НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

«До захисту допущено»
Завідувач кафедри
_____ О.І. Ролік
«___»_____2019 р.

Дипломний проект

на здобуття ступеня бакалавра

з напряму підготовки 6.050103 «Програмна інженерія»

на тему: «Телеграм-бот консультант з нотаріальних питань»

Виконав :
студент IV курсу, групи ІТ-51
Рязанцев Єгор Сергійович _____

Керівник:
Доцент кафедри АУТС Писаренко А.В.

_____

Рецензент:

_____

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2019 рік

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Рівень вищої освіти – перший (бакалаврський)
Напрям підготовки – 6.050103 «Програмна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.І. Ролік
«___»_____2019 р.

ЗАВДАННЯ
на дипломний проект студенту
Рязанцеву Єгору Сергійовичу

1.Тема проекту «Інтегрований веб-застосунок для допомоги у повсякденному житті» керівник проекту ст доцент кафедри АУТС Писаренко Андрій Володимирович, затверджені наказом по університету від «___»_____ 2019 р. №_____

2. Термін подання студентом проекту _____

3. Вихідні дані до проекту

Операційна система Windows 7, мова програмування C#, бібліотека TelegramBot, бібліотека ApiAiSDK.

4. Зміст пояснювальної записки

1. Вступ 2. Перелік умовних позначень 3. Аналіз предметної області дипломного проекту 4. Опис існуючих рішень 5. Математичний апарат 6. Опис власної реалізації 7. Тестування та тренування боту 8. Список використаних джерел

5. Перелік графічного матеріалу:

UML Діаграма варіантів використання, UML Діаграма активності телеграм-боту, UML Діаграма компонентів, UML Діаграма послідовності обробки користувацького запиту.

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1	Вибір тематичного напрямку та узгодження теми дипломного проекту	22.02.2019	
2	Аналіз теоретичних матеріалів та вивчення предметної області	15.04.2019	
3	Розробка технічного завдання, вибір методів та засобів реалізації задачі	24.04.2019	
4	Огляд існуючих рішень з тематики роботи	27.04.2019	
5	Реалізація проекту	20.05.2019	
6	Налагодження та перевірка програми	23.05.2019	
7	Оформлення пояснювальної записки	03.06.2019	
8	Передзахист дипломного проекту	04.06.2019	
9	Доопрацювання пояснювальної записки та підготовка презентації	19.06.2019	
10	Захист дипломного проекту	20.06.2019	

Студент Рязанцев Є.С.
Керівник проекту Писаренко А.В

АНОТАЦІЯ

Рязанцев Є.С. Телеграм-бот консультант з нотаріальних питань КПІ ім. Ігоря Сікорського, Київ, 2019.

Пояснювальна записка дипломного проекту складається з трьох розділів, містить 3 таблиці, 4 додатків, 11 рисунків та 22 джерел – загалом 73 сторінок.

Об`єкт дослідження: чат-бот для надання консультацій в нотаріальній сфері.

Мета дипломного проекту: розробити чат-боти який зможе розуміти природну мову користувача, зможе отримувати та оброблювати запити таким чином, щоб надавати необхідну інформацію стосовно вчинення нотаріальних правочинів, або продовжувати вести діалог і не вдасться досягти кінцевої цілі діалогу.

У першому розділі було проведено аналіз предметної області дипломного проекту.

У другому розділі було проведено огляд існуючих рішень по створенню чат-ботів та методів розпізнавання контексту.

У третьому розділі було описано математичний апарат методу розпізнавання контексту та навчання чат-ботів.

У четвертому розділі було описано алгоритм реалізації чат-боту для мессенджеру Telegram.

У додатках наведено: діаграма активності, діаграма компонентів, діаграма варіантів використання та діаграма послідовності виконання процесу обробки повідомлення.

КЛЮЧОВІ СЛОВА: БОТ, ЧАТ-БОТ, РОБОТ, КОНТЕКСТ, МЕССЕНДЖЕР, DIALOGFLOW, TELEGRAM.

ABSTRACT

Ryazantsev Ye.S. Telegram bot consultant on notary issues KPI them. Igor Sikorsky, Kyiv, 2019.

The explanatory note of the diploma project consists of three sections, contains 3 tables, 4 annexes, 11 figures and 22 sources - a total of 73 pages.

Object of research: chat bot for advice in the notarial field.

The purpose of the diploma project is to develop a chat room that can understand the natural language of the user, be able to receive and process requests in such a way as to provide the necessary information concerning the performance of notarial acts, or to continue the dialogue and fail to reach the ultimate goal of the dialogue.

In the first section, an analysis of the subject area of the diploma project was conducted.

In the second section, an overview of existing decisions on creation of chat bots and methods of recognition of the context was conducted.

The third section describes the mathematical method of the method of recognition of the context and the training of chat bots.

In the fourth section, an algorithm for implementing the chat-bot for the Telegram messenger was described.

The applications include: Activity Diagram, Component Diagram, Usage Diagram, and Message Processing Sequence Diagram.

KEYWORDS: BOT, CHAT-BOT, ROBOT, CONTEXT, MESSENGER, DIALOGFLOW, TELEGRAM.

Номер рядка	Формат	Позначення	Найменування	Кіл. листів	№ екз.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІТ51.210БАК.002 ПЗ	Пояснювальна записка	73		
6						
7	А3	ІТ51.210БАК.003 Д1	Діаграма активності	1		
8						
9	А3	ІТ51.210БАК.004 Д2	Діаграма варіантів використання	1		
10						
11	А3	ІТ51.210БАК.005 Д3	Діаграма послідовностей	1		
12						
13	А3	ІТ51.210БАК.006 Д4	Діаграма компонентів	1		
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						

ІТ51.210БАК.001 ТП

Змн.	Арк.	№ докум.	Підпис	Дата
Розроб.		Рязанцев Є.С.		
Перевір.		Тимофєєва Ю.С.		
Реценз.				
Н. Контр.		Шинкевич М.К.		
Затверд.				

Телеграм-бот нотаріальний консультант
Відомість технічного проекту

Літ.	Арк.	Акрушів
	1	1

КПІ ім. Ігоря Сікорського
ФІОТ, гр. ІТ-51

					ІТ51.210БАК.001 ТП		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Рязанцев Є.С.			Телеграм-бот нотаріальний консультант Відомість технічного проекту	Літ.	Арк. / Акрушів
Перевір.		Тимофєєва Ю.С.					1 / 1
Реценз.						КПІ ім. Ігоря Сікорського ФІОТ, гр. ІТ-51	
Н. Контр.		Шинкевич М.К.					
Затверд.							

ЗМІСТ

-

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення;

API – прикладний програмний інтерфейс (англ. application programming interface);

Dialogflow – продукти для комунікацій, серед яких є засоби створення ботів для різних платформ;

DF – те саме, що і Dialogflow.

Intent – варіант ведення діалогу для агентів Dialogflow, яка вибирається серед сукупності Intents;

Бот – спеціальна програма, що виконує автоматично за заданим розкладом які-небудь дії через ті ж інтерфейси, що й звичайний користувач;

Чат-бот – бот який працює в середині месенджера, соціальній мережі, основною функціональністю якого є ведення діалогу з користувачем;

Telegram — месенджер, програмне забезпечення для смартфонів, планшетів та ПК, яке дозволяє обмінюватися текстовими повідомленнями та різноманітними файлами;

Месенджер – програма для обміну повідомлення між користувачами;

ПК – персональний комп’ютер;

REST – передача стану подання (англ. Representation State Transfer);

XML – розширювана мова розмітки (англ. Extensible Markup Language);

HTTP – прикладний протокол передачі даних (англ. Hyper Text Transfer Protocol);

HTTPS – (HyperText Transfer Protocol Secure) розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки;

UML – стандартизованою мовою моделювання (англ. Unified Modeling Language);

Linq – мова інтегрованих запитів (англ. Language Integrated Query);

SDK – (software Development Kit) — набір із засобів розробки, утиліт і документації, який дозволяє програмістам створювати прикладні програми за визначеною технологією або для певної платформи (програмної або програмно-апаратної).

ВСТУП

Під час стрімкого розвитку різного роду технологій, необхідності автоматизації різноманітних процесів, особливо пов’язаних з інформацією, виникає потреба у роботизації рутинної праці для покращення умов життя суспільства.

Перші роботи були створені ще у середині 20-го століття, і від того часу попит на них тільки збільшувався, а їх можливості розширювалися. Під час розвитку технології створення ботів, та розвитку технології машинного навчання з’являлися нові і нові види ботів, які з часом інтегрувалися в життя людей.

Один з видів ботів, який з’явився в процесі еволюції та розвитку технології став чат-бот. Чат-боти – це віртуальні помічники, які зазвичай вбудовуються в месенджери та мають можливість розмовляти з реальними людьми. Їх головна функція – це вміння вести діалог з користувачем, і відповідно до функцій, закладених розробником, вони можуть отримувати, оброблювати, надавати певну інформацію. Вони можуть стежити за деякими подіями, отримувати та надсилати данні про погоду, можуть вести облік витрат та доходів, можуть знаходити різного роду інформацію, і звичайно просто покращувати настрій користувачу.

Популярність чат-ботів почала різко зростати приблизно з 2015 року. Одна із причин – появлення багатьох мобільних застосунків, кількість яких перевищує вже декілька мільйонів, а то і десятки мільйонів, але користувачі з часом втрачають бажання встановлювати нові додатки. 80% свого часу користувачі проводять в соціальних додатках, це месенджери типу Telegram, WhatsApp та інші. Вже у 2018 році загальна кількість користувачів месенджерів обігнала найпопулярніші соціальні мережі. Месенджери більш

захищенні від недоброзичливих осіб, вони не виставляють акаунти користувачів на загальний огляд, забирають менше ресурсів системи, не вимагають дорогих апаратних можливостей, а головне майже нема реклами, та іншої непотрібної інформації.

Ідея створення чат-бота для месенджерів (наприклад Telegram) базується на використанні відкритого API та https запитів. В ролі клієнта може виступати як вебсайт месенджера так і мобільна версія. Реалізацією ідеї може бути як сервіс, що створено власноруч так і об'єднання вже існуючих рішень з використанням їх переваг. Обране рішення має взаємодіяти з користувачем, отримувати повідомлення від нього, формувати та відправляти запит на сервер. Сервер, отримавши цей запит, обробляє його, та відправляє текст користувача на NLP сервіс, який повинен обробити «сирий» текст, повернути дані в придатному для обробки вигляді назад на сервер. Тому маючи усі вхідні дані про запит користувача, необхідно проаналізувати його, та сформувати відповідь і відправити її назад на месенджер.

Для автоматизації робочого процесу було обрано діяльність нотаріуса, а саме її консультативна частина, яка складається з питань-відповідей та уточнення наявності всіх необхідних документів для вчинення правочинів та інших нотаріальних дій. Дана підготовча частина діяльності нотаріуса потребує чимало часу, та вимагає оптимізації. Ці рутинні операції потрібно покласти на програмні застосунки, які буде можливо навчати та масштабувати для інших додаткових консультацій і роз'яснень.

Також дуже важливою складовою є реалізація певного методу розуміння ботом рідної мови користувача, та отримання з повідомлення, яке надсилає користувач саме важливої інформації, з якою вже можна працювати. Тобто важливою функціональністю чат-боту є вміння розуміти контекст повідомлень, а не тільки опрацювання заданих варіантів. Для цього

існує досить багато методів на основі нейронних мереж та методи машинного навчання, також є сервіси які надають необхідні можливості для реалізації мовних модулів для боту. Лідером у сфері обробки запитів користувачів, та розпізнавання контексту є компанія Google, яка надає два безкоштовних сервіси для інтеграції мовних модулів в створювані програмні продукти.

У процесі створення чат-бота, дуже важливо побудувати правильний хід взаємодії з користувачем. На основі багатьох досліджень [1], було виявлено, що чим більш вузькою є спеціалізація чат-бота, тим він більш ефективний та точний. Також у створенні програмного бота було приділено увагу забезпеченню простої та інтуїтивно зрозумілої взаємодії з користувачами. На основі дослідів, та опитувань [2], стало відомо те, що чим більше підказок чат-бот надає користувачеві, тим кращою буде взаємодія із ним.

Створення чат-ботів є досить актуальною темою, яка в свою чергу є певним історичним моментом переходу з різного роду застосунків до реалізації цієї функціональності за допомогою чат-боту, а також важливим є перехід від рутинної роботи працівників сфери консультування клієнтів та надання відповідей на телефонні дзвінки, що може впливати на емоційний стан консультантів, які там працюють, до автоматизації надання певних відповідей на запитання ботами, які в свою чергу ніколи ні на що не поскаржаться.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Відомості про ботів

Завданням дипломного проекту було обрано створення телеграм-боту консультанта з нотаріальних питань.

Бот (скорочення від «робот») – програма, яка автоматично по команді, або розкладом виконує різні дії.[3] Простіше кажучи, програма для здійснення рутинних операцій. Причому вона робить це через ті ж інтерфейси, що і звичайний користувач, як би імітуючи реального користувача.

Робота з ботом реалізована через використання методів Telegram API за допомогою звичайного HTTPS інтерфейсу [4].

Також потрібно зазначити, що з кожним днем, все більше фірм і компаній мають потребу не тільки у власних сайтах, а й ботах. За прогнозами на 2020 рік від Business Insider 80% компаній світу будуть мати та певним чином використовувати чат боти [5].

Більшість свого часу користувачі проводять у певних застосунках, це месенджери типу Telegram, WhatsApp, соціальні мережі Facebook, Instagram, застосунки для перегляду відео Yotube та інші. У 2018 році загальна кількість користувачів найпопулярніших месенджерів обігнала найпопулярніші соціальні мережі. Месенджери мають кращу приватність, забирають менше ресурсів системи, не вимагають дорогих апаратних можливостей, а головне майже нема реклами, та іншої непотрібної інформації.

Тому стало очевидно, що набагато простіше знайти до користувача там, де він проводить більшість свого часу, а не створювати нові застосунки, і намагатися нав’язати їх йому. Також чат-бот набагато простіше запустити, через те що його не потрібно скачувати та встановлювати на пристрій, не займає місця в пам’яті, та на дисплеї пристрою.

Працювати з чат-ботом просто: потрібно додати його в список контактів і почати переписку. Найчастіше у відповідь бот надішле інформацію про себе,

список доступних команд або виведе на екран кнопки, здатні перетворювати вікно діалогу в інтуїтивно-зрозумілий міні-застосунок.

Для того, щоб знайти необхідного бота можна використовувати Telegram bot store, де боти згруповані за категоріями, датою виходу, популярністю, та мовою використання.

Деякі види ботів використовуються для шкідливих цілей : для розсилання спаму, DoS та DDoS атак, відправки вірусів для збору та використання інформації проти певних користувачів, та іншого роду обману.

Зазвичай ботів використовують для автоматизації рутинних дій, які вони можуть виконувати самостійно або допомагати в пошуку інформації, в перекладі, відстежувати певні події, відповідати на повідомлення, проводити опитування, взаємодіяти з датчиками та речами, які підключені до інтернету, зберігати данні, нагадувати про певні події, вбудовуватися в інші сервіси і платформи, тобто вони можуть робити все, що їм буде запрограмовано відповідно до інструментів розроблення платформи.

1.2 Телеграм боти

В Telegram зазвичай використовуються чат-боти [6]. Для того щоб відрізняти їх від реальних користувачів було обумовлено, що їх назви повинні закінчуватися словом «bot». Реалізація аккаунта бота в телеграмі не суттєво відрізняється від реалізації аккаунта користувачів, головна відмінність, що аккаунтом бота керує програма, а не людина.

Бота можна відрізнити від звичайного облікового запису користувача за наступними ознаками:

1) замість статусів «онлайн» і «був у мережі», відображається напис «бот»;

2) через певний час повідомлення надіслані до ботів автоматично видаляються;

3) бот автоматично вступає в діалог тільки після підключення його до групи, або після запуску особистого діалогу;

4) у ботів відсутні можливості до самостійного початку діалогу з користувачем;

5) будь-який робот у своїй назві повинен закінчуватися ключовим словом “bot”;

6) без включеного режиму публічної приватності, бот не буде отримувати повідомлення, які надсилаються до групи; для того, щоб написати боту, потрібно вказати його назву, і після неї повідомлення;

7) ботам не потрібні додаткові ресурси, крім тих, які необхідні для виконання ними запрограмованого функціоналу.

Telegram боти підтримують ряд можливостей не доступних звичайним користувачам, наприклад, такі як: спеціальні клавіатури, додаткові інтерфейси для команд за замовчуванням, зовнішнє зв'язування і спеціальні режими приватності для груп.

1.2.1 Режим приватності ботів

Бот має два режими роботи в групі, перший використовується за замовчуванням, не дозволяє боту отримувати ніяких повідомлень з групи, окрім тих, в яких згадується ім’я бота, або які починаються з символу "/" (так зазвичай починаються команди боту). Завдяки цьому режиму бот отримує лише ті повідомлення, які стосуються саме його, і розробникам не потрібно опрацьовувати тисячі непотрібних повідомлень, які могли б приходити не за призначеннями при вимкненому режимі приватності, також деякі користувачі не бажають, щоб хтось інший отримував їх повідомлення, тому вони можуть бути спокійні, що бот не скористається цим повідомленням при включеному режимі приватності.

