaWork	@BlaBlaWorkBot	Допомагає знайти вакансію або кандидата. Рекомендує фільм на вечір.	Обрати фільм на вечір. Налаштування вподобань: за жанром, країною, роками. Обирає фільм з нових фільмів або ті які надирають популярність.
Popcorn Time Bot	@popcorntime_bot	Нагадування коли виходить новий епізод вашого серіалу.	Додати серіал/шоу до списку. Показати список збережених серіалів/шоу. Перевірити чи вийшли нові епізоди серіалу/шоу. Зупинити нагадування. Оцінити та надати зворотній зв'язок.
Movies Tracker Bot	@movies4bot	Надає інформацію про фільм.	Виводить повну інформацію про фільм використовуючи бібліотеку IMDb.

Так, як ми не маємо доступу до відкритого вихідного коду цих ботів, будемо розглядати методи взаємодії з ними та за якою логікою вони працюють. Розглянемо кожне рішення окремо.

BlaBlaWork – бот, який більше призначений для пошуку кандидатів та вакансій, але має невелику можливість обрати фільм на вечір. Тому є сенс розглянути тільки функціонал рекомендації фільму. Взаємодія з ботом відбувається переважно через команди та inline-клавіатуру. Для того, щоб

отримати рекомендацію до фільму потрібно написати команду «/film». Та обрати в inline-клавіатурі один з варіантів «Нові фільми», «Набирають популярність» або «Налаштувати мої уподобання». При налаштуванні моїх уподобань можна обрати жанри, країну та роки випуску. Для подальшого користування бота потрібно мати «жетони», які дозволяють отримувати рекомендації фільмів. Для отримання «жетонів» потрібно запросити нових користувачів, що є корисним для збільшення користування ботом, але зменшує доступність бота та його зручність. Тому у своєму проєкті я не буду використовувати систему з «жетонами», для більшої зручності користувачам.

Popcorn Time Bot – бот призначений для зручного сповіщення про вихід нового епізоду серіалу або шоу. Взаємодія відбувається за допомогою команд, для початку роботи потрібно прописати команду «/start». Потім потрібно написати команду «/addshow» та написати назву серіалу або шоу. Після чого бот запам'ятовує цю назву та поміщає в список, який можна перевірити за допомогою команди «/showlist». При команді «/check» бот виводить чи з'явився новий епізод. Команда «/stop» зупиняє сповіщення про вихід нової серії. У цьому боті не зовсім зручна навігація та є необхідність вводити кожен раз команду для додавання серіалу. Тому у своєму проєкті, я хочу реалізувати більш зручну взаємодію з ботом за допомогою вбудованої клавіатури та inline-клавіатури, без використання команд. До того ж функціонал бота є досить корисним та можна буде реалізувати подібний функціонал в майбутніх оновленнях мого телеграм-бота.

Movies Tracker Bot – бот для отримання детальної інформації про фільм. Взаємодія реалізована за допомогою команд. Для пошуку фільму достатньо написати команду «/search + назва_фільму» або просто назву фільму без команди, бот відреагує на обидва варіанти коректно. Також бот може працювати в inline-режимі за допомогою команди в будь-якому чаті «@movies4bot назва_фільму». Функціонал бота не є дуже різноманітним, але його добре можна використовувати саме через inline-режим. Бот використовує бібліотеку IMDb, що я неодмінно буду використовувати при розробці свого

телеграм-бота. А також використання бота у режимі inline є доволі зручним, це також можна буде реалізувати у майбутніх оновленнях мого телеграм-бота.

Проаналізувавши існуючі рішення можна зробити висновок, що телеграм-ботів зі збереженням фільмів та рекомендаціями за жанрами та схожістю доволі мало. Кожен з ботів спеціалізується на якомусь конкретному напрямку, майже немає таких охоплюють увесь необхідний функціонал. Багато зі знайдених мною ботів вже не працюють та зняті з хостингу. Тому, мій телеграм-бот має можливість зайняти місце багатофункціонального бота помічника у світі кіно.

1.3. Огляд інформаційних технологій та інструментів для розробки телеграм-ботів

Для того, щоб розробити телеграм-бота потрібно обрати мову програмування на якій буде написаний бот, фреймворк та бібліотеки для розробки функціональності бота. Також необхідною частиною буде вибір редактор коду та система управління версіями.

Для розробки телеграм-ботів підтримуються різні високорівневі мови програмування. Основні мови, за допомогою яких можуть бути розроблені телеграм-боти:

- *Python*: Це одна з найпопулярніших мов програмування для розробки телеграм-ботів. Існує багато бібліотек, таких як `python-telegram-bot` або `pyTelegramBotAPI`, які дозволяють легко створювати і взаємодіяти з телеграм-ботами за допомогою Python.
- *JavaScript*: Це мова програмування, яка широко використовується для розробки вебдодатків, але може використовуватись для розробки телеграм-ботів. Для розробки на JavaScript можна використовувати фреймворки, такі як `Node.js`, `Telegraf` або `Botpress`.

- *Java*: Популярна мова програмування, для розробки телеграм-ботів на мові Java можна використовувати такі фреймворки, як TelegramBots API або TelegramBots.
- *Ruby*: Для розробки телеграм-ботів на Ruby можна використовувати фреймворки, такі як Telegram-bot-ruby або telegram-bot.

З усіх варіантів, найкращою мовою програмування для мого проєкту це – Python.

Свій вибір можу обґрунтувати тим, що Python легко використовувати, він має чистий та зрозумілий синтаксис, що дозволяє швидко розробити програмний засіб. Python має багато вбудованих бібліотек, а також сторонніх бібліотек та фреймворків, які дуже допомагають при розробці. Python досить швидкий, що дозволяє створювати швидкі та ефективні боти, які можуть обробляти великий потік запитів. На момент написання проєкту остання версія пакета Python 3.10.11.

Наступним кроком буде вибір фреймворку для створення телеграм-бота. Існує декілька видів фреймворків та бібліотек для цієї задачі:

- *pyTelegramBotAPI*: Це одна з найпопулярніших бібліотек для розробки телеграм-ботів на Python. Вона надає простий інтерфейс для створення ботів і обробки подій, таких як повідомлення, команди, клавіатури та інше.
- *python-telegram-bot*: Це ще одна популярна бібліотека для розробки телеграм-ботів на Python. Вона також надає зручний інтерфейс для створення та взаємодії з ботами.
- *aiogram*: Це асинхронний фреймворк для розробки телеграм-ботів на Python. Він підтримує використання асинхронних функцій та дозволяє зручно створювати ботів зі складнішою логікою.

Для розробки проєкту найбільше підходить *pyTelegramBotAPI* тому, що синтаксис бібліотеки доволі простий для розуміння, що прискорює розробку

бота. `pyTelegramBotAPI` надає широкі можливості для розробки різноманітних функцій телеграм-ботів. Вона підтримує обробку повідомлень, клавіатур, команд, стікерів, медіафайлів та багато іншого. У цієї бібліотеки хороша документація, яка надає вичерпну інформацію про всі можливості бібліотеки. Також `pyTelegramBotAPI` має велику та активну спільноту розробників, яка надає підтримку та відповідає на питання, що може бути корисним при виникненні проблем з використанням бібліотеки.

Наступним постає питання: «Звідки бот буде брати інформацію про фільми?»

Для цього необхідно використати бібліотеку з базою даних фільмів. Існуючі варіанти, це створити базу даних власноруч, або використати готові бібліотеки. Для того, щоб створити власну базу даних необхідно обробити та розписати інформацію до понад 7 мільйонів фільмів, що є не дуже швидким та раціональним рішенням. Тому, можна використати існуючу бібліотеку для мови Python це – `IMDbPY`.

`IMDbPY` – це бібліотека для мови програмування Python, яка надає доступ до однієї з найбільших баз даних IMDb (Internet Movie Database). `IMDbPY` дозволяє отримувати інформацію про фільми, серіали, акторів, режисерів, жанри та іншу інформацію, яка є в базі даних IMDb.

Основні функції та можливості `IMDbPY`:

- *Запит до бази даних IMDb:* Бібліотека надає можливість виконувати різні запити до бази даних IMDb для отримання інформації про фільми, серіали, акторів, жанри тощо. Є можливість шукати фільми за назвою, роком випуску, жанром, режисером, акторами та багатьма іншими параметрами.
- *Отримання деталей про фільми:* `IMDbPY` дозволяє отримувати розширену інформацію про фільми, таку як назва, рік виходу, тривалість, рейтинг, жанр, режисер, актори, опис, посилання на постер тощо.

- *Робота з акторами та режисерами:* За допомогою IMDbPY можна отримати інформацію про акторів, режисерів та інших учасників фільмів. Можна знайти фільми, в яких вони брали участь, та отримати додаткову інформацію про них.
- *Кешування даних:* Бібліотека IMDbPY має здатність кешування отриманих даних, що дозволяє зберігати локальну копію бази даних IMDb локальному сервері. Кешування дозволяє прискорити час відповіді на запити та зменшити навантаження на сервер IMDb.
- *Можливості редагування:* IMDbPY також надає деякі можливості для редагування бази даних IMDb. Можна додавати нові фільми, акторів, режисерів та змінювати існуючі дані.

Ця бібліотека є дуже корисною для розробки проєкту «Чат-бот для споживачів відеоіндустрії» тому, що дозволить отримати доступ до великої бази даних фільмів і шанс того, що користувач не знайде той фільм який він хоче буде мінімальним.

Також необхідно створити алгоритми для рекомендацій фільмів. Для цього я планую використати методи штучного інтелекту. А саме відкритий програмний інтерфейс (API) – OpenAI API.

OpenAI API – це програмний інтерфейс, який надає доступ до різноманітних моделей штучного інтелекту, розроблених OpenAI. Цей API дозволяє розробникам і дослідникам використовувати потужності штучного інтелекту, щоб розв'язувати різноманітні завдання, такі як генерація тексту, переклад, розпізнавання мови, створення чат-ботів та багато іншого. Моделі штучного інтелекту OpenAI підтримують велику кількість мов, у тому числі українську.

