

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
факультет комп'ютерних наук і технологій
(повне найменування інституту, факультету)

кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до магістерської роботи

магістра

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
МЕТОДІВ АНАЛІЗУ КОНТЕНТУ ДЛЯ СТРУКТУРИЗАЦІЇ
ТЕКСТОВОЇ ІНФОРМАЦІЇ. RESEARCH AND SOFTWARE
IMPLEMENTATION OF CONTENT ANALYSIS METHODS FOR
STRUCTURING TEXTUAL INFORMATION

Виконав: студент II курсу, групи КНТ-129м

Спеціальності 121 Програмна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Рязанов Д.А.

(прізвище та ініціали)

Керівник Каплієнко Т.І.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ПРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри
програмних засобів

д.т.н., проф. Субботін С.О.
« » 2020 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Рязанов Данило Андрійович

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів аналізу контенту для структуризації текстової інформації. Research and Software Implementation of Content Analysis Methods for Structuring Textual Information
керівник проєкту (роботи) Каплієнко Тетяна Ігорівна, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від «03» листопада 2020 року № 301

2. Строк подання студентом проєкту (роботи) 12.12.2020

3. Вихідні дані до проєкту (роботи) технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Реферат, Перелік умовних позначень, символів, одиниць, скорочень і термінів, Вступ, 1. Аналіз проблеми та постановка завдань дослідження, 2. Обґрунтування методу для побудови системи аналізу контенту для структуризації текстової інформації 3. Проєктування архітектури системи, 4. Основні рішення щодо реалізації компонентів системи, 5. Експлуатація та тестування програми, 6 Економіко-організаційна частина, 7 Охорона праці та безпека в надзвичайних ситуаціях, Висновки, Перелік посилань, Додатки
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації.

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5	Каплієнко Т. І., доцент		
6	Остапенко В.В., доцент		
7	Коробко О.В., ст.викладач		
Нормо-контроль	Камінська Ж.К.М.В., асистент		

7. Дата видачі завдання «01» вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Аналіз та дослідження предметної області	1-2 тиждень	
2	Розробка та затвердження технічного завдання	3 тиждень	Технічне завдання
3	Проектування структури програмного забезпечення	4 тиждень	Структурна схема
4	Розроблення схеми функціонування програмного забезпечення та модулів	5 тиждень	Функціональна схема. Модулі
5	Розроблення програмного коду	6-8 тиждень	Текст програми
6	Тестування та відлагодження програмного забезпечення	9 тиждень	Працездатна програма
7	Розробка розділів пояснювальної записки	10-13 тиждень	Пояснювальна записка
8	Розробка графічного матеріалу	14 тиждень	Графічний матеріал
9	Захист магістерської роботи	16 тиждень	

Студент

(підпис)

Рязанов Д.А.

(прізвище та ініціали)

Керівник роботи

(підпис)

Каплієнко Т.І.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
124 с., 35 рис., 15 таблиць, 2 додатки, 33 джерел.

СТРУКТУРУВАННЯ, ТЕКСТОВА ІНФОРМАЦІЯ, АНАЛІЗ
КОНТЕНТУ, JS, TELEGRAM, BOT, TELEGRAF, EXCEL.

Об'єктом дослідження є процес аналізу для структуризації текстової інформації.

Предмет дослідження – методи аналізу для структуризації текстової інформації.

Мета дослідження – розробити інтерфейс на базі чат-бота, здатний приймати та перетворювати таблиці Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

Матеріали, методи та технічні засоби: функціональне та об'єктно-орієнтоване програмування, технології Telegram Bot API, node.js, мови програмування JavaScript, бібліотеки Telegraf, xlsx-parser, docx-parsers.

Результати. Розроблено програмне забезпечення, яке надає можливість аналізувати і структурувати текстові дані, також надає інтерфейс для користувача у вигляді чат бота, який обробляє запити і віддає відповідь.

Наукова новизна роботи полягає в запропонованому методі аналізу та структуруванню текстової інформації, а також у способі надання інтерфейсу.

Висновки. Проаналізовано основні принципи і методи, які використовуються в сучасних системах аналізу контенту. Виконано огляд програмних аналогів. Виконано проектування та реалізацію програмного забезпечення за допомогою мови програмування JavaScript. У процесі проектування створено діаграми прецедентів, структурну і функціональну схеми програмного забезпечення.

ABSTRACT

Explanatory note to the master thesis: 124 p., 35 fig, 15 tables, 2 supplements, 33 sources.

STRUCTURING, TEXT INFORMATION, CONTENT ANALYSIS, JS, TELEGRAM, BOT, TELEGRAF, EXCEL.

The object of study of the master's thesis is the process of analysis for structuring textual information.

The subject of research of the master's work - methods of analysis for structuring textual information.

The purpose of the study is to develop an interface based on a chatbot, capable of receiving and converting the tables of the Rules of admission to higher education institutions into a database of chatbots to provide information to applicants on the conditions of admission.

Materials, methods and technical means: functional and object-oriented programming, technologies: Telegram Bot API, node.js, JavaScript programming languages, libraries: Telegraf, xlsx-parser, docx-parses.

Results. Developed software that allows you to analyze and structure text data, also provides a user interface in the form of a chat bot that processes requests and responds.

The scientific novelty of the work lies in the proposed method of analysis and structuring of textual information, as well as in the method of providing the interface.

Conclusions. The basic principles and methods used in modern content analysis systems are analyzed. A review of software analogues has been performed. Design and implementation of software using JavaScript programming language. In the design process, precedent diagrams, structural and functional diagrams of the software were created.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	8
ВСТУП.....	9
1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ.....	10
1.1 Основні підходи до побудови системи аналізу даних	10
1.2 Огляд програмних засобів для аналізу контенту даних.....	14
1.1 Висновки за розділом 1 та постановка задачі дослідження	19
2 ОБҐРУНТУВАННЯ МЕТОДУ ДЛЯ ПОБУДОВИ СИСТЕМИ АНАЛІЗУ КОНТЕНТУ ДЛЯ СТРУКТУРИЗАЦІЇ ТЕКСТОВОЇ ІНФОРМАЦІЇ	21
2.1 Огляд типів аналізу контенту для текстової інформації.....	21
2.2 Вибір типу аналізу контенту для текстової інформації	24
2.3 Висновки за розділом 2	27
3 ПРОЄКТУВАННЯ СИСТЕМИ.....	28
3.1 Вибір середовища проєктування	28
3.2 Проєктування функціональних вимог до програмного забезпечення.....	31
3.3 Проєктування архітектури програмного забезпечення.....	33
3.4 Проєктування структури бази даних.....	35
3.5 Структурна схема програмного забезпечення	37
3.6 Висновки за розділом 3	39
4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ .	41
4.1 Алгоритм функціонування системи	41
4.2 Опис модулів реалізованої системи	43
4.3 Висновки за розділом 4	50
5 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ	52
5.1 Призначення програми	52
5.2 Умови виконання програми	52
5.3 Виконання програми.....	53
5.4 Висновки за розділом 5	63
6 ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА	64

6.1 Вступ.....	64
6.2 Планування розробки програмного продукту	67
6.3 Визначення витрат на розробку програми	69
6.4 Розрахунок капітальних вкладень	79
6.5 Оцінка економічної ефективності	81
7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ	83
7.1 Аналіз потенційних небезпек.....	83
7.2 Заходи з забезпечення безпеки	84
7.3 Заходи з виробничої санітарії та гігієни праці.....	89
7.4 Заходи безпеки у надзвичайних ситуаціях	91
7.4.2 Заходи з цивільного захисту	92
ВИСНОВКИ.....	96
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ.....	97
ДОДАТОК А Текст програми.....	102
ДОДАТОК Б Слайди презентації.....	117

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – База Даних;

API - Application Programming Interface;

JS – JavaScript;

NPM – Node Package Manager;

UX – User Experience

ВСТУП

Сучасне суспільство продукує безліч матеріальних та інформаційних одиниць, котрі поширюються, споживаються та відтворюються в соціальній структурі. Дослідження цих одиниць є надзвичайно цінним для розуміння явищ, що мають місце в суспільстві, а також змін, які відбулися або відбуваються.

На даний момент існує багато програмних засобів, які можуть використовуватися для аналізу даних. Частина з них застосовується для аналізу Big Data, частина – для окремих випадків, в яких використовуються парсер.

Також способів поглинання інформації з кожним днем стає все більше, той інтерфейс, який надає найприємніший UX і швидкий відгук, отримує найбільше уваги.

Необхідне рішення, яке надасть найзручніший спосіб аналізу даних і її виведення для звичайного користувача. Проблема полягає в тому, що не часто можна знайти такі рішення, які б поєднали ці ознаки. Тому в якості вирішення цієї задачі було розроблено чат-бота на основі месенджера Telegram.

В якості вихідних даних для аналізу використовуються текстові документи, а саме таблиці, а на виході буде структурована інформація, через чат-бота в Telegram.

Мета роботи - розробити програмне забезпечення для аналізу і структурування даних, яке буде обробляти текстову інформацію та буде надавати її у відповідь на запити користувача через чат-бота Telegram. Галуззю використання даного програмного забезпечення буде приймальна комісія Національного університет «Запорізька політехніка», практичною сферою застосування – перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Основні підходи до побудови системи аналізу даних

Існує два основних типи аналізу контенту: кількісний і якісний. Якщо якісний аналіз пов'язаний з фіксацією нетривіальних виразів, мовних інтонацій з розумінням цінності змісту повідомлення, кількісний аналіз спрямований на виявлення частоти окремих тем, слів або символів, що містяться в тексті.

Кількісний аналіз контенту повинен включати стандартизовані процедури обчислення вибраних категорій. Для формулювання висновків вирішальне значення мають кількісні значення, що характеризують категорію. Наприклад, якщо дослідник прагне зрозуміти, наскільки важливим є поняття "мирне врегулювання" для лідерів воюючих сторін, і погоджується з припущенням, що це грубо визначається частотою згадування цього поняття в офіційних промовах, то, після відповідних арифметичних обчислень він отримає певні кількісні показники. Показники можуть відрізнятися або, навпаки, бути близькими за абсолютною величиною, що буде враховано при інтерпретації результатів обробки. Завдання можна ускладнити, встановивши як передумову виділення всіх семантично значущих одиниць відповідних текстів, а потім обчисливши відносну важливість цього виразу в порівнянні з іншими. Примітно, що в обох випадках основною частиною розрахунків може бути виконана із застосуванням простих комп'ютерних програм.

Якісний аналіз контенту спрямований на глибоке змістовне вивчення текстового матеріалу, в тому числі з точки зору контексту, в якому представлені обрані категорії. Результати сформульовано тут з урахуванням взаємозв'язків елементів змісту та їх відносної значущості (рангу) у структурі тексту. Таким чином, для порівняння ставлення різних політиків до проблеми мирного врегулювання, дослідник повинен прагнути не лише

визначити відповідну концепцію, а й визначити, чи є її проблеми центральними для системи декларованих позицій, її конотацій, рівня деталей, емоційного забарвлення тощо. Залежно від цілей дослідження якісний аналіз змісту може бути доповнений деякими елементами кількісного аналізу контенту.

1.1.1 Етапи дослідження

Основи процедури аналізу контенту були розроблені Гарольдом Ласвелл. На сучасному етапі існує безліч думок про послідовність етапів аналізу контенту і ступенем їх важливості.

Виділяють такі стадії аналізу:

Крок 1. Підготовка програми аналізу документів (маються на увазі завдання, гіпотези, концепції, об'єкт аналізу і ін.). На цьому етапі, як правило, формулюється так звана емпірична теорія дослідження. Тобто, готуючись до аналізу контенту, вчений систематизує гіпотези, існуючі в контексті цієї проблеми, і відкидає ті, які не піддаються перевірці на масиві даних.

Крок 2. Вибір джерел аналізу. Необхідно визначити коло джерел, що містять матеріали по заданій темі. Далі важливо встановити додаткові умови відбору матеріалу: визначити тип джерела (телебачення, преса, рекламні матеріали, радіо і т. Д). Потім необхідно визначити тип повідомлення (журналістські статті в електронному або друкованому вигляді, інформаційні замітки, рекламні плакати або таблиці в контексті даного завдання), роль комунікаційного агента (відправник або одержувач повідомлення). Визначаються мінімальні і максимальні межі обсягу тексту, їх довжина, частота, час, місце і способи трансляції повідомлень цільової аудиторії. Існують і інші критерії відбору повідомлень, їх кількість і вибір варіюється в залежності від завдань дослідження.

Можна виділити два основні види документів для аналізу контенту:

- текстові або змішані документи (тексти з графіками, діаграмами, таблицями тощо), в яких аналізуються лише текстові матеріали з метою виявлення суттєвої інформації. Також ці джерела використовуються для стиснення текстів, щоб отримати основну ідею (тему) таких типів документів: наприклад, доповіді, анотації, тези, наукові праці тощо;

- змішані документи або табличні дані, які можна отримати з таких документів, як звіти, заяви та багато інших документів. Головною характеристикою цього типу документів є те, що це безліч табличних матеріалів, на основі яких формуються вторинні дані, які включаються до різних звітів та інших аналітичних матеріалів. Зазвичай такі дані супроводжуються текстовими матеріалами. [1].

На цьому кроці мають бути виконані наступні дії:

- Якщо знайдено, який тип даних необхідний для аналізу, можна перейти до процесу написання джерела даних;
- визначення кола і обсягу документів, які є носіями необхідної інформації (назва, періодичність публікації, період, тираж);
- побудова вибірки - які документи та за якими критеріями будуть аналізуватись;
- перевірка побудови вибіркової сукупності.

Крок 3. Визначення емпіричних моделей аналізу, вибірки (тобто відбору тіл зв'язку, відбору матеріалів за певні періоди часу, визначення типів повідомлень, типу вибірки).

У випадку обмеженої кількості матеріалів на дану тему, зразок може бути еквівалентний загальній. Класична інтерпретація методу аналізу змісту передбачає можливість зменшення вибірки повідомлень, коли вони подібні та однорідні. Таке зменшення є прийнятним, якщо чисельність загальної сукупності дуже велика. Вибірка при дослідженні великих наборів даних є випадковою. Звичайно, необхідно розрахувати його об'єм, щоб він залишався

репрезентативним, важливо визначити допустиму помилку вибірки. Слід додати, що часто обсяг вибірки визначається дослідниками на основі міркувань здорового глузду, доступності матеріалу, терміновості дослідження, а не обчислення допустимої похибки вибірки та репрезентативності масиву джерел. Розробка методики даного конкретного аналізу.

Можна виділити такі проміжні етапи:

- класифікація соціальних ситуацій відповідно до вивчених проблем;
- перевірка якості методики.
- визначення одиниць аналізу. Одиниці аналізу - структурні одиниці, які представлені в відношенні всього тексту і його окремих істотних частин [2].

Підставою для виділення одиниць аналізу може слугувати:

- цілі та завдання дослідження;
- особливості конкретного об'єкта аналізу, тобто документа.

Одиницями аналізу можуть бути:

- окремі слова, фрази, терміни. Наприклад, економічні поняття: ринок, зайнятість, приватизація, управління тощо; політичні: демократія, референдум, вибори, влада тощо;
- Теми, повідомлення, виражені в семантичних зображеннях, статтях, частинах тексту, що дають більш повний опис змісту документа, ніж попередні одиниці аналізу. Так, теми, що впливають на міжнародну ситуацію в Україні, перехід до ринкових відносин, боротьбу зі злочинністю, спосіб життя та здоров'я нації тощо.
- Прізвища історичних діячів, політиків, видатних учених та митців, представників різних соціальних спільнот, є певним узагальненим типом фігури, що має певні соціальні риси. Їх аналіз дає дослідникові важливу інформацію про досліджувану історичну епоху, домінування

конкретних політичних, соціально-економічних ідей, вплив певних діячів на формування громадської думки тощо. До цієї групи належать згадки про організації, установи та ін. інші соціальні інститути.

– судження, закінчена думка, логічний ланцюжок. Це найскладніші одиниці аналізу, так як вони мають високий ступінь конструктивності. Їх структура більш диференційована, ніж в інших одиницях аналізу, і містить кілька елементів.

Крок 5. Пілотне дослідження, перевірка надійності методу.

Крок 6. Збір первинної емпіричної інформації.

Крок 7. Кількісна обробка зібраних даних.

Крок 8. Інтерпретація здобутих результатів, висновки дослідження.

1.2 Огляд програмних засобів для аналізу контенту даних

1.2.1 Intelligent Miner for text (IBM)

Intelligent Miner for text (IBM) - це набір окремих утиліт, які незалежно запускаються з командного рядка або сценаріїв. Ця система є одним з кращих інструментів для глибокого аналізу тексту і включає наступні основні утиліти:

– Language identification tool, Categorization tool - утиліта класифікації, що дозволяє автоматично розподіляти текст по декількох категоріях (вхідна інформація на етапі навчання цього інструменту може бути результатом роботи наступної утиліти - Clustering Tool);

– Cauterization tool - утиліта кластеризації, дозволяє розбивати велику кількість документів на групи по близькості стилю, форми, різних частотних характеристиках виявлених ключових слів.

- Feature Extraction Tool - утиліта для виявлення нових ключових слів в документі (імена власні, заголовки, скорочення) на основі аналізу визначеного словника;
- Annotation tool — утиліта для «визначення сенсу» текстів і створення рефератів, тобто анотацій до вихідних текстів.

Intelligent Miner for text (IBM) об'єднує в собі великий і значний набір інструментів, які в основному займаються пошуком інформації - в чому специфіка програми [3] (рис. 1.1).

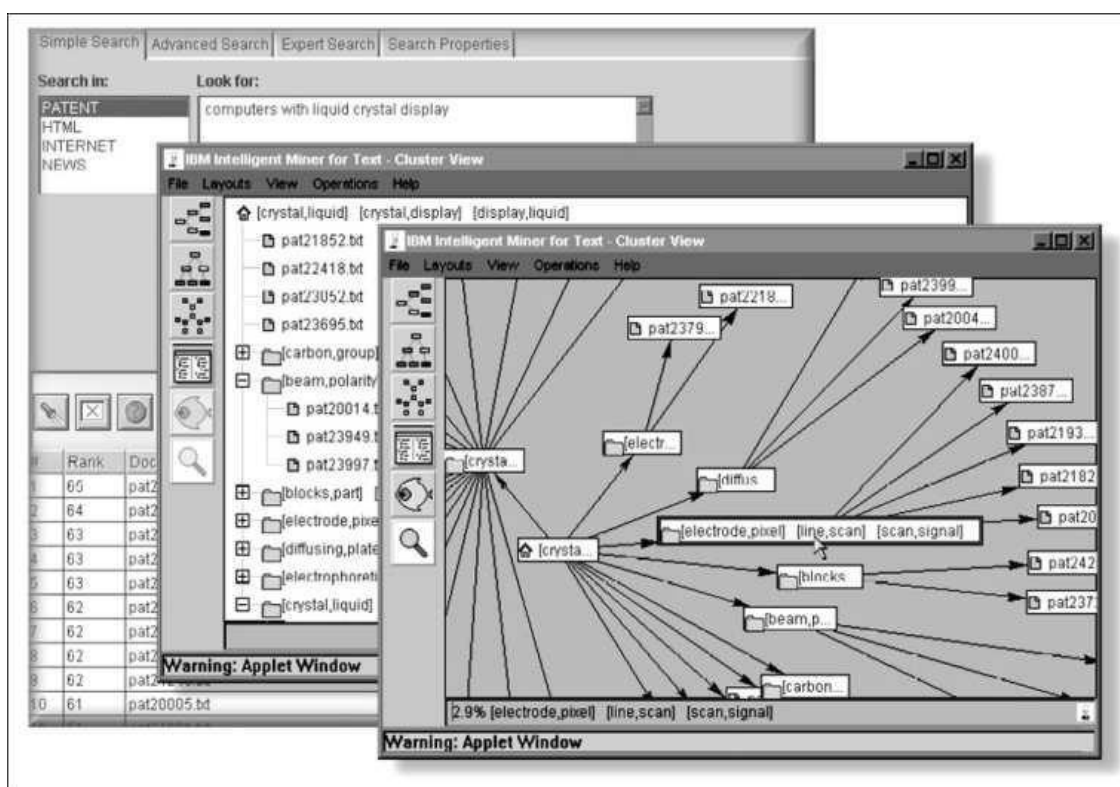


Рисунок 1.1 – Інтерфейс Intelligent Miner for text (IBM) [3]

1.2.2 SemioMap

SemioMap - це продукт компанії Entrieva, відкритий у 1996 році. Учень семіотики Клод Фогель. У трав 1998 р. продукт був випущений як

промисловий комплекс SemioMap 2.0 - система першого текстового майнінгу, що працює в архітектурі клієнт-сервер [3] (рис. 1.2).

Характеристики програми:

а) складається з двох основних компонентів: сервера SemioMap та клієнта SemioMap;

б) робота програми проходить наступні три фази:

1) індексування - сервер SemioMap автоматично читає масиви неструктурованого тексту, визначає ключові фрази і становить з них індекс;

2) кластеризація - сервер SemioMap виявляє взаємозв'язку між певними поняттями і будує лексичну структуру на основі того, як ці вирази зустрічаються в тексті;

3) графічне відображення і навігація - візуалізація карт посилок, що забезпечує швидку навігацію за ключовими фразами і посиленнями, які виникають між ними, а також можливість швидкого доступу до конкретних документів;

в) сортує матеріали по папках, створюючи окремі бази даних для різних папок;

г) відносини між поняттями, виявленими програмою, засновані на тому, як часто слова/фрази зустрічаються разом в абзацах вихідного текстового масиву.

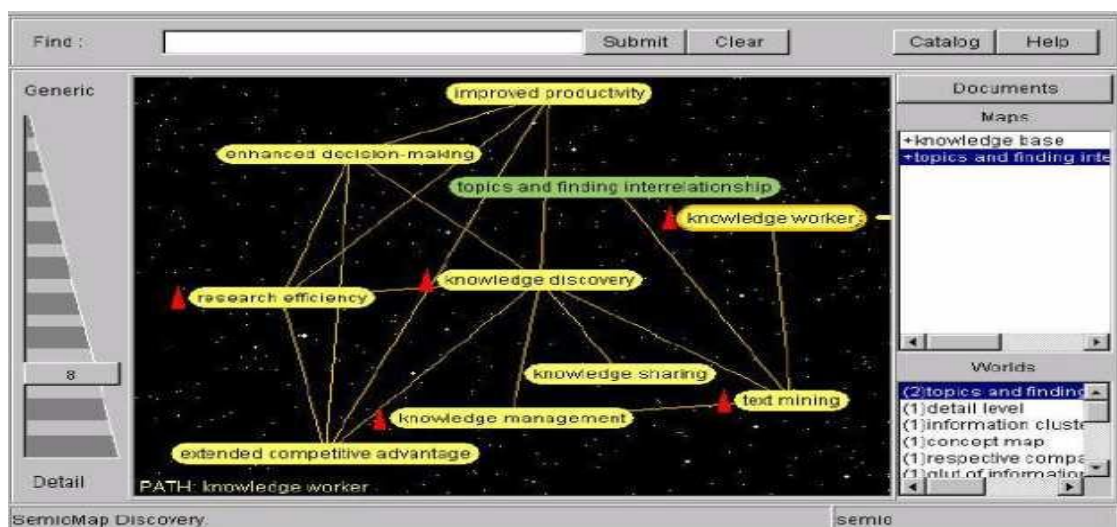


Рисунок 1.2 – Інтерфейс SemioMap [3]

1.2.3 Oracle Text

Oracle Text (рис. 1.3) дозволяє ефективно працювати з запитами, що відносяться до неструктурованих текстів.

Основне завдання програми - пошук документів за їх змістом, словами і словосполученнями, які при необхідності комбінуються з використанням логічних операцій. Результати пошуку ранжуються за релевантністю з урахуванням частоти повторення даного дослідника слова в знайдених документах. Для підвищення ефективності пошуку за запитом в програмі є певні розширення, які розділені на три групи:

- дозвіл слів в запиті по всім морфологічним формам, яке реалізується за рахунок знання морфології мови;
- Oracle Text дозволяє розширити слова в запиті близькими за змістом словами, підключивши тезаурус і розширивши запит слів, близьких за написанням і звучанням - нечіткий пошук і пошук приголосних слів;
- Нечіткий пошук слід застосовувати для пошуку слів з помилками, що виникли при написанні, і у випадках, коли є сумніви в правильності написання - прізвищ, імен власних і т.д.

Характеристики програми:

- забезпечує тематичний аналіз англійською мовою (для такого аналізу масивів російського тексту окремо була надана програма Russian Context Optimiser - RCO);
- програма аналізує текстові масиви через лінгвістичну і аналітичну призму, в результаті чого аналізує основні теми тексту і будуються їх тематичне резюме і анотації. [3].

