

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА АНАЛІТИЧНОЇ СИСТЕМИ АНАЛІЗУ ТА ПІДТРИМКИ
ПРИЙНЯТТЯ РІШЕНЬ ЄДИНОЇ РЕГІОНАЛЬНОЇ ІНФОРМАЦІЙНОЇ
СИСТЕМИ ЗАХИСТУ КРИТИЧНОЇ ІНФРАСТРУКТУРИ DEVELOPMENT OF
AN ANALYTICAL SYSTEM FOR ANALYSIS AND DECISION SUPPORT OF
THE UNIFIED REGIONAL INFORMATION SYSTEM FOR CRITICAL
INFRASTRUCTURE PROTECTION

Виконав(ла): студент(ка) 4 курсу, групи КНТ-121

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЩЕРБИНА О.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ДУБРОВІН В.І.

(ПРИЗВИЩЕ та ініціали)

Рецензент ПОЛЯКОВ М.О.

(ПРИЗВИЩЕ та ініціали)

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

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
 “ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЩЕРБИНИ Ольги Олегівни

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка аналітичної системи аналізу та підтримки прийняття рішень Єдиної регіональної інформаційної системи захисту критичної інфраструктури. Development of an Analytical System for Analysis and Decision Support of the Unified Regional Information System for Critical Infrastructure Protection.

керівник проєкту (роботи) к.т.н., професор, ДУБРОВІН Валерій Іванович
 (науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 28 ” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ДУБРОВІН В.І., професор		
Нормоконтроль	БСЛОВА А. В., асистент		

7. Дата видачі завдання « 14 » квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студентка



(підпис)

Ольга ШЕРБИНА

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)



(підпис)

Валерій ДУБРОВІН

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 90 с., 11 табл., 46 рис., 3 дод., 24 джерел.

БПЛА, РАКЕТИ, КРИТИЧНА ІНФРАСТРУКТУРА, АНАЛІЗ ДАНИХ З ВІДКРИТИХ ДЖЕРЕЛ, АВТОМАТИЗОВАНА СИСТЕМА

Об'єкт дослідження – процес аналізу інформації, отриманої з відкритих джерел, пов'язаної з переміщенням БПЛА.

Мета роботи – розробити застосунок, що здійснює рекурсивний пошук тематичних груп у Telegram для подальшого прогнозування потенційно небезпечних дій.

Методи дослідження – використання мови програмування C#, API для вступу в Telegram-групи, аналіз даних, тестування програмного забезпечення.

Результати. Створено програмний Windows-застосунок, який готує дані для подальшого прогнозування пересування БПЛА.

Висновки. У ході дослідження було розроблено програмний застосунок для Windows, який здійснює рекурсивний пошук тематичних Telegram-груп, пов'язаних із переміщенням БПЛА. Застосунок дозволяє автоматично збирати та обробляти повідомлення з відкритих джерел, що може бути використано для подальшого аналізу та прогнозування потенційно небезпечних дій. Використані підходи до роботи з Telegram API, асинхронна обробка даних і тестування функціональності підтвердили ефективність обраних методів. Отримані результати можуть бути корисними в інформаційній безпеці та системах моніторингу.

Галузь використання – застосунок може бути використаний у сферах національної безпеки, інформаційного аналізу та кіберрозвідки.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 90 pages, 46 figures, 11 tables, 3 appendixes, 24 sources.

UAVs, MISSILES, CRITICAL INFRASTRUCTURE, ANALYSIS OF DATA FROM OPEN SOURCES, AUTOMATED SYSTEM.

Object of study is an analysis of information obtained from open sources related to the movement of UAVs.

Subject of study are analysis of information obtained from open sources related to the movement of UAVs.

The purpose of the work is to develop an application that performs a recursive search of thematic groups in Telegram in order to forecast potentially dangerous activities.

Materials, methods, and tools: programming language: C#, telegram API for joining and interacting with groups, data analysis techniques, asynchronous data processing, software functionality testing.

Results. A Windows software application was developed that prepares data for further forecasting of UAV movement.

Conclusions. During the research, a Windows software application was developed that performs a recursive search of thematic Telegram groups related to UAV movement. The application allows for automatic collection and processing of messages from open sources, which can be used for further analysis and forecasting of potentially dangerous actions. The approaches used—working with the Telegram API, asynchronous data processing, and functionality testing—confirmed the effectiveness of the chosen methods. The obtained results may be useful in the fields of information security and monitoring systems.

The field of use is a national security, information analysis, and cyber intelligence.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області та постановка завдання роботи	10
1.1 Опис предметної області	10
1.2 Огляд відомих методів.....	12
1.3 Огляд існуючих програмних рішень.....	13
1.4 Постановка завдання роботи.....	15
1.5 Висновок за розділом 1.....	16
2 Вибір та обґрунтування структури системи, яка проектується.....	17
2.1 Вибір мови програмування	17
2.2 Вибір технологій графічної оболонки	19
2.3 Огляд обраного середовища та інструментарію вирішення задачі	20
2.4 Вибір системи управління базами даних.....	23
2.5 Структурна схема програми.....	23
2.6 Опис модулів розроблювальної програми	24
2.7 Функціональні вимоги до програми.....	25
2.8 Висновок за розділом 2.....	26
3 Основні рішення щодо реалізації компонентів системи.....	28
3.1 Вибір шаблону проектування	28
3.2 Опис модулів розроблювальної системи.....	30
3.3 Проектування бази даних	39
3.4 Опис алгоритмів функціонування застосунку	40
3.5 Обробка виняткових ситуацій	43
3.6 Основні рішення щодо розробки інтерфейсу.....	44
3.7 Висновок за розділом 3.....	47
4 Експлуатація та тестування програми	48
4.1 Призначення програми	48

4.2 Умови використання системи.....	48
4.3 Експлуатація програми.....	48
4.4 Висновок за розділом 4.....	56
Висновки	58
Перелік джерел посилання	59
Додаток А Технічне завдання	62
Додаток Б Текст програми	60
Додаток В Слайд презентації	77

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- API – Application Programming Interface;
- UAV – Unmanned Aerial Vehicle;
- OSINT – Open Source Intelligence;
- GUI – Graphical User Interface;
- БПЛА – Беспілотний літальний апарат;
- .NET – Software development platform by Microsoft;
- КІ – критична інфраструктура;
- ПЗ – програмне забезпечення.

ВСТУП

За словами президента Міжнародного комітету Червоного Хреста в 2024 році в світі налічувалося понад 120 збройних конфліктів. За даними від Генштабу від 24 лютого 2022 року було близько півтора мільйона поранених та загиблих унаслідок збройного конфлікту між росією та Україною. На сьогодні питання захисту людських життів та створення умов для постійного функціонування об'єктів критичної інфраструктури є актуальним та важливим.

Важливо мати автоматизовані інструменти для виявлення пересування ворожої техніки. Такі інструменти зменшують кількість ресурсів, які можуть знадобитися для прогнозування подальших дій ворога та здійснення адаптації війська для певної ділянки фронту. Якщо прогноз, який був наданий системою щодо часу і типу зброї є вірним, то завдяки цьому, можна завчасно підготувати засоби протидії відповідному виду зброї та визначити час, необхідний для евакуації людей і організації заходів оборони пов'язаних як з військовими так і з цивільними об'єктами.

Тобто, в сучасному світі справді є потреба у використанні автоматизованих інструментів, які можуть протидіяти терористичним актам, виявлення протиправної діяльності з боку населення країни так і з боку країн-агресорів. Цей інструмент може бути розроблений для військових цілей, але в майбутньому може бути адаптований для різних сфер життя.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Опис предметної області

Починаючи з 24 лютого 2022 року на території всієї України постало питання захисту критичної інфраструктури переважно від надзвичайних ситуацій соціального, військового характеру та втручань в кіберпросторі.

Згідно звіту Київської школи економіки з початку повномасштабного вторгнення загальна сума прямих збитків нерухомості, іншій інфраструктурі, транспортним засобам та запасам склала майже 170 млрд. дол., а збитки інфраструктури становлять близько 35% від загальних втрат. У табл 1.1 наведено розподіл втрат за секторами інфраструктури [1],[2].

Таблиця 1.1 – збитки інфраструктури за період повномасштабної військової агресії проти України

	Млрд. \$
Транспортна інфраструктура	38,5
Енергетика	14,6
Житлово-комунальне господарство	3,47
Цифрова інфраструктура	1,2
Всього:	57,77

Український аналітичний центр Dixi Group опублікував звіт щодо здійснених росією масованих ударів ракетами та БПЛА по об’єктах критичної інфраструктури (КІ) України. Розподіл використаної зброї за типами відображено на рис 1.1. [3]. Під час позитивного прогнозування на 2025 рік щодо можливої кількості застосованої зброї проти об’єктів КІ може становити таку саму кількість як і на момент 2024 року, а під час негативного сценарію її кількість може збільшитися майже в 2 рази.

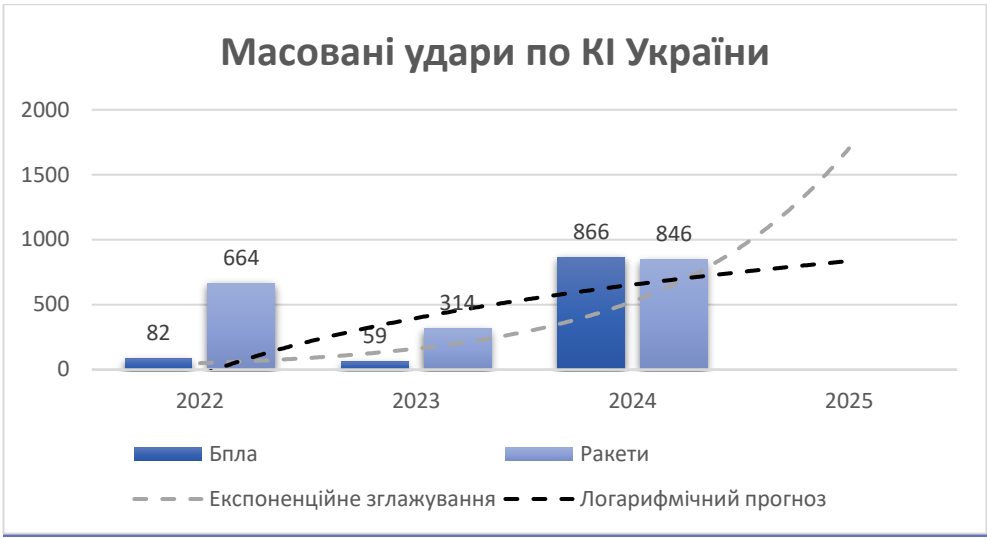
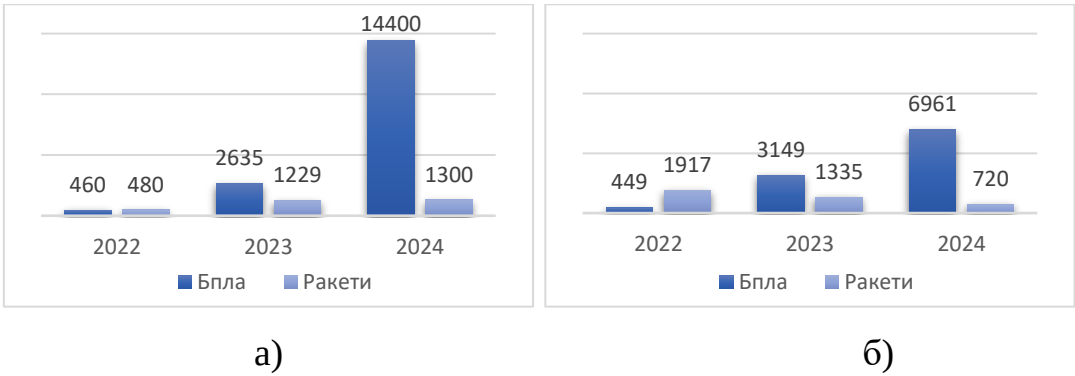


Рисунок 1.1 – Аналітичні дані про масовані удари по КІ України

На рис 1.2 відображено дані, надані силами ППО про кількість збитої техніки [4]-[6] та від української агенції Top Lead, щодо кількості випущеної техніки ворогом з 2022 року по жовтень 2024 року [7]-[8], можна дійти до висновку, що кількість випущених БПЛА ворогом зростає в експоненціальному темпі. Це може бути зумовлено низкою факторів, серед яких низька собівартість виробництва, технологічний розвиток тощо.



а) кількість збитої техніки [2]-[4]; б) кількість випущеної техніки ворогом [5],[6].

Рисунок 1.2 – Аналітичні дані про кількість техніки за 2022 - 2024 роки

Враховуючи ці фактори, ефективний захист критичної інфраструктури потребує не лише посилення протиповітряної оборони, а й розробки програмного забезпечення для аналізу інформаційних загроз. Такий підхід

дасть змогу виявляти потенційно небезпечні повідомлення, що можуть містити дані про пересування БПЛА, і на основі цього прогнозувати можливі загрози для конкретних об'єктів.

1.2 Огляд відомих методів

Для вирішення завдання захисту критичної інфраструктури не можна обійтися виключно одним методом. Рішення має комбінувати в собі кілька способів, які вирішують різні типи завдань, серед них: обробка текстових даних, прогнозування та робота з даними.

1.2.1 Текстологічний метод

Текстологічний метод – це метод вибутку знань з документів. Завдання отримання інформації з тексту ховає в собі такі підзавдання: розуміння та видокремлення сенсу тексту. Складність даного процесу полягає в принципіальній неможливості співпадиння знань сенсу, який намагався закласти автор та сенсу, який був трактований користувачем.

Умовно текст можна поділити на наступні ключові елементи:

- спостереження;
- наукові поняття – «спільні місця» для різних текстів та «вода»;
- суб'єктивні погляди – особисті погляди автора;
- спільні місця – система наукових понять;
- запозичення – первинний матеріал спостережень.

Тобто, головне завдання аналітика – це здійснення декомпозиції тексту для реалізації бази знань фрагментів [9].

1.2.2 Емпіричний метод

Емпіричний метод охоплює в собі різноманітні способи дослідження, серед яких є спостереження, статистика, порівняння, тощо.

Спостереження – систематичне та цілеспрямоване вивчення об’єктів, явищ, процесів у їх природних умовах. Завдяки чому отримуються дані, які відображають об’єктивність поведінки чи розвитку подій. У процесі спостереження можуть фіксуватися окремі характеристики або кількісні показники, на основі яких потім визначаються певні закономірності з використанням інших технік, припустимо порівняння [10].

1.3 Огляд існуючих програмних рішень

1.3.1 Сервіс IBM resilient SOAR

IBM Resilient SOAR є продуктом IBM для управління інцидентами кіберзагроз та допомагає організаціям автоматично реагувати на них, що значно мінімізує їх втрати [11].

Основні переваги сервісу:

- відстеження та аналіз кіберзагроз в режимі реального часу;
- інтеграція з іншими системами безпеки;
- відносно дружній інтерфейс, що відображено на рис.1.3;
- створення та виконання автоматизованих робочих процесів.

Недоїлки:

- працює виключно в межах доступності компанії;
- вартість.

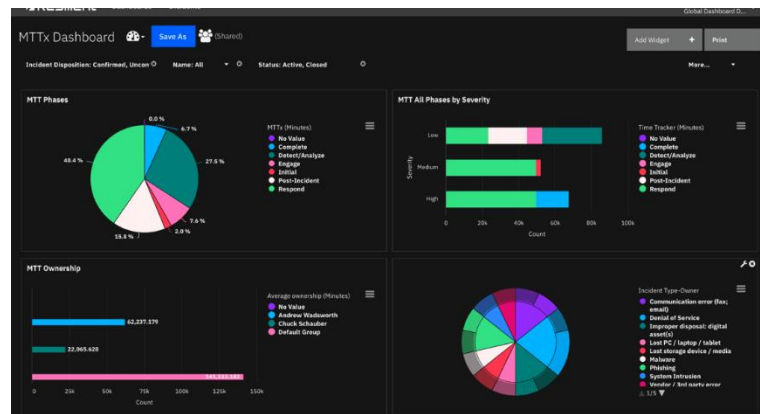


Рисунок 1.3 – Сервіс IBM resilient SOAR

1.3.2 Avada-media парсер-телеграм

Парсер для телеграму від компанії AVADA-media орієнтований на автоматизацію рутинного процесу збору інформації та управління великими масивами даних.

Основні переваги сервісу:

- автоматизований збір контактної інформації учасників групи;
- фільтрація спаму в групі за заданими ключовими словами [12].

Недоліки:

- орієнтований виключно на автоматизовану взаємодію з клієнтами;

На рис. 1.4 наведено графічний інтерфейс парсеру, який розроблений компанією AVADA-media.

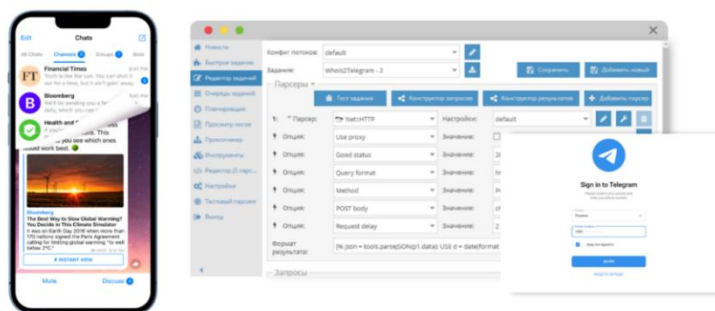


Рисунок 1.4 – Парсер AVADA-media

1.3.3 Open Cyber Threat Intelligence Platform

OpenCTI — це платформа з відкритим вихідним кодом, яка була розроблена компанією Filigran. Ресурс дозволяє організаціям впорядковувати, зберігати та візуалізувати інформацію про кіберзагрози [13].