Ключові особливості OpenAI API:

- *Мовні моделі:* OpenAI API використовує передові мовні моделі, такі як GPT-3 (для безплатного доступу) і GPT-4 (для платного доступу), які навчені на величезних обсягах текстових даних. GPT-4 натренована на

декілька разів більшому об'ємі даних, ніж GPT-3. Ці моделі володіють здатністю зрозуміти та генерувати текст, демонструючи натуральну мову та когнітивні здібності.

- *Простота інтеграції:* API надає простий та зрозумілий інтерфейс для взаємодії з мовними моделями OpenAI. Для інтеграції використовується спеціальний створений токен, як і у випадку з Telegram API.
- *Кастомізація та настройка:* В OpenAI API є можливість налаштовувати параметри мовних моделей, що дозволяє керувати різними аспектами генерації тексту, такими як кількість випробувань (temperature), довжина виводу, вказівки (prompts) та інші параметри.
- *Підтримка різних типів даних:* API надає можливості взаємодії з широким спектром типів даних, включаючи рядки, списки, числа, дати та інші.

OpenAI API був створений з метою демократизації доступу до потужних моделей штучного інтелекту і дозволяє розробникам у різних галузях використовувати їх для вирішення різних завдань.

Для створення бази даних користувача мною була обрана вбудована у Python бібліотека SQLite3. Для мого проєкту не потрібна велика громіздка база даних, тому можна використати саме SQLite3 для створення невеликої локальної бази даних.

SQLite3 – це вбудована бібліотека, яка дозволяє працювати з базою даних SQLite за допомогою мови програмування Python. SQLite - це відносна база даних, яка не вимагає окремого сервера та може зберігатися в локальному файлі.

Бібліотека SQLite3 дозволяє робити запити до бази даних SQLite з допомогою мови програмування Python. Вона має простий та зрозумілий інтерфейс та надає можливість виконувати різні операції з базою даних, такі як створення таблиць, вставка та вибірка даних, модифікація та видалення записів та багато іншого.

Основні операції з базою даних SQLite, які можна виконувати за допомогою бібліотеки SQLite3 для Python:

- Підключення до бази даних.
- Створення таблиці.
- Вставка даних.
- Вибірка даних (за потрібними параметрами).
- Редагування даних.
- Видалення даних.

Варто також розглянути корисні інструменти при створенні телеграм-бота. Насамперед це редактор коду, я вирішив використовувати Microsoft Visual Studio Code.

Visual Studio Code (VS Code) – це безплатний та відкритий редактор коду, розроблений компанією Microsoft. Він є одним з найпопулярніших інтегрованих середовищ розробки (IDE) серед розробників. Visual Studio Code була випущена у 2015 році і швидко завоювала популярність завдяки своїй легкості використання, розширюваності та силам, які вона надає розробникам.

VS Code розроблений на основі проєкту Electron. VS Code може працювати на різних операційних системах, включаючи Windows, macOS та Linux.

Це зручне інтегроване середовище розробки, оснащене багатим функціоналом, який допомагає розробникам писати код більш ефективно. VS Code має масив розширень, які можна встановлювати для налаштування та розширення функціонала редактора. Це дозволяє користувачам адаптувати редактор до своїх потреб. Visual Studio Code має вбудовану підтримку систем контролю версій, таких як Git. Це дозволяє легко працювати із репозиторіями, переглядати історію змін (commit), вирішувати помилки та багато іншого, без необхідності перемикатися на зовнішні інструменти. VS Code надає потужні інструменти для відладки коду. Є можливість встановлення точок зупину,

крокування по коду, перегляду значень змінних та багато іншого. Це спрощує процес виявлення та усунення помилок при розробці. Також у VS Code є інтелектуальні функції, які автоматично надає доповнення коду та перевіряє синтаксис на помилки. VS Code підтримує безліч мов програмування у тому числі Python. Це робить його універсальним інструментом для розробки різноманітних проєктів.

Для контролю версій я буду використовувати Git, до того ж він вже інтегрований у Visual Studio Code.

Git є потужним і популярним інструментом для керування версіями програмного коду та спільної роботи над проєктами. Git розроблений у 2005 році Лінусом Торвальдсом.

Git відрізняється від інших систем контролю версій тим, що він працює з розподіленою моделлю. Кожен клон репозиторію Git є повноцінною копією історії проєкту з усіма його гілками (branch), комітами (commit) та змінами у файлах. Це означає, можна працювати над проєктом навіть при відсутності мережевого підключення, а пізніше синхронізувати свої зміни з репозиторієм.

Основні поняття в Git:

- *Репозиторій*: Git зберігає всю інформацію про проєкт у репозиторії. Це може бути локальний репозиторій на локальному пристрої або віддалений репозиторій на сервері, наприклад GitHub.
- *Коміт (commit)*: Він становить собою зафіксований стан проєкту у момент написання коміту. Він включає зміни файлів, повідомлення, яке пояснює зміни, ім'я користувача, що вносив зміни та його email, а також час змін.
- *Гілки (branch)*: Вони дозволяють розробникам працювати паралельно над різними функціоналом або задачами. Кожна гілка являє собою окрему лінію розвитку, в якій вносяться зміни. Після завершення роботи над гілкою можна її злити з основною гілкою (зазвичай з гілкою «master» або «main»).

- *Злиття (merge)*: Злиття в Git дозволяє об'єднати зміни з однієї гілки в іншу, при цьому в основній гілці створюються файли яких немає в гілці.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ТЕЛЕГРАМ-БОТА

2.1. Проектування архітектури та алгоритму роботи бота

Перед початком роботи над експериментальною (практичною) частиною проєкту необхідно встановити вимоги до нього, визначити функціональність, продумати архітектуру програмного засобу та алгоритм його роботи.

Telegram-бот має відповідати наступним вимогам:

- Бути простим у користуванні.
- Мати зручний та гарний інтерфейс.
- Бути доволі швидким та ефективним.

З функціональності бот має вміти:

- Зберігати, очищати назви фільмів та виводити цей список користувачу.
- Рекомендувати фільми на основі жанрів або за схожістю з визначеним фільмом.
- Виводити випадкові фільми з посиланням на нього.
- Виводити Топ-100 фільмів за версією IMDb.

Telegram-бот, а саме його директорія буде складатися з головного файлу «main.py» та 3 модулів, кожен з яких відповідає за відповідний функціонал або збереження важливих змінних, та додаткових необхідних файлів (див. рис. 2.1.).

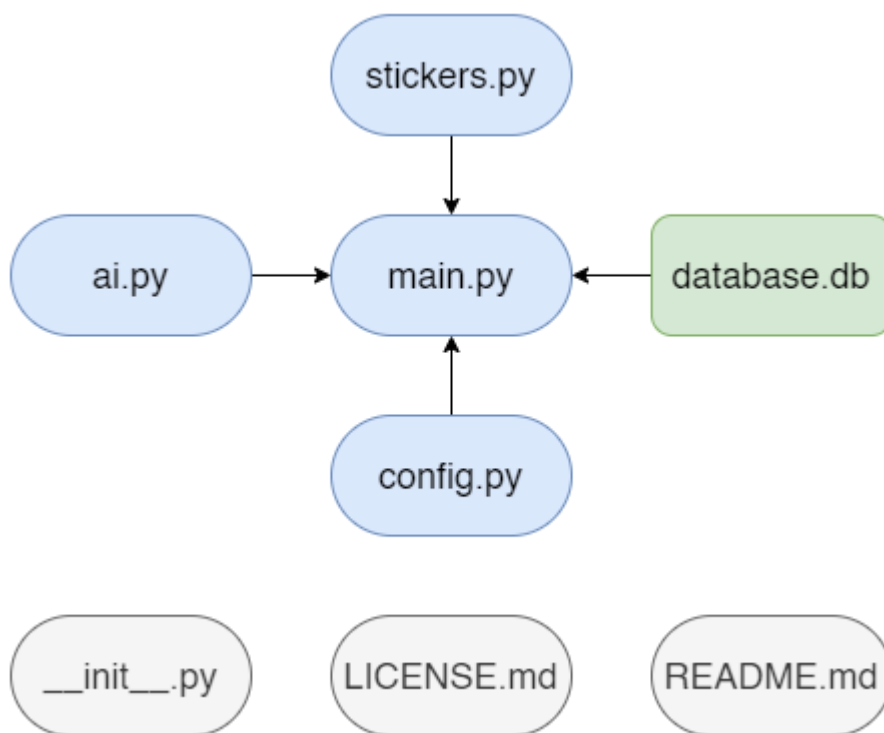


Рис. 2.1. Архітектура телеграм-боту та його директорії

«ai.py» потрібний для функціонала штучного інтелекту. Саме в цьому модулі буде відбуватись основний процес обробки запита від користувача та передачі його мовній моделі штучного інтелекту.

«config.py» для збереження токенів від TelegramBot API та OpenAI API. Відокремлення токенів від основного файлу «main.py» зумовлено забезпеченням безпеки для наших токенів, які бажано зберігати окремо від основного коду програми.

«stickers.py» в цьому файлі будуть зберігатись спеціальні ID від стікерів (наліпок) які буде надсилати бот користувачу перед кожним повідомленням.

Важливим елементом є файл бази даних – «database.db», він буде зберігати інформацію про користувача, а саме його унікальний ID та назви фільмів які він записав. Ця база даних повинна автоматично створюватись, якщо вона не міститься в директорії бота.

Файл «__init__.py» використовується для вказання, що певна тека є пакетом Python. Основна його мета полягає в ініціалізації пакета під час

імпортування. Цей файл необхідний для підключення модулів до файлу «main.py».

Не менш важливими файлами є «LICENSE.md» та «README.md». Хоч вони і не належать до функціоналу самого бота, але вони виконують роль документації та інструкції як використовувати бота. Я планую опублікувати телеграм-бота у відкритий доступ, тому важливо мати документ «LICENSE.md» який вказує на яких умовах можна його використовувати.

Алгоритм роботи можна розділити на головний процес, та процес роботи модуля ШІ. Алгоритм роботи штучного інтелекту виглядає наступним чином (див. рис. 2.2.).



