

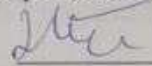
Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: **«Проект підсистеми виявлення атак
соціальної інженерії в електронних повідомленнях»**

Виконав: студент 4 курсу групи КБп-21-1

спеціальності 125 «Кібербезпека»

 Теймур АХМЕДОВ

Керівник: Доцент кафедри КІ та КБ

 Юрій БРОДСЬКИЙ

Рецензент: Доцент кафедри КІ та КБ

 Олександр МАЄВСЬКИЙ

Житомир – 2025

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Сфіменко

1 серпня 2025 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Ахмедов Теймур Беюкага огли

Тема роботи: «Проект підсистеми виявлення атак соціальної інженерії в електронних повідомленнях»

затверджена Наказом університету від 1 серпня 2025 р. № (269/с)

Термін здачі студентом закінченої роботи 29 серпня 2025 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

- Методи емоційного аналізу тексту для виявлення соціальної інженерії в електронних повідомленнях.
- Електронні повідомлення, що містять потенційно шкідливий або нейтральний вміст, з метою автоматизованого аналізу.
- Програмне забезпечення, розроблене мовою Python, що здійснює аналіз текстів електронних листів за допомогою емоційного аналізу та класифікації через API Microsoft Azure.
- Графічний інтерфейс користувача, реалізований на WPF (.NET/C#) для Windows, який дублює та спрощує використання функціональності Python-рішення.

Обладнання та програмне забезпечення:

1. Операційна система: Windows або сумісна Unix-система (Linux).

2. Середовище та мова розробки: Python 3.10+ / PyCharm. Для WPF-клієнта: .NET 6/7 SDK, Visual Studio 2022

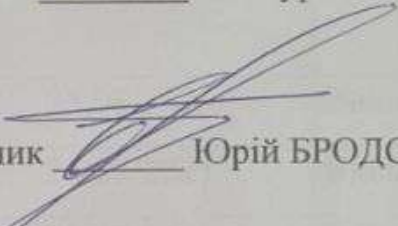
3. Використані бібліотеки: requests, matplotlib, pandas, deep-translator, azure-ai-textanalytics. Для UI (WPF): CommunityToolkit.Mvvm, Newtonsoft.Json/System.Text.Json, виклики HttpClient до Azure

4. Хмарний сервіс: Microsoft Azure Cognitive Services – Language/Text Analytics API

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Пірог О.В.	01.08.2025	01.08.2025

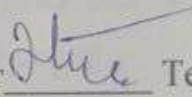
Студент  Теймур АХМЕДОВ

Керівник  Юрій БРОДСЬКИЙ

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Вибір теми, постановка мети і завдань	01.08.2025	Виконано
2	Пошук та аналіз літератури з соціальної інженерії та емоційного аналізу	03.08.2025	Виконано
3	Вивчення API Azure Language та інших інструментів NLP	07.08.2025	Виконано
4	Формування та маркування навчального корпусу листів	09.08.2025	Виконано

5	Розробка програми емоційного аналізу та класифікації листів	11.08.2025	Виконано
6	Експериментальне дослідження та тестування	15.08.2025	Виконано
7	Створення графічного інтерфейса користувача	18.08.2025	Виконано
8	Написання дипломної роботи, включаючи вступ, теоретичну частину, розділ з результатами дослідження, висновки та список використаної літератури	20.08.2025	Виконано
9	Підготовка презентації дослідження для захисту дипломної роботи перед комісією	23.08.2025	Виконано
10	Отримання відгуку керівника	25.08.2025	Виконано
11	Допуск завідувача кафедри	26.08.2025	Виконано
12	Подання роботи на рецензування	27.08.2025	Виконано
13	Захист	29.08.2025	Виконано

Студент  Теймур АХМЕДОВ

Керівник  Юрій БРОДСЬКИЙ

АНОТАЦІЯ

Кваліфікаційна робота бакалавра присвячена проекту підсистеми виявлення атак соціальної інженерії в електронних повідомленнях. У межах роботи розроблено повноцінну програму на мові Python та графічний інтерфейс, які інтегруються з сервісом Azure Language API, що підтримує українську мову.

Пояснювальна записка містить 58 сторінок, включає 12 рисунків, 6 таблиць. Список використаних джерел налічує 17 найменувань і розміщений на 2 сторінках.

Метою кваліфікаційної роботи є розробка ефективної інтелектуальної підсистеми, що виявляє шкідливі електронні повідомлення на основі емоційного аналізу їх текстового змісту.

У ході роботи:

- визначено основні загрози, що пов'язані з соціальною інженерією в електронному листуванні;
- сформовано репрезентативну базу українськомовних листів;
- інтегровано хмарну модель емоційного аналізу Microsoft Azure Text Analytics API;
- реалізовано підсистему для обчислення ризику на основі емоційних коефіцієнтів, реалізовано графічний інтерфейс користувача;
- проведено дослідження залежності метрик від порогового значення класифікації та побудовано відповідні графіки;

Робота демонструє повний цикл створення, налаштування та тестування підсистеми виявлення соціальної інженерії, що може застосовуватись для підвищення кібербезпеки електронного листування.

Ключові слова: СОЦІАЛЬНА ІНЖЕНЕРІЯ, ЕМОЦІЙНИЙ АНАЛІЗ, ЕЛЕКТРОННИЙ ЛИСТ, AZURE TEXT ANALYTICS.

					ККІК<.КР.Б – 125 – 2 – ПЗ			
Змін.	Арк.	№ докум.	Підпис	Дата	Прост підсистеми виявлення атак соціальної інженерії в електронних повідомленнях	Літ.	Арк.	Аркуші
Розроб.		Ахмедов Т.Б.огли		22.08.25			5	65
Керівник		Бродський Ю.Б.		22.08.25				
Реценз.		Масвський О.В.		22.08.25				
Н. Контр.		Бродський Ю.Б.		22.08.25				
Затверд.		Єфименко А.А.		22.08.25				ФІКТ, гр. КБп-21-1

ANNOTATION

The bachelor's qualification thesis is devoted to the development of a system for detecting social engineering attacks in emails. As part of the work, a fully functional Python-based application was created, integrated with the Azure Language API, which supports the Ukrainian language.

The explanatory note consists of 58 pages, includes 12 figures and 7 tables. The list of references contains 16 sources and is placed on 2 pages.

The aim of the thesis is to develop an effective intelligent system capable of detecting malicious emails based on sentiment analysis of their textual content.

During the course of the work:

- identified the main threats associated with social engineering in email communication;
- formed a representative dataset of Ukrainian-language emails;
- integrated the cloud-based sentiment analysis model from Microsoft Azure Text Analytics API;
- implemented an emotion-coefficient-based risk-calculation subsystem and a graphical user interface (GUI);
- conducted an experimental study with variable thresholds and visualized key classification metrics through graphs;
- implemented a graphical user interface that mirrors and streamlines the use of the software module.

The work demonstrates the full cycle of designing, configuring, and testing a system for detecting social engineering attacks, which can be applied to improve the cybersecurity of email communication.

Keywords: SOCIAL ENGINEERING, SENTIMENT ANALYSIS, EMAIL, AZURE TEXT ANALYTICS.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	8
ВСТУП.....	9
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ТА ВИБІР ЗАСОБІВ ДЛЯ РОЗРОБКИ ПІДСИСТЕМИ	11
1.1 Аналіз сучасного стану та обґрунтування актуальності теми роботи	11
1.2 Існуючі підсистеми кібербезпеки для виявлення соціальної інженерії в електронних повідомленнях	13
1.3 Послідовність роботи та функціональні етапи підсистеми	18
1.4 Інструментальні засоби реалізації та вимоги до апаратних засобів	22
Висновки до розділу 1	26
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ВИЯВЛЕННЯ СОЦІАЛЬНОЇ ІНЖЕНЕРІЇ В ЕЛЕКТРОННИХ ЛИСТАХ	27
2.1 Загальна структура програмної підсистеми	27
2.2 Формування та маркування навчального корпусу електронних повідомлень	28
2.3 Вибір та налаштування моделі емоційного аналізу	32
2.4 Реалізація алгоритму емоційного аналізу та класифікації електронних повідомлень ..	33
2.5 Можливі сценарії застосування підсистеми в реальних умовах	41
Висновки до розділу 2	44
РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ.....	45
3.1 Інтерфейс реалізованої підсистеми	45
3.2 Аналіз ефективності програми	48
3.3 Графічний інтерфейс	53
Висновки до розділу 3	55
ВИСНОВКИ.....	56
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	57
ДОДАТКИ.....	59

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AI – Artificial Intelligence

API – Application Programming Interface

CERT-UA – Computer Emergency Response Team of Ukraine

DB – Database

EML – Electronic Mail File

ML – Machine Learning

NLP – Natural Language Processing

PII – Personally Identifiable Information

REST – Representational State Transfer

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасному цифровому середовищі соціальна інженерія стала одним із найнебезпечніших інструментів кіберзлочинців. Багато атак починаються саме з електронного листа, в якому шахрай намагається викликати у користувача сильні емоції — страх, терміновість, жадібність — щоб спонукати до імпульсивних і небезпечних дій.

Стандартні спам і фішингові фільтри переважно орієнтуються на технічні ознаки атаки (посилання, вкладення, домен). Проте в багатьох випадках ці ознаки відсутні або замасковані. У таких ситуаціях емоційне забарвлення повідомлення може бути критично важливою додатковою ознакою, яка допомагає виявити потенційно небезпечні електронні листи, навіть якщо вони технічно виглядають легітимними.

У кваліфікаційній роботі розглядається розробка підсистеми виявлення соціальної інженерії в електронних листах на основі емоційного аналізу тексту. Ідея полягає в тому, щоб автоматично визначати емоційну тональність електронного повідомлення (негативну, нейтральну або позитивну) з використанням хмарних сервісів Azure Language API, а також враховувати ознаки маніпуляцій - залякування, терміновості, обіцянки вигоди і тому подібне. Це дозволяє виявляти шкідливі електронні повідомлення навіть за відсутності технічних індикаторів.

Розроблена підсистема виконує послідовну обробку електронних листів: аналізує текстовий вміст, визначає емоційне забарвлення, обчислює ризиковий бал і класифікує повідомлення. Проведено тестування на контрольному корпусі, а також досліджено можливі сценарії застосування підсистеми. Створений графічний інтерфейс спрощує використання підсистеми.

Актуальність запропонованої теми полягає в потребі посилення виявлення соціально-інженерних атак через аналіз емоційного контексту листів, зокрема — українською мовою, що досі недостатньо підтримується в сучасних підсистемах захисту.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

Метою кваліфікаційної роботи є розробка підсистеми виявлення електронних повідомлень із ознаками соціальної інженерії шляхом застосування емоційного аналізу текстового вмісту повідомлень.

Для досягнення мети було поставлено наступні завдання:

- здійснити аналіз поточного стану проблеми фішингових атак та методів соціальної інженерії;
- дослідити сучасні інструменти емоційного аналізу текстів;
- сформувати навчальний корпус електронних листів українською мовою;
- реалізувати власну програму для визначення ризику шкідливості листа на основі результатів емоційного аналізу;
- протестувати підсистему на реальній вибірці листів та оцінити її ефективність за метриками класифікації;
- розробити та впровадити графічний інтерфейс користувача, який дублює та спрощує використання підпрограми на основі тестування підсистеми.

Об'єктом дослідження є процес автоматизованого виявлення електронних повідомлень із ознаками соціальної інженерії на основі контентного аналізу тексту. Дослідження виконується в парадигмі бінарної класифікації («шкідливий»/«нейтральний») без визначення підтипів атак.

Предметом дослідження є моделі та алгоритми контентного виявлення соціально-інженерних електронних листів на основі емоційних ознак тексту: формування ознак, обчислення показнику ризиковості, вибір порогового значення та оцінювання якості бінарної класифікації.

У процесі виконання кваліфікаційної роботи було розроблено програмні рішення, що взаємодіють з інтелектуальним хмарним сервісом для аналізу емоційного забарвлення електронних повідомлень та виявлення ознак соціальної інженерії.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ТА ВИБІР ЗАСОБІВ ДЛЯ РОЗРОБКИ ПІДСИСТЕМИ

1.1 Аналіз сучасного стану та обґрунтування актуальності теми роботи

Соціальна інженерія в кібербезпеці – це психологічне маніпулювання користувачами з метою примусити їх виконати дії, вигідні зловмиснику (розкрити конфіденційні дані, завантажити шкідливий файл тощо). Найпоширенішим різновидом таких атак є фішинг – шахрайські електронні електронні повідомлення, які маскуються під легітимні повідомлення від відомих організацій або осіб [1]. Зловмисники видають себе за довірених відправників і спонукають жертв перейти за фальшивим посиланням чи надати чутливу інформацію. Якщо атака успішна, наслідками можуть стати витік даних, фінансові збитки, зараження системи шкідливим ПЗ або навіть захоплення облікових записів. За даними CISA (Агенції з кібербезпеки США), 91% усіх кібератак починаються саме з фішингового електронного листа. Фішинг уже кілька років поспіль очолює статистику кібершахрайств: наприклад, у 2022 році на фішинг припало понад 300 тисяч скарг (включно з email-, смішинг- та вішіинг-атаками) – більше, ніж на будь-який інший тип кіберзлочинів [7].

Обсяги фішингових розсилок досягли рекордних рівнів і продовжують зростати. Станом на 2023 рік щоденно у світі розсилається приблизно 3,4 мільярда фішингових електронних листів (понад 1 трлн на рік), що робить фішинг одним із наймасовіших кіберзагроз. 2023 рік став найгіршим за кількістю зафіксованих фішингових атак – близько 5 мільйонів випадків (дещо більше, ніж 4,74 млн у 2022 році). За статистикою Kaspersky Lab, лише їхні антифішингові системи заблокували 709,6 млн фішинг-спроб у 2023 році – на 40% більше, ніж торік. Фішингові кампанії часто мають сплески активності: один із провайдерів безпеки зафіксував стрибок кількості фішинг-листів на 173% у третьому кварталі 2023 порівняно з другим кварталом. Близько 96% фішингових атак доставляються саме через email (електронну пошту), попри зростання інших каналів на кшталт SMS чи телефонних дзвінків. Фактично, електронна пошта

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

залишається домінуючим вектором загроз, і майже кожна організація регулярно стикається з фішинговими листами: 57% компаній повідомляють про такі атаки щонайменше щотижня. Розмах проблеми підкреслює і те, що 94% організацій у 2023 році стали жертвами фішингу, причому 96% із них зазнали негативних наслідків (фінансових втрат, втрати даних тощо) [1].

