МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

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

КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка чат-боту для опитування здобувачів вищої освіти

Здобувача вищої освіти першого (бакалаврського) рівня,
IV курсу, групи КБ-21-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології (за галузями)»


(підпис)

Міщука Євгенія Миколайовича
(прізвище, ім'я, по батькові)

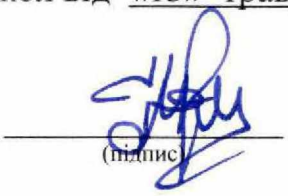
Науковий керівник 
(підпис)

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

Керівник освітньо-професійної програми 
(підпис)

Омельчук Антон Анатолійович, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри комп'ютерних та інформаційних технологій і систем протокол від «13» травня 2025 № 18

Завідувач кафедри 
(підпис)

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

Ірпінь – 2025

Реєстрація № 56 26.05.2025 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 26.05.2025 _____
(до захисту, на доопрацювання) (дата) (підпис)

Результат захисту 5 (відмінно) 04.06.2025 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ В.М. Франчук
(підпис)

Члени комісії: _____ М.М. Філоненко
(підпис)

_____ С.М. Грищенко
(підпис)

_____ А.А. Омельчук
(підпис)

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

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

Оцінювання кваліфікаційної роботи науковим керівником - 43 балів.

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

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
90	A	5(відмінно)	04.06.2025	Франчук В.М. Філоненко М.М. Грищенко С.М. Омельчук А.А. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

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

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітня програма: «Інформаційні управляючі системи і технології (за галузями)»

Затверджую
Завідувач кафедри _____ ФілоненкоМ.М.
«14» листопада 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Міщуку Євгенію Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи розробка чат-боту для опитування здобувачів вищої освіти
Затверджена наказом ректора від «13» листопада 2024 № 1717
2. Строк здачі закінченої роботи «08» травня 2025
3. Мета роботи автоматизація збору зворотного зв'язку від здобувачів вищої освіти.
Об'єкт дослідження процес автоматизації збору зворотного зв'язку від здобувачів вищої освіти за допомогою сучасних інформаційних технологій.
Предмет дослідження методи та засоби розробки інтерактивного чат-бота для проведення опитувань студентів, зокрема технології обробки даних, сценарії взаємодії користувача з ботом і підходи до аналізу отриманих результатів.

Орієнтовна база дослідження середовище тестування Telegram-бота на основі штучно змодельованої бази даних та сценаріїв опитувань.
4. Вихідні дані до роботи функціональні вимоги до чат-бота, Telegram Bot API, результати анкетування та тестування.
5. Перелік графічного матеріалу Діаграми варіантів використання, архітектура системи, інтерфейс бота, графіки статистики.
6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
розглянути сучасні підходи до реалізації чат-ботів для опитування; проаналізувати наявні інструменти та технічні рішення у цій сфері; здійснити проєктування та реалізацію функціональних модулів програмного забезпечення; провести оцінювання його функціональності, якості та економічної доцільності застосування

1. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційноїроботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	03.11.2024	22.10.2024
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	27.12.2024	15.11.2024
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	21.02.2025	26.01.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	16.04.2025	13.02.2025
5	**Висновки**	01.05.2025	11.03.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	09.05.2025	08.05.2025
7	Отримання відгуку керівника	27.05.2025	27.05.2025
8	Отримання зовнішньої рецензії	29.05.2025	29.05.2025

2. Дата видачі завдання «14» листопада 2024

3. Керівник кваліфікаційної роботи _____ (підпис) Філоненко М (прізвище, ініціали)

4. Керівник освітньо-професійної програми _____ (підпис) Омельчук А. (прізвище, ініціали)

5. Завдання до виконання отримав здобувач вищої освіти _____ (підпис) Міщук Є.М. (прізвище, ініціали)

АНОТАЦІЯ

Тема роботи: **розробка чат-бота для опитування здобувачів вищої освіти**

Кваліфікаційна робота присвячена дослідженню та розробці чат-бота для автоматизованого збору зворотного зв’язку в освітньому середовищі. Виконано аналіз сучасних методів опитування студентів, досліджено особливості функціонування чат-ботів та обґрунтовано вибір Telegram Bot API для реалізації системи. Проектування включає створення сценаріїв взаємодії, моделювання бізнес-процесів та розробку програмних модулів. Проведено тестування функціональності, оцінку продуктивності та техніко-економічне обґрунтування варіантів реалізації. Результати роботи можуть бути використані для вдосконалення автоматизованих систем опитування в навчальних закладах.