Рис. 2.2. Блок-схема алгоритму роботи модуля ШІ

Алгоритм роботи головного файлу «main.py» виглядає наступним чином (див. додаток А).

Починатись алгоритм буде з підключення бібліотек, а саме головною буде `openai`, яка і відповідає керуванням штучного інтелекту. Далі йде

ініціалізація токенів від OpenAI API, які необхідні для роботи зі штучним інтелектом.

Варто розглянути саму мовну модель штучного інтелекту, а також алгоритм обробки запита від користувача. ШІ використовує глибокі нейронні мережі з великою кількістю шарів та нейронів. Загальна кількість параметрів в моделі становить мільярди. Модель побудована на основі трансформерної архітектури, яка включає у себе багатоголові уваги, розташовані в послідовній мережі.

Одна з ключових особливостей мовної моделі ШІ полягає в її здатності до самонавчання на великих обсягах текстових даних. Модель була піддана інтенсивному тренуванню на різноманітних джерелах, таких як Інтернет, книги, наукові статті та інші текстові ресурси. Процес тренування включав в себе безліч ітерацій, під час яких модель навчалася зв'язувати контекст з вихідним текстом.

Крім того, модель володіє широким лексичним запасом та граматичною точністю. Вона може розуміти та відтворювати тексти в різних мовах, включаючи англійську, іспанську, французьку, німецьку, українську та багато інших. Це було також важливим критерієм при виборі моделі ШІ.

Маючи певні параметри, які будуть задані в майбутньому в залежності від потреб. Модель насамперед отримує `prompt`, який наперед заготовлений, наприклад, наступним чином: «Виведи мені добірку фільмів за жанром (*текст_користувача*)», де запит може містити текст користувача. Користувач вводить назву жанру, яка записується в змінну. Зміна ж у свою чергу використовується в `prompt` для ШІ, після обробки запиту виводить відповідь у форматі JSON, з великою кількістю інформації, наприклад, час запиту, час обробки і т.д.. Нас цікавить тільки параметр `response` який варто буде витягти з відповіді та відправити користувачу. В залежності від потреб сам `prompt` змінюється для отримання потрібного результату. Таким чином працює алгоритм ШІ в моєму проєкті.

2.2. Розробка телеграм-бота у середовищі розробки

Перед початком кодування потрібно встановити пакети бібліотек `pyTelegramBotAPI`, `OpenAI API` та `IMDbPY`. Для цього в консоль Windows або в PowerShell потрібно ввести наступні команди.

Для встановлення `pyTelegramBotAPI`:

```
python3 -m pip install telebot
```

Для встановлення `OpenAI API`:

```
python3 -m pip install imdbpy
```

Для встановлення `IMDbPY`:

```
python3 -m pip install openai
```

А також потрібно, ініціалізувати Git-репозиторій у директорії телеграм-бота. Для цього потрібно зайти в Git Bash та прописати шлях до директорії, у моєму випадку це наступні команди:

```
cd Desktop/Cinebot
```

```
git init
```

Тепер все готово для початку розробки телеграм-бота. Першим кроком можна вказати в файлі `config.py` згенеровані токени. Для цього використаємо змінну `TOKEN` для токена від телеграм-бота, а `AIkey` для токена від OpenAI. А також додаймо декілька стікерів в «`stickers.py`».

Наступним кроком буде кодування головного файлу під назвою «`main.py`». В першу чергу потрібно імпортувати обрані бібліотеки, це 3 наші модулі `config`, `ai`, `stickers`, бібліотека фреймворку для створення телеграм-ботів `telebot` (`pyTelegramBotAPI`), бібліотека для створення баз даних `sqlite3`, вбудована бібліотека `random` для генерації випадкових чисел, бібліотека `openai` для доступу до мовних моделей ШІ, бібліотека бази даних фільмів `imdb`. А також потрібно імпортувати функцію `types` з `telebot` для створення клавіатури користувача. (див. лістинг 1).

Лістинг 1 – Підключення бібліотек в «`main.py`»

```
import config
```

```
import ai
```

```

import stickers
import telebot
import sqlite3
import random
import openai
import imdb
from imdb import IMDb
from telebot import types

```

Тепер потрібно ініціалізувати токен для того, щоб зв'язати створеного бота з його кодом. Та під'єднати існуючу базу даних, але у випадку якщо її не існує потрібно створити базу даних з таблицею messages в якій будуть об'єкти user_id, який має тип даних INTEGER та message, який має тип даних TEXT (див. лістинг 2).

Лістинг 2 – Створення бази даних в «main.py»

```

# токен телеграм-бота
bot = telebot.TeleBot(config.TOKEN)

# створення об'єкту IMDb
ia = IMDb()

# підключення бази даних
conn = sqlite3.connect('C:\\Users\\usach\\Desktop\\CineBot\\database.db',
check_same_thread=False)

# створення курсора
cursor = conn.cursor()

# створення бази даних якщо її не існує
cursor.execute("CREATE TABLE IF NOT EXISTS messages
                (user_id INTEGER, message TEXT)")

conn.commit()

```

Наступним кроком потрібно написати обробник команди /start, а також це буде перше, що буде виводити бот при першому запуску його через кнопку «Розпочати» (див. лістинг 3). Створюється клавіатура, яка має параметри «resize_keyboard = True», що дозволяє адаптувати клавіатуру під різні пристрої. Далі, створюється кожна кнопка, яка містить в собі текст команди, та додається саму клавіатуру. Бот виводить нам стікер та вітальне повідомлення, в якому за допомогою «{0.first_name}» дізнається ім'я користувача та звертається саме до цього конкретного користувача. За допомогою «parse_mode='html'» можна використовувати HTML теги для форматування тексту.

Лістинг 3 – Обробник команди /start в «main.py»

```
# команда /start
@bot.message_handler(commands=['start'])
def start(message):
    # створення клавіатури
    markup = types.ReplyKeyboardMarkup(resize_keyboard = True, row_width
= 3)

    # створення кнопок для клавіатури
    item1 = types.KeyboardButton('Додати запис')
    item2 = types.KeyboardButton('Показати записи')
    item3 = types.KeyboardButton('Очистити список')
    item4 = types.KeyboardButton('Топ-100 фільмів')
    item5 = types.KeyboardButton('Рандомний фільм')
    item6 = types.KeyboardButton('Схожий фільм')
    item7 = types.KeyboardButton('Добірка за жанром')

    # додаємо кнопки до клавіатури
    markup.add(item1, item2, item3, item4, item7, item6, item5)

    # вивід вітального повідомлення
    bot.send_sticker(message.chat.id, stickers.sticker_hi)

    bot.send_message(message.chat.id, "Привіт, <b>{0.first_name}</b>! \nЯ
ваш персональний помічник у світі кіно - <b>CineBot</b>.\nЧим саме я можу
```

допомогти?\n\n*<i>(Оберіть один з варіантів на*
*клавіатурі)</i>".format(message.from_user), parse_mode='html', reply_markup =
 markup)*

Наступним кроком буде створення обробника тексту (див. лістинг 4). Насправді клавіатура працює як заготовлений текст, який відправляється в чат. Тому за допомогою if можна перевіряти текст з якої кнопки було надіслано. Це і потрібно використати при обробці клавіатури. По-перше, проходить перевірка чи повідомлення є приватним. Далі, якщо текст користувача «Додати запис», то бот надсилає стікер та надсилає повідомлення з запитом «Надішліть мені назву фільму» чекаючи поки користувач не введе наступне повідомлення. Отримавши повідомлення він визиває функцію adddb.

Лістинг 4 – Обробник клавіатури якщо кнопка «Додати запис» в «main.py»

```
@bot.message_handler(content_types=['text'])
def menu(message):
    #якщо це приватне повідомлення
    if message.chat.type == 'private':
        #якщо це кнопка "додати запис"
        if message.text == 'Додати запис':
            bot.send_sticker(message.chat.id, stickers.sticker_type)
            msg = bot.send_message(message.chat.id, 'Надішліть мені назву
фільма, я її запишу!')
            # очікує текст від користувача та визиває функцію adddb
            bot.register_next_step_handler(msg, adddb)
```

Функція adddb має наступний вигляд (див. лістинг 5). Вона додає текст та id користувача у зміну та вносить зміни в базу даних. Після чого надсилає стікер з текстом про успішне збереження.

Лістинг 5 – функція adddb в «main.py»

```

def addddb(message):
    user_id = message.from_user.id
    message_text = message.text
    # Додавання даних до бази даних
    cursor.execute(f"INSERT INTO messages (user_id, message) VALUES
({user_id}, '{message_text}')"
    conn.commit()
    bot.send_sticker(message.chat.id, stickers.sticker_fun)
    bot.reply_to(message, 'Я записав ваш фільм!')

```

Якщо натискається кнопка «Показати записи», то виконується наступний фрагмент коду (див. лістинг 6). Спочатку у змінну зберігається id користувача для пошуку його в базі даних, потім відбувається вибірка даних яка записується в змінну «results» та виводиться користувачу разом зі стікером. Якщо змінна «results» має нульове значення, тобто ніяких фільмів користувача не записано в базі даних, то виводить повідомлення про відсутність записів.

Лістинг 6 – Обробник клавіатури якщо кнопка «Показати записи» в «main.py»

```

#якщо це кнопка "показати запис"
elif message.text == 'Показати записи':
    user_id = message.from_user.id
    # вибірка даних з бази даних для даного користувача
    cursor.execute(f"SELECT message FROM messages WHERE user_id
= {user_id}")
    results = cursor.fetchall()
    # відправлення результатів користувачу
    if results:
        message_text = "\n".join([result[0] for result in results])
        bot.send_sticker(message.chat.id, stickers.sticker_hug)

```

```

        bot.send_message(message.chat.id, f"Ваш список збережений
        фільмів:\n\n{message_text}")
    else:
        bot.send_sticker(message.chat.id, stickers.sticker_question)
        bot.send_message(message.chat.id, 'У Вас поки немає записаних
        фільмів.')

```

Якщо натискається кнопка «Показати записи», то виконується наступний фрагмент коду (див. лістинг 7). Який видаляє дані з бази даних за id користувача. Тому, можна не боятися, що користувач очистить дані всієї бази, він зможе очистити лише свої. Після чого бот виводить стікер та повідомлення про успішне очищення списку.