Актуальність теми зумовлена не тільки масштабами фішингових атак, а й їхньою ефективністю, що базується на психологічному впливі. Людський фактор традиційно є «найслабшою ланкою» кібербезпеки. Атаки соціальної інженерії майстерно експлуатують емоції користувачів — страх, довіру, цікавість, жадібність, співчуття — щоб спонукати їх діяти імпульсивно. Наприклад, злоумисники часто навмисно створюють відчуття терміновості або паніки в листі чи погрожують негативними наслідками (штрафи, судові позови), щоб жертва, налякана або стривожена, не встигла критично обмірковувати ситуацію. Такі емоційні тригери значно підвищують результативність фішингу. Дослідження показують, що найпоширеніші емоції, на яких грають шахраї, – це страх (залякати користувача повідомленням про проблеми з безпекою або борги), жадоба вигоди (обіцянка великого виграшу чи вигоди), терміновість (тиск часових обмежень) та авторитет/довіра (вдавання посадових осіб або відомих брендів) [2]. Наприклад, у 2025 році зафіксовано SMS-фішинг кампанію, що обдурила тисячі людей фальшивими повідомленнями про сплату мита за посилку – жертви, боячись штрафів чи прагнучи отримати відправлення, переходили за шкідливими посиланнями, розкриваючи фінансові дані. ФБР повідомило, що кількість подібних схем зросла в 4 рази за рік. Таким чином, соціальна інженерія в електронних листах є вкрай небезпечною: вона апелює до базових людських емоцій, змушуючи людей діяти швидко і не задумуючись [11].

Попри розвиток технічних засобів захисту (спам-фільтри, шлюзи безпеки, антивіруси), фішингові атаки продовжують успішно вражати користувачів. Статистика демонструє, що значна частка шкідливих листів таки прослизає повз автоматичні підсистеми фільтрації. Згідно з оцінкою CISA, 70% небезпечних вкладень або посилань у листах не були заблоковані мережевими засобами

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

захисту на периметрі, а 15% – пройшли навіть захист кінцевих точок (антивіруси на комп'ютерах). Ба більше, якщо користувач таки отримав фішингового листа, він, найімовірніше, швидко піддається на приманку: 84% працівників, які стали жертвами умовної фішингової розсилки, зробили це впродовж перших 10 хвилин після отримання листа [9]. Така ефективність атак вимагає нових підходів до їх виявлення. Традиційні методи (чорні списки URL-адрес, сигнатури відомих загроз, перевірка заголовків) корисні, але не завжди встигають за появою нових схем соціальної інженерії.

Актуальним є дослідження контенту листів — аналіз тональності та емоційного забарвлення тексту — щоб розпізнати приховані спроби маніпуляції. Ідея полягає в тому, щоб «навчити» систему розуміти, коли в повідомленні одержувача навмисно викликають страх чи азарт, тиснуть на довіру чи авторитет — ознаки, притаманні шахрайству. Такий емоційний аналіз може стати додатковим шаром фільтрації, що підвищить виявлення соціально-інженерних атак. Наприклад, фішингові електронні повідомлення часто містять саме психологічні прийоми: створення терміновості, неправдиві обіцянки вигоди, залякування користувача. Розпізнавши подібні аномально емоційні повідомлення, підсистема могла б помітити лист як підозрілий навіть за відсутності явних технічних ознак атаки. Таким чином, дослідження, присвячене виявленню соціальної інженерії в email шляхом емоційного аналізу тексту, є своєчасним та актуальним. Воно спрямоване на підвищення ефективності захисту від фішингу, доповнюючи існуючі технічні засоби розумінням психологічних аспектів атак [2].

1.2 Існуючі підсистеми кібербезпеки для виявлення соціальної інженерії в електронних повідомленнях

Розглянемо сучасні рішення, що протидіють фішинговим електронним листам, а також проекти, які застосовують аналіз тональності чи емоцій для виявлення шахрайства. Серед широко використовуваних засобів – вбудовані системи фільтрації пошти (наприклад, Gmail та Outlook), а також спеціалізовані

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

рішення класу Secure Email Gateway (такі як Proofpoint чи Barracuda Email Protection) (див.табл.1.2.1). Окремо зазначимо дослідницькі розробки, що використовують аналіз тексту листів.

Найпоширенішими є вбудовані антифішингові системи популярних поштових служб — Google Gmail та Microsoft Outlook/Office 365. Вони автоматично перевіряють усі вхідні електронні повідомлення та мають багаторівневий підхід до фільтрації [13].

Gmail використовує масштабні моделі машинного навчання, натреновані на величезних масивах даних про спам і фішинг. Google повідомляє, що штучний інтелект Gmail блокує щоденно понад 100 мільйонів фішингових листів і оперативно виявляє нові техніки атак в реальному часі. Для аналізу змісту листів застосовується NLP (обробка природної мови): система розпізнає підозрілі шаблони тексту, наприклад надто ургентні формулювання («Ваш аккаунт зламано! негайно змініть пароль»), характерні орфографічні помилки шахраїв, нетипові звертання тощо. Також Gmail перевіряє посилання та вкладення: URL-адреси звіряються з чорними списками Google Safe Browsing, аналізується перенаправлення за ними, а вкладені файли скануються на наявність відомих шкідливих програм [9].

Outlook / Office 365 має схожий підхід. Хмарний захист Microsoft Defender for Office 365 відстежує фішингові індикатори у вхідних повідомленнях. ШІ-алгоритми Microsoft перевіряють кожен лист на фішингові ознаки, небезпечні вкладення та посилання. В разі виявлення загрози система автоматично *видаляє* підозрілий email ще до його потрапляння в папку «Вхідні» користувача. Microsoft використовує технології Safe Links (переписування URL-лінків в електронних листах на безпечні проксі-посилання, які перевіряються при кліку) та Safe Attachments (відкриття вкладень у захищеній ізольованій «пісочниці» для аналізу поведінки). Обидві платформи (Gmail і Outlook) постійно оновлюють свої моделі на основі глобальної телеметрії загроз. Завдяки масштабам даних та проактивному ШІ, ці системи здатні відфільтровувати переважну більшість фішингових спроб (заявляється ефективність понад 99% у виявленні

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

спаму/фішингу). Втім, найвитонченіші або цілеспрямовані атаки (сpear-фішинг, BEC) все ж можуть оминати фільтри, що стимулює розвиток додаткових методів виявлення [3].

Для корпоративного сектору широко застосовуються окремі платформи кібербезпеки, що забезпечують розширений захист пошти від фішингу, зокрема продукти компаній Proofpoint та Barracuda Networks. Proofpoint Email Protection — лідер серед захисних рішень для великих організацій (встановлений у більшості компаній зі списку Fortune 1000). Він реалізує багатошаровий підхід: поєднує передові алгоритми машинного навчання (NexusAI), що аналізують вміст листів, вкладення і URL-адреси, з масштабною базою глобальної загрозової інформації (Nexus Threat Graph) та індикаторами атак. Proofpoint щоденно перевіряє мільярди URL-посилань, виявляючи й блокуючи шкідливі.

Система ефективно розпізнає підроблені адреси відправників, спроби спуфінгу доменів, підозрілі вкладення (використовуючи динамічну «пісочницю» для їх виконання) тощо. Окрім автоматичного фільтру, важливим компонентом є навчання користувачів: Proofpoint пропонує модуль Security Awareness Training, що знижує ризик успішного фішингу до 90% шляхом тренувальних фішинг-розсилок і освітніх програм. Також впроваджені інструменти для інцидент-реагування — наприклад, кнопка «Report Phish» для співробітників та автоматизована система Closed-Loop Email Analysis and Response (CLEAR), яка видаляє з поштових скриньок раніше доставлені шкідливі електронні повідомлення по мірі виявлення загрози [11].

Barracuda Email Protection – інший популярний шлюз безпеки пошти, який забезпечує хмарну фільтрацію. Підхід Barracuda включає комбінацію декількох методів: по-перше, Anti-Fraud Intelligence — спеціалізована база даних (байєсівський фільтр), натренована для розпізнавання шаблонів фішингових шахрайств. По-друге, поведінковий та евристичний аналіз: система аналізує аномалії у поведінці відправників і змісті листів, виявляючи підозрілі ознаки (нетиповий стиль листа від знайомого адресата, невідповідність між полем «Від кого» та реальним доменом тощо). По-третє, захист від спуфінгу і перевірка

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

легітимності доменів (використання SPF, DKIM, DMARC), а також виявлення тайпсквотингу — навмисних дрібних помилок у назвах доменів, схожих на справжні (наприклад, paupa1.com замість paupal.com), що автоматично перенаправляються в пісочницю для аналізу. Barracuda також виконує поглиблене сканування вкладень і посилань: підозрілі файли ізолюються та «вибухають» в sandbox-середовищі (Advanced Threat Protection), а всі URL в листах переписуються на проміжні, щоб при кліку спочатку перевірити їх на сервері безпеки (Link Protection). Такий підхід дозволяє блокувати як масові фішингові розсилки, так і більш таргетовані атаки (спуфінг керівників, розсилки від імені партнерів тощо). В цілому, спеціалізовані рішення на кшталт Proofpoint і Barracuda надають розширені функції захисту, перевершуючи базові фільтри поштових сервісів, проте потребують окремого впровадження та адміністрування. У таблиці 1 наведено порівняння основних характеристик розглянутих систем [5].

Окрім комерційних продуктів, ведуться наукові дослідження та експериментальні розробки, що прагнуть підвищити ефективність виявлення фішингу за рахунок аналізу текстового змісту повідомлень — зокрема, їх тональності, емоційного забарвлення та принципів переконання. Напрямок доволі новий, але набирає обертів. Так, у 2024 р. дослідники Bustio-Martínez та ін. запропонували методику виявлення фішингу шляхом аналізу принципів переконання у тексті листа — наприклад, пошук ознак надмірної суб’єктивності, апеляції до авторитету чи терміновості, що не характерні для легітимної ділової переписки [7]. Інша робота (Duan et al., 2025) продемонструвала поєднання великих мовних моделей (LLM) із sentiment analysis для розпізнавання фішингових email. Дослідники керували LLM через спеціально сформовані промпти, щоб модель оцінювала вміст листа і шукала емоційні маркери соціальної інженерії. Експерименти показали, що такий підхід покращує точність виявлення фішингових повідомлень [8]. Зазначається, що фішингові електронні повідомлення часто мають негативну або невідношену тональність (панічні повідомлення, агресивні вимоги) або ж неприродно

позитивну (обіцянки виграшу, винагороди) — і машинний аналіз настрою здатен це виловити.

Також відзначимо пов’язані дослідження в сфері vishing (голосового фішингу). Наприклад, проєкт Emoti-Shing (2024) використовує моделі Hidden Markov для аналізу емоційних змін у голосі жертви під час телефонної розмови, щоб виявити шахрайство по телефону [8]. Хоча голосові атаки – окрема тема, ця робота підтверджує ефективність відстеження емоційної динаміки для розпізнавання соціальної інженерії. Загалом, включення аналізу тональності тексту в системи кібербезпеки поки не є масовою комерційною практикою, проте перспективність напрямку доведена дослідженнями. У проєкті пропонується розвинути ці ідеї, а саме інтегрувати емоційний аналіз тексту як одну зі складових системи фільтрації фішингових листів.

Таблиця 1.2.1

Порівняння підходів до виявлення фішингових електронних листів у різних системах.

Система / сервіс	Тип рішення	Основні методи виявлення фішингу	Додаткові можливості
Google Gmail	Вбудований фільтр пошти. (хмарний сервіс)	Машинне навчання на великій вибірці даних; аналіз тексту листа (NLP) на наявність шахрайських шаблонів; перевірка URL (Safe Browsing) та вкладень на відомі загрози	Блокує ~100 млн шкідливих листів щодня; автоматично виносить підозрілі листи в Spam; попереджає користувача про потенційно небезпечні листи або файли.[3]
Microsoft Outlook / Office 365	Вбудований фільтр пошти. (хмарний сервіс)	ШІ-алгоритми (Microsoft Defender) для сканування вмісту, посилань (Safe Links) і вкладень (Safe Attachments); правила відповідності (політики безпеки)	Автоматичне видалення чи карантин листів з фішинг-ознаками; інтеграція з екосистемою Microsoft 365;

Продовження табл. 1.2.1

Система / сервіс	Тип рішення	Основні методи виявлення фішингу	Додаткові можливості
			захист від спуфінгу доменів (DMARC). [9]
Proofpoint Email Protection	Захисний шлюз для email (Enterprise)	Мультишаровий AI-аналіз: NexusAI (ML-модель) перевіряє контент, заголовки, вкладення; Nexus Threat Graph звіряє URL і відправників з глобальною базою загроз; евристики для виявлення BEC. (шахрайства від імені керівників)	Пісочниця для виконання вкладень; ізоляція підозрілих URL; тренування співробітників (фішинг- симуляції) дозволяє знизити ризик атак на 90%; автоматизоване реагування . (видалення доставлених фішинг-листів зі скриньок) [11]
Barracuda Email Protection	Захисний шлюз для email (Cloud/SaaS)	Anti-Fraud Intelligence (байєсовський фільтр фішингу); поведінковий аналіз аномалій; евристичне виявлення спуфінгу (перевірка відправника, домену); валідація посилань з урахуванням тайпсквотингу.	Хмарна пісочниця для вкладень (ATP); переписування URL- лінків для перевірки при кліку; захист від доменних підробок (SPF/DKIM); гнучкі варіанти розгортання (шлюз або сервіс) [12]

1.3 Послідовність роботи та функціональні етапи підсистеми

Проектована підсистема виявлення електронних повідомлень з ознаками соціальної інженерії складається з кількох основних етапів, а саме збір і попередня обробка даних, аналіз тексту за допомогою хмарного сервісу, класифікація повідомлення як шкідливого або нейтрального. Узагальнена структура рішення представлена на рисунку 1.3.1.