Основні переваги сервісу:

- відображення загроз у вигляді графів;
- інтеграція з іншими системами безпеки;
- дружній інтерфейс, що відображено на рис. 1.5;
- наявний вбудований аналітичний модуль.

Недоїлки:

- складність налаштування;
- обмежена підтримка.

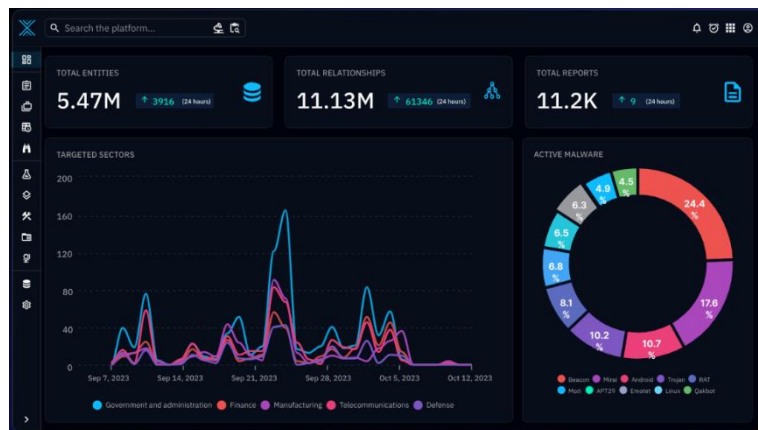


Рисунок 1.5 – OpenCTI

1.4 Постановка завдання роботи

Єдина аналітична система має здійснювати наступний функціонал:

- авторизація через Telegram – акаунт;
- збір інформації, що містить посилання з Telegram, які починаються на <https://t.me>;

- керувати ключовими словами, групи у які був здійснений вступ, заблоковані групи користувачем;
- пошук повідомлень за ключовими словами в Telegram-каналі;
- аналізувати отриману інформацію;
- робити прогнозування ймовірної ситуації.

1.5 Висновок за розділом 1

У першому розділі було досліджено предметну область, що дозволило визначити, що одну з найбільших загроз для об'єктів КІ становлять БПЛА. Також, в межах даного розділу було здійснено огляд вже існуючих програмних забезпечень та методів для вирішення задачі визначення пересування БПЛА та прогнозування ураження певного об'єкту КІ.

2 ВИБІР ТА ОБҐРУНТУВАННЯ СТРУКТУРИ СИСТЕМИ, ЯКА ПРОЕКТУЄТЬСЯ

Невід’ємним кроком розробки кожної системи є її проектування для того, щоб уникнути проблем пов’язаних з ймовірними конфліктами під час додавання до проєкту фреймворки, бібліотеки, модулі та ін. Даний підхід дозволяє використовувати більш раціонально час та заощаджувати фінансові ресурси.

2.1 Вибір мови програмування

Мова програмування – це одна з найбільш важливих етапів проектування застосунку, адже це є його основою. Сучасні реалії вимагають постійне оновлення мови програмування, можливість масштабування, подальшу підтримку.

Для того, щоб продукт зміг розвиватися після розробки, необхідно враховувати актуальність мов програмування. DOU – найбільша ІТ-спільнота в Україні провела анкетування майже 7500 осіб, що працюють в сфері ІТ та провела аналітику найбільш популярних мов програмування серед розробників в 2025 році: TypeScript, JavaScript, C#, Java, PHP [14]. Відсотковий розподіл вподобання мови серед розробників відображено на рис. 2.1, а у таблиці 2.1 наведено порівняльний аналіз даних мов.

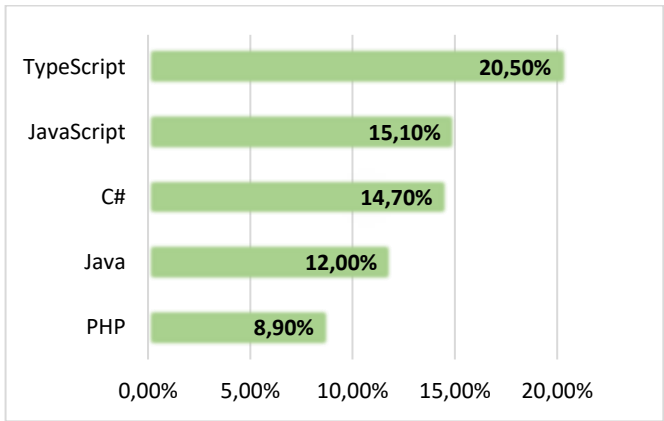


Рисунок 2.1 – Найпоширеніші мови програмування серед розробників України [14]

Таблиця 2.1 – Порівняльна характеристика мов програмування

№	Критерій	TypeScript	JavaScript	C#	Java	PHP
1	Платформи	Web, node.js	Web, node.js	.NET, windows	Багатолатформенний	Web
2	Типізація	Статична	Динамічна	Статична	Статична	Динамічна
3	Продуктивність	Висока	Висока	Висока	Висока	Середня
4	Підтримка асинхронності	+	+	+	+	+
5	Інтеграція з github	+	+	+	+	+
6	ООП	+	+-	+	+	+
7	Підтримка спільноти	+	+	+	+	+
8	Розробка під ios	+	+	+	+	-
9	Розробка під Android	+	+	+	+	-
10	Розробка desktop	+	+	+	+	-
11	Кросплатформена розробка	+-	+-	+	+	-
12	Витрати ресурсів	+	+	Середні	Середні	-

Здійснивши аналіз таблиці 2.1 можна дійти до висновку, що найоптимальнішою мовою програмування серед інших для розроблювального застосунку становить C#. Не дивлячись на те, що C# та Java схожі у багатьох

аспектах, C# є більш кросплатформений, підтримує сучасні графічні бібліотеки та частіше оновлюється.

C# переважно використовується для розробки десктопних додатків для операційної системи Windows. Завдяки використанню додаткових фреймворків таких як, Xamarin та .NET MAUI, можливості застосування C# значно розширюються. Xamarin дозволяє розробляти кросплатформенні мобільні додатки для iOS та Android. .NET MAUI (Multi-platform App UI) дозволяє створювати кросплатформенні десктопні та мобільні додатки для Windows, macOS, Android і iOS.

C# підтримує строгую типізацію та асинхронне програмування, що дозволяє розробляти ПЗ, яке здатне працювати з великою кількістю інформації. Величезною перевагою також є інтеграція з іншими інструментами і сервісами в екосистемі .NET, такими як Azure, що значно полегшує створення хмарних рішень та веб-сервісів. Завдяки такій широкій екосистемі, C# стає не лише мовою для створення програм, а й потужним інструментом для побудови комплексних, високонавантажених додатків, що працюють на різних платформах.

2.2 Вибір технологій графічної оболонки

Графічний інтерфейс – це один з головних способів комунікації програми з користувачем чи з аналітиком. Він має бути сучасним, інтуїтивно зрозумілим, а також легким у реалізації, щоб мінімізувати час розробки. Обраний інструментарій повинен забезпечувати швидкий відгук графічних елементів для того, щоб залишити позитивний досвід користувачеві від використання ПЗ.

Одними з найбільш популярних графічних оболонок, які використовуються в C# є фреймворк WPF та WinForms.

WinForms дозволяє створювати прості класичні графічні додатки для Windows за допомогою графічного конструктора у Visual Studio [15]. На відміну від WinForms, фреймворк WPF є більш гнучким і потужним, оскільки дозволяє проєктувати графічні елементи за допомогою розмітки XAML. WPF є більш ресурсомістким та вимагає більше часу на проєктування форми відносно WinForms [16].

Таблиця 2.2 – Порівняльна характеристика графічних компонентів

№	Критерій	WPF	Windows Forms
1	Складність	+	-
2	Гнучкість дизайну	+	-
3	Сучасність	+	-
4	Кросплатформеність	-	-
5	Підтримка графіки та анімації	+	-
6	Продуктивність при навантаженні	-	+

Відповідно до детального порівняння графічних компонентів, наведених в таблиці 2.2, можна зробити висновок, що WPF є найбільш оптимальним варіантом для розроблювального застосунку. Даний фреймворк є сучасним, підтримує гнучкість дизайну, надає зручну роботу з графікою та анімацією. Основним недоліком становить те, що він не підтримує кросплатформеність і вимагає більше ресурсів під час роботи з великими обсягами даних. Проблематику кросплатформеності можна виправити шляхом окремої розробки під інші девайси. Враховуючи, що зараз є більш актуальним розробка застосунку під ОС Windows, то цим фактом можна знехтувати.

2.3 Огляд обраного середовища та інструментарію вирішення задачі

2.3.1 Вибір середовища розробки

За даними DOI у 2025 році розробники, які використовують C# під час роботи, зазначають, що надають перевагу Visual Studio, Visual Studio Code та Rider [14].

Таблиця 2.3 – Порівняльна характеристика графічних компонентів

№	Критерій	Visual Studio	Visual Studio Code	Rider
1	Підтримка C#	+	+	+
2	Підтримка WPF	+	-	+
3	WPF Designer/XAML Designer	+	-	+-
4	Швидкість	-	+	+
5	Ціна	+	+	-

За інформацією, що наведена у таблиці 2.3 найбільш оптимальним варіантом для розроблювального програмного застосунку становить Visual Studio, адже він єдиний має повну та зручну підтримку фреймвоку WPF з можливістю легко та швидко впроваджувати модифікації до дизайну програми.

2.3.2 Вибір репозитарію коду

Велика кількість розробників стикалися з проблематикою втрати даних у зв'язку з вимкненням електроенергії, стихійними діями та іншими непередбачуваними ситуаціями. Для збереження першочергового коду та для зручного майбутнього масштабування через роботу в команді необхідно використовувати репозитарій коду. У таблиці 2.4 наведено порівняння найбільш популярних репозитаріїв серед розробників.

Таблиця 2.4 – Порівняльна характеристика репозитаріїв

№	Критерій	GitHub	SourceForge
1	Системи контролю версій	Git	Git, SVN, Mercurial
2	Інтерфейс	+	-
3	Спільна робота	+	-
4	Поширення коду	репозиторій	бінарники, інсталятор

SourceForge — це попередник GitHub, який помітно поступається йому за функціональністю. У зв'язку з тим, що GitHub містить набагато зручніший та ширший функціонал для спільної роботи, а також надає можливість поширювати код, а не лише бінарники, він є оптимальним варіантом для цього проєкту [17]-[18].

2.3.3 Огляд TelegramAPIs

На території ворога та України є дуже поширене використання Telegram-месенджера, що зумовлює необхідність здійснювати аналітику щодо пересування ворожих БПЛА саме через цей месенджер. Одним з найбільш зручних інструментів для швидкої інтеграції є TelegramAPI.

Telegram APIs дозволяє розробникам створювати ботів, інтегрувати сторонні сервіси, а також взаємодіяти з обліковими записами та чатами. API ділиться на дві основні категорії: Telegram Bot API та Telegram APIs. У проєкті буде використано саме Telegram APIs, це зумовлено тим, що його використання не вимагає створення в межах акаунту бота, діяльність якого має запускати стороння особа. Тобто, Telegram API дозволяє здійснювати ту саму діяльність, що й користувач, але більш автономно [19].

2.4 Вибір системи управління базами даних

MySQL — це популярна реляційна система керування базами даних (СКБД), яка використовується для зберігання та обробки структурованих даних. Вона широко застосовується у веб-розробці, корпоративних додатках і аналітичних системах.

Систему MySQL було обрано через: простий інтерфейс і легкість у використанні; швидку роботу з великими обсягами даних; широку підтримку спільноти та документації; легкість інтеграції з VisualStudio та з C# [20].

2.5 Структурна схема програми

Через використання фреймворку WPF в застосунку буде використано шаблон проєктування MVVM, який є вбудований у WPF. Структура системи складається з логічного блоку, який складається з наступних частин:

- зберігання даних (БД), що відповідає за частину “Model”;
- алгоритмічна частина відноситься до “View-Model”, що здійснює функцію обміну даними між “Model” та “View”;
- інтерфейс відповідає за частину “View” [21].

На рис. 2.2 відображено архітектурну схему програми.

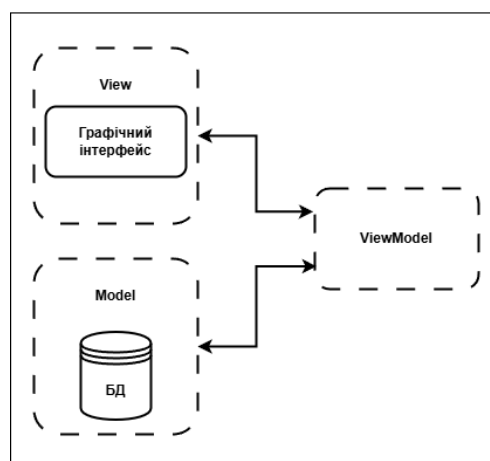


Рисунок 2.2 – Архітектурна схема ПЗ

2.6 Опис модулів розроблювальної програми

У таблиці 2.5 наведено опис модулів, які визначають роботу програми, з описом їх функціональності.

Таблиця 2.5 – Опис модулів реалізованого ПЗ з визначенням основних функцій

Модуль	Опис
Графічний	Модуль призначений для спрощеної комунікації системи з користувачем
Модуль взаємодії з БД	В даному модулі буде здійснено обробку запитів до БД та надання відповіді з неї
Модуль роботи з Telegram	Модуль призначений для взаємодії з Telegram-спільнотами, а також авторизації на початку роботи застосунку
Модуль обробки вмісту Telegram-чатів	Модуль призначений для здійснення парсеру груп та отримання повідомлень за правилами, які задає користувач
Аналітичний модуль	Модуль визначає траєкторію переміщення БПЛА та здійснює прогнозування щодо подальших дій ворога

На рисунку 2.3 відображено яким чином модулі будуть розміщені у структурній схемі зстосунку.

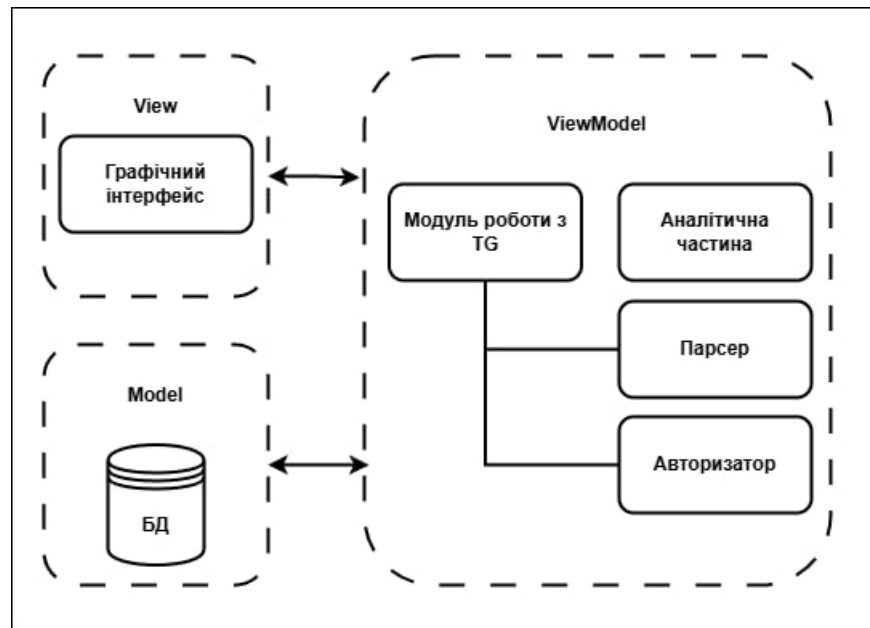


Рисунок 2.3 – Структурна схема

2.7 Функціональні вимоги до програми

В першому розділі було проаналізовано предметну область та визначено, що системою буде користуватися один тип користувачів, а саме: аналітик.

Функціональні можливості аналітика:

- авторизація в системі;
- розпочати парсер груп в Telegram;
- здійснювати видалення та блокування знайдених груп;
- перегляд усіх груп;
- можливість редагувати, додавати та видаляти ключові слова;
- можливість додавати та видаляти групи, які було додано до чорного списку;
- пошук повідомлень за ключовими словами в Telegram-групах з можливістю здійснювати аналітиком редагування отриманих даних, які необхідні для подальшого прогнозування.

На рис. 2.4 відображено use-case діаграму щодо функціональних можливостей аналітика.