При необхідності можна виключити режим приватності і тоді бот буде отримувати всі повідомлення з групи, але без необхідності цього робити не варто. Зазвичай включений режим приватності забезпечує якісну та коректну роботу боту.

1.2.2 Команди ботів

Команди є загальним способом виклику певної події від боту, команди рекомендуються створювати за наступним шаблоном:

/ Команда [необов'язковий] [аргумент]

Кожна команда починається з символу косої риски «/» та має довжину до 32 символів. Команди використовують букви латинського алфавіту, цифри та підкреслення. Декілька прикладів:

/ Set_My_name BotKing

/ Send_message_at 5pm

/ Find something

Повідомлення, які починаються з косої риски, будуть завжди доставлятися боту (так само, як і при відповіді на його повідомлення і на @ згадки бота в чаті). Боти Telegram автоматично будуть:

а) створювати список відповідних команд та пояснення до них. При натисненні користувачем символу «/» буде працювати тільки при відповідних налаштувань описів команд у @BotFather. Відповідна команда буде оброблена при натисканні на відповідний опис;

б) відображати всі доступні команди, для яких створено опис при натисненні на кнопку (/);

в) виділяти команди в чатах, при натисненні на відповідне повідомлення команда буде оброблена ботом.

1.2.3 Процес створення телеграм-боту

Створення ботів в телеграмі майже для всіх однакове, для цього потрібно виконати декілька пунктів і бот буде створено:

- для початку потрібно в телеграмі в рядку пошуку ввести BotFather та вибрати потрібний контакт;
- далі ввести початкову команду для запуску ботів в телеграмі /start;
- далі ввести команду /newbot;
- ввести ім’я боту;
- ввести його назву для телеграму.

1.3 Поняття нотаріату

Вивчаючи та аналізуючи область нотаріату для дипломного проекту, було розглянуто, що таке нотаріат, хто такі нотаріуси, та які нотаріальні дії вони можуть вчиняти [7].

Нотаріат в Україні – це система органів і посадових осіб, які посвідчують права, факти, вчиняють нотаріальні дії, з метою надання їм юридичної сили.

Правова основа діяльності нотаріату – це Конституція України, Закон України «Про нотаріат» та інші законодавчі акти України.

Вчиненням нотаріальних дій в Україні займаються державні нотаріуси, які працюють в державних нотаріальних конторах, державних нотаріальних архівах та приватні нотаріуси, які займаються приватною нотаріальною діяльністю.

Нотаріус – це уповноважена державою фізична особа, яка здійснює нотаріальну діяльність. Нотаріусом може бути громадянин України з повною вищою юридичною освітою, який володіє державною мовою, склав

кваліфікаційний іспит і отримав свідоцтво про право на зайняття нотаріальною діяльністю.

Нотаріуси вчиняють такі нотаріальні дії:

- посвідчують правочини (договори, заповіти, довіреності тощо);
- вживають заходів щодо охорони спадкового майна;
- видають свідоцтва про право на спадщину;
- видають свідоцтва про право власності на частку в спільному майні подружжя в разі смерті одного з подружжя;
- видають свідоцтва про придбання майна з прилюдних торгів (аукціонів);
- видають свідоцтва про придбання майна з прилюдних торгів (аукціонів), якщо прилюдні торги (аукціони) не відбулися;
- провадять опис майна фізичної особи, яка визнана безвісно відсутньою або місце перебування якої невідоме;
- видають дублікати нотаріальних документів, що зберігаються у справах нотаріуса;
- накладають заборону щодо відчуження нерухомого майна (майнових прав на нерухоме майно), що підлягає державній реєстрації;
- засвідчують вірність копій (фотокопій) документів і виписок з них;
- засвідчують справжність підпису на документах;
- засвідчують вірність перекладу документів з однієї мови на іншу;
- посвідчують факт, що фізична чи юридична особа є виконавцем заповіту;
- посвідчують факт, що фізична особа є живою;
- посвідчують факт перебування фізичної особи в певному місці;
- посвідчують час пред'явлення документів;
- передають заяви фізичних та юридичних осіб іншим фізичним та юридичним особам;
- приймають у депозит грошові суми та цінні папери;

- вчиняють виконавчі написи;
- вчиняють протести векселів;
- вчиняють морські протести;
- приймають на зберігання документи.

Відповідно до зазначеного вище, було зроблено акцент на нотаріальних діях, які можуть вчиняти нотаріуси, та які документи потрібні для їх вчинення, і яку додаткову інформацію потрібно знати, перед зверненням до самого нотаріусу.

1.4 Висновки до розділу

Під час аналізу предметної області було встановлено, що бот – це програма, яка автоматизує певні процеси (отримання та обробка даних, вчинення відповідних дії до отриманих даних, проведення процесу створення чого-небудь тощо).

Одним з видів ботів є чат-боти. Чат-боти зазвичай використовуються для різного роду роботи з інформацією (збір даних з опитувань, побудова відповідних графіків, надання консультації замість реальних людей, відсилання сигналу щодо певних змін в певній системі тощо). Чат-боти дуже легко вбудовуються в системи, де відбувається процес спілкування з користувачем (соціальні мережі, месенджери, інтернет ресурси в яких відбувається процес спілкування).

Також було проаналізовано предметну область, а саме нотаріат, за допомогою Закону України «Про нотаріат», Порядку вчинення нотаріальних дій нотаріусами України, Правил ведення нотаріального діловодства. Було визначено, що входить в предметну область, хто такі нотаріуси, та які нотаріальні дії вони можуть вчиняти.

2 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

2.1 Огляд ботів консультантів

2.1.1 Weather bot

Weather bot [8] – бот для слідкування за погодою, за допомогою декількох простих операцій, натисканні на певні кнопки та введені міста, в якому потрібно дізнатися про погоду. Також бот може надати інформацію про погоду на поточний день та на наступні дні.

Оскільки значну частину людей в певний момент часу може турбувати питання, а яка сьогодні буде погода, чи буде дощ, і чи не потрібно взяти парасольку з собою, або дізнатися, яка буде погода на вихідні дні, щоб вибрати місце для відпочинку, weather bot став досить швидко популярним.

Weather bot має декілька простих функцій. При запуску бота, надається інформація про можливості цього бота, та його функції. Далі можна вибрати період (поточний день/декілька днів) на який потрібно дізнатися погоду, і після введення міста, в якому потрібно дізнатися погоду, бот надає необхідну інформацію, вказавши мінімальну та максимальну температуру, можливість опадів.

Також, при необхідності, користувач може налаштувати автоматичні повідомлення від боту, які будуть надсилати інформацію про погоду в певний момент часу.

Серед переваг можна виділити:

а) простота інтерфейсу;

б) швидка реакція на запит;

в) простий алгоритм реалізації.

Серед недоліків: пов’язаність з іншим застосунком, для отримання інформації про погоду.

2.1.2 GreenzBot

GreenzBot – фінансовий консультант, який допомагає вести рахунок доходів, витрат та інших грошових операцій, він синхронізується з таблицями, які підв’язані до облікового запису Google, розраховує баланс на певний момент часу. Після запуску боту, пропонує створити обліковий запис у три кроки. Для цього потрібно надіслати йому свою електронну адресу @gmail.com, після чого будуть ініціалізовані таблиці. Цей процес займає деякий час, тому після завершення кожного етапу бот надсилає відповідне повідомлення. Далі потрібно вказати боту часовий пояс і ліміт місячних витрат, на цьому процес реєстрації завершено.

Додатково наявні шість команд, які можна побачити ввівши в строку повідомлення знак "/":

- /balance – мій баланс;
- /table – посилання на google-таблицю;
- /sync – з’єднання з таблицею;
- /settings – налаштування;
- /help – список команд;
- /delete – видалення облікового запису в боті.

Коли потрібно додати витрати, необхідно ввести дату, категорію, та суму, яку витратили, після чого бот додає цю інформацію до таблиці, в якій можливо побачити всі проведені операції. Також бот розуміє скорочення і вміє пов’язувати зв’язані категорії. Бот нагадає про баланс, якщо баланс майже вичерпався.

Бот досить зручний у використанні і може знадобитися, як для ведення сімейного бюджету, так і для підрахунків витрат на певний захід, або витрат невеликої компанії.

Серед переваг боту можна виділити:

- збереження часу на заповнення таблиці, за рахунок простого вводу даних і надсилання їх до боту;
- можливість відслідковувати баланс, як в інтерфейсі Telegram, так і google-таблицях;
- бот розуміє скорочення та контекст повідомлення;
- повідомляє про важливі події або зміни в балансі;
- можна подивитися приклади заповнення повідомлення для бота за командою "/samples".
- Серед недоліків боту можна перерахувати:
- розробник боту може отримати данні про витрати користувача;
- для деяких користувачів може не вистачити функціональності;
- нема функції відміни останніх змін.

2.1.3 Chgk_bot

Chgk_bot, бот для перевірки своїх знань та кмітливості. Він зберігає в базі знань понад тисячу запитань та відповідей на них в різних категоріях, також дозволяє змагатися в турнірі з іншими учасниками. Для використання боту потрібно підписатися на новини бота, також є можливість запуску таймеру для слідкування за часом.

Для того, щоб отримати запитання, потрібно ввести команду "/next", далі ввести відповідь, і якщо вона правильна, то будуть виведені пояснення до неї. Також можна натиснути на одну з кнопок inline клавіатури для отримання відповіді, або перейти до наступного питання.

Бот є розважально-пізнавальним застосунком, який допоможе цікаво провести свій вільний час.

Інтерфейс боту не складний, але в довідці забагато функції, які майже не потрібні, також деякі з них виконують схожі дії.

Алгоритм дії також простий, користувач вводить команду вибору категорій, йому приходить відповідь в вигляді клавіатури, на якій він вибирає певну категорію. Далі програма оброблює цей запит, підключається до відповідної таблиці бази даних, зчитує звідти певне питання, і надсилає його користувачеві. Після вводу відповіді, бот звіряє її з вірною, і або надсилає пояснення по запитанню, або з'являється повідомлення, що відповідь не вірна і пропонує спробувати ще. При цьому ведеться облік питань, на які користувач відповів, щоб запитання не повторювалися. Бот цікавий тим, що використовує базу даних для збереження питань і стану прогресу користувача.

Серед переваг можна виділити:

- ведення статистики;
- збереження даних у базі даних.

Серед недоліків:

- бот не вміє розпізнавати контекст повідомлення;
- при відсутності з'єднання з базою даних, бот не зможе надіслати питання та вести статистику;
- має не потрібну функціональність, яка нагромаджена в довідці, що призводить до гіршого сприйняття.

2.1.4 OpenDataUABot

OpenDataUABot – застосунок для моніторингу відкритих даних в юридичній сфері, може бути корисним для юристів, підприємців та журналістів. Бот може знайти більшу частину українських підприємств за їх назвою, кодом з Єдиного державного реєстру підприємств та організацій України, або прізвищем керівника та буде надсилати інформацію про відповідні зміни в підприємствах, на які підписався користувач.

Бот має доступ до інформації з Єдиного державного реєстру юридичних осіб, фізичних осіб-підприємців та громадських формувань та документів

Єдиного державного реєстру судових рішень. Також бот отримує дані з Міністерства юстиції і за кодом з Єдиного державного реєстру підприємств та організацій України, користувачі можуть отримати повідомлення відповідно до їх запиту.

Маючи OpenDataUABot, користувач може отримати про підприємство інформацію, яка знаходяться у відкритому доступі, про судове рішення, якщо підприємство в ньому фігурує, та історію змін даних. Також бот має два акаунти: простий і професійний. Їх головна відмінність, що звичайний акаунт оновлюється раз на тиждень, а професійний отримує нові дані кожного дня.

Після старту боту, потрібно ввести назву підприємства або код з Єдиного державного реєстру підприємств та організацій України, чи прізвище, ім’я, по-батькові особи, яка цікавить, і отримати необхідні дані про неї. Можливо підписатися на те, щоб при змінах чи при появленні нових постанов відповідно до конкретного суб’єкту, було відправлене відповідне повідомлення до акаунту, та перейти за посилання до відповідного документу, для отримання більш детальної інформації.

Серед переваг боту можна виділити:

- швидка реакція боту на запити;
- надання необхідної інформація через пару простих операцій;
- лаконічне оформлення текстів.
- Значних недоліків не було виявлено, застосунок добре працює відповідно до заданої функціональності.

2.1.5 Telegram-бот "Мій Юрист"

Telegram-бот "Мій Юрист" надає можливість клієнту обрати галузь права, в якій необхідна консультація і поділитися своєю локацією, якщо він хоче знайти юриста поблизу. Виконавши ці дві прості дії, можливо задати питання. Бот, ґрунтуючись на цих даних, звертається із запитом до бази

юристів, обирає найбільш кваліфікованого по даному питанню фахівця і підключає його до чату.

При виникненні питання, бот "Мій Юрист" обирає з бази десять юристів, які знаходяться найближче до клієнта, і відправляє їм питання з кнопкою відповісти. Хто перший натисне на кнопку, той і підключиться до діалогу з клієнтом. Щоб максимально скоротити час очікування консультації, бот кожні дві хвилини вибирає наступну десятку юристів за тими ж критеріями і відправляє їм питання. Цей процес завершується, коли хтось із юристів натисне кнопку відповісти.

Також сервіс передбачає рейтинг юристів. Клієнти можуть залишати відгуки у вигляді оцінок, і при підключенні в чат юриста, можна бачити його середню оцінку. За допомогою бота можна оцінити кваліфікованість юриста при вирішенні проблем, і навіть призначити зустріч для подальшої співпраці за межами чату.

Серед переваг даного телеграм-боту можна виділити такі: користувач може отримати консультацію щодо своєї проблеми від реальної людини, не виходячи зі свого дому або офісу, існує рейтинг юристів відповідно до якого можна робити висновки про компетентність того чи іншого фахівця, бот є сполучною частиною між юристом та людиною, якій знадобилася допомога, охоплюється велика частина права.

Серед недоліків основними є такі: не відомо, який час буде затрачено на пошук відповідного спеціаліста, немає автоматизації для надання консультації для типових запитань.

2.1.6 Підсумковий аналіз розглянутих застосунків

Аналіз вище переглянутих ботів надав змогу виявити позитивні та негативні якості кожного з них. Основною перевагою вищезазначених ботів є те, що для їх підтримки потрібні мінімальні ресурсні витрати системи.

Найбільш близьким за темою є чат-бот "Мій Юрист", але через майже відсутність автоматизації процесу надання консультації, він не є релевантним.

У таблиці 2.1 наведено порівняльний аналіз ботів.

Таблиця 2.1 – Порівняльний аналіз телеграм ботів

Критерій оцінки	Weather bot	GreenzBot	chgk_bot	OpenData UABot	Мій Юрист
Корисність	+	+	+	+	+
Гнучкість (вміння розпізнавати контекст повідомлення)	-	+	-	-	-
Простота використання	+	+	-	+	-
Автоматизація процесу	+	+	+	+	-
Достатність інформації	-	+	-	+	-

В певний момент життя кожному з нас може знадобитися допомога нотаріуса, для того щоб купити або продати майно, переоформити спадщину, посвідчити довіреність, засвідчити підпис на документі, тощо.

Для тих, хто вперше вирішив скористатися послугами нотаріату може виникнути питання, де можна відшукати компетентного нотаріуса, який пакет документів с собою потрібно мати, хто повинен бути присутній при вчинені тої чи іншої нотаріальної дії.

Тому, для роз’яснення певних питань, щоб зберегти час, як нотаріусів, так і людей, які до них вирішили звернутися, було вирішено створити телеграм-бот консультант з нотаріальних питань.

Було поставлено завдання: створити телеграм-бота, який зможе відповідно до запитів користувачів з приводу проведення нотаріальних дій, надати необхідну інформацію, за необхідністю поставити уточнюючі запитання, щоб надати саме ту інформацію яку потрібно, без не потрібної надлишковості, також потрібно передбачити реакцію боту, на питання, які не стосуються предметної області, або за певними причинами не було оброблено, також при можливості, знайти спосіб для тренування боту, відповідно до запитів користувачів, проаналізувати, які можуть виникнути некоректні ситуації, та вирішити їх.

2.2 Вибір додаткового сервісу для реалізації

2.2.1 TensorFlow

TensorFlow – програмне забезпечення з відкритим кодом, компанії Google [9]. вона використовує технологію глибинного навчання і стала популярною досить швидко через те, що нею зацікавилося багато користувачів.

TensorFlow – це певна математична бібліотека для операцій машинного навчання. В ній використовується Google Brain, який став доступний не тільки всередині компанії, а й іншим розробникам. Вона своєчасно оновлюється і широко застосовується в різних IT сферах. TensorFlow може виконувати такі завдання, як розпізнавання місць на фотографіях, надання точних результатів пошуку, точне визначення голосів і поліпшення алгоритмів розпізнавання мови. Завдяки переходу на основу відкритого вихідного коду, був поштовх до початку використання TensorFlow не тільки в межах компанії Google, а й окремими розробниками та компаніями.

Tensorflow надає обчислюване середовище, в якому можна побудувати різні моделі для машинного навчання. В ньому знаходиться набір різних інструментів, які дозволяють будувати моделі на більш високому рівні

абстракції. Також можна використовувати API нижнього рівня та визначити математичні операції для побудови моделі.

API вищого рівня використовують для побудови архітектури системи, нейронних мереж. Tensorflow спрощує процес створення та навчання різного типу нейронних мереж. Також Tensorflow підтримує алгоритм машинного навчання, які більше підходять для різних досліджень, а не практичних завдань.