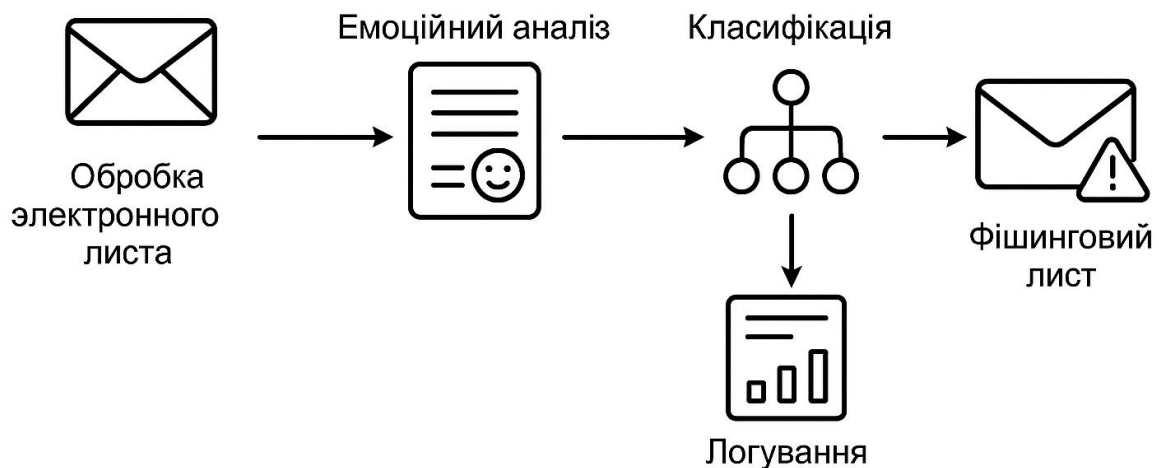


Рисунок 1.3.1 Узагальнена структура системи виявлення соціальної інженерії в електронних повідомленнях з використанням емоційного аналізу тексту [2].

Основні етапи оброблення електронних повідомлень такі:

- а) отримання електронного повідомлення;
- б) інтеграція з поштовим сервером або клієнтом;
- в) попередня обробка;
- г) емоційний аналіз тексту електронного повідомлення;
- г) класифікація та виявлення соціальної інженерії;
- д) збереження результатів та реагування:
 - 1) блокування або карантин;
 - 2) маркування повідомлення;
 - 3) сповіщення адміністратора або користувача;
 - 4) логування.

Система інтегрується з поштовим сервером або клієнтом, отримуючи копію кожного вхідного електронного повідомлення для аналізу. Це може реалізовуватись шляхом перенаправлення пошти на аналіз або виклик системи сканування при надходженні повідомлення (подія). Вхідний email може бути представлений у форматі RAW (EML), що містить заголовки та тіло повідомлення [15].

На етапі попередньої обробки виділяється зміст повідомлення, релевантний для аналізу. З тексту повідомлення можуть видалятися HTML-теги,

скрипти, зайві пробіли, щоб отримати чистий текст. Також бажано вилучити цитати попередніх листувань (reply chain), сигнатури, щоб аналізувати саме текст, створений відправником. Можна виконати нормалізацію тексту - привести його до нижнього регістру, виправити поширені спеціальні символи (наприклад, «\$@» на «@»), транслітерувати юнікодні схожі символи тощо. Якщо повідомлення багатомовне – визначити мову і за потреби здійснити машинний переклад тексту на підтримувану мову аналізу (наприклад, англійську). На цьому ж кроці система може видобути інші ознаки для класифікації. Наприклад, перевірити домен відправника на наявність у списках шахрайських, витягнути всі посилання для окремої перевірки тощо. Однак центральним для нашого підходу є саме аналіз змісту тексту з погляду емоційної тональності [14].

Ключова підсистема – модуль аналізу тональності, реалізований через хмарний API Azure Language Service (сервіс обробки природної мови від Microsoft). Підготовлений на попередньому кроці текст електронного повідомлення передається до цього сервісу для визначення емоційного забарвлення. Azure Cognitive Service for Language надає готовий метод Sentiment Analysis, який повертає оцінки тональності тексту — як правило, це рівні позитивності, нейтральності та негативності із відповідними балами впевненості [10]. У контексті виявлення соціальної інженерії нас найбільше цікавить негативна тональність (страх, злість, терміновість) або неприродно позитивна/оманлива тональність (обіцянки нагород) [13]. Наприклад, якщо сервіс аналізу повертає високий відсоток негативного забарвлення і ключові слова, пов’язані зі страхом чи терміновістю, – це може слугувати ознакою фішингу. Azure Language API також дозволяє виконати Opinion Mining – витягнути із тексту окремі фрази, що несуть думку чи емоцію. Це може допомогти визначити, яка саме частина повідомлення є залякувальною чи маніпулятивною (наприклад, фрази на кшталт «негайно», «ви виграли приз» тощо). Важливо зазначити, що використання готового хмарного сервісу звільняє від потреби навчати власну модель аналізу тональності – Azure постійно

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

вдосконалює свої алгоритми на основі великих обсягів даних, гарантуючи актуальність визначення емоцій [5].

Отримавши результати емоційного аналізу, система комбінує їх з іншими правилами та ознаками для ухвалення рішення щодо безпечності електронного повідомлення. Для класифікації може використовуватись спеціалізований скрипт з евристичними правилами або модель машинного навчання. Якщо емоційний аналіз виявив високу негативну тональність тексту, наявність слів-переконань («терміново», «негайно», «загроза», «виграш») та одночасно повідомлення містить запит на персональні дані або підозріле посилання – класифікувати його як ймовірно фішинговий. Для більшої точності можна враховувати перевірку SPF/DKIM (чи автентичний відправник), репутацію домену посилань, відповідність мови повідомлення типу запиту тощо. На виході класифікатор присвоює мітку - «шкідливий» або «нейтральний». Можливі градації з впевненістю (наприклад, бал ризику 0 —100). Якщо використовується модель ML, її треба попередньо навчити на корпусі електронних повідомлень, де відомо, які з них шкідливі, а які ні, використовуючи як ознаки результати емоційного аналізу, наявність ключових слів, довжину тексту, наявність вкладень, час відправки тощо [16].

Залежно від ухваленого рішення, система вживає відповідних дій. Якщо повідомлення визнано шкідливим, передбачаються такі варіанти реакції:

- 1) блокування або карантин;
- 2) маркування листа;
- 3) сповіщення адміністратора або користувача;
- 4) логування.

Електронне повідомлення не доставляється в скриньку адресата, а поміщається в карантинне сховище. Адміністратор чи користувач (за наявності прав) можуть переглянути карантин і вирішити, чи це була помилкова спрацювання.

Тема повідомлення або його вміст позначається попередженням (наприклад, додається префікс [Phishing suspected] у тему або банер у тілі листа

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				21
Змн.	Арк.	№ докум.	Підпис	Дата		

з попередженням). Це менш інвазивний метод – користувач отримає лист, але буде попереджений про підозру.

Система може надіслати окремий сигнал (email, повідомлення) адміністратору безпеки про виявлену соціальну інженерію. Також може інтегруватися з SIEM-системою для реєстрації інциденту. Користувачу можна надіслати автоматичну відповідь, що лист, ймовірно, є шахрайським.

Вся інформація про проаналізований лист (заголовки, отримані оцінки тональності, рішення класифікатора) зберігається в базі даних або файлі журналу. Це дозволить надалі вдосконалювати алгоритм, аналізувати помилкові спрацювання та успішні атаки, що минули систему.

У випадку, якщо лист визнано безпечним, він безперешкодно доставляється адресату. Проте і в цьому разі може виконуватися логування результатів (для моніторингу й удосконалення). Важливо забезпечити мінімум хибнопозитивних спрацювань, аби не заважати нормальній роботі – тому поріг чутливості класифікатора слід вибирати обережно, можливо, налаштовуючи різні рівні (помірна підозра — просто позначити, висока – заблокувати).

Робота системи передбачає використання хмарних сервісів і компонентів, які взаємодіють між собою за допомогою API. Зокрема, надходження нового електронного листа може ініціювати виконання безсерверної функції (наприклад, Azure Function), яка викликає Azure Language API для аналізу тексту. Отримані результати емоційного аналізу обробляються згідно з визначеними правилами класифікації, після чого зберігаються у хмарному сховищі (Azure Blob Storage або Azure SQL).

Такий підхід забезпечує масштабованість і стійкість системи, оскільки обробка кожного листа відбувається окремо в ізольованому середовищі, що дає змогу ефективно працювати навіть при великому навантаженні.

1.4 Інструментальні засоби реалізації та вимоги до апаратних засобів

Для реалізації програмного комплексу доцільно обрати мови і платформи, зручні для роботи з текстом та хмарними сервісами. Одним із варіантів є мова

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

Python — вона має багату екосистему бібліотек для обробки тексту (NLTK, TextBlob, spaCy тощо) і зручні SDK для виклику Azure Cognitive Services. Python добре підходить для швидкої розробки прототипів, скриптів обробки листів та інтеграції з API. Альтернативно (або в доповнення) можна використовувати C# на платформі .NET, особливо якщо планується задіяти Azure Functions або інші служби в екосистемі Microsoft. Для C# існує офіційний клієнтський SDK Azure AI, що спрощує виклик сервісів аналізу тексту. Обидві мови — Python і C# — дозволяють побудувати необхідну логіку. Вибір може залежати від кваліфікації команди розробки та вимог до інтеграції. Можливий підхід використати Python для створення основного модуля аналізу (демонстраційного скрипту), а згодом перенести його в Azure Functions (які підтримують як Python, так і .NET).

Як зазначено, центральним інструментом для аналізу емоційної тональності є Azure Cognitive Services: Language (раніше відомий як Text Analytics API). Необхідно зареєструвати відповідний сервіс у Azure (Azure AI Language) та отримати ключ API і endpoint. Для викликів з Python можна використати офіційну бібліотеку azure-ai-textanalytics або звертатися до REST API напряду (через HTTP-запити). В C# аналогічно використовується пакет Azure.AI.TextAnalytics. Окрім аналізу тональності, цей сервіс можна задіяти й для інших NLP-задач, наприклад, виявлення ключових фраз (Key Phrase Extraction) — щоб визначити основні об'єкти чи теми листа, або Language Detection — для визначення мови повідомлення [4]. Доцільно також використати Azure SDK for Storage (наприклад, azure-storage-blob для Python) для роботи зі сховищем (якщо електронні повідомлення або результати треба складати у Azure Blob Storage). Якщо планується реалізація логіки як безсерверної функції, знадобиться Azure Functions — для Python реалізація функцій здійснюється, наприклад, через пакет azure.functions, а для .NET — через відповідні бібліотеки та шаблони проектів Azure Function.

В процесі розробки можуть бути корисними інші інструменти:

- а) середовища розробки: PyCharm для Python — забезпечать зручне написання та налагодження коду;

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

- б) бібліотеки для роботи з email: модуль Python електронної пошти (із стандартної бібліотеки) чи сторонні пакети (наприклад, imaplib для отримання листів з поштової скриньки через IMAP, або ruzmail для розбору EML-файлів). В .NET можна використати пакет MailKit для доступу до поштових скриньок;
- в) інтеграція з поштовим сервером: якщо потрібна автоматична обробка, можна налаштувати правила пересилання в Exchange/Gmail або використати SMTP relay, що надсилатиме копії листів на аналіз. Для тестування — згенерувати тестову скриньку і наповнити її контрольними прикладами.

Оскільки основне навантаження з аналізу тексту бере на себе хмарний сервіс Azure, вимоги до локального обладнання невисокі. Достатньо стандартного комп'ютера розробника (Windows або Linux) з доступом до інтернету.

Рекомендованими вимогами є процесор Intel Core i3 (або аналогічний), 2 ГБ ОЗП і більше – для комфортної роботи IDE та можливого локального тестування. Місця на диску потрібно небагато (кілька сотень МБ для середовища Python та SDK). Якщо планується запуск системи на власному сервері (наприклад, аналізатор, розгорнутий on-premises), сервер має відповідати вимогам виробничого навантаження, для обробки сотень листів на годину підійде віртуальна машина з 1–2 vCPU, 2–4 ГБ RAM. У разі великого потоку (тисячі листів на годину) краще масштабувати горизонтально, запустивши декілька екземплярів служби або використовуючи безсерверну архітектуру, яка сама масштабується (Azure Functions автоматично додають інстанси при збільшенні вхідних подій).

Хоча в проєкті робиться акцент на Azure, варто згадати, що існують аналогічні сервіси від інших постачальників:

- Google Cloud Natural Language API;
- IBM Watson Tone Analyzer ;
- та інші.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				24
Змн.	Арк.	№ докум.	Підпис	Дата		

Ці се можуть визначати тональність тексту. Проте використання саме Azure обґрунтоване інтеграцією з іншими компонентами (Functions, Storage) в єдиній екосистемі. Для зберігання даних результатів можна застосувати як просте Blob Storage (зберігання EML-файлів і JSON-звітів аналізу), так і Azure SQL / Cosmos DB для структурованого зберігання журналу перевірок. Щодо інструментів тестування – доцільно мати набір тестових фішингових email (наприклад, відомі шаблони зі phishing database або згенеровані власноруч), на яких перевіряти роботу емоційного аналізу та правил класифікації.

Підсумуємо вибір технологій для даного проєкту:

- а) мова програмування: Python для інтеграції в Azure: WPF (Windows Presentation Foundation, .NET, C#) — для розроблення графічного інтерфейсу користувача;
- б) хмарний сервіс: Azure Cognitive Services (Language – Sentiment Analysis);
- в) SDK та бібліотеки: azure-ai-textanalytics, azure-functions, deep_translator (GoogleTranslator) для перекладання листів, bs4 (BeautifulSoup) для парсингу HTML-контенту з листів;
- г) обладнання: стандартний ПК розробника: у хмарі – обліковий запис Azure з достатнім рівнем сервісного плану (для Cognitive Service).

Обрана конфігурація інструментів дозволить ефективно реалізувати систему, забезпечивши гнучкість розгортання (локально чи в хмарі) та масштабування, а також використавши сучасні можливості ІІІ для аналізу емоцій в тексті електронних листів. Це створює міцне підґрунтя для прототипування і подальшого впровадження системи в реальне середовище з метою підвищення кібербезпеки організації.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до розділу 1

Здійснено аналіз проблеми соціальної інженерії в електронному листуванні та обґрунтовано необхідність створення системи, здатної автоматично виявляти загрози маніпулятивного характеру на основі емоційного аналізу тексту. Особливу увагу приділено масштабам поширення таких атак, як фішинг, їхній динаміці зростання та ефективності, що значною мірою зумовлена використанням емоційного впливу – залякування, терміновості, викликання довіри тощо. Також проведено огляд існуючих підходів до виявлення соціальної інженерії, зокрема досліджень, що поєднують емоційний та тональний аналіз, що підтверджує перспективність обраного напрямку.