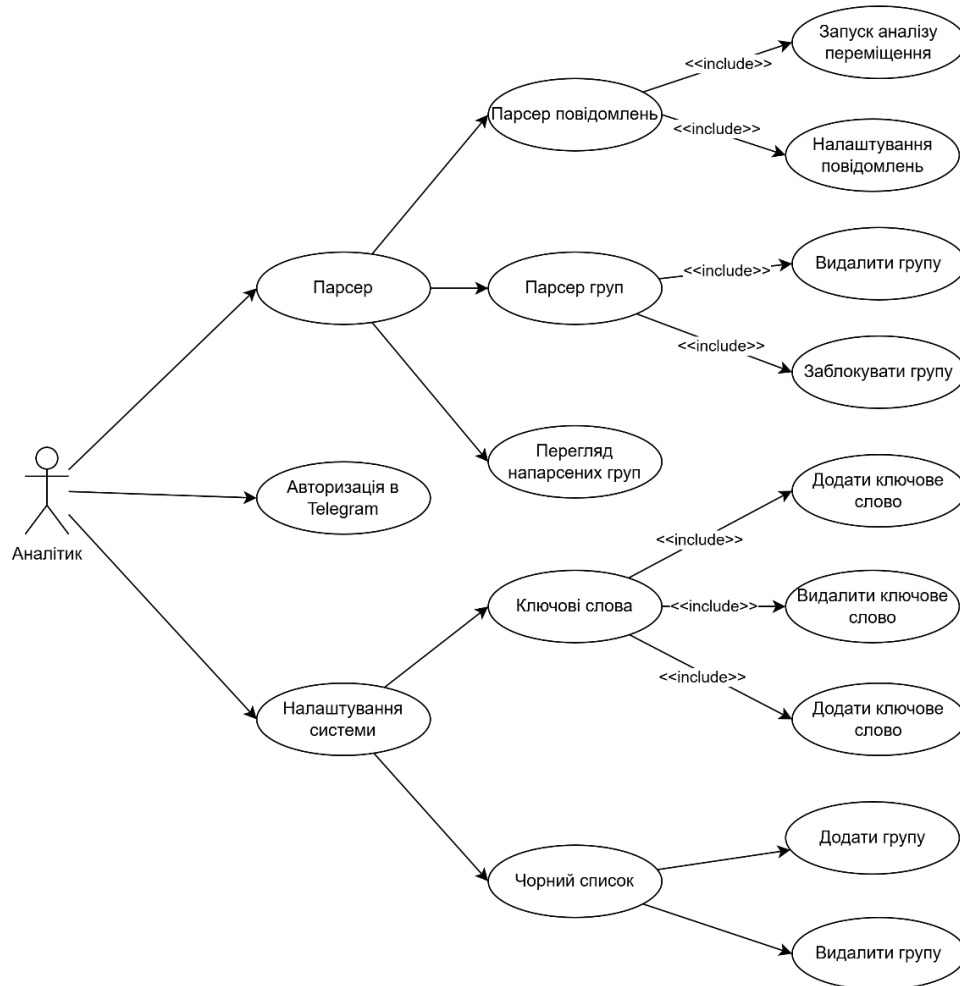


Рисунок 2.4 – Use-case діаграма

2.8 Висновок за розділом 2

У другому розділі пояснювальної записки було здійснено проєктування архітектури застосунку та обрано стек технологій для його розробки. Мовою програмування було обрано C# через її високу продуктивність, підтримку об'єктно-орієнтованого програмування та велику кількість можливостей для інтеграції з іншими компонентами. WPF - це фреймворк, який дозволяє

розроблювати сучасний графічний інтерфейс з мінімальними витратами часу та ресурсів.

Для керування базою даних було обрано MySQL через її стабільність для збереження та обробки даних. У межах розділу також було представлено структурну схему застосунку, описано його основні модулі та визначено функціональні можливості, доступні для кінцевого користувача.

В якості шаблону проєктування було обрано MVVM. Даний вибір був зумовлений через використання фреймворку wrf, в який вже є вбудований даний шаблон.

Під час проєктування системи важливо враховувати ймовірні ризики, які можуть бути пов'язані зі стихійними або іншими видами лиха. Для забезпечення резервного копіювання коду та подальшої командної роботи було обрано github. Для здійснення основних дій щодо взаємодії з месенджером telegram буде використано TelegramAPIs.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Опис архітектури проєкту

Архітектура програмного забезпечення є важливою складовою кожного проєкту, що дозволяє пришвидшити розробку та забезпечити стабільну підтримку, не витрачаючи на проєкт велику кількість ресурсів. Таким чином, було розроблено архітектуру ґрунтуючись на «бізнес-цілі» проєкту.

На рис. 3.1 наведено пакетну діаграму, яка відображає взаємодію кожного шару архітектури. Проєкт умовно розділено на 4 шари: презентаційний (відповідає за графічне представлення інформації), бізнес-шар (відповідає за обробку даних, які виконують поставлені завдання застосунку), шар-сховище (відповідає за збереження даних в БД), шар передачі даних (відповідає за передачу даних між шарами).

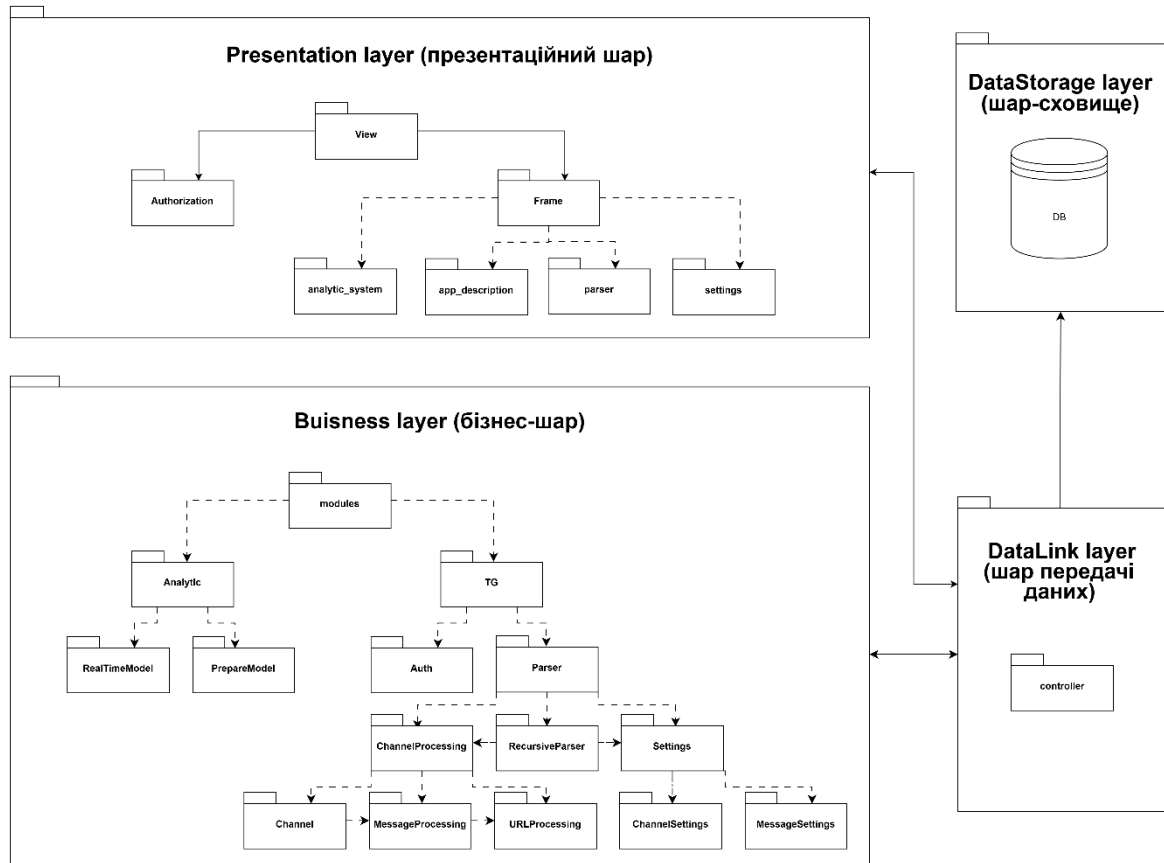


Рисунок 3.1 – Package-object діаграма

Presentation Layer було умовно розділено на 2 незалежні частини: авторизація та Frame, об'єднання яких створює цілу графічну реалізацію проєкту. Пакет Frame містить в собі пакети, які дозволяють відобразити основні компоненти бізнес-логіки, а саме: аналітична частина, парсер, налаштування та опис застосунку.

У 2 розділі було визначено, що View-Model, який виступає аналогом BuisnessLayer містить 2 основних компоненти: аналітика та телеграм. Аналітика в свою чергу поділяється на аналітику, яку можна здійснювати в реальному часі та навчальну модель, тобто підготовка даних для того, щоб в майбутньому здійснювати аналітику в реальному часі. Телеграм складається з компоненту авторизації та парсеру, дані компоненти умовно знаходяться на одному рівні через неможливість існувати окремо одне від одного. Парсер складається з каналів, пакету, який реалізує рекурсивний парсер та

налаштувань парсеру. В каналах є можливість вступати, виходити, дивитися їх опис, дану функціональність реалізує Channel. Окрім попередньо згаданих можливостей, канал складається з повідомлень, роботу з якими реалізує пакет MessageProcessing. Повідомлення складаються з тексту та посилань, обробку посилань в свою чергу реалізує пакет URLProcessing. В програмі можна налаштовувати повідомлення та канали, реалізація даних можливостей знаходиться в пакетах ChannelSettings та MessageSettings відповідно.

3.2 Опис модулів розроблювальної системи

У першому розділі було описано, що проєкт складається з 4 шарів. BusinessLayer відповідає за обробку основних компонентів системи. PresentationLayer відповідає за графічну реалізацію проєкту. DataStorageLayer – відповідає за збереження даних в базі даних, а DataLink Layer за передачу даних між шарами.

Таблиця 3.1 – Опис business layer

Модуль	Структура	Функції
Аналітичний	Class PrepareModel	Отримання повідомлень з усіх наявних груп в базі за обраний проміжок часу, які містять ключові слова; Створення нових правил.
	Class RealTime	Отримання повідомлень в реальному часі з ключовими словами; Обробка отриманих повідомлень та їх співставлення з наявними правилами.
Авторизатор	Class Authorization	Здійснення авторизації в Telegram

Кінець таблиці 3.1

Модуль	Структура	Функції
Парсер	Class BasicFetcher IBasicFetcher	Вступ в приватні та публічні канали; Отримання усіх каналів, в які було здійснено вступ; Покинути канал; Отримання заданої кількості повідомлень з групи; Отримання повідомлень після заданої дати.
	IMessageComponentBase	Встановити рядок підключення до БД; Отримати посилання на телеграм групу з повідомлення; Отримати ід групи; Відобразити усі ТГ-групи; Отримати хеш-телеграм груп; Видалити групу за ід; Перевірка чи існує група; Оновити кількість підписників групи; Додати телеграм-групу.
	LinkUtils	Отримати унікальні посилання; Перевірка чи це посилання на ТГ-групу; Перевірка чи це є публічний канал; Перетворення посилання до виду https://t.me..... ;
	IBasicMessageHandler	Додати групу до списку-ігнорування; Повернути групи в яких було здійснено вступ.
	IParser	Почати парсер; Зупитини парсер; Відновити парсер.
	IParserList	Зберігає групи, які були знайдені під час парсингу.

На рисунку 3.2 наведено схему взаємодії компонентів парсеру. На наступних рисунках відображено методи та змінні, які вони використовують. Більшість класів наслідуються від інтерфейсів. Шари між собою взаємодіють

за допомогою інтерфейсів, а не через класи. Це дозволяє зберігати принципи SOLID, зокрема принципу інверсії залежностей (DIP) [18].

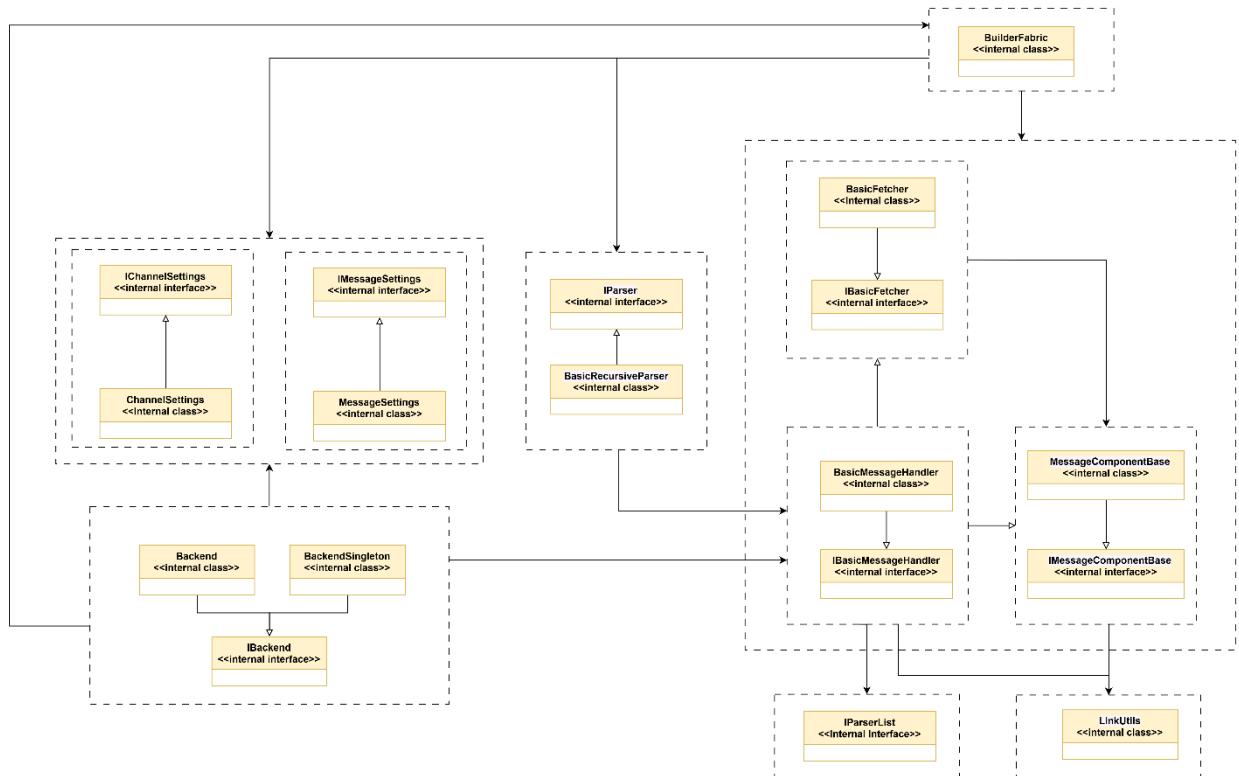


Рисунок 3.2 – Схема взаємодії компонентів парсеру

На рис. 3.3 наведено деталізацію класів які реалізують налаштування каналів та повідомлень. Клас `MessageSettings` та `ChannelSettings` умовно знаходяться на одному рівні через їх тематичну схожість – відповідають за налаштування парсеру. `ChannelSettings` відповідає за налаштування груп: заблоковані групи, видалити групу за ід, отримує список заблокованих груп від користувача. `MessageSettings` містить функціонал для налаштування повідомлень, відповідно до яких буде здійснюватися аналітика повідомлень.

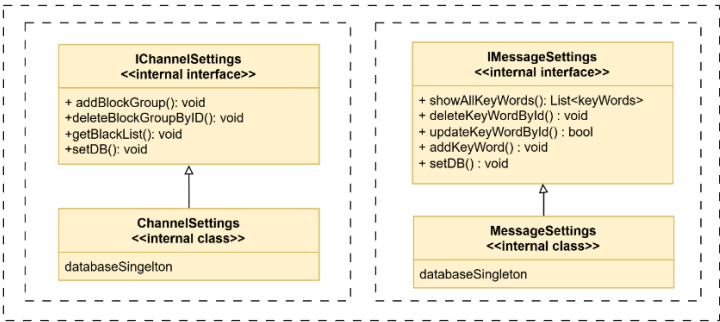


Рисунок 3.3 – Деталізація класів, які реалізують налаштування парсеру

На рис. 3.4 наведено деталізацію класу **BasicRecursiveParser** та **MessageComponentBase** відповідно.

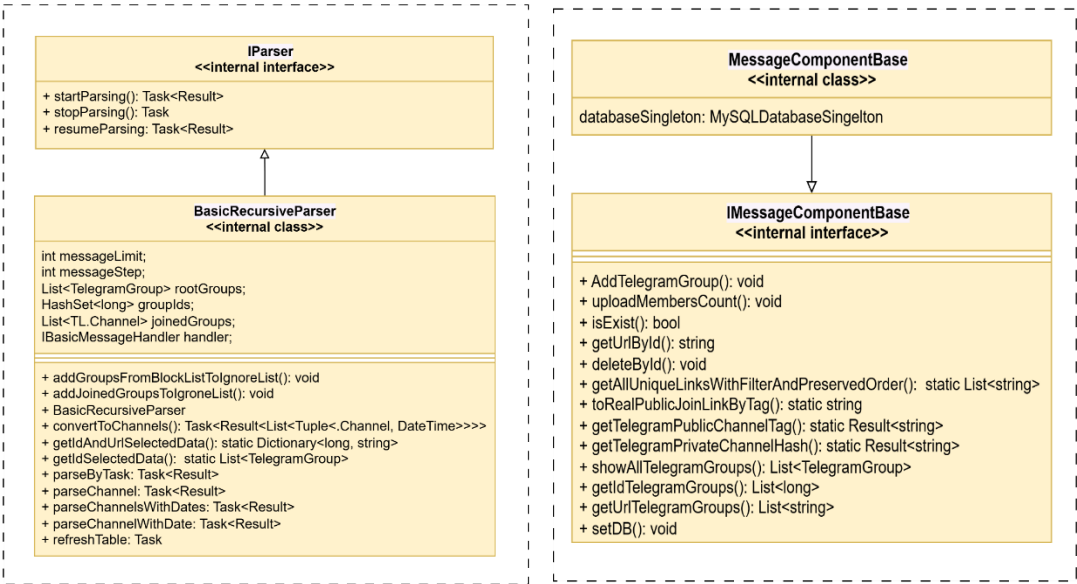


Рисунок 3.4 – Деталізація класів та їх інтерфейсів

На рис. 3.5 наведено деталізацію класів **BasicFetcher** та **BasicMessageHandler**, також їх інтерфейсів: **IBasicFetcher** та **IBasicMessageHandler**.

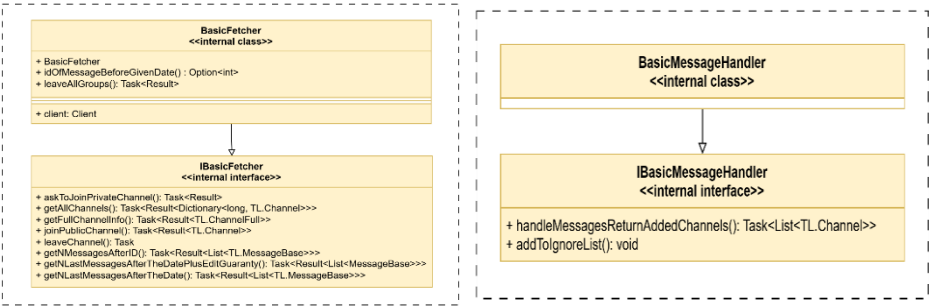


Рисунок 3.5 – Деталізація класів та їх інтерфейсів

На рис. 3.6 наведено деталізацію класу LinkUtils та інтерфейсу IParserList.