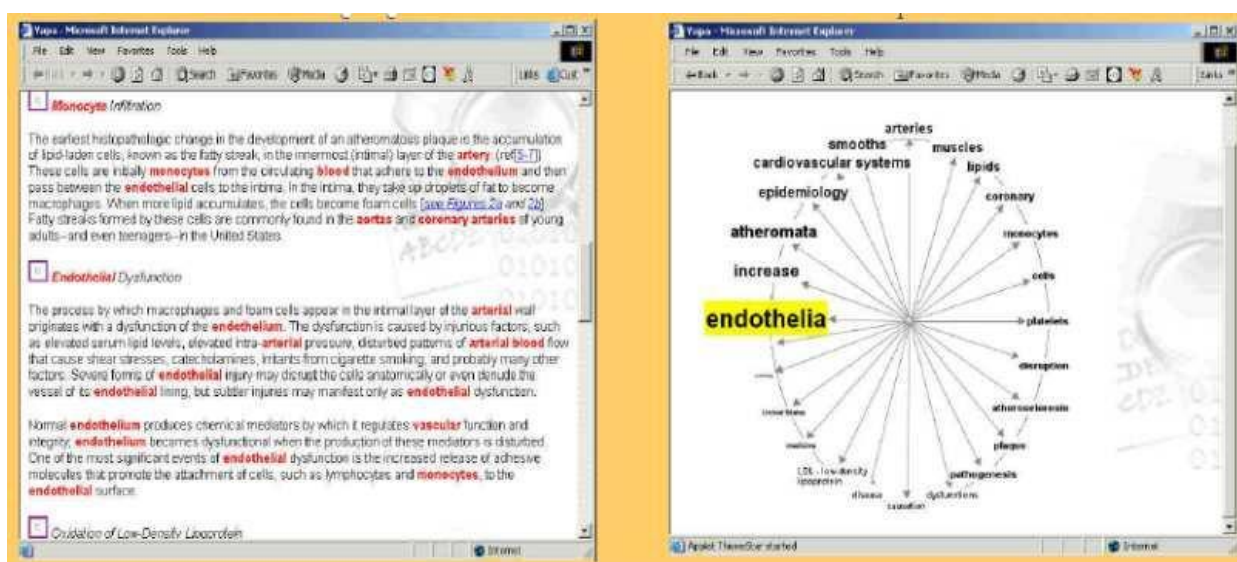


Рисунок 1.3 – Інтерфейс Oracle Text [3]

1.2.4 Knowledge Server (Autonomy)

Архітектура IDOL (Intelligent Data Operating Layer) сервера Autonomy (рис. 1.4), відома своїми розробками в області статистичного аналізу контенту, поєднує в собі інтелектуальний аналіз шаблонів зі складними методами аналізу контексту і вилучення значень для вирішення завдань автоматичної класифікації. і зшивання. .

Головна перевага системи Autonomy - інтелектуальні алгоритми, засновані на статистичній обробці. Ці алгоритми засновані на теорії інформації Шеннона, байєсівської ймовірності і нейронних мережах.

Autonomy включає такі основні можливості:

- автоматична класифікація;
- кластеризація;
- генерація таксонометричних дерев;
- автоматичне проставлення гіперпосилань;
- автоматичне створення профілів (інформаційних портретів);
- створення і маніпулювання метаданими;
- персоналізація;
- інтелектуальна обробка XML-даних;
- пошук.

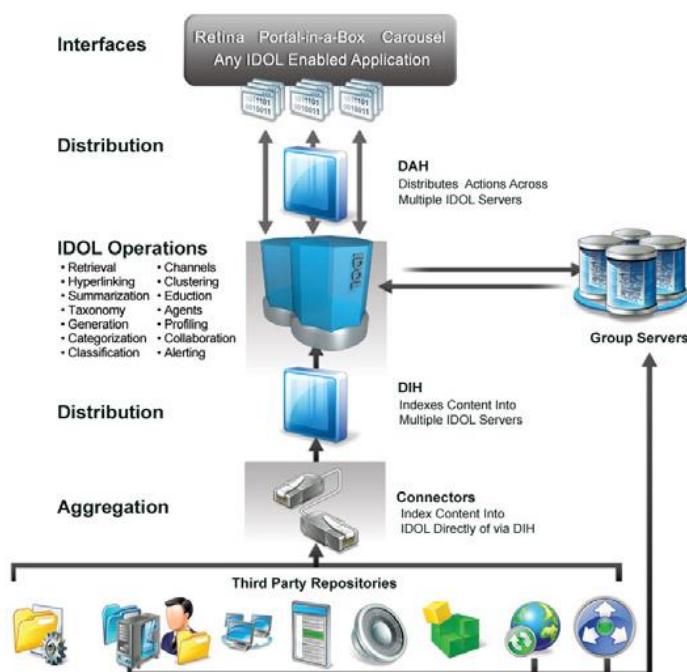


Рисунок 1.4 – Knowledge Server (Autonomy) [3]

1.1 Висновки за розділом 1 та постановка задачі дослідження

Виконано аналіз основних підходів до побудови систем аналізу контенту, розглянуто два основних типи аналізу контенту - кількісний і якісний. Розглянуто основні стадії контент аналізу.

У якості програмних засобів для аналізу контенту розглянуто Intelligent Miner for text (IBM), SemioMap, Oracle Text, Autonomy Knowledge Server.

Для кожного програмного засобу проаналізовано його функціональні можливості і особливості.

У зв'язку з тим, що проаналізовані програмні засоби є важкими до освоєння, мають певні обмеження з середою виконання, а головне не надають можливості виконати завдання по обробці таблиць до Правил прийому до закладів вищої освіти з занесенням у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

Відповідно, метою даної роботи є розроблення програмного забезпечення, яке надасть можливість аналізувати і структурувати текстові дані, також інтерфейс для користувача у вигляді чат-бота, який обробляє запити і повертає відповідь.

Для досягнення поставленої мети необхідно розв'язати наступні завдання:

- дослідження методів контент аналізу;
- розробка технічного завдання;
- вибір засобів розробки;
- проєктування структури програмного забезпечення;
- розробка програмного забезпечення для аналізу контенту та його структурування;
- тестування та відлагодження програми;
- економічне обґрунтування доцільності розробки.

2 ОБГРУНТУВАННЯ МЕТОДУ ДЛЯ ПОБУДОВИ СИСТЕМИ АНАЛІЗУ КОНТЕНТУ ДЛЯ СТРУКТУРИЗАЦІЇ ТЕКСТОВОЇ ІНФОРМАЦІЇ

2.1 Огляд типів аналізу контенту для текстової інформації

Одним з відомих методів аналізу текстової інформації є аналіз контенту, стандартний метод дослідження в соціальних науках, предметом якого є аналіз змісту текстових файлів і комунікативних продуктів (наприклад, коментарів, форумів, електронних листів, статей, і т.д.). Аналіз контенту починався як кількісно орієнтований метод аналізу тексту для дослідження масових комунікацій [4]. Вперше він був використаний в 1910 році соціологом Максом Вебером для оцінки освітлення в пресі політичних мітингів в Німеччині. Американський дослідник комунікацій Гарольд Лассуелл (Harold Lasswell) 30-40 років використовував подібну техніку для отримання змісту пропагандистських повідомлень під час війни. З появою засобів автоматизації електронного тексту з 1960-х років в оригінальному інституціональному репозиторії Львівського політехнічного національного університету був розроблений аналіз змісту великих обсягів інформації - баз даних та інтерактивних медіа-джерел. Традиційне політичне використання сучасних інформаційних технологій для аналіз контенту було доповнено необмеженим списком заголовків, що охоплюють промислову і соціальну сфери, бізнес і фінанси, культуру і науку. Цей процес супроводжувався великою кількістю різних програмних пакетів. Поняття аналіз контенту не має однозначного визначення [5-8], що створює проблему: системи, засновані на різних підходах до аналіз контенту, зазвичай несумісні (табл. 2.1).

Таблиця 2.1 – Неоднозначні визначення поняття аналізу контенту різними авторами

Автор	Визначення
Д. Джері, ДЖ. Джері	методи об'єктивного якісного і систематичного вивчення змісту засобів комунікації
Д. Мангейм, Р. Рич	систематична кількісна обробка, оцінка та інтерпретація форми та змісту джерела інформації
В. Іванов	якісний і кількісний метод дослідження документів (характеризується об'єктивністю висновків і строгістю процедури) і кількісна оцінка тексту з подальшою інтерпретацією результатів; предмет дослідження - проблеми соціальної реальності, виражені і приховані в документах, і внутрішні закономірності об'єкта дослідження.
Б. Краснов	складається з пошуку в тексті певних значущих понять (одиниць аналізу), визначення частоти їх появи і взаємозв'язку зі змістом всього документа
Е. Таршис	методика дослідження для отримання результатів шляхом аналізу змісту тексту про стан і властивості соціальної реальності

Виділяють такі основні складові контент-аналітичного дослідження (табл. 2.2).

Таблиця 2.2 – Основні складові контент-аналітичного дослідження

Назва	Властивості основних складових контент-аналітичного дослідження
Спостереження	розробка масового набору текстів з використанням типових соціологічних процедур безперервного / вибіркового спостереження з дотриманням вимог репрезентативності
Структурування	допущення структурування, сегментації, розчленування текстів чи виділення значущих інваріантів (повторення у всіх / деяких текстах) в досліджуваній масовій популяції.
Формалізація	забезпечення однаковості сегментації і вибору інваріантів, застосування високого ступеня формалізації, строгих правил роботи і формальних алгоритмів в аналітичних процедурах.
Реферування	формалізоване поділ цілісних текстів чи відбір їх окремих елементів для подальшого збору з використанням аналітико-синтетичної процедури.
Аналіз	використання методів теорії ймовірностей і математичної статистики для обробки текстів.

Таким чином, аналіз контенту - це кількісний і якісний аналіз текстової інформації і текстових масивів з метою подальшої змістовної інтерпретації отриманих кількісних і якісних закономірностей. Аналіз контенту використовується для вивчення джерел, які не зазнали змін за структурою/змістом, але існують у вигляді несистематизованого, довільно організованого текстового матеріалу. Метод аналізу контенту полягає в формуванні абстрактної моделі текстового контенту з різноманітного текстового матеріалу.

Існує два типи методів аналізу контенту: кількісний і якісний (таблиця 2.3).

Таблиця 2.3 – Основні складові контент-аналітичного дослідження

Назва	Кількісний (змістовний)	Якісний (структурний)
Визначення	дослідження слів, тем і повідомлень, які фокусується на утриманні контенту	дослідження, яке досліджує не зміст контенту, а його форму і структуру
Приклад	в якості першого кроку дослідник повинен створити словник, в якому кожне спостереження буде визначено і класифіковано	визначити період часу або кількість місця для друку, яке призначене темі в конкретне джерело, або скільки слів або стовпців призначено кожній темі у відповідній категорії
Особливість	перед аналізом обраних мовних одиниць спрогнозуйте їх зміст (створення словника) і визначте кожен можливий результат спостереження відповідно до очікувань дослідника	розрахувати питомі ваги P кожної теми і категорії ($P = R / T$, R - кількість одиниць даної категорії, T - загальна кількість одиниць) і провести порівняльний аналіз актуальних тем для подальшого прогнозування подій, процесів

Отже, серед двох виділених типів аналізу контенту, які наведено в таблиці 2.3, для вирішення поставленої задачі, а саме перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу підходить кількісний тип аналізу контенту.

2.2 Вибір типу аналізу контенту для текстової інформації

Існують суворі вимоги до методу аналізу контенту:

- об'єктивність, тобто проводиться аналіз повинен підкорятися ясним і точним правилами;
- систематичність, тобто зміст аналізованого матеріалу має бути

упорядковано і інтегровано в категоріях;

- вимірність, тобто можливість підрахунку частоти народження значущих елементів.

В цілому метод аналізу контенту розглядається дослідниками, перш за все, як кількісний метод дослідження. Але в літературі з'являється поняття «якісний аналіз контенту». Так А.Л. Джордж вказує, що «якісний аналіз враховує наявність або відсутність характеристики, а кількісний розглядає частоту окремих тем, слів, символів, що містяться в тексті» [9]. При цьому наголошується, що «контентний аналіз - досить складна процедура. Він вимагає, так само як інтуїції і уяви, багато часу для з'ясування того, що є якісно важливим, і вибору потрібних категорій» [9]. Але при всьому при цьому, аналіз контенту залишається кількісною процедурою, що замінює інтуїцію дослідника більш точними даними. Цінність даного методу залежить від якості концептуалізації, проведеного дослідником до початку проведення аналізу і від тієї точності, з якою поняття переводяться в змінні.

Специфіка даного методу полягає в тому, що дослідник «Переводить» вербальну інформацію в більш «об'єктивну» невербальну форму (перш за все, кількісну); в дослідженні відбувається сходження від тексту до нетекстової реальності (соціальної Насправді в усьому її різноманітті). Кількісний аналіз контенту - це строгий науковий метод, що передбачає систематичну і надійну фіксацію певних елементів змісту тексту з подальшої квантифікації отриманих даних.

Слід зазначити, що у вітчизняній традиції аналіз контенту визначається як кількісний аналіз текстів з метою подальшої змістовної інтерпретації виявлених числових закономірностей. Сенса аналізу контенту як дослідницького методу полягає в сходженні від різноманіття текстового матеріалу до абстрактної моделі змісту тексту. Перевага тут віддається стандартизованим процедурам, які дозволяють швидко і надійно отримати кількісні показники.

При аналізі контенту відбувається робота з символічним матеріалом (текстом). Як текст може розглядатися різний словесний матеріал, невербальні прояви людини, картини і т.п. Особливе значення має систематичність досліджуваних явищ, аналіз підпорядковується певними правилами. Хороший аналіз контенту відштовхується від теорії і його предметом є аналіз комунікації. При цьому аналізу піддається тільки зафіксована комунікація.

Якісний аналіз контенту, на відміну від кількісного аналізу, - орієнтований на ідеографічний підхід в науці і спирається на індуктивний метод отримання знань, що підкреслює одиничність досліджуваних явищ, їх неоднозначність і комплексність. Основна відмінність якісного аналізу контенту від кількісного полягає в тому, що кількісний аналіз орієнтований на пояснення змісту, на загальні принципи аналізу матеріалу, на пошук загального через аналіз, розкладання тексту на складові його частини і аналіз цих змінних. Якісний аналіз контенту спрямований на розуміння досліджуваних явищ; на аналіз взаємозв'язків і процесів між цими явищами; він орієнтований на охоплення всій сукупності і складності досліджуваних явищ і спрямований на вивчення одиничних випадків.

Завдання якісного аналізу контенту полягають у формулюванні гіпотез, створення нових теорій, в поглибленні досліджуваного матеріалу, класифікації та перевірці цих гіпотез і теорій. Німецький дослідник Ф. Мейрінген виділяє 5 областей застосування якісного аналізу контенту:

- комунікація (власне аналіз контенту);
- герменевтика (як мистецтво інтерпретації);
- якісні соціальні дослідження (інтерпретативна парадигма);
- літературознавство і психологія [10].

Ці галузі використання якісного аналізу контенту орієнтують дослідника на розуміння мовного матеріалу і припускають:

- необхідність систематизації досліджуваних явищ;
- необхідність моделі комунікації;
- виділення категорій аналізу;
- приділення особливої уваги прихованим сенсам;
- орієнтацію на повсякденні процеси розуміння і інтерпретації;
- можливість бачення ситуації з протилежного перспективи;
- орієнтацію на семіотику і прагматичну теорію значень;
- особливі правила інтерпретації структурного текстового аналізу.

2.3 Висновки за розділом 2

Виконано аналіз існуючих методів, які використовуються для побудови системи контент аналізу.

Виконано вибір кількісного типу для побудови системи контент аналізу, так як він має ближчі до умов поставленої задачі характеристики (табл 2.3).

3 ПРОЄКТУВАННЯ СИСТЕМИ

3.1 Вибір середовища проєктування

Під час вибору середовища проєктування і реалізації потрібно враховувати основні фактори, пов'язані з розробленням програмного забезпечення, що має бути створено в кваліфікаційній роботі магістра:

- необхідність використання методу кількісного контент аналізу для реалізації запропонованого в роботі методу та супутніх засобів наукових обчислень;

- необхідність розроблення чат-боту, оскільки він є найзручнішим на даний момент засобом реалізації мережевого доступу для широкого кола різних користувачів.

Основними засобами реалізації чат-ботів загалом та таких, що мають забезпечувати загальну функціональність програмного забезпечення, на даний момент є безліч мов програмування.

При цьому одночасно на загальний вибір дуже впливає другий фактор, оскільки він на даному етапі розвитку має передбачати наявність якісних і документованих засобів роботи з Application Programming Interface (API) Telegram.

На даний момент бібліотека Telegraf забезпечує повну підтримку функціональності саме для мови програмування JavaScript, яка виконується в середі Node.js, що є важливим фактором.

Так як з огляду на перший фактор, виконавча node.js має величезну перевагу за рахунок свого пакетного менеджера npm, який надає необхідні засоби для реалізації методу якісного контент аналізу. Такі бібліотеки як `xlsx-parser` і `docx-parser` допоможуть конвертувати дані до формату JSON.

Тож загалом, зважуючи перший та другий фактори, потрібно визначити, що виходячи з них, найоптимальнішим вибором для розв'язання даної задачі є мова JavaScript у середі виконання node.js. Проте необхідно

зважити її інші характеристики, щоб підтвердити або спростувати даний вибір.

До основних переваг загалом мови JavaScript можна віднести наступні [11]:

- багата екосистема - за замовчуванням, npm є менеджером пакетів Node.js, він також служить майданчиком для інструментів JavaScript з відкритим кодом, що відіграє важливу роль у просуванні цієї технології. Наразі в реєстрі npm доступно близько 836 000 бібліотек і щотижня публікується понад 10 000 нових, екосистема Node.js досить багата. Та сама статистика вказує, що 97 відсотків сучасних вебзастосунків складаються з модулів npm. І це є доказом його безперечної популярності серед розробників. Завдяки такій безлічі безкоштовних інструментів, доступних за кілька кліків, існує величезний потенціал для використання Node.js. У той же час програмне забезпечення з відкритим кодом користується зростаючою популярністю, оскільки дозволяє створювати нові рішення, зменшуючи загальні витрати на розробку та час виходу на ринок;

- безшовна підтримка JSON - хоча інші Back End технології, такі як PHP та Ruby on Rails, можуть використовувати формат JSON для зв'язку, Node.js робить це без перетворення між бінарними моделями та використовує JavaScript. Це особливо зручно, коли потрібно створити RESTful API для підтримки баз даних NoSQL, наприклад MongoDB. Це безперервне спілкування з одним із основних стандартів передачі даних - ще одна перевага екосистеми JavaScript;

- потужна корпоративна підтримка - у 2015 році було створено Фондом Node.js, щоб «забезпечити широке впровадження та сприяти пришвидшенню розвитку Node.js.» Основними членами організації стали IBM, Microsoft, PayPal, Fidelity та SAP. Список організацій, які використовують Node.js у виробництві, постійно збільшується. На даний момент до її складу входять майже триста відомих компаній, таких як PayPal,

Medium, Trello, Uber та Zendesk. Дуже мало проєктів з відкритим кодом колись користувалися такою потужною підтримкою провідних світових компаній. І це передбачає, що Node.js має надзвичайний потенціал.

Вище було перелічено основні бібліотеки, які можуть використовуватися для реалізації методу якісного контент аналізу та роботи з Telegram API і основою роботи яких є, як було видно, мова програмування JavaScript.

Серед великої кількості бібліотек, для написання чат-бота було обрано Telegraf, так як він має наступні переваги:

- Telegraf – це фреймворк з відкритим вихідним кодом, призначений насамперед для надання інтерфейсу програмісту для роботи з Telegram API;
- фреймворк включає себе велику множину готових методів для роботи з API які під капотом виконують безліч рутинної роботи для програміста;
- фреймворк надає програмісту готовий механізм для роботи з сокетом;
- наявність якісної документації.

Таким чином, для програмної реалізації чат-боту (з функцією аналізу та структуризації контенту) запропоновано використовувати наступний набір рішень:

- мова програмування JavaScript;
- середовище виконання Node.js;
- пакетний менеджер NPM;
- бібліотека для написання боту Telegraf.

3.2 Проектування функціональних вимог до програмного забезпечення

В якості мети роботи було обрано розроблення програмного забезпечення для аналізу та структуризації контенту для підвищення зручності пошуку нової інформації та автоматизації роботи.

Для визначення функціональних вимог було також використано результати огляду програмних аналогів, представлені в розділі 1.

Сформований в результаті набір функціональних вимог до програмного забезпечення аналізу та структуризації контенту включає наступні:

- аутентифікація користувача;
- додавання документів, які будуть реалізовуватися;
- можливість переглядати документи;
- можливість редагувати документи;
- переглядати актуальну статистику;
- надання прав на адміністрування користувачам;
- перегляд існуючої бази даних (БД).

На рис. 3.1 виділено варіанти використання системи звичайним користувачем. Таким звичайним користувачем загалом є користувач месенджеру Telegram, тобто він не повинен мати ніякого додаткового облікового запису, потрібно тільки знайти бота через пошук і по ідентифікатору користувачу надається необхідний рівень можливостей.

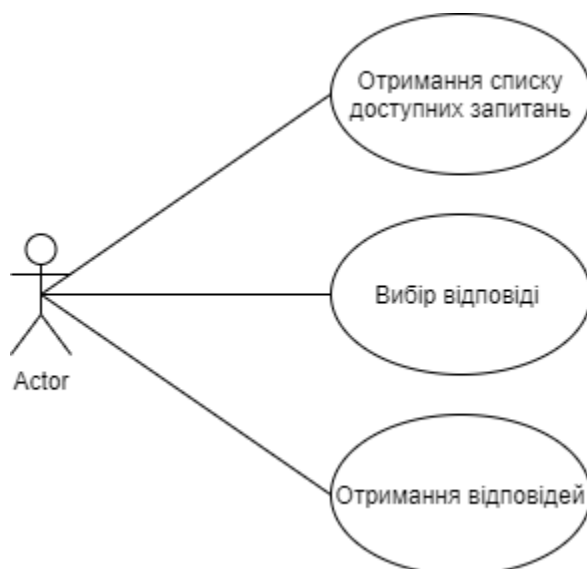


Рисунок 3.1 – Діаграма прецедентів для роботи користувача з системою

На рис. 3.2 відповідно виділено варіанти використання системи адміністратором. Адміністратор забезпечує наповнення системи даними, з якими взаємодіє в подальшому звичайний користувач.

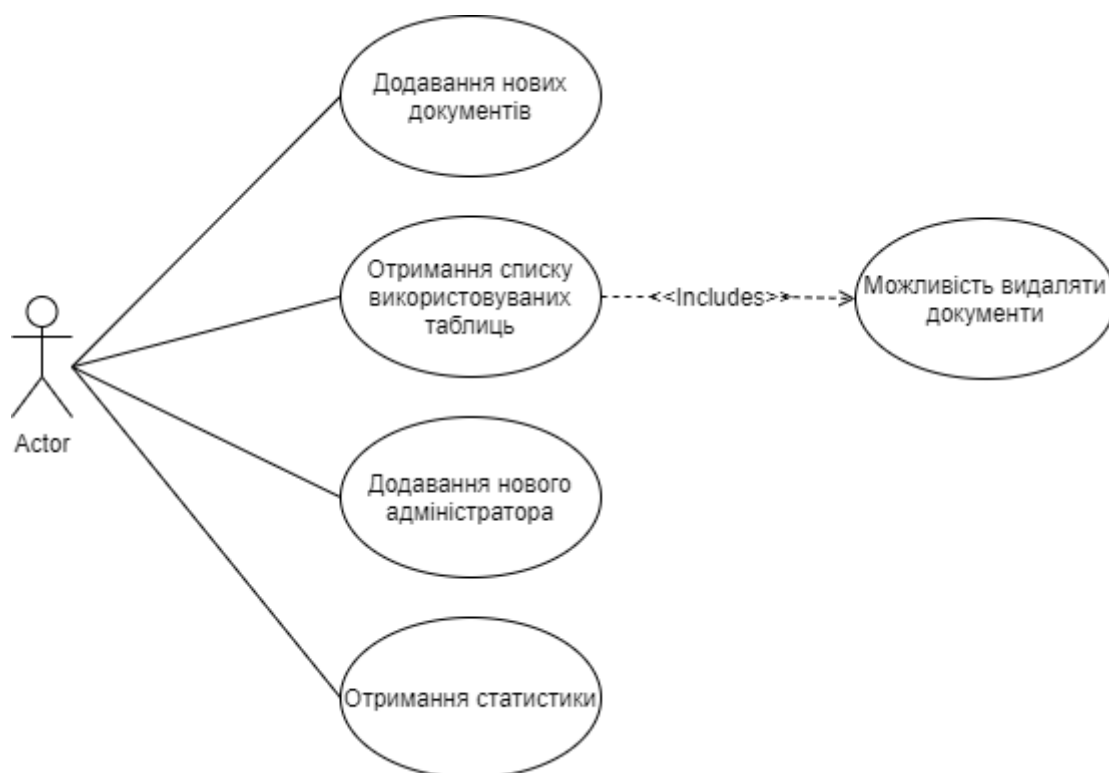


Рисунок 3.2 – Діаграма прецедентів для роботи адміністратора з системою

3.3 Проєктування архітектури програмного забезпечення

Оскільки програмне забезпечення для аналізу і структуризації контенту має бути представлено у вигляді чат-бота, в якості платформи, на якій даний бот буде працювати є Telegram. Telegram є одним з найпопулярніших месенджерів в даний момент у всьому СНД просторі. Платформа є дуже оптимізованою, крос-платформною, а також має досить хороший рівень безпеки для особистих даних користувачів. Все це надає хорошу інфраструктуру для користувачів, які будуть користуватися програмним забезпеченням для аналізу та структуризації контенту. Telegram бере на себе всі рутинні складності у вигляді аутентифікації і безпеки користувачів. Залишається тільки скористатися готовим Bot API від Telegram для написання програмного забезпечення для аналізу і структуризації контенту.