В Tensorflow є готові бібліотеки, які автоматизують деякі процеси навчання певних аспектів штучного інтелекту, які пов’язані з розпізнаванням та обробкою природної мови.

Переваги TensorFlow:

- швидкі оновлення – після виходу Tensorflow у 2017 році, вона весь час оновлюється і доповнюється новими можливостями;

- крос-платформна програма – Tensorflow використовує ресурси системи до її відповідних можливостей, краще працює на комп’ютерах з графічними процесорами, але і при обмежених ресурсах можна створювати застосунки з використанням технологій Tensorflow;

- може працювати на різних операційних системах Windows, Linux, Android, macOs;

- швидке виконання – TensorFlow підтримує функцію швидкого виконання, яка надає можливість спростити процес та налагодити його, також вона полегшує процес створення, перевірки, змін та ін. Також для цих процесів можна використовувати потік керування Python;

- широкий набір інструментів – TensorFlow налічує велику кількість різних інструментів та бібліотек для створення відповідних застосунків, які покращують ефективність праці, та автоматизують певні процеси аналізу, зміни, налагодження програмного забезпечення.

2.2.2 Dialogflow

У вересні 2016 року Google купив компанію API.AI [10], яка надає інструменти для розробників, що створюють програми для віртуального помічника Google Assistant, в якому використовуються різні методи для розпізнавання природної мови. Завдяки реалізації алгоритмів машинного навчання використаним при розробці API, в процесі роботи механізми пов’язані з розпізнаванням природної мови самовдосконалюються.

Dialogflow надає розробникам застосунків можливість інтегрувати механізми взаємодії між людиною та машиною, які основані на діалозі за допомогою природної мови [11]. Завдяки цьому розробники можуть покращити своє програмне забезпечення, реалізувавши доступ до певних аспектів застосунку через голосові та текстові обміни, основані за допомогою штучного інтелекту.

Одна з головних переваг Dialogflow, що він є абсолютно безкоштовним, та доступним для будь-якого користувача. Dialogflow являє собою одну з найкращих платформ, які використовуються в світі для створення діалогового інтерфейсу. Сервіс використовується для обробки природної мови в різного роду застосунках, пристроях інтернет речей, чат-ботах. Dialogflow може отримувати контекст повідомлення і передавати оброблені дані до програми, що може спонукати її на виклик певних методів відповідно до отриманих даних.

Dialogflow підтримує підключення webhook, щоб зв’язати Dialogflow зі своїм застосунком, який, при його визові, буде реалізовувати виконання бізнес-логіки, викликати зовнішні API, звертатися до сховищ даних. Отримуючи повідомлення у вигляді тексту на природній мові, Dialogflow звіряє запит з заготовленими шаблонами, та вибирає найбільш підходящий. При цьому Dialogflow ґрунтується на інформації, що міститься в шаблоні (приклади, сутності, контекст, параметри) і машинному навчанні.

Dialogflow формує відповідь відповідно до шаблону і повертає дані у вигляді об'єкта JSON.

Основні положення роботи з Dialogflow [12]:

- створення та налагодження агентів;
- створення запитів (Intents);
- визначення контексту;
- створення сутностей (Entities);
- тренування агентів;
- підключення лінгвістичних модулів;
- інтеграція з месенджерами і іншими застосунками;
- підключення лінгвістичних корпусів;
- обробка json-даних на сервері та підключення агентів;
- наявний власний SDK.

Завдяки вищезазначеним функціям платформи можна досить швидко створити та інтегрувати до відповідного ресурсу чат-бота і вдосконалювати його потім в процесі спілкування з користувачами. Функціональні можливості агентів сервісу надають можливості для створення унікальних алгоритмів для ведення діалогу на природній мові користувача, а також певним чином аналізувати запити і реагувати на них.

Завдяки даним, зібраним на протязі існування компанії Google було сформовано бази знань, які допомагають в аналізі контексту повідомлень та правильної її обробки. В бази знань включені різного роду аналітичні дані, маршрути авіаліній, дані с транспортних служб, дані з пізнавальних ресурсів та інших. В основі реалізації методів спілкуванні з користувачами лежать методи машинного навчання на прикладах, наданих розробниками, а обробка запитів користувачів надає можливість постійно покращувати програмний продукт.

Dialogflow може взаємодіяти та інтегруватися з різними соціальними мережами (Facebook, Twitter), месенджерами (Telegram, Skype, Viber) та

іншими. Для різних платформ можна налаштувати алгоритм ведення діалогу таким чином, щоб результат одного і того самого запиту був різним для кожної з них в контексті одного агента. Dialogflow SDK для підтримки кількох платформ і мов програмування, включаючи iOS, Apple Watch, Mac OS X, Android, HTML, Cordova, JavaScript, Python, C #, Xamarin і Unity. Підтримує понад 16 мов, включаючи російську та українську. Щодня час неперервної роботи сервісу досягає позначки майже 100% при обробці більше десяти мільйонів запитів користувачів. Основним видом даних, які оброблюються в Dialogflow є текст. За необхідністю можна використати Google Speech API для підтримки голосових команд або підключити власні бібліотеки для обробки голосових повідомлень та перетворені їх в текст. На виході алгоритму надається відповідь текстового формату. Проте можна налаштувати повернення результату у вигляді іншого об’єкту, наприклад картинки.

Dialogflow може запропонувати взаємодію з користувачем на природній мові. Надавши кілька прикладів того, що користувач може сказати, Dialogflow створить модель, яка зможе розпізнавати, які дії запускати і які дані отримувати зі сказаного або написаного користувачем. Саме це і створює ефект розмови зі звичайною людиною.

Dialogflow надає можливість розгортати створені застосунки на 32 різних платформах, наприклад, таких як Telegram, Facebook, та на інших популярних платформах схожого типу.

Dialogflow зараз працює приблизно на 23 різних мовах. Підтримує можливість тренування на англійській, російській мові, але поки що не підтримує тренування боту на українській мові.

Вбудований редактор коду дозволяє кодувати, тестувати і реалізовувати ці дії безпосередньо в консолі Dialogflow. Голосове управління з розпізнаванням мови Dialogflow дозволяє застосункам з підключеними розмовними можливостями відповідати на голосові команди користувача. Ця

функція доступна в рамках одного виклику API, що поєднує розпізнавання мови з розумінням природної мови. Dialogflow Enterprise Edition вже використовується багатьма компаніями. Наприклад, на ньому побудовані ігровий асистент Sam від Ubisoft і бот для замовлення піци від Domino's Pizza.

Нижче представлена зведена таблиця порівняння, між бібліотекою TensorFlow і платформою Dialogflow

Таблиця 2.2 – Порівння сервісів Dialogflow та Tensorflow

	DIALOGFLOW	TENSORFLOW
Опис сервісу	Надає можливості створювати застосунки з використанням модулів природної мови	Надає основний інструментарній для машинного навчання
Можливість підключення до проектів	Надає можливості підключення сторонніх застосунків та сервіс, та використання Dialogflow агентів для побудови діалогу на природній мові	Може використовуватися на різних мовах програмування для підключення в створюваний проект
Підтримувані операційні системи	Windows, Android, Linux, macOs	Windows, Android, Linux, macOs, iOS
Основна функціональ-ність	Створення агентів; Створення дерева діалогу з користувачем; Заповнення сутностей; Використання API інтерфейсів для підключення у проекти	Створення алгоритмів машинного навчання; Використання власноруч створених алгоритмів; Підключення до проектів через API інтерфейси

2.2.3 Підсумковий аналіз

TensorFlow і DialogFlow займають провідні позиції в роботі з нейронними мережами. Сьогодні вони вважаються кращими в сфері розпізнавання природної мови. Кожен гарний по своєму, для цього була складена таблиця порівняння бібліотеки TF і платформою DF. Щоб досягти поставлених цілей, в якості основного інструменту буде використовуватися DF, оскільки він володіє необхідними засобами для досягнення поставлених завдань.

Було розглянуто декілька сервісів для створення телеграм-ботів, але основна функціональність, яку вони надавали, полягала в тому, що є можливість задавати тільки відповіді на певні запитання користувача, які прописані на даному сервісі.

Для створення бота, який зможе з мінімальною участю людини у процесі консультування, допомогти клієнтам з пошуком інформації у нотаріальній сфері, щоб заощадити їх час, було вирішено обрати один аспект права, а саме – нотаріат, і автоматизувати процес надання першочергової консультації з нотаріальних питань, було обрано тему: телеграм-бот консультант з нотаріальних питань.

2.3 Висновки до розділу

Під час огляду існуючих рішень було розглянуто п’ять різних ботів, було проаналізовано їх функціональність, основні переваги та недоліки, також було розглянуто два сервіси для того щоб зробити бота більш інтелектуальним, щоб він вмів розпізнавати контекст повідомлення.

Було виявлено, що боти є досить корисними застосунками, вони досить просто інтегруються в месенджер дозволяють автоматизувати ті чи інші

процеси, але більшість з них не вміє розпізнавати контекст деяких повідомлень.

В процесі пошуку системи для визначення контексту повідомлень було розглянуто дві системи компанії Google.

В обох сервісах для тих чи інших завдань використовуються нейронні мережі, та машинне навчання, тому створюванні застосунки можна навчати, і вони самі за певних умов можуть навчатися. В Dialogflow агенти завдяки цій технології можуть самі створювати алгоритми розпізнавання природної мови, та процес отримання контексту запиту.

Було з’ясовано, що для розробки чат-боту для месенджеру Telegram краще використати сервіс Dialogflow, оскільки він більше спрямований на роботу з природною мовою, має зручний інтерфейс, дозволяє створюваним застосункам сприймати контекст повідомлень. Також Dialogflow легко інтегрується та підтримується на різних платформах, і працює з понад 20 різними мовами, в тому числі і з українською. Агентів створюваних в цьому сервісі можна в певній мірі назвати розумними, оскільки їх можна навчати, а в подальшому вони навчаються автоматично, лише з мінімальною участю людини.

3 МАТЕМАТИЧНИЙ АПАРАТ

3.1 Математичний апарат Dialogflow.

В Dialogflow використовуються алгоритми машинного навчання для того, щоб агенти змогли зрозуміти, що має на увазі користувач, зіставити з тими варіантами, які знаходяться в базі агенту, і яким чином потрібно обробити повідомлення користувача та видати структуровані дані.

Агент навчається процесом роз'яснення фраз, які йому надаються для обробки, та використовує мовні моделі, які вбудовані в Dialogflow. Далі він аналізує ці данні, та підбирає відповідний алгоритм для того, як саме потрібно опрацювати те, чи інше повідомлення.

Dialogflow оновлює алгоритм машинного навчання створеного агента щоразу, як вносяться зміни до його бази знань, або певним чином змінюються чи оновлюється його сутність.

Dialogflow має два режими машинного навчання: гібридний режим та тільки машинне навчання.

Режим збігу визначає алгоритми, які слід використовувати для всіх агентів, у яких включено машинне навчання. Можна вибрати один із наступних режимів:

гібрид – цей режим спочатку намагається відповідати граматиці на основі правил. Якщо відповідність не виконано, він перемикається на режим машинного навчання.

Тільки машинне навчання – режим використовує лише ті алгоритми, та ті данні, які використовувалися і були підтвердженні в процесі машинного навчання.

Щоб відфільтрувати помилкові позитивні результати та отримати різноманітність відповідних входів для агента, можна налаштувати поріг класифікації машинного навчання.

Збіги намірів мають значення довіри, яке варіюється від 0,0 (повністю невизначене) до 1,0 (повністю певне). Для кращого визначення рівня довіри потрібно визначити негативні варіанти, які будуть відповідати рівню довіри, менше за поріг класифікації ML.

3.1.1 Автоматична корекція орфографії

При включеному режимі автокорекції орфографії агент буде автоматично виправляти орфографічні чи граматичні помилки, та шукати відповідний запит до виправленого варіанту. Наприклад, якщо користувач вводить "Який согодні день", буде оброблено так, наче користувач ввів "Який сьогодні день". Це також стосується відповідності як системних, так і розроблюваних об'єктів.

Якщо значення оригінального і виправленого повідомлення користувача збігаються з різними моделями розпізнавання, то агент вибере ту модель для обробки запиту, яка відповідає оригінальному повідомленню користувача.

Автокорекція правопису доступна для всіх підтримуваних мов, використовуваних Dialogflow.

Можливі збої та найкращі варіанти використання.

Автокорекцію правопису не рекомендовано використовувати для агентів, які використовують входи ASR (автоматичного розпізнавання мовлення), бо цей метод не може виправити помилки ASR.

В деяких випадках виправлене повідомлення може відповідати не правильній моделі розпізнавання. Для уникнення невідповідностей, можна або додати негативні приклади, або скорегувати відповідь в режимі тренування агенту.

Коригування орфографії трохи збільшує час відгуку агента.

Виправлення правопису проводиться на загальних запитах користувачів. Якщо агент визначається з використанням специфічного для домену жаргону, виправлення можуть бути небажаними.

3.2 Методи машинного навчання

Розрізняють декілька видів машинного навчання, основними з яких є [14]: дедуктивне навчання та навчання по прецедентах (індуктивне навчання). Дедуктивне навчання зазвичай використовують для використання в різного роду експертних системах. Зазвичай, кажучи про машинне навчання, мають на увазі саме індуктивне навчання. Оскільки в наш час відбувається перехід на максимальну можливу автоматизацію різного роду процесів, шляхом створення певної машини для відповідної задачі, і вони потребують навчання, то машинне навчання зараз дуже широко використовується і є досить популярним, чого не можна сказати про дедуктивне навчання. Оскільки в основі дедуктивного методу лежать бази знань, які важко узгодити з реляційною моделлю даних, бази знань експертних систем дуже важко наповнювати, через низьку ефективність різного роду СУБД для управління базами знань системи.

Розрізняють три основних типи машинного навчання: контрольоване навчання, або навчання з учителем (supervised learning), неконтрольоване навчання (unsupervised learning), або навчання без учителя, і навчання з підкріпленням (reinforcement learning).

Крім названих, розробляються і інші методи навчання: активне, багатозадачне, різноманітне, трансферне тощо. Особливо успішно розвивається в останні роки «глибинне навчання», при використанні якого можуть успішно поєднуватися алгоритми навчання з вчителем і без вчителя.

Далі розглянемо основні три типи навчання, їх основні переваги та недоліки.

3.2.1 Контрольоване навчання

Метод контрольованого навчання доцільно використовувати при наявності великих обсягів даних, наприклад є декілька тисяч м’ячів для різних видів спорту, і потрібно їх якимось чином класифікувати, це м’яч для футболу, баскетболу, волейболу чи ще для чогось. Потрібно створити алгоритм, який надав би змогу машині класифікувати об’єкти за певними критеріями, навіть ті, які вона раніше не бачила, щоб вона визначила до якого роду спорту потрібно віднести той чи інший м’яч. У ролі «вчителя» в даному випадку виступає людина, яка заздалегідь проставила маркери. Машина сама вибирає ознаки, за якими вона сортує м’ячі до різних категорій. Таким чином, машина сама знаходить алгоритм, за яким вона буде діяти. В подальшому можливо дещо змінити його і використовувати для рішення схожого роду задач.

3.2.2 Неконтрольоване навчання

При неконтрольованому навчанні одним з головних завдань машини є пошук зв’язку між різними даними, які поки не класифіковані, вона намагається виявити певні закономірності, або зіставити з деякими шаблонами, певним чином структурувати їх, класифікувати дані за певними ознаками. Цей метод досить необхідний у сферах пов’язаних зі злочинами, коли потрібно пов’язати між собою декілька злочинів, знайти в них щось спільне, щоб вийти на слід злочинця, і схопити його. Також цей метод використовується для аналізу наших переходів по різним посиланням в Інтернеті, компанією Google, щоб потім на основі цих даних пропонувати нам певні можливі варіанти пошуку, які нас можуть зацікавити.

3.2.3 Навчання з підкріпленням

Навчання з підкріплення є окремим випадком контрольованого навчання в основі якого є аналіз навколишнього середовища, яке виступає в ролі вчителя. Робот при даному типі навчання начебто навчається на власних помилках, тому що в нього не закладено початкових даних про його оточення.

При такому методі навчання оточення машини виступає в ролі вчителя, і він реагує на будь-які дії машини, і надає зворотну інформацію про результати її дій. Отримавши ці данні, робот їх аналізує і редагує свою поведінку відповідно до необхідного результату. Тобто між середовищем і роботом утворюється система із зворотнім зв’язком.

Зазвичай навчання з підкріпленням використовується для вирішення складних завдань, розв’язок яких не вдається отримати іншими методами. Досить часто саме цей метод використовується в автоматизованих системах навігації. Автопілот електрокарів та деяких літаків було створено саме з використанням цього методу. Також таким методом навчають різного роду ботів, наприклад ігрових ботів, щоб створити штучний інтелект, який зможе вигравати навіть у професіоналів в шахи, карткові ігри, нарди, тощо.

3.3 Огляд методів векторного представлення текстів

3.3.1 Розвиток методів векторного представлення текстів

В 80-х роках минулого століття Герард Салтон в своїх роботах приніс у світ векторну модель, яка стала одним із варіантів заміни лексичного безконтекстного індексування. В цій моделі кожному документу ставиться у відповідність вектор в лексичному просторі відповідно до частотного спектра слів. В цій моделі розглядають частотну модель запиту, як певний вектор в тому ж лексичному просторі, і визначають ступінь відповідності (відстань, або кут між векторами), в результаті знаходять найбільш близькі документи.