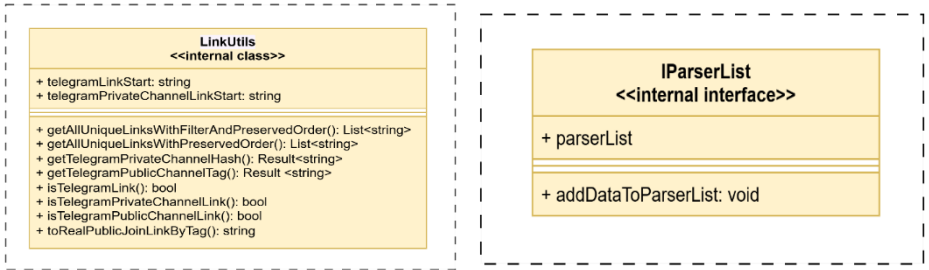


Рисунок 3.6 – Деталізація класу та інтерфейсу

На рис. 3.7 неведено деталізацію класу BuilderFabric. Даний клас використовується для створення нових об’єктів інших класів, що дозволяє зменшити кількість залежностей між різними класами.

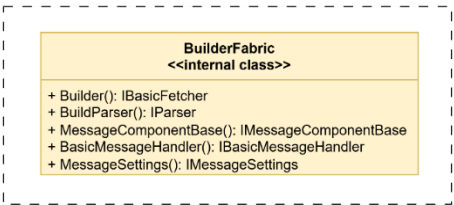


Рисунок 3.7 – Деталізація класу BuilderFabric

Для авторизації в телеграмі використовується лише один клас: Auth. Він зберігає в собі конфігураційні дані які дозволять здійснити авторизацію в

телеграмі. На рис 3.8 відображено яким чином здійснюється обмін даних в проєкті. Котроллер звертається до даного класу для отримання авторизаційних даних.

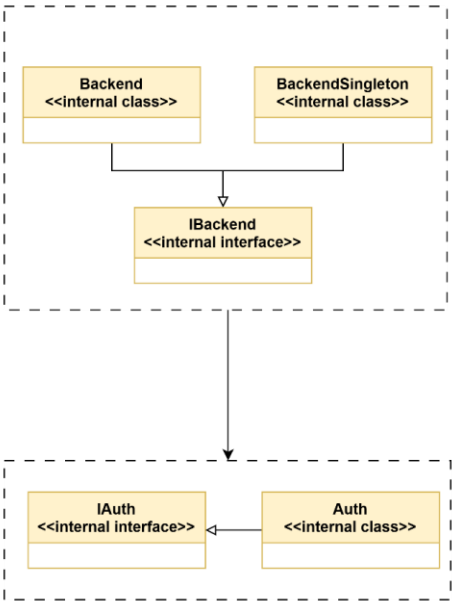


Рисунок 3.8 – Схема взаємодії компонентів авторизатору

На рис. 3.9 відображено деталізацію класу **Auth** та його інтерфейсу **IAuth**, яка містить в собі опис методів класу.

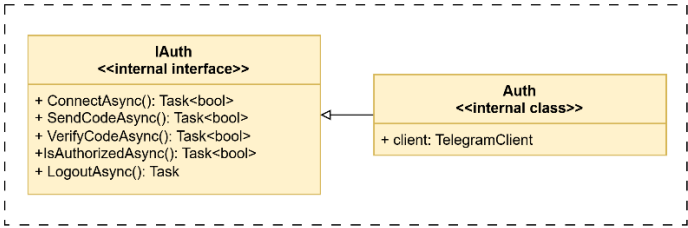


Рисунок 3.9 – Деталізація класу-авторизатора

На рис. 3.10 наведено схему взаємодії компонентів аналітичного модулю. Клас **RealTime** та **PrepareModel** взаємодіє з контроллером **IBackend**, який дозволяє взаємодіяти з користувачем.

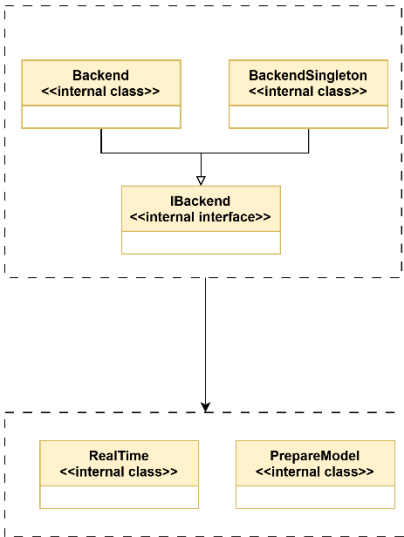


Рисунок 3.10 – Схема взаємодії компонентів аналітичного модулю

На рис. 3.11 наведено деталізацію методів класу RealTime та PrepareModel.

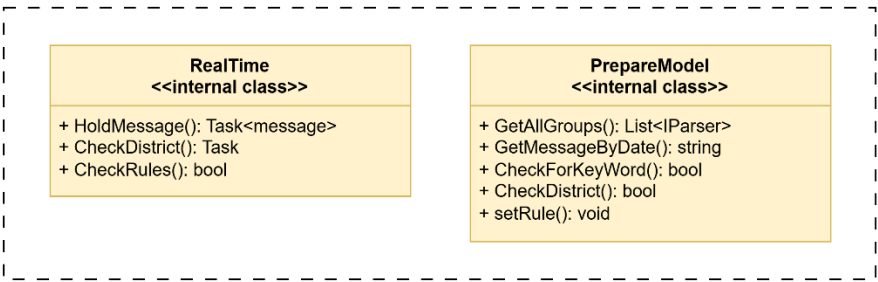


Рисунок 3.11 – Деталізація класів аналітичного модулю

Таблиця 3.2 – Опис DataLinkLayer

Модуль	Структура	Функція
controller	IBackend Backend	Реалізує логіку взаємодії з користувачем: парсинг, обробка запитів, збереження налаштувань.
	BackendSingleton	Містить той самий функціонал, що й Backend, але відповідає за те, щоб даний клас був єдиний в проєкті

На рис. 3.12 наведено опис методів та змінних класів Backend та BackendSingleton, а також інтерфейсу IBackend, які використовуються для реалізації котролеру.

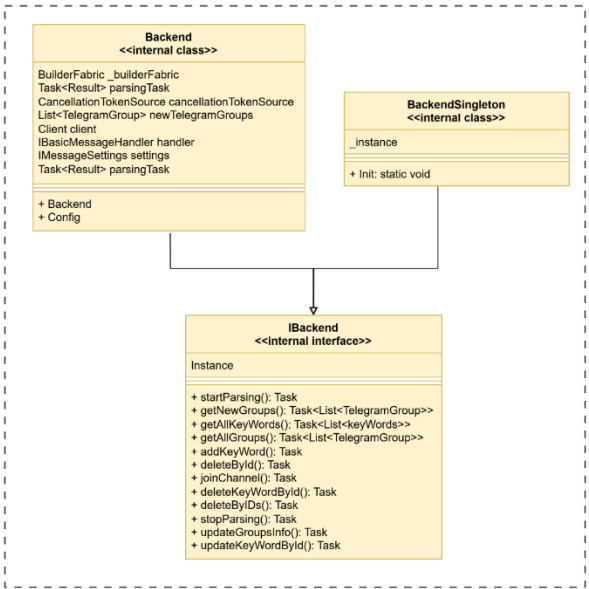


Рисунок 3.12 – Деталізація компонентів котролеру

Таблиця 3.3 – DataStorageLayer

Модуль	Структура	Функція
Збереження даних	IDatabase MySQLDatabase	Отримати посилання на Telegram-групи, які наявні в БД; Отримати ІД усіх ТГ-груп; Отримати ід групи за назвою; Додає ТГ-групу до БД; Відобразити інформацію про усі групи, які є в БД; Видалити групу за ід; Оновлює інформацію про групи; Оновлює інформацію про країни, регіони та міста.
	MySQLDatabaseSingleton	Функціонал аналогічний до класу MySQLDatabase, але забезпечує єдиний екземпляр в усьому проєкті.

На рис. 3.13 відображено деталізацію класів MySQLDatabase та MySQLDatabaseSingelton, а також інтерфейсу від якого вони наслідуються IDatabse.

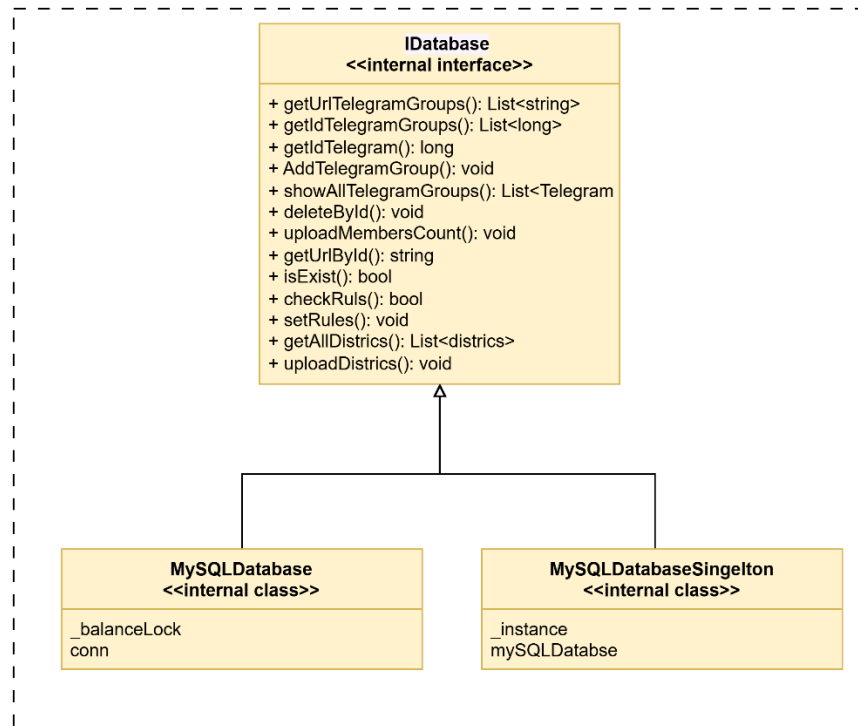


Рисунок 3.13 – Деталізація шару DataStorage

3.3 Проєктування бази даних

Розроблювальне програмне забезпечення зберігає дані в базі даних щодо Telegram груп, регіонів та правила за допомогою яких буде здійснюватися прогназування подальшого пересування БПЛА. На рис. 3.14 відображено концептуальну (логічну модель) бази даних.

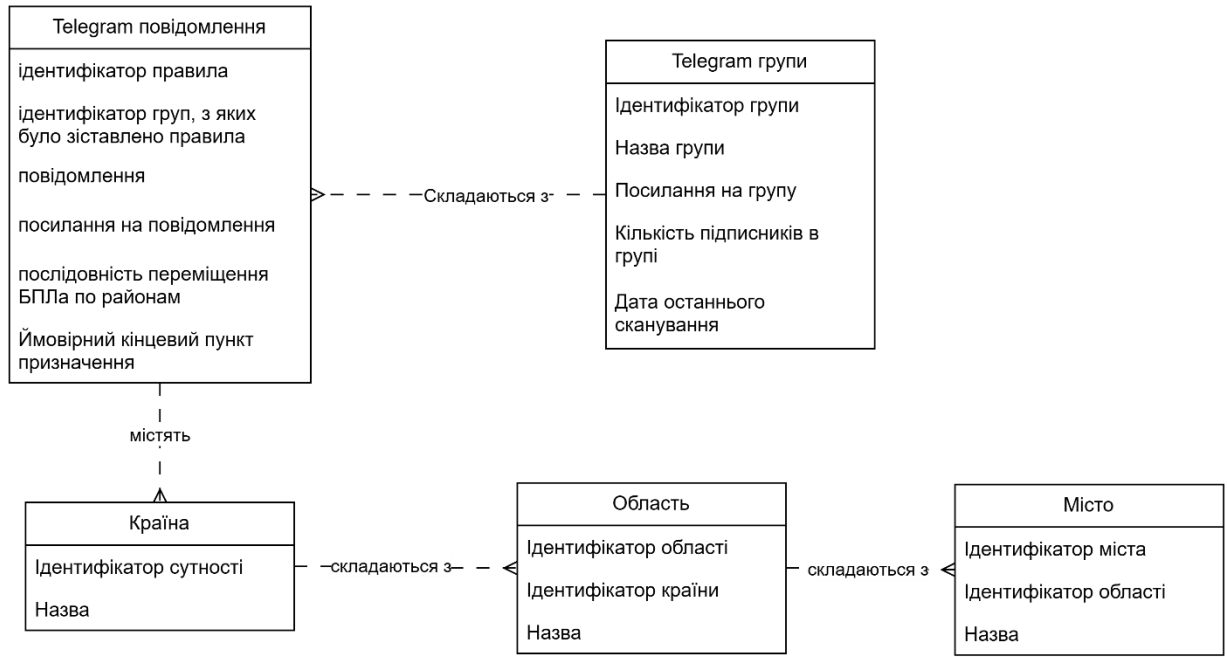


Рисунок 3.14 – Концептуальна модель бази даних [22]-[24]

Сутність TelegramGroup містить в собі інформацію про канали, які були виявлені за вказаною темою.

Сутність TelegramMessage містить в собі набір правил, які є ключовими для аналізу історії пересування БПЛА.

Сутності country, area та city об’єднують в собі набір правил, які використовуються для ідентифікації місця пересування БПЛА.

Опис полів та типів сутностей описано в таблиці 3.4.

Таблиця 3.4 – Опис сутностей

№	Назва поля	Тип	Опис
TelegramMessage			
1	Id	BIGINT	ідентифікатор правила
2	id_Telegram	BIGINT	ідентифікатори груп, які були використані для зіставлення правила

Кінець таблиці 3.4

№	Назва поля		Тип	Опис
TelegramMessage				
3	messages		TEXT	текстове повідомлення
4	url_messages		TEXT	посилання на повідомлення
5	districs		TEXT	Послідовність переміщення
6	Result		BIGINT	Ймовірний кінцевий пункт призначення БПЛА
TelegramGroup				
1	idTelegram		BIGINT	ідентифікатор групи
2	groupName		TEXT	назва групи
3	membersCount		INT	кількість підписників в групі
4	dateTimeLatestScan		DATE	дата останнього сканування
5	url		TEXT	посилання на групу
Country				
1	Id	BIGINT	Ідентифікатор сутності країни	
2	Area_name	Text	Назва країни	
Area				
1	Id	BIGINT	Ідентифікатор області	
2	Id_country	BIGINT	Ідентифікатор країни	
3	Area_name	Text	Назва області	
City				
1	Id	BIGINT	Ідентифікатор міста	
2	Id_area	BIGINT	Ідентифікатор області	
3	City_name	Text	Назва міста	

На рис. 3.15 відображено ER-діаграму.

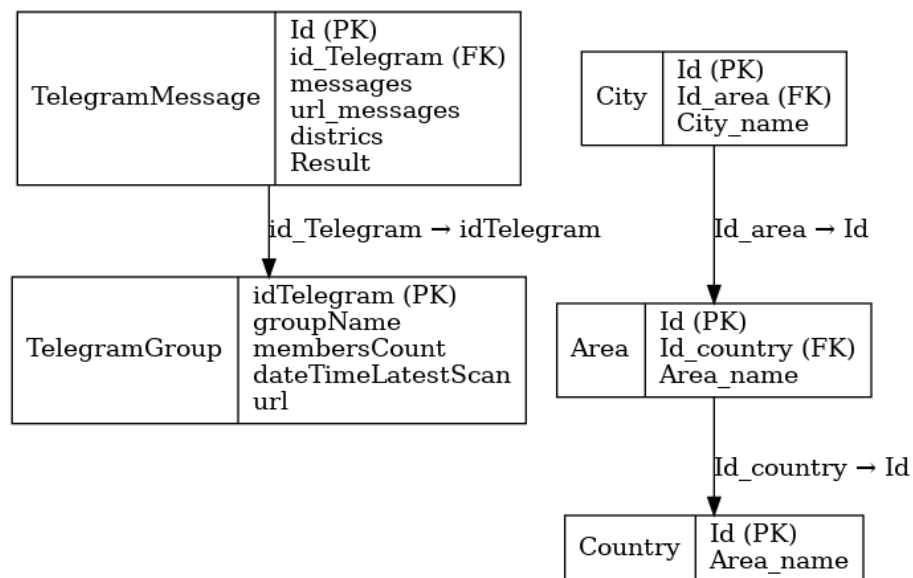


Рисунок 3.15 – ER-діаграма [22]-[24]

3.4 Опис алгоритмів функціонування застосунку

Алгоритм авторизації складається з декількох основних кроків:

- отримання номеру телефону від користувача;
- система надсилає одноразовий код (ОПТ) для підтвердження входу;
- користувач вводить отриманий код. Клієнтська частина передає його на сервер для перевірки;
- у разі успішної перевірки сервер встановлює сесію й повертає її клієнту.

На рис. 3.16 наведено блок-схему функціонування алгоритму авторизації.