Запропоновано загальну структуру функціонування системи, яка включає етапи отримання листа, його попередню обробку, емоційний аналіз тексту, класифікацію повідомлення як шкідливого або нейтрального та відповідні дії. Такий підхід дозволяє виявляти загрози соціальної інженерії навіть за відсутності явних технічних ознак, спираючись на емоційний контекст повідомлення.

Визначено відповідні інструментальні засоби реалізації проєкту. Основною мовою програмування обрано Python через його зручність інтеграції з хмарними сервісами та обробки тексту. Обґрунтовано вибір Azure Cognitive Services як базової платформи для реалізації аналізу емоційної тональності.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ВИЯВЛЕННЯ АТАК СОЦІАЛЬНОЇ ІНЖЕНЕРІЇ В ЕЛЕКТРОННИХ ЛИСТАХ

2.1 Загальна структура програмної підсистеми

Проектована підсистема передбачається як консольна програма, розроблена мовою Python, яка виконуватиме автоматизований емоційний аналіз текстів електронних листів із подальшим виявленням ознак соціальної інженерії. Вона передбачає підтримку декількох режимів роботи, а також наявність етапу попереднього формування навчального корпусу листів, що слугуватиме основою для аналізу та перевірки працездатності реалізованого алгоритму.

Основна логіка обробки включатиме такі етапи:

- 1) отримання вхідного тексту листа (від користувача або з корпусу);
- 2) надсилання запиту з текстом до хмарного API Azure Language Service для визначення емоційної тональності;
- 3) обробка отриманих результатів: зчитування рівнів негативної та позитивної емоційності;
- 4) обчислення ризикового балу та класифікація листа;
- 5) виведення результату в консоль: вказівка, чи вважається лист шкідливим або нейтральним, значення ризику та метрики.

Схему роботи майбутньої системи показано на рисунку 2.1.1.



Рисунок 2.1.1 – Схема роботи програмної системи

2.2 Формування та маркування навчального корпусу електронних повідомлень

Формується вхідний набір даних, для подальшого тестування й калібрування алгоритмів емоційного аналізу та детектування соціальної інженерії. Процес складається з трьох ключових кроків:

- 1) формування сирого корпусу;
- 2) попередня обробка повідомлень;
- 3) маркування та формування навчальної, валідаційної й тестової вибірок.

Крок 1. Збір і структурування сирого корпусу

Формування якісного корпусу листів — критичний етап для розробки та тестування системи автоматичного виявлення соціальної інженерії. У цій роботі для створення репрезентативного набору даних були використані різноманітні джерела, що дозволило охопити широкий спектр реальних сценаріїв електронного листування.

Джерела листів для тестування:

- 1) відкриті датасети із мітками (phishing / ham);
- 2) власноруч зібрані україномовні приклади.

Відкриті датасети із мітками — використано міжнародні публічні корпуси електронних листів, де кожен лист вже містить анотацію - «шкідливий» (phishing, spam, scam) чи «нейтральний» (ham). Це дозволило швидко зібрати значний обсяг англomовного матеріалу для аналізу структури та типових патернів соціальної інженерії.

До корпусу включено реальні україномовні електронні повідомлення з офіційних джерел CERT-UA (національна команда реагування на кіберінциденти) та анонімізовані корпоративні розсилки, отримані з дозволу користувачів. Це забезпечило якісне представлення сучасних атак, адаптованих під локальний контекст.

Загальний обсяг корпусу склав 1134 електронних повідомлень:

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

- 1) 684 нейтральних електронних листів, що не містять ознак маніпуляцій або психологічного впливу;
- 2) 450 шкідливих листів, у яких явно присутні елементи соціальної інженерії (фішинг, вимагання, інші шкідливі дії).

Кожен лист було приведено до уніфікованого JSON-формату з мінімальними полями - унікальний ідентифікатор (id), клас (label: нейтральний або шкідливий), чистий текст (text).

Для зручності автоматичної обробки всі файли згруповано у тематичних папках:

- /data/Neutral/all.json — нейтральні електронні повідомлення;
- /data/Malicious/all.json — шкідливі електронні повідомлення.

Цей підхід забезпечив стандартизацію корпусу, репрезентативність даних та високу якість маркування, що є необхідною умовою для ефективного навчання й тестування алгоритмів виявлення соціальної інженерії.

Крок 2. Попередня обробка повідомлень

Оскільки значна частина доступних корпусів електронних листів у сфері соціальної інженерії представлена англійською мовою, для уніфікації та забезпечення подальшого аналізу було реалізовано автоматичний переклад текстів на українську.

Переклад виконувався за допомогою окремого Python-скрипта `translate.py` (листинг якого наведено у додатках до кваліфікаційної роботи). Скрипт використовує API Google Translate через бібліотеку `deep-translator` [17] для автоматичної обробки великих масивів тексту. Для уникнення обмежень на довжину запиту всі електронні повідомлення попередньо розбивалися на частини оптимального розміру, після чого кожна частина перекладалася окремо за допомогою виклику:

```
translated_parts.append(GoogleTranslator(source='auto',target='uk')
.translate(part))
```

У підсумку, всі неукраїномовні електронні повідомлення було конвертовано в єдиний формат українською мовою, що забезпечило повну

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				29
Змн.	Арк.	№ докум.	Підпис	Дата		

сумісність із модулями подальшого емоційного аналізу та класифікації у складі проєкту.

Для забезпечення коректності та уніфікованості подальшої обробки, всі електронні повідомлення були піддані комплексній попередньої обробки з метою видалення зайвих структурних елементів, стандартизації форматування та захисту персональних даних користувачів

Попередня обробка текстів включала такі основні кроки:

- 1) видалення HTML- розмітки (тегів, стилів і скриптів);
- 2) стандартизацію кодування та форматування;
- 3) уніфікацію структури текстів.

Більшість сучасних електронних повідомлень містить HTML-розмітку, стилі, а іноді й вбудовані скрипти. Для коректного аналізу тексту ці елементи видалено, залишивши лише змістовну частину повідомлення. Це реалізовано за допомогою спеціального Python-скрипта `clearHtml.py`, який розроблено у межах цієї кваліфікаційної роботи (листинг наведено у додатках). Скрипт проходить кожен лист, видаляє HTML-розмітку, CSS-стилі та JavaScript-фрагменти, після чого залишається лише чистий текст.

Всі тексти було перекодовано у формат UTF-8, що гарантує коректне відображення кирилиці та символів будь-яких мов. Додатково з текстів автоматично видалялися надлишкові пробіли, табуляції, непечатні та спеціальні символи (наприклад, символи керування рядком, нульові байти). Після завершення попередньої обробки усі електронні повідомлення були автоматизовано переведені у формат JSON. Кожен лист став окремим елементом масиву об'єктів із полями `id`, `text` та (для розмічених вибірок) `label`. Такий підхід забезпечив стандартизацію, зручність подальшої обробки й уніфікований доступ до всіх текстів для аналізу.

Завдяки зазначеним етапам, кожний результативний файл у форматі `*.txt` містить лише лаконічний і структурований зміст листа - без додаткової розмітки, сміттєвих даних та персональної інформації. Це значно спрощує подальшу автоматизовану обробку й мінімізує ризик помилок при аналізі тексту.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

Всі етапи попередньої обробки були виконані автоматизовано з використанням власного Python-скрипта, що дозволило швидко й якісно підготувати корпус для машинного аналізу та забезпечити відтворюваність усіх дій. Опис і листинг скрипта clearHtml.py наведено у додатках до роботи.

Крок 3. Маркування та формування вибірок, валідаційної й тестової вибірок

Для подальшого аналізу електронних листів усі зібрані тексти було автоматично конвертовано у структурований формат JSON за допомогою власного Python-скрипта folder.py, який розроблено у межах цієї кваліфікаційної роботи (листинг наведено у додатках)

Скрипт здійснює пакетне зчитування JSON файлів з відповідних директорій, присвоює кожному листу унікальний ідентифікатор (id) та, у разі необхідності, додає мітку класу (label). Для зручності аналізу даних і тестування алгоритмів було організовано три основні папки (див. табл. 2.2.1).

Таблиця 2.2.1

Опис папок та їх призначення

Папка	Опис	Призначення
Malicious	Електронні повідомлення з ознаками соціальної інженерії	База для тренування і перевірки на небезпечних листах
Neutral	Звичайні (нейтральні) електронні повідомлення	База для тренування і перевірки на безпечних листах
Mixed	Змішаний набір електронних повідомлень	Експериментальна вибірка для сліпого тестування ефективності

Кожен лист у цих файлах має однакову структуру запису з наступними полями (див табл. 2.2.2).

Опис полів у файлах

Поле	Тип	Опис	Присутність
id	int	Унікальний ідентифікатор листа у межах корпусу	Malicious, Neutral, Mixed
text	str	Основний текст листа (очищений і анонімізований)	Malicious, Neutral, Mixed
label	str	Клас листа — "malicious" (шкідливий) або "neutral" (нейтральний)	Malicious, Neutral

2.3 Вибір та налаштування моделі емоційного аналізу

Для реалізації алгоритму емоційного аналізу обрано сервіс Azure Language (TextAnalytics) — оскільки він офіційно підтримує українську мову та надає швидкий і надійний процес обробки тексту (рис.2.3.1) [9].

Основним завданням даного сервісу у підсистемі є виявлення емоційного забарвлення тексту листа - зокрема, негативних, позитивних та нейтральних елементів, які часто виступають індикаторами соціальної інженерії (психологічний тиск, погрози, емоційна маніпуляція).

Основні переваги Azure Language (TextAnalytics):

- а) підтримка української мови: аналіз тональності без необхідності перекладу;
- б) масштабованість: хмарні інстанси автоматично адаптуються під навантаження;
- в) інтеграція: REST API з простою структурою запитів і підтримкою SDK для Python (azure-ai-textanalytics) — що дозволяє швидко реалізувати інтеграцію в код;
- г) документованість: повна технічна документація, підтримка, приклади.

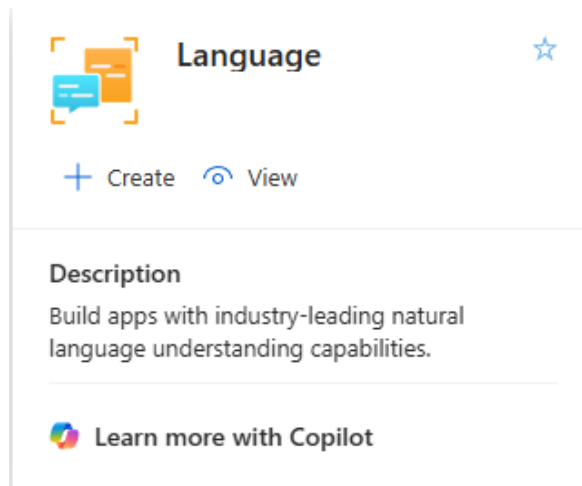


Рисунок 2.3.1 Azure Language

Кроки підготовки налаштування моделі емоційного аналізу:

- 1) реєстрація ресурсу: у Azure Portal створити ресурс типу «Language (Text Analytics)» з безкоштовним рівнем (F0, до 5 000 запитів на місяць);
- 2) отримання доступу: у налаштуваннях ресурсу копіюємо два значення — ключ підписки (Key) та URL-адресу кінцевої точки (рис.2.3.2);
- 3) налаштування безпеки: переконуємося, що до REST API можна звертатися тільки з авторизованими запитами.

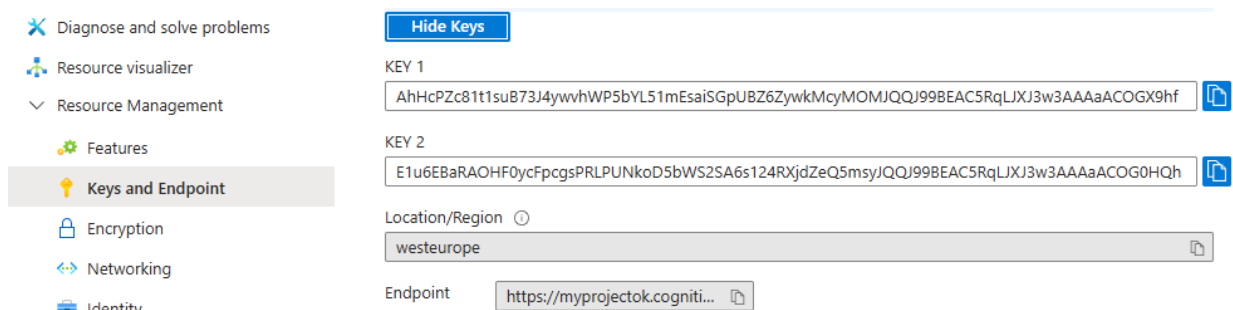


Рисунок 2.3.2 Приклад Key та Endpoint

2.4 Реалізація алгоритму емоційного аналізу та класифікації електронних повідомлень

Загальна послідовність виконання основних етапів емоційного аналізу та класифікації листів показана на схемі нижче (рис. 2.4.1). Побудована схема відображає розроблений автором алгоритм функціонування підсистеми

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				33
Змн.	Арк.	№ докум.	Підпис	Дата		

виявлення фішингових листів на основі емоційного аналізу. Вона є частиною реалізованої програмної логіки підсистеми.