Telegram Bot API реалізує RESTful API - це архітектурний стиль для інтерфейсу прикладних програм (API), який використовує HTTP-запити для доступу і використання даних. Ці дані можуть використовуватися для типів запитів GET, PUT, POST і DELETE, що відноситься до читання, оновлення, створення і видалення операцій, що стосуються ресурсів. Також для отримання команд в режимі реального часу використовуються Telegram використовує AJAX long-polling.

AJAX long-polling - це найпростіший спосіб підтримувати постійне з'єднання з сервером, не використовуючи при цьому ніяких специфічних протоколів (типу WebSocket або Server Sent Events).

Для того, щоб не писати безліч методів і класів, організовують з'єднання між сервером розробленого програмного забезпечення і Telegram Bot API, в рамках вирішення даного завдання, буде використовуватися бібліотека Telegraf, написана на мові JavaScript. Бібліотека являє обгортки

для стандартних запитів до Telegram Bot API, що значно скорочує час на розробку.

На рис 3.3 наведено архітектуру проєкту чат-бота

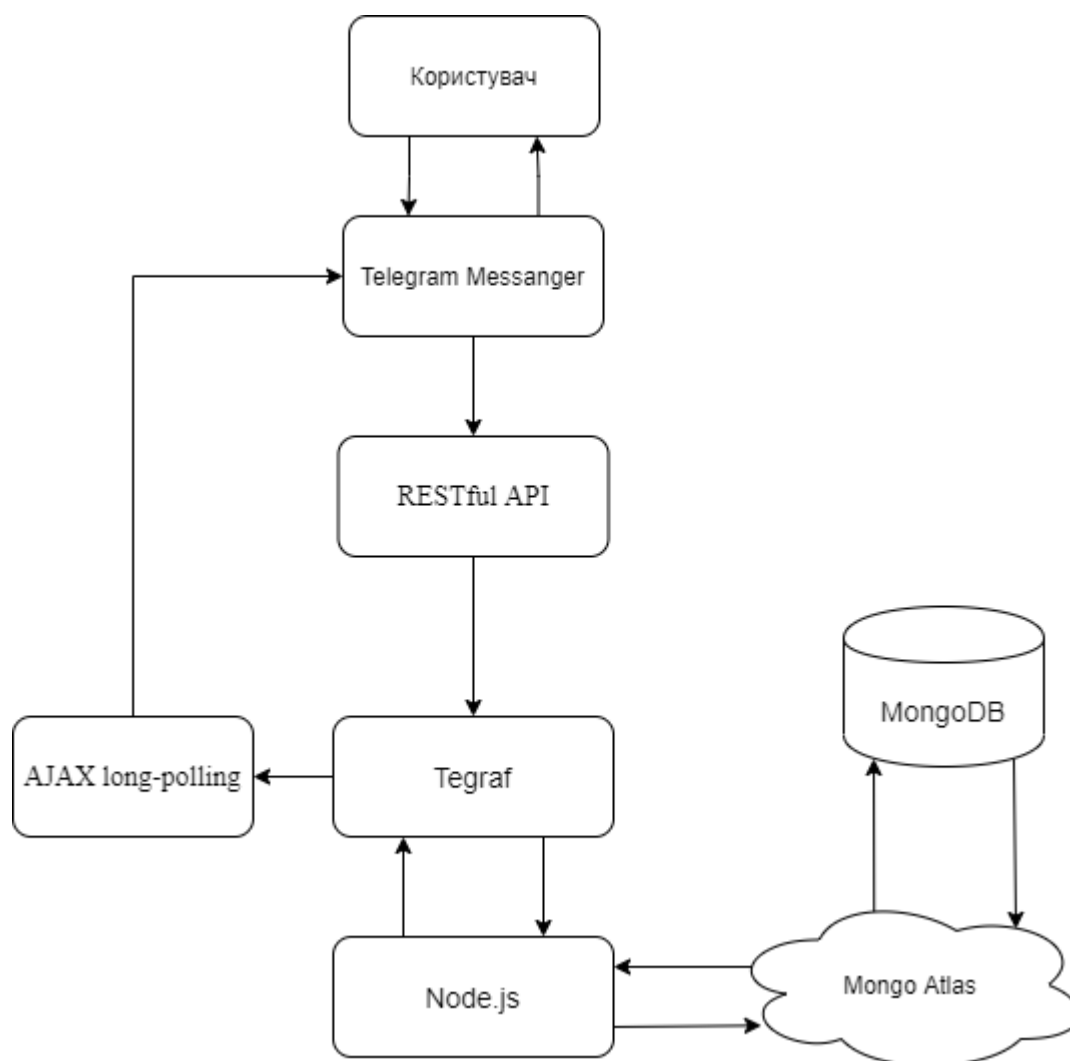


Рисунок 3.3 – Архітектура чат-бота

Таким чином, архітектура чат-бота, розробленого на основі використаних технологій Node.js та Tegram, складається з 4 наступних компонентів:

- Telegram Messenger - інтерфейс, через який користувач спілкується з Bot API;

- Bot API - логічний рівень, реалізований компанією Telegram для надання методів;
- BE Stack - логічний рівень, в якому реалізуються основні методи для вирішення поставленого завдання;
- база даних - логічний рівень, в якому обробляються і зберігаються дані користувачів.

3.4 Проєктування структури бази даних

3.4.1 Аналіз и вибір системи управління базою даних

Для повної підтримки роботи програмного забезпечення для аналізу і структурування контенту, необхідно забезпечити місце для зберігання і обробки даних користувачів. В даній реалізації програмного забезпечення використовується два основних способи роботи зі зберіганням даних.

Перший і найпростіший, але тим не менш дуже важливий механізм для роботи з даними користувача, є зберігання актуальної сесії користувача. Дані, які зберігаються в сесії не мають відношення до конкретного користувача, а мають відношення більше до актуального стану застосунку, для того щоб поліпшити UX і в якихось місцях програмного забезпечення для аналізу і структурування контенту полегшити та скоротити час на розробку для програміста. Припустимо, це може бути історія переміщень за застосунком, або зберігання якихось інших станів, які актуальні тільки в даний момент роботи програми. Як приклад даної реалізації програмного забезпечення для аналізу і структурування контенту, сесія використовується як місце, в якому можна зберегти попередню сцену застосунку. Так виглядає ініціалізація сесії на момент старту програми:

```
bot.use(session());
bot.use( async (ctx, next) => {
```

```

    ctx.session.previousScene = "";
    await next()
  });

```

Другий спосіб зберігати і обробляти дані потрібен для тривалого зберігання інформації, для цього використовуються бази даних.

Є два основні підходи для роботи з базами даних на даний момент:

- SQL - декларативна мова програмування, яка застосовується для створення, модифікації та управління даними в реляційній базі даних, керованої відповідною системою управління базами даних (СУБД);
- NoSQL - це підхід до реалізації масштабується сховища (бази) інформації з гнучкою моделлю даних, що відрізняється від класичних реляційних СУБД.

Вибір, який з підходів вибрати, стає очевидним, якщо зрозуміти, що необхідно від застосунку, який повинно вирішувати вищепоставлені завдання. NoSQL-бази оптимізовані для застосунків, які повинні працювати швидко, з низькою часовою затримкою (low latency) обробляти великий обсяг даних з різною структурою.

NoSQL-бази бувають також різні. В основному прийнято їх розділяти на 4 типи [12]:

- ключ-значення (Key-value);
- документо-орієнтоване сховище;
- колоночне сховище;
- графове сховище.

Серед наведених типів, для вирішення поставленої задачі найбільше підходить документо-орієнтоване сховище на базі MongoDB, так як у своїй основі він використовує JSON. Раніше вже наводилася інформація про те, що Node.js підтримує безшовну роботу з JSON.

MongoDB надає програмісту безкоштовну можливість створити кластери віддалено на хмарі використовуючи Mongo Atlas, що скорочує час на розгортання і підтримки БД, так програміст отримує готовий набір інструментів для налагодження роботи БД.

Нижче наведені всі колекції, які використовуються застосунком для його функціонування:

- admins - колекція користувачів, які мають права на маніпулювання даними;
- questions – колекція, яка містить в собі список поширених запитань, на які застосунок може дати відповідь, аналізуючи завантажені в нього документи;
- answers – колекція, яка містить в собі список відповідей на поширені питання;
- owners - колекція користувачів, які мають права на маніпулювання адміністраторами;
- stats – колекція, яка містить в собі статистику про користувачів, що користуються чат-ботом;
- tables - колекція таблиць, які використовуються в даний момент.

3.5 Структурна схема програмного забезпечення

Програмне забезпечення для аналізу і структуризації контенту спроектовано на основі використання бібліотеки Telegraf та деталізованої на рис. 3.3 архітектури. Відповідно в основі програмного проєкту лежить наступний набір файлів і тек:

- index.js – точка входу до програми;
- stages.js – файл, що відповідає за з'єднання сцен з основною програмою;
- scenes.js – файл, що генерує всі сцени в програмі;

- services.js – файл, в якому описуються всі HTTP запити;
- auth.js – сцена, яка активується після тільки для адміністратора і власника та надає основний функціонал для керування станом програми;
- info.js – сцена, в якій можна вибирати актуальні питання та отримувати на них відповідь;
- getTables.js – сцена, в якій можна отримати список актуальних таблиць;
- deleteTable.js – сцена, в якій можна видаляти таблиці;
- main.js – головна сцена, з якої користувач може перейти до інших сцен;
- scanTable.js – сцена для контент аналізу та структуризації документів;
- constants.js – файл, який містить в собі літеральні значення програми та інші константи;
- utils.js – файл містить в собі чисті функції, які не залежать від контексту;
- dbConnect.js – файл, який відповідає за з'єднання з базою даних;
- commands - директорія, яка містить в собі обробники команд для чат-бота (рис. 3.5).

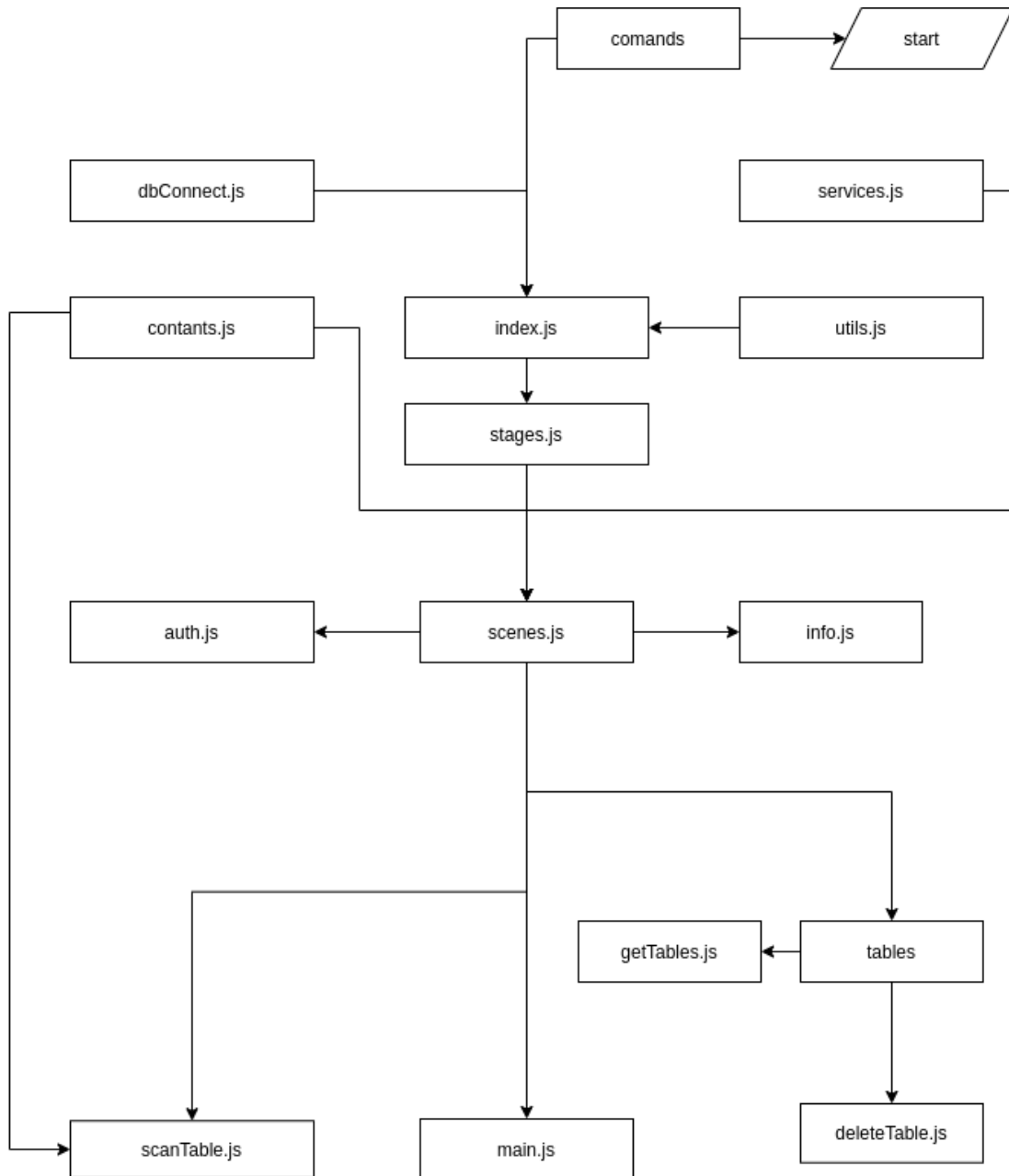


Рисунок 3.5 – Структурна схема програмного забезпечення

3.6 Висновки за розділом 3

В якості мови програмування для реалізації програмного забезпечення для аналізу та структурування контенту обрано мову програмування JavaScript, в якості виконавчої середовища використовується Node.js та Telegraf, а

для забезпечення роботи з базою даних використано систему керування базами даних MongoDB.

Визначено функціональні вимоги до програмного забезпечення за допомогою діаграм прецедентів. Виконано проєктування архітектури застосунку на основі використання бібліотеки Telegraf. Описано структурну схему програмного проєкту, що розробляється.

4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

4.1 Алгоритм функціонування системи

Робота програмного забезпечення для аналізу та структуризації текстової інформації ґрунтується на взаємодії її двох складових частин, які забезпечують адміністративну частину та роботу звичайного користувача. Вибір відповідної частини виконується за рахунок параметрів запиту користувача.

У разі роботи адміністратора з системою необхідно, щоб в БД, а саме в колекції `admins` або `owners`, був унікальний ідентифікатор користувача. В такому випадку у користувача з'являється можливість перейти в меню, в якому він отримує основний функціонал програмного забезпечення для наповнення бази знань, а саме:

- отримання списку документів, що використовуються;
- додавання нових документів для аналізу і структуривання інформації;
- видалення документів;
- надання прав адміністратора іншим користувачам;
- надання прав власника іншим користувачам;
- можливість отримувати статистику.

Якщо системою користується звичайний користувач, то відразу після старту чат-бота, йому буде доступна можливість отримати список поширених запитань, на основі документів, який проаналізувала система. Після вибору потрібного питання, користувач отримує список відповідей, відповідні даному питанню.

Отриману в результаті функціональну схему роботи програмного забезпечення представлено на рис. 4.1.

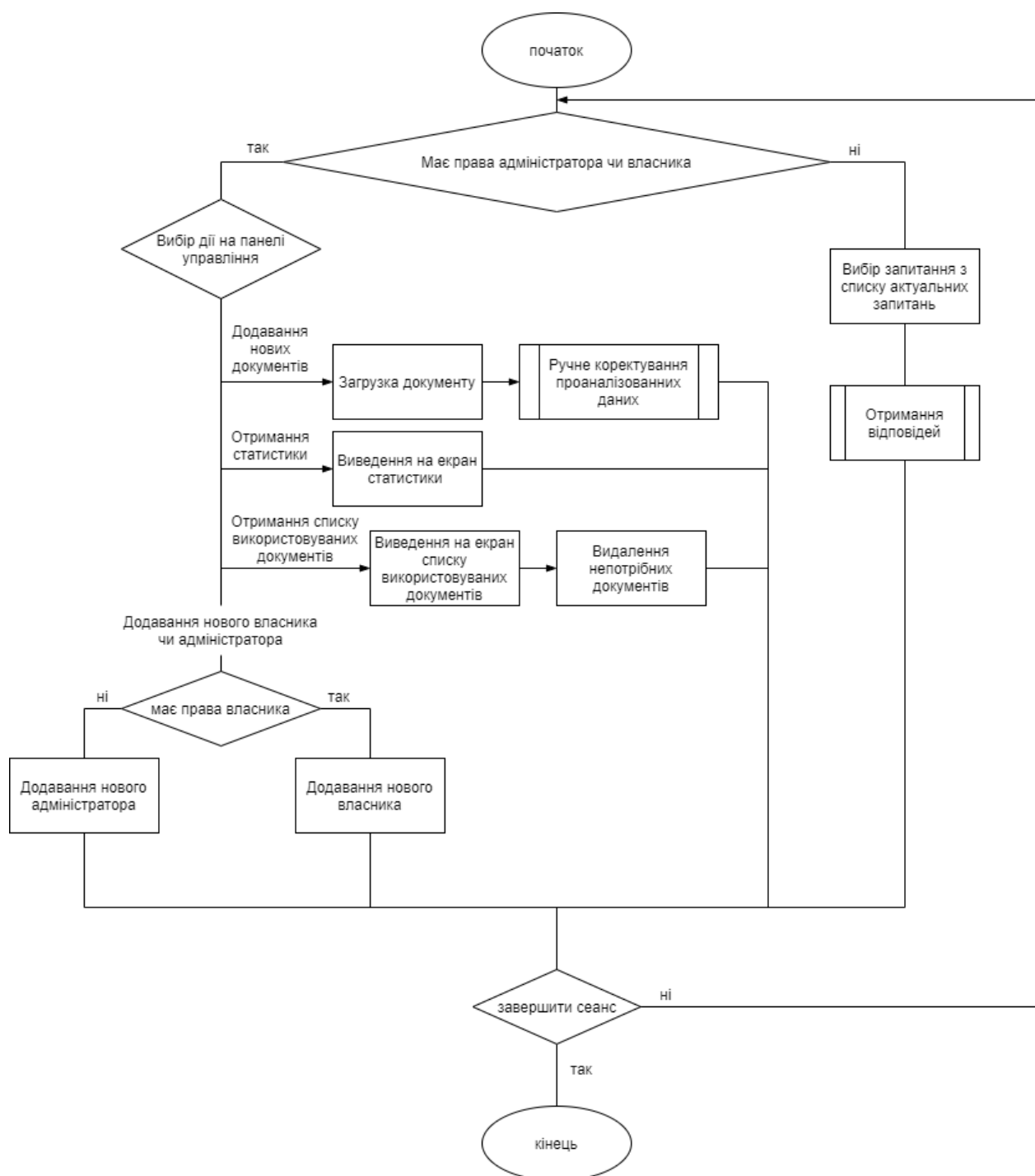


Рисунок 4.1 – Функціональна схема програми

4.2 Опис модулів реалізованої системи

Структура програма побудована на основі бібліотеки Telegraf, тому структурно розділена на сцені. Сцена в цьому застосунку - це атомарна одиниця логіки, яка відпрацьовує в даний момент роботи чат-бота.

В контексті сцени користувач може взаємодіяти з ботом та використовувати тільки методи, які надає бот в даний момент. А це означає, що користувач не може ніяк викликати команду або дію, яка в даний момент не “слухається” ботом. Чат-бот підказує все, що може реалізувати користувач в даний момент. Це зроблено для того, щоб забезпечити відмовостійку систему, в якій обробка різних подій відбувається по очікуваному для користувача потоку. Це полегшує також роботу для програміста, так як налагоджувати такий код значно простіше.

Основною точкою входу в програму є `index.js`, в даному файлі відбувається наступна черговість операцій:

- ініціалізується підключення до БД. Відбувається це за допомогою виклику функції `connectDB`, яка імпортується з файлу `db/connect.js`; відразу після ініціалізації успішного з'єднання асинхронно викликається функція, в якій відбувається замикання екземпляра БД, а також запускаємо бота;
- перевизначається контекст майбутнього екземпляра застосунка, відбувається це за допомогою створення класу, який успадковується від `Context`, класу імпортованого з бібліотеки Telegraf. Усередині класу реалізований метод `goBack`, який приймає в себе сцену, до якої потрібно повернутися, а повертає виклик - функцію, в яку передається показник `this` і назва сцени, до якої потрібно повернутися. Метод був створений для того, щоб з будь-якої частини програми, можна було повернутися повернутися в обрану сцену, за допомогою виклику функції з контексту. Тут також в

контекст додається властивість, яким присвоюється раннє замкнутий примірник БД;

- створюємо екземпляр застосунку за допомогою класу `Telegraf` і передаємо туди токен, і вище створений перевизначений контекст;
- визначаємо три проміжних обробника, перший створює сесію, другий додає в контекст програми підготовлені сцени і третій проміжний обробник служить для додавання в сесію значення за замовчуванням для попередньої сцени;
- створюємо конфігураційний файл з основними командами, для того щоб їх можна було отримати зі списку команд для зручності користувача;
- передаємо функцію зворотного виклику для команди `start`, яка запускає чат-бота з програми `Telegram`.
- Після того як всі основні елементи були проініціалізовані, користувач може ввести команду та почати взаємодіяти з програмою, для цього потрібно ввести команду `/start`.

4.2.1 Описання головної сцени

Основна сцену під назвою `main`, яка знаходяться в каталозі зі сценами `/scenes/main.js`. Дана сцена реалізує наступні події:

- подія входу в сцену виводить користувачеві клавіатуру для вибору наступної дії;
- подія перехвату вибраної дії з клавіатури.
- подія, яка потрібна для того, щоб повідомити користувача про те, чого чекає від нього бот.

4.2.2 Описання сцени панелі управління

Сцена панелі управління доступна, якщо користувач має необхідні права доступу. За неї відповідає файл, який знаходиться по шляху `scenes/auth.js`.

Сцена реалізує в собі такі події:

- подія входу в сцену виводить користувачеві клавіатуру для вибору наступного дії;
- подія перехоплення дії, в яку передається виклик функції `useBack`. Функція потрібна для того щоб не писати повторюється шматок логіки з реакцією на подію з клавіатури "back". Усередині даної функції викликається раніше передана в контекст функція `goBack` і відбувається повернення на попередню сцену;
- подія перехоплення обраної дії з клавіатури.

З цієї сцени в розробленому застосунку є невеликий вибір функціоналів, які представлені на рис. 4.1.

4.2.3 Описання сцени сканування документів

Функція додавання нових документів відправляє користувача на сцену, яка знаходиться в директорії `scenes/scanTable.js`. Дана сцена по суті є ядром програми, в який реалізуються основні функції для аналізу та структуризації текстової інформації. Сцена реагує на такі події:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступного дії;
- подія перехоплення дії, в яку передається виклик функції `useBack`;
- подія, яке перехоплює документ відправлений в чат-бота.

Після того як користувач відправить документ чат-бот, Telegram збереже файл у своїй файлового сховищі на сервері. А в подія відправиться

meta, з якої чат-бот може далі працювати. Якщо документ є підтримуваний для чат-бота, бот видасть про це відповідну помилку. Для того, щоб отримати файл і обробити його, потрібно отримати шлях до файлу, в файловому просторі сервера. Для цього реалізована функція в файлі services.js під назвою getFilePath.

```
const getFilePath = async (id) => {
  try {
    const {data: { result }} = await
    axios.get(`https://api.telegram.org/bot${config.get('token')}/getFile?file_id=${id}`
  ); return result
  }
  catch (e) { throw e }
};
```

Функція приймає в себе id і відправляє назад чат-боту шлях в файлової системі сервера. Функція є асинхронної, і віддасть результат чат-боту як тільки буде виконано запит.

Як тільки шлях отримано, чат-бот може отримати документ, виконавши наступний асинхронний запит getFile, який також знаходиться в файлі services.js.

Файл отримано тепер потрібно конвертувати отримані дані в формат JSON. Для цього запускається функція transformToJson знаходиться в файлі utils, в неї передається отриманий документ і його тип. Тип передається для того, щоб визначити подальший спосіб аналізу інформації. Якщо передано тип файлу xlsx, запускається функція parseXLSX.

```
const parseXLSX = async (file) => {
  const XLSX = require('xlsx');
```

```

const workbook = await XLSX.read(file, { type: 'array' });
const sheet_name_list = workbook.SheetNames;
const data = [];
sheet_name_list.forEach(function(y) {
    const worksheet = workbook.Sheets[y];
    const headers = {};
    for(z in worksheet) {
        if(z[0] === '!') continue;
        // Проаналізуйте стовпець, рядок та значення
        const col = z.substring(0,1);
        const row = parseInt(z.substring(1));
        const value = worksheet[z].v;
        // Зберігати імена заголовків
        if(row == 1) {
            headers[col] = value;
            continue;
        }
        if(!data[row]) data[row]={};
        data[row][headers[col]] = value;
    }
    //Видалити порожні перші два порожні рядки
    data.shift();
    data.shift();
});
return data}

```

У даній функції за допомогою бібліотеки/npm пакета `xlsx-parser` проходить по всіх колонках і рядах таблиці і дані записуються у відповідні поля JSON об'єкта. А якщо тип файл `docx`, запускається `parseDOCX`, і

відбувається те ж саме, обираються необхідні таблиці, і здійснюється по ним прохід, тільки за допомогою бібліотеки/npm пакета `docx-parser`. Після того як чисті JSON дані отримані, в MongoDB зберігається ідентифікатор і дані даного документа для того, щоб в подальшому з ним можна було маніпулювати і зв'язати його з даною базою знань.