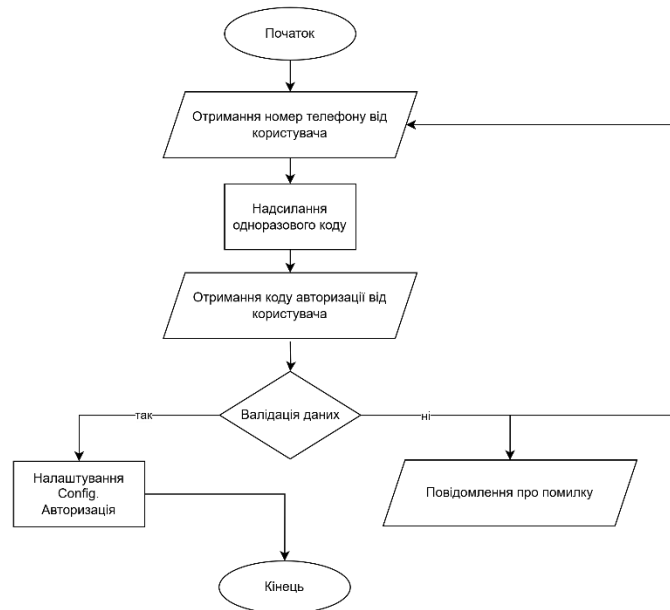


Рисунок 3.16 – Алгоритм процесу авторизації

Алгоритм парсеру містить в собі рекурсивний характер. Він працює до тих пір, поки користувач не зупинить парсер.

Основні етапи роботи парсеру:

- отримання груп, обраних користувачем, на основі яких розпочинається робота парсеру;
- отримуються усі повідомлення з першої обраної групи;
- починається поступова обробка повідомлень. Якщо повідомлення містить посилання на групу та посилання є дійсне, то відбувається вступ, в іншому випадку парсер продовжує обробку інших повідомлень;
- після завершення обробки повідомлень, першої обраної групи користувачем, починається обробка наступної групи, що була обрана користувачем та програма переходить до 2-го кроку;

– після обробки усіх обраних груп користувачами, парсер розпочинає пошук посилань для вступу в канали з каналів, які були знайдені в результаті роботи парсеру з попередніх кроків.

На рис. 3.17 відображено алгоритм функціонування парсеру.

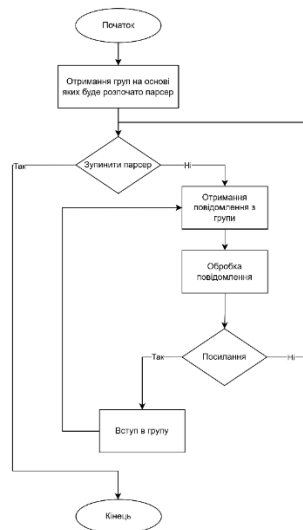


Рисунок 3.17 – Алгоритм функціонування парсеру

3.5 Обробка виняткових ситуацій

Для того, щоб програмістам було простіше виявити можливі проблеми з роботою програми, було додано StackTrace для усіх функцій застосунку. Повідомлення умовно розділено на 2 типи: інформаційні (info) та ті, які були не передбачені програмою (warning та error). На рис. 3.18 наведено приклад файлу в який записано результати обробки таких подій.

```

16:45:13 понеділок, 12 травня 2025 р.
Info: showAllTelegramGroupsFromDataBase Returned 129 groups
16:45:14 понеділок, 12 травня 2025 р.
Info: showAllKeywords Returned 3 words
17:21:53 понеділок, 12 травня 2025 р.
Info: showAllTelegramGroupsFromDataBase Returned 129 groups
17:21:53 понеділок, 12 травня 2025 р.
Info: showAllKeywords Returned 3 words
  
```

Рисунок 3.18 – Запис результатів в файл log.txt

Окрім логування даних, у разі виникнення помилок, відбувається сповіщення користувача за допомогою інструменту: MessageBox. На рис. 3.19 відображено інформацію про помилку.

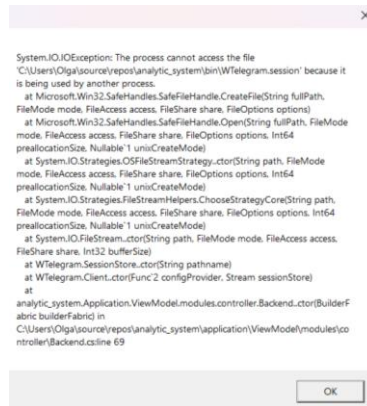


Рисунок 3.19 – Повідомлення користувачу про помилку

3.6 Основні рішення щодо розробки інтерфейсу

Для розроблювального застосунку було розроблено одну сторінку на якій знаходяться вкладки:

- вкладка «Головне меню» (рис. 3.20), де відображено опис застосунку;
- вкладка «Парсер» (рис. 3.21) на якій аналітик має змогу вибрати групи для початку парсеру груп, а також, для керування групами;
- вкладка «Налаштування» (рис. 3.22) дозволяє аналітику керувати заблокованими групами, а також керувати ключовими словами та інформацією про регіони, які наявні в БД;
- вкладка «Аналітика» (рис. 3.23) дозволяє аналітику додавати правила щодо переміщення БПЛА до БД.

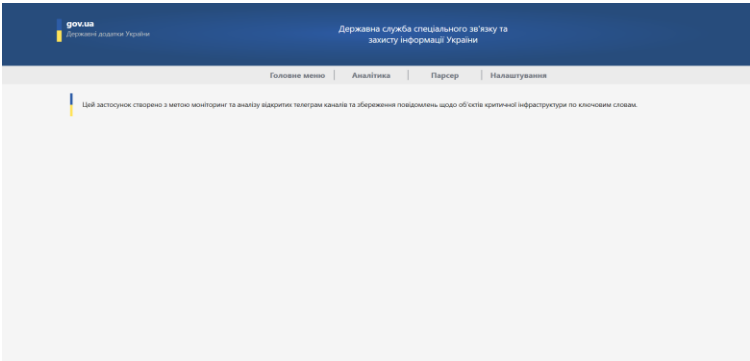


Рисунок 3.20 – Вкладка «Головне меню»

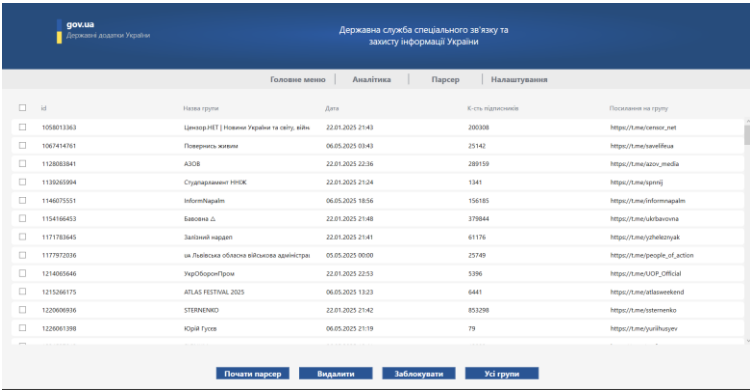


Рисунок 3.21 – Вкладка «Парсер»

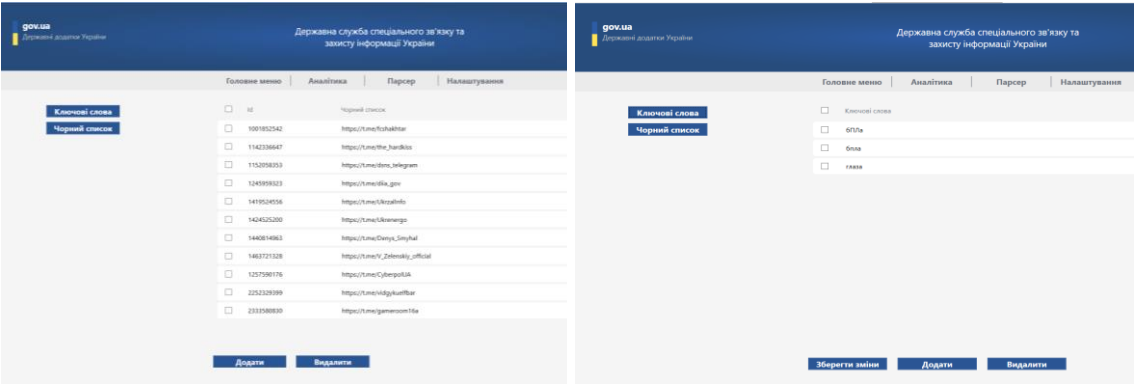


Рисунок 3.22 – Вкладка «Налаштування»

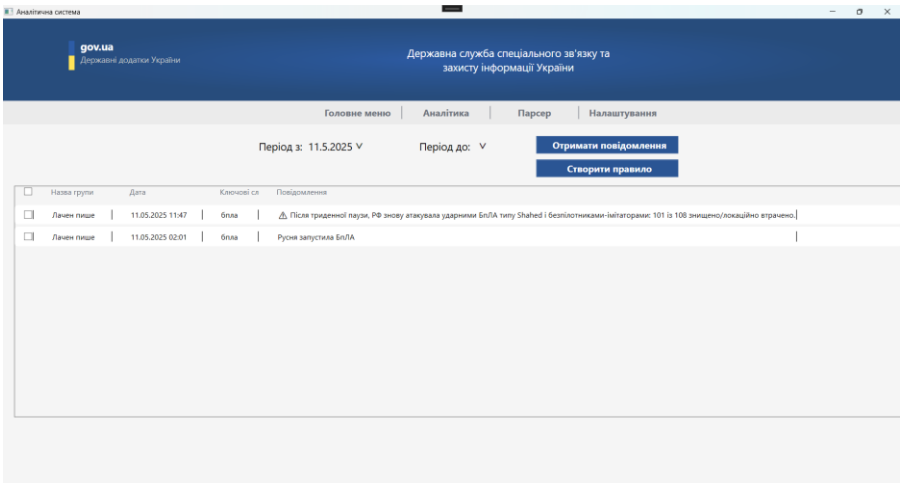


Рисунок 3.23 – Вкладка «Аналітика»

Окрім вищезазначених вкладок, було розроблено сторінки, які призначені для авторизації користувача:

- отримання номеру телефону від користувача (рис. 3.24);
- отримання коду підтвердження (рис. 3.25).

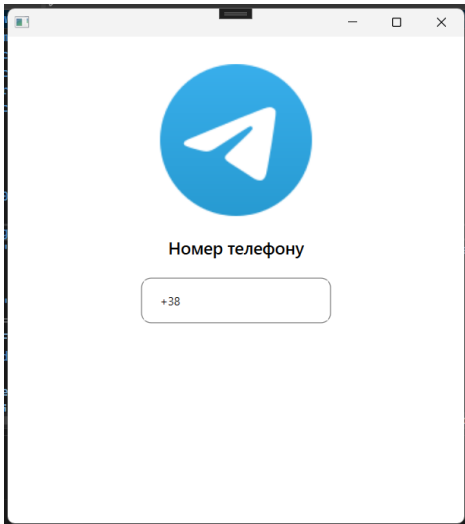


Рисунок 3.24 – Сторінка вводу номеру телефону

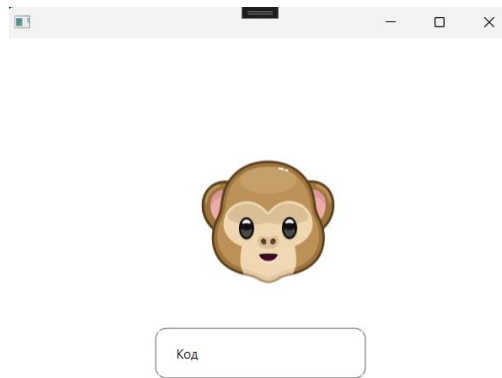


Рисунок 3.25 – Сторінка для вводу коду-підтвердження

3.7 Висновок за розділом 3

У 3-му розділі було описано практичну реалізацію застосунку, а саме: архітектуру, що використовується; опис модулів з яких складається система з схемами, що демонструють їх взаємодію; опис концептуальної моделі бази даних та основні рішення, які були використані під час реалізації графічного інтерфейсу. Крім цього, у 3-му розділі було описано яким чином обробляються виняткові ситуації застосунком.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Програма призначена для аналітики даних з відкритих джерел про пересування ворожих бпла з метою захисту об'єктів критичної інфраструктури.

4.2 Умови використання системи

Для нормального функціонування програми, повинні виконуватися наступні умови апаратної частини клієнту:

- операційна система Windows 10 (x64);
- .NET v6;
- 2 ядерний процесор 1.8 GHz;
- 2 Гб ОЗУ;
- 150 мб вільного місця;
- вбудована відеокарта;
- підключення до інтернету.

4.3 Експлуатація програми

4.3.1 Функціональні можливості програми

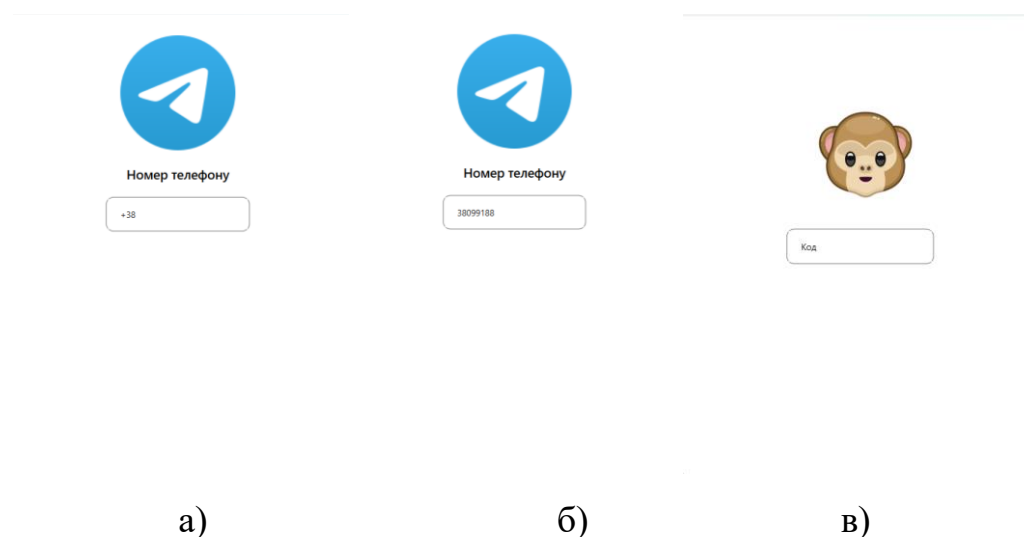
Функціональні можливості аналітика:

- авторизація в системі;
- розпочати парсер груп в Telegram;
- здійснювати видалення та блокування знайдених груп;
- перегляд усіх груп;
- можливість редагувати, додавати та видаляти ключові слова;

- можливість додавати та видаляти групи, які було додано до чорного списку;
- пошук повідомлень за ключовими словами в Telegram-групах з можливістю здійснювати аналітиком редагування отриманих даних, які необхідні для подальшого прогнозування;
- створення правил «пересування БПЛА» для подальшої роботи.

4.3.2 Виконання програми

Після запуску застосунку у користувача з'являється можливість здійснити авторизацію у ньому, як показано на рис. 4.1. Після чого, користувач має ввести свій номер телефону, як поканазно на рис. 4.1. Наступним етапом авторизації є введення коду, який був надісланий власнику номеру телефону в Telegram або sms-повідомленням на телефон (рис. 4.1).



а) початкова сторінка; б) ввід номеру телефону; в) ввід ОРТ-коду.

Рисунок 4.1 – Авторизація в застосунку

У разі успішної авторизації користувач побачить головне вікно, у якому буде надано опис застосунку, як відображено на рис. 4.2.

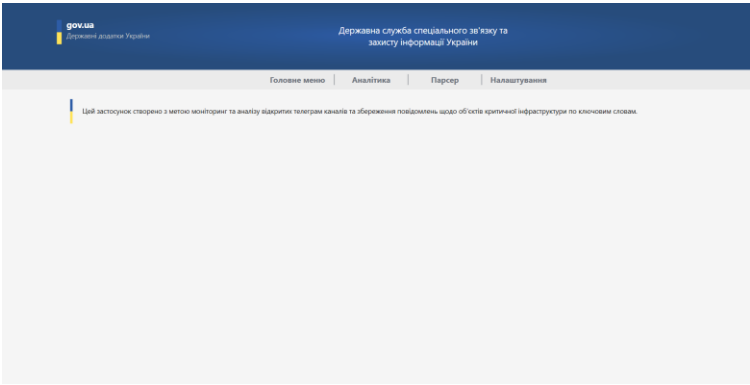


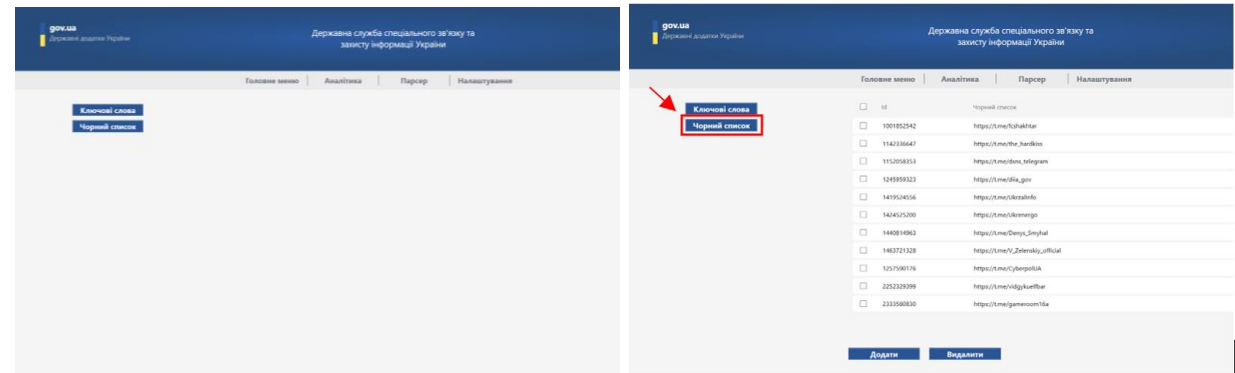
Рисунок 4.2 – Головна сторінка

На панелі управління користувач бачить можливості перейти до аналітики, парсеру та налаштувань застосунку.