Лістинг 7 – Обробник клавіатури якщо кнопка «Очистити список» в «main.py»

```

#якщо це кнопка "очистити записи"
    elif message.text == 'Очистити список':
        # видаляє дані для конкретного користувача
        cursor.execute(f"DELETE FROM messages WHERE user_id =
        {message.from_user.id}")
        conn.commit()
        bot.send_sticker(message.chat.id, stickers.sticker_yes)
        bot.send_message(message.chat.id, "Ваші записи були очищені!")

```

Якщо натискається кнопка «Топ-100 фільмів», то виконується наступний фрагмент коду (див. лістинг 8). Який, спочатку отримує топ-250 фільмів за допомогою бібліотеки IMDb. Потім відповідь підготовлюється для відповіді використовуючи цикл від 1 до 100 для виводу лише 100 фільмів. Бот надсилає стікер та відповідь зі списком топ-100 фільмів.

Лістинг 8 – Обробник клавіатури якщо кнопка «Топ-100 фільмів» в «main.py»

#якщо це кнопка "top 100"

```
elif message.text == 'Топ-100 фільмів':
    # отримуємо топ 250 фільмів
    top_movies = ia.get_top250_movies()
    # формування тексту повідомлення
    text = "Топ-100 фільмів за версією IMDb:\n\n"
    for i in range(100):
        text += f'{i+1}. {top_movies[i]['title']} ({top_movies[i]['year']})\n'
    # відправлення повідомлення з топ-100 фільмів
    bot.send_sticker(message.chat.id, stickers.sticker_ride)
    bot.send_message(message.chat.id, text)
```

Якщо натискається кнопка «Рандомний фільм», то виконується наступний фрагмент коду (див. лістинг 9). Для цього знову отримується список з топ-250 фільмів, та за допомогою бібліотеки random отримується id випадкового фільму, який зберігається в змінній «random_movie». Далі, формується повідомлення в якому міститься назва, рік, жанр та режисер фільму. До того ж створюється inline-клавіатура з кнопкою, яка посилає на сторінку фільму на вебсайті IMDb. Бот надсилає стікер разом з заготовленим повідомленням.

Лістинг 9 – Обробник клавіатури якщо кнопка «Рандомний фільм» в «main.py»

```
# якщо це кнопка "рандомний фільм"
elif message.text == 'Рандомний фільм':
    top_movies = ia.get_top250_movies()
    # повертаємо 1 фільм з вибірки у 100 фільмів
    random_movie = ia.get_movie(top_movies[random.randint(0, 99)].getID())
    # формування тексту повідомлення
    title = random_movie['title']
    year = random_movie['year']
```

```

genres = ', '.join(random_movie['genres'])
directors = ', '.join([director['name'] for director in
random_movie['directors']])
response = f'{title} ({year})\n\n'
response += f'Жанр(u): {genres}\n'
response += f'Режисер(u): {directors}\n'
imdb_id = random_movie.getID()
# створюємо інлайн кнопку для переходу до фільму
markup = types.InlineKeyboardMarkup()
markup.add(types.InlineKeyboardButton("Відкрити на IMDb",
url=f"https://www.imdb.com/title/tt{imdb_id}/"))
bot.send_sticker(message.chat.id, stickers.sticker_fun)
bot.send_message(message.chat.id, "Ваш випадковий фільм:")
bot.send_message(message.chat.id, response, reply_markup = markup)

```

Якщо натискається кнопка «Схожий фільм», то виконується наступний фрагмент коду (див. лістинг 10). Бот надсилає стікер та просить вести назву фільму, та зберігає його в змінній та викликає функцію «AiRec.rec_name», що знаходить в модулі «ai.py». Що виконує ця функція буде розказано далі при кодуванні файлу «ai.py».

Лістинг 10 – Обробник клавіатури якщо кнопка «Схожий фільм» в «main.py»

```

# якщо це кнопка "схожий фільм"
elif message.text == 'Схожий фільм':
    # просить вести назву фільму для пошуку
    bot.send_sticker(message.chat.id, stickers.sticker_type)
    msg = bot.send_message(message.chat.id, 'Введіть назву фільму,
щоб отримати добірку фільмів схожих на нього')
    # визиває функцію з файлу ai
    bot.register_next_step_handler(msg, ai.AiRec.rec_name)

```

Якщо натискається кнопка «Добірка за жанром», то виконується наступний фрагмент коду (див. лістинг 11). Бот надсилає стікер та просить вести назву жанру, та зберігає його в змінній та викликає функцію «AiRec.rec_genre», що знаходить в модулі «ai.py». Що виконує ця функція буде розказано далі при кодуванні файлу «ai.py».

Лістинг 11 – Обробник клавіатури якщо кнопка «Добірка за жанром» в «main.py»

```
# якщо це кнопка "добірка за жанром"
elif message.text == 'Добірка за жанром':
    # просить вести назву фільму для пошуку
    bot.send_sticker(message.chat.id, stickers.sticker_type)
    msg = bot.send_message(message.chat.id, 'Введіть жанр за яким
хочете отримати добірку')
    # визиває функцію з файлу ai
    bot.register_next_step_handler(msg, ai.AiRec.rec_genre)
```

Якщо сталася помилка, або користувач сам відправив якусь невідому команду, то буде виконуватись наступний фрагмент коду (див. лістинг 11). Він виводить стікер та повідомлення про те, що він не знає цієї команди.

Лістинг 12 – Обробник клавіатури якщо інший текст в «main.py»

```
else:
    oops = "Вибачте, я не знаю таку команду :("
    bot.send_sticker(message.chat.id, stickers.sticker_question)
    bot.send_message(message.chat.id, oops)
```

Обов'язково в кінці коду має бути «bot.polling(none_stop = True)», вона забезпечує роботу бота без перерви.\

Тепер перейдемо до кодування «ai.py». Виконаємо підключення бібліотек та прив'язку токенів, тут нам треба вказати відразу два токени від Telegram API та OpenAI API.

Лістинг 13 – Підключення бібліотек та токенів в «ai.py»

```
#підключення бібліотек

import telebot
import config
import openai
import stickers

#токени

openai.api_key = (config.AIkey)
bot = telebot.TeleBot(config.TOKEN)
```

При роботі з цією мовною моделлю, доволі часто виникає невелика помилка, яка візуально погано відображається у тексті. Результат може починатися з крапки, і для того, щоб це позбутися була створена функція, яка перевіряє текст, і якщо результат починається на крапку, то функція прибирає зайві символи (див. лістинг 14).

Лістинг 14 – Функція для видалення перших двох символів в «ai.py»

```
class AiRec:

    def check_and_remove_prefix(variable):
        if variable.startswith("."):
            variable = variable[2:]
        return variable
```

Функція працює наступним чином, вона генерує за допомогою мовної моделі штучного інтелекту результат (response) в залежності від внутрішніх параметрів. Модель – це параметр який відповідає, яку мовну модель використовується, у моєму випадку «text-davinci-003» на момент написання є найновішою мовною моделлю. Запит (prompt) – це саме те як можна надавати

запит до штучного інтелекту, я використав запит з необхідним мені змістом який доповнюється текстом користувача бота. Ще існують деякі параметри, як «max-tokens» – це максимальна кількість символів в одній відповіді. Параметр «temperature» використовується для контролю генерації тексту, він впливає на рівень "творчості" та випадковості відповідей моделі. Далі, бот збереже в змінну тільки текст відповіді, використає функція «check_and_remove_prefix» для перевірки на зайві символи, та відправить користувачу стікер з готовою відповіддю, а саме добіркою зі схожими фільмами (див. лістинг 15).

Лістинг 15 – Функція отримання схожих фільмів в «ai.py»

```
def rec_name(message):
    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=f"Виведи мені добірку фільмів схожих на: {message.text},
        виведи тільки пронумерований список фільмів з назвою на англійській мові та з
        роком виходу",
        temperature=0,
        max_tokens=1000,
        top_p=1.0,
        frequency_penalty=0.5,
        presence_penalty=0.0,
    )
    text=response['choices'][0]['text']
    text = AiRec.check_and_remove_prefix(text)
    bot.send_sticker(message.chat.id, stickers.sticker_fun)
    bot.send_message(message.chat.id, f"Ось ваша добірка за схожим
    фільмом:\n{text}")
```

Функція з добіркою за жанром «tes_genre» працює точно так само, як і «tes_name». Єдине, що відрізняється це запит.

Лістинг 16 – Функція отримання добірки за жанрами в «ai.py»

```

def rec_genre(message):
    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=f"Виведи мені добірку фільмів за жанром: {message.text},
        виведи тільки пронумерований список фільмів з назвою на англійській мові та з
        роком виходу",
        temperature=0,
        max_tokens=1000,
        top_p=1.0,
        frequency_penalty=0.5,
        presence_penalty=0.0,
    )
    text=response['choices'][0]['text']
    text = AiRec.check_and_remove_prefix(text)
    bot.send_sticker(message.chat.id, stickers.sticker_fun)
    bot.send_message(message.chat.id, f"Ось ваша добірка за
    жанрами:\n{text}")

```

На цьому кодування телеграм-бота завершено, залишилось зберегти зміни за допомогою Git Bash або використовуючи вбудований в VS Studio інтерфейс Git. Та написати технічну документацію в файл «README.md» (див. додаток Б) та ліцензію в «LICENSE.md».

2.3. Тестування телеграм-бота та оцінка ефективності телеграм-бота

Тестування функціонала телеграм-ботів – це процес перевірки правильності та надійності роботи функцій і можливостей телеграм-бота. Це включає в себе перевірку коректності обробки вхідних повідомлень, відповідність очікуваним результатам, взаємодію з користувачами та інші функції, що реалізовані в боті. Тестування функціонала телеграм-ботів допомагає виявити та виправити помилки, забезпечити якість роботи бота.