Наступною дією користувачеві/адміністратору чат-бота на екран виводиться список отриманих питань, для того, щоб вручну можна було скорегувати дані, те ж саме відбувається з відповідями.

Після закінчення процесу аналізу чат-бот може повторити весь процес заново або повернутися назад на сцену панелі управління.

4.2.4 Описання сцени отримання таблиць

Сцена отримання використовуваних документів під назвою `getTables`, яка знаходиться по шляху `scenes /tables/getTables.js` і реалізує наступні події:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступного дії, а також завантажує з БД колекцію завантажених раніше документів, після чого виводить користувачеві на екран список таблиць у вигляді повідомлення;
- подія перехоплення дії, в яку передається виклик функції `useBack`.

4.2.5 Описання сцени видалення таблиць

З сцени отримання таблиць користувач може перейти в сцену `deleteTables` по шляху `scenes/tables/deleteTables.js`. У цій сцені реалізуються наступні події:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступної дії, а також завантажує з БД колекцію завантажених

раніше документів, після чого виводить користувачеві на екран список таблиць у вигляді клавіатури, кожною кнопкою якої є певна таблиця;

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступного дії, а також завантажує з БД колекцію завантажених раніше документів, після чого виводить користувачеві на екран список таблиць у вигляді клавіатури, кожною кнопкою якої є певна таблиця;

- подія перехоплення дії, в яку передається виклик функції `useBack`;

- подія, яке перехоплює натиснення кнопки з клавіатури і видаляє таблицю.

4.2.6 Описання сцени отримання статистики

Сцена отримання актуальної статистики називається `stats` і знаходиться в директорії `scenes/stats.js` і реалізує наступні методи:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступного дії, а також завантажує з БД колекцію отриманих даних про те які питання задаються частіше всього користувачами чат-бота;

- подія перехоплення дії, в яку передається виклик функції `useBack`.

4.2.7 Описання сцени додавання нового адміністратора та власника

Сцена `admins` реалізує різну логіку тільки для різних прав доступу. Так, власник може додати адміністратора і видалити адміністратора, а адміністратор тільки додати нового адміністратора. Сцена реалізує наступні методи:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору наступного дії, а саме додавання нового адміністратора або

видалення адміністратора в разі роль є власник, а також завантажує з БД колекцію отриманих адміністраторів;

- подія перехоплення дії з клавіатури;
- подія перехоплення дії, в яку передається виклик функції useBack.

4.2.8 Описання сцени отримання відповідей на запитання

Для користувача без прав адміністратора або власника з головної сцени доступні тільки сцена отримання поширених запитань answersByQuestion по шляху scenes/answersByQuestion.js, яка реалізує наступні події:

- подія входу в сцену, виводить користувачеві клавіатуру для вибору потрібного питання взятого з БД колекції questions;
- подія перехоплення дії з клавіатури і виведення відповідей по обраному питанню з БД колекції answers;
- подія перехоплення дії, в яку передається виклик функції useBack.

4.2.9 Описання подій для всіх сцен

Для всіх сцен реалізується обов'язково реалізуються наступні події:

- подія, яка реагує на будь-яке повідомлення, яке сцена не очікує і перезавантажує дану сцену;
- подія, яка викликається при виході зі сцени і видаляє всі непотрібні елементи управління.

4.3 Висновки за розділом 4

Описано алгоритм функціонування системи з наведенням функціональної схеми.

Описано процес реалізації програмного забезпечення з відповідними рішеннями щодо реалізації чат-боту та забезпечення реалізації запропонованого в роботі методу для перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

5 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

5.1 Призначення програми

Розроблене програмне забезпечення для аналізу контенту та структуризації текстової інформації призначене для перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу. Для цього був розроблений чат-бот в месенджері Telegram, це сучасний спосіб для реалізації бізнес логіки та інтерфейсу інтерфейсу користувача. Програмне забезпечення для аналізу контенту і структуризації текстової інформації може отримувати в себе документи, формувати з отриманих даних базу знань, з якої в майбутньому може отримувати інформацію звичайний користувач, отримуючи список актуальних питань, а також отримуючи список відповідей на питання, що цікавить.

5.2 Умови виконання програми

Для повноцінної роботи програмного забезпечення, користувачеві необхідно надати доступ до застосунку Telegram. Застосунок Telegram є крос-платформний, а отже у користувача не виникне проблем з підключенням, як б платформу він не використовував з наступного переліку:

- Android;
- iOS;
- Windows Phone;
- Windows NT;
- macOS;
- GNU / Linux;
- будь-який веб-браузер.

Для роботи програми Telegram на пристрої повинні бути як мінімум наступні характеристики:

- процесор з не менш ніж двома ядрами з частотою кожного не менше 1.4 ГГц;
- обсяг оперативного запам'ятовуючого пристрою – не менше 512мб;
- обсяг вільного простору на жорсткому диску – не менше 30мб.

Застосунок Telegram є дуже легковагим для систем за сучасними мірками.

5.3 Виконання програми

Для того щоб розпочати роботу програмного забезпечення, потрібно спочатку запустити на виконання серверну частину програмного забезпечення, для чого необхідно виконати наступні команди через командний рядок:

`npm install` - встановлює залежності;

`npm start` - запускає сервер

Щоб дані команди успішно відпрацювали в оточенні сервера повинен бути глобально встановлений `npm`.

Після того як сервер успішно запущений, можна приступати до роботи з ботом, для цього потрібно знайти бота в додатку Telegram з ім'ям `@nuzp_faq_bot` (рис. 5.1).

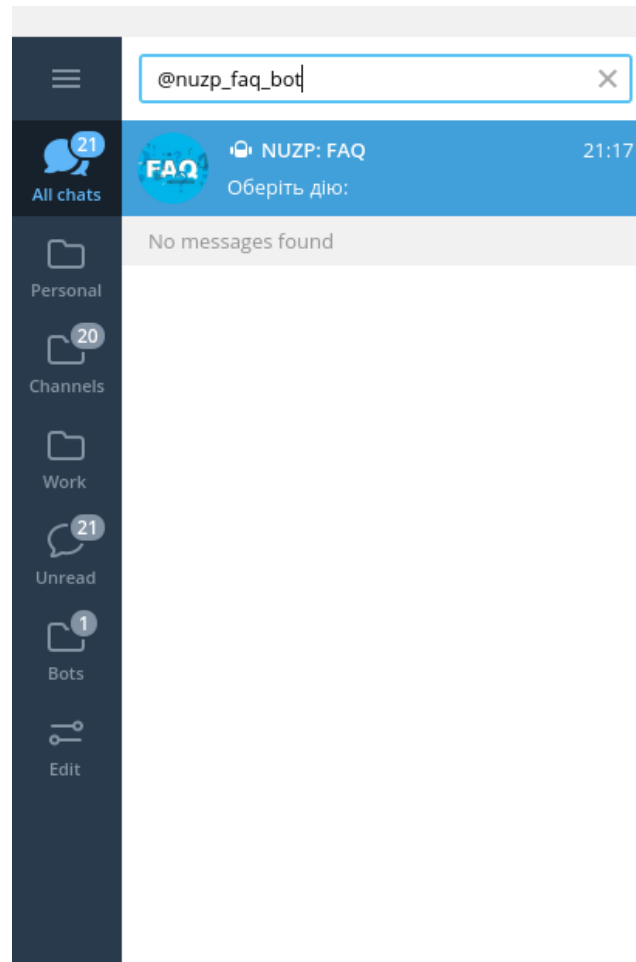


Рисунок 5.1 – Розташування в месенджері Telegram

Натиснувши по даному боту відкриється діалог, в якому можна дізнатися список актуальних команд на даний момент (рис. 5.2).

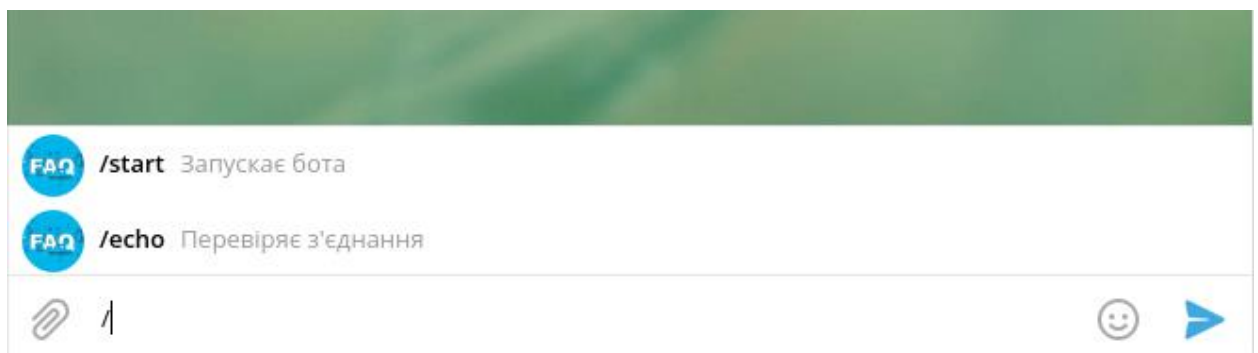


Рисунок 5.2 – Команди чат-бота

Почати працювати з ботом можна прописавши команду /start. Перед користувачем з'явиться наступний набір дій (рис. 5.3). Дане меню є основною точкою входу в подальший функціонал. Доступно на вибір дві дії:

- отримати відповіді на найбільш поширені питання;
- панель управління.

Перша дія направляє користувача в список до найбільш поширених запитань та відповідей на конкретні питання.

Друга дія надає панель управління програмним засобом.

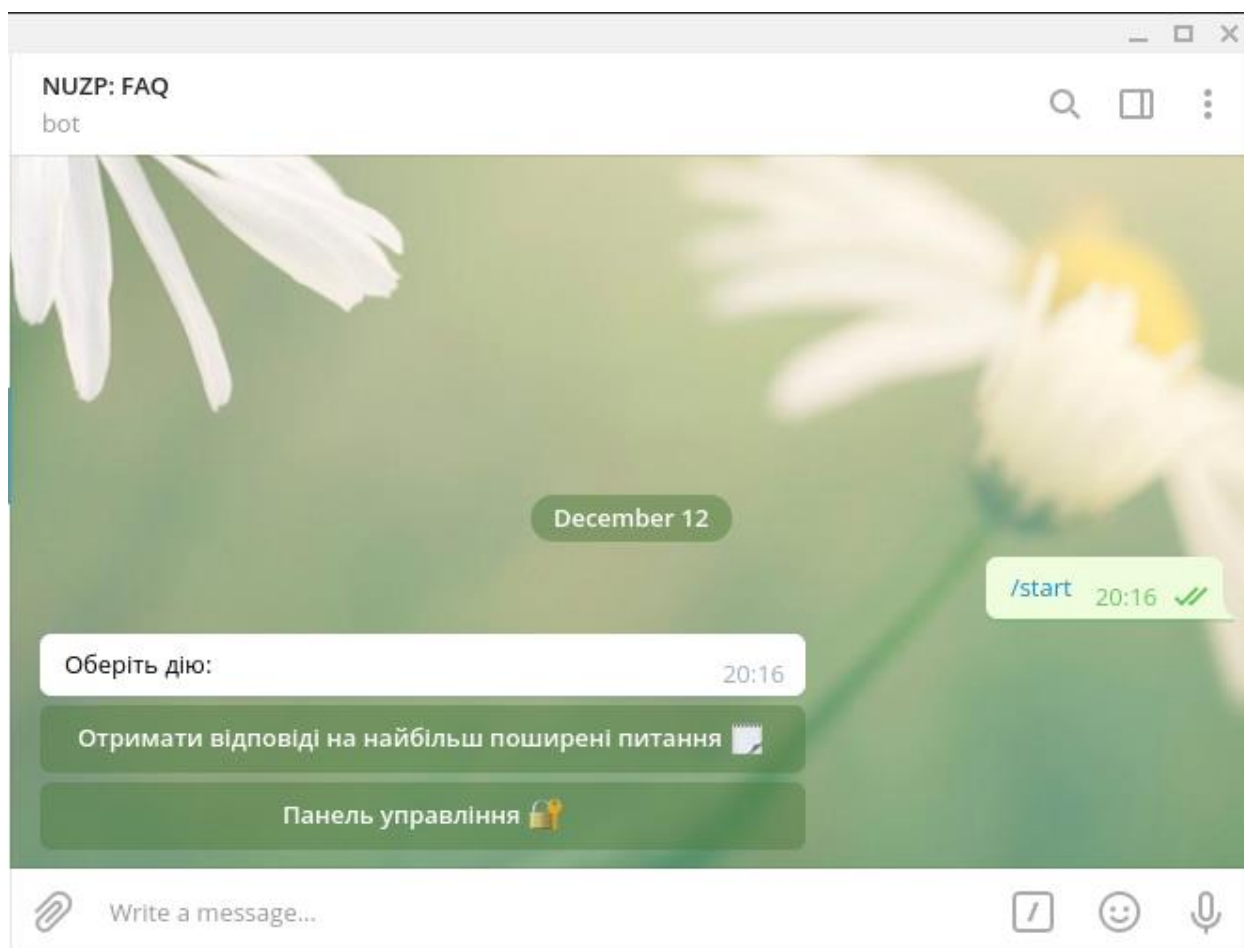


Рисунок 5.3 – Головне меню чат-бота

Сцена найпоширеніших запитань (рис. 5.4) відразу після переходу виводить користувачеві список поширених питань. Питання є кнопками,

натиснувши на які, можна отримати відповіді з конкретного питання, отримані системою при аналізі документів.



Рисунок 5.4 – Сцена найпоширеніших запитань

Панель управління (рис. 5.5) є основною сценою для управління станом застосунка і володіє наступним функціоналом:

- сканування таблиці;
- отримання списку використаних таблиць та їх видалення;
- отримання актуальної статистики;
- додавання нового адміністратора.

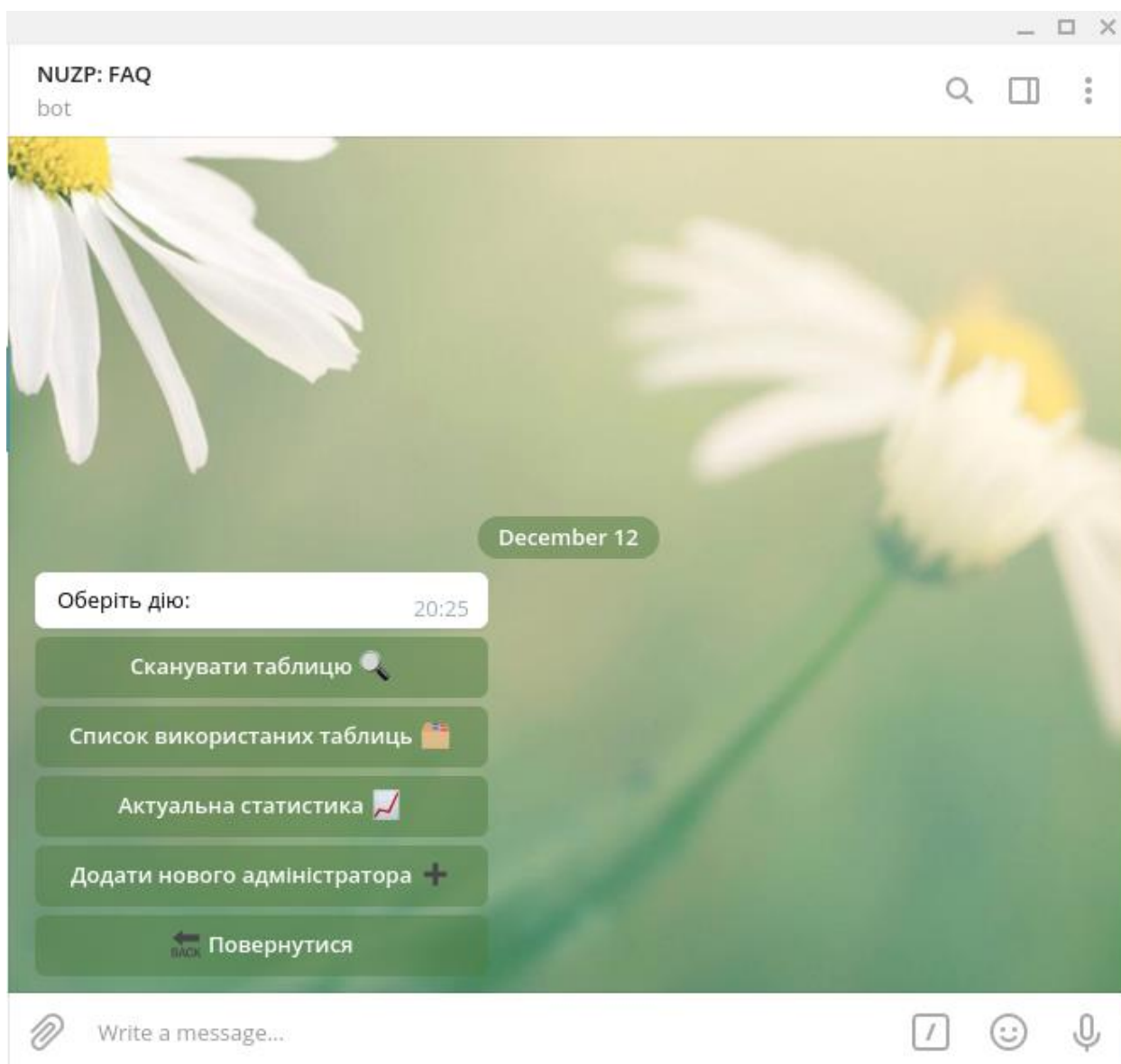


Рисунок 5.5 – Панель управління застосунком

Сцена сканування таблиць (документів) (рис. 3.6): дозволяє розпарсити доданий в неї документ. Документ повинен бути один з наступного типів: `xlsx`, `docx`, інакше користувач отримати повідомлення "Даний формат документу не підтримується. Спробуйте ще раз". Якщо користувач відправить щось окрім документу, чат-бот надішле повідомлення "Краще таблицю". Після додавання документа, чат-боту знадобиться деякий час, залежить від розміру файлу, щоб обробити документ і додати його в БД. Після цього користувач може додати ще один документ, або повернутися назад.

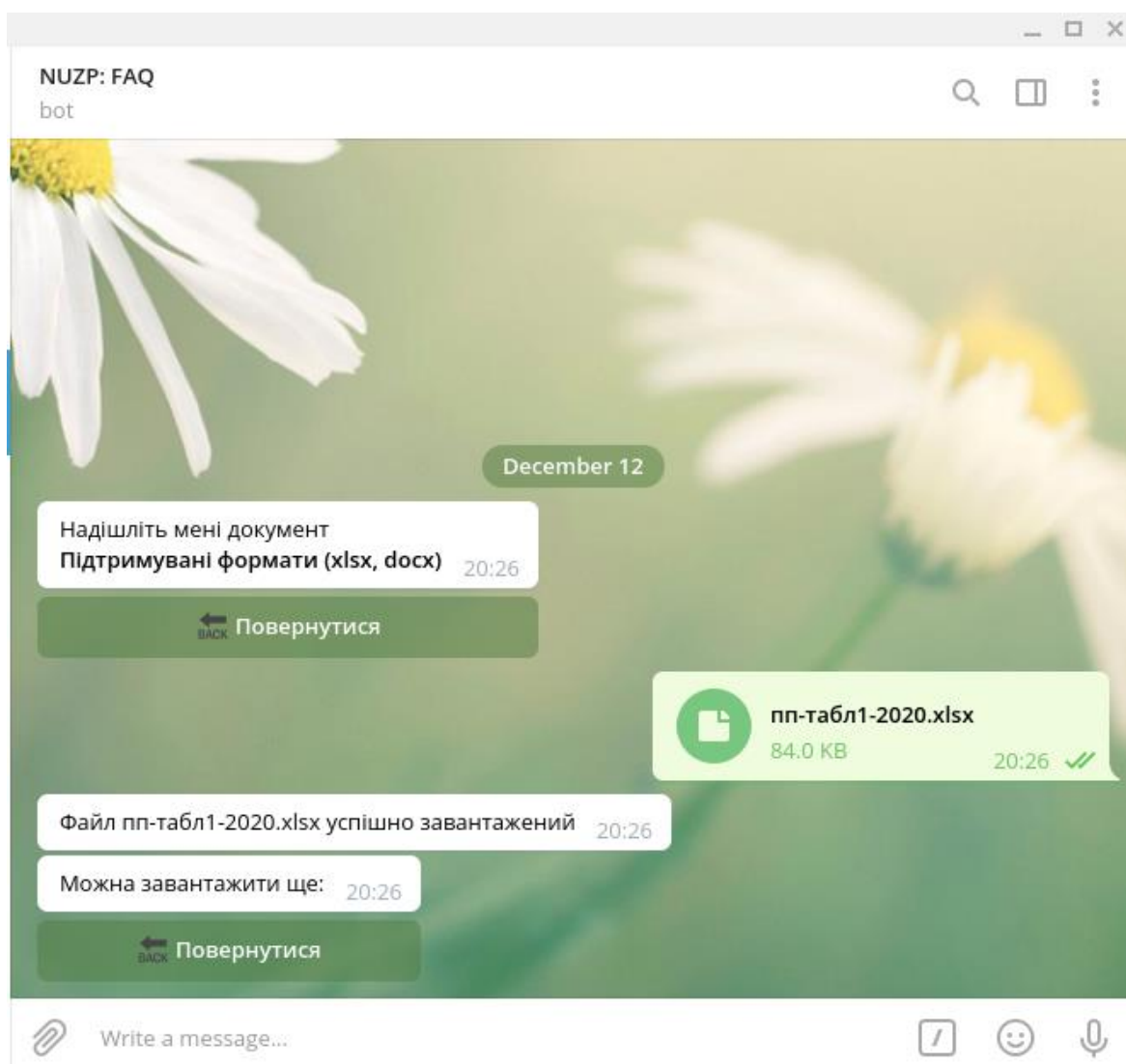


Рисунок 5.6 – Сцена сканування таблиць

На рисунках 5.7, 5.8 наведені приклади використаних таблиць Правил прийому

Таблиця 5

**ПЕРЕЛІК СПЕЦІАЛЬНОСТЕЙ,
яким надається особлива підтримка**

Шифр галузі	Галузь знань	Код спеціальності	Назва спеціальності
12	Інформаційні технології	124	Системний аналіз
13	Механічна інженерія	131	Прикладна механіка
		132	Матеріалознавство
		133	Галузеве машинобудування
		134	Авіаційна та ракетно-космічна техніка
		136	Металургія
14	Електрична інженерія	141	Електроенергетика, електротехніка та електромеханіка
		144	Теплоенергетика
15	Автоматизація та приладобудування	151	Автоматизація та комп'ютерно-інтегровані технології
		152	Метрологія та інформаційно-вимірювальна техніка
		153	Мікро- та наносистемна техніка
17	Електроніка та телекомунікації	172	Телекомунікації та радіотехніка
		173	Авіоніка
19	Архітектура та будівництво	192	Будівництво та цивільна інженерія
27	Транспорт	275	Транспортні технології (за видами)

Рисунок 5.7 – Приклад docx документу

Перелік освітніх ступенів та спеціальностей, за якими оголошується прийом на навчання, ліцензовані обсяги та нормативні терміни навчання Національний університет "Запорізька політехніка" (назва вищого навчального закладу)											
Бакалавр											
Термін навчання на денний та заочний формах: на основі повної загальної середньої освіти - 4 роки на основі освітніх ступенів (освітньо-кваліфікаційних рівнів) вищої освіти - 2 або 3 роки											
Код	Назва	Код	Назва	Спеціалізація (освітня програма) <i>/коже повторювати назву спеціальності/</i>	Ліцензовані обсяги		Нормативні терміни навчання		Вартість одного року навчання, грн. *		
					Довша форма навчання	Зокращена форма навчання	На основі ПЗСО	На основі освітніх ступенів (рівнів)	Довша форма навчання	Зокращена форма навчання	
01	Освіта	016	Спеціальна освіта	016 01 логопедія / корекційна освіта	25	35	4р	2р	11500	7500	
01	Освіта	017	Фізична культура і спорт	фізичне виховання	100	50	4р	2р (3р)	11500	7500	
02	Культура і мистецтво	022	Дизайн	графічний дизайн	40	20	4р	3р	13900	9800	
02	Культура і мистецтво	022	Дизайн	дизайн середовища			4р	3р	13900	9800	
02	Культура і мистецтво	022	Дизайн	промисловий дизайн			4р	3р	13900	9800	
03	Гуманітарні науки	035	Філологія	035 041 германські мови та літератури (переклад виключно), перша - англійська	85	15	4р	3р	15000	9700	
05	Соціальні та поведінкові науки	051	Економіка	управління персоналом та економіка праці	150	60	4р	2р	12300	8500	
05	Соціальні та поведінкові науки	051	Економіка	цифрова економіка та електронний бізнес			4р	2р	12300	8500	
05	Соціальні та поведінкові науки	051	Економіка	митна справа			4р	2р	12300	8500	
05	Соціальні та поведінкові науки	052	Політологія	політологія	40	-	4р	2р	10800	-	
05	Соціальні та поведінкові науки	053	Психологія	психологія	60	40	4р	3р	13000	9200	
06	Журналістика	061	Журналістика	журналістика	50	25	4р	2р	13000	9200	
06	Журналістика	061	Журналістика	зв'язки з громадськістю та діяльність прес-служб	25	-	4р	-	-	-	
07	Управління та адміністрування	071	Облік і оподаткування	облік і аудит	100	100	4р	3р	12300	8500	
07	Управління та адміністрування	072	Фінанси, банківська справа та страхування	фінанси і кредит	125	90	4р	3р	12300	8500	
07	Управління та адміністрування	072	Фінанси, банківська справа та страхування	фінансово-економічна безпека			4р	3р	12300	8500	
07	Управління та адміністрування	073	Менеджмент	менеджмент організацій і адміністрування			4р	3р	12300	8500	
07	Управління та адміністрування	073	Менеджмент	менеджмент в будівництві	100	80	4р	3р	12300	8500	
07	Управління та адміністрування	075	Маркетинг	маркетинг	30	30	4р	3р	12300	8500	
07	Управління та адміністрування	076	Підприємництво, торгівля та біржова діяльність	організація торгівлі та комерційна логістика			4р	2р	12300	8500	

Рисунок 5.8 – Приклад xlsx документу

Сцена списку використовуваних таблиць (рис. 5.9) складається з двох частин. Перша частина надається користувача при переході в сцену і виводить користувачеві список використовуваних таблиць. Якщо документів не буде знайдено користувач отримає повідомлення "Документів не знайдено!". Якщо натиснути на кнопку "Перейти в режим видалення" користувач отримає другу частину функціоналу і перейде в режим видалення таблиць з БД.