Для швидшої дії алгоритму та зменшення використовуваних ресурсів, а також з розвитком методів почали відкидати елементи алфавіту, а також ті, які дуже часто використовуються, відповідно до певного словника.

Однією з головних переваг векторної моделі є визначення близькості розташування текстів в векторному просторі за допомогою пошуку послідовного розміщення текстів, в залежності від їх значимості за певною відповідністю.

Проте є і певні недоліки, наприклад, якщо один з документів, який зіставляється буде містити малу кількість слів, а інший набагато більше за перший, то алгоритм може не вірно розпізнати взаємо розташування їх в векторному просторі.

Тому почали шукати більш релевантну модель, яка змогла б більш вірно оцінити відповідність одного документу до іншого. І на початку 90-х років 20 століття було запропоновано новий метод прихованого семантичного індексування. В його основі покладений алгоритм сингулярного розкладу матриці.

Новий метод показав набагато кращі результати на відміну від лексичного методу, але через свою складність час, затрачений на пошук відповідностей, був більший, ніж у методі, в основі якого лежала булева алгебра, особливо це було помітно при аналізі великої кількості документів. Одна з найбільш релевантних моделей була створена в Берклі Майклом Беррі і Тодом Летч.

3.3.2. Векторна модель представлення текстової інформації

В векторній моделі текст розглядається як певна структура, тобто не просто ні чому не підпорядкована інформація, а як множина певних, пов’язаних між собою речень, які знаходяться в певній множині абзаців, які в свою чергу являють собою складову параграфів, і це входить в множину

книги (документу). Сформовану структуру можна вважати наглядним прикладом суті, головної ідеї, певним чином можна назвати алгоритмом (або схемою алгоритму) досягнення мети твору автора, через досягнення певних цілей різної ваги, які визначені певними елементами множин.

Далі наведено приклади математичної інтерпретації векторної моделі деякого закінченого елемента тексту, що складається, наприклад, з розділів, що містять абзаци, які, в свою чергу, складаються з пропозицій, тобто:

G = {G1, G2, ..., Gi, ..., Gn}

Vg = {Vg1, Vg2, ... , Vgi ,, ..., Vgn},

де G - множина глав; Gi - i-а глава, i = 1 ... n; Vg – множина векторів цілей глави; Vgi - вектор мети i-ої глави.

В свою чергу

Ai = {Ai1, Ai2, ..., Aij, ..., Am}

Vai = {Vai1, Vai2, ..., Vaij, ..., Vaim},

де Ai - множина абзаців i-ї глави; Aij - j-ий абзац i-ї глави, j = 1 ... m; Vai - множина векторів цілей абзаців; Vaij - вектор мети j-го абзацу i-ї глави.

Математична інтерпретація векторної моделі для пропозицій виражається у вигляді:

Pij = {Pij1, Pij2, ..., Pijh, ..., Pk}

Vpij = {Vpij1, Vpij2, ..., Vpijh, ..., Vpijk},

де Pij - множина пропозицій i-ої глави j-го абзацу; Pijh - h-е пропозицію i-ї глави j-го абзацу, h = 1 ... k; Vpij - множина векторів цілей пропозицій i-ї глави j-го абзацу; Vpijh - вектор мети h пропозиції i-ї глави j-го абзацу.

З представленого опису видно, що кожному елементу (фрагменту) тексту ставиться у відповідність деякий вектор мети.

Як відомо, смисловим і граматичним центром пропозиції зазвичай є присудок, виражене дієсловом. При певних умовах і імені групи (іменники з залежними словами або без них) можуть виступати в якості закінченої

пропозиції. Прикладами таких пропозицій, званих іменними, або номінативними, є, наприклад:

Тридцять друге. Вівторок. День.

Крім іменних пропозицій можна розглядати і неповні, які утворюються з повних, шляхом певних скорочень, наприклад:

Телефон! (Замість: Дзвонить телефон!).

Як вже зазначалося вище, кожне речення несе в собі певний сенс, який закладається автором, і забезпечує просування до кінцевої мети, як основної ідеї в контексті системи цілей смислової групи пропозицій, абзацу і т.д. У загальному випадку кожне речення має відповідний вектор мети. Таким чином, текст можна визначити як структуру взаємопов'язаних понять, що забезпечує просування до кінцевої мети, яка виражається авторською ідеєю.

Запропонована модель вектора мети може бути представлена у вигляді трьох компонент:

Vbegin – представлення початкова мета, що виражається через початковий вектор із заданими координатами;

Vend – кінцева мета, що виражається через кінцевий вектор із заданими координатами;

Z – вид зв'язку між початковим і кінцевим вектором.

У якості координат вектора можуть виступати окремі слова, поняття, іменні групи, окремі пропозиції, смислові групи пропозицій, абзаци і т.д.

Оскільки є три складові, які визначають вектор, то відповідно для подальшого аналізу виділяють наступні типи векторів:

- простий вектор: = () або V = ();
- нульовий вектор: = (Ø);
- повний вектор: = (,) зі зв'язком Z;
- порожній вектор: = (,) без зв'язку Z;
- лівий вектор: = ();
- правий вектор: = ().

У свою чергу вектори можуть складатися з декількох векторів, як окремих самостійних частин – координат, що належать поточному вектору.

Кожна координата має свої атрибути. Атрибутами можуть бути тимчасові або просторові характеристики координати.

Склад координат вектора визначається складністю побудови пропозиції. У загальному випадку підпорядкованість окремих частин пропозиції може бути усунена шляхом нормалізації.

Розглянемо застосування описаної вище векторної моделі на конкретному прикладі.

Проаналізуємо таку пропозицію.

У всі часи люди стикаються з одними і тими ж проблемами економіки.

Дана пропозиція може бути представлена у векторній формі: Vp (x1; y1) зі зв'язком виду z1, або в спрощеній формі Vp (x1; y1) (z1),

де координата x1 = {люди};

координата y1 = {проблеми економіки};

зв'язок виду z1 = (стикаються).

При цьому атрибутами координати x1 є atrx = (в усі часи), а атрибутами координати y1 є atry = (одні і ті ж).

На основі векторного уявлення можуть бути вирішені деякі проблеми обробки текстової інформації, зокрема:

- скорочення обсягу вихідної інформації для виконання процедур аналізу тексту і формування систем і баз знань;

- синтез тексту з використанням інформації, що збережена у базу знань.

3.3.3 Застосування технології векторного уявлення текстової інформації

Розглянута вище векторна модель представлення текстової інформації може бути використана при аналізі і синтезі тексту.

Нехай існує три вектора Vp1 (x1, y1) (z1); Vp2 (x2, y2) (z2); Vp3 (x3, y3) (z3) і їх проекція на площину x, atr. У загальному випадку проекцію можна здійснити на різні площини: (x, y), (x, atr), (y, atr).

Атрибути можуть мати тимчасові, просторові і інші вимірювані характеристики.

Координати xi визначають початкові координати вектора. Координати yi визначають кінцеві координати вектора.

Виходячи з попереднього опису вектор V визначає кінцеву мету даної одиниці тексту і має структуру вектора мети.

Розглянемо використання описаної моделі на прикладі.

Нехай задані три вектори:

Vp1 (x1, y1) (z1); Vp2 (x2, y2) (z2); Vp3 (x3, y3) (z3),

де y1 = x2, y2 = x3.

Фактично сукупність цих трьох векторів означає якийсь результуючий вектор V (x1, y3) (z '), відповідний спільній меті вихідних векторів

Vp1 (x1, y1) (z1); Vp2 (x1, y2) (z2); Vp3 (x3, y3) (z3),

де y2 = x3.

Іншим застосуванням векторної моделі є можливість реалізації синтезу текстової інформації.

Припустимо, що вирішується завдання, пов'язане з розкриттям поняття x1. В цьому випадку, вектор мети для опису певних процесів або явищ може бути представлений вектором Vp0 (x1, yi) (zi), де координата yi і вид зв'язків zi, визначаються в процесі конструювання підцілей.

Припустимо, в базі знань поняття x1 визначено на безлічі онтологій через вектор Vp0 (x1, y1) (z1). У свою чергу підвектор y1 має координати Vp1 (x2, y2) (z2). Підвектор y2 також може мати свої координати. Таким чином, отримуємо ланцюжок векторів для розкриття поняття x1. Механізм розгортання вектора для опису процесів і явищ може бути двояким: або на основі збереженого початкового тексту, шляхом вилучення вже сформованих

фраз, або шляхом генерації нових пропозицій на основі алгоритму побудови речень на природній мові.

На основі запропонованої моделі розроблено технологію обробки текстової інформації на основі векторної моделі тексту.

На основі запропонованої моделі розроблено технологію обробки текстової інформації на основі векторної моделі тексту.

Виходячи з зазначеного вище технології векторного представлення текстової інформації і автоматизація цих процесів можуть бути застосовані для:

- створення професійних систем і баз знань;
- підтримки професійної діяльності працівників різних галузей;
- підвищення рівня компетенції фахівців за рахунок отримання можливості швидкого аналізу та подання в зручній формі результатів цього аналізу;
- проведення синтезу текстових документів з різним ступенем узагальнення інформації;
- автоматизації процесів формування системи онтологій в тій чи іншій професійній області;
- проведення спрямованого пошуку і фільтрації текстових документів;
- автоматичного реферування текстів документів.

3.4 Висновки до розділу

Оскільки чат-боти певного роду – машини, в основу яких можна закласти все, що забажає розробник, то вони певним чином досить схожі на нейроні мережі. І для навчання ботів нерідко використовують алгоритми машинного навчання, основними методами якого є контрольоване навчання, навчання з підкріпленням та неконтрольоване навчання. Основною

складністю чат-ботів є навчити їх сприймати природню мову, розпізнавання контексту повідомлень.

В процесі пошуку оптимального рішення цього питання, було обрано сервіс Dialogflow, в основу якого закладено саме машинне навчання.

Агент Dialogflow навчається процесом роз’яснення фраз, які йому надані для обробки, та використовує мовні моделі, які вбудовані в Dialogflow. Далі він аналізує ці данні, та підбирає відповідний алгоритм для того як саме потрібно опрацювати те чи інше повідомлення.

Також було розглянуто методи векторного представлення текстів, в основі якого є представлення текстів в деякому векторному просторі, і в подальшому для їх порівняння використовувати відстані між векторами та кути між ними, для знаходження відсотку збігу текстів. В процесі розвитку було запропоновано порівнювати не самі тексти, а мету тексту, створення векторів початкової та кінцевої мети. Кожне речення має певний сенс закладений автором, і завдяки ним поступово відбувається перехід від початкової до кінцевої мети.

4 ОПИС ВЛАСНОЇ РЕАЛІЗАЦІЇ

4.1 Завдання дипломного проекту

Відповідно до завдання дипломного проекту та відповідно до розглянутих підходів було вирішено за допомогою засобів .Net framework на мові програмування C# з підключенням сервісу Dialogflow[15] створити телеграм-бота, який буде надавати необхідну інформацію, відповідно до запиту, щодо надання консультативних послуг з питань нотаріату користувачеві.

Основні функції, які було розроблено у процесі виконання дипломного проекту:

- отримання запита користувача;
- обробка запита користувача;
- за необхідністю уточнення параметрів запиту;
- створення можливого варіанту вибору та передача його користувачу;
- відповідно до обробленого запиту видання необхідної інформації користувачеві;
- передбачення реакції боту на запити, які не стосуються предметної області ;
- можливість тренування боту.
- Також за необхідністю в майбутньому можливо реалізувати такі функції:
- створення бази даних нотаріусів;
- отримання координат місцезнаходження користувача;
- запропонування декількох варіантів звернення до фахівця з нотаріальних питань відповідно до місцезнаходження користувача та його запиту.

В процесі використання телеграм-боту користувач матиме змогу задати своє запитання на природній мові, або вибрати з запропонованих варіантів,

далі це повідомлення буде передаватися у сервіс Dialogflow за допомогою якого можна отримати необхідний контекст повідомлення, потім ця інформацію буде передаватися до програми, в якій буде визначатися, що робити з цим повідомленням, відразу надавати інформацію, або потрібно задати певні уточнюючі питання. Процес уточнення може повторюватися, доки не буде отримано необхідну інформацію для видачі максимально повної відповіді для користувача. Після завершення процесу отримання запиту користувача та його аналізування, користувачу надсилається відповідь з його питання.

Якщо користувач ввів запит, який не стосується предметної області, то є два варіанти розвитку. Якщо запит не стосується теми, то буде виводитися повідомлення про те, щоб користувач вводив питання тільки стосовно нотаріату. Якщо ж користувач ввів запит, який стосується нотаріату, але бот за якоїсь причини або не зрозумів його, або не зміг знайти необхідну інформацію щодо цього запиту, можна перейти у процес тренування бота відповідно до запитів користувачів. Саме сервіс Dialogflow надає таку можливість - отримати запити користувачів і проаналізувавши реакцію боту, надати йому роз'яснення чи правильно він зрозумів той, чи інший запит, і у разі необхідності, змінити варіант розвитку діалогу для даного запиту ботом.

При необхідності, можливо створити базу даних, яка буде містити в собі місцезнаходження та контакти нотаріусів, як державних, так і приватних, їх спеціалізацію та рейтинг відповідно до наданих ними нотаріальних послуг. Тоді можна буде додати функцію, за згодою користувача, отримавши його геолокацію, надати йому список найближчих нотаріусів, які допоможуть йому вирішити його питання.

Для зв'язку програми з телеграм-ботом використовується Telegram.Bot package, який надає інструменти зв'язку з ботом, отримання повідомлень з серверів Telegram. А для роботи з Dialogflow використовується ApiAiSDK,

який дозволяє зв’язатися з сервісом Dialogflow, для відправки та отримання повідомлень від нього.

4.2 Реалізація методу розпізнавання контексту повідомлень

Для реалізація методів розпізнавання контексту повідомлень користувачів було обрано використання сервісу Dialogflow і підключення необхідних компонент до боту.

Спочатку потрібно зареєструватися на сайті Dialogflow, зробити це досить просто використавши електронну пошту gmail.

Далі у консолі потрібно створити першого агента, пов’язавши його з назвою області використання. Після створення агенту можна отримати токен, який потім потрібно використати у підключенні сервісу до програми.

Також потрібно вказати мову на якій буде працювати агент. Необхідно зазначити, що хоча у списку знаходиться приблизно 23 варіанти вибору мов, в тому числі і українська, проте функціональні можливості для них різні. Найкраще мовні модулі реалізовані для англійської мови, для української мови нажаль недоступний модуль тренування боту, але якщо підключити іншу мову як основну, і використати українську, як додаткову, то можна розблокувати режим тренування, і інші недоступні для української мови функції.

Основою роботи агенту є Intents: вони дозволяють створити базу знань для відповідного застосунку, задавши назву, варіанти вводу та що потрібно надати на вихід, у текстовому форматі.

Intents – це певна сукупність варіантів ведення діалогу, кожний елемент якої відповідає за певну дію.

За замовчуванням існує два Intents обробка непередбачених варіантів, та привітання. Щоб створити власні сутності потрібно перейти на вкладку Intents

та заповнити відповідні поля вказавши назву сутності, варіанти запиту користувача, та варіанти відповідей.

Відповідно до структури нотаріальних дій реалізованих в проекті (пункт 4.3) було створено відповідні сутності.

4.3 Структура нотаріальних дій, реалізованих в проекті

Документи, що надаються фізичною особою нотаріусу для укладення правочинів:

Громадянин України:

- паспорт громадянина України;
- паспорт громадянина України для виїзду за кордон (з відміткою про взяття фізичної особи на консульський облік);
- довідка про присвоєння ідентифікаційного номеру;
- довіреність на право укладання правочину, якщо договір укладається на підставі довіреності (представник надає паспорт та довідку про присвоєння ідентифікаційного номеру).

Іноземець або особа без громадянства:

- національний паспорт іноземця або документ, що його замінює (з перекладом на українську мову);
- посвідка на постійне проживання в Україні або посвідка на тимчасове проживання в Україні;
- довідка про присвоєння ідентифікаційного номеру;
- довіреність на право укладання правочину, якщо договір укладається на підставі довіреності (представник надає паспорт та довідку про присвоєння ідентифікаційного номеру).

Особа віком до 14 років:

- свідоцтво про народження;
- довідка про присвоєння ідентифікаційного номеру;

- дозвіл органів опіки та піклування на відчуження;

- нотаріальна заява про згоду одного з батьків на вчинення іншим із батьків правочинів щодо нерухомого майна малолітньої дитини.

Особа віком від 14 років до 18 років:

- паспорт громадянина України;

- свідоцтво про народження;

- довідка про присвоєння ідентифікаційного номеру;

- дозвіл органів опіки та піклування на відчуження ;

- нотаріальна заява про згоду батьків на посвідчення правочинів від імені неповнолітніх.