Користувач має змогу керувати заблокованими групами, а саме:

- додавати нові;
- видаляти існуючі групи.

На рис. 4.3 наведено початкове вікно вкладки «Налаштування». Для того, щоб відобразити заблоковані групи треба натиснути на кнопку, що розташована на панелі зліва та має напис «Чорний список», як показано на рис. 4.3.



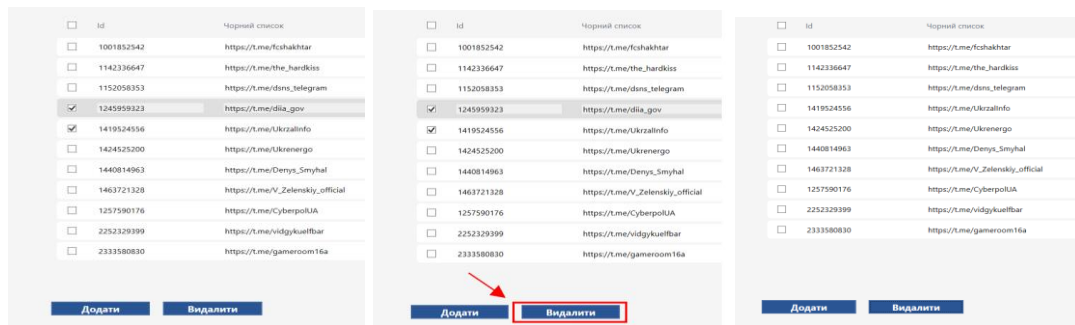
а)

б)

а) початкове вікно; б) відображено список з заблокованими групами.

Рисунок 4.3 – Заблоковані групи

Для того, щоб видалити заблоковану групу з переліку, необхідно поставити галочку біля бажаної групи (рис. 4.4) та натиснути на кнопку «Видалити» (рис. 4.4). Результат видалення наведено на рис. 4.4.



а)

б)

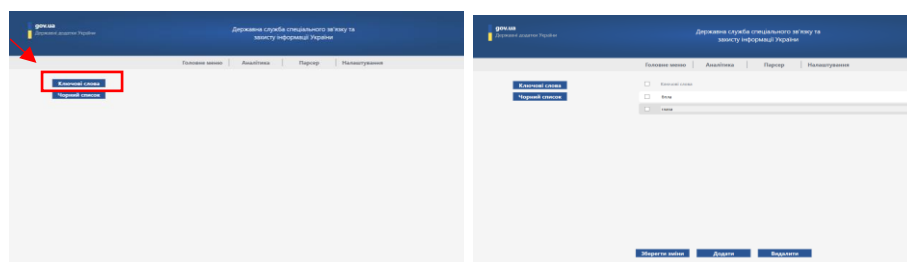
в)

а) вибір груп; б) видалення груп; в) результат видалення.

Рисунок 4.4 – Керування заблокованими групами

Окрім керування заблокованими групами, у користувача є можливість налаштовувати «Ключові слова», які використовуються для виявлення повідомлень. Користувач може видаляти, додавати та редагувати ключові слова.

На рис. 4.5 наведено початкове вікно вкладки «Налаштування». Для того, щоб відобразити ключові слова треба натиснути на кнопку, що розташована на панелі зліва та має напис «Ключові слова». Результат відображення ключових слів наведено на рис. 4.5.



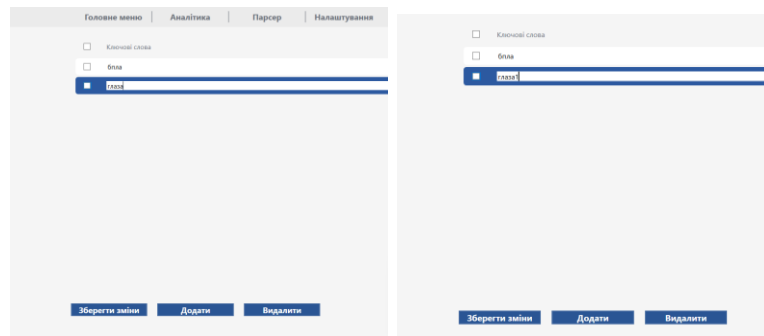
а)

б)

а) початкове вікно; б) відображено список з ключовими словами

Рисунок 4.5 – Керування ключовими словами

Для того, щоб здійснити редагування ключового слова, необхідно клікнути по ньому 2 рази (рис. 4.6), ввести нове значення (рис. 4.6), після чого натиснути на панелі знизу на кнопку «Зберегти дані».



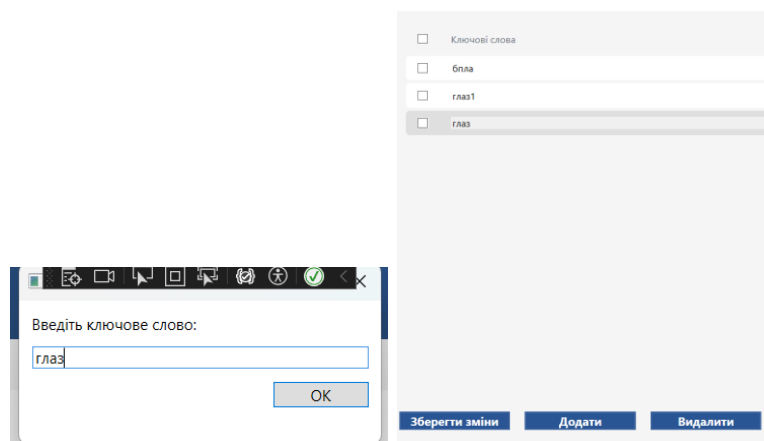
а)

б)

а) вибір ключового слова для редагування; б) внесення змін.

Рисунок 4.6 – Редагування ключових слів

Для того, щоб додати нове ключове слово необхідно натиснути на кнопку «Додати», яка знаходиться на панелі знизу. Після чого відкривається спливаюче вікно для додавання нового ключового слова.



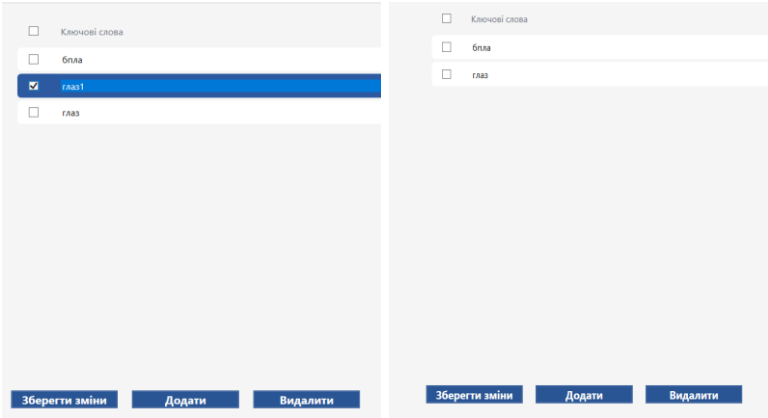
а)

б)

а) введення ключового слова; б) результат додавання нового ключового слова

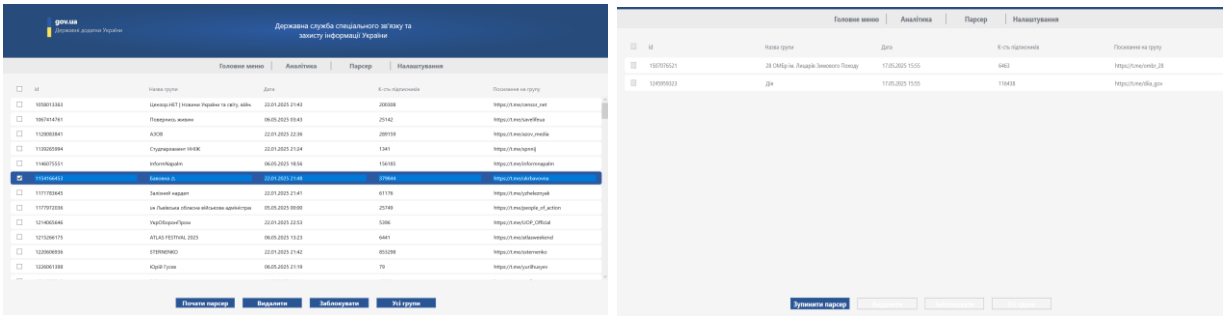
Рисунок 4.7 – Додавання ключових слів

Для видалення ключових слів необхідно вибрати необхідні елементи в checkbox (рис. 4.8), після чого натиснути на кнопку «Видалити». На рис. 4.8 неведено результат видалення.



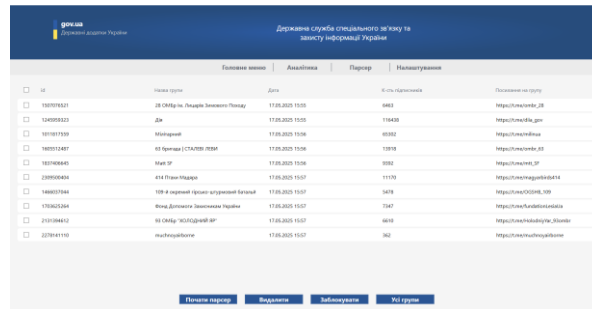
а) вибір значення; б) результат видалення.
Рисунок 4.8 – Видалення ключових слів

Для запуску парсеру, необхідно на відповідній вкладці обрати групу на основі якої, буде запущено парсер (рис. 4.9), після чого, натиснути на кнопку «Почати парсер». На рис. 4.9 наведено результат роботи парсеру.



а) вибір групи; б) результат додавання груп.
Рисунок 4.9 – Запуск парсеру

Для того, щоб зупинити парсер, треба натиснути на кнопку «Зупинити парсер». Після цього, буде активовані інші кнопки, які дозволять керувати щойно знайденими групами, як наведено на рис. 4.10.



Група	Дата	К-ть повідомлень	Посилання на групу
☐ 160101001	20.08.2023 10:08	4802	https://m.facebook.com/160101001
☐ 160101002	20.08.2023 10:08	12008	https://m.facebook.com/160101002
☐ 160101003	20.08.2023 10:08	8002	https://m.facebook.com/160101003
☐ 160101004	20.08.2023 10:08	13019	https://m.facebook.com/160101004
☐ 160101005	20.08.2023 10:08	8002	https://m.facebook.com/160101005
☐ 160101006	20.08.2023 10:07	11019	https://m.facebook.com/160101006
☐ 160101007	20.08.2023 10:07	3478	https://m.facebook.com/160101007
☐ 160101008	20.08.2023 10:07	7547	https://m.facebook.com/160101008
☐ 160101009	20.08.2023 10:07	4616	https://m.facebook.com/160101009
☐ 160101010	20.08.2023 10:07	362	https://m.facebook.com/160101010

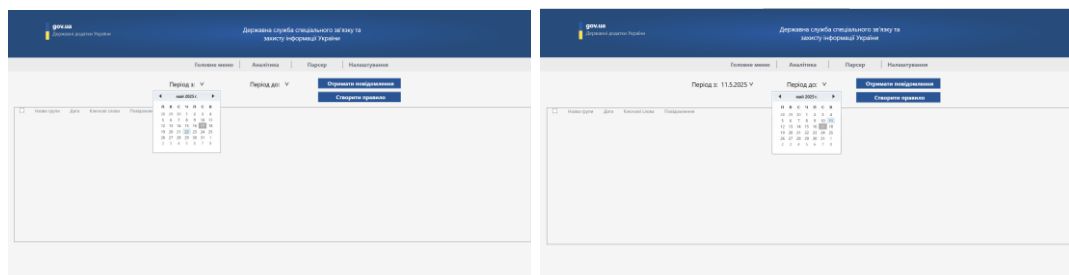
Рисунок 4.10 – Зупинка роботи парсеру

Для видалення або блокування груп необхідно:

- вибрати групу, виділивши їх у checkbox;
- натиснути на кнопку «Заблокувати» для додавання групи до чорного списку або «Видалити» для видалення групи.

Кнопка «Усі групи» дозволяє оновити інформацію на екрані про групи, в які було здійснено вступ.

На вкладці «Аналітика» у користувача є можливість вибрати діапазон дат для того, щоб розпочати збір повідомлень. Для цього необхідно натиснути на кнопку, яка знаходиться біля фраз «Період з» та «Період до». Приклад встановлення даних наведено на рис. 4.11.



а)

б)

а) період з; б) період до.

Рисунок 4.11 – Вибір дати для початку аналітики

Після встановлення дат необхідно натиснути на кнопку «Отримати повідомлення». Результат збору повідомлень наведено на рис. 4.12.

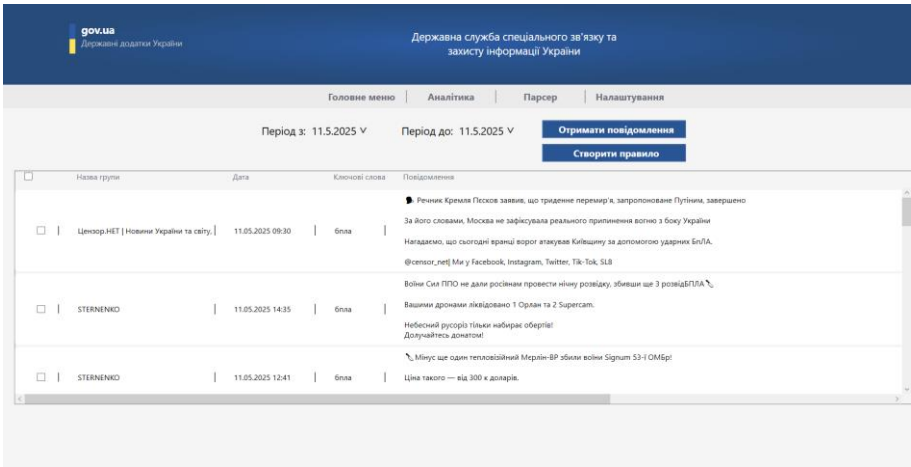


Рисунок 4.12 – Отримання повідомлень за обраний період

Після того, як було отримано повідомлення у користувача є можливість створити правило за яким можуть пересуватися бпла. Для цього користувачу необхідно виділити обрані елементи в полі checkbox, після чого натиснути на кнопку «Створити правило».

4.3.3 Тестування програми

Результати тестування застосунку наведено у чек-листі (таблиця 4.1).

Таблиця 4.1 – Чек-лист з тестування

№	Опис тесту	Windows 11
Парсер		
1	Видалення групи та вихід з неї	passed
2	Запуск парсеру груп	passed
3	Зупинка парсеру груп	passed

Кінець таблиці 4.1

№	Опис тесту	Windows 11
Парсер		
4	Додавання груп до чорного списку	passed
5	Відобразити усі групи	passed
Аналітика		
1	Вибір діапазону дат, на основі яких буде здійснено пошук повідомлень	passed
2	Отримання повідомлень з діапазону дат, які містять ключові слова	passed
3	Створення правила для подальшого аналізу	passed
4	Запуск прогнозу в реальному часі	Не реалізовано
Налаштування		
1	Редагування ключових слів	passed
2	Видалення ключових слів	passed
3	Додання ключового слова	passed
4	Видалення заблокованої групи	passed
5	Додати групу до чорного списку	passed

4.4 Висновок за розділом 4

В даному розділі було визначено, що мінімальними апаратними вимогами до пристрою на якому запускається програма є:

- операційна система Windows 10 (x64);
- .NET v6;
- 2 ядерний процесор 1.8 GHz;

- 2 Гб ОЗУ;
- 150 мб вільного місця;
- вбудована відеокарта;
- підключення до інтернету.

Було здійснено огляд функціональних можливостей застосунку та наведено чек-лист в якому наведено результати тестувань. Застосунок успішно пройшов усі тести. Один тест пропущено через відсутність його реалізації.

ВИСНОВКИ

У ході виконання дипломної роботи, проведено детальне ознайомлення із предметною областю – розроблення застосунку, що здійснює обробку отриманих даних з відкритих каналів інформації. Ознайомлення із найбільш важливими питаннями цієї проблемної області та найвагомішими рішеннями: було визначено, що для реалізації даного застосунку знадобиться реалізувати парсер, реалізувати алгоритми авторизації, збирання повідомлень відповідно до налаштувань користувача та здійснювати обробку подій від нього.

У першому розділі було розглянуто предметну область та визначено функціонал системи.

У другому розділі розглянуто та здійснено порівняння мов програмування, а також було здійснено огляд gitHub, API для Telegram, середу розробки Visual Studio, фреймворк WPF.

У третьому розділі було описано основні рішення, що були прийняті для реалізації алгоритмічної частини ПЗ, а саме: парсер, збір повідомлень за ключовими словами, авторизація користувачів та обробка подій від них.

Тестування ПП довело, що більшість запланованого функціоналу було реалізовано.