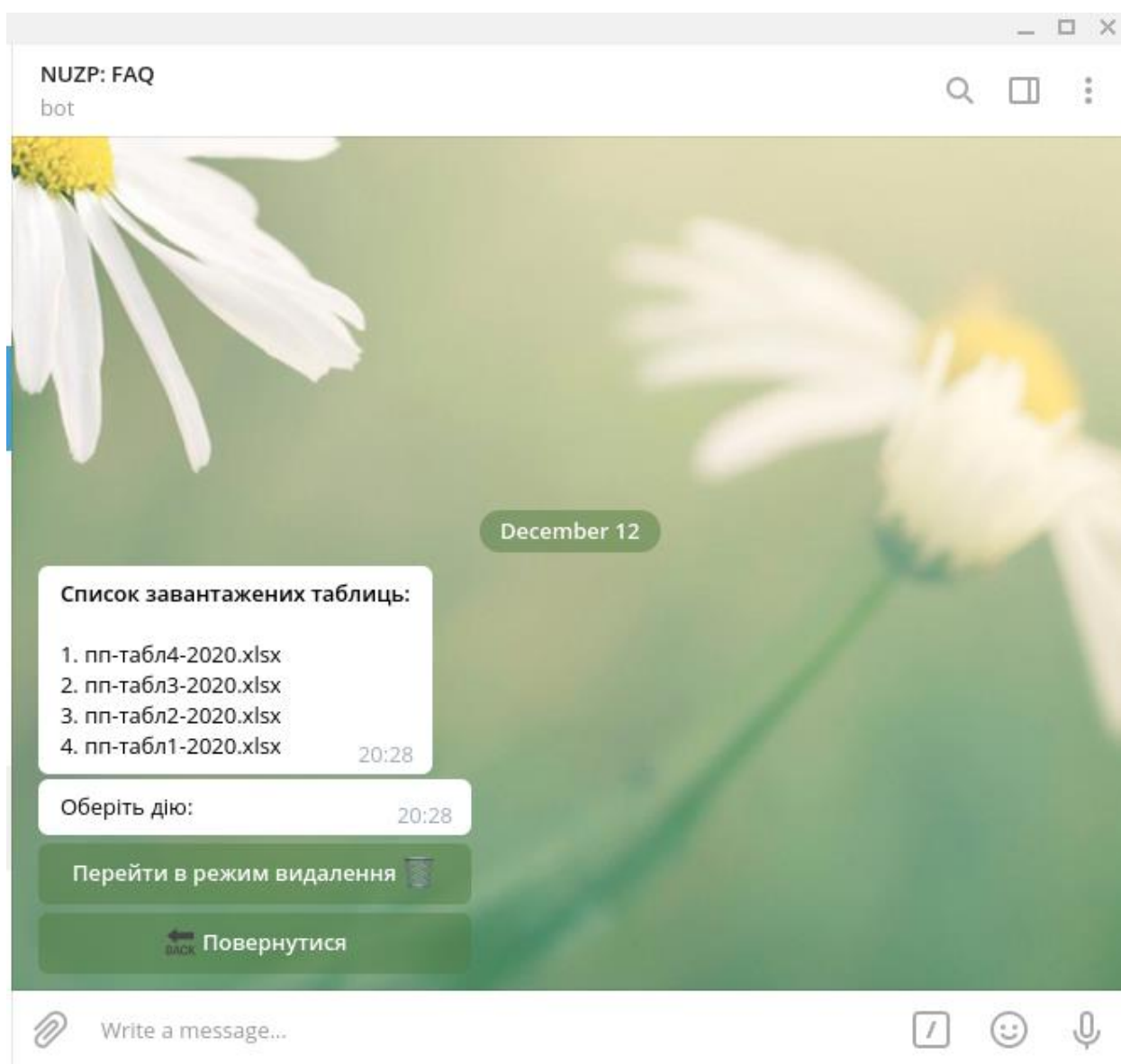


Рисунок 5.9 – Сцена списку завантажених таблиць

Сцена видалення використовуваних таблиць (рис. 5.10) виводить користувачеві на екран той же список таблиць, що і на рисунку 5.9, тільки пункти стають кнопками, натиснувши на які, у користувача з'являється можливість видалити таблицю і все що з нею пов'язано з БД.

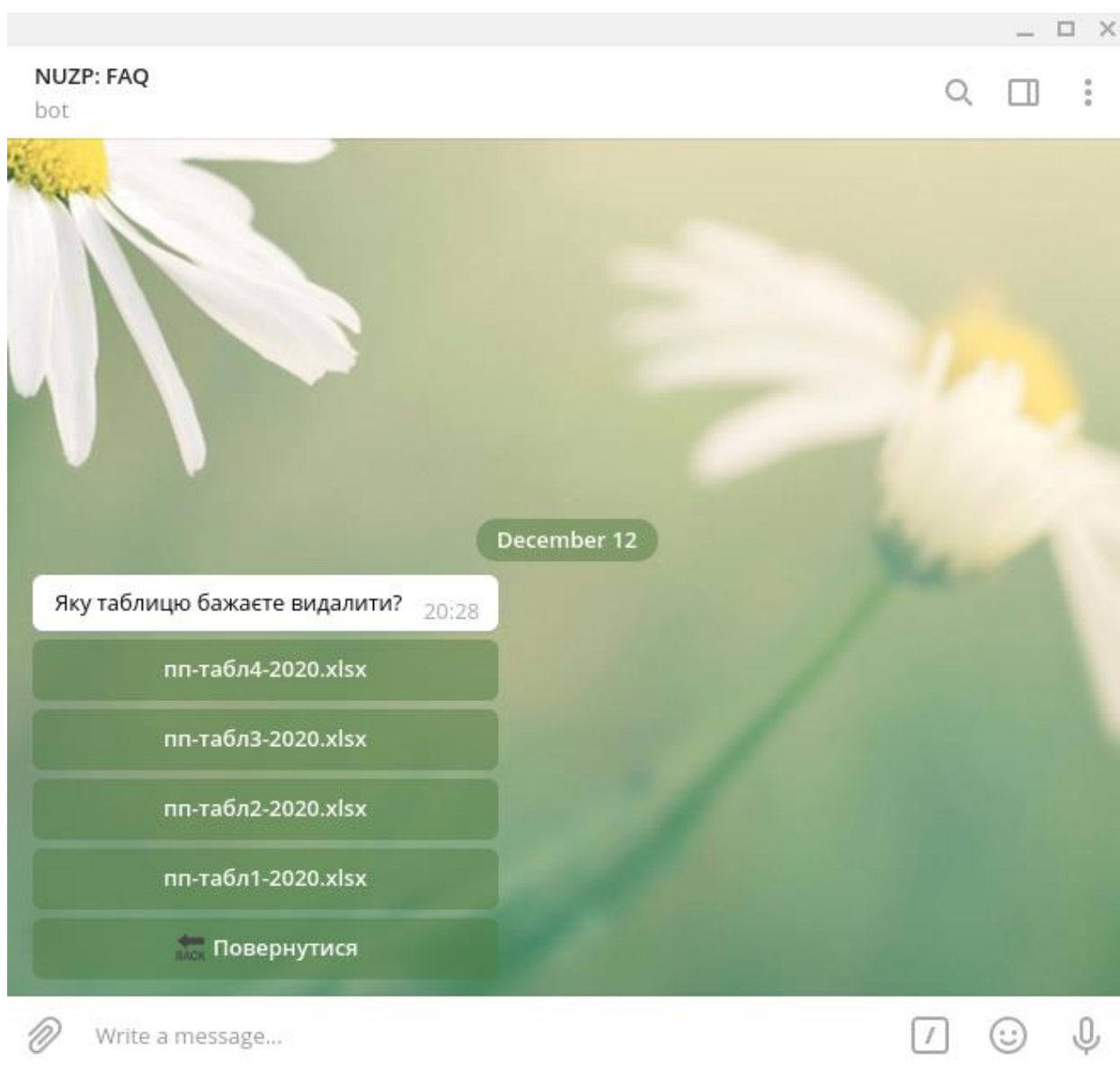


Рисунок 5.10 – Сцена видалення завантажених таблиць

Сцена статистики (рис. 5.11) виводить на екран користувачеві дані статистики по запитах користувачів, завдяки цьому можна визначити тенденції серед користувачів.

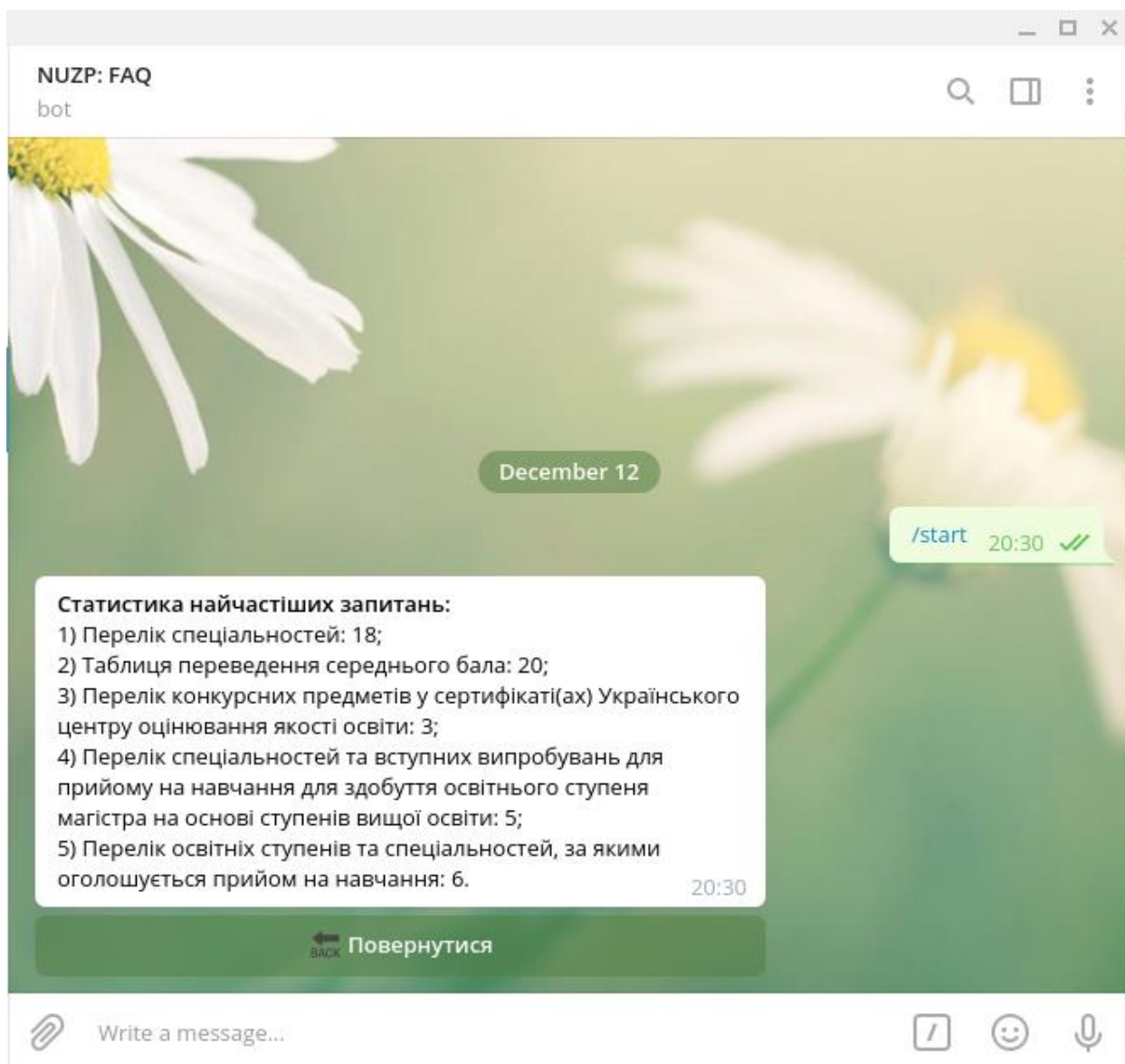


Рисунок 5.11 – Сцена статистики

Сцена додавання нового адміністратора представлено на рисунку 5.12. Щоб додати нового адміністратора потрібно поділитися з чат-ботом контактом майбутнього адміністратора в телеграм. Спосіб може відрізнятись в залежності від пристрою на якому воно відбувається. Важливо щоб користувач якого додають, був у адміністратора в контактах. Якщо користувач вже присутній в системі чат-бот відправить повідомлення "Даний користувач вже є адміністратором!". Якщо спробувати відправити, щось крім контакту користувач отримає повідомлення "Краще контакт".

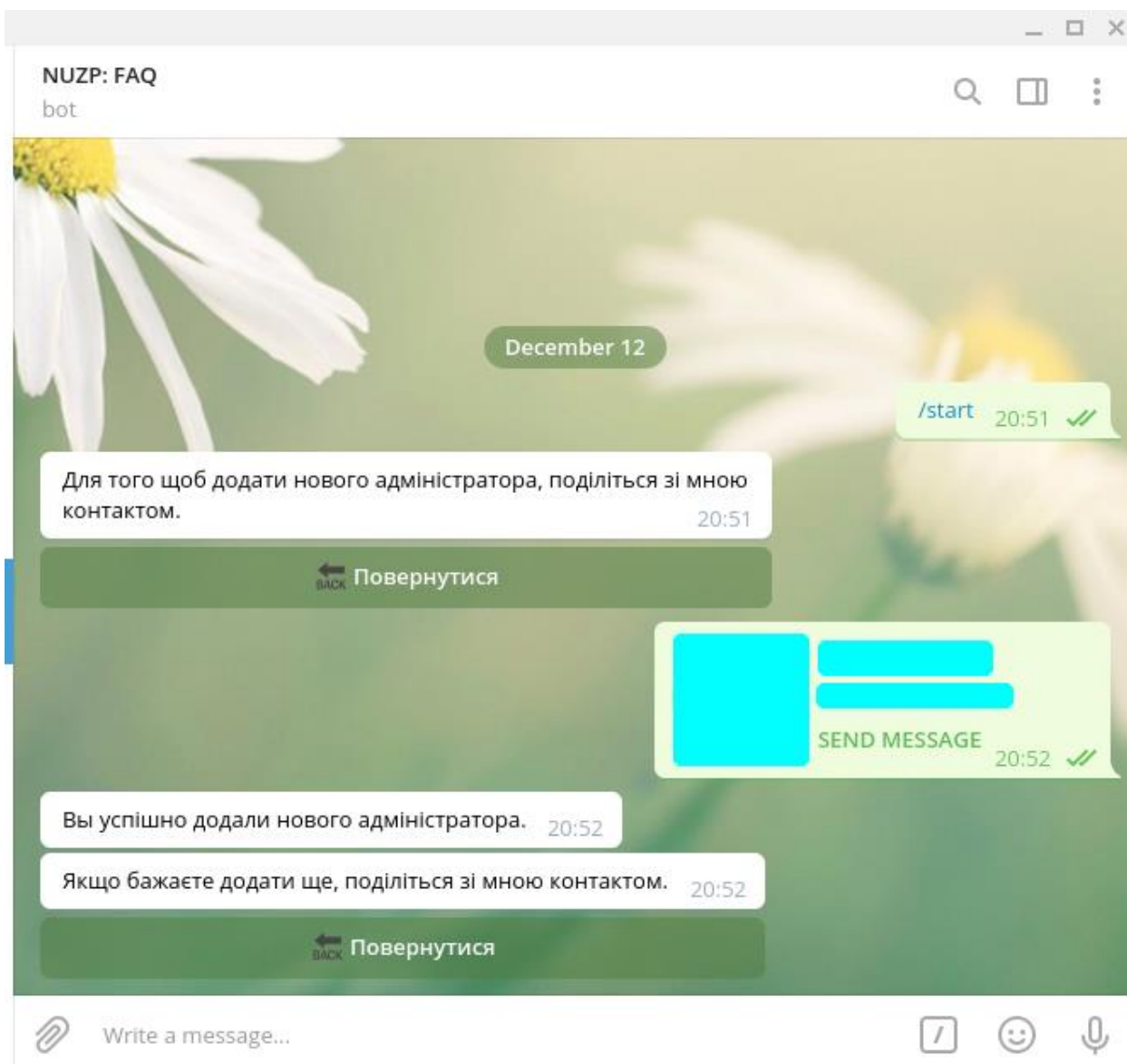


Рисунок 5.12 – Сцена додавання нового адміністратора

5.4 Висновки за розділом 5

Виконано перевірку роботи програмного забезпечення щодо реалізації всіх запланованих функціональних можливостей. Описано процес роботи звичайного користувача та адміністратора з програмою щодо виконання відповідних задач.

6 ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА

6.1 Вступ

Проведено економічне обґрунтування програмної реалізації методу аналізу контенту для структурування текстової інформації.

Цей програмний продукт повинен поліпшити якість і швидкість обміну інформацією між керівниками освітніх установ та студентами. Також система дозволяє працювати з даними через чат-бота, що робить їх дуже доступними. Крім того, система дозволяє збирати аналітичні дані.

В даний час на ринку такого програмного забезпечення немає повних аналогів (існуючі програми не використовують чат-ботів для аналізу контенту і структурування текстової інформації).

Очікується отримання прибутку за рахунок покупки підписки на розроблений продукт користувачів.

Опис ідеї проєкту наведено в табл. 6.1.

Таблиця 6.1 – Опис ідеї проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Система аналізу контенту для структурування текстової інформації, завдяки якому можна збирати дані з таблиць та відобразити їх у чат-боті	Автоматизація	Керівники зможуть збирати базу знань автоматично
	Комунікація (комунікація з студентами)	Студенти отримають єдину базу знань завітавши до чат бота
	Аналітика	На основі запитів до чат бота можна збирати дані та виявляти моду

Попередня характеристика потенційного ринку наведено в табл. 6.2.

Таблиця 6.2 – Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од	800
3	Динаміка ринку	Стагнація національної економіки. Темпи розвитку світової економіки позитивні, але з ознаками зменшення росту.
4	Наявність обмежень для входу	Відсутні.
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні.
6	Середня норма рентабельності в галузі або по ринку, %	60

Попередня характеристика потенційних клієнтів наведено в табл. 6.3.

Таблиця 6.3 – Попередня характеристика потенційних клієнтів

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів (користувачів)
1	Швидке і зручне отримання інформації	Студенти	Програмний засіб надає можливість отримувати інформацію через чат-бота	Зручність у використанні. Швидка робота системи. Спроможність швидко освоїти як користуватися системою.

Продовження таблиці 6.3

2	Автоматизація ручного труду	Керівники навчальних закладів	Програмний засіб надає можливість парсити дані, потрібно тільки перенести таблиці до необхідної папки	
3	Аналітика даних	Керівники навчальних закладів	Програмний засіб надає можливість отримувати аналітичні дані зібрані	

Обираємо першу альтернативу, для якої ймовірність отримання ресурсів є не нижчою за показники для інших альтернатив, а строк реалізації – відносно невеликий.

SWOT- аналіз наведено в табл. 6.4.

Таблиця 6.4 – SWOT- аналіз

Сильні сторони:	Слабкі сторони:
Інноваційність, унікальність, актуальність, націленість як на державний, так і на приватний сектор	Цінова політика, швидкий темп розвитку галузі
Можливості:	Загрози:
Впровадження інноваційних рішень, розширення бізнесу, вихід на міжнародний ринок	Поява сильних конкурентів на вітчизняному ринку

Вибір цільових груп потенційних споживачів наведено в таблиці 6.5

Таблиця 6.5 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Студенти, навчальних закладів	Присутня	Середній	Присутня	Легка
2	Керівники навчальних закладів	Присутня	Середній	Майже відсутня	Середня

6.2 Планування розробки програмного продукту

Весь процес розробки програмного модуля можна розділити на етапи. Для кожного етапу вказується трудомісткість, кількість виконавців і тривалість робіт. В розробці приймають участь програміст протягом 2,5 місяця і консультант протягом 0,5 місяця. Розробка розпочинається дев'ятого вересня і повинна бути виконана до п'ятнадцятого листопада 2020 року.

Тривалість робіт визначається за формулою (6.1):

$$T_{\text{ц}} = \frac{Q}{R}, \quad (6.1)$$

де $T_{\text{ц}}$ – тривалість циклу, днів;

Q – трудомісткість, людино-днів;

R – кількість виконавців, чоловік.

Отримана інформація наведена в таблиці 6.6.

Таблиця 6.6 – Етапи розробки продукту

Найменування роботи	Трудомісткість		Виконавці		Тривалість, днів
	люд.-днів	% до підсумку	Спеціальність	Кількість, чол.	
1. Аналіз та дослідження предметної області	4	9,5	Консультант Програміст	2	2
2. Розробка та затвердження технічного завдання	4	8	Консультант Програміст	2	2
3. Проектування структури програмного забезпечення	8	11,5	Програміст	1	8
4. Розроблення схеми функціонування програмного забезпечення та модулів	8	10,5	Програміст	1	8
5. Розроблення програмного коду	16	24	Програміст	1	16
6. Тестування та відлагодження програмного забезпечення	8	12,5	Програміст	1	8
7. Дослідження предметної області на основі розробленого програмного забезпечення	2	6	Консультант Програміст	2	1
8. Розробка розділів пояснювальної записки	5	12	Консультант Програміст	2	2,5
9. Розробка графічного матеріалу	5	6	Консультант Програміст	2	2,5
Разом	60	100		14	50

За даними таблиці 6.6 складається зведений стрічковий графік планування розробки програмного модуля, який представляє собою таблицю, в першому стовпці якої розміщені в порядку збільшення термінів початку виконання всі види роботи, а навпаки – календарний період їх виконання. Даний графік наведено в таблиці 6.7.

Таблиця 6.7 – Зведений стрічковий графік планування розробки програмного продукту

Дата / Найменування роботи	03.09.2020	09.09.2020	16.09.2020	23.09.2020	30.09.2020	07.10.2020	14.10.2020	21.10.2020	28.10.2020	04.11.2020	11.11.2020
1. Аналіз предметної області	■										
2. Визначення вимог до програмного продукту	■										
3. Аналіз принципів побудови експертних систем		■									
4. Дослідження методів вирішення задач			■								
5. Побудова моделі системи				■							
6. Створення програмного коду						■	■	■	■	■	
7. Тестування і налагодження програмного продукту										■	■
8. Складання програмної документації											■

В таблиці 6.7 сірим кольором позначено роботу програміста, а чорним – консультанта.

6.3 Визначення витрат на розробку програми

Для визначення витрат на розробку програми складається калькуляція кошторисної вартості робіт, яка включає наступні статті:

- основна заробітна плата;
- додаткова заробітна плата;
- відрахування на соціальне страхування і в інші фонди;
- витрати на спеціальне обладнання;
- матеріали і комплектуючі вироби;
- накладні витрати;
- податки.

6.3.1 Розрахунок основної заробітної плати

Витрати за цією статтею складаються з планового фонду зарплати всіх категорій працівників, зайнятих в розробці програми. Розрахунок зарплати ведеться на підставі даних про трудомісткості, представлених в таблиці 6.8.

Таблиця 6.8 – Розрахунок основної заробітної плати

Посада виконавця	Чисельність, чол	Місячний оклад, грн	Кількість місяців роботи	Сума заплати, грн.
Програміст	1	7480	2,5	18700
Консультант	1	9170	0,5	4585
Всього	2	16650	3	23285

6.3.2 Розрахунок додаткової заробітної плати

Додаткову заробітну плату приймають рівною 10% від основної заробітної плати працівників і розраховують за формулою 6.2:

$$ЗП_{\text{дод}} = ЗП_{\text{осн}} \cdot 0,1, \quad (6.2)$$

Підставивши величину основної заробітної плати в дану формулу, отримуємо:

$$ЗП_{\text{дод}} = 23285 \cdot 0,1 = 2328,5 \text{ грн.}$$

6.3.3 Відрахування на єдиний соціальний внесок

Відрахування на єдиний соціальний внесок становлять 22% і беруться від основної та додаткової заробітної плати. Вони розраховуються за формулою 6.3:

$$\text{ЄСВ} = (\text{ЗП}_{\text{осн}} + \text{ЗП}_{\text{дод}}) \cdot 0,22, \quad (6.3)$$

де ЄСВ – єдиний соціальний внесок, грн.;

$\text{ЗП}_{\text{осн}}$ – основна заробітна плата, грн.;

$\text{ЗП}_{\text{дод}}$ – додаткова заробітна плата, грн.