Документи на нерухоме майно (квартиру, житловий будинок, нежитлове приміщення, гараж, машиномісце, садовий будинок):

- правовстановлюючий документ (договір купівлі-продажу, договір міни, договір дарування, свідоцтво про право на спадщину за законом, свідоцтво про право на спадщину за заповітом, свідоцтво про право власності на нерухоме майно, рішення суду, що набрало законної сили тощо);

- витяг з Державного реєстру речових прав про реєстрацію права власності на нерухоме майно, якщо право власності на нерухомість зареєстровано з 01.01.2013 р.;

- витяг про реєстрацію в Спадковому реєстрі (якщо правовстановлюючий документ свідоцтво про право на спадщину);

- витяг з Державного реєстру правочинів, якщо правочин (договір купівлі-продажу) був вчинений у період (02.08.2004 р. – 31.12.2012 р.);

- реєстраційне свідоцтво або реєстраційний напис бюро технічної інвентаризації або Витяг з Електронного реєстру прав власності на нерухоме майно (до 01.01.2013 р.);

- витяг з Реєстру територіальної громади про зареєстрованих осіб у житловому приміщенні (береться нотаріусом при вчиненні нотаріальної дії,

або надається на запит нотаріуса, якщо нерухомість знаходиться в іншому нотаріальному окрузі);

- технічний паспорт;
- звіт про оцінку майна;
- нотаріальна заява про згоду на відчуження нерухомого майна, якщо це спільна сумісна власність.

Документи на земельну ділянку:

- правовстановлюючий документ (договір купівлі-продажу земельної ділянки, договір міни, договір дарування, свідоцтво про право на спадщину за законом, свідоцтво про право на спадщину за заповітом, державний акт на право власності на земельну ділянку, рішення суду тощо);
- витяг з Державного реєстру речових прав про реєстрацію права власності на нерухоме майно, якщо право власності на нерухомість зареєстровано з 01.01.2013;
- витяг про реєстрацію в Спадковому реєстрі (якщо правовстановлюючий документ свідоцтво про право на спадщину);
- витяг з Державного земельного кадастру щодо наявних обмежень/обтяжень на земельну ділянку;
- звіт про експертно-грошову оцінку земельної ділянки;
- довідка про відсутність забудов на земельній ділянці (якщо цільове призначення земельної ділянки: для будівництва, індивідуального садівництва);
- нотаріальна заява про згоду на відчуження нерухомого майна, якщо це спільна сумісна власність.

Документи на транспортні засоби, інші самохідні машини і механізми:

- свідоцтво про реєстрацію транспортних засобів;
- звіт про експертну оцінку транспортного засобу;
- нотаріальна заява про згоду на відчуження рухомого майна, якщо це спільна сумісна власність.

Документи на товари в обороті:

- контракти на поставку;
- специфікацій товару;
- довідка (договори оренди) про місце зберігання товару;
- балансова довідка.

Відповідно до Сімейного кодексу України, нерухоме майно є особистою приватною власністю одного з подружжя, яке набуте до реєстрації шлюбу, за час шлюбу, але на умовах, передбачених шлюбним або іншим договором, укладеним між подружжям, за договором дарування, або в порядку спадкування, та за кошти, які належали одному з подружжя особисто, житло, набуте особою за час шлюбу внаслідок його приватизації відповідно до Закону України "Про приватизацію державного житлового фонду", земельна ділянка, набута особою за час шлюбу внаслідок приватизації земельної ділянки, що перебувала у її, його користуванні, або одержана внаслідок приватизації земельних ділянок державних і комунальних сільськогосподарських підприємств, установ та організацій, або одержана із земель державної і комунальної власності в межах норм безоплатної приватизації, визначених Земельним кодексом України.

- Договори відчуження нерухомого майна
    1) купівлі-продажу
    2) міни
    3) дарування
        - квартири
        - житлового будинку
        - садового будинку
        - гаражу
        - машиномісця
        - нежитлового приміщення
        - інше

4) довічного утримання (догляду)

5) договори пожертви

6) договори ренти

7) інші договори відчуження нерухомого майна

- Договори відчуження земельних ділянок, а також земельних часток (паїв)

1) купівлі-продажу

2) міни

3) дарування

- Договори відчуження транспортних засобів, інших самохідних машин і механізмів
- Інші договори відчуження рухомого майна
- Договори іпотеки

1) квартир

2) житлових будинків

3) земельних ділянок

4) інші договори іпотеки

- Договори застави

1) транспортних засобів

2) інші договори застави

- Шлюбні договори
- Договори найму (оренди)

1) квартир та житлових будинків;

2) нежитлової будівлі або іншої капітальної споруди (їх окремої частини);

3) земельних ділянок;

4) транспортних засобів.

- Договори управління нерухомим майном
- Договори про зміну черговості одержання права на спадкування

- Договори про зміну часток у спадщині
- Інші договори
- Посвідчення заповітів
- Відкриття спадкових справ
- Вжито заходів до охорони спадкового майна
- Видання свідоцтва виконавцям заповіту
- Видання свідоцтва про право на спадщину
    1) за законом
    2) за заповітом
- Видано свідоцтв про право власності на частку в спільному майні подружжя, у разі смерті одного з подружжя [17]
- Посвідчено довіреностей
    1) на право розпорядження майном
    2) на право розпорядження транспортними засобами
    3) інших довіреностей
- Накладення заборони відчуження нерухомого майна та транспортних засобів, що підлягають реєстрації
- Засвідчення вірності копій (фотокопій) документів
- Засвідчення справжність підпису на документах
- Засвідчення вірності перекладів
- Виконавчий напис
- Вчинення протесту векселя
- Інші нотаріальні дії

4.4 Алгоритм реалізації телеграм-боту

4.4.1 Реєстрація боту в месенджері

Щоб створити будь-якого бота в телеграмі потрібно спочатку його зареєструвати. Для цього в месенджері потрібно знайти бота з назвою @BotFather [18].

Далі потрібно запустити знайденого бота, та діяти за інструкцією. Для створення нового боту відправити команду /newbot, вибрати ім’я боту, та назву за якою його можна буде відшукати в месенджері, оскільки бот є нотаріальним консультантом, то він матиме назву NotaryConsultantBot, важливою ознакою є спеціальне закінчення Bot.

Після вибору імені боту буде наданий токен доступа до бота за допомогою якого можна буде отримати доступ до боту, оскільки він є унікальним, і будь-хто, хто матиме токен доступу зможе використовувати бота, то значення його буде приховане.

4.4.2 Програмування боту

Для початку було обрано спосіб отримання повідомлень від користувачів. Оскільки застосунок поки використовується не для широкої аудиторії, при відсутності Azure акаунту та Google хмари, при необхідності подальшого доповнення текстами та різного роду інформацією було вибрано спосіб отримання оновленням методом getUpdate.

Метод getUpdate [19] отримує повідомлення через long polling. На відповідь викликання методу отримується масив об’єктів Update.

Метод update має поле message, в якому знаходяться необхідні нам данні, які потрібно обробити, і надати відповідну відповідь користувачеві, для цього було реалізовано наступний алгоритм роботи програми:

1) запускається програма з ботом;
2) користувач відправляє повідомлення;
3) програма отримує повідомлення методом getUpdate;
4) спрацьовує метод Bot.OnMessageRecived;

5) програма надсилає повідомлення до Dialogflow;

6) Dialogflow агент отримує повідомлення;

7) агент зіставляє повідомлення з доступними Intents;

8) при знаходженні відповідності (процес отримання контексту повідомлення) формується відповідь;

9) програма отримує відповідь від Dialogflow;

10) програма звіряє відповідь з назвами команд;

11) при відповідності назви команди і відповіді Dialogflow викликається метод Execute відповідної команди. В залежності від команди є декілька варіантів розвитку:

- користувачеві надсилається остаточний результат з необхідною інформацію;
- потрібне уточнення, користувачеві надсилається в якому вказано, якої інформації не вистачає, також надсилається клавіатура з можливими варіантами вибору;
- програма не змогла розпізнати повідомлення користувача, користувачеві надсилається клавіатура з можливими варіантами вибору;

12) програма чекає наступної дії користувача, якщо користувач надсилає повідомлення, алгоритм переходить на пункт 2).

Для реалізації програмної частини боту було обрано мову програмування C#. Для зв’язку боту з програмою булу встановлено бібліотеку TelegramBot. А для зв’язку з сервісом Dialogflow – ApiAiSDK.

Бібліотека TelegramBot надає ряд методів для отримання та обробки повідомлень з серверів Telegram до боту.

Основним класом цієї бібліотеки є TelegramBotClient, який у конструкторі приймає токен доступу, і надає основні функції до роботи з ботом. Цей клас містить обробники подій які викликаються при отримані відповідних даних в запиті. В роботі було використано три із них OnMessage, OnCallbackQuery, OnUpadte. Також для початку, щоб бот запрацював потрібно

задати боту команду початку отримання оновлень Bot.StartReciving (UpateType[]), ця команда приймає масив типу UpdateType.

UpdateType являє собою перерахування для типів можливих оновлень, бот повинен сприймати типи: Message, CallbackQuery.

Message має всю необхідну інформацію про повідомлення отримане від користувача, основним полем в ньому є Text, яке якраз і представляє собою необхідну інформацію для боту, відносно якої він буде надсилати подальші повідомлення, і вести користувача до отримання кінцевого списку необхідних документів.

CallbackQuery повертається при натисненні кнопки вбудованої клавіатури, але бот отримує не той текст, який ввів користувач, а дані, які були на моменті програмування в кнопці, яку натиснув користувач.

Для обробки запиту було створено функцію BotActionOnMessageReceived(), яка приймає повідомлення. Тут виконується перевірка на тип повідомлення та чи воно не пусте, і при задовільному результаті виконується функція ExecuteCommands(), яка приймає індитифікатор користувача, та текст повідомлення.

ExecuteCommands() пробігає весь список команд, і якщо знаходить відповідну (текст == назві команди) то виконує її, інакше передає повідомлення на обробку до сервісу Dialogflow, який отримує контекст повідомлення і надає відповідь у вигляді тексту який збігається з назвою команди, у разі якщо відповідність не була знайдена , то виконується команда за замовчуванням, яка відправляє користувачу клавіатуру з варіантами відповідей.

Кожна команда представляє собою обєкт класу Command, яка має поле для власної назви, поле для зберігання тексту повідомлення, може мати клавіатури та список команди на які вона посилається. Серед методів має метод Execute() та Createboard(). Метод Execute() слугує для відправлення повідомлення до користувача який прислав запит боту. Метод Createboard()

повертає клавіатуру відповідно до списку команд які зв’язані з даною, яка потім надсилається користувачу.

Таблиця 4.1 Опис методів

Назва методу	Данні на вході	Дані на виході	Опис методу
BotActionOn MessageReceived()	Object sender, Message EventArgs e	void	Викликається при отриманні повідомлення, запускає виконання команди ExecuteCommands()
BotActionOn CallbackReceived()	object sender, CallbackQuery EventArgs e	void	Викликається при отриманні callback, запускає виконання команди ExecuteCommands()
ExecuteCommands()	long chatid, string Text	void	Спрацьовує при отриманні повідомлення, шукає відповідну команду до заданого тексту
async Execute()	Long chatid	void	Надсилає повідомлення команди до користувача з індетифікаром чата chatid

Createboard()	List<Command> ListCommands	IReply Markup	Створює об’єкт який реалізує інтерфейс IReplyMarkup відповідно до заданого списку

Продовження таблиці 4.1

BotStart()	-	Void	Запускає бота, програма починає посилає запити до телеграму на отримання повідомлень для боту
BotStop()	-	void	Зупиняє бота, припиняється отримання повідомлень

Під час реалізації боту було передбачено варіант використання, який схематично показано в Додатку А (схема варіантів використання системи).

На діаграмі розглянуті найбільш важливі запити користувача до системи (в даному випадку користувач хоче отримати інформацію про купівлю-продаж гаражу (тут може бути інша нотаріальна дія)), було із прогнозовано, що користувач скоріше за все буде діяти за наступним алгоритмом:

1) користувач хоче отримати інформацію про продаж гаражу, та вирішив знайти інформацію звернувшись до розробленого боту;

2) спочатку він знаходить бота за назвою @NotaryConsultantBot;

3) далі він вибирає зі списку необхідний контакт і переходить у чат з ботом;

4) для початку роботи він натискає кнопку (запустити), надсилається повідомлення з текстом (/start);

5) далі Телеграм отримавши повідомлення від користувача знаходить відповідний контакт (бота) і надсилає йому запит;

6) коли бот запущений, то програма автоматично перевіряє наявність нових повідомлень, і оскільки користувач надіслав (/start) то програма знаходить з команду з відповідною назвою, та виконується метод Execute(), який в даному випадку надсилає повідомлення з текстом для привітання;

7) після того, як користувач отримав привітання, він розуміє, що можна задати необхідне питання і надсилає повідомлення з текстом “гараж”;

8) телеграм пересилає повідомлення програмі, яка в свою чергу не знаходить команду з такою назвою, тому передає повідомлення до Dialogflow;

9) Dialogflow отримавши повідомлення знаходить відповідного агента, якому було надіслано повідомлення, та передає його йому;

10) агент отримавши повідомлення починає пошук відповідності в своїй базі Intents;

11) знайшовши Intent (нерухоме майно) і проаналізувавши список пов’язаних слів, розуміє, що гараж відноситься саме до цієї сутності, і надає відповідь до програми (нерухоме майно);

12) отримавши повідомлення після обробки агентом Dialogflow, програма знову перевіряє чи існує команда з даною назвою;

13) знайшовши команду з назвою (нерухоме майно) відбувається виконання команди в результаті якої користувачеві надсилається список документів які необхідні для переоформлення гаражу, а також клавіатура зі списком роз’яснень по деяким з документів;

14) користувач отримавши необхідну інформацію вирішив порозважатися з ботом і ввів повідомлення з текстом (де купити морозива?);

15) бот отримавши повідомлення не знаходить відповідної команди і надсилає до агента;

16) агент не знайшовши відповідної сутності, надсилає пусте повідомлення;

17) отримавши пусте повідомлення від агенту програма виконує команду за замовчуванням для незрозумілих повідомлень, надсилає повідомлення з текстом (вибачте я вас не розумію, спробуйте вибрати варіант зі списку);

18) користувач отримавши повідомлення припиняє діалог з ботом.

4.5 Висновки до розділу

Практичним результатом дипломного проекту стало створення чат-боту для месенджеру Telegram, з використанням технологій сервісу Dialogflow, на мові програмування C#, з використанням бібліотек APIAISDK та Telegram.Bot. За допомогою цих технологій вдалося створити автоматизованого консультанта, який вміє розпізнавати певний контекст повідомлень, зв’язаний з нотаріальною діяльністю, ставити уточнюючі запитання, надавати варіанти вибору певних запитів, та в результаті надавати список документів, які необхідні для тої чи іншої нотаріальної дії, а також надавати роз’яснення по деяким з них.

5 ТЕСТУВАННЯ ТА ТРЕНУВАННЯ БОТУ

Оскільки було створено чат-бот саме для мессенджеру Telegram, то для тестування було використано пристрої сумісні з Telegram. Мессенджер можна використовувати, як на мобільних пристроях, так і на персональних комп’ютерах та ноутбуках.

Тестування боту проводилося на мобільному пристрої Xiaomi Redmi 5 з технічними характеристиками:

Екран: 5,7 ", TFT-IPS, 1440x720, мультитач

Процесор: Qualcomm Snapdragon 450, 8x1,8 ГГц (Cortex-A53)

Графічний прискорювач: Adreno 506

Операційна система: Android 7.1.2

Оперативна пам'ять: 4 ГБ

Вбудована пам'ять: 32 ГБ

Навігація: GPS

Акумулятор: 3300 мАг

Габарити: 151,8x72,8x7,7 мм

Telegram: Telegram v 5.5.0

Також на персональному комп’ютері з технічними характеристиками:

Відеоадаптер: AMD Radeon HD 7480D

Процесор: AMD Radeon 3.40GHz

Операційна система: Microsoft Windows 7 Ultimate

Оперативна пам'ять: 6 ГБ

Вбудована пам'ять: 1,5 ТБ

Telegram: Telegram desktop v 1.7.7

5.1 Тестування основних команд та переходів

Після заповнення списку команд та відповідних реакцій боту, а також після початкового заповнення Intents агента в Dialogflow було проведене первинне тестування боту.

Спочатку було встановлено Telegram на ПК та проведений процес авторизації до месенджеру. Далі було знайдено створеного бота за його назвою та виконана команда запуску.