Мета роботи розробка інтерактивного чат-бота для автоматизації збору зворотного зв’язку від здобувачів вищої освіти, який забезпечує ефективний, гнучкий та зручний механізм проведення опитувань і подальшого аналізу отриманих даних.

Об’єкт дослідження процес автоматизації збору зворотного зв’язку від здобувачів вищої освіти за допомогою сучасних інформаційних технологій.

Предмет дослідження методи та засоби розробки інтерактивного чат-бота для проведення опитувань студентів.

Кваліфікаційна робота містить 102 сторінок, 40 рисунків, список використаних джерел 40 найменувань, 2 додатка.

Ключові слова: автоматизація, база даних, зворотний зв’язок, опитування, чат-бот, Telegram Bot API.

Рік виконання кваліфікаційної роботи 2025

Рік захисту роботи 2025

ABSTRACT

Thesis Topic: Development of a Chatbot for Surveying Higher Education Students

This qualification thesis is dedicated to the research and development of a chatbot for the automated collection of feedback in the educational environment. It presents an analysis of modern student survey methods, explores the features of chatbot functionality, and justifies the choice of the Telegram Bot API for system implementation. The design process includes the creation of interaction scenarios, business process modeling, and the development of software modules. Functionality testing, performance evaluation, and a technical and economic analysis of implementation options were conducted. The results of this work can be used to improve automated survey systems in educational institutions.

The aim of the thesis is to develop an interactive chatbot for automating the collection of feedback from higher education students, providing an efficient, flexible, and user-friendly mechanism for conducting surveys and analyzing the collected data.

Object of research: the process of automating feedback collection from higher education students using modern information technologies.

Subject of research: methods and tools for developing an interactive chatbot for conducting student surveys.

The qualification thesis consists of 102 pages, 40 figures, 40 references, and 2 appendices.

Keywords: automation, database, feedback, survey, chatbot, Telegram Bot API.

Year of thesis completion: 2025

Year of defense: 2025

ЗМІСТ

ВСТУП 4
РОЗДІЛ 1. АНАЛІЗ ТЕМАТИЧНОЇ СФЕРИ ТА ПОСТАНОВКА ЗАДАЧІ 8
1.1 Аналіз сучасних технологій опитування студентів 8
1.2 Види чат-ботів і особливості їх застосування 16
1.3 Аналіз чат-ботів для проведення опитувань 19
1.4 Постановка задачі 26
1.5 Висновок 27
РОЗДІЛ 2. ПРОЕКТУВАННЯ РОЗРОБКА ТА ТЕСТУВАННЯ ЧАТ-БОТУ ДЛЯ ОПИТУВАННЯ 29
2.1 Вибір платформи та інструментів для розробки 29
2.2 Сценарії взаємодії з користувачем, діаграми варіантів використання 32
2.3 Проєктування основних бізнес-процесів 36
2.4 Розробка основних модулів системи 40
2.5 Тестування функціоналу чат-боту 48
2.6 Висновок 57
РОЗДІЛ 3. АНАЛІЗ ФУНКЦІОНАЛЬНОСТІ ТА ВАРТОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 58
3.1 Визначення обґрунтованих функцій програмного забезпечення 58
3.2 Встановлення оптимальних параметрів програмного забезпечення 61
3.3 Експертний аналіз параметрів та їх оцінка 65
3.4 Оцінка рівня якості реалізації різних функціональних варіантів 69
3.5 Економічне обґрунтування варіантів розробки ПЗ 71
3.6 Обрання оптимального варіанту ПЗ на основі техніко-економічних показників 76
3.7 Висновок 77
ВИСНОВКИ 78

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 80
ДОДАТКИ 84
Додаток А. Скрипти бази даних 84
Додаток Б. Лістинги програми 87

ВСТУП

Сучасна освітня система все більше інтегрує цифрові технології для оптимізації навчального процесу, поліпшення взаємодії між студентами та викладачами, а також для моніторингу ефективності навчання. Одним із важливих напрямків є створення механізмів для оперативного збору й аналізу зворотного зв’язку від здобувачів освіти [1]. Зворотний зв'язок дозволяє отримувати важливі дані для вдосконалення навчальних програм, виявлення проблем та формування кращого освітнього середовища. Проте традиційні способи опитування студентів, такі як паперові анкети чи окремі онлайн-опитування, часто є затратними за часом, вимагають значних ресурсів для обробки даних і мають обмежену ефективність.