Підставивши величину основної заробітної плати і додаткової заробітної плати в дану формулу, отримуємо:

$$\text{ЄСВ} = (23285 + 2328,5) \cdot 0,22 = 5634,97 \text{ грн.}$$

Відрахування в єдиний соціальний внесок становлять 5634,97 грн.

6.3.4 Визначення витрат на матеріали

Використовується 2 найменування матеріалів: картридж - 335 грн. і папір 92 грн. (1 упаковка).

Прямі матеріальні витрати, грн. Розраховуються за формулою 6.4:

$$\text{В}_{\text{мат.}} = \sum q_i \cdot \text{Ц}_i, \quad (6.4)$$

де Ц_i – ціна і-го матеріалу, грн./од.;

q_i – норма витрат матеріалу, нат.од.

$$B_{\text{мат.}} = 1 \cdot 335 = 335 \text{ грн.}$$

Аналогічно розраховуються інші матеріали.

Розрахунок прямих матеріальних витрат приведений у таблиці 6.9.

Таблиця 6.9 – Прямі матеріальні витрати

Назва матеріалу	Одиниця вимірювання	Норма витрат, нат.од.	Ціна, грн./нат.од.	Сума, грн.
1. Папір	пачка	1	335	335
2. Картридж	шт.	1	92	92
Разом	-	-	-	427

6.3.5 Витрати на спеціальне обладнання

Ця стаття витрат є всеосяжною і покриває такі витрати, як амортизація, витрати на електроенергію, заробітну плату з відповідними відрахуваннями на соціальну діяльність працівників, які обслуговують обладнання.

Спочатку складається річний кошторис витрат на обслуговування і експлуатацію обладнання.

Річна сума амортизаційних відрахувань, грн. Розраховуються за формулою 6.5:

$$A_{\text{річ.}} = B_{\text{пер.}} \cdot \frac{H_a}{100\%}, \quad (6.5)$$

де $B_{\text{пер.}}$ – первісна вартість комп'ютера, грн.;

H_a – річна норма амортизації, % (річна норма амортизації для ПК приймається 25%).

Первісна вартість комп'ютера, грн. Розраховуються за формулою 6.6:

$$B_{\text{пер.}} = B_{\text{придб.}} \cdot (1 + K_{\text{тр.в.}}), \quad (6.6)$$

де $B_{\text{пер.}}$ – вартість придбання (сума, сплачена постачальнику), грн.,

$K_{\text{тр.в.}}$ – коефіцієнт транспортних витрат, який враховує витрати на доставку, монтаж та налагодження, приймається 0,05.

Підставивши значення в формулу 6.6, отримуємо:

$$B_{\text{пер.}} = 15850 \cdot (1 + 0,05) = 16642,5 \text{ грн.}$$

Підставивши значення в формулу 6.5, отримуємо:

$$A_{\text{річ.}} = 16642,5 \cdot \frac{25\%}{100\%} = 4160,63 \text{ грн.}$$

Річна сума амортизаційних відрахувань становить 4160,63 грн.

Річні витрати на електроенергію, грн. Розраховуються за формулою 6.7:

$$B_{\text{ел.річ.}} = P_{\text{н}} \cdot F_{\text{еф}} \cdot K_{\text{зав}} \cdot K_{\text{зп}} \cdot C_{\text{ел}}, \quad (6.7)$$

де $P_{\text{н}}$ – номінальна потужність ПК, кВт/год.;

$F_{\text{еф}}$ – річний ефективний фонд роботи ПК, год.;

$K_{\text{зав}}$ – середній коефіцієнт завантаження за часом;

$K_{\text{зп}}$ – коефіцієнт завантаження за потужністю;

$C_{\text{ел}}$ – ціна електроенергії, грн./кВт-год.

Номинальна потужність ЕОМ - 0,2 кВт. Середні коефіцієнти завантаження за часом і за потужністю рівні відповідно 0,9 і 0,6. Ціна однієї кіловат-години електроенергії складає 2,11 грн. Розраховуються за формулою 6.8:

$$F_{\text{еф}} = (D_{\text{кал}} - D_{\text{вих}} - D_{\text{св}}) \cdot t_{\text{зм}} - t_{\text{п.св.}} - t_{\text{обсл}}, \quad (6.8)$$

де $D_{\text{кал}}$, $D_{\text{вих}}$, $D_{\text{св}}$ – відповідно календарні, вихідні та святкові дні (на 2020 рік - 366, 104, 12);

$t_{\text{зм}}$ – тривалість зміни, год., за даними підприємства приймається вісім;

$t_{\text{п.св.}}$ – скорочення робочих днів перед святковими днями на 1 год., на 2020 рік становить 6 год.;

$t_{\text{обсл}}$ – час обслуговування ПК, приймається 104 год.

$$F_{\text{еф}} = (366 - 104 - 12) \cdot 8 - 6 - 104 = 1890$$

$$B_{\text{ел.річ}} = 0,2 \cdot 1890 \cdot 0,9 \cdot 0,6 \cdot 2,11 = 430,69$$

Заробітна плата обслуговуючого персоналу розраховується за формулою 6.9:

$$ЗП_{\text{обсл}} = ФЗР_{\text{р}} (1 + K_{\text{від}}) \frac{t_{\text{обсл}}}{F_{\text{еф.обсл}}}, \quad (6.9)$$

де $ФЗР_{\text{р}}$ – річний фонд заробітної плати (основної та додаткової) обслуговуючих робіт (грн.),

$K_{\text{від}}$ – коефіцієнт відрахувань на соціальне страхування і до інших фондів (%),

$t_{\text{обсл}}$ – час протягом року, необхідний на технічне обслуговування ЕОМ, (ч/рік),

$\Phi_{\text{еф. обсл.}}$ – річний ефективний фонд часу обслуговуючого персоналу.

Місячна заробітна плата обслуговуючого персоналу становить 4173 грн., а річний фонд заробітної плати становить 50076 грн. Річний ефективний фонд робочого часу обслуговуючого ПК робітника становить 1750 ч/год. На обслуговування одного ПК витрачається по 1 часу на місяць, в на рік становить 12 годин. З чого виходить, що:

$$ЗП_{\text{обсл}} = 50076 \cdot (1 + 0.22) \cdot \frac{12}{1750} = 415.43 \text{ грн.},$$

Стаття «Поточний ремонт обладнання» приймається рівною 3% від балансової вартості обладнання і складає 499,27 грн.

Стаття «Інші витрати» приймається рівною п'яти відсоткам від суми всіх попередніх статей витрат на утримання та експлуатацію устаткування. Сума всіх попередніх статей дорівнює 5500,55 грн., 5% від суми становлять 275,02 грн. (табл. 6.10).

Таблиця 6.10 – Кошторис витрат на утримання та експлуатацію обладнання

Найменування статей витрат	Сума, грн.
Амортизація устаткування	4160,63
Експлуатація устаткування (окрім витрат на поточний ремонт)	430,69
Поточний ремонт устаткування	499,27
Продовження таблиці 6.10	
Інші витрати	275,02

Разом	5775,57
-------	---------

Витрати на оплату машинного часу ЕОМ для написання й відлагодження програми вираховується за формулою 6.10:

$$C_{\text{мо}} = B_{\text{екс}} t_{\text{мч}}, \quad (6.10)$$

де $C_{\text{мо}}$ – витрати на сплату машинного часу, грн.;

$B_{\text{екс}}$ – експлуатаційні витрати на одну годину машинного часу цієї цифрової ЕОМ, грн. / машино-год.;

$t_{\text{мч}}$ – машинний час цифрової ЕОМ для написання і відлагодження даного програмного продукту, машино-год.

Експлуатаційні витрати на одну годину машинного часу використовуваної ЕОМ розраховують діленням суми витрат за кошторисом «Витрати на утримання та експлуатацію обладнання (ЕОМ)» на річний ефективний фонд часу роботи ЕОМ. Річний ефективний фонд часу роботи ЕОМ дорівнює 1866 годин. В результаті експлуатаційні витрати на одну годину машинного часу рівні:

$$B_{\text{екс.}} = \frac{5775,57}{1866} = 3,095 \text{ грн./машино-год.}$$

ЕОМ експлуатується 40 днів в одну зміну, що становить в сумі 320 годин. Таким чином, витрати на оплату машинного часу складуть:

$$C_{\text{мо}} = 3,095 \cdot 320 = 990,4 \text{ грн.}$$

6.3.6 Інші прямі витрати

В інші прямі витрати включаються витрати на що використовується при розробці системи комерційне програмне забезпечення: діловий ПП, що використовується постійно при роботі ПК (Windows 10 Pro) – 6967,03 грн. без ПДВ, (Microsoft Office Для дому та бізнесу 2020 для 1 ПК) – 6399 грн. без ПДВ. Розраховується за формулою 6.11.

$$S_{\text{діл.ПП}} = \frac{(\text{Ц}_{\text{ппWindows}} + \text{Ц}_{\text{ппOffice}}) \cdot T_{\text{КТС}}}{\Phi_{\text{еф.КТС}} \cdot T_{\text{с.ПП}}}, \quad (6.11)$$

де $S_{\text{діл.ПП}}$ – витрати на діловий ПП, грн .;

$\text{Ц}_{\text{ппWindows}}$ – ціна ПП Windows (без ПДВ), грн;

$\text{Ц}_{\text{ппOffice}}$ – ціна ПП Microsoft Office (без ПДВ), грн;

$T_{\text{КТС}}$ – машинний час КТС, необхідний користувачеві для розробки системи, машино-год / рік;

$\Phi_{\text{еф.КТС}}$ – річний ефективний фонд часу роботи КТС, машино-год / рік;

$T_{\text{с.ПП}}$ – термін служби ділового ПП, років.

$$S_{\text{діл.ПП}} = \frac{(6967,03 + 6399) \cdot 320}{1866 \cdot 5} = 458,43 \text{ грн.}$$

6.3.7 Накладні витрати

До накладних витрат відносяться витрати на загальне управління і загальногосподарські потреби (заробітна плата апарату управління, канцелярські витрати і т.д.), утримання та експлуатацію будівель.

Накладні витрати включаються до вартості розробки програми непрямым шляхом – у відсотках до основної заробітної плати розробників. В

даному випадку накладні витрати становлять 40% до основної заробітної плати розробників, що складає 9314 грн.

Результати визначення витрат на розробку програми у вигляді калькуляції кошторисної вартості робіт наведені в таблиці 6.11.

Таблиця 6.11 – Вартість розробки програми

Стаття витрат	Сума (грн.)	Питома вага до підсумку (%)
Основна заробітна плата	23 285	39,77
Додаткова заробітна плата	2328,5	3,98
ЄСВ	5634,97	16,71
Вартість матеріалів	427	0,33
Витрати на ПК	990,4	1,23
Витрати на сервер	458,43	22,08
Накладні витрати	9314	15,90
Разом	42438,3	100,00

За результатами розрахунків, наведених у таблиці 6.6, можна зробити висновок, що найбільшу питому вагу у витратах на розробку програмного продукту займають витрати на оплату праці фахівців.

6.3.8 Розрахунок експлуатаційних витрат

Витрати, пов'язані з експлуатацією програми протягом року, визначають за формулою (6.12):

$$И = S_{\text{екс.кТС}} \cdot T_{\text{кТС}} + \frac{S_{\text{пр}}}{n \cdot T_{\text{с}}}, \quad (6.12)$$

де $S_{\text{екс.кТС}}$ – експлуатаційні витрати, що припадають на 1 ч машинного часу КТС споживача програмного продукту, грн / машино–год;

$T_{\text{ктс}}$ – сумарний машинний час КТС протягом року, потрібний даному споживачеві програми для задач, що вирішуються за її допомогою, машино–год;

S_{pn} – сумарні витрати на розробку програми, грн .;

n – кількість споживачів даної програми;

T_c – термін служби даної програми до її морального зносу, років.

$$И = 5775,57 + \frac{42\,438}{1 \cdot 10} = 10\,019,3 \text{ грн.}$$

6.3.9 Розрахунок витрат на адміністратора

Місячна заробітна плата адміністратора становить 5500 грн., а річний фонд заробітної плати становить:

$$ЗП_{\text{адм}} = 5500 \cdot 12 \cdot 1,22 = 80\,520 \text{ грн.}$$

6.4 Розрахунок капітальних вкладень

Обсяг капітальних вкладень споживача програми K , пов'язаних з її розробкою / купівлею / і впровадженням, визначають за формулою 6.13:

$$K = K_{\text{пп}} + K_{\text{пф}}, \quad (6.13)$$

де $K_{\text{пп}}$ – виробничі витрати, грн .;

$K_{\text{пф}}$ – капітальні вкладення в виробничі фонди, необхідні для впровадження програми, грн.

Виробничі витрати включають витрати, пов'язані з розробкою і налагодженням програмного, математичного та інформаційного забезпечення, проектуванням системи. Виробничі витрати визначають за формулою 6.14:

$$K_{\text{пп}} = \frac{S_{\text{pn}}}{n} + K_{\text{пр.пп}}, \quad (6.14)$$

де S_{pn} – сумарні витрати на розробку або придбання програмного продукту, грн;

n – кількість споживачів даної програми;

$K_{\text{пр.пп}}$ – інші виробничі витрати, грн.

Капітальні вкладення в виробничі фонди включають витрати на придбання, установку, монтаж і налагодження комплексу технічних засобів (КТЗ) з урахуванням можливого використання КТЗ як для вирішення завдань для яких призначена розробляється програма, так і для завдань, що вирішуються за допомогою інших програм. Питомі капітальні вкладення в виробничі фонди, припадають на частку даного програмного продукту визначають за формулою 6.15:

$$K_{\text{ктс}} = \frac{K_{\text{ктс}} \cdot T_{\text{ктс}}}{\Phi_{\text{эф.ктс}}}, \quad (6.15)$$

де $K_{\text{ктс}}$ – витрати на придбання, установку, монтаж і налагодження КТЗ, грн;

$T_{\text{ктс}}$ – машинний час КТЗ, потрібне даному споживачеві програми для завдань, що вирішуються за допомогою даної програми, машино–год / рік;

$\Phi_{\text{эф.ктс}}$ – річний ефективний фонд часу роботи КТЗ, машино–год / рік.

$$K = \frac{42\,438}{1} + \frac{16\,000 \cdot 8\,760}{8\,760} = 58\,438 \text{ грн.}$$

6.5 Оцінка економічної ефективності

Створення даного застосунку направлено на отримання додаткового прибутку, залучення нових клієнтів і збільшення обсягу послуг.

За рахунок продажу підписки на розроблений програмний продукт користувачам (очікується продати 1 500 підписок за мінімальною вартістю 100 гривень за копію).

Прибутком будуть вважатися кошти, отримані від підписок користувачі за виключенням заробітної плати адміністратору та експлуатаційних витрат. Адміністратор буде контролювати правильність функціонування системи та за необхідності вносити зміни.

Прибуток розраховується за формулою 6.16.

$$\Pi = \text{Пост}_{\text{від продажу}} - 3\Pi_{\text{адм}} - \text{И}, \quad (6.16)$$

$$\Pi = 1500 \cdot 100 - 80\,520 - 10\,019,3 = 59\,461 \text{ грн.}$$

Річна економія розраховується за формулою 6.17.

$$\mathcal{E} = \Pi - E_{\text{н}}^a \cdot K, \quad (6.17)$$

де $\mathcal{E}$ – річна економія від впровадження системи, грн .;

Π – річний прибуток від впровадженої системи;

$E_{\text{н}}^a$ – нормативний коефіцієнт ефективності капітальних вкладень в засоби автоматизації;

K – обсяг капітальних витрат, пов'язаних зі створенням ПП.

$$\mathcal{E} = 59\,461 - 0,15 \cdot 58\,438 = 50\,695,3 \text{ грн.}$$

Коефіцієнт ефективності вкладених коштів в програму E і термін їх окупності $T_{ок}$ розраховують по формулам 6.18, 6.19:

$$E = \frac{\Pi}{K}, \quad (6.18)$$

$$T_{ок} = \frac{K}{\Pi}, \quad (6.19)$$

Таким чином:

$$E = 57\,297,4 / 63\,546 = 1.01$$

$$T_{ок} = 63\,546 / 57\,297,4 = 0,98 \text{ року.}$$

Таким чином можна зробити висновок про економічну доцільність розробки даного програмного забезпечення, адже окупність продукту відбувається за 0,98 року, що менше за допустиме значення часу на окупність продукту, яке дорівнює 6,67 року.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенційних небезпек

Тема кваліфікаційної роботи магістера «Дослідження та програмна реалізація методів аналізу контенту для структуризації текстової інформації». Для її виконання потрібне виробниче приміщення в контексті даної роботи використовується офіс

Основними потенційними небезпеками при проведенні робіт в офісі є такі:

- небезпека загоряння у зв'язку з несправністю електричного обладнання, недотримання, або порушення правил протипожежної безпеки обслуговуючим персоналом, що може призвести до пожежі.
- порушення роботи кістково-м'язового апарату внаслідок тривалих статичних навантажень при роботі з ПК.
- неправильні дії персоналу у надзвичайних ситуаціях.
- незадовільні ергономічні характеристики робочого місця внаслідок нераціонального планування робочого місця, що може призвести до механічних травм, уражень електричним струмом та порушень кістково-м'язового апарату;
- негативний вплив недостатнього освітлення робочої зони на зір та продуктивність роботи працюючого, внаслідок несправності освітлювальних приладів або неправильного проектування освітлювальної системи;
- негативний вплив незадовільних параметрів повітряного середовища робочої зони на здоров'я працюючого, внаслідок неправильного проектування системи вентиляції або несправності її несправності;

- негативний вплив підвищеного рівня шуму на психоемоційний стан працюючого, який пов'язаний з використанням застарілої периферійної техніки, кондиціонерів, копіювальної техніки, освітлювальних приладів;
- нервово-психічні перевантаження внаслідок постійного контакту з клієнтами, колегами по роботі, керівництвом при вирішенні робочих питань, які можуть носити конфліктний характер і призвести до емоційного дискомфорту, внутрішнього роздратування, емоційної нестабільності та захворювань нервової системи;
- небезпека ураження електричним струмом, внаслідок недотримання правил електробезпеки або виходу з ладу електроприладів;

7.2 Заходи з забезпечення безпеки

Приміщення офісу, в яких перебувають співробітники галузі управління персоналом, відносяться до приміщень без підвищеної небезпеки ураження електричним струмом.

Обладнання, що використовується в цих приміщеннях є споживачем електроенергії, що живиться від змінного струму 220В від мережі з заземленою нейтраллю, та відноситься до електроустановок до 1000В закритого виконання. За способом захисту людини від ураження електричним струмом відповідає згідно [13].

Згідно [14] виконані такі групи заходів з електробезпеки:

- конструктивні заходи забезпечують захист від випадкового дотику до струмопровідних частин за допомогою їх ізоляції та захисних оболонок.
- згідно з [15] у приладах II класу захисту використовується подвійна ізоляція - електрична ізоляція, що складається з робочої і додаткової ізоляції.

– так як згідно [16]. Электрооборудование специальных установок» офісні приміщення у більшості своїй відносяться до класу пожежонебезпечної зони П-Па (приміщення, в яких містяться тверді горючі речовини), тому передбачений ступінь захисту ізоляції обладнання IP44.

– схемно-конструктивні заходи;

– призначені для забезпечення захисту від ураження електричним струмом при дотику до металевих оболонок, які можуть опинитися під напругою в результаті аварій;

– згідно [17] у приміщеннях галузі управління персоналом влаштовується занулення;

– організаційні заходи;

– експлуатація електроустановок і електроустаткування проводиться відповідно до [18] та [19].

Для запобігання статистичного навантаження при користуванні ПК рекомендовано використовувати перерви в роботі 10 хв. через кожні дві години. Синдром зап'ястного каналу, або тунельний синдром зап'ястя, який може бути наслідком хронічної травми, трапляється у людей внаслідок тривалої роботи з мишею: постійні напруга і здавлювання приводить до мікротравм, здавлювання нерва прилеглими оточуючими тканинами, через що виникає набряк.

Щоб тунельний синдром вас не турбував, потрібно дотримуватися кількох правил організації робочого місця:

– оптимальна висота клавіатури від підлоги – 65-75 см;

– наявність ергономічних і зручних особисто для вас миші і клавіатури;

– можливість регулювання висоти і нахилу клавіатури (відстань від поверхні стола до середини клавіатури – не більше 30 мм, кут підйому клавіатури – від 2° до 15°);

- наявність у клавіатури підставки для рук; - наявність килимка для миші з захистом від тунельного синдрому (спеціальний виступ забезпечує правильне положення кисті);

- наявність стільця або крісла з підлокітниками.

При роботі з мишкою і клавіатурою також слід дотримуватися певних правил. Коли ви набираєте текст, рука повинна бути зігнута в лікті під прямим кутом (90°), а при роботі з мишкою стежте, щоб кисть була прямою і лежала на столі якнайдалі від краю. До речі, час роботи з комп'ютером слід обмежити до дійсно розрахованого.

Щоб попередити тунельний синдром потрібно робити спеціальні вправи для кистей – чим частіше, тим краще. Ці вправи допоможуть поліпшити кровообігу в м'язах і розтягнути їх. Комплекс вправ потрібно повторювати приблизно кожні 45 хвилин, тривалість однієї вправи – 1-2 хв.

Нервове напруження впливає на серцево-судинну систему, збільшуючи артеріальний тиск і частоту пульсу, а також на терморегуляцію організму та емоційні стани працівника. Особливу роль у запобіганні втоми працівників відіграють професійний відбір, організація робочого місця, правильне робоче положення, ритм роботи, раціоналізація трудового процесу, використання емоційних стимулів, впровадження раціональних режимів праці і відпочинку тощо. Боротьба зі втомою, в першу чергу, зводиться до покращення санітарно-гігієнічних умов виробничого середовища (ліквідація забруднення повітря, шуму, вібрації, нормалізація мікроклімату, раціональне освітлення тощо).

Крім того, для профілактики втоми працівників застосовуються специфічні методи, до яких можна віднести засоби відновлення функціонального стану зорового та опорно-рухового апарату, зменшення гіподинамії, підсилення мозкового кровообігу, оптимізацію розумової діяльності.

Загальні ергономічні вимоги встановлено [20]. Організація робочого місця передбачає: правильне розміщення робочого місця у виробничому приміщенні; вибір ергономічно-обґрунтованого робочого положення, виробничих меблів з урахуванням антропометричних характеристик людини; раціональне компонування обладнання на робочих місцях; врахування характеру та особливостей трудової діяльності.

В результаті технічного переоснащення, що базується на впровадженні інформаційних технологій у приміщенні, що знаходиться на другому поверсі виробничого корпусу заплановано встановити комп'ютери. Визначимо скільки комп'ютеризованих робочих місць, оснащених відеодисплейними терміналами (ВДТ) можна встановити у даному приміщенні і як їх розташувати відповідно до встановлених норм та правил з охорони праці. Розміри приміщення: довжина $a = 5,1$ м, ширина $b = 4,1$ м, висота $h = 2,5$ м.

розраховано, що для даного приміщення для того, щоб розмістити в ньому комп'ютеризовані робочі місця.

Відповідно до [21] є неприпустимим розташування приміщень, призначених для роботи з ВДТ у підвалах та цокольних поверхах. Також забороняється розташування вибухонебезпечних приміщень категорії А і Б [22] та виробництв з мокрими технологічними процесами поряд з приміщенням, де розташовуються ЕОМ, а також над такими приміщеннями, або під ними. Окрім того, виробничі приміщення для роботи з ВДТ не повинні межувати з приміщеннями, у яких рівень шуму і вібрації перевищує допустимі значення.

Вибране приміщення відповідає вищезазначеним вимогам, тому переходимо до наступного етапу, а саме, до визначення кількості комп'ютеризованих робочих місць, що можна розмістити в даному приміщенні. Оскільки площа приміщення становить $S_{ПР} = 20,91$ м², а площа, на якій розташовується одне робоче місце з ВДТ, відповідно до вимог

[23] повинна становити не менше $6,0 \text{ м}^2$, то в даному приміщенні можна розмістити щонайбільше три комп'ютеризованих робочих місць. Такої кількості достатньо для технічного переоснащення виробництва.

Перевіримо, чи відповідає це число нормативу щодо мінімального об'єму приміщення на одне робоче місце з ВДТ, відповідно до вимог [23] ($VP.M.Min = 20 \text{ м}^3$). Об'єм приміщення становить $52,275 \text{ м}^3$, а об'єм, що припадає на одне комп'ютеризоване робоче місце - $VP.M. = 17,425 \text{ м}^3$. Таким чином норматив щодо об'єму приміщення на одне робоче місце з ВДТ не виконується. Тому розміщуємо в даному приміщенні щонайбільше два комп'ютеризованих робочих місць.

Планування розміщення комп'ютеризованих робочих місць у приміщенні проводимо із врахуванням наступних вимог:

- робочі місця з ВДТ розміщуються на відстані не менше 1 м від стіни зі світловими прорізами;
- відстань між бічними поверхнями ВДТ має бути не менше за 1,2м;
- відстань між тильною поверхнею одного ВДТ та екраном іншого не повинна бути меншою за 2,5 м;
- прохід між рядами робочих місць має бути не меншим 1 м.