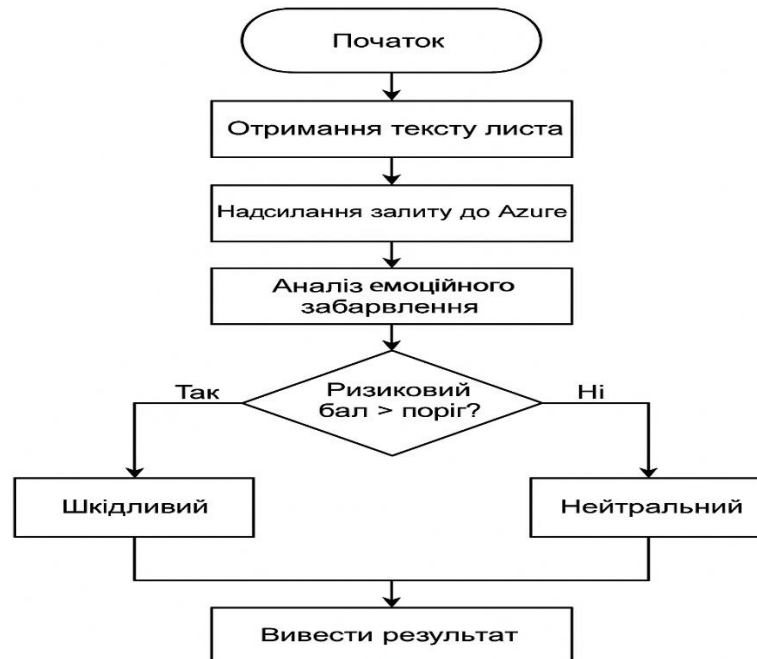


Рисунок 2.4.1 Алгоритм емоційного аналізу тексту електронного листа

Розглянуто практичну реалізацію компонента на мові програмування Python, який здійснює отримання емоційних характеристик тексту листа з хмарного сервісу та обчислення оцінки ризику (*risk_score*) на основі цих характеристик.

В основі модуля використано офіційну бібліотеку *azure-ai-textanalytics* для Python. Після встановлення пакета та отримання доступу до сервісу (endpoint, ключ), створюється єдиний клієнт для взаємодії з Azure Text Analytics [7]:

```

from azure.core.credentials import AzureKeyCredential
from azure.ai.textanalytics import TextAnalyticsClient

ENDPOINT = "https:// <your-resource-
name>.cognitiveservices.azure.com"
KEY = "<your-key>"

client = TextAnalyticsClient(endpoint=ENDPOINT,
                             credential=AzureKeyCredential(KEY))
  
```

Реалізація функції емоційного аналізу тексту та класифікації результату:

```

def analyze_and_classify(text: str, threshold: float) -> dict:
    doc = trim_text(text)
    resp = client.analyze_sentiment([doc], language="uk")[0]
  
```

```

if hasattr(resp, 'error'):
    resp = client.analyze_sentiment([doc], language="uk")[0]
sentiment = resp.sentiment
pos = resp.confidence_scores.positive
neg = resp.confidence_scores.negative
neu = resp.confidence_scores.neutral
risk_score = min(neg * 0.7 + pos * 0.3, 1.0)
label = "шкідливий" if risk_score > threshold else
"нейтральний"
sentences_info = []
for sent in resp.sentences:
    sentences_info.append({
        'text': sent.text,
        'sentiment': sent.sentiment,
        'positive': round(sent.confidence_scores.positive, 2),
        'neutral': round(sent.confidence_scores.neutral, 2),
        'negative': round(sent.confidence_scores.negative, 2)
    })
return {
    'sentiment': sentiment,
    'positive': round(pos, 2),
    'neutral': round(neu, 2),
    'negative': round(neg, 2),
    'risk_score': round(risk_score, 2),
    'classification': label,
    'sentences': sentences_info
}

```

Функція приймає один лист (рядок text), викликає метод сервісу analyze_sentiment з параметром language="uk" (uk — українська мова). Для кожного листа з підготовленого корпусу здійснюється запит до Azure із параметром language="uk". Відповідь містить:

- загальну тональність документа (sentiment) — одне із значень: "positive", "neutral", "negative";

- оцінки впевненості (confidence scores) - positive, neutral, negative (від 0 до 1), що характеризують рівень кожної з тональностей;
- детальний аналіз по реченнях — для кожного речення тексту окремо визначається тональність та відповідні оцінки;
- ключові фрази (key phrases) - окремий запит до Azure повертає перелік найбільш змістовних фраз у тексті (наприклад, "банківський рахунок", "оновіть дані", "підтвердження").

Всі ці дані збираються і зберігаються для подальшої класифікації.

У зв'язку з відсутністю усталених стандартних формул розрахунку ризику на основі емоційної тональності в цій роботі запропоновано емпіричну формулу (2.1), що зважено агрегує показники негативності (Neg) та позитивності (Pos) з подальшим обмеженням результату до діапазону 0...1. Такий підхід (зважене об'єднання нормованих ознак із використанням ваг) є типовою інженерною практикою у задачах прийняття рішень та ML [6].

$$R = \min(\text{neg} * A + \text{pos} * B), \quad (2.1)$$

де R — число в діапазоні 0...1, яке узагальнює емоційні ознаки листа й відображає ймовірність ознак соціальної інженерії (вищий бал - більший ризик);

$\min(a, b)$ — функція мінімуму, повертає менше з двох значень;

neg — коефіцієнт негативності листа;

A — ваговий коефіцієнт негативності листа;

pos — коефіцієнт позитивності листа;

B — ваговий коефіцієнт позитивності листа.

У розрахунку оцінки ризику (risk_score) значення нейтральної тональності (neutral) свідомо не використовується, оскільки воно не несе інформації про потенційну загрозу. Замість цього фокус робиться на негативних та позитивних емоційних сигналах, які найчастіше виявляються у фішингових повідомленнях як інструменти впливу або маніпуляції.

У процесі розробки підсистеми реалізовано механізм класифікації електронних листів на шкідливі (із ознаками соціальної інженерії) та нейтральні. Рішення приймається на основі емоційних коефіцієнтів, сформованих після аналізу вмісту повідомлення. Для цього використовується гнучка схема оцінювання ризику, яка дозволяє налаштовувати поріг виявлення загроз відповідно до вимог до безпеки. Такий підхід забезпечує можливість адаптації підсистеми до різних сценаріїв використання.

Кожен лист, який надходить на аналіз, характеризується такими параметрами:

- `risk_score` — ризиковий бал, обчислений за формулою на основі ймовірностей негативної та позитивної тональності;
- ключові фрази — виділені автоматично для пояснення результату або підсилення інтерпретації (опціонально);
- детальний аналіз по реченнях — для додаткової інтерпретації рішення (визначення, у яких саме частинах листа є тривожні патерни).

Подальша класифікація листа здійснюється на основі порогового значення. Якщо розрахована оцінка ризику перевищує встановлений поріг, лист визначається як шкідливий, у протилежному випадку — як нейтральний.

Для класифікації використовується наступний код:

```
label = "шкідливий" if risk_score > threshold else "нейтральний"
```

Опис змінних:

- `label`: категорія, до якої віднесено лист після класифікації: "шкідливий" або "нейтральний", залежно від ризикового балу;
- `risk_score`: числовий показник рівня ризику, розрахований на основі емоційної тональності листа. Значення в межах від 0 до 1;
- `threshold`: порогове значення (від 0 до 1), з яким порівнюється `risk_score`. Якщо `risk_score` перевищує це значення, лист вважається шкідливим. Поріг можна змінювати вручну, підбираючи його експериментально для досягнення кращого балансу між точністю (`precision`) та повнотою (`recall`).

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				37
Змн.	Арк.	№ докум.	Підпис	Дата		

У режимі автоматизованого аналізу система використовує всю базу або випадково формує підвибірку листів для тестування та самостійно обчислює ключові показники якості класифікації — точність (precision), повноту (recall), F1-міру, специфічність (specificity) і загальну точність (accuracy) — разом із підрахунком кількості правильних і помилкових спрацювань (TP, FP, TN, FN), що дає змогу об'єктивно порівнювати конфігурації моделі та калібрувати порогові значення.

Для розрахунку метрик якості класифікації в програмному коді використовуються стандартні формули з машинного навчання. Вони реалізовані у вигляді обчислювальних виразів мовою Python:

```
precision = tp / (tp + fp) if (tp + fp) > 0 else 0
recall = tp / (tp + fn) if (tp + fn) > 0 else 0
f1 = 2 * precision * recall / (precision + recall) if (precision + recall) > 0 else 0
specificity = tn / (tn + fp) if (tn + fp) > 0 else 0
accuracy = (tp + tn) / (tp + fp + tn + fn) if (tp + fp + tn + fn) > 0 else 0
```

Для оцінювання роботи двокласового класифікатора електронних повідомлень у роботі застосовано стандартні метрики та формули машинного навчання [6]:

- точність (precision);
- повнота (recall);
- F1-міра (F1-score);
- специфічність (Specificity);
- загальна точність (accuracy).

Точність (precision) — частка справді шкідливих електронних повідомлень серед усіх, які система позначила як шкідливі. Показник високий, коли мало хибно-позитивних спрацювань (FP) (2.2).

$$P = \frac{TP}{TP+FP}, \quad (2.2)$$

де P — точність класифікації (precision);

TP — кількість шкідливих повідомлень, правильно класифікованих як шкідливі;

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				38
Змн.	Арк.	№ докум.	Підпис	Дата		

FP — кількість нейтральних повідомлень, помилково класифікованих як шкідливі.

Повнота (recall, чутливість) — частка всіх реально шкідливих електронних повідомлень, які система правильно виявила. Висока повнота означає мало пропущених шкідливих листів (FN) (2.3).

$$R = \frac{TP}{TP+FN}, \quad (2.3)$$

де R — повнота (recall, чутливість), частка всіх реально шкідливих повідомлень, які система правильно виявила;

TP — кількість шкідливих повідомлень, правильно класифікованих як шкідливі;

FN — кількість шкідливих повідомлень, помилково класифікованих як нейтральні.

F1-міра (F1-score) — гармонічне середнє між точністю та повнотою. Зростає тільки коли і precision, і recall високі, тому зручна як єдиний узагальнений показник (2.4).

$$F1 = \frac{2 \cdot P \cdot R}{P + R}, \quad (2.4)$$

де F1 — F1-міра, гармонічне середнє між точністю та повнотою, що узагальнює їх у один показник і зростає лише тоді, коли високі і P, і R;

P — точність за (2.2);

R — повнота за (2.3).

Специфічність (specificity) — частка правильно відкинутих нейтральних електронних повідомлень серед усіх реально нейтральних; вимірює «уважність» моделі до негативного класу (2.5).

$$S = \frac{TN}{TN+FP}, \quad (2.5)$$

де S — специфічність (specificity), частка правильно відкинутих нейтральних листів серед усіх справді нейтральних;

TN — кількість нейтральних повідомлень, правильно класифікованих як нейтральні;

FP — кількість нейтральних повідомлень, помилково класифікованих як шкідливі.

Загальна точність (accuracy) — частка всіх правильно класифікованих електронних повідомлень у вибірці незалежно від класу. За зауваженням підручників з ML, на незбалансованих даних accuracy може бути оманливо високою, тож її слід інтерпретувати разом з іншими метриками (precision/recall/F1) (2.6).

$$A = \frac{TP+TN}{TP+FP+TN+FN}, \quad (2.6)$$

де A — загальна точність, частка всіх правильно класифікованих листів у вибірці незалежно від класу;

TP — правильно класифіковані шкідливі; TN — правильно класифіковані нейтральні;

FP — нейтральні, помилково віднесені до шкідливих; FN — шкідливі, помилково віднесені до нейтральних.

Реалізований алгоритм класифікації відзначається простотою, прозорістю та гнучкістю. Він дозволяє ефективно інтегрувати емоційний аналіз як основу для захисту від соціальної інженерії, а також легко модифікується чи масштабується під інші типи текстів або задач.

Додаткові функції підсистеми:

- поріг спрацювання (DEFAULT_THRESHOLD) задається користувачем і може бути оптимізований експериментально для досягнення найкращого балансу між точністю та повнотою;
- для кожного рішення підсистема виводить не лише клас, а й ключові показники емоційного аналізу та ключові фрази, а також деталізацію по реченнях — що дає змогу користувачу або адміністратору зрозуміти причину класифікації.

Для реалізації графічного інтерфейсу обрано Windows Presentation Foundation (WPF, .NET/C#) (рис. 2.4.2). По-перше, вибір відповідає цільовому середовищу експлуатації — лабораторні машини та робочі станції під управлінням Windows. По-друге, WPF підтримує архітектурний шаблон MVVM,

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

що забезпечує чітке розділення подання, логіки та даних: обчислювальний пайплайн і виклики до Azure AI Language інкапсульовано в сервісному шарі та ViewModel, тоді як XAML-розмітка відповідає лише за відображення.

Семантично графічний інтерфейс дублює режим «аналіз власного тексту», який реалізовано в скриптовій частині на Python: застосовуються ті самі виклики сервісу, та сама послідовність обчислень і та сама формула розрахунку оцінки ризику. Завдяки цьому результати GUI-версії повністю співставні з консольною реалізацією. Водночас режим «аналіз бази електронних повідомлень» (масова/пакетна обробка попередньо сформованої вибірки) на поточному етапі залишено у Python — це відповідає його сценаріям (офлайн-експерименти, калібрування порогів, побудова таблиць і графіків) і дозволяє зберегти спрощену, легку для користувача структуру графічного інтерфейсу. Розширення GUI для пакетної обробки розглядається як напрям подальшого розвитку.

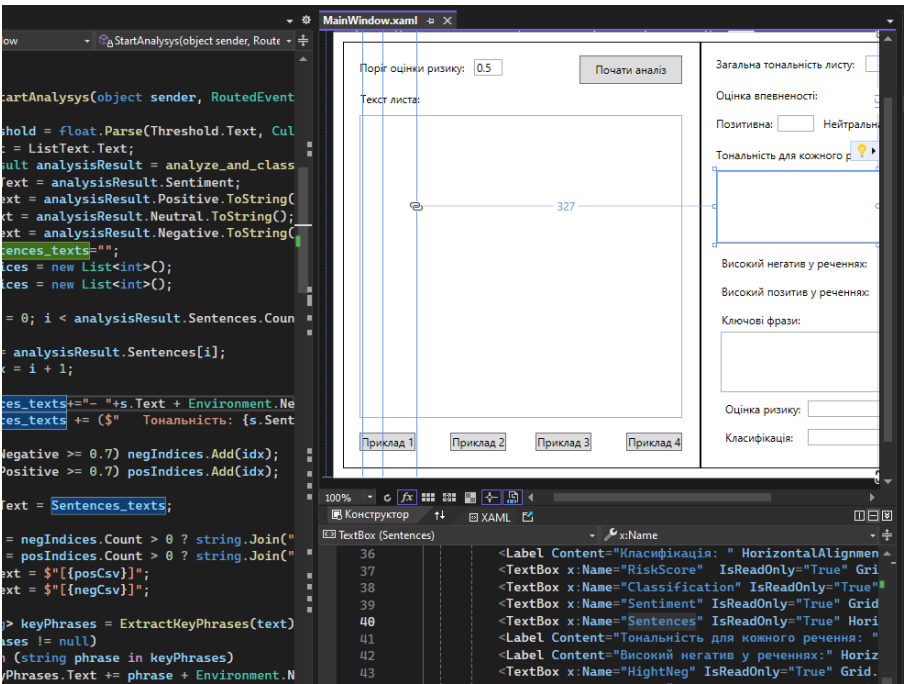


Рисунок 2.4.2 Створення графічного інтерфейсу

2.5 Можливі сценарії застосування підсистеми в реальних умовах

Аналіз загроз соціальної інженерії, показав, що їхня ефективність значною мірою базується на експлуатації людського чинника - емоцій, довіри, страху, неухважності та психологічного тиску. Зловмисники свідомо формують у листах

атмосферу терміновості, паніки або надмірної впевненості, щоб спровокувати імпульсивні дії з боку жертви. Виявлення таких маніпулятивних стратегій вимагає не лише технічного аналізу структури повідомлення, але й розуміння його емоційного контексту. Саме тому застосування емоційного аналізу для виявлення соціальної інженерії розглядається як перспективний напрямок. Теоретичне обґрунтування цієї ідеї лягло в основу реалізації підсистеми, представленої у попередніх розділах.