В подальшому можливо модернізувати ПП в напрямку аналізу даних, а саме: додати аналіз кожної групи окремо та зробити припущення згідно отриманих повідомлень про приналежність групи до певного регіону, це дозволить збільшити ефективність та точність роботи застосунку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Тиждень науки-2025. Факультет комп'ютерних наук і технологій. Тези доповідей науково-практичної конференції, Запоріжжя, 14–18 квітня 2025 р. – в друку.
2. Звіт про прямі збитки інфраструктури [Електронний ресурс]. – Режим доступу: <https://surli.cc/gbklvw>.
3. Інформація щодо масованих ударів по критичній інфраструктурі України [Електронний ресурс]. – Режим доступу: <https://map.ua-energy.org/uk/resources/12f3148d-841a-478d-b9ed-72bf0764b286/>.
4. Підсумок бойової роботи [Електронний ресурс]. – 2022. – Режим доступу: <https://t.me/kpszsu/2240>.
5. Підсумок бойової роботи [Електронний ресурс]. – 2023. – Режим доступу: <https://t.me/kpszsu/9208>.
6. Підсумок бойової роботи [Електронний ресурс]. – 2024. – Режим доступу: <https://t.me/kpszsu/25847>.
7. Звіт про бойову роботу [Електронний ресурс]. – 2024. – Режим доступу: <https://surl.li/sjwake>.
8. 9100 ракет та ударних дронів запустили росіяни по Україні із початку повномасштабного вторгнення [Електронний ресурс]. – Режим доступу: <https://surl.li/osmgyi>.
9. Навчальний посібник з дисципліни «Системний аналіз» для здобувачів спеціальності 122 – Комп'ютерні науки / укл. : В.М. Тонконогий, В.О. Вайсман, Л.В. Бовнегра, К.Г. Кіркопуло. – Одеса : Нац. ун-т «Одеська політехніка», 2022. – 84 с.
10. Методологія наукових досліджень : конспект лекцій з навчальної дисципліни для здобувачів другого (магістерського) рівня вищої освіти освітньої програми «Комп'ютерні науки» галузі знань 12 Інформаційні

технології спеціальності 122 Комп'ютерні науки денної та заочної форм навчання / укл. І. М. Козубцов. – Луцьк: ЛНТУ, 2022. – 242 с.

11. IBM QRadar SOAR. IBM [Electronic resource]. – Access mode : <https://www.ibm.com/products/qradar-soar>.

12. Розробка парсерів для Telegram: телеграм-канали, боти. Avada Media [Електронний ресурс]. – Режим доступу : <https://avada-media.ua/services/parser-telegram/>.

13. OpenCTI. Filigran [Electronic resource]. – Access mode : <https://filigran.io/solutions/open-cti/>.

14. Рейтинг мов програмування 2025. TypeScript і Python — найпопулярніші, частки C# та Java зменшуються [Електронний ресурс]. – Режим доступу : <https://dou.ua/lenta/articles/language-rating-2025/>.

15. What is Windows Forms - Windows Forms. Microsoft Learn: Build skills that open doors in your career [Electronic resource]. – Access mode : <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-9.0>.

16. What is Windows Presentation Foundation - WPF. Microsoft Learn: Build skills that open doors in your career [Electronic resource]. – Access mode : <https://learn.microsoft.com/en-us/dotnet/desktop/wpf/overview/?view=netdesktop-9.0>.

17. GitHub [Electronic resource]. – Access mode : <https://github.com>.

18. SourceForge.net [Electronic resource]. – Access mode : <https://sourceforge.net>.

19. Telegram APIs [Electronic resource]. – Access mode : <https://core.telegram.org>.

20. MySQL [Electronic resource]. – Access mode : <https://www.mysql.com>.

21. Download Enterprise Application Patterns Using .NET MAUI (PDF) | .NET. Microsoft. [Electronic resource]. – Access mode : <https://aka.ms/maui-ebook>.

22. Мартін, Р. Чистий код. Створення і рефакторинг за допомогою Agile [Текст] / Р. Мартін. – Харків: Фабула, 2019. – 448с.

23. Мартиненко Г. Ю. Тема лекції №10. Модель "сутність-зв'язок" (ER-діаграми) : конспект лекцій [Електронний ресурс] / Г. Ю. Мартиненко. — Режим доступу : https://web.kpi.kharkov.ua/dpm/wp-content/uploads/sites/123/2020/05/LEKTSIYA-10-11-OBD_ukr.pdf.

24. Мартиненко Г. Ю. Тема лекції №11. Приклад розробки простої ER-моделі : конспект лекцій [Електронний ресурс] / Г. Ю. Мартиненко. — Режим доступу : https://web.kpi.kharkov.ua/dpm/wp-content/uploads/sites/123/2020/05/LEKTSIYA-10-11-OBD_ukr.pdf.

ДОДАТОК А
Технічне завдання

Вступ

Програмний продукт призначений для аналітики даних з відкритих джерел про пересування ворожих бпла з метою захисту об'єктів критичної інфраструктури.

A.1 Підстави для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Розробка аналітичної системи аналізу та підтримки прийняття рішень Єдиної регіональної інформаційної системи захисту критичної інфраструктури», затверджене наказом Національного університету «Запорізька політехніка» № 209 від 28 квітня 2025 р.

A.2 Призначення розробки

Програма буде використовуватися для отримання даних з відкритих джерел, здійснювати їх обробку та здійснювати прогнозування пересування БПЛА.

A.2.1 Функціональне призначення

Програма містить функції, які надають можливість аналітику додавати, видаляти та редагувати ключовими словами за якими здійснюється збір повідомлень, а також містить парсер, який автоматично збирає групи до яких здійснюється вступ. У аналітика має бути можливість здійснювати збір та обробку даних за ключовими словами за певним періодом.

A.2.2 Експлуатаційне призначення

Користувач має експлуатувати програму на десктопному пристрої, які задовольняють мінімальні технічні вимоги для нормального функціонування застосунку.

A.3 Вимоги до програми або програмного продукту

A.3.1 Вимоги до функціональних характеристик

A.3.1.1 Користувач

Користувач / аналітик — це людина, яка бажає користуватися проєктуємим програмним застосунком.

A.3.1.2 Вимоги до складу виконуваних функцій

Функціональні можливості програми:

- здійснювати авторизацію в системі шляхом авторизації в Telegram;
- розпочати парсер груп в Telegram;
- здійснювати видалення та блокування знайдених груп;
- перегляд усіх груп;
- можливість редагувати, додавати та видаляти ключові слова;
- можливість додавати та видаляти групи, які було додано до чорного списку;
- пошук повідомлень за ключовими словами в Telegram-групах з можливістю здійснювати аналітиком редагування отриманих даних, які необхідні для подальшого прогнозування.

A.3.2 Вимоги до надійності

Імовірність безвідмовної роботи системи повинна бути не менше 99%.

A.3.2.1 Вимоги до забезпечення надійного функціонування програми

Функціонування надійного програмного забезпечення має забезпечитись виконанням сукупності організаційно-технічних заходів, насамперед:

- організацією безперебійного живлення технічних засобів;
- використанням ліцензійного програмного забезпечення;
- регулярним виконанням вимог стандартів захисту інформації.

A.3.3 Умови експлуатації

A.3.3.1 Вимоги до складу та параметрів технічних засобів

Для нормального функціонування програми, повинні виконуватися наступні умови апаратної частини клієнту:

- операційна система Windows 10 (x64);
- .NET v6;
- 2 ядерний процесор 1.8 GHz;
- 2 Гб ОЗУ;
- 150 мб вільного місця;
- вбудована відеокарта;
- підключення до інтернету.

A.3.8 Спеціальні вимоги

Програма має здійснювати взаємодію з користувачем використовуючи графічний інтерфейс, кольорова гама якого, має відповідати кольорам ДССЗЗІ.

A.4 Стадії та етапи розробки

Розробка повинна бути проведена в 3 стадії. На першій стадії має бути затверджено технічне завдання; на другій розроблено проєкт; на останій стадії відбувається впровадження готового продукту.

Виконавець повинен розробити десктопний застосунок та надати вихідні коди та документацію до розробленої програми не пізніше 30.05.2025.

ДОДАТОК Б
Текст програми

Б1. Текст файлу MySQLDatabase.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.OleDb;
using System.IO;
using System.Windows;
using MySqlConnector;
using System.Data;
using System.Diagnostics;
using System;
using System.Collections.ObjectModel;
using System.Data.SqlClient;
using analytic_system.Application.DataStructure;

namespace analytic_system.Application.Model
{
    class MySQLDatabase : IDatabase
    {
        MySqlConnection conn;
        private readonly System.Threading.Lock _balanceLock = new();
        /*
        * Передумова: передається назва групи, для якої необхідно
отримати ідентифікатор телеграм групи
        * Результат: Повертає ід телеграм-каналу/чату/соціальної мережі
за назвою групи
        */
        public MySQLDatabase(string connStr)
        {
            conn = new MySqlConnection(connStr);
            conn.Open();
        }
        public long getIdTelegram(string groupName) =>
            executeScalar("SELECT idTelegram FROM Telegram WHERE
groupName = @groupName", groupName) ?? 0;

        public void AddTelegramGroup(string from, long id, string
groupName, int memberCount, DateTime dateTimeLatestScan, string url) =>
            AddTelegramGroupToDatabase(from,
                "INSERT INTO TelegramGroup (idTelegram, groupName, membersCount,
dateTimeLatestScan, url) VALUES (@id, @groupName, @memberCount,
@dateTimeLatestScan, @url)",
                id, groupName, memberCount, dateTimeLatestScan, url);
        public static void Log(StackTrace st, Exception ex)
        {
            using (StreamWriter w = File.AppendText("log.txt"))
            {
                string stackIndent = "";

                w.WriteLine($"{DateTime.Now.ToLongTimeString()}
{DateTime.Now.ToLongDateString()}");
                w.WriteLine("Warning: " + ex.Message);

                for (int i = 0; i < st.FrameCount; i++)
                {

```

```

StackFrame sf = st.GetFrame(i);
w.WriteLine(stackIndent + "    Method:  {0}" +
sf.GetMethod(), w);
w.WriteLine(stackIndent + "    File:    {0}" +
sf.GetFileName(), w);
w.WriteLine(stackIndent + "    Line Number: {0}" +
sf.GetFileLineNumber(), w);
    stackIndent += "  ";
}
}
}
public static void LogInfo(string method, string description)
{
    using (StreamWriter w = File.AppendText("log.txt"))
    {
        string stackIndent = "";

        w.WriteLine($"{DateTime.Now.ToLongTimeString()}
{DateTime.Now.ToLongDateString()}");
        w.WriteLine($"Info: {method} {description}");
    }
}

private long? executeScalar(string query, string groupName)
{
    lock (_balanceLock)
    {
        try
        {
            MySqlCommand cmd = new MySqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@groupName",
groupName);

            using (MySqlDataReader rdr = cmd.ExecuteReader())
            {
                rdr.Read();
                return Convert.ToInt64(rdr["idTelegram"]);
            }
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
    }
}

public void addKeyWord(string word)
{
    try
    {
        if (word == null || word.Length == 0)
        {
            return;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    lock (_balanceLock)
    {
        try

```

```

        {
            string query = "INSERT INTO KeyWords (keyWords)
VALUES (@word_)";
            MySqlCommand mySqlCommand = new MySqlCommand(query,
conn);
            mySqlCommand.Parameters.AddWithValue("@word_",
word);
            int rdr = mySqlCommand.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
    }
    private void AddTelegramGroupToDatabase(string from, string
query, long idTelegram, string groupName,
        int membersCount, DateTime dateTimeLatestScan, string url)
    {
        lock (_balanceLock)
        {
            try
            {
                MySqlCommand command = new MySqlCommand(query,
conn);
                command.Parameters.AddWithValue("@id", idTelegram);
                command.Parameters.AddWithValue("@groupName",
groupName);
                command.Parameters.AddWithValue("@memberCount",
membersCount);
                command.Parameters.AddWithValue("@dateTimeLatestScan", dateTimeLatestScan);
                command.Parameters.AddWithValue("@url", url);
                int rdr = command.ExecuteNonQuery();
                MessageBox.Show($"Added {idTelegram} {url} from
{from}");
                LogInfo("AddTelegramGroupToDatabase", $"Added
{idTelegram} {url}");
            }
            catch (Exception ex)
            {
                Log(new StackTrace(ex, true), ex);
                throw;
            }
        }
    }

    public List<long> getIdTelegramGroups()
    {
        lock (_balanceLock)
        {
            string query = "SELECT idTelegram FROM TelegramGroup";
            List<long> tableList = new List<long>();

            try
            {

```

```

        MySqlCommand command = new MySqlCommand(query,
conn);

        using (var reader = command.ExecuteReader())
        {
            while (reader.Read())
            {
                tableList.Add(Convert.ToInt64(reader["idTelegram"]));
            }
        }
    }
    catch (Exception ex)
    {
        Log(new StackTrace(ex, true), ex);
        throw;
    }
    LogInfo("getIdTelegramGroups", $"Returned
{tableList.Count} groups");
    return tableList;
}

public bool isExist(long idTelegram)
{
    bool result = false;
    try
    {
        result = getIdTelegramGroups().Contains(idTelegram);
        LogInfo("isExist", $"Checking {idTelegram} witStaus
{result}");
    }
    catch (Exception ex)
    {
        Log(new StackTrace(ex, true), ex);
        throw;
    }
    return result;
}

public string getUrlById(long id)
{
    lock (_balanceLock)
    {
        string query = "SELECT url FROM TelegramGroup WHERE
idTelegram = @id";

        try
        {
            MySqlCommand command = new MySqlCommand(query,
conn);

            command.Parameters.AddWithValue("@id", id);
            using (var reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    string url = reader.GetString("url");
                    LogInfo("getUrlById", $"Returned {url}");
                    return url;
                }
                else
                {

```

```

        LogInfo("getUrlById", $"No data found for id
{id}");
        return null; // Повертаємо null, якщо запис
не знайдено
    }
}
catch (Exception ex)
{
    Log(new StackTrace(ex, true), ex);
    throw;
}
}
}
public void uploadMembersCount(long id, int membersCount)
{
    lock (_balanceLock)
    {
        string query = "UPDATE TelegramGroup SET membersCount =
@membersCount WHERE idTelegram = @id";
        try
        {
            MySqlCommand command = new MySqlCommand(query,
conn);
            command.Parameters.AddWithValue("@membersCount",
membersCount);
            command.Parameters.AddWithValue("@id", id);
            command.ExecuteNonQuery();
            LogInfo("uploadMembersCount", $"Updated {id} with
{membersCount}");
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
    }
}
public void deleteKeyWordById(int id_)
{
    lock (_balanceLock)
    {
        string query = "DELETE FROM KeyWords WHERE id = @id_";
        try
        {
            MySqlCommand mySqlCommand = new MySqlCommand(query,
conn);
            mySqlCommand.Parameters.AddWithValue("@id_", id_);
            mySqlCommand.ExecuteNonQuery();
            MessageBox.Show("I'm in database");
            LogInfo("deleteById", $"Deleted {id_}");
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
    }
}
public void deleteById(long id)

```

```

        {
            lock (_balanceLock)
            {
                string query = "DELETE FROM TelegramGroup WHERE
idTelegram = @id";
                try
                {
                    MySqlCommand command = new MySqlCommand(query,
conn);

                    command.Parameters.AddWithValue("@id", id);
                    command.ExecuteNonQuery();
                    LogInfo("deleteById", $"Deleted {id}");
                }
                catch (Exception ex)
                {
                    Log(new StackTrace(ex, true), ex);
                    throw;
                }
            }
        }
        public List<string> getUrlTelegramGroups()
        {
            lock (_balanceLock)
            {
                string query = "SELECT url FROM TelegramGroup";
                List<string> tableList = new List<string>();

                try
                {
                    MySqlCommand command = new MySqlCommand(query,
conn);

                    using (var reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            tableList.Add(reader["url"].ToString());
                        }
                    }
                }
                catch (Exception ex)
                {
                    Log(new StackTrace(ex, true), ex);
                    throw;
                }
            }
            LogInfo("getUrlTelegramGroups", $"Returned
{tableList.Count} urls");
            return tableList;
        }
        public bool updateKeyWordById(int id_, string keyWord)
        {
            lock (_balanceLock)
            {
                string query = "UPDATE KeyWords SET KeyWords = @keyWord
WHERE id = @id";
                using (MySqlCommand command = new MySqlCommand(query,
conn))
                {

```

```

keyWord);
        // Додаємо параметри до запиту
        command.Parameters.AddWithValue("@keyWord",
        command.Parameters.AddWithValue("@id", id_);

        // Виконуємо запит і перевіряємо кількість змінених
        int rowsAffected = command.ExecuteNonQuery();

        return rowsAffected > 0;
    }
}
}
public List<analytic_system.application.DataStucture.keyWords>
showAllKeyWords()
{
    lock (_balanceLock)
    {
        string query = "SELECT * FROM KeyWords";

        List<analytic_system.application.DataStucture.keyWords> keyWordsList = new
        List<analytic_system.application.DataStucture.keyWords>();
        try
        {
            //MessageBox.Show("Line 206");
            MySqlCommand command = new MySqlCommand(query,
            conn);

            using (var reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    analytic_system.application.DataStucture.keyWords row = new
                    analytic_system.application.DataStucture.keyWords
                    {
                        id = Convert.ToInt32(reader["id"]),
                        KeyWords = reader["keyWords"] is DBNull
                        ? string.Empty : reader["keyWords"].ToString()
                    };
                    keyWordsList.Add(row);
                }
            }
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
        LogInfo("showAllKeyWords", $"Returned
        {keyWordsList.Count} words");
        return keyWordsList;
    }
}
public List<TelegramGroup> showAllTelegramGroups() =>
showAllTelegramGroupsFromDataBase();
private List<TelegramGroup> showAllTelegramGroupsFromDataBase()
{
    lock (_balanceLock)
    {
        string query = "SELECT * FROM TelegramGroup";

```

```

List<TelegramGroup>() ;
        List<TelegramGroup> tableList = new
        try
        {
            //MessageBox.Show("Line 206");
            MySqlCommand command = new MySqlCommand(query,
conn);

            using (var reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    TelegramGroup row = new TelegramGroup
                    {
                        id =
Convert.ToUInt32(reader["idTelegram"]),
                        groupName = reader["groupName"] is
DBNull ? string.Empty : reader["groupName"].ToString(),
                        memberCount =
Convert.ToInt32(reader["membersCount"]),
                        dateTime =
Convert.ToDateTime(reader["dateTimeLatestScan"]),
                        groupURL = reader["url"] is DBNull ?
string.Empty : reader["url"].ToString()
                    };
                    tableList.Add(row);
                }
            }
        }
        catch (Exception ex)
        {
            Log(new StackTrace(ex, true), ex);
            throw;
        }
        LogInfo("showAllTelegramGroupsFromDataBase", $"Returned
{tableList.Count} groups");
        return tableList;
    }