Також розраховано розміри меблів на комп'ютеризованих робочих місцях, зокрема робочого столу. Відповідно до [21] рекомендовані розміри столу для робочого місця з ВДТ становлять: висота – 725 мм, ширина – 600-1400 мм, глибина – 800-1000 мм. Приймаємо, що робочий стіл має такі розміри: ширина – 1200 мм, глибина – 800 мм.

Найкраще розмістити комп'ютеризовані робочі місця рядами вздовж стіни з вікнами. Це дасть змогу унеможливити дзеркальне відбиття на екрані ВДТ джерел природного світла (вікон) та потрапляння останніх у поле зору операторів, що погіршує умови їх зорової роботи.

На (рис. 7.1) наведено план приміщення з комп'ютеризованими робочими місцями.

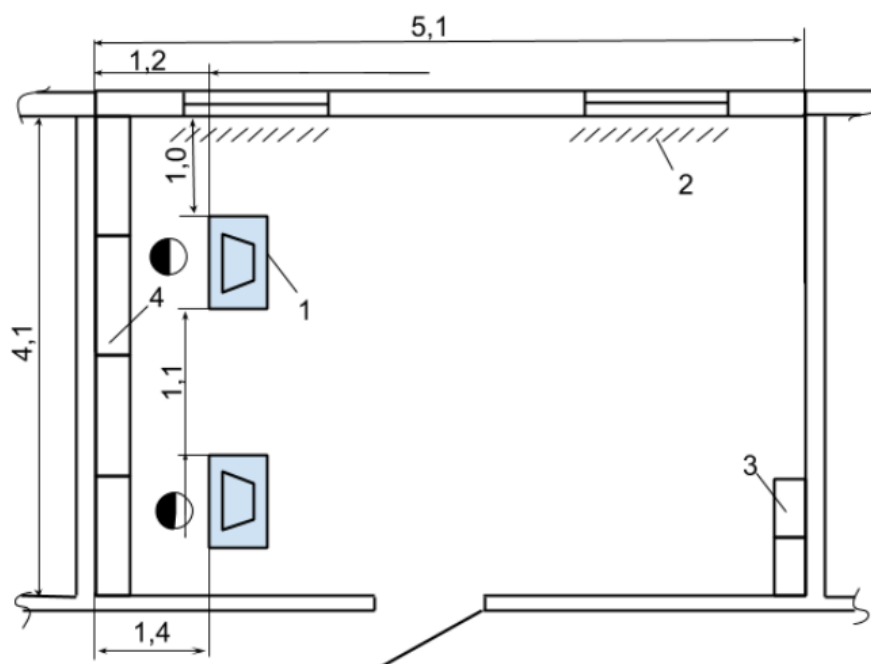


Рисунок 7.1 - План приміщення з комп'ютеризованими робочими місцями: 1 - комп'ютеризоване робоче місце з ВДТ; 2 - сонцезахисні жалюзі; 3 - шафи для зберігання твердотільних накопичувачів та програмного забезпечення; 4 - шафи для зберігання документації та фахової літератури.

7.3 Заходи з виробничої санітарії та гігієни праці

Згідно [24], в офісних приміщеннях використовується природне і штучне освітлення. Природне освітлення здійснюється через світлові щілини, що забезпечують коефіцієнт природного освітлення (КПО) не менше 1,5%. Для захисту від прямих сонячних променів, які створюють прямі та відбиті відблиски на поверхні екранів і клавіатур, використовуються сонцезахисні пристрої, на вікнах встановлюються жалюзі або штори. Штучне освітлення в приміщенні проводиться системою загального рівномірного освітлення. Величина освітленості на поверхні робочого столу в області

документів

300-500 лк. Люмінесцентні лампи LB використовуються як джерела штучного освітлення в приміщенні. Показники мікроклімата в офісних приміщеннях відповідають встановленим санітарно-гігієнічним вимогам [25-27].

Роботи в приміщенні туристичного обслуговування відносяться до категорії Ів - легкі роботи, тому встановлені наступні оптимальні значення параметрів мікроклімата:

- в холодний період року: температура 22-24 °С; відносна вологість: 40-60%; швидкість руху повітря: 0,1 м/с;
- в теплий період року: температура 23-25 °С; відносна вологість: 40-60%; швидкість руху повітря: 0,2 м/с.

Оптимальні рівні укладальних (n+) та негативних (n-) іонів у повітряних приміщеннях з ПК нормовані згідно [27] і складають: n+=1500-30000 (шт. на 1 см³); n-=3000-5000 (шт. на 1 см³). Підтримування оптимального рівня легких позитивних і негативних аероіонів у повітрі на робочому місці забезпечує біполярні коронні іонізатори повітря.

Відповідність вимогам цих документів досягається за рахунок оснащення приміщень та транспортних засобів систем кондиціонування та вентиляції, дезодорації повітря, опалення відповідно до вимог [28].

Рівні звукових даних в октавних полосах, рівні звуку та еквівалентні рівні звуку на робочому місці в приміщенні нормуються відповідно до [23,29].

Зниження рівня шуму в приміщенні здійснюється за допомогою:

- використання більш сучасного обладнання;
- розміщення принтерів та різного колективного обладнання на значущому видаленні від більшості робочих місць;

- перевести жорсткого диска в сплячий режим (режим очікування), якщо комп'ютер не працює певний час;
- використання блоків харчування ПК з вентиляторами на резинових підвісках.

7.4 Заходи безпеки у надзвичайних ситуаціях

7.4.1 Заходи з пожежної безпеки

Закон України «Про пожежну безпеку» визначає загальні правові, економічні та соціальні основи пожежної безпеки в Україні, регулює відносини державних органів, юридичних і фізичних осіб у цій сфері незалежно від їх виду діяльності і форм власності [30].

Пожежна безпека - стан об'єкта, при якому з регульованою ймовірністю виключена можливість виникнення і розвитку пожежі та впливу на людей її небезпечних факторів, а також забезпечена охорона матеріальних цінностей.

Для забезпечення пожежної безпеки в установах проводиться протипожежна профілактика, яка включає комплекс організаційних і технічних заходів, спрямованих на забезпечення безпеки людини, запобігання пожежі, обмеження її розповсюдження, а також створення умов для успішного гасіння пожежі.

Засоби первинного пожежогасіння застосовуються для ліквідації пожежі на початковій стадії його розвитку персоналом об'єктів. До них відносяться: вогнегасники, протипожежне обладнання (ковдри з негорючого утеплювача, пісочниці, пожежні відра, лопати, ломи, сокири та ін.), Автоматичні системи пожежогасіння.

Первинні вогнегасники залежно від категорії приміщення можуть розташовуватися як окремо, так і в складі протипожежних щитів.

Залежно від фізичного стану і особливостей горіння різних горючих речовин і матеріалів пожежі згідно [31] діляться на відповідні класи.

В офісі дерев'яні меблі, електронне обладнання, паперові носії.

Клас пожежі в офісному приміщенні згідно [32] - пожежі твердих тіл, переважно органічного походження, горіння яких супроводжується тлінням (дерево, пластмаса, папір), визначається як клас А. Категорія приміщень згідно [33] визначається як категорія Д.

Визначення типу і розрахунок кількості основних засобів пожежогасіння згідно [32] - для адміністративного приміщення площею 20,91 м² слід використовувати два порошкових вогнегасника типу ВП-5 [31, 33]. Крім того, адміністративні приміщення повинні бути обладнані автоматичними пожежними сповіщувачами, що реагують на підвищення температури, дим і полум'я. Наприклад, детектори моделей DTL, ВМІ.

7.4.2 Заходи з цивільного захисту

Ударна хвиля - це область різкого стиску середовища, що поширюється у вигляді сферичного шару від місця вибуху з надзвуковою швидкістю. Ударні хвилі класифікуються залежно від середовища поширення. Ударна хвиля в повітрі виникає через передачу стиснення і розширення повітряних шарів. У міру віддалення від місця вибуху хвиля слабшає і переходить в нормальну акустичну. Хвиля, що проходить через задану точку простору, викликає зміна тиску, яке характеризується наявністю двох фаз: стиснення і розширення. Період стиснення настає негайно і триває відносно короткий час у порівнянні з періодом розширення. Деструктивну дію ударної хвилі характеризується надлишковим тиском на її фронті (передній кордоні), швидкісним напором, тривалістю фази стиснення. Ударна хвиля у воді відрізняється від повітря значеннями своїх характеристик (велике надлишковий тиск і менший час впливу).

Ударна хвиля в землі при видаленні від місця вибуху стає схожою на сейсмічну хвилю. Вплив ударної хвилі на людей і тварин може привести до прямих або непрямих пошкоджень. Для нього характерні легкі, середньо важкі, важкі і у край важкі травми і травми. Вражаючий ефект ударної хвилі характеризується величиною надлишкового тиску. Надлишковий тиск - це різниця між максимальним тиском у фронті ударної хвилі та нормальним атмосферним тиском перед ним.

При надлишковому тиску 20-40 кПа незахищені люди можуть отримати легкі ураження (легкі синці та контузії). Вплив ударної хвилі з надлишковим тиском 40-60 кПа призводить до помірних уражень: втрати свідомості, пошкодження органів слуху, сильних вивихів кінцівок, кровотечі з носа та вух. Важкі травми трапляються при надлишковому тиску більше 60 кПа. Надзвичайно важкі ураження спостерігаються при надлишковому тиску більше 100 кПа.

Ударна хвиля ядерного вибуху може перемогти людей, зруйнувати будівлі та пошкодити військову техніку на значній відстані від центру вибуху. Ударна хвиля - це область сильного стиснення повітря, яка поширюється з великою швидкістю в усіх напрямках від центру вибуху. Швидкість його поширення залежить від тиску повітря в передній частині ударної хвилі; поблизу центру вибуху це в кілька разів перевищує швидкість звуку, але зі збільшенням відстані від місця вибуху різко падає.

Вражаюча дія ударної хвилі на людей та руйнівний вплив на військову техніку, інженерні споруди та матеріальні ресурси в першу чергу визначаються надмірним тиском та швидкістю руху повітря в її фронті. Незахищені люди також можуть бути поранені осколками скла, що летять з великою швидкістю, та уламками руйнуються будівель, падіння дерев, а також розсипанням шматків військової техніки, каміння та інших предметів, які приводиться в рух швидкісною ударною хвилею. Найбільші непрямі поразки спостерігаються в населених пунктах та в лісі; у цих випадках втрата

військ може бути більшою, ніж від прямої дії ударної хвилі. Ударна хвиля може завдати шкоди всередині приміщення, проникаючи туди через тріщини та отвори. Пошкодження, спричинені ударною хвилею, поділяються на легкі, середні, важкі та дуже важкі. Легкі ураження характеризуються тимчасовим ураженням органів слуху, загальний легкий удар, удари і вивихи кінцівок. Важкі поразки характеризуються сильним забоєм всього тіла; можуть бути ураження головного мозку і органів черевної порожнини, сильна кровотеча з носа і вух, важкі переломи і вивихи кінцівок. Ступінь поразки ударною хвилею залежить передусім від потужності і типу ядерного вибуху.

Механічний вплив ударної хвилі оцінюється за ступенем руйнування, викликаного дією хвилі (бувають слабке, середнє, сильне і повне руйнування). Енергетичне, промислове і комунальне устаткування в результаті ударної хвилі може бути пошкоджено, що також оцінюється за ступенем їх серйозності (слабка, середня і сильна). Вплив ударної хвилі також може привести до пошкодження транспортних засобів, гідротехнічних споруд, лісів. Як правило, збитки від впливу ударної хвилі дуже великий, він наноситься здоров'ю людини, а також різноманітних конструкцій, обладнання і т.д.

Середнє руйнування проявляється в руйнуванні покрівлі та вбудованих елементів, внутрішніх перегородок, вікон, а також у виникненні тріщини у стінах, обвалення окремих ділянок горищних поверхів і стін верхніх поверхів. Підвали збереглися. Після очищення і ремонту частина нижніх поверхів можна використовувати. При капітальному ремонті можливе відновлення будівель.

Для сильного руйнування характерно руйнування несучих конструкцій і перекриттів верхніх поверхів, утворення тріщини в стінах і деформація підлоги нижніх поверхів. Використання приміщення стає неможливим, а ремонт і реконструкція часто недоцільні.

Повне руйнування. Всі основні елементи будівлі зруйновані, в тому числі несучі конструкції. Використовувати споруди неможливо. Підвали в разі сильного і повного руйнування можна зберегти, а після демонтажу завалів частково використовувати.

ВИСНОВКИ

В кваліфікаційній роботі магістра виконано дослідження проблеми аналізу контенту та структуризації текстової інформації. Виконано огляд програмних засобів, які використовуються для розв'язання проблеми на даний момент. Виконано аналіз методів, які використовуються в даних системах для аналізу та структуризації текстової інформації.

В якості мови програмування для реалізації програмного забезпечення обрано мову програмування JavaScript, в якості бібліотеки аналізу контенту – бібліотеку XLSX, docx-parser, а для забезпечення роботи з базою даних використано систему керування базами даних MongoDB. В якості підходу для реалізації процесу структуризації текстової інформації використано Telegram Bot API та бібліотеку для роботи за API на мові JS - Telegraf. Розроблено діаграми прецедентів для роботи з системою, структурну схему програми, алгоритм функціонування програми. Архітектуру програми побудовано на основі node.js. Описано рішення, прийняті під час реалізації програмного забезпечення.

У результаті програмної реалізації розроблено чат-бота, здатного отримувати документи та аналізувати їх для вилучення текстової інформації, щоб зібрати базу знань, та яку в подальшому можна використовувати для формування списку запитань та надавання для них відповідей. Практичною сферою застосування є – перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

Наукова новизна роботи полягає в запропонованому методі для структуризації текстової інформації для наповнення БД, що використовує чат-бот Telegram для крос-платформної, швидкої, легкодоступної та захищеної взаємодії з користувачем.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Рабочая книга социолога / Отв. ред. Г. В. Осипов. — М.: Наука, 1976. — С. 322
2. Соковнин В. М. Об объективности исследования в контент-анализе // Методологические и методические проблемы контент-анализа: (Тезисы докладов рабочего совещания социологов) / АН СССР. Институт социологических исследований; Отв. ред. А. Г. Здравомыслов. — М.; Л., 1973. — Вып. 1. — С. 59.
3. Презентация на тему Лекция 11 “Основы концепции “глубинного анализа текстов” (Text Mining)” [Электрон. ресурс]. — Режим доступа: <https://thepresentation.ru/uncategorized/307591-lektsiya-11%E2%80%99Cosnovy-kontseptsii%E2%80%99Cglubinnogo-analiza-tekstov%E2%80%99Dtext-mining%E2%80%99D>
4. Федорчук, А. Г. Контент-мониторинг информационных потоков [Электрон. ресурс] // Б-ки нац. акад. наук: пробл. функционирования, тенденции развития. — Электрон. дан. (1 файл). — К., 2005. — Вып. 3. — Режим доступа: <http://www.nbuv.gov.ua/articles/2005/05fagmip.html>. — Назва з екрана.
5. Алексеев, А.Н. Контент-анализ в социологии и точки соприкосновения с другими отраслями знания / А.Н. Алексеев // Проблемы контент-анализа в социологии: М-лы Сибирского социологического семинара — Новосибирск, 1970. — С.11–12.
6. Иванов, В.Ф. Контент-аналіз: Методологія і методика дослідження ЗМК: Навч. посібник / Наук. ред. А.З. Москаленко / В.Ф. Иванов. — К., 1994. — 112 с. 3. Литвин В. В. Інтелектуальні системи / Литвин В. В., Пасічник В. В., Яцишин Ю.В. — Львів: “Новий Світ — 2000”, 2009. — 406 с.
7. Пиотровский, Р. Г. Математическая лингвистика: Учебное пособие / Пиотровский Р. Г., Бектаев К. Б., Пиотровская А. А. — М. : Высшая школа, 1977. — 384 с.

8. Сорока, М.Б. Використання методу контент-аналізу при створенні автоматизованих інформаційних систем / М.Б. Сорока, Н.В. Танатар // Бібліотека. Наука. Культура. Інформація: Наукові праці НБУВ. – 1998. – Вип. 1. – С. 318–322.
9. Пэнто Р., Гравитц М., Методы социальных наук. М., 1972.
10. Mayring Philipp, Qualitative Inhaltsanalyse: Grundlagen und Techniken, Weinheim, 1994.
11. The Advantages and Disadvantages of JavaScript [Electronic resource]. – Access mode: <https://www.freecodecamp.org/news/the-advantages-and-disadvantages-of-javascript/>
12. Какие бывают NoSQL-СУБД: основные типы нереляционных баз данных [Электрон. ресурс]. – Режим доступа: <https://www.mindinventory.com/blog/pros-and-cons-of-node-js-web-app-development/>
13. ГОСТ 12.2.007.0-75* (2001) «ССБТ. Изделия электротехнические. Общие требования безопасности» I (стаціонарні комп'ютери,) та II (освітлювальні прилади, кондиціонери, опалювальні пристрої, ноутбуки, сканери) класу захисту [Электрон. ресурс]. – Режим доступа: <http://docs.cntd.ru/document/1200008440>
14. Правилам улаштування електроустановок [Электрон. ресурс]. – Режим доступа: <https://art-energetyka.com.ua/Правила-улаштування-электроустановок>
15. ГОСТ 12.1.009-76 (1999) «ССБТ. Электробезопасность. Термины и определения» [Электрон. ресурс]. – Режим доступа: <http://docs.cntd.ru/document/5200278>
16. НПАОП 40.1-1.32-01 «Правила устройства электроустановок [Электрон. ресурс]. – Режим доступа: https://dnaop.com/html/1692/doc-%D0%9D%D0%9F%D0%90%D0%9E%D0%9F_40.1-1.32-01

17. ГОСТ 12.1.030-81 (2001) «ССБТ. Электробезопасность. Защитное заземление, зануление» [Электрон. ресурс]. – Режим доступа: <http://docs.cntd.ru/document/5200289>

18. НПАОП 40.1-1.01-97 «Правила безпечної експлуатації електроустановок» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/1691/doc-%D0%9D%D0%9F%D0%90%D0%9E%D0%9F_40.1-1.01-97

19. НПАОП 40.1-1.21-98 «Правила безпечної експлуатації електроустановок споживачів» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/2029/doc-%D0%9D%D0%9F%D0%90%D0%9E%D0%9F_40.1-1.21-98

20. ДСТУ ISO 9241-1:2003 «Ергономічні вимоги до роботи з відеотерміналами в офісі. Частина 1. Загальні положення» [Електрон. ресурс]. – Режим доступу: http://online.budstandart.com/ua/catalog/doc-page?id_doc=53294

21. НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/31562/doc-%D0%9D%D0%9F%D0%90%D0%9E%D0%9F_0.00-1.28-10

22. НАПБ Б.03.002-2007 «Нормы определения категорий помещений, зданий и наружных установок по взрывопожарной и пожарной безопасности» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/32350/doc-%D0%9D%D0%90%D0%9F%D0%91_%D0%91.03.002-2007

23. ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [Електрон. ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/rada/show/v0007282-98#Text>

24. ДБН В.2.5-28:2018 «Природне і штучне освітлення» [Електрон. ресурс]. – Режим доступу: https://ledeffect.com.ua/images/__branding/dbn2018.pdf
25. ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень» [Електрон. ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/rada/show/va042282-99#Text>
26. ГОСТ 12.1.005-88 (1991) «ССБТ. Общие санитарно-гигиенические требования к воздуху рабочей зоны» [Электрон. ресурс]. – Режим доступа: <http://docs.cntd.ru/document/1200003608>
27. ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/2296/doc-%D0%93%D0%9D_2152-80
28. ДБН В.2.5-67:2013 «Опалення, вентиляція та кондиціонування» [Електрон. ресурс]. – Режим доступу: <https://dbn.co.ua/load/normativy/dbn/1-1-0-1018>
29. ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку». [Електрон. ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/rada/show/va037282-99#Text>
30. Закон України «Про пожежну безпеку» [Електрон. ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/laws/show/3745-12#Text>
31. ДБН В.1.1.7-2002 «Пожежна безпека об'єктів будівництва» [Електрон. ресурс]. – Режим доступу: http://online.budstandart.com/ru/catalog/doc-page?id_doc=5009
32. ДБН В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об'єктів будівництва» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/3608/doc-%D0%94%D0%91%D0%9D_%D0%92.1.1.7-2002

33. НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» [Електрон. ресурс]. – Режим доступу: https://dnaop.com/html/32980/doc-%D0%9D%D0%90%D0%9F%D0%91_%D0%91.03.002-2007

ДОДАТОК А
Текст програми

A.1 Текст package.json

```
{
  "name": "chatbot_diplom",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "nodemon src/index.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "axios": "^0.21.0",
    "config": "^3.3.3",
    "mongodb": "^3.6.3",
    "node-telegram-bot-api": "0.50.0",
    "node-telegram-keyboard-wrapper": "^2.0.1",
    "nodemon": "^2.0.6",
    "telegraf": "^3.38.0",
    "xlsx": "^0.16.9"
  }
}
```

A.2 Текст index.js

```
const { handleStartCommand } = require('./scenes/main/commands');
const { Telegraf, Context } = require('telegraf');
const {
  session,
} = Telegraf;
const config = require('config');
const connectDb = require('./db/connect.js');

const stages = require('./stages');
const { goBack } = require("./utils.js");

let db = {};

connectDb().then(
  (value) => {
    db = value;

    bot.launch();
  }
);

class CustomContext extends Context {
  constructor(update, telegram, options) {
    super(update, telegram, options)
  }

  async reply(...args) {
```

```

    return super.reply(...args)
  }

  goBack = (scene) => goBack(this, scene);

  db = db;
}

const bot = new Telegraf(config.get('token'), { contextType: CustomContext });

bot.use(session());
bot.use(stages);

bot.use(
  async (ctx, next) => {
    ctx.session.previousScene = "";
    await next()
  }
);

bot.settings(async (ctx) => {
  await ctx.setMyCommands([
    {
      command: '/start',
      description: 'Запускає бота'
    },
    {
      command: '/echo',
      description: 'Перевіряє з\'єднання'
    },
  ]);
  return ctx.reply('Ok');
});

bot.help(async (ctx) => {
  const commands = await ctx.getMyCommands();
  const info = commands.reduce((acc, val) => `${acc}/${val.command} - ${val.description}\n`, '');
  return ctx.reply(info);
});

bot.start(handleStartCommand);
bot.command('echo', (ctx) => ctx.reply('Echo'));

```

A.3 Тест services.js

```

const axios = require('axios');
const config = require('config');

const getFile = async (path) => {
  try {
    const { data } = await
    axios.get(`https://api.telegram.org/file/bot${config.get('token')}/${path}`);

    return data
  } catch (e) {

```

```

        throw e
    }
};
const getFilePath = async (id) => {
    try {
        const {data: { result }} = await
        axios.get(`https://api.telegram.org/bot${config.get('token')}/getFile?file_id=${id}`
        );

        return result
    } catch (e) {
        throw e
    }
};

module.exports = {
    getFile,
    getFilePath
}

```

A.4 Tekst stages.js

```

const { Telegraf } = require('telegraf');
const sceneGenerator = require('./scenes');
const {
    Stage,
} = Telegraf;

const scenes = sceneGenerator();

const stages = new Stage(scenes);

module.exports = stages.middleware();

```

A.5 Tekst utils.js

```

const XLSX = require('xlsx');
const { allowedDocuments } = require("./constants.js");

const transformToJson = async ({ file, type }) => {

    switch (type) {
        case allowedDocuments[0]: {
            parseXLSX(file);

            break;
        }
        case allowedDocuments[1]: {
            break;
        }
        default:
            return {};
    }
}

```

```

};

const parseXLSX = async (file) => {
  const XLSX = require('xlsx');
  const workbook = await XLSX.read(file, { type: 'array' });
  const sheet_name_list = workbook.SheetNames;
  const data = [];
  sheet_name_list.forEach(function(y) {
    const worksheet = workbook.Sheets[y];
    const headers = {};
    for(z in worksheet) {
      if(z[0] === '!') continue;
      //parse out the column, row, and value
      const col = z.substring(0,1);
      const row = parseInt(z.substring(1));
      const value = worksheet[z].v;
      //store header names
      if(row == 1) {
        headers[col] = value;
        continue;
      }
      if(!data[row]) data[row]={};
      data[row][headers[col]] = value;
    }
    //drop those first two rows which are empty
    data.shift();
    data.shift();
  });
  return data
}

const clearPreviousScene = async (ctx) => {
  if (ctx.update.hasOwnProperty('message')) {
    const id = ctx.update.message.message_id;
    await ctx.deleteMessage(id);
  }
  //
  if (ctx.update.hasOwnProperty('callback_query')) {
    const id = ctx.update.callback_query.message.message_id;
    await ctx.deleteMessage(id);
  }
};

const goBack = async (ctx, scene) => {
  await ctx.scene.enter(scene);
  await ctx.answerCbQuery('Вы вернулись обратно');
};

const useBack = (scene) => [
  'callback_query',
  async (ctx, next) => {
    const action = ctx.callbackQuery.data;

    console.log({ action, });

    if (action === 'back') {
      await ctx.goBack(scene);
    }
  }
];

```

```

        } else {
            next();
        }
    }
};

module.exports = {
    transformToJson,
    goBack,
    useBack,
    clearPreviousScene
};