Підсистема інтегрується до потоку обробки електронної пошти та аналізує зміст листа на наявність емоційних тригерів. Якщо виявлено ознаки емоційного тиску або маніпуляції, модуль класифікації присвоює листу підвищений рівень ризику. Електронні повідомлення, що перевищують заданий поріг ризику, автоматично позначаються як потенційно шкідливі та можуть бути заблоковані або винесені на перегляд адміністратором.

Наведені нижче сценарії демонструють можливі напрями впровадження та розвитку підсистеми в реальних умовах (табл.2.5.1).

Приклад застосування: уявімо, що HR-менеджер компанії отримує електронний лист нібито від відділу IT-підтримки з темою «Негайно змініть пароль». У тексті листа повідомляється про порушення безпеки і вимагається терміново перейти за посиланням для зміни пароля, інакше обліковий запис буде заблоковано. Наша підсистема аналізує зміст цього листа і виявляє високий рівень емоційного тиску (слова на кшталт «негайно», «терміново», погроза блокування доступу). Лист автоматично позначається як підозрілий і сповіщає HR-менеджера про можливу фішингову атаку. Таким чином, співробітник вчасно отримує попередження і не переходить за шкідливим посиланням, а інцидент фіксується для перевірки IT-відділом.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

Можливі сценарії застосування

Сценарій застосування	Потенційні переваги впровадження підсистеми
Корпоративне середовище	Захист конфіденційної інформації бізнесу і клієнтів; виявлення цільових атак до нанесення шкоди; доповнення програм навчання співробітників практичними попередженнями.
Державні установи	Запобігання витоку секретних даних та несанкціонованому доступу; зниження ймовірності успішних фішингових атак на держоргани; автоматизована підтримка кібербезпеки при низькому рівні обізнаності персоналу.
Освітні заклади	Захист акаунтів студентів і викладачів; блокування шахрайських листів (фейкові пропозиції роботи, стипендій тощо); підвищення цифрової грамотності шляхом демонстрації виявлених загроз.
Особисті поштові скриньки	Зменшення ризику фінансового шахрайства та крадіжки особистих даних; попередження користувача про електронні повідомлення, що викликають підозри, ще до взаємодії з ними; додатковий рівень безпеки понад стандартні антиспам-фільтри.

Така підсистема є універсальною: вона підвищує стійкість до соціально-інженерних атак у різних середовищах, де люди взаємодіють із електронною поштою. Застосування емоційного аналізу дозволяє виявити загрози на ранньому етапі, до того як жертва перейде за шкідливим посиланням чи надасть конфіденційні дані. Отже, впровадження розробленої підсистеми сприяє проактивному виявленню та нейтралізації фішингових атак, доповнюючи інші заходи кібербезпеки і мінімізуючи вплив людського фактору.

Висновки до розділу 2

Спроектовано та реалізовано підсистему виявлення соціальної інженерії в електронних повідомленнях з використанням емоційного аналізу. Підсистема реалізована у вигляді консольного застосунку на мові Python з інтеграцією до хмарного сервісу Azure Text Analytics API.

Виконано такі етапи роботи:

- описано загальну структуру програмної підсистеми, яка охоплює основні компоненти, послідовність обробки даних та логіку взаємодії з користувачем;
- сформовано та оброблено корпус електронних листів, зібраного з відкритих джерел для аналізу в наступному розділі кваліфікаційної роботи;
- реалізовано Python-скрипти для автоматичного очищення HTML-коду, перекладу текстів та маркування листів;
- створено алгоритм емоційного аналізу листів та графічний інтерфейс для користувача, що використовує результати Azure API для обчислення ризикового балу;
- запропоновано та реалізовано емпіричну формулу оцінки ризику, яка враховує тональність тексту, код класифікації листів за рівнем загрози;
- здійснено реалізацію підрахунку метрик точності класифікації (precision, recall, F1, specificity, accuracy) на основі результатів роботи алгоритму;
- проведено моделювання реалістичних сценаріїв використання підсистеми в корпоративному, державному та особистому середовищі.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				44
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1 Інтерфейс реалізованої підсистеми

Розроблена підсистема має інтерфейс командного рядка (CLI), підтримує зміну порогу оцінки ризику та два основних режими роботи, що обираються користувачем на старті (див. рисунок 3.1.1). Зміна порога оцінки ризику корисна, щоб повторювати тестування та підбирати оптимальне значення для досягнення кращої точності або повноти. Це особливо корисно для емпіричного підбору порогу перед впровадженням її в реальні умови.

```
Вкажіть поріг оцінки ризику (THRESHOLD), наприклад 0.5 (за замовчуванням 0.5):  
Оберіть режим (1 – Свій текст, 2 – Аналіз бази): 2
```

Рисунок 3.1.1 Вибір режиму

Режими роботи підсистеми:

- 1) аналіз власного тексту електронного повідомлення (див. рис.3.1.2);
- 2) аналіз готової бази електронних повідомлень (див. рис.3.1.4).

У першому режимі користувач вводить текст електронного повідомлення для аналізу. Після введення тексту підсистема автоматично виконує повний цикл оброблення, що охоплює:

- обчислення загальної емоційної тональності листа;
- визначення рівня впевненості по кожному емоційному компоненту;
- обрахунок оцінки ризику (risk_score) за формулою, що враховує негативну та позитивну тональність;
- класифікація листа на основі порогу (default_threshold).

Далі підсистема переходить до глибшого реченнєвого аналізу:

- виводиться тональність кожного окремого речення;
- визначаються ключові фрази, які можуть свідчити про наявність соціально-інженерного змісту;

- окремо зазначено кількість речень, де виявлено високий позитив або негатив, що може вказувати на маніпулятивний тон.

Деталізовані результати подаються у зручному форматі:

- окремо відображаються ключові фрази;
- інтерпретація тональності по реченнях;
- інші показники.

Вивід цього режиму в командному рядку (терміналі) показано на рисунку

3.1.2 та 3.1.3.

```
Вкажіть поріг оцінки ризику (THRESHOLD), наприклад 0.5 (за замовчуванням 0.5):
Оберіть режим (1 – Свій текст, 2 – Аналіз бази): 1
Вкажіть порогове значення ризику (THRESHOLD, від 0 до 1, наприклад 0.5).
Якщо ризик листа (risk_score) буде БІЛЬШЕ цього порогу – лист вважатиметься ШКІДЛИВИМ (класифікація: "шкідливий").
Якщо ризик буде МЕНШЕ або дорівнює пороговому значенню – лист вважатиметься НЕЙТРАЛЬНИМ.
За замовчуванням: 0.5
Введіть значення або натисніть Enter для використання значення за замовчуванням: Глобальні операції з управління ризиками Саллі вітають в

Загальна тональність: mixed
Оцінка впевненості: Позитивна: 0.28, Нейтральна: 0.21, Негативна: 0.51
Оцінка ризику: 0.44
Класифікація: нейтральний

Тональність для кожного речення:
- "Глобальні операції з управління ризиками Саллі вітають вашу нову роль. "
  Тональність: neutral (Позитивна: 0.11, Нейтральна: 0.8, Негативна: 0.09)
- "Якщо ви ще не знали, я зараз перебуваю в Расі в Х'юстоні, і я підозрюю, що наші обов'язки означатимуть, що ми поговоримо при нагоді. "
```

Рисунок 3.1.2 Режим роботи — Власний текст

```
- "Будь ласка, приєднуйтеся до мене у підтримці Саллі в цій додатковій координаційній ролі для глобалізації. "
Тональність: positive (Позитивна: 0.53, Нейтральна: 0.28, Негативна: 0.19)

Високий негатив у реченнях: 2, 8, 12, 14, 15, 17
Високий позитив у реченнях: 4

Ключові фрази:
- віце-президент
- ризиками
```

Рисунок 3.1.3 Режим роботи — Власний текст

У другому режимі, користувачу надається доступ до існуючої структури даних, у якій він може обрати кількість листів для обробки та встановити поріг

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

ризик. Підсистема автоматично обробляє усю попередньо сформовану вибірку електронних листів (у даному випадку — 1134 електронних повідомлень, з яких 684 нейтральних і 450 шкідливих). Кожне електронне повідомлення аналізується за тими самими критеріями, після чого виводяться метрики точності аналізу (precision, recall, F1, specificity, accuracy).

Після завершення аналізу вона генерує розширену статистику, що включає:

- TP (True Positive) — кількість шкідливих листів, які підсистема правильно класифікувала як шкідливі (389);
- FP (False Positive) — кількість нейтральних листів, які помилково позначено як шкідливі (585);
- TN (True Negative) — кількість нейтральних листів, правильно класифікованих як нейтральні (99);
- FN (False Negative) — кількість шкідливих листів, які підсистема не виявила (61).

Додатково обчислюються метрики якості класифікації:

- 1) точність (Precision);
- 2) повнота (Recall);
- 3) F1-міра;
- 4) специфічність (Specificity);
- 5) загальну точність (Accuracy).

Призначення та формули метрик якості класифікації детально описано в підпункті 2.4.

Вивід цього режиму в командному рядку (терміналі) показано на рисунку 3.1.4.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Вкажіть поріг оцінки ризику (THRESHOLD), наприклад 0.5 (за замовчуванням 0.5):
Оберіть режим (1 – Свій текст, 2 – Аналіз бази): 2
Всього листів у базі: 1134 (нейтральних=684, шкідливих=450)
Поточний поріг оцінки ризику: 0.5

Скільки листів обробити? Введіть 'all' або число: all
У вибраній вибірці 1134 листів:
    нейтральних: 684
    шкідливих: 450

Оброблено: 1134/1134 (100.0%)