Для того, щоб то запрацював потрібно запуснути головний файл «main.py», це можна зробити просто запустивши його, або за допомогою VS Code (у разі помилки при запуску буде повідомлення про тип помилки, це допоможе її вирішити). Бот успішно запустився без помилок, далі необхідно перейти за посиланням на бота @cine_rec_bot. Нас зустрічає вітальний екран з інформацією «Що вміє бот?» (див. рис. 2.3.). Цю інформацію та картинку було додано через BotFather.

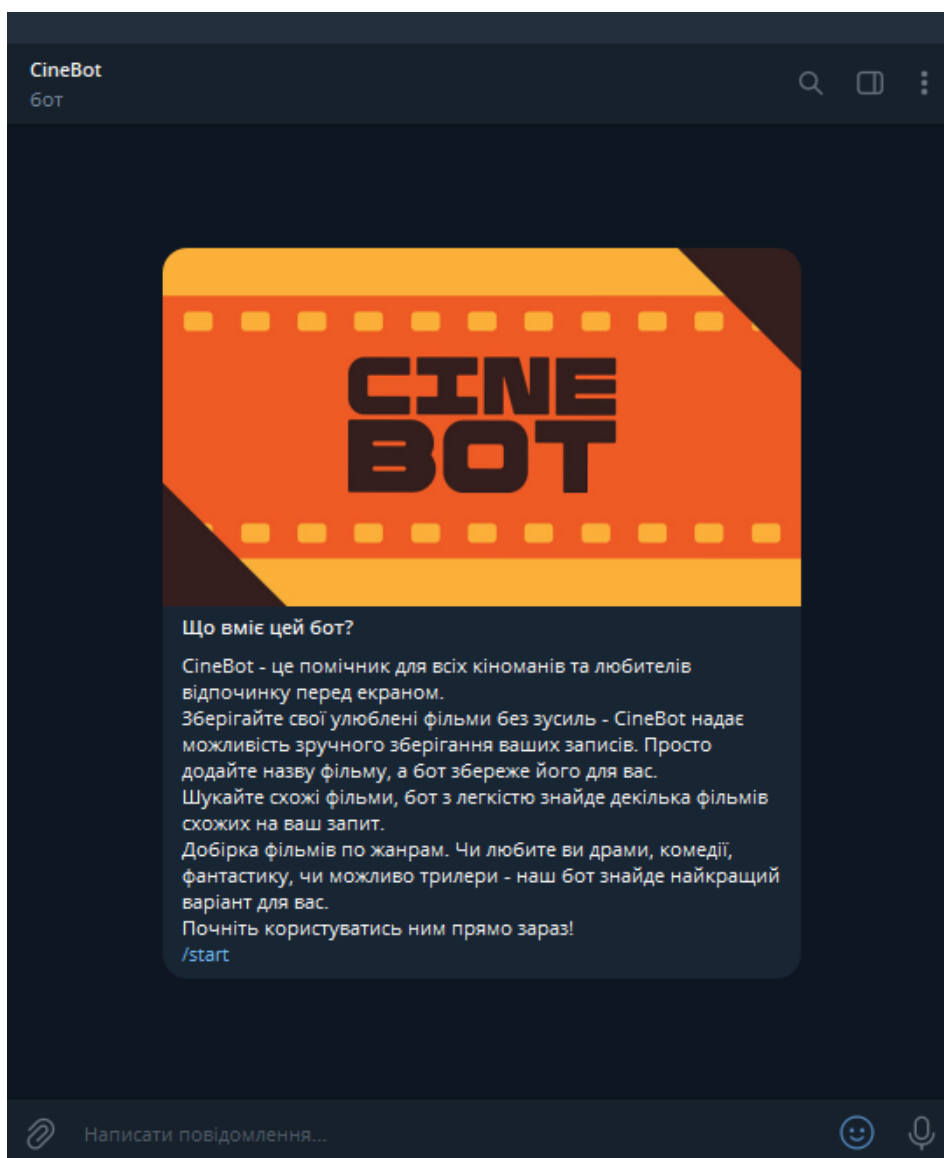


Рис. 2.3. Стартове вікно перед початком роботи

Далі потрібно натиснути кнопку «Розпочати» або прописати команду «/start», бот відправить на цю команду стікер, привітається та створиться зручна клавіатура вводу команд (див. рис. 2.4.).



Рис. 2.4. Виконання команди «/start»

Пропоную розглянути готовий елемент навігації клавіатуру більш детальноше (див. рис. 2.5.).

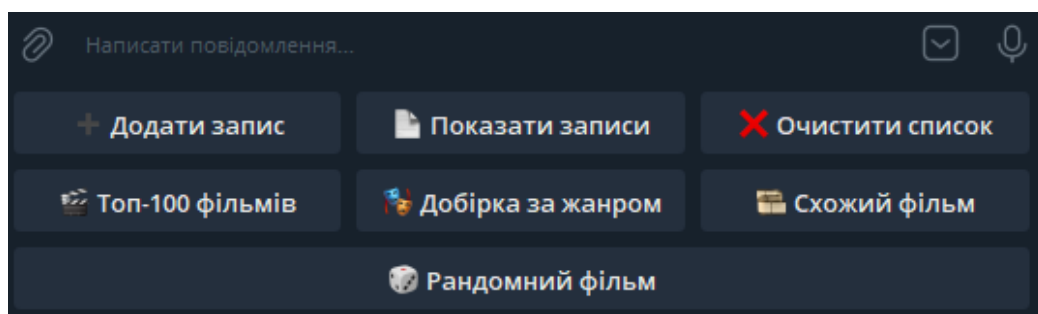


Рис. 2.5. Клавіатура для навігації

Клавіатура виглядає зручною, підтримує спеціальні символи такі, як «емодзі». Це полегшує розуміння, за що відповідає кожна кнопка. Розмір кнопок автоматично підлаштовується під пристрій.

Почнемо з першої кнопки «Додати запис», натиснемо на неї та напишемо боту фільм який ми хочемо записати, наприклад «Кримінальне чтиво» (див. рис. 2.6.).



Рис. 2.6. Виконання кнопки «Додати запис»

У результаті можна побачити, що бот успішно обробив наш запит та записав назву фільму в базу даних. Це можна перевірити відкривши файл «database.db» за допомогою застосунку SQLite (див. рис. 2.7.).

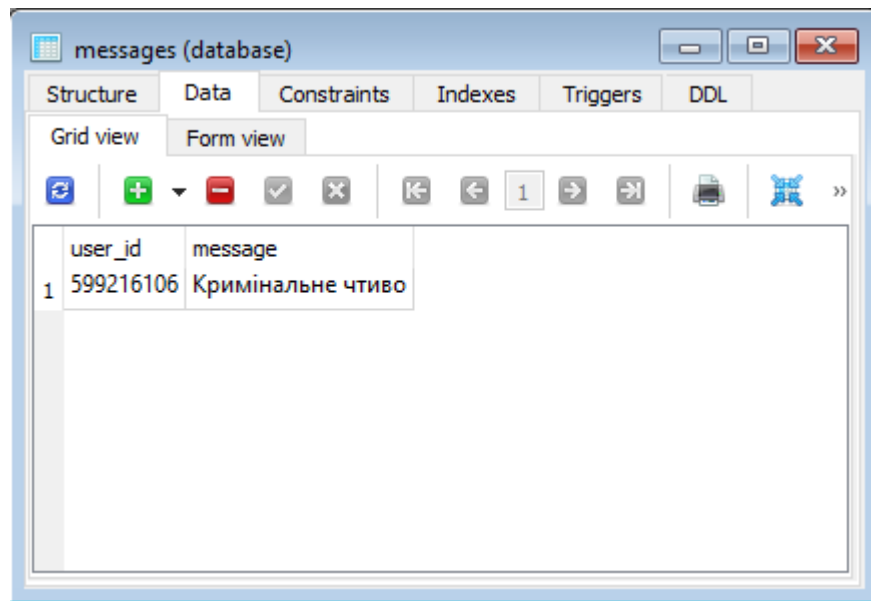


Рис. 2.7. Вигляд стану бази даних після запису даних

Дійсно, назва фільму записалась у базу даних разом з id користувача.

Тепер перевіримо другу кнопку «Показати записи», вона повинна показати користувачу список фільмів які він записав (див. рис. 2.8.).