```

A.6 Текст constants.js

```

const emojiIcons = require("../icons.js");
const backKey = [
    {
        text: emojiIcons.back + ' Повернутися',
        callback_data: 'back'
    }
];

const allowedDocuments = [
    'application/vnd.openxmlformats-officedocument.spreadsheetml.sheet',
    'application/vnd.openxmlformats-officedocument.wordprocessingml.document'
]

module.exports = {
    backKey,
    allowedDocuments
}

```

A.7 Текст sceneGenerator.js

```

const main = require('./main');
const auth = require('./auth');
const addAdmin = require('./addAdmin.js');
const scanTable = require('./scanTable');
const getTables = require('./tables/getTables');
const deleteTables = require('./tables/deleteTables');
const answerByQuestions = require('./answerByQuestions.js');
const stats = require('./stats.js');
const { clearPreviousScene } = require("../utils.js");

const scenes = [
    main,
    auth,
    scanTable,
    addAdmin,
    deleteTables,
    answerByQuestions,
    stats,
    getTables
];

```

```

const sceneGenerator = () => {

  return scenes.map(
    scene => scene().leave(
      async (ctx) => {
        ctx.session.previousScene = scene().id;

        await clearPreviousScene(ctx);
      }
    )
  );
};

module.exports = sceneGenerator;

```

A.8 Текст сцени answerByQuestions.js

```

const Scene = require('telegraf/scenes/base');
const { useBack } = require("../utils.js");
const { backKey } = require("../constants.js");

const answers = [[' - Інформаційні технології;', ' - Механічна;', ' - Інженерія;',
' - Електрична інженерія;', ' - Автоматизація та приладобудування;', ' - Електроніка
та телекомунікації;', ' - Архітектура та будівництво;', ' - Транспорт.']]

const keyboard = [
  [
    {
      text: 'Перелік спеціальностей',
      callback_data: 0
    }
  ],
  [
    {
      text: 'Таблиця переведення середнього бала',
      callback_data: 1
    }
  ],
  [
    {
      text: 'Перелік конкурсних предметів у сертифікаті(ах) Українського центру
оцінювання якості освіти',
      callback_data: 2
    }
  ],
  [
    {
      text: 'Перелік спеціальностей та вступних випробувань для прийому на навчання
для здобуття освітнього ступеня магістра на основі ступенів вищої освіти',
      callback_data: 3
    }
  ],
  [
    {
      text: 'Перелік освітніх ступенів та спеціальностей, за якими оголошується
прийом на навчання',

```

```

        callback_data: 4
      }
    ],
    backKey,
  ];

const opts = {
  reply_markup: {
    inline_keyboard: keyboard
  }
};

const answerByQuestions = () => {
  const answerByQuestions = new Scene('answerByQuestions');
  answerByQuestions.on(...useBack('main'));
  answerByQuestions.enter(async (ctx) => {

    return await ctx.reply('Оберіть запитання', opts);
  });

  answerByQuestions.on(
    'callback_query',
    async (ctx) => {
      const fileId = ctx.callbackQuery.data;
      console.log(`ctx.callbackQuery`, answers[fileId]);

      ctx.replyWithMarkdown(`Відповідь на запитання:
***Перелік спеціальностей***\n${answers[fileId].join('\n')}`)

    }
  );

  answerByQuestions.on(
    'message',
    (ctx) => ctx.scene.reenter()
  )

  return answerByQuestions;
};

module.exports = answerByQuestions;

```

A.9 Текст сцени addAdmin.js

```

const Scene = require('telegraf/scenes/base');
const { useBack } = require("../utils.js");
const { backKey } = require("../constants.js");

const keyboard = [
  backKey,
];

const opts = {

```

```

    reply_markup: {
      inline_keyboard: keyboard
    }
  };

const addAdmin = () => {
  const addAdmin = new Scene('addAdmin');

  addAdmin.enter(async (ctx) => {
    await ctx.reply('Для того щоб додати нового адміністратора, поділіться зі мною контактом.', opts)
  });

  addAdmin.on(
    'contact',
    async (ctx) => {
      const { user_id } = ctx.message.contact;

      const hasAdmin = await ctx.db.collection('admins').findOne({ userId: user_id });

      if(!hasAdmin) {
        await ctx.db.collection('admins').insertOne({ userId: user_id })
        await ctx.reply('Вы успішно додали нового адміністратора.');
```

```
      } else {
        await ctx.reply('Даний користувач вже є адміністратором!');
```

```
      }
      await ctx.reply('Якщо бажаєте додати ще, поділіться зі мною контактом.', opts)
    }
  );

  addAdmin.on(...useBack('auth'));

  addAdmin.on(
    'message',
    (ctx) => ctx.reply('Краще контакт')
  );

  return addAdmin;
};

module.exports = addAdmin;
```

A.10 Текст сцени auth.js

```

const Scene = require('telegraf/scenes/base');
const keyboard = require('./keyboard.js');
const { useBack } = require("../utils.js");
```

```

const genAuthScene = () => {
  const auth = new Scene('auth');

  const opts = {
    reply_markup: {
```

```

        inline_keyboard: keyboard
    }
};

auth.enter(async (ctx) => {
    await ctx.reply('Оберіть дію:', opts)
});

auth.on(...useBack('main'));

auth.on('callback_query', async (ctx) => {
    const action = ctx.callbackQuery.data;

    try {
        await ctx.scene.enter(action)
    } catch (e) {
        await ctx.answerCbQuery('Дана сцена ще в розробці');
    }
});

auth.on('message', async (ctx) => {
    await ctx.scene.reenter()
});

return auth
};

module.exports = genAuthScene;

```

A.11 Текст сцени main.js

```

const Scene = require('telegraf/scenes/base');
const keyboard = require('../commands/start/keyboard.js');

const genMainScene = () => {
    const main = new Scene('main');
    const opts = {
        reply_markup: {
            inline_keyboard: keyboard
        }
    };

    main.enter(async (ctx) => {
        await ctx.reply('Оберіть дію:', opts)
    });

    main.on('callback_query', async (ctx) => {
        const scene = ctx.callbackQuery.data;

        try {
            switch (scene) {
                case 'auth': {
                    const { id: userId } = await ctx.getChat();
                    const isOwner = await ctx.db.collection('owners').findOne({ userId });
                    const isAdmin = await ctx.db.collection('admins').findOne({ userId });

```

```

        if (isOwner || isAdmin) {
            await ctx.scene.enter(scene);
        } else {
            await ctx.answerCbQuery('Ви не є адміністратором або власником!');
        }

        break;
    }

    default: {
        await ctx.scene.enter(scene);
        break;
    }
}
await ctx.answerCbQuery();
} catch (e) {
    await ctx.answerCbQuery('Упс, щось пішло не так!');
}

});

main.on('message', async (ctx) => {
    await ctx.scene.reenter();
});

return main
};

module.exports = genMainScene;

```

A.12 Текст сцени scanTable.js

```

const { getFilePath, getFile } = require('../services.js');
const { transformToJson } = require('../utils.js');

const Scene = require('telegraf/scenes/base');

const { useBack } = require("../utils.js");
const { backKey, allowedDocuments } = require("../constants.js");

const keyboard = [
    backKey,
];

const opts = {
    reply_markup: {
        inline_keyboard: keyboard
    }
};

const scanTable = () => {
    const add = new Scene('scanTable');
    add.enter(async (ctx) => {

```

```

    await ctx.replyWithMarkdown('Надішліть мені документ\n***Підтримувані формати (xlsx, docx)***', opts)
  });

  add.on(
    'document',
    async (ctx) => {
      const {
        file_id: id,
        file_name: name,
        mime_type
      } = ctx.message.document;

      if(!allowedDocuments.includes(mime_type)) {
        await ctx.reply('Даний формат документу не підтримується\nСпробуйте ще раз');
        await ctx.scene.reenter()
      }

      const { file_path } = await getFilePath(id);
      const file = await getFile(file_path);

      const parsed = await transformToJson({file, type: mime_type});

      const collection = ctx.db.collection('tables');

      await collection.insertOne({ name, fileId: id });

      await ctx.reply(`Файл ${name} успішно завантажений`);
      await ctx.reply(`Можна завантажити ще:`, opts)
    }
  );

  add.on(...useBack('auth'));

  add.on(
    'message',
    (ctx) => ctx.reply('Краще таблицю')
  );

  return add;
};

module.exports = scanTable;

```

A.12 Текст сцени getTables.js

```

const Scene = require('telegraf/scenes/base');
const emojiIcons = require("../icons.js");
const { useBack } = require("../utils.js");
const { backKey } = require("../constants.js");

const keyboardWithoutDocuments = [
  backKey,
];

```

```

const keyboard = [
  [
    {
      text: 'Перейти в режим видалення ' + emojiIcons.basket,
      callback_data: 'deleteTables'
    }
  ],
  backKey,
];

const opts = {
  reply_markup: {
    inline_keyboard: keyboard
  }
};

const getTables = () => {
  const getTables = new Scene('getTables');

  getTables.enter(async (ctx) => {

    const collection = ctx.db.collection('tables');
    const cursor = await collection.find().sort({ name: -1 }).project({ _id: 0,
name: 1 }).limit(10);

    if ((await cursor.count()) === 0) {
      const opts = {
        reply_markup: {
          inline_keyboard: keyboardWithoutDocuments
        }
      };
      return await ctx.reply('Документів не знайдено!', opts);
    }

    const list = [];

    await cursor.forEach(
      ({ name }) => list.push(name)
    );

    const tables = list.reduce((acc, item, index) => `${ acc }${ index + 1 }. ${
item }\n`, "");

    await ctx.replyWithMarkdown(`***Список завантажених таблиць:***\n\n${ tables
}`);

    return await ctx.reply('Оберіть дію:', opts);
  });

  getTables.on(...useBack('auth'));

  getTables.on('callback_query', async (ctx) => {
    const action = ctx.callbackQuery.data;

    await ctx.scene.enter(action)
  });

  return getTables;
};

```

```
module.exports = getTables;
```

A.12 Текст сцени getTables.js

```
const Scene = require('telegraf/scenes/base');
const { useBack } = require("../utils.js");
const { backKey } = require("../constants.js");

const deleteTables = () => {
  const deleteTables = new Scene('deleteTables');
  deleteTables.on(...useBack('auth'));

  deleteTables.enter(async (ctx) => {

    const collection = ctx.db.collection('tables');
    const cursor = await collection.find().sort({ name: -1 }).project({ _id: 1,
      fileId: 1, name: 1 }).limit(10);

    const list = [];

    await cursor.forEach(
      (value) => list.push(value)
    );

    if ((await cursor.count()) === 0) {
      const opts = {
        reply_markup: {
          inline_keyboard: [backKey]
        }
      };
      return await ctx.reply('Документів не знайдено!', opts);
    }

    const keyboard = list.map(
      (value) => [
        {
          text: value.name,
          callback_data: value.name,
        },
      ]
    );

    keyboard[keyboard.length] = backKey;

    const opts = {
      reply_markup: {
        inline_keyboard: keyboard
      }
    };

    return await ctx.reply('Яку таблицю бажаєте видалити?\n', opts);
  });

  deleteTables.on(
```

```

    'callback_query',
    async (ctx) => {
        const fileId = ctx.callbackQuery.data;

        const collection = ctx.db.collection('tables');
        await collection.deleteOne({ name: fileId });
        await ctx.answerCbQuery('Таблиця успішно була видалена');

        await ctx.scene.reenter()
    }
);

deleteTables.on(
    'message',
    (ctx) => ctx.scene.reenter()
)

return deleteTables;
};

module.exports = deleteTables;

```

ДОДАТОК Б
Слайди презентації

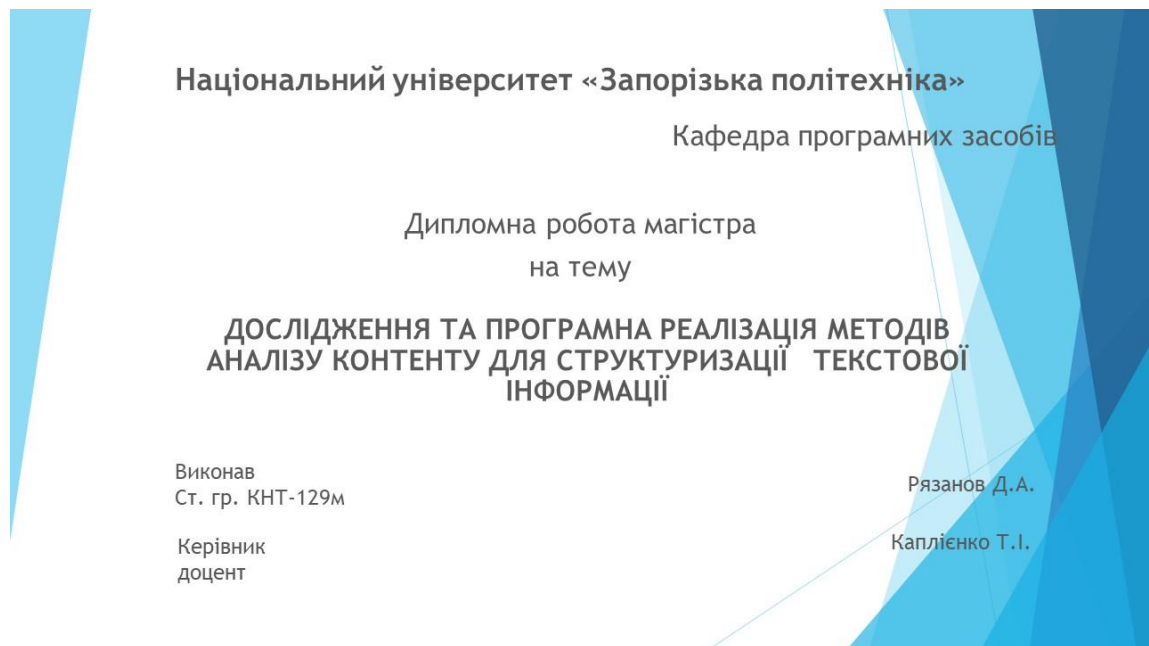


Рисунок Б.1 – Слайд 1

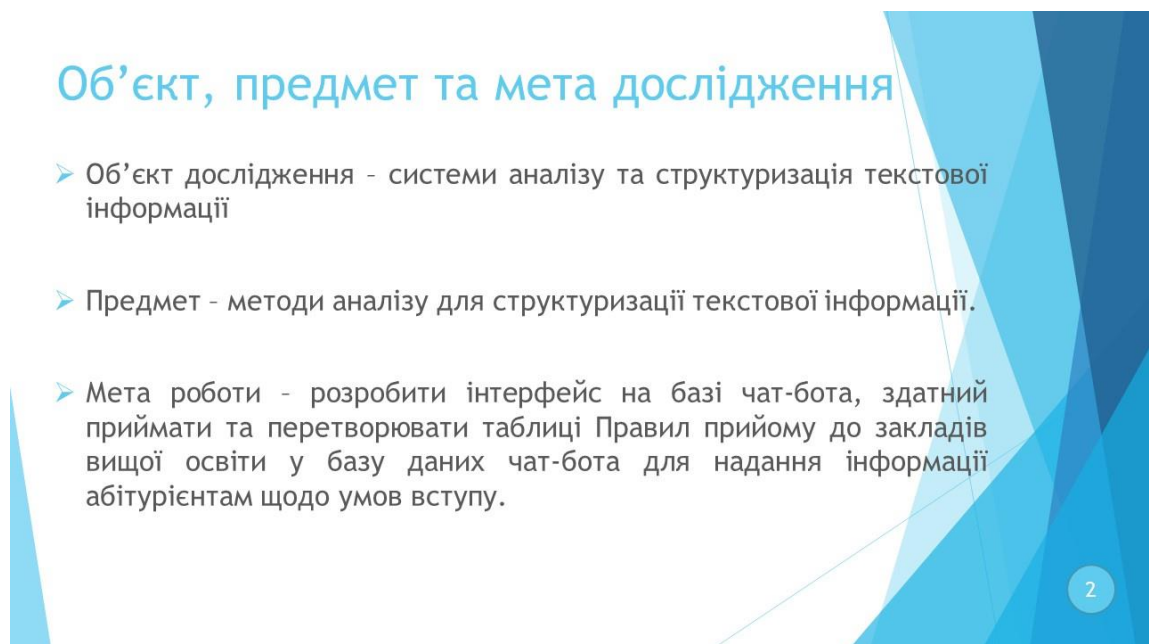


Рисунок Б.2 – Слайд 2

Типи аналізу контенту:



3

Рисунок Б.3 – Слайд 3

Огляд існуючих засобів для аналізу контенту

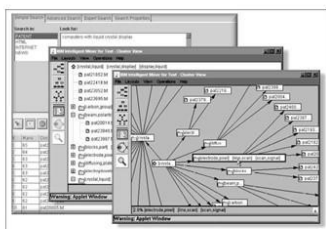


Рисунок 1 - Інтерфейс
Intelligent Miner for text (IBM)

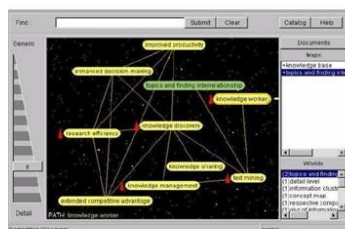


Рисунок 2 - Інтерфейс
SemioMap

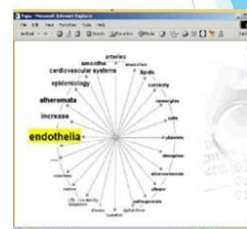


Рисунок 3 - Інтерфейс
Oracle Text

4

Рисунок Б.4 – Слайд 4

Вибір середі виконання

	легковажність	легкість і швидкість написання	співтовариство	використовує сучасні парадигми	багатопоточність
Java	-	-	+	-	+
Node.js	+	+	+	+	-

5

Рисунок Б.5 – Слайд 5

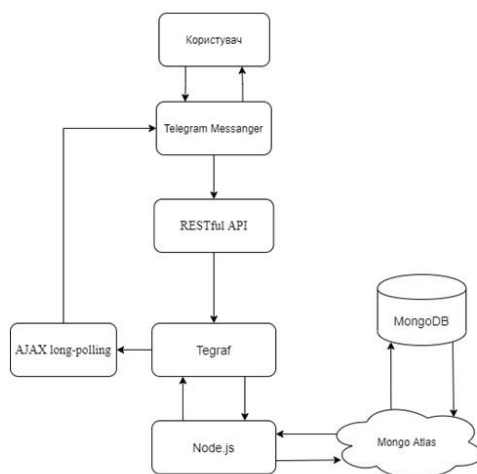
Структура Бази Даних

- **admins** - колекція користувачів, які мають права на маніпулювання даними;
- **questions** - колекція, яка містить в собі список поширених запитань, на які застосунок може дати відповідь, аналізуючи завантажені в нього документи;
- **answers** - колекція, яка містить в собі список відповідей на поширені питання;
- **owners** - колекція користувачів, які мають права на маніпулювання адміністраторами;
- **stats** - колекція, яка містить в собі статистику про користувачів, що користуються чат-ботом;
- **tables** - колекція таблиць, які використовуються в даний момент.

6

Рисунок Б.6 – Слайд 6

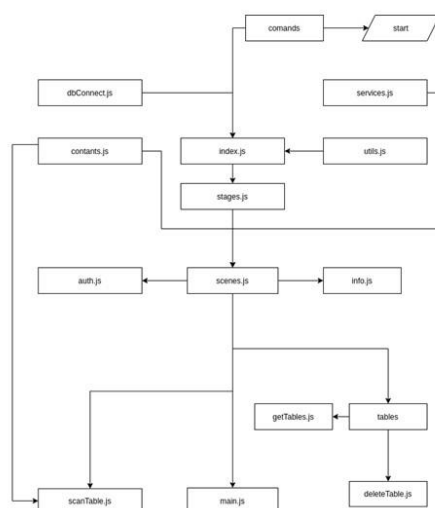
Архітектура програми



7

Рисунок Б.7 – Слайд 7

Структурна схема застосунку



8

Рисунок Б.8 – Слайд 8

Функціональна схема програми

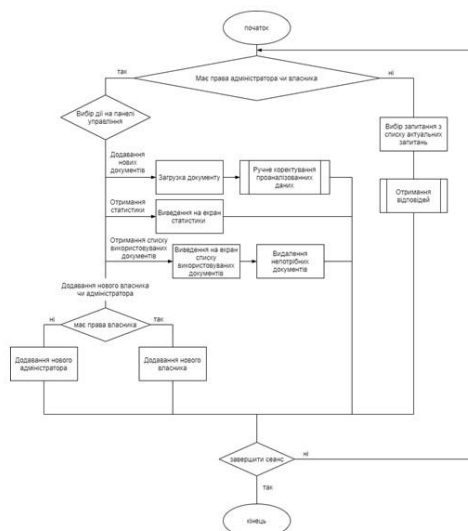


Рисунок Б.9 – Слайд 9

Вхідні дані

ПЕРЕЛІК СПЕЦІАЛЬНОСТЕЙ,
яким надається особлива підтримка

Шифр галузі	Галузь знань	Код спеціальності	Назва спеціальності
12	Інформаційні технології	124	Системний аналіз
13	Механічна інженерія	131	Прикладна механіка
		132	Матеріалознавство
		133	Галузеве машинобудування
		134	Авіаційна та ракетно-космічна техніка
		136	Металургія
14	Електрична інженерія	141	Електроенергетика, електротехніка та електромеханіка
		144	Теплоенергетика
15	Автоматизація та приладобудування	151	Автоматизація та комп'ютерно-інтегровані технології
		152	Метрологія та інформаційно-вимірювальна техніка
		153	Мікро- та наносистемна техніка
17	Електроніка та телекомунікації	172	Телекомунікації та радіотехніка
19	Архітектура та будівництво	173	Авіоніка
		192	Будівництво та цивільна інженерія
27	Транспорт	275	Транспортні технології (за видами)

Рисунок 4 - Приклад docx документу

Таблиця 5

Рисунок 5 - Приклад xlsx документу

Рисунок Б.10 – Слайд 10

Інтерфейс програми

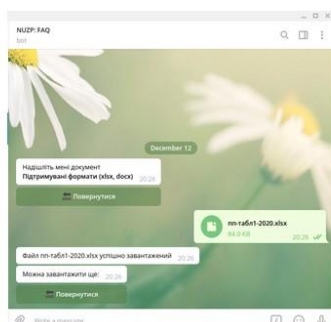


Рисунок 6 - Сцена сканування таблиць



Рисунок 7 - Сцена найпоширеніших запитань

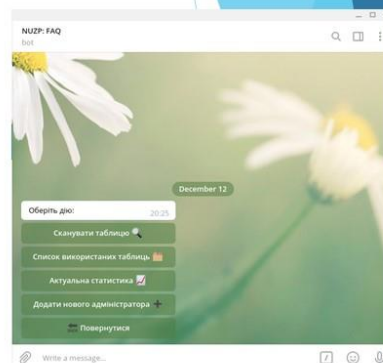


Рисунок 8 - Панель управління застосунком

11

Рисунок Б.11 – Слайд 11

Інтерфейс програми

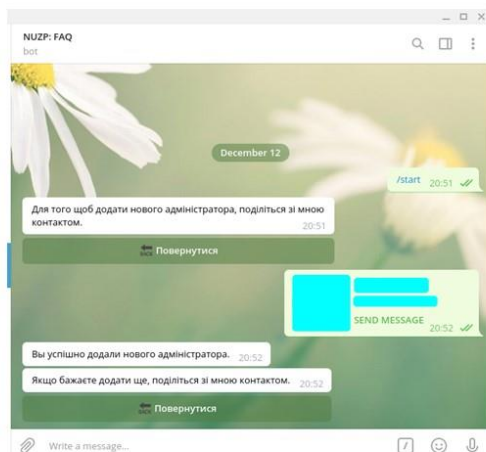


Рисунок 9 - Сцена додавання нового адміністратора

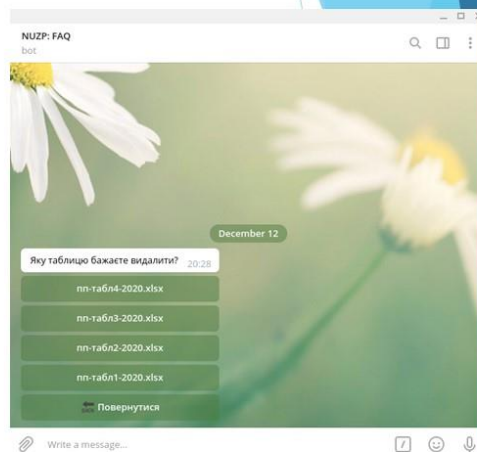


Рисунок 10 - Сцена видалення завантажених таблиць

12

Рисунок Б.12 – Слайд 12

Висновки

У результаті програмної реалізації розроблено чат-бота, здатного отримувати документи та аналізувати їх для вилучення текстової інформації, щоб зібрати базу знань, яку в подальшому можна використовувати для формування списку запитань та надавання для них відповідей. Практичною сферою застосування є - перетворення таблиць Правил прийому до закладів вищої освіти у базу даних чат-бота для надання інформації абітурієнтам щодо умов вступу.

Наукова новизна роботи полягає в запропонованому методі для структуризації текстової інформації для наповнення БД, що використовує чат-бот Telegram для крос-платформної, швидкої, легкодоступної та захищеної взаємодії з користувачем.

Рисунок Б.13 – Слайд 13

ДЯКУЮ ЗА
УВАГУ!

Рисунок Б.14 – Слайд 14