TP (True Positive) шкідливі листи, які система правильно класифікувала як шкідливі: 89
FP (False Positive) нейтральні листи, які система помилково класифікувала як шкідливі: 237
TN (True Negative) нейтральні листи, які система правильно класифікувала як нейтральні: 447
FN (False Negative) шкідливі листи, які система помилково класифікувала як нейтральні: 361
Точність (Precision): 0.27
Повнота (Recall): 0.20
F1-міра: 0.23
Специфічність (Specificity): 0.65
Загальна точність (Accuracy): 0.47

```

Рисунок 3.1.4 Режим роботи — Аналіз наявної бази електронних повідомлень

3.2 Аналіз ефективності програми

Для оцінки якості роботи розробленої підсистеми виявлення соціальної інженерії було проведено експериментальне тестування на повній базі електронних листів (1134 приклади: 684 нейтральних та 450 шкідливих).

Для оцінки ефективності класифікації було проведено серію тестів за різними пороговими значеннями (threshold). Результати для кожного значення наведено в таблицях 3.2.1–3.2.3.

Результати тестування:

Таблиця 3.2.1

Результати для порогу (threshold): 0.5

Метрика	Позначення	Значення
True Positive	TP	89
False Positive	FP	237
True Negative	TN	447

Метрика	Позначення	Значення
False Negative	FN	361
Точність	Precision	0.27
Повнота	Recall	0.20
F1-міра	F1	0.23
Специфічність	Specificity	0.65
Загальна точність	Accuracy	0.47

Інтерпретація результатів:

- recall (0.20): має найнижче значення — підсистема пропускає близько 80 % шкідливих листів, що критично небезпечно;
- specificity (0.65): має найвище значення — підсистема майже не дає хибних тривог для нейтральних листів, але ціною високого ризику пропущених загроз;
- precision (0.27): лише приблизно кожен четвертий позначений як «шкідливий» лист справді є таким;
- F1-міра (0.23): дуже низька, баланс між точністю та повнотою слабкий;
- accuracy (0.47): залишається на середньому рівні, але показник оманливий — більшість «успіхів» досягається за рахунок правильного визначення нейтральних листів.

Таблиця 3.2.2

Результати для порогу (threshold): 0.4

Метрика	Позначення	Значення
True Positive	TP	206
False Positive	FP	355
True Negative	TN	329
False Negative	FN	244
Точність	Precision	0.37
Повнота	Recall	0.46

Метрика	Позначення	Значення
F1-міра	F1	0.41
Специфічність	Specificity	0.48
Загальна точність	Accuracy	0.47

Інтерпретація результатів:

- recall (0.46): підсистема виявляє менше половини шкідливих листів (TP = 206), тобто ймовірність пропущених загроз висока (FN = 244);
- precision (0.37): трохи більше третини листів, позначених як шкідливі, справді є такими. Хибнопозитивних спрацювань усе ще багато (FP = 355);
- F1-міра (0.41): свідчить про слабкий баланс між виявленням і точністю;
- specificity (0.48): помірне значення, підсистема частіше правильно класифікує нейтральні електронні повідомлення, ніж на порозі 0,3;
- accuracy (0.47): загальна точність покращилася проти менших порогів, однак і далі недостатня для практичного застосування.

Таблиця 3.2.3

Результати для порогу (threshold): 0.3

Метрика	Позначення	Значення
True Positive	TP	284
False Positive	FP	450
True Negative	TN	234
False Negative	FN	166
Точність	Precision	0.39
Повнота	Recall	0.63
F1-міра	F1	0.48
Специфічність	Specificity	0.34
Загальна точність	Accuracy	0.46

Інтерпретація результатів:

- recall (0.63): підсистема виявляє понад 60 % шкідливих листів, що є прийнятним рівнем для виявлення загроз;
- precision (0.39): трохи менше половини позначених як шкідливі листів справді є такими, хибнопозитивних спрацювань багато ($FP = 450$), що може створювати навантаження на модерацію;
- F1-міра (0.48): кращий компроміс між точністю та повнотою; перевищує сценарії з порогами 0,4 і 0,5;
- specificity (0.34): низька, підсистема часто помилково визначає нейтральні електронні повідомлення, як шкідливі;
- ассигасу (0.46): лишається середньою і з огляду на дисбаланс класів не є визначальною метрикою.

Проаналізувавши зміну значень двох ключових метрик класифікації - Recall (повнота) та Specificity (специфічність) — для різних порогів оцінки ризику (threshold) (0.3, 0.4, 0.5) на основі графіків (див. рис. 3.1.5, 3.1.6), було встановлено, що значення порогу (threshold) = 0.4 є оптимальним. Воно забезпечує ефективне виявлення шкідливих листів при прийнятному рівні хибних спрацювань і дозволяє підсистемі:

- виявляти майже половину шкідливих листів;
- мінімізувати навантаження на користувача шляхом зменшення FP;
- зберігати загальний баланс між виявленням загроз (Recall) і зменшенням хибних спрацювань (Specificity);

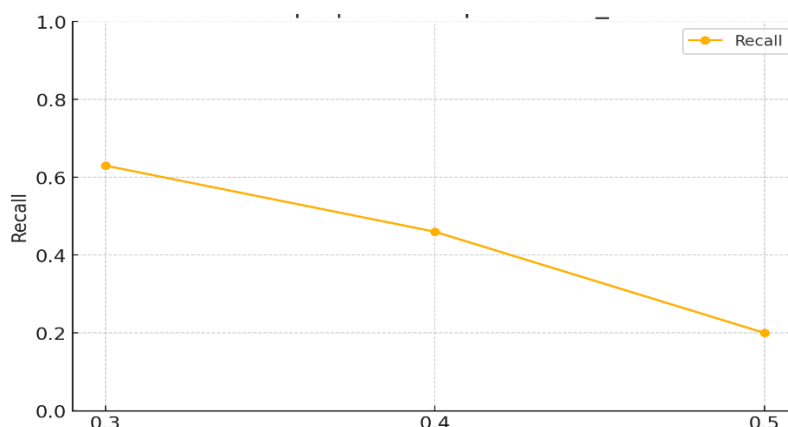


Рисунок 3.2.5 Залежність метрики Recall від порогу (threshold)

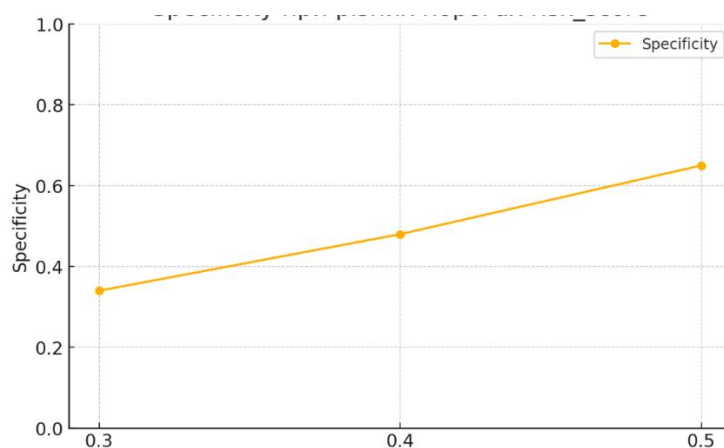


Рисунок 3.2.6 Залежність метрики Specificity від порогу (threshold)

Слід зазначити, що це значення є оптимальним саме для наявної бази даних, використаної у дослідженні. У разі розширення чи заміни корпусу листів — особливо при використанні більш збалансованих, репрезентативних або автентичних наборів — значення порогу може потребувати перегляду. У такому випадку рекомендовано повторно провести тестування та переналаштування підсистеми з урахуванням нових даних, зокрема — підвищити поріг для зменшення кількості хибних спрацювань.

Невисокий рівень точності (precision) підсистеми пояснюється кількома чинниками, що мають як технічне, так і лінгвістичне підґрунтя:

- а) переклад частини корпусу з англійської мови;
- б) обмеження емоційного аналізу для української мови;
- в) складність реальних текстів і неочевидне емоційне забарвлення.

У зв'язку з обмеженою доступністю великих та збалансованих україномовних датасетів, частину навчального корпусу було сформовано шляхом машинного перекладу англомовних листів. Це могло вплинути на якість емоційного аналізу, оскільки автоматичний переклад не завжди зберігає тональні відтінки, емоційний контекст та інтенцію оригінального тексту. Як наслідок, модель могла хибно інтерпретувати деякі перекладені фрази.

На відміну від англійської, українська мова ще не має достатньо якісних і великих відкритих корпусів, які б дозволили створити глибоко натреновані моделі емоційного аналізу. Поточний інструмент (Azure Text Analytics)

підтримує українську мову, однак має низьку точність, що негативно впливає на класифікацію листів у підсистемі.

Деякі електронні повідомлення, формально нейтральні за змістом, можуть містити фрази, які модель інтерпретує як емоційно забарвлені або потенційно небезпечні (наприклад, використання емоційно заряджених слів у маркетингових розсилках). Це призводить до підвищеної кількості хибнопозитивних спрацювань (false positives), знижуючи загальний precision моделі.

У ході тестування підсистеми було розглянуто кілька варіантів порогу (threshold), що визначає межу між «нейтральним» та «шкідливим» листом. Було оцінено вплив порогу на метрики якості: точність (Precision), повноту (Recall), F1-міру, специфічність (Specificity) та загальну точність (Accuracy).

3.3 Графічний інтерфейс

Графічний інтерфейс підсистеми реалізовано у середовищі WPF (.NET/C#), що дозволяє відокремити логіку оброблення від подання і забезпечити надійне зв'язування даних. Вікно програми (див. рис. 3.1) складається з двох смислових областей: ліворуч користувач задає параметри аналізу та вводить текст, праворуч у реальному часі відображаються результати. Такий спосіб подання повторює робочий сценарій «Режим 1 — аналіз власного тексту», який раніше було реалізовано у Python. Графічний інтерфейс фактично дублює цей режим, звертається до тих самих сервісів і використовує ту саму послідовність обчислень, тож отримані висновки повністю співставні між консольною й GUI-версіями.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				53
Змн.	Арк.	№ докум.	Підпис	Дата		

Проект підсистеми виявлення соціальної інженерії в електронних повідомленнях

Поріг оцінки ризику:

Текст листа:

Виробник Toshiba відремонтовані ноутбуки ToshibaFactory RefurbishedNoteBooks \"Угоди тижня\" (див. Нижче Тошіба) РЕХІДНИ ЗАБЕЗПЕЧЕННЯ! Навіщо вагатися з найкращим значенням, отримуючи саме таке ж обладнання, з оригінальною гарантією виробників за значно нижчою ціною. Гарантія повернення грошей, якщо ви не задоволені - дійсні лише протягом 7 днів. 3: Що вважається фабрикою відремонтовано? Відповідь: Є багато речей, які можуть позначити фабрики фабрики. Деякі настільки легковажні, його смішно. Але оскільки споживчі закони заважають виробникам продати що -небудь, що вважається фабрикою, відремонтованим як \"новий\", другий споживач отримує найкращу угоду. Ось кілька прикладів: Виробники / дистриб'ютори не дозволяють експортувати за межами своєї території / країни. Щоб уникнути \"тривожного\" каналу дилера / реселлера в різних країнах, виробники та дистриб'ютори не можуть експортувати продукт, як новий, а отже, позначити його як ремонт та зменшення гарантії. Більшість великих роздрібних супермаркетів пропонують 30 - денну гарантію повернення грошей на свою продукцію, і є ті споживачі, які просто повертають гроші. Ці виробники

Загальна тональність листу:

Оцінка впевненості:

Позитивна: Нейтральна: Негативна:

Тональність для кожного речення:

- Виробник Toshiba відремонтовані ноутбуки ToshibaFactory RefurbishedNoteBooks \"Угоди тижня\" (див. Нижче Тошіба) РЕХІДНИ ЗАБЕЗПЕЧЕННЯ!
Тональність: neutral (Позитивна: 0,01, Нейтральна: 0,97, Негативна: 0,02)

Високий негатив у реченнях:

Високий позитив у реченнях:

Ключові фрази:

сервісному рівні
інтерфейс
результатами
запит

Оцінка ризику:

Класифікація:

Приклад 1 Приклад 2 Приклад 3 Приклад 4

Рисунок 3.3.1 Графічний інтерфейс

У верхній частині лівої панелі розташовано поле для задання порога прийняття рішення і кнопка запуску оброблення . Нижче розміщено багаторядкове поле введення, куди користувач вставляє або друкує текст листа. Для демонстрацій та відтворюваності експериментів додано кілька кнопок із заздалегідь підготовленими прикладами. Після натискання «Почати аналіз» застосунок передає введений фрагмент до сервісу Azure AI Language, отримує узагальнену тональність, показники впевненості за позитивною, нейтральною та негативною складовими, а також оцінки для кожного речення. На основі цих даних підсистема видокремлює ключові фрази, обчислює ризик за формулою, наведеною у розділі 2.4, і зіставляє отримане значення з порогом, формуючи фінальне рішення щодо класу листа.

Права панель слугує «інформаційною панеллю» результатів: у верхній частині відображається загальна тональність повідомлення і показники. Нижче подано зведення по реченнях, що дозволяє швидко локалізувати проблемні фрагменти, а також індикатори наявності високого негативу чи високого позитиву. Окремо відображаються виявлені ключові фрази, числове значення ризику та підсумкова класифікація («шкідливий» / «нейтральний»).

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Висновки до розділу 3

У третьому розділі протестовано інтерфейс командного рядка (CLI) та графічний інтерфейс розробленої підсистеми виявлення соціальної інженерії в електронних повідомленнях, а також проведено аналіз ефективності її роботи на основі експериментальних даних.

Виконано такі тестування:

- зміна порогу ризику та тестування двох режимів роботи;
- режим індивідуального аналізу листа, що передбачає обчислення емоційної тональності, ризикового балу та глибокий реченнєвий розбір із виявленням ключових фраз;
- режим обробки бази електронних листів, з автоматичним виводом метрик якості;
- експериментальне тестування підсистеми на корпусі з 1134 електронних листів та з використанням трьох різних значень порогу (0.3, 0.4, 0.5);
- Протестовано графічний інтерфейс, який дублює «Режим 1 — аналіз власного тексту» з рішення Python.

Побудовано графіки залежностей ключових метрик (Recall, Specificity) від порогового значення та на їх основі обґрунтовано вибір оптимального значення порогу для поточної бази даних. Результати тестування засвідчили, що найбільш ефективним компромісом між виявленням шкідливих листів та мінімізацією хибнопозитивних спрацювань є поріг 0.4. Низький рівень точності підсистеми частково пояснюється складністю емоційного аналізу для української мови, недоліками автоматичного перекладу англомовного корпусу та лінгвістичними особливостями реальних текстів.

Загалом реалізована підсистема продемонструвала працездатність, гнучкість у налаштуванні та здатність до подальшого масштабування й адаптації для виявлення ознак соціальної інженерії в електронних листах.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У межах кваліфікаційної роботи було реалізовано повний цикл створення підсистеми, призначеної для автоматичного виявлення атак соціальної інженерії в електронних листах, із застосуванням емоційного аналізу тексту.

У ході дослідження створено програмний засіб для емоційного аналізу тексту електронних листів українською мовою та виявлення потенційно шкідливих повідомлень. Рішення дозволяє аналізувати текст повідомлень та ідентифікувати потенційно шкідливі електронні повідомлення на основі емоційного профілю тексту.

Відповідно до визначених завдань:

- проведено аналіз сучасного стану проблеми фішингу та соціальної інженерії, обґрунтовано актуальність теми та перспективність застосування емоційного аналізу в задачах кібербезпеки;
- розглянуто існуючі рішення в галузі виявлення шкідливих листів, охарактеризовано їх переваги та недоліки;
- розроблено прототип підсистеми на основі Azure Language з використанням Python та REST API;
- сформовано та розмічено навчальний корпус електронних листів, що включає україномовні та перекладені англomовні зразки (1134 приклади).
- реалізовано алгоритм емоційного аналізу з використанням Azure Text Analytics, обробку та стандартизацію вхідних даних, а також їх класифікацію з динамічним порогом ризику;
- проведено експериментальне тестування з визначеними метриками;
- створено графічний інтерфейс.

Розроблена підсистема продемонструвала ефективність застосування емоційного аналізу в процесі кіберзахисту від атак соціальної інженерії в електронних повідомленнях. Її можливо інтегрувати у більші платформи обробки електронної пошти або використовувати як допоміжний модуль для підвищення рівня інформаційної безпеки.

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ерліхман І. Сучасна кібербезпека: методи виявлення фішингу та соціальної інженерії. — Київ: ВЦ «Академія», 2020.
2. Харченко В.С. Інформаційна безпека: сучасні виклики, рішення і практики захисту. — Харків: ФОП "Март", 2021.
3. Коваленко А. А. Емоційний аналіз тексту: теорія, практика, алгоритми. — Київ: НаУКМА, 2022.
4. Bird S., Klein E., Loper E. Natural Language Processing with Python. — O'Reilly Media, 2009.
5. Chollet F. Deep Learning with Python. — 2nd ed. — Manning Publications, 2021.
6. Powers, D. M. W. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. JMLT, 2011
7. Microsoft Azure. Text Analytics documentation [Електронний ресурс]. — Режим доступу: <https://learn.microsoft.com/en-us/azure/ai-services/language-service/sentiment-opinion-mining/overview>
8. ControlD. Phishing Statistics and Industry Trends: 2024 Report. [Електронний ресурс]. — Режим доступу: <https://controld.com/blog/phishing-statistics-industry-trends>
9. Cybersecurity & Infrastructure Security Agency (CISA). *Phishing Infographic* 2023. [Електронний ресурс]. — Режим доступу: <https://www.cisa.gov/sites/default/files/2023-02/phishing-infographic-508c.pdf>
10. WebAsha. AI-Powered Phishing Detection: How Machine Learning Is Transforming Cybersecurity. [Електронний ресурс]. — Режим доступу: <https://www.webasha.com/blog/ai-powered-phishing-detection>
11. Proofpoint. Protect Against Phishing. [Електронний ресурс]. — Режим доступу: <https://www.proofpoint.com/us/solutions/protect-against-phishing>
12. Barracuda. Email Protection: Anti-Phishing, Anti-Spam and Threat Detection. [Електронний ресурс]. — Режим доступу: <https://www.barracuda.com/products/email-protection/features>

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				57
Змн.	Арк.	№ докум.	Підпис	Дата		

13. Ali, N., Al-Hadhrami, T., Al Shidhani, A. Effective Phishing Emails Detection Method. // ResearchGate, 2021. [Электронный ресурс]. — Режим доступа: https://www.researchgate.net/publication/356880064_Effective_Phishing_Emails_Detection_Method
14. Rawat, M. et al. Detecting Social Engineering Attacks Using Machine Learning Techniques. // Springer Proceedings in Intelligent Systems, 2024. — Режим доступа: https://link.springer.com/chapter/10.1007/978-981-96-4245-8_18
15. Zeng, Y., & Wang, M. An Email Filtering System Based on NLP and Emotion Analysis. // Open Journal of Social Sciences, 2024. [Электронный ресурс]. — Режим доступа: <https://www.scirp.org/journal/paperinformation?paperid=135703>
16. TheCloudBlog. Analyzing Sentiment of Your Emails with Azure Text Analytics Service. [Электронный ресурс]. — Режим доступа: <https://thecloudblog.net/post/analyzing-sentiment-of-your-emails-with-azure-text-analytics-service>
17. Deep-Translator. Documentation [Электронный ресурс]. — Режим доступа: <https://deep-translator.readthedocs.io/en/latest/README.html>

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

Додаток А

Файл Main.py