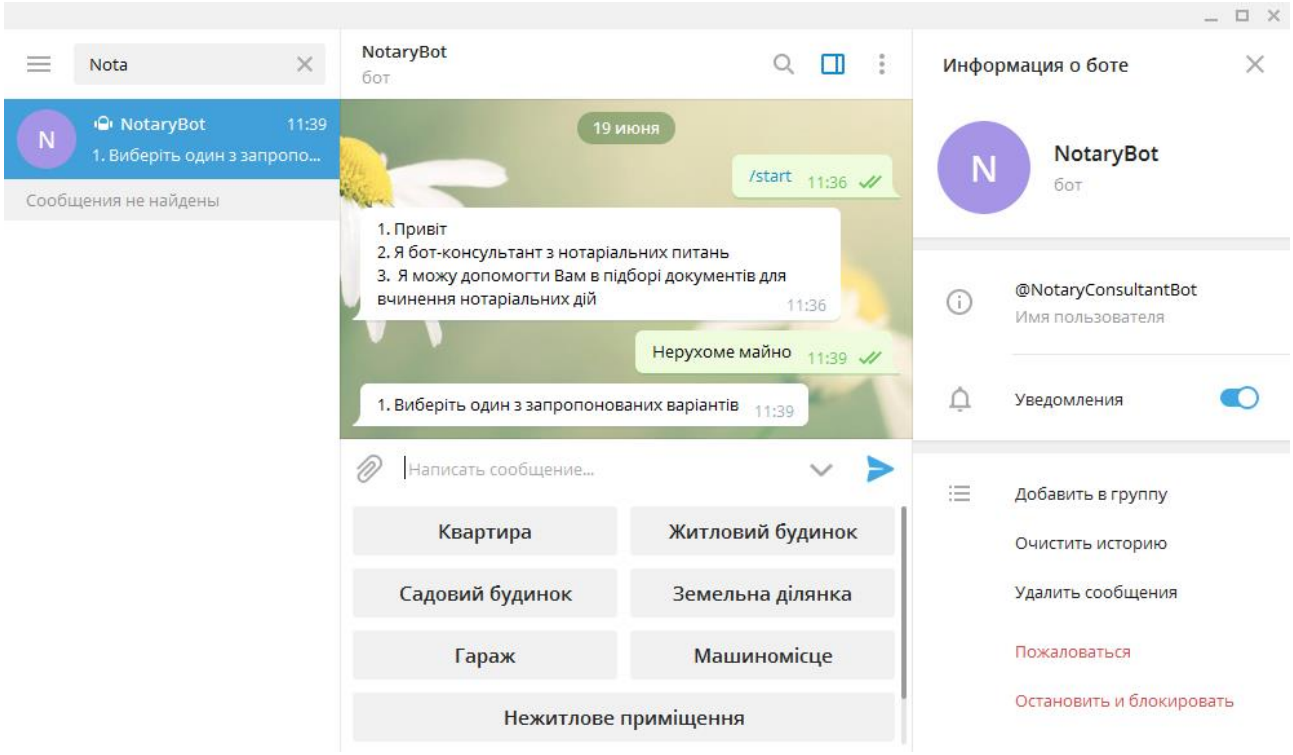

Рисунок 5.1 – Запуск боту

Протестуємо, що користувач хоче отримати інформацію про нерухоме майно, наприклад він хоче дізнатися, які необхідні документи мати для купівлі гаражу.

Після вибору варіанта на клавіатурі (гараж) користувач отримує список необхідних документів, які будуть задіяні при проведені правочинів (рисунок 5.2).

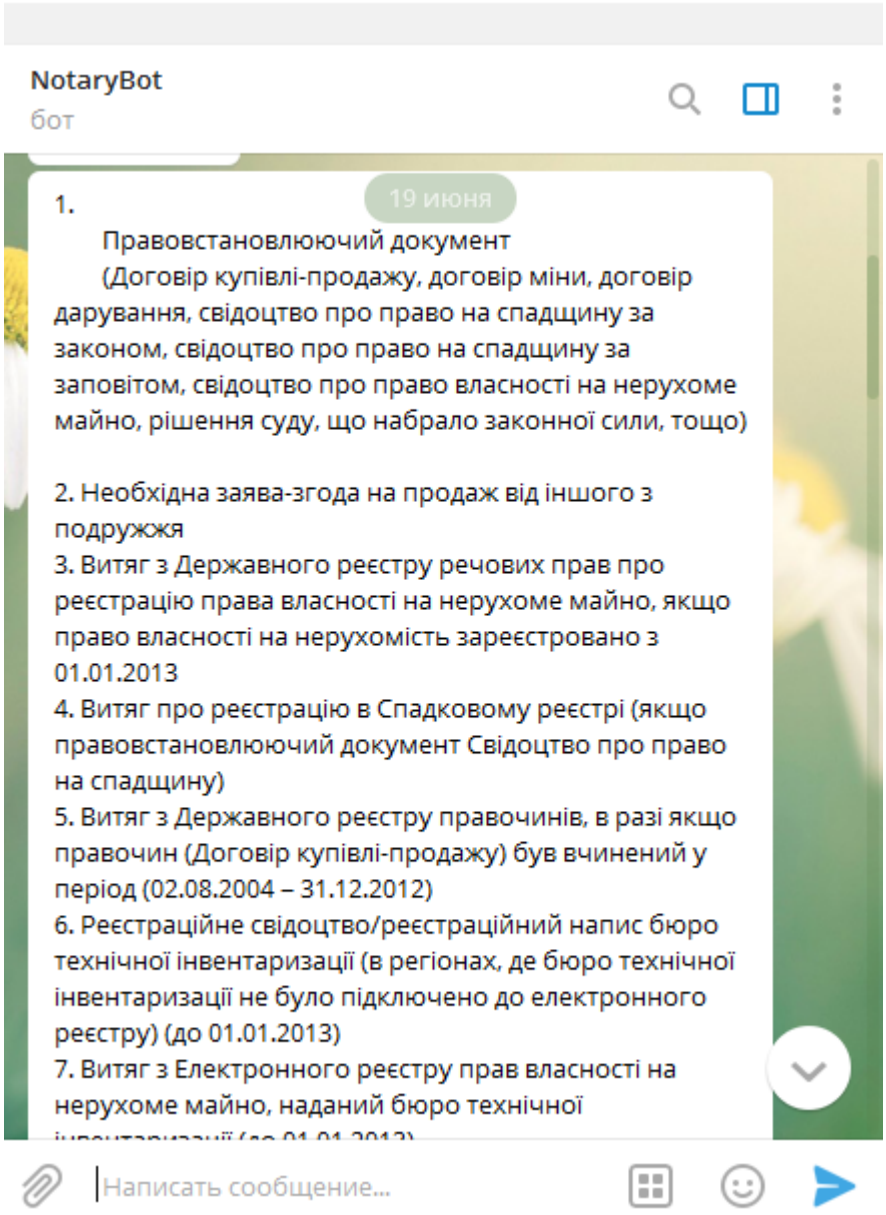

Рисунок 5.2 – Список документів

Користувачеві надасться вбудована клавіатура, зі списком документів з декількома варіантами вибору відповідно до статусу користувача в межах вчинення правочинів на території України (рисунок 5.3).

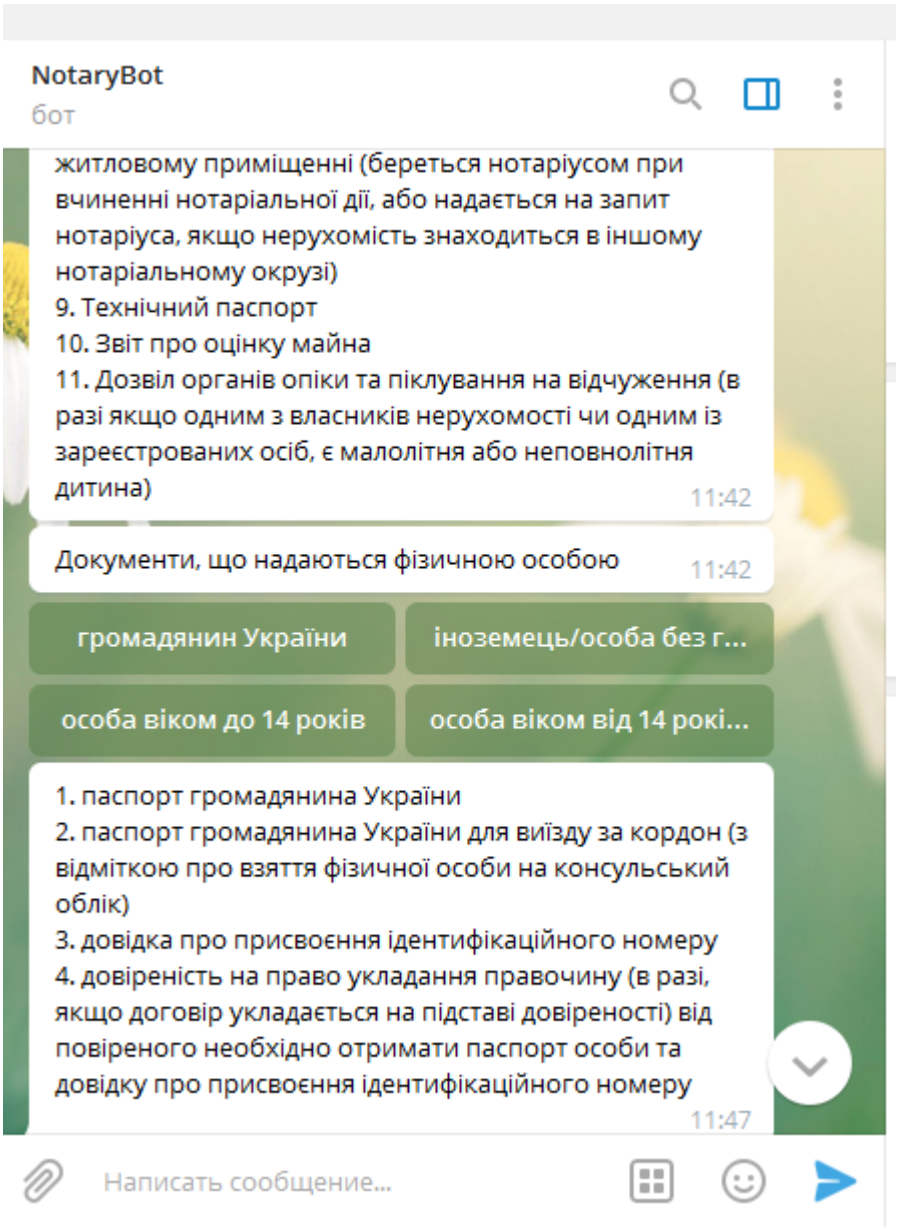

Рисунок 5.3 – Уточнення запиту за допомогою вбудованої клавіатури

Також потрібно перевірити, як буде себе поводити бот при некоректному вводі, наприклад, якщо користувач вирішив використати бота не для предметної області (рисунок 5.4).

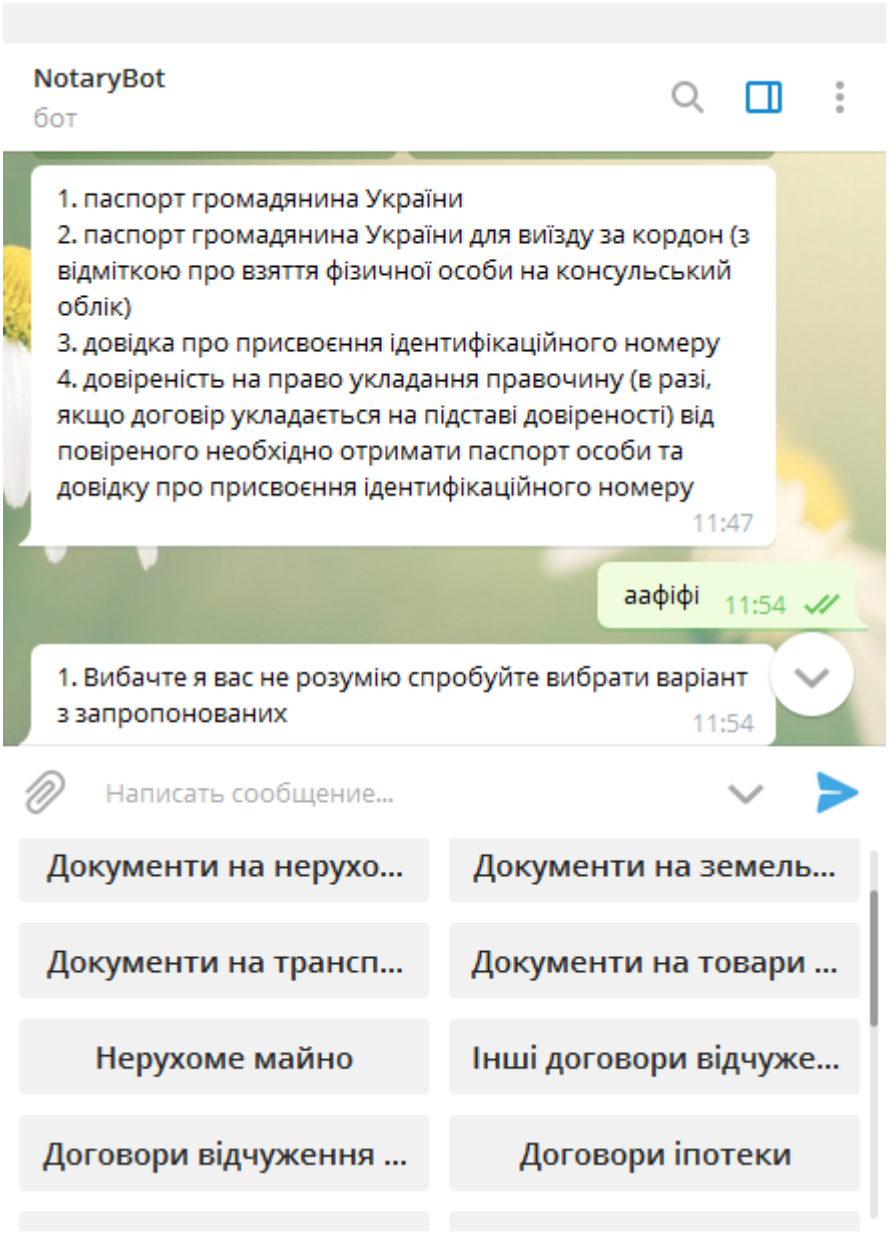

Рисунок 5.4 – Тестування реакції боту на некоректний запит

Як видно з графічного матеріалу, при некоректному запиті бот буде виводити повідомлення, що він не зрозумів користувач і надасть варіанти вибору.

5.2 Тестування методів розпізнавання контексту боту

Як було зазначено раніше, для того, щоб бот вмів відповідати не тільки на повідомлення, які в точності співпадають з назвою команди, був підключений сервіс Dialogflow. Завдяки цьому сервісу було створено агента та його базу знань (рисунок 5.5) відповідно до якої він повинен розпізнавати природню мову користувача, та повертати відповідну команду боту.

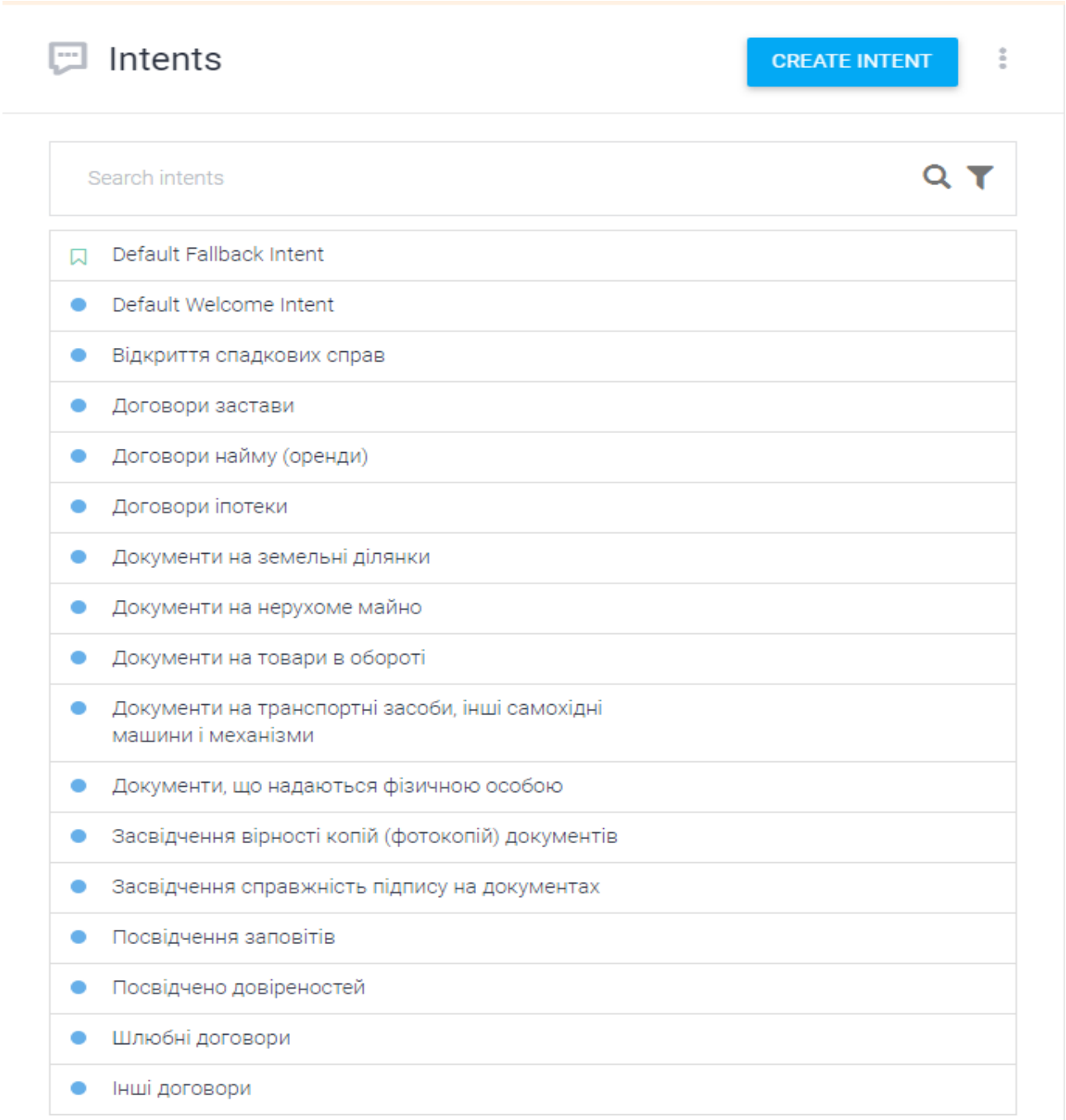

Рисунок 5.5 – Список відповідних сутностей боту

Перевіримо, як бот буде реагувати на ввід різних видів транспортів з різною структурою повідомлення.