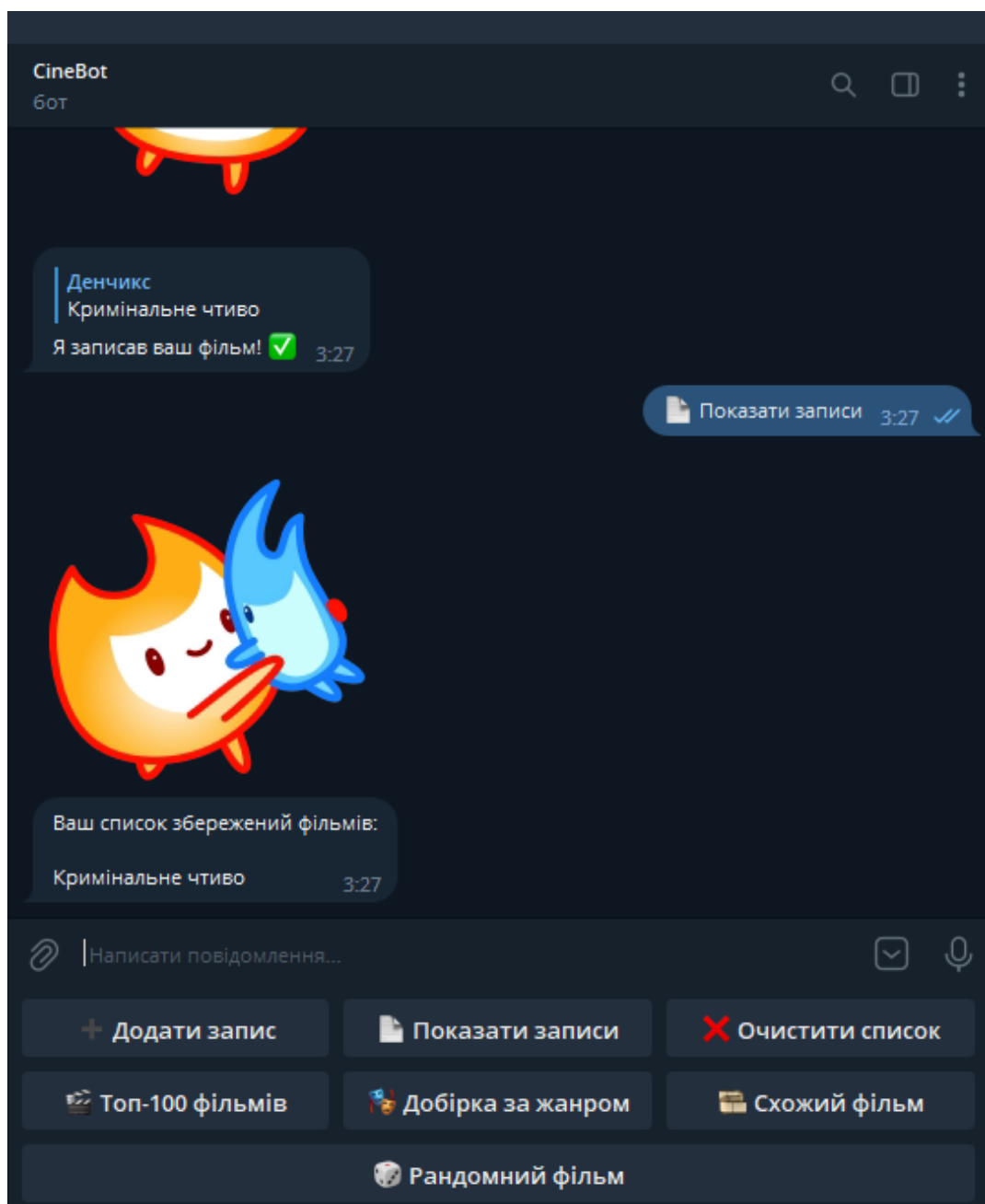


Рис. 2.8. Виконання кнопки «Показати записи»

Після обробки команди, бот видає повідомлення про список збережених фільмів в якому є записаний фільм.

Тепер спробуємо очистити список за допомогою кнопки «Очистити список» (див. рис. 2.9).



Рис. 2.9. Виконання кнопки «Очистити список»

Бот видає повідомлення про успішну очистку записів, перевіримо це викликавши знову команду «Показати список» (див. рис. 2.10.).



Рис. 2.10. Виконання кнопки «Показати список» при пустому списку

Тепер бот видає повідомлення про відсутність записів тому, що база даних була очищена користувачем. Отже, функціонал запису назв фільмів працює правильно та без помилок. Команди виконуються швидко без затримок тому, що ми використовуємо локальну базу даних. При розгортанні бота на віддаленому сервері швидкість команд буде залежати від швидкості передачі даних між сервером та комп'ютером.

Розглянемо функціонал інших кнопок, наприклад, «Топ-100 фільмів» (див. рис. 2.11.).



Рис. 2.11. Виконання кнопки «Топ-100 фільмів»

Бот отримав список фільмів Топ-100 фільмів за версією IMDb та вивів у вигляді повідомлення.

Наступна кнопка «Добірка за жанром» в основі виконання цієї команди лежить рекомендація на основі штучного інтелекту (див. рис. 2.12.). Потрібно ввести назву жанру на основі якого буде добірка фільмів, наприклад, комедія.



Рис. 2.12. Виконання кнопки «Добірка за жанром»

Бот вивів повідомлення з добіркою фільмів за жанром комедія. Проаналізувавши цей список, можна зробити висновок, що фільми з добірки мають жанр комедія. Отже, функція працює та виводить дійсно добірку за жанром. Хоча швидкість обробки довга тому, що запит відправляється на віддалені сервера.

Тепер перевіримо кнопку «Схожий фільм» в основі виконання якої також лежить штучний інтелект. Тепер має вийти добірка схожих фільмів, наприклад, на «Кримінальне читиво» (див. рис. 2.13.).



Рис. 2.13. Виконання кнопки «Схожий фільм»

Бот виводить повідомлення з добіркою схожих фільмів на «Кримінальне читиво». Проаналізувавши цей список, можна зробити висновок, що фільми з добірки дійсно схожі на «Кримінальне читиво». Наприклад перший фільм у списку «Скажені пси» від режисера Квентін Тарантіно дійсно схожий на

«Кримінальне читиво». Отже, функція працює та виводить дійсно добірку за схожістю.

Остання кнопка для перевірки це «Рандомний фільм» (див. рис. 2.14.).

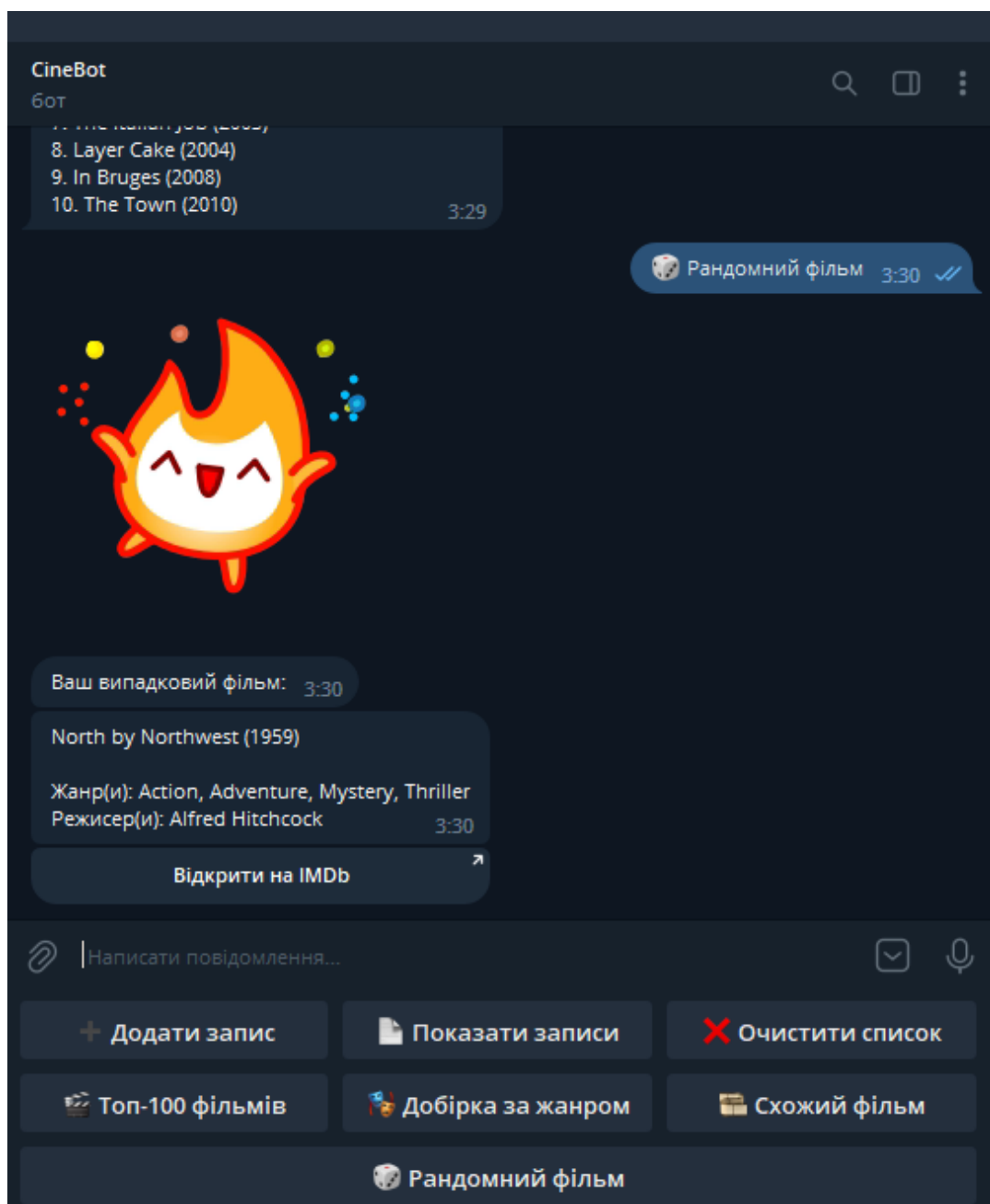


Рис. 2.14. Виконання кнопки «Рандомний фільм»

Бот виводить випадковий фільм з inline-кнопкою з посиланням на сайт з цим фільмом на IMDb. При натиску на кнопку ми потрапим на веб-сайт IMDb.

Залишилось перевірити, що виведе бот якщо буде якась команда яка боту невідома (див. рис. 2.15.).



Рис. 2.15. Вивід при невідомій команді

Бот виводить повідомлення про те, що він не знає цю команду.

Перевірку функціональності завершено, бот функціонує добре без проблем та помилок. Під час розробки було декілька багів, наприклад некоректний вивід повідомлення з добіркою, цю проблему та інші було вирішено. Бот працює ефективно та швидко. Навігація в боті відчувається зручною.

РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ ТЕЛЕГРАМ-БОТА

3.1. Порівняння розробленого бота з існуючими рішеннями

Мій телеграм-бот для споживачів відеоіндустрії «CineBot» є унікальним та покращеним проєктом порівняно з існуючими аналогами зі схожим функціоналом. Він включає більшість можливих функцій з інших ботів, проте покращує логіку роботи та зручність користування ними. Користувачам легше знайти все в одному боті ніж користуватися різними для різних потреб.

У моєму боті є деякий унікальний функціонал який не зустрічається в жодному знайденому мною боті, це збереження списку фільмів у боті та генерація добірки за схожістю. Це дає користувачам можливість зручно зберігати список фільмів і отримувати рекомендації на основі їх власних вподобань.

Багатьом аналогам не вистачає зручного інтерфейсу для користувачів, і вони вимагають введення команд вручну. У моєму боті я реалізував зручну клавіатуру, що допомагає швидше та ефективніше користуватися ним. Це відрізняє «CineBot» від конкурентів і робить його більш привабливим для користувачів.