    ~MySQLDatabase()
    {
        conn.Close();
    }
}

```

Б2. Текст файла MySQLDatabaseSingelton.cs

```

using analytic_system.Application.DataStucture;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;

```

```

namespace analytic_system.Application.Model
{
    internal class MySQLDatabaseSingleton : IDatabase
    {
        private static MySQLDatabaseSingleton _instance;
        private MySQLDatabase mySQLDatabase;

        public static MySQLDatabaseSingleton GetInstance(string conn)
        {
            if (_instance == null)
            {
                _instance = new MySQLDatabaseSingleton(conn);
            }

            return _instance;
        }

        private MySQLDatabaseSingleton(string conn)
        {
            mySQLDatabase = new MySQLDatabase(conn);
        }

        public void addKeyWord(string word)
        {
            try
            {
                mySQLDatabase.addKeyWord(word);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        public void AddTelegramGroup(string from, long id, string
groupName, int memberCount, DateTime dateTimeLatestScan, string url)
        {
            try
            {
                mySQLDatabase.AddTelegramGroup(from, id, groupName,
memberCount, dateTimeLatestScan, url);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        public void deleteById(long id)
        {
            try
            {
                mySQLDatabase.deleteById(id);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        public void deleteKeyWordById(int id)
        {

```

```

        try
        {
            mySQLDatabase.deleteKeyWordById(id);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }

    public long getIdTelegram(string groupName)
    {
        try
        {
            return mySQLDatabase.getIdTelegram(groupName);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            return 0;
        }
    }

    public List<long> getIdTelegramGroups()
    {
        try
        {
            return mySQLDatabase.getIdTelegramGroups();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            return null;
        }
    }

    public string getUrlById(long id)
    {
        try
        {
            return mySQLDatabase.getUrlById(id);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            return null;
        }
    }

    public List<string> getUrlTelegramGroups()
    {
        try
        {
            return mySQLDatabase.getUrlTelegramGroups();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            return null;
        }
    }

```

```

public bool isExist(long idTelegram)
{
    try
    {
        return mySQLDatabase.isExist(idTelegram);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return false;
    }
}

public List<analytic_system.application.DataStucture.keyWords>
showAllKeyWords()
{
    try
    {
        return mySQLDatabase.showAllKeyWords();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return null;
    }
}

public List<TelegramGroup> showAllTelegramGroups()
{
    try
    {
        return mySQLDatabase.showAllTelegramGroups();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return null;
    }
}

public bool updateKeyWordById(int id, string keyWord)
{
    try
    {
        return mySQLDatabase.updateKeyWordById(id, keyWord);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return false;
    }
}

public void uploadMembersCount(long id, int membersCount)
{
    try
    {
        mySQLDatabase.uploadMembersCount(id, membersCount);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

```
    }
  }
}}
```

Б3. Текст файла BasicFetcher.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using FluentResults;
using TL;
using Optional;
using static System.Net.WebRequestMethods;
using System.Windows;
using System.Collections.ObjectModel;

namespace
analytic_system.application.ViewModel.modules.TG.Parser.ChannelProcessing.Channel
{
    internal class BasicFetcher : IBasicFetcher
    {
        private WTelegram.Client client;

        public async Task leaveChannel(TL.Channel channel)
        {
            await client.Channels_LeaveChannel(channel);
        }
        public async Task<Result> leaveAllGroups()
        {
            try
            {
                var chats = await client.Messages_GetAllChats();
                List<Task> tasks = new List<Task>();
                foreach (var chat in chats.chats.Values)
                {
                    if (chat is TL.Channel)
                    {
                        Console.WriteLine(chat);
tasks.Add(client.Channels_LeaveChannel((TL.Channel) chat));
                    }
                }
                foreach (var task in tasks)
                {
                    await task;
                }
            }
            catch (Exception ex)
            {
                Result.Fail(ex.ToString());
            }
            return Result.Ok();
        }
        public async Task<Result<TL.Channel>> joinPublicChannel(string
channelname)
```

```

    {
        try
        {
            var resolved = await
client.Contacts_ResolveUsername(channelname);
            if (resolved.Chat is TL.Channel channel)
                await client.Channels_JoinChannel(channel);
            else
                return Result.Fail("Channel is not resolved");
            return Result.Ok((TL.Channel)resolved.Chat);
        }
        catch (Exception e)
        {
            return Result.Fail(e.ToString());
        }
    }
    public async Task<Result> askToJoinPrivateChannel(string hash)
    {
        try
        {
            var chats = await
client.Messages_ImportChatInvite(hash);
            return Result.Ok();
        }
        catch (Exception e)
        {
            return Result.Fail(e.ToString());
        }
    }
    public async Task<Result<ChannelFull>>
getFullChannelInfo(TL.Channel channelBase)
    {
        try
        {
            if (client == null)
            {
                MessageBox.Show("Client is not initialized");
                return Result.Fail("Client is not initialized");
            }

            if (channelBase == null)
            {
                MessageBox.Show("Channel base is null");
                return Result.Fail("Channel base is null");
            }

            try
            {
                var channelFullInfo = await
client.Channels_GetFullChannel(channelBase);
                if (channelFullInfo == null)
                {
                    MessageBox.Show("Failed to get full channel
info");
                    return Result.Fail("Failed to get full channel
info");
                }

                if (channelFullInfo.full_chat == null)
                {
                    MessageBox.Show("Full chat data is null");
                    return Result.Fail("Full chat data is null");
                }
            }
        }
    }

```

```

        }
        return
Result.Ok((ChannelFull)channelFullInfo.full_chat);
    }
    catch (Exception exception)
    {
        MessageBox.Show($"Exception occurred:
{exception.Message}");
        return Result.Fail(exception.ToString());
    }
}
catch (Exception exception)
{
    MessageBox.Show($"Exception occurred:
{exception.Message}");
    return Result.Fail(exception.ToString());
}
}
public async Task<Result<Dictionary<long, TL.Channel>>>
getAllChannels()
{
    try
    {
        var allChats = await client.Messages_GetAllChats();
        Dictionary<long, TL.Channel> channels = new
Dictionary<long, TL.Channel>();
        foreach (var chat in allChats.chats)
        {
            if (chat.Value is TL.Channel)
            {
                channels.Add(chat.Key, (TL.Channel)chat.Value);
            }
        }
        return Result.Ok(channels);
    }
    catch (Exception exception)
    {
        return Result.Fail(exception.ToString());
    }
}
public BasicFetcher(WTelegram.Client client)
{
    this.client = client;
}

public async Task<Result<List<MessageBase>>>
getNLastMessages(TL.Channel channel, int n)
{
    return await getNMessagesAfterID(channel, n, 0);
}
public async Task<Result<List<MessageBase>>>
getNMessagesAfterID(TL.Channel channel, int n, int afterId)
{
    try
    {
        List<MessageBase> messages = new List<MessageBase>();
        int lastID = afterId;
        while (n > 0)
        {
            int oldSize = messages.Count;
            messages.AddRange((await
client.Messages_GetHistory(channel, lastID, default, default, n)).Messages);

```

```

        int wereAdded = messages.Count - oldSize;
        if (wereAdded == 0)
            break;
        n -= wereAdded;
        lastID = messages.Last().ID;
    }
    return Result.Ok(messages);
}
catch (Exception exception)
{
    return Result.Fail(exception.ToString());
}
}
private Option<int> idOfMessageBeforeGivenDate(MessageBase[]
messages, DateTime theAfterDate)
{
    for (int i = 0; i < messages.Length; ++i)
    {
        if (messages[i].Date.CompareTo(theAfterDate) < 0)
        {
            return Option.Some(i);
        }
    }
    return Option.None<int>();
}
/*
public async Task<Result<List<MessageBase>>>
getNLastMessagesAfterTheDate(TL.Channel channel, DateTime theAfterDate, int n)
{
    try
    {
        n = int.MaxValue;

        int lastID = 0;
        List<MessageBase> messages = new
List<MessageBase>();

        while (n > 0)
        {
            int oldSize = messages.Count;
            var messagesArray = (await
client.Messages_GetHistory(channel, lastID, default, default, n)).Messages;
            messages.AddRange(messagesArray);
            //TODO: test if binary search would be
benefitful (probably not)
            var idOfMessageOlderThanTheGivenDate =
idOfMessageBeforeGivenDate(messagesArray, theAfterDate);
            if
(idOfMessageOlderThanTheGivenDate.HasValue)
            {
                for (int i = 0; i <
idOfMessageOlderThanTheGivenDate.ValueOr(messagesArray.Length); ++i)
                {
                    messages.Add(messagesArray[i]);
                }
                return Result.Ok(messages);
            }
            messages.AddRange(messagesArray);
            int wereAdded = messages.Count - oldSize;
            if (wereAdded == 0)
                break;
            n -= wereAdded;
            lastID = messages.Last().ID;
        }
    }
}

```

```

        return Result.Ok(messages);

    }
    catch (Exception exception)
    {
        return Result.Fail(exception.ToString());
    }
}*/

public async Task<Result<List<MessageBase>>>
getNLastMessagesAfterTheDate(TL.Channel channel, DateTime targetDate, int n)
{
    try
    {
        int lastID = 0;
        List<MessageBase> result = new List<MessageBase>();

        while (true)
        {
            var response = await
client.Messages_GetHistory(channel, lastID, default, default, 100);
            var messagesArray = response.Messages;

            if (messagesArray.Length == 0)
                break;

            foreach (var msgBase in messagesArray)
            {
                if (msgBase is TL.Message msg &&
msg.date.ToLocalTime().Date == targetDate.Date)
                {
                    //MessageBox.Show(msgBase.ToString());
                    result.Add(msgBase);
                }
                else if (msgBase is TL.MessageService ms &&
ms.date.ToLocalTime().Date == targetDate.Date)
                {
                    MessageBox.Show(msgBase.Date.ToString());
                    result.Add(msgBase);
                }
                else if (msgBase.Date.ToLocalTime() ==
targetDate.Date)
                {
                    MessageBox.Show(msgBase.Date.ToString());
                }
            }

            // Якщо всі повідомлення старіші за targetDate,
            завершуй
            if (messagesArray
                .OfType<TL.Message>()
                .All(m => m.date.Date < targetDate.Date))
                break;

            lastID =
messagesArray.OfType<TL.Message>().Last().ID;
        }
    }
}

```

```

        return Result.Ok(result);
    }
    catch (Exception ex)
    {
        return Result.Fail(ex.ToString());
    }
}

public async Task<Result<List<MessageBase>>>
getNLastMessagesAfterTheDatePlusEditGuaranty(TL.Channel channel, DateTime
theAfterDate, int n = int.MaxValue)
{
    return await getNLastMessagesAfterTheDate(channel,
theAfterDate.AddDays(-8), n);
}
}

```

ДОДАТОК В
Слайди презентації

Национальний університет "Запорізька політехніка"
Кафедра програмних засобів

Розробка аналітичної системи аналізу та підтримки прийняття рішень Єдиної регіональної інформаційної системи захисту критичної інфраструктури

Виконала:
ст. групи КНТ - 121
Керівник:
к. т. н., професор

ЩЕРБИНА О. О.

ДУБРОВІН В. І.

Рисунок В.1 – Слайд №1

Мета, предмет, об'єкт дослідження

Мета

- розробити застосунок, що здійснює рекурсивний пошук тематичних груп у Telegram для подальшого прогнозування потенційно небезпечних дій.

Об'єкт

- процес аналізу інформації, отриманої з відкритих джерел, пов'язаної з переміщенням БПЛА

Предмет дослідження

- методи захисту критичної інфраструктури

Рисунок В.2 – Слайд №2

Огляд аналогів

Можливості	IBM resilient SOAR	Avada-media парсер-телеграм	Open Cyber Threat Intelligence Platform	Власний застосунок
Керування інформацією/ загрозами в реальному часі	+	+	+	+
Інтеграція з системами безпеки	+	-	+	-
Керування групами Telegram	-	+	-	+
Вартість	-	-	-	+

Рисунок В.3 – Слайд №3

Вибір мови програмування

Можливості	C#	Java	PHP
Платформи	.NET, windows	Багатолатф.	Web
Типизація	Статична	Статична	Динамічна
Продуктивність	Висока	Висока	Середня
Підтримка асинхронності	+	+	+
Інтеграція з github	+	+	+
ООП	+	+	+

Можливості	C#	Java	PHP
Розробка під ios	+	+	-
Розробка під Android	+	+	-
Розробка desktop	+	+	-
Кросплатформена розробка	+	+	-
Витрати ресурсів	+	+	-
Розробка під ios	+	+	-

Рисунок В.4 – Слайд №4

Вибір середовища розробки

Критерій	Visual Studio	Visual Studio Code	Rider
Підтримка C#	+	+	+
Підтримка WPF	+	-	+
WPF Designer/XAML Designer	+	-	+-
Швидкість	-	+	+
Ціна	+	+	-

Рисунок В.5 – Слайд №5

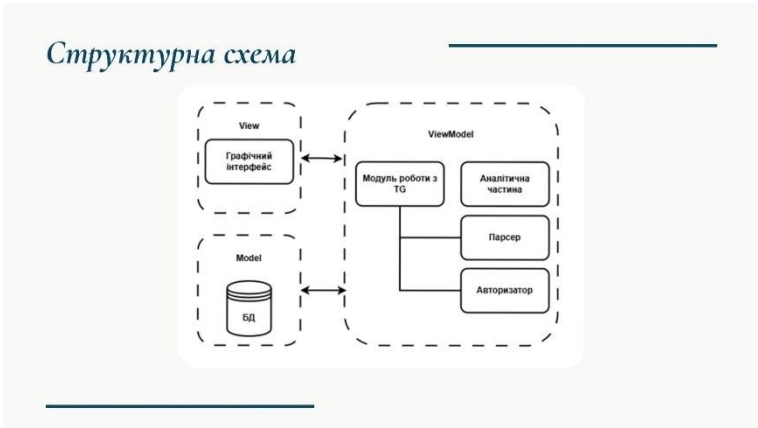


Рисунок В.6 – Слайд №6

Use-case діаграма

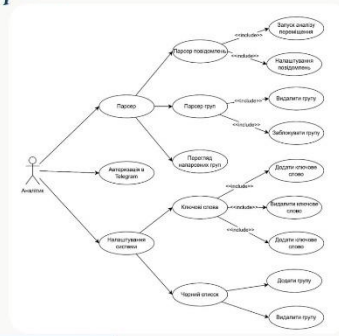


Рисунок В.7 – Слайд №7

Package-object діаграма

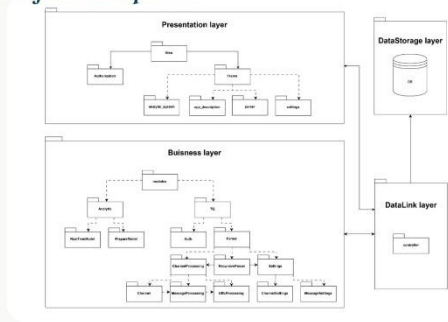


Рисунок В.8 – Слайд №8

ER-діаграма

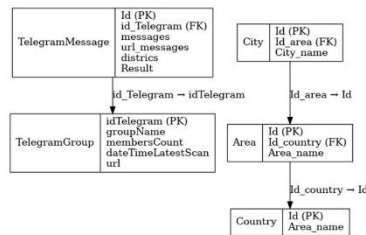


Рисунок В.9 – Слайд №9



Рисунок В.10 – Слайд №10

Висновки

Парсер:

- Розроблено функціонал, який дозволяє автоматично вступати в тематичні групи та керувати ними

Обробка повідомлень:

- Розроблений функціонал дозволяє збирати повідомлення за певними ключовими словами та датою, після чого, створювати правила за якими буде здійснюватися прогнозування

Подальший розвиток

- В майбутньому, до застосунку планується ядати модуль з ШІ, завдяки якому буде здійснюватися виявлення пересування БПЛА та прогноз щодо можливих подальших дій прогнозування війни

Рисунок В.11 – Слайд №11

Наукові публікації

2023 Тиждень наук

- Огляд методів боротьби з SQL-ін'єкціями
- ст. викладач Дейнега Л. Ю., Щербина О. О.

2024 Тиждень наук

- Дослідження евристичного аналізу антивірусів та методи їх покращення
- ст. викладач Качан О. І., Щербина О. О.

2025 Тиждень наук

- Огляд методів захисту критичної інфраструктури
- к.т.н., професор Дубровін, В.І., Щербина О. О.

Рисунок В.12 – Слайд №12

Досягнення у творчих конкурсах

- 2 місце в університетському конкурсі студентських наукових робіт 2024/2025 н.р.;
- фіналістка BootCamp 2024 від Kharkiv IT Cluster;
- учасниця менторської програми від Vona Tech Community (2024);
- фіналістка конкурсу бізнес-ідей «STARTUP із Запорізькою політехнікою – 2024»;
- - перемогла в спеціальній номінації від АТ «Мотор Січ»;
- - перемогла в спеціальній номінації від «Креді Агріколь Банк»;
- 3 місце в Запорізькій області в олімпіаді ICPC-Ukraine 2023;
- 3 місце в Запорізькій області в олімпіаді ICPC-Ukraine 2022;
- 1 місце в ОНЛАЙН-ХАКАТОНІ «НАСК4GOOD: ЯК ЗАЦІКАВИТИ ДІВЧАТ ТЕХНОЛОГІЯМИ» та активну участь у майстер-класах з гендерної рівності, дизайн-мислення, мистецтва презентацій та пітчингу, кар'єрного розвитку (2021).

Рисунок В.13 – Слайд №13