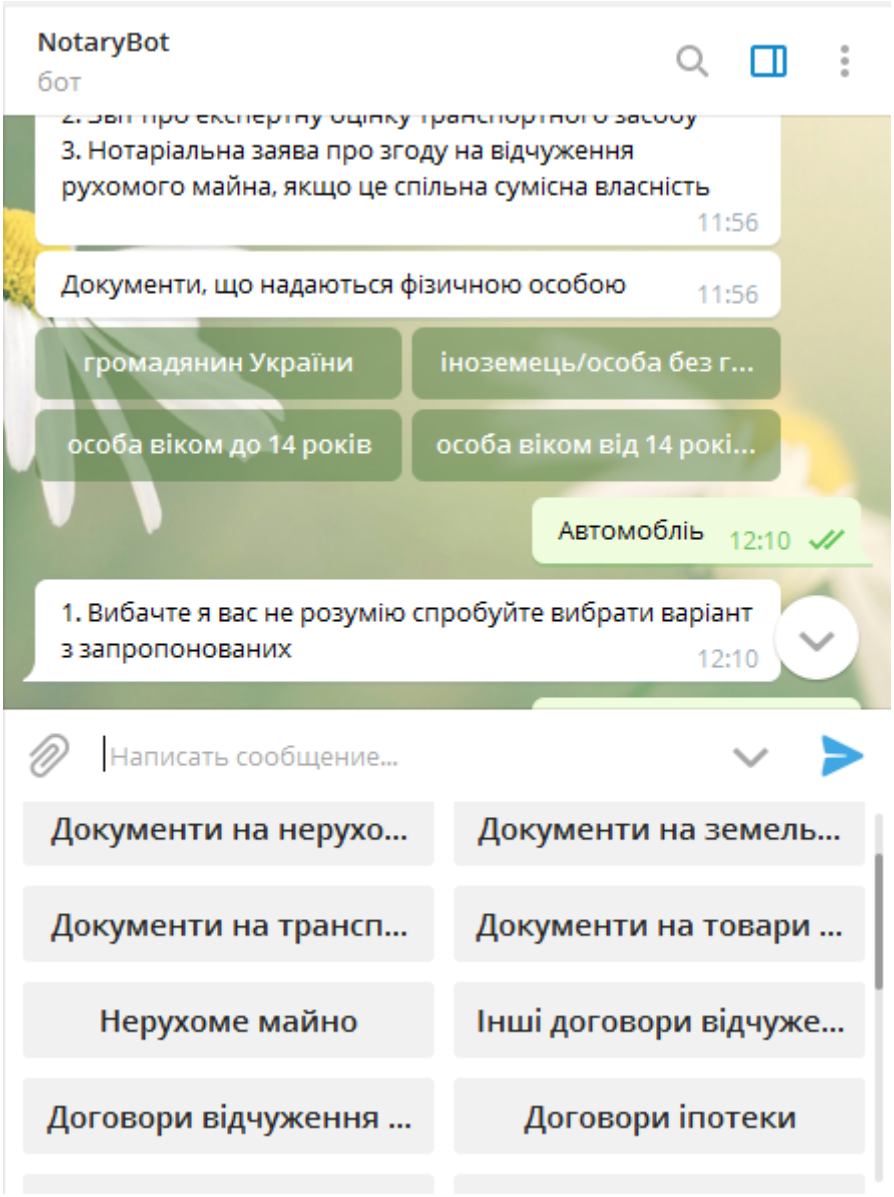

Рисунок 5.6 – Помилка вводу користувача

Під час вводу слова автомобіль було допущено помилку, на що бот відреагував, що він не розуміє користувача, для роз’яснення відповідного повідомлення, потрібно зайти в режим тренування агента DF.

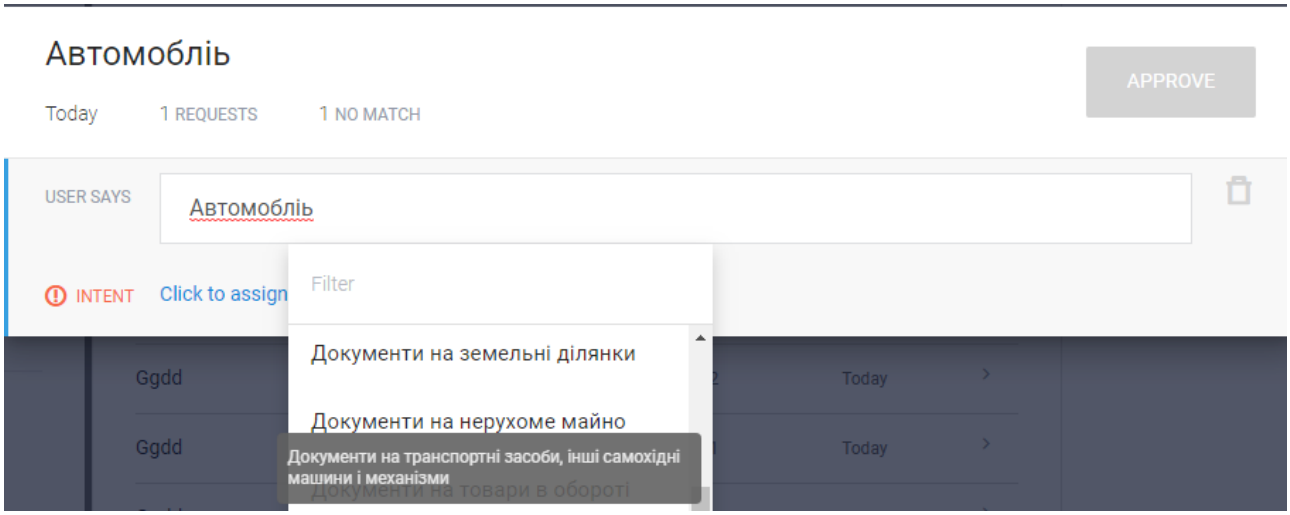

Рисунок 5.7 – Тренування агента

Також на сторінці тренування можна переглянути всі запити користувачів та пояснити агенту, чи правильно він відповів в тій чи іншій ситуації (рисунок 5.8).

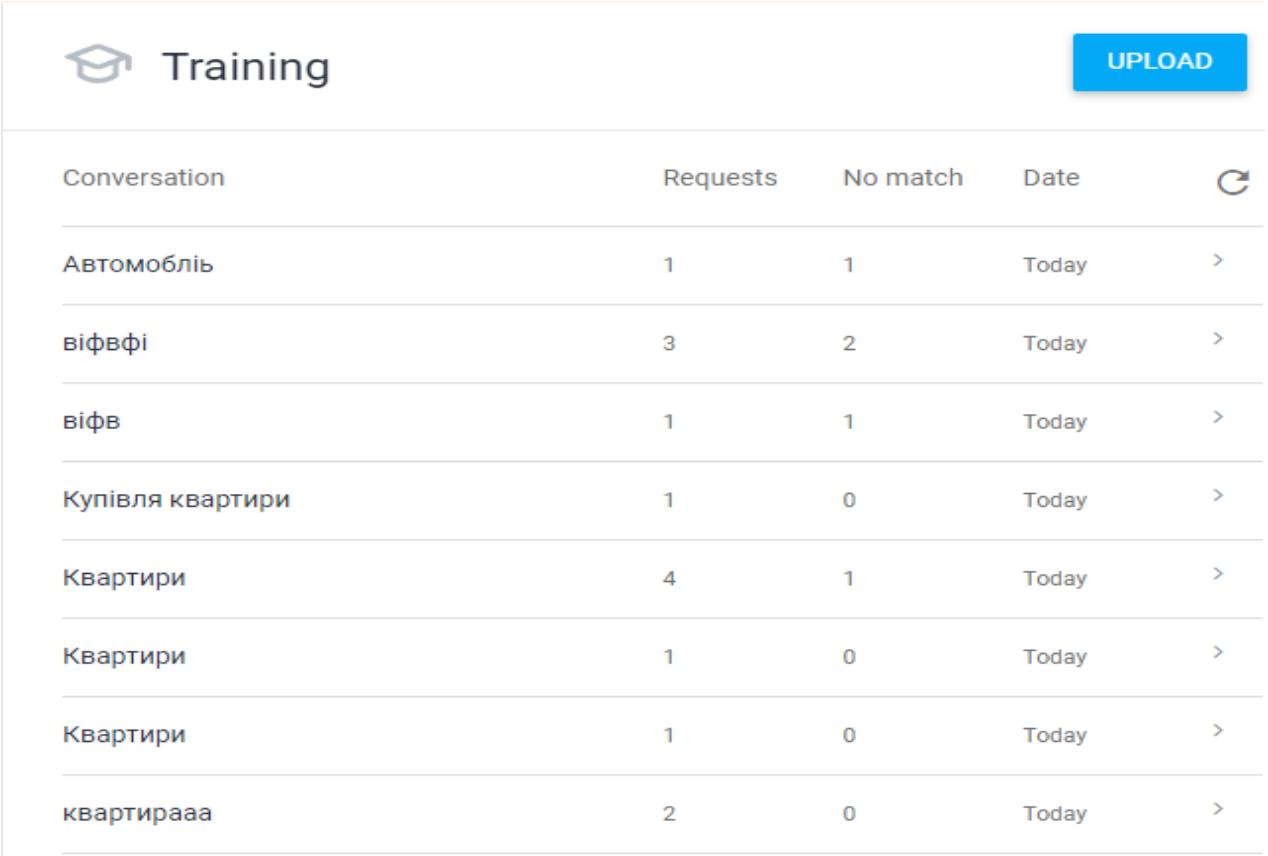

Рисунок 5.8 – Запити користувачів

Після внесення змін перевіримо, чи зможе бот зрозуміти повідомлення (рисунок 5.9).

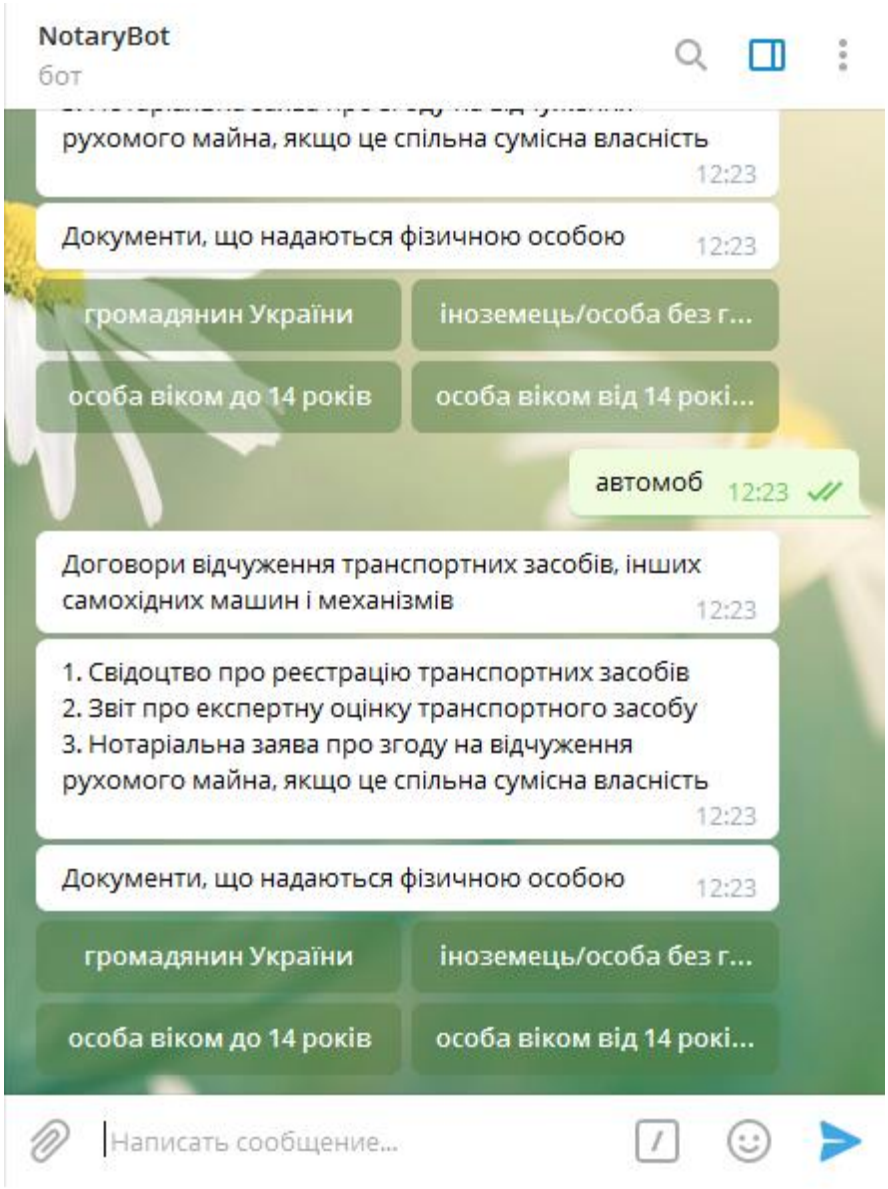

Рисунок 5.9 – Перевірка отримання контексту повідомлення

5.3 Висновки до розділу

В процесі тестування та тренування боту, було протестованно, як бот виконує свою першочергову функцію, та допомогає з отриманням інформації, щодо збору документів, які необхідні для вчинення тієї чи іншої нотаріальної дії. Також було описано процес тренування боту, і реалізації методів розпізнавання контексту повідомлень.

ВИСНОВОК

В процесі виконання дипломного проекту було створено чат-бот нотаріального консультанта для месенджеру Telegram. Було доведено актуальність теми створення чат-ботів, бо майже кожна велика компанія потребує персонал який займається консультуванням їхніх користувачів по питанням створюваної продукції, а завдяки чат-ботам можна замінити навіть call-center на більше ніж 50 осіб всього одним ботом. Завдяки ботам можна автоматизувати різного роду процеси, можна знайти та обробити необхідну інформацію просто надавши боту відповідний запит, який зможе в разі чого уточнити, що користувач мав на увазі, з ботом можна грати в ігри, бот може вести щоденний органайзер та нагадувати про певні події. Тобто боти можуть спрощувати, або самі виконувати певні рутинні операції покладенні на людей. Ця рутина з часом навіть може призводити до погіршення стану здоров'я людини.

Основними перевагами чат-ботів було визначено: не потребують великої кількості ресурсів системи, легко інтегруються інтернет ресурси, соціальні мережі, месенджери, тощо, досить просто реалізувати, можливість оброблювати та реагувати на певну інформацію.

Створений бот може стати досить часто використовуваним ресурсом, і полегшити життя людей при оформленні різних нотаріальних правочинів. Він автоматизує процес роз'яснення та виконує роль консультанта при зверненні до нього користувачів, які хочуть отримати інформацію про певну нотаріальну дію. Однією з переваг боту, що він підключений до сервісу Dialogflow, та вміє розпізнавати контекст повідомлень, а у разі не знайдення відповідності його досить легко і швидко можна навчити, як правильно розпізнавати той чи інший запит.

Математичним апаратом боту стало використання сервісу Dialogflow, в основу агентів якого покладено алгоритми машинного навчання, які надають

методи і функції для обробки запитів користувачів на природній мові. Сервіс може реалізовувати функцію автоматичної корекції орфографії, що надає боту можливість розуміти навіть ті повідомлення, в яких було просто допущено помилку вводу.

Також було розглянуто альтернативні варіанти для створення алгоритму розпізнавання контексту наприклад методи векторного представлення тексту.

Під реалізації боту було реалізовано зв'язок між Telegram, програмним кодом бота та сервісів Dialogflow.

В реалізації чат-боту було використано доступ до даних що можуть наповнюватись і змінюватись незалежно від програмної реалізації без необхідності перекомпіляції проекту. Така реалізація дозволяє створювати нові теми для консультування, а також в подальшому більш детально отримувати відповіді на поставлені питання.

В подальшому можливо реалізувати модулі аналізу та отриманню статистики по найбільш актуальним питанням від користувачів. Це дозволить мати зворотній зв’язок, аналізуючи запити і відповіді розширювати тематику та поглиблювати рівень наданих консультативних послуг.

Ідеї створення чат-ботів об’єднали безліч розробників, бізнесів, сферу послуг для створення нової екосистеми зі своїми правилами, методами конкуренції, ідеями розвитку. За допомогою чат-ботів для невеликого бізнесу або самозайнятих осіб стало можливим просування товарів і послуг з обмеженим бюджетом на автоматизацію діяльності, рекламні компанії. Також підвищився рівень зворотного зв’язку від надавача послуг або товарів до кінцевого користувача. Відомі проблеми малого бізнесу: мала кількість співробітників, слабкі технічні потужності, обмеженість ресурсів - все це позначається на якості і ефективності взаємодії як з існуючими так і потенційними клієнтами. Дана екосистема допомагає створювати більш інтерактивний аналог доставлення інформації у відмінності від статичного пошуку по веб-сайтам, класичним пошуковим системам, завдяки вузької

спеціалізації та більш точним відповідям без необхідності обробки безлічі сторінок з приблизно релевантними результатами. Все це доводить актуальність розробки, а також перспективність обраної теми та її користь для багатьох замовників подібних додатків.