3.2. Ідеї щодо додаткового функціоналу та рекомендації щодо подальшої розробки телеграм-бота

Для будь-якого проєкту є місце куди можна рости, телеграм-бот який вміє записувати список фільмів користувача та зберігати його, виводити топ-100 фільмів, рекомендувати рандомний фільм, добірки за жанром та схожістю, це вже хороший функціонал. Проте, деякі додаткові функції можуть поліпшити користувацький досвід і зробити бота більш привабливим для користувачів.

В мене є декілька ідей для додаткового функціонала мого телеграм-бота:

- **Рейтинг:** Можливість користувачам оцінювати фільми зі списку збережених фільмів. Потім згідно з оцінками, бот зможе вивести персональний рейтинг фільмів.
 - **Пошук за акторами та режисерами:** Додати можливість пошуку фільмів за іменами акторів або режисерів. Це дасть користувачам можливість знайти фільми з участю їхніх улюблених акторів або відомих режисерів.
 - **Рекомендації на основі списку збережених фільмів:** Аналіз списку збережених фільмів користувача і надання персоналізованих рекомендацій фільмів для доповнення списку, враховуючи їхні вподобання. Це зробить рекомендації більш точними та індивідуальними для кожного користувача.
 - **Інформація про акторів та режисерів:** Можливість отримати додаткову інформацію про акторів і режисерів, їх біографію, список фільмів в яких вони знялися або зняли.
 - **Планування перегляду:** Можливість користувачам планувати свої перегляди. Використовувати списки фільмів, які вони хочуть подивитися, для нагадування. А також отримувати нагадування про вихід прем'єру нових фільмів або серіалів.
 - **Додаткові інтеграції з іншими сервісами:** Можливість інтеграції з іншими платформами або сервісами, такими як Rotten Tomatoes або Netflix, щоб отримувати більш повну інформацію про фільми, їхні рейтинги для порівняння та доступність для перегляду.
 - **Розваги та вікторини:** Додати невеликі вікторини по фільмах та серіалам, на які користувач повинен вибрати правильну відповідь. Це надасть додатковий розважальний характер телеграм-бота.
- Даний функціонал можливо додати в майбутніх оновленнях телеграм-бота, оцінюючі дійсні потреби в цьому функціоналі.
- Щодо подальшого розвитку рекомендується:

- Збирати фідбек від користувачів щодо їхнього досвіду використання бота і їхніх потреб. Це допоможе виявити слабкі місця і зробити вдосконалення, щоб задовольнити потреби користувачів. Для цього необхідно додати додаткову функцію фідбеку в боті.
- Вдосконалювати алгоритми рекомендацій і пошуку. Штучний інтелект та машинне навчання добре допомагають поліпшити точність рекомендацій та швидкість пошуку фільмів.
- Розглянути можливість інтеграції з іншими платформами спілкування, такими як Facebook Messenger або WhatsApp, для збільшення охоплення користувачів.
- Враховувати потреби мобільних користувачів. Розробка мобільного додатку або адаптація бота для мобільних пристроїв може полегшити доступ користувачів до функціоналу та забезпечити зручність використання.

3.3. Використання телеграм-бота в реальних умовах, потенційні користувачі та можлива комерційна вигода

Телеграм-бот з функціоналом, пов'язаним з фільмами, може бути корисним широкому спектру користувачів та потенційним власникам такого бота. Ось декілька груп, які можуть скористатися таким ботом:

Кінолюбителі: Бот може бути корисним для будь-якої людини, яка цікавиться фільмами. Він допоможе користувачам зберігати назви фільмів, виводити рекомендації, відкривати нові фільми за жанром або схожістю. Кінолюбителям буде легше знайти фільми, які їм сподобаються, та отримати персоналізовані рекомендації.

Кінотеатри та платформи для стрімінгового відео: Бот може стати додатковим каналом для популяризації фільмів, які пропонують кінотеатри або платформи для стрімінгового відео. Вони можуть використовувати бота для

презентації нових фільмів, надання рекомендацій та нагадувань про вихід фільмів. Крім того, це може стимулювати взаємодію користувачів з їхніми послугами.

Кіножурналісти та блогери: Бот може бути корисним інструментом для кіножурналістів і блогерів, які пишуть про фільми. Вони можуть використовувати бота для збору інформації про фільми, створення списків та підготовки матеріалів для своїх статей або відео.

Рекламні агентства та маркетологи: Бот може бути ефективним інструментом для рекламних агентств та маркетологів, які працюють у сфері фільмів і розваг. Вони можуть використовувати бота для просування нових фільмів, організації розіграшів або пропаганди акцій і знижок пов'язаних з кінотеатрами або стрімінговими платформами.

Комерційна вигода такого телеграм-бота може полягати в:

- Залученні нових користувачів до кінотеатрів та стрімінгових платформ.
- Збільшенні популярності фільмів та покращенні їхнього продажу або перегляду.
- Посиленні зв'язку з аудиторією, збільшенні взаємодії та вірогідності повторного використання платформи або послуги.
- Генерації рекламних доходів через співпрацю з рекламодавцями або партнерами.

У кінцевому підсумку, користувачі, кінотеатри, платформи для стрімінгового відео, кіножурналісти, блогери та маркетологи можуть одержати значну користь від такого телеграм-бота.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи та розробки проєкту телеграм-бота для споживачів відеоіндустрії я мав можливість застосувати широкий спектр навичок та знань, які набув під час своєї освіти та практичного досвіду.

В першу чергу, я використав мову програмування Python, яка є популярною серед розробників програмного забезпечення. Використання Python дозволило мені ефективно реалізувати функціональність телеграм бота та забезпечити його плавну інтеграцію з Telegram API.

Також, в процесі розробки, я використовував різні фреймворки та бібліотеки Python, такі як pyTelegramBotAPI, OpenAI, API, IMDbPY та інші, для спрощення процесу розробки та забезпечення швидкого впровадження функцій та можливостей, які необхідні для бота. Це дозволило зекономити час і зусилля, що було особливо важливо в рамках дипломного проєкту.

Окрім того, я використовував систему контролю версій Git для збереження та керування кодом проєкту. Це дозволило мені стежити за змінами в коді, працювати над різними функціоналом паралельно, а також зручно відновлювати попередні версії коду у разі потреби.

Написання технічної документації було важливою частиною мого дипломного проєкту. Я детально описав функціональність, архітектуру та особливості реалізації бота, що дозволило зрозуміло представити його роботу і результати моєї роботи.

Нарешті, під час виконання проєкту, я продемонстрував свою здатність самостійно знаходити необхідну інформацію. У багатьох випадках, коли стикаючись з новими завданнями або проблемами, я проводив додаткові дослідження, шукав відповіді у документації та ресурсах спільноти розробників. Це допомогло мені успішно розв'язати складні завдання та досягти бажаних результатів.

У підсумку, завдяки використанню мови програмування Python, фреймворків та бібліотек, системи контролю версій та написання технічної

документації, я зумів успішно розробити та впровадити проєкт телеграм-бота для споживачів відеоіндустрії. Мій дипломний проєкт підтвердив мою здатність чітко формулювати результати та висновки, а також мою здатність самостійно знаходити необхідну інформацію для успішного виконання завдань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Barker S. Deep learning chatbots: everything you need to know | HackerNoon. *HackerNoon - read, write and learn about any technology*. URL: <https://hackernoon.com/deep-learning-chatbot-everything-you-need-to-know-r11jm30bc> (дата звернення: 20.05.2023).
2. Bots: An introduction for developers. *Telegram APIs*. URL: <https://core.telegram.org/bots> (дата звернення: 19.05.2023).
3. Chacon S. Pro git. Berkeley, CA : Apress, 2009. URL: <https://doi.org/10.1007/978-1-4302-1834-0> (дата звернення: 19.05.2023).
4. Cr0n. Що таке GIT та як почати працювати з ним?. *Medium*. URL: <https://blog.kr-labs.com.ua/що-таке-git-та-як-почати-працювати-з-ним-90c7f460f564> (дата звернення: 20.05.2023).
5. Documentation for Visual Studio Code. *Visual Studio Code - Code Editing. Redefined*. URL: <https://code.visualstudio.com/docs> (дата звернення: 19.05.2023).
6. Foong N. W. How to create telegram bot in Python. *Medium*. URL: <https://betterprogramming.pub/how-to-create-telegram-bot-in-python-cccc4babcc30> (дата звернення: 20.05.2023).
7. Git – Book. *Git*. URL: <https://git-scm.com/book/en/v2> (дата звернення: 19.05.2023).
8. Git – Вікіпедія. *Вікіпедія*. URL: <https://uk.wikipedia.org/wiki/Git> (дата звернення: 19.05.2023).
9. Günther T. Learn version control with git: a step-by-step course for the complete beginner. Independently Published, 2017.
10. How to Use Git {Beginner's Guide} + Getting Started with GitHub. *Knowledge Base by phoenixNAP*. URL: <https://phoenixnap.com/kb/how-to-use-git> (date of access: 20.05.2023).
11. IMDbPY. *PyPI*. URL: <https://pypi.org/project/IMDbPY/> (дата звернення: 19.05.2023).

- 12.Introduction – OpenAI API. *OpenAI API*. URL: <https://platform.openai.com/docs/introduction/next-steps> (дата звернення: 19.05.2023).
- 13.Jung A. Machine learning – the basics. Springer Singapore Pte. Limited, 2022.
- 14.Krishna A. How to create a telegram bot using Python. *freeCodeCamp.org*. URL: <https://www.freecodecamp.org/news/how-to-create-a-telegram-bot-using-python/> (дата звернення: 20.05.2023).
- 15.Lakshmanan V., Görner M., Gillard R. Practical machine learning for computer vision. O'Reilly Media, Incorporated, 2021.
- 16.Microsoft. Source Control with Git in Visual Studio Code. *Visual Studio Code - Code Editing. Redefined*. URL: <https://code.visualstudio.com/docs/sourcecontrol/overview> (дата звернення: 20.05.2023).
- 17.Modrzyk N. Building telegram bots: develop bots in 12 programming languages using the telegram bot API. Apress, 2018. 296 с.
- 18.Porter J. Deep learning: fundamentals, methods and applications. Nova Science Publishers, Incorporated, 2016. 76 с.
- 19.pyTelegramBotAPI. *PyPI*. URL: <https://pypi.org/project/pyTelegramBotAPI/> (дата звернення: 19.05.2023).
- 20.PYTHON - ЩО TAKE SQLITE І ЯК З НЕЮ ПРАЦЮВАТИ? – Олексій Лавриненко - PR, SMM, WEB. *Олексій Лавриненко - PR, SMM, WEB*. URL: <https://lavrynenko.com/uk/python-shho-take-sqlite-i-yak-z-neyu-pracyuvati/> (дата звернення: 20.05.2023).
- 21.Python IMDbPY – Searching a movie – GeeksforGeeks. *GeeksforGeeks*. URL: <https://www.geeksforgeeks.org/python-imdbpy-searching-a-movie/> (дата звернення: 20.05.2023).
- 22.Python IMDbPY – A library for movies - Javatpoint. *Javatpoint*. URL: <https://www.javatpoint.com/python-imdbpy-a-library-for-movies> (дата звернення: 20.05.2023).