```
import json
import os
import sys
import random
from azure.core.credentials import AzureKeyCredential
from azure.ai.textanalytics import TextAnalyticsClient
ENDPOINT = "https://resource-name.cognitiveservices.azure.com"
KEY = "<key>"
client = TextAnalyticsClient(endpoint=ENDPOINT,
credential=AzureKeyCredential(KEY))
DEFAULT_THRESHOLD = 0.5
MAX_LEN = 5120
def trim_text(text: str) -> str:
    return text if len(text) <= MAX_LEN else text[:MAX_LEN]
def analyze_and_classify(text: str, threshold: float) -> dict:
    doc = trim_text(text)
    resp = client.analyze_sentiment([doc], language="uk")[0]
    if hasattr(resp, 'error'):
        resp = client.analyze_sentiment([doc], language="uk")[0]
    sentiment = resp.sentiment
    pos = resp.confidence_scores.positive
    neg = resp.confidence_scores.negative
    neu = resp.confidence_scores.neutral
    risk_score = min(neg * 0.7 + pos * 0.3, 1.0)
    label = "шкідливий" if risk_score > threshold else "нейтральний"
    sentences_info = []
    for sent in resp.sentences:
        sentences_info.append({
            'text': sent.text,
            'sentiment': sent.sentiment,
            'positive': round(sent.confidence_scores.positive, 2),
            'neutral': round(sent.confidence_scores.neutral, 2),
            'negative': round(sent.confidence_scores.negative, 2)
        })
    return {
        'sentiment': sentiment,
        'positive': round(pos, 2),
        'neutral': round(neu, 2),
        'negative': round(neg, 2),
        'risk_score': round(risk_score, 2),
        'classification': label,
        'sentences': sentences_info
    }

def extract_key_phrases(text: str) -> list:
    """Повертає список ключових фраз з Azure Text Analytics."""
    doc = trim_text(text)
    response = client.extract_key_phrases([doc], language="uk")[0]
    if hasattr(response, "key_phrases"):
        return response.key_phrases
    return []

def get_true_label_by_id(doc_id, neutral_docs, malicious_docs):
    """Повертає 'нейтральний' або 'шкідливий' для даного id, шукаючи у
двох all.json"""
```

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

```

for doc in neutral_docs:
    if str(doc.get("id")) == str(doc_id):
        value = doc.get("label", 0)
        return "нейтральний" if value == 0 else "шкідливий"
for doc in malicious_docs:
    if str(doc.get("id")) == str(doc_id):
        value = doc.get("label", 0)
        return "нейтральний" if value == 0 else "шкідливий"
return "нейтральний" # якщо не знайдено
def main():
    thr = input(f"Вкажіть поріг оцінки ризику (THRESHOLD), наприклад 0.5
(за замовчуванням {DEFAULT_THRESHOLD}): ")
    try:
        threshold = float(thr) if thr.strip() else DEFAULT_THRESHOLD
    except ValueError:
        print("Невірний формат порогу, використовую значення за
замовчуванням.")
        threshold = DEFAULT_THRESHOLD

    mode = input("Оберіть режим (1 – Свій текст, 2 – Аналіз бази): ")
    if mode.strip() == '1':
        text = input(
            "Вкажіть порогове значення ризику (THRESHOLD, від 0 до 1,
наприклад 0.5).\n"
            "Якщо ризик листа (risk_score) буде БІЛЬШЕ цього порогу –
лист вважатиметься ШКІДЛИВИМ (класифікація: \"шкідливий\").\n"
            "Якщо ризик буде МЕНШЕ або дорівнює пороговому значенню –
лист вважатиметься НЕЙТРАЛЬНИМ.\n"
            f"За замовчуванням: {DEFAULT_THRESHOLD}\n"
            "Введіть значення або натисніть Enter для використання
значення за замовчуванням: "
        )
        result = analyze_and_classify(text, threshold)
        print(f"\nЗагальна тональність: {result['sentiment']}")
        print(f"Оцінка впевненості: Позитивна: {result['positive']},
Нейтральна: {result['neutral']}, Негативна: {result['negative']}")
        print(f"Оцінка ризику: {result['risk_score']}")
        print(f"Класифікація: {result['classification']}")

        print("\nТональність для кожного речення:")
        neg_indices = []
        pos_indices = []
        for idx, sent in enumerate(result['sentences'], start=1):
            print(f"- \"{sent['text']}\"")
            print(f"    Тональність: {sent['sentiment']} (Позитивна:
{sent['positive']}, Нейтральна: {sent['neutral']}, Негативна:
{sent['negative']})")
            if sent['negative'] >= 0.7:
                neg_indices.append(idx)
            if sent['positive'] >= 0.7:
                pos_indices.append(idx)
        if neg_indices:
            print(f"\nВисокий негатив у реченнях: {'', '.join(map(str,
neg_indices))}")
        if pos_indices:
            print(f"\nВисокий позитив у реченнях: {'', '.join(map(str,
pos_indices))}")

```

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

```

key_phrases = extract_key_phrases(text)
print("\nКлючові фрази:")
if key_phrases:
    for phrase in key_phrases:
        print(f"- {phrase}")
else:
    print("Не знайдено")
else:
    neu_path = os.path.join('data', 'Neutral', 'all.json')
    mal_path = os.path.join('data', 'Malicious', 'all.json')
    neutral_docs = json.load(open(neu_path, encoding='utf-8'))
    malicious_docs = json.load(open(mal_path, encoding='utf-8'))
    total_base = len(neutral_docs) + len(malicious_docs)
    print(f"Всього листів у базі: {total_base}
(нейтральних={len(neutral_docs)}, шкідливих={len(malicious_docs)})")
    print(f"Поточний поріг оцінки ризику: {threshold}\n")
    mix_path = os.path.join('data', 'Mixed', 'mixed.json')
    docs = json.load(open(mix_path, encoding='utf-8'))
    choice = input("Скільки листів обробити? Введіть 'all' або число:
")

    if choice.strip().lower() != 'all':
        try:
            count = int(choice)
            random.shuffle(docs)
            docs = docs[:count]
        except ValueError:
            print("Невірний формат, будуть оброблені всі листи.")
    n = len(docs)
    # Додаємо вивід про кількість листів кожного типу у вибірці
    num_neutral = 0
    num_malicious = 0
    for d in docs:
        doc_id = d.get('id')
        label = get_true_label_by_id(doc_id, neutral_docs,
malicious_docs)
        if label == 'нейтральний':
            num_neutral += 1
        elif label == 'шкідливий':
            num_malicious += 1
    print(f"У вибраній вибірці {n} листів:")
    print(f" нейтральних: {num_neutral}")
    print(f" шкідливих: {num_malicious}\n")
    tp = fp = tn = fn = 0
    for idx, d in enumerate(docs, start=1):
        res = analyze_and_classify(d['text'], threshold)
        pred = res['classification']
        doc_id = d.get('id')
        true = get_true_label_by_id(doc_id, neutral_docs,
malicious_docs)
        if pred == "шкідливий" and true == "шкідливий":
            tp += 1
        elif pred == "шкідливий" and true == "нейтральний":
            fp += 1
        elif pred == "нейтральний" and true == "нейтральний":
            tn += 1
        elif pred == "нейтральний" and true == "шкідливий":
            fn += 1
    percent = (idx / n) * 100

```

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        sys.stdout.write(f"\rОброблено: {idx}/{n} ({percent:.1f}%)")
        sys.stdout.flush()
    print("\n")
    precision = tp/(tp+fp) if tp+fp>0 else 0
    recall    = tp/(tp+fn) if tp+fn>0 else 0
    f1        = 2*precision*recall/(precision+recall) if
precision+recall>0 else 0
    specificity = tn / (tn + fp) if (tn + fp) > 0 else 0
    accuracy = (tp + tn) / (tp + fp + tn + fn) if (tp + fp + tn + fn)
> 0 else 0
    print(f"TP (True Positive) шкідливі листи, які система правильно
класифікувала як шкідливі: {tp}")
    print(f"FP (False Positive) нейтральні листи, які система
помилково класифікувала як шкідливі: {fp}")
    print(f"TN (True Negative) нейтральні листи, які система
правильно класифікувала як нейтральні: {tn}")
    print(f"FN (False Negative) шкідливі листи, які система помилково
класифікувала як нейтральні: {fn}")
    print(f"Точність (Precision): {precision:.2f}")
    print(f"Повнота (Recall): {recall:.2f}")
    print(f"F1-міра: {f1:.2f}")
    print(f"Специфічність (Specificity): {specificity:.2f}")
    print(f"Загальна точність (Accuracy): {accuracy:.2f}")

if __name__ == '__main__':
    main()

```

Файл clearHtml.py

```

import json
from bs4 import BeautifulSoup
def remove_html(text):
    return BeautifulSoup(text, "html.parser").get_text()
with open("all.json", "r", encoding="utf-8") as f:
    data = json.load(f)
def clean_json(obj):
    if isinstance(obj, dict):
        return {k: clean_json(v) for k, v in obj.items()}
    elif isinstance(obj, list):
        return [clean_json(elem) for elem in obj]
    elif isinstance(obj, str):
        return remove_html(obj)
    else:
        return obj
cleaned_data = clean_json(data)
with open("all_nohtml.json", "w", encoding="utf-8") as f:
    json.dump(cleaned_data, f, ensure_ascii=False, indent=2)

```

Файл Folder.py

```

import json
import os
import random
input_file = "all_nohtml_ukr.json"
neutral_dir = "Neutral"
malicious_dir = "Malicious"
mixed_dir = "Mixed"
os.makedirs(neutral_dir, exist_ok=True)
os.makedirs(malicious_dir, exist_ok=True)
os.makedirs(mixed_dir, exist_ok=True)
with open(input_file, "r", encoding="utf-8") as f:

```

		Ахмедов Т.Б.О.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        data = json.load(f)
    neutral = []
    malicious = []
    unique_id = 1 # Починаємо з 1
    for item in data:
        mail = item.copy()
        mail["id"] = unique_id
        label = mail.get("label")
        if label == 0:
            neutral.append(mail)
        else:
            malicious.append(mail)
        unique_id += 1
    with open(os.path.join(neutral_dir, "all.json"), "w", encoding="utf-8")
    as f:
        json.dump(neutral, f, ensure_ascii=False, indent=2)
    with open(os.path.join(malicious_dir, "all.json"), "w", encoding="utf-8")
    as f:
        json.dump(malicious, f, ensure_ascii=False, indent=2)
    mixed = neutral + malicious
    random.shuffle(mixed)
    mixed_no_label = []
    for item in mixed:
        item_no_label = item.copy()
        if "label" in item_no_label:
            del item_no_label["label"]
        mixed_no_label.append(item_no_label)
    with open(os.path.join(mixed_dir, "mixed.json"), "w", encoding="utf-8")
    as f:
        json.dump(mixed_no_label, f, ensure_ascii=False, indent=2)
    print("Готово! Додано id у всі листи, id однаковий у всіх файлах для
    кожного листа.")

```

Файл Translate.py

```

import json
from deep_translator import GoogleTranslator
import time
def split_text(text, chunk_size=4500):
    return [text[i:i+chunk_size] for i in range(0, len(text),
    chunk_size)]
def translate_text(text):
    if 0 < len(text) <= 4500:
        try:
            return GoogleTranslator(source='auto',
    target='uk').translate(text)
        except Exception as e:
            print("Помилка перекладу:", e, "Текст:", text[:100])
            return text
    elif len(text) > 4500:
        print(f"Ділимо довгий текст на шматки по {4500} символів
    (довжина: {len(text)})")
        parts = split_text(text, 4500)
        translated_parts = []
        for part in parts:
            try:
                translated_parts.append(GoogleTranslator(source='auto',
    target='uk').translate(part))
                time.sleep(1) # Google любить паузу!
            except Exception as e:

```

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				64
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        print("Помилка в шматку:", e, part[:100])
        translated_parts.append(part)
    return ''.join(translated_parts)
else:
    return text
def translate_json(obj):
    if isinstance(obj, dict):
        return {k: translate_text(v) if k == "text" and isinstance(v,
str) else translate_json(v) for k, v in obj.items()}
    elif isinstance(obj, list):
        return [translate_json(item) for item in obj]
    else:
        return obj
with open("all_nohtml.json", "r", encoding="utf-8") as f:
    data = json.load(f)
translated_data = translate_json(data)
with open("all_nohtml_ukr.json", "w", encoding="utf-8") as f:
    json.dump(translated_data, f, ensure_ascii=False, indent=2

```

		Ахмедов Т.Б.о.			ККІК<.КР.Б – 125 – 2 – ПЗ	Арк.
		Бродський Ю.Б.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