Подальший розвиток систем штучного інтелекту дозволить інтегрувати в чат-боти системи голосового консультування, системи керування розумним офісом, автомобілем, систем автоматичного замовлення товарів і послуг, запису на прийом до фахівця, замовлення туристичних турів, екскурсій, резервування житла та безліч інших можливостей.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1) Most popular global mobile messenger apps [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.statista.com/statistics/258749/most-popular-globalmobile-messenger-apps/.

2) Radziwill N. Evaluating Quality of Chatbots and Intelligent Conversational Agents [Електронний ресурс] / N. Radziwill, M. Benton. – 2017. – Режим доступу до ресурсу: https://arxiv.org/ftp/arxiv/papers/1704/1704.04579.pdf

3) Матеріал з Вікіпедії – вільної енциклопедії (Робот (програма)) [Електронний ресурс] : Режим доступу: https://uk.wikipedia.org/wiki/Робот_(програма)

4) Telegram Bot API [Електронний ресурс] : Режим доступу: https://core.telegram.org/bots/api

5) Від порятунку життя і до мін у метро: 7 українських ботів, що змінюють світ [Електронний ресурс] : Режим доступу: https://mmr.ua/show/vid_poryatunku_zhittya_i_do_min_u_metro_7_ukrayinsykih_botiv__shto_zminyuyuty_svit

6) Документация Telegram [Електронний ресурс] – Режим доступу: https://tlgrm.ru/docs/bots/api

7) Закон України «Про нотаріат» [Електронний ресурс] – Режим доступу: (https://zakon.rada.gov.ua/laws/show/3425-12)

8) Подборка: 30 полезных ботов [Електронний ресурс] – Режим доступу: https://ain.ua/2017/08/14/30-telegram-botov/

9) Tensorflow official website [Електронний ресурс] – Режим доступу: https://www.tensorflow.org/

10) Google приобрела российский стартап по разработке голосового интерфейса Api.ai [Електронний ресурс] – Режим доступу: https://vc.ru/tribuna/18544-google-buy-apiai

11) Get started Dialogflow [Електронний ресурс] – Режим доступу:https://dialogflow.com/docs

12) Умные чат-боты с Dialogflow [Електронний ресурс] – Режим доступу:https://singularika.com/ru/technologies/dialogflow-development/

13) How intent classification works in NLU [Електронний ресурс] – Режим доступу: https://mrbot.ai/blog/natural-language-processing/understanding-intent-classification/

14) Machine Learning - Машинне навчання [Електронний ресурс] – Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/machine-learning

15) Векторная модель представления текстовой информации С. В. Моченов, А. М. Бледнов, Ю. А. Луговских Ижевский государственный технический университет. Материалы международной научной конференции Ижевск, 13–17 июля 2006 г.

16) Создаем Telegram бота на API.AI [Електронний ресурс] : Режим доступу: https://habr.com/ru/post/336668/

17) Порядок вчинення нотаріальних дій нотаріусами України [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/z0282-12

18) Правила ведення нотаріального діловодства [Електронний ресурс] : Режим доступу: (https://zakon.rada.gov.ua/laws/show/z0282-12)

19) Сімейний кодекс України [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/2947-14

20) Инструкция: Как создавать ботов в Telegram [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/262247/

21) Examples for the Telegram.Bot C# Library [Електронний ресурс] – Режим доступу: https://github.com/TelegramBots/Telegram.Bot.Examples/blob/master/Telegram.Bot.Examples.Echo/Program.cs

22) C# (CSharp) ApiAiSDK RequestExtras примеры использования [Електронний ресурс] – Режим доступу: https://csharp.hotexamples.com/ru/examples/ApiAiSDK/RequestExtras/-/php-requestextras-class-examples.html

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Linq;
using System.Threading.Tasks;
using Telegram.Bot.Args;
using Telegram.Bot;
using NotaryBot.Commands;
using ApiAiSDK;
using ApiAiSDK.Model;
using Telegram.Bot.Types.Enums;
using Telegram.Bot.Types.ReplyMarkups;
using System.Xml.Linq;
using System.Xml;

namespace NotaryBot
{
    class Bot
    {
        private static TelegramBotClient BOT;
        private static List<Command> commandsList = new List<Command>();
        public static IReadOnlyList<Command> Commands { get =>
commandsList.AsReadOnly(); }
        private static ApiAi apiAi;
        private static List<Command> startList = new List<Command>();
        private static Command standarcom;
        public Bot()
        {
```

```
            apiAi = new ApiAi(AppSettings.configuration);
            BOT = new TelegramBotClient(AppSettings.Key);
            CreateCommandList();
            createdefaultcom();
            foreach (var item in commandsList)
            {
              // Console.WriteLine(item.Name, item.Text);
               if (item.ListOfCommands != null)
               {
                  foreach (var item2 in item.ListOfCommands)
                  {
                    // Console.WriteLine("{0} принадлежит колекции {1}",
item2.Name, item.Name);
                  }
               }
            }
            BOT.OnMessage += BotOnMessageReceived;
            BOT.OnCallbackQuery += BotOnCallBackRecived;

        }
        private static void createdefaultcom()
        {
            standarcom = new Command("default");
            standarcom.addtext("Вибачте я вас не розумію спробуйте вибрати
варіант з запропонованих");
            standarcom.ListOfCommands = startList;
            standarcom.createboard(standarcom.ListOfCommands);
            standarcom._id(0);
            standarcom._ridADD(3337);
```

```
    }

    private void BotOnMessageReceived(object sender, MessageEventArgs e)
    {
        Telegram.Bot.Types.Message message = e.Message;
        if (message == null || message.Type != MessageType.Text) return;
        string answer = message.Text;
        Console.WriteLine(message.Text);
        ExecuteCommands(message.Chat.Id, answer);
        Console.WriteLine(message.Text);
    }

    public void BotStart()
    {
        BOT.StartReceiving(new UpdateType[] { UpdateType.Message,
UpdateType.CallbackQuery,UpdateType.InlineQuery,UpdateType.Unknown });
    }
    public void BotStop()
    {
        BOT.StopReceiving();
    }
    private static void BotOnCallBackRecived(object sender,
CallbackQueryEventArgs e)
    {
        string Text = e.CallbackQuery.Data;
        if (Text == null) return;
        string answer = Text;
        ExecuteCommands(e.CallbackQuery.From.Id, answer);
```

```
    Console.WriteLine("CallBack");
}
public static void ExecuteCommands(long chatid, string Text)
{
    bool defaultcount = true;
    foreach (var com in commandsList)
    {
        if (com.Name == Text)
        {
            com.Execute(chatid, BOT);
            defaultcount = false;
            return;
        }
    }
    if (defaultcount)
    {
        var response = apiAi.TextRequest(Text);
        Text = response.Result.Fulfillment.Speech;

    }
    foreach (var com in commandsList)
    {
        if (com.Name == Text)
        {
            com.Execute(chatid, BOT);
            defaultcount = false;
            return;
        }
    }
```

```
            if (defaultcount)
            {
                standarcom.Execute(chatid, BOT);
            }
        }
        private static void CreateCommandList()
        {
            string fileName = "U:/Video/NotaryBot -
копия/NotaryBot/NotaryBot/CommandsData.xml";
            BotClassObj bot = new BotClassObj(fileName);
            Command command;

            foreach (var item in bot.commandObjsList)
            {
                command = new Command(item.Name);
                if (item.textslist != null)
                    command.SetText(item.textslist);
                else if (item.Text != null)
                    command.addtext(item.Text);
                    command._id( item.id);



                if (item.refid != null)
                {
                    string[] s = item.refid.Split(",");
                    foreach (var rid in s)
                    {
                        command._ridADD(Convert.ToInt32(rid));
                    }
```

```
    }
    if (item.ComList.Count > 0)
    {
        foreach (var com in item.ComList)
        {
            command.AddCommandToList(getnode(com));
           // Console.WriteLine(com.Name);
        }
    }
    if (item.keyboard != null && item.keyboard != "")
    {
        command.boardname = item.keyboard;
        if (item.keyboard == "reply")
        {

            command.createboard(command.ListOfCommands);
        }
        else if(item.keyboard == "inline")
        {
            command.createInlineboard(command.ListOfCommands);
        }
    }
    commandsList.Add(command);
    startList.Add(command);
}
foreach (var item in commandsList)
{
    if (item.refid != null)
    {
```

```
            for (int i = 0; i < item.refid.Count; i++)
            {
                foreach (var com in commandsList)
                {
                    if (item.refid[i] == com.id)
                    {
                        item.AddCommandToList(com);
                        if (item.boardname == "reply")
                        {
                            item.createboard(item.ListOfCommands);
                        }
                        else if (item.boardname == "inline")
                        {
                            item.createInlineboard(item.ListOfCommands);
                        }
                    }
                }
            }
        }
    }
private static Command getnode(CommandObj com)
{
    Command command = new Command(com.Name);

        command = new Command(com.Name);
        if (com.textslist != null)
            command.SetText(com.textslist);
        else if (com.Text != null)
```

```
            command.addtext(com.Text);
        command._id(com.id);

            if (com.refid != null)
            {
                string[] s = com.refid.Split(",");
                foreach (var rid in s)
                {
                    command._ridADD(Convert.ToInt32(rid));
                }
            }
            if (com.ComList.Count > 0)
            {
                foreach (var com2 in com.ComList)
                {
                    command.AddCommandToList(getnode(com2));
                    Console.WriteLine(com2.Name);
                }
            }
            if (com.keyboard != null && com.keyboard != "")
            {
            command.boardname = com.keyboard;
            if (com.keyboard == "reply")
                {
                    command.createboard(command.ListOfCommands);
                }
                else if (com.keyboard == "inline")
                {
                    command.createInlineboard(command.ListOfCommands);
```

```
                }
            }
                commandsList.Add(command);
        return command;
```

XMLPARSER

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Xml;
using System.Linq;
using System.Xml.Linq;

namespace NotaryBot
{
    class xmlparser
    {

    }
    public class CommandObj
    {
        public Int32 id = 0;
        public String Name = "";
        public String Text = "";
        public String refid = "";
        public String notrefid = "";
        public String keyboard = ""; // inline,reply
```

```
        public List<CommandObj> ComList = new List<CommandObj>();
        public List<string> textslist= new List<string>();

        public static String GetXmlAttrib(XElement elm, string attrname, String
defVal)
        {
            XAttribute xmlAttrib = elm.Attribute(attrname);
            if (xmlAttrib == null)
                return defVal;

            string val = xmlAttrib.Value;
            if (val == null || val == string.Empty)
                return defVal;

            return val;
        }

        public static String GetElmText(XElement elm, String elName, String defVal)
        {
            XElement xmlElm = elm.Element(elName);
            if (xmlElm == null)
                return defVal;

            var val = xmlElm.Value;
            if (val == null) // || val.IsEmpty)
                return defVal;

            return val;
        }
```

```
        public CommandObj(XElement elm)
        {
            id = Int32.Parse(GetXmlAttrib(elm, "id", "0"));
            Name = GetElmText(elm, "name", "No name");
            Text = GetElmText(elm, "text", Name);
            refid = GetXmlAttrib(elm, "refid", "3337");
            notrefid = GetXmlAttrib(elm, "notrefid", "0");
            keyboard = GetXmlAttrib(elm, "keyboard", ""); // inline,reply

            if (elm.Elements("commands").Count() > 0)
            {
                var items = from xel1 in
elm.Elements("commands").Elements("command")
                            where (xel1 != null) && (xel1.Element("name").Value != "")
                            select new CommandObj(xel1);

                foreach (var CommandObj in items)
                {
                    ComList.Add(CommandObj);
                //    Console.WriteLine("id={0} - {1} - {2}", CommandObj.id,
CommandObj.Name, CommandObj.Text);
                //    Console.WriteLine("      refid={0} - {1} - {2}", CommandObj.refid,
CommandObj.notrefid, CommandObj.keyboard);
                }

            }
            if (elm.Elements("texts").Count() > 0)
```

```
        {
            List<string> strlisttext = new List<string>();
            var items = from xel1 in elm.Elements("texts").Elements("text")
                        where (xel1 != null)
                        select new string(xel1.Value);

            foreach (var txt in items)
            {
                strlisttext.Add(txt);
            }
            textslist = strlisttext;
        }

    }
}


public class BotClassObj
{
    public List<CommandObj> commandObjsList = new List<CommandObj>();
    public BotClassObj(string document)
    {
        XDocument xdoc = XDocument.Load(document);
       // Console.WriteLine("command counts={0} ",
xdoc.Root.Elements("command").Count());

        var items = from xel in xdoc.Root.Elements("command")
                    where (xel != null) && (xel.Element("name").Value != "")
                    select new CommandObj(xel);
```

```
            foreach (var CommandObj in items)
            {
                commandObjsList.Add(CommandObj);
             //  Console.WriteLine("id={0} - {1} - {2}", CommandObj.id,
CommandObj.Name, CommandObj.Text);
            }

        }
    }
}

Command.cs
using System;
using System.Collections;
using System.Collections.Generic;
using System.Text;
using Telegram.Bot;
using Telegram.Bot.Args;
using Telegram.Bot.Types;
using Telegram.Bot.Types.ReplyMarkups;
namespace NotaryBot.Commands
{
    class Command : IEnumerable
    {
        public string Name { get; }
        private List<string> Text= new List<string>();
        public int id { get; private set; }
        public List<int> refid { get; private set; } = new List<int>();
```

```
public List<int> norefid { get; private set; } = new List<int>();
public List<Command> ListOfCommands= new List<Command>();
public string boardname;
private ReplyKeyboardMarkup KeyboardSimple;
private InlineKeyboardMarkup InlineKeyboard;



public void _id(int n)
{
   id = n;
}
public void _ridADD(int n)
{
   refid.Add(n);
}
public void _nridADD(int n)
{
   norefid.Add(n);
}
public void addtext(string t)
{
   if (Text[0] == Name)
      Text[0] = t;
   else
   Text.Add(t);
}
public Command(string name, List<string> text=null,int ID=0 )
{
   Name = name;
```

```
        id = ID;

        if (text == null||text[0]=="")
        {
            Text.Add(name);
        }
        else
        {
            Text = text;
        }

    }

    public void createboard(List<Command> CommandList)
    {
        var rows = new List<KeyboardButton[]>();
        var cols = new List<KeyboardButton>();
        for (var Index = 1; Index < CommandList.Count + 1; Index++)
        {
            cols.Add(new KeyboardButton(CommandList[Index - 1].Name));
            if (CommandList.Count % 2 != 0 && Index == CommandList.Count)
            {
                rows.Add(cols.ToArray());
            }
            if (Index % 2 != 0 && Index != CommandList.Count) continue;
            rows.Add(cols.ToArray());
            cols = new List<KeyboardButton>();

        }
```

```
        ReplyKeyboardMarkup board = new ReplyKeyboardMarkup();
        board.Keyboard = rows.ToArray();
        board.ResizeKeyboard = true;
        KeyboardSimple = board;
    }
    public void createInlineboard(List<Command> CommandList)
    {
        var rows = new List<InlineKeyboardButton[]>();
        var cols = new List<InlineKeyboardButton>();
        for (var Index = 1; Index < CommandList.Count + 1; Index++)
        {
            cols.Add(new InlineKeyboardButton()
            {
                Text = CommandList[Index-1].Name,
                CallbackData = CommandList[Index-1].Name,
            });
            if (CommandList.Count % 2 != 0 && Index == CommandList.Count)
            {
                rows.Add(cols.ToArray());
            }
            if (Index % 2 != 0 && Index != CommandList.Count) continue;
            rows.Add(cols.ToArray());
            cols = new List<InlineKeyboardButton>();

        }
        InlineKeyboardMarkup board = new
InlineKeyboardMarkup(rows.ToArray());
```

```
            InlineKeyboard = board ;
        }
        public async void Execute(long chatId, TelegramBotClient client)
        {
            string sendtext="";
            if (Text.Count == 0)
            {
                sendtext = Name;
            }
            else
            {
                for (int i = 0; i < Text.Count; i++)
                {
                    sendtext += (i + 1)+". " + Text[i] + "\n";
                }

            }

                if (KeyboardSimple != null)
                {
                    await client.SendTextMessageAsync(chatId, sendtext, replyMarkup:
KeyboardSimple);
                }
                else if (InlineKeyboard != null)
                {
                    await client.SendTextMessageAsync(chatId, sendtext, replyMarkup:
InlineKeyboard);
                }
                else
```

```
        {
            await client.SendTextMessageAsync(chatId, sendtext);
        }

    if (refid != null && refid[0]<3337)
    {
        for (int i = ListOfCommands.Count-1; i >= 0; i--)
        {
            ListOfCommands[i].Execute(chatId, client);
        }
    }
}

public void SetText(List<string> txtlist)
{
    Text = txtlist;
}
public void SetCommands(List<Command> comlist)
{
    ListOfCommands = comlist;
}
public void AddCommandToList(Command com)
{
    ListOfCommands.Add(com);
}

public IEnumerator GetEnumerator()
{
    return ((IEnumerable)ListOfCommands).GetEnumerator();
```

```
        }
    }
}
```

ДОДАТОК Д-1 ДІАГРАМА КОМПОНЕНТІВ

ДОДАТОК Д2 ДІАГРАМА АКТИВНОСТІ

ДОДАТОК Д3 ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

ДОДАТОК Д4 ДІАГРАМА ПОСЛІДОВНОСТІ ОБРОБКИ ЗАПИТУ