23. Rahaman R., Hoffman C. How to get an OpenAI API Key. *How-To Geek*. URL: <https://www.howtogeek.com/885918/how-to-get-an-openai-api-key/> (дата звернення: 20.05.2023).
24. Social Network for programmers and developers. *Morioh*. URL: <https://morioh.com/p/4e1ea0961a1d> (дата звернення: 20.05.2023).
25. SQLite Tutorial - An Easy Way to Master SQLite Fast. *SQLite Tutorial*. URL: <https://www.sqlitetutorial.net/> (дата звернення: 20.05.2023).
26. Sqlite3 DB-API 2.0 interface for SQLite databases. *Python documentation*. URL: <https://docs.python.org/3/library/sqlite3.html> (дата звернення: 19.05.2023).
27. Telegram – Вікіпедія. *Вікіпедія*. URL: <https://uk.wikipedia.org/wiki/Telegram> (дата звернення: 20.05.2023).
28. Telegram APIs. *Telegram APIs*. URL: <https://core.telegram.org> (дата звернення: 13.05.2023).
29. Telegram Bot API. *Telegram APIs*. URL: <https://core.telegram.org/bots/api> (дата звернення: 19.05.2022).
30. Telegram FAQ. *Telegram*. URL: <https://telegram.org/faq> (дата звернення: 20.05.2023).
31. Visual Studio Code – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Visual_Studio_Code (дата звернення: 19.05.2023).
32. Welcome to python.org. *Python.org*. URL: <https://www.python.org> (дата звернення: 20.05.2023).
33. Yi J. Introduction to the Telegram API. *Medium*. URL: <https://towardsdatascience.com/introduction-to-the-telegram-api-b0cd220dbed2> (дата звернення: 20.05.2023).
34. Введення в базу даних Sqlite - Yoip. *Yoip*. URL: <http://yoip.com.ua/vvedennya-v-bazu-danih-sqlite/> (дата звернення: 20.05.2023).

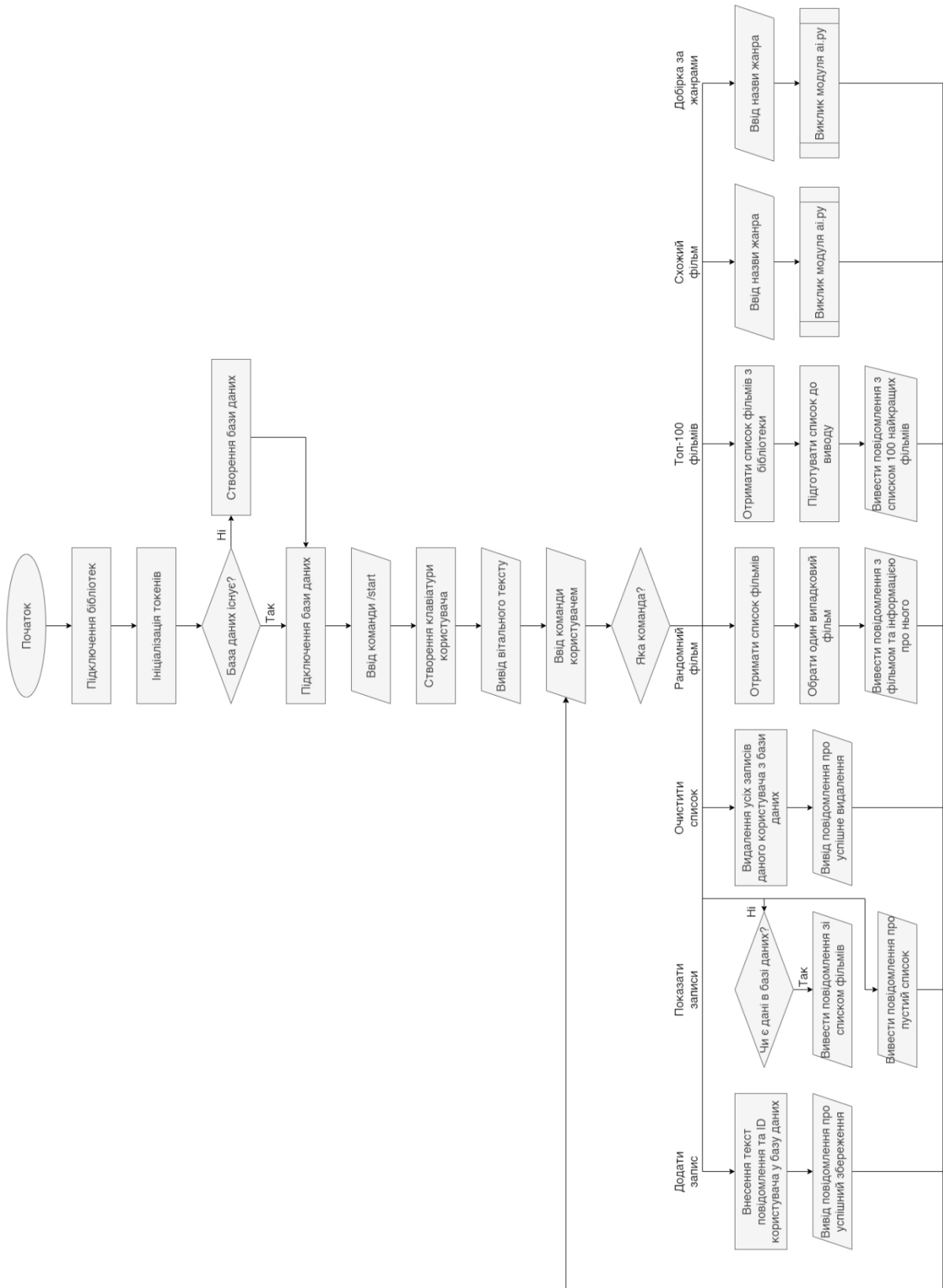
- 35.Іванова І. М. Боти у месенджерах як засіб автоматизації ділових процесів. Економічні науки. Серія "Економіка". 2019. Випуск 182. С. 109-116.
- 36.Карабута Н. О., Калюжна І. О. Огляд платформи розробки ботів та переваги її використання. Системні дослідження та інформаційні технології, 2018. Випуск 3. С. 48-54.
- 37.Путівник мовою програмування Python. *Путівник мовою програмування Python*. URL: <https://pythonguide.rozh2sch.org.ua/> (дата звернення: 20.05.2023).
- 38.Стандартна бібліотека Python. *Python documentation*. URL: <https://docs.python.org/uk/3/library/index.html> (date of access: 20.05.2023).
- 39.Штучний інтелект – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Штучний_інтелект (дата звернення: 20.05.2023).
- 40.Що таке GitHub і як з ним працювати | Безкоштовний онлайн-курс від компанії QATestLab. *Безкоштовний онлайн-курс від компанії QATestLab | Головна сторінка*. URL: <https://training.qatestlab.com/blog/technical-articles/what-is-github-and-how-to-work/> (дата звернення: 20.05.2023).

03.06.2023
(дата)


(підпис)

ДОДАТКИ

Додаток А
Блок схема-алгоритму «main.ru»



Додаток Б

Технічна документація

CineBot

Що таке **CineBot**? Це Telegram-бот, який вміє:

- зберігати назви фільмів та виводити цей список користувачу;
- рекомендувати фільми на основі жанрів або за схожістю з визначеним фільмом;
- виводити випадкові фільми;
- виводити Топ-100 фільмів за версією IMDb.

Важливо

Цей проєкт розробляється як дипломна робота здобувача вищої освіти рівня бакалавр.

Оновлення

[8 Травня 2023]

- Додано нові функції (добірка за жанром).
- Додані стікери при відправці повідомлення ботом.
- Виправлені дрібні помилки.

Встановлення

1. Встановити наступні пакети:

```
pip install telebot
```

```
pip install imdbpy
```

```
pip install openai
```

2. Клонувати проєкт:

```
git clone https://github.com/X3NK/CineBot
```

3. Додати токени **Telegram** та **OpenAi** у файл `config_EXAMPLE.py` та змінити назву файла на `config.py`.

4. Зайти в деректорію з ботом

5. Запустити файл `main.py`

```
python main.py
```

Використання

1. Відкрити вашого бота через **Telegram**

2. Натиснути кнопку "Розпочати" або прописати команду `/start`

Після цього відкриється клавіатура навігації, яка містить наступні кнопки:

-  **Додати запис**
Зберігає назву фільму, яку вводить користувач.
-  **Показати записи**
Виводить список збережених записів користувачем.
-  **Очистити список**
Повністю очищає список збережених фільмів
-  **Топ-100 фільмів**
Виводить список Топ-100 фільмів за версією IMDb
-  **Рандомний фільм**
Виводить випадковий фільм з посиланням на сторінку цього фільму на IMDb
-  **Схожий фільм**
Виводить список фільмів схожих на той, який ввів користувач
-  **Добірка за жанром**
Виводить добірку фільмів за жанром, який ввів користувач

Автор

Усачик Денис Костянтинович студент КБ-19-1

Ліцензія

Вихідний код у цьому репозиторії доступний за [MIT License](#)