На сьогодні дедалі популярнішими стають автоматизовані системи опитування, які базуються на використанні сучасних інформаційних технологій, таких як чат-боти. Вони дозволяють реалізувати інтерактивні й масштабовані підходи до збору інформації, зменшуючи людський фактор і прискорюючи обробку результатів [2]. Чат-боти інтегрують зручні засоби комунікації, що особливо актуально для молодої аудиторії, яка активно використовує мобільні додатки та соціальні мережі.

Необхідність розробки автоматизованих рішень у сфері освітніх опитувань зумовлена кількома факторами. По-перше, це потреба забезпечення прозорості й об’єктивності в отриманні зворотного зв’язку. По-друге, оперативність збору та обробки даних сприяє швидшій адаптації освітнього процесу до потреб студентів. По-третє, автоматизовані системи створюють можливості для аналізу великих обсягів інформації, що важливо для прийняття стратегічних рішень керівництвом закладів освіти.

У міжнародній практиці вже є низка успішних рішень для автоматизації опитувань. Наприклад, системи типу SurveyMonkey чи Google Forms забезпечують широкий функціонал для створення опитувань, проте вони не враховують специфіку освітнього процесу, зокрема адаптацію до різних груп

студентів чи інтеграцію з внутрішніми базами даних навчальних закладів. Існують також спеціалізовані платформи для освітніх потреб, але їх функціонал часто обмежується певним регіоном або мовою, що знижує їхню застосовність для українських закладів освіти.

Критичний аналіз наявних рішень виявляє низку недоліків, таких як обмежена інтерактивність, низька гнучкість у налаштуванні сценаріїв опитувань та недостатній рівень локалізації для українських користувачів. З іншого боку, розробка власного рішення, яке буде враховувати специфічні потреби української освітньої системи, є перспективною та доцільною. Використання таких інструментів сприятиме підвищенню якості освіти, стимулюванню активності студентів та створенню більш демократичного освітнього середовища.

Україна, як держава з амбітними освітніми реформами, має значний потенціал для впровадження інновацій у систему управління якістю освіти. Водночас адаптація міжнародного досвіду до локальних реалій, включаючи мовну та культурну специфіку, є важливим аспектом для успішної реалізації таких рішень.

Об’єктом дослідження є процес автоматизації збору зворотного зв’язку від здобувачів вищої освіти за допомогою сучасних інформаційних технологій.

Предметом дослідження є методи та засоби розробки інтерактивного чат-бота для проведення опитувань студентів, зокрема технології обробки даних, сценарії взаємодії користувача з ботом і підходи до аналізу отриманих результатів.

Метою дослідження є розробка інтерактивного чат-бота для автоматизації збору зворотного зв’язку від здобувачів вищої освіти, який забезпечує ефективний, гнучкий та зручний механізм проведення опитувань і подальшого аналізу отриманих даних.

Задачі дослідження складаються із:

– аналізу сучасних технологій збору зворотного зв’язку в освітньому середовищі, зокрема методів опитування студентів, для виявлення їхніх переваг та недоліків;

– дослідження різновидів чат-ботів, їх можливостей та вибору оптимальної платформи для реалізації автоматизованого рішення для опитувань;

– розробки сценаріїв взаємодії користувачів із системою, проектування бізнес-процесів та створення основних модулів чат-бота з інтеграцією бази даних;

– тестування функціональності чат-бота, оцінки його зручності й продуктивності, а також здійснення експертного аналізу функцій і економічної оцінки розробленого рішення.

Для досягнення поставленої мети використовувалися такі методи:

– аналітичний метод для дослідження сучасних технологій опитувань та аналізу існуючих рішень у сфері автоматизації опитувань (розділ 1, п.1.1, п.1.3);

– методи системного аналізу для обґрунтування вибору платформи, проектування бізнес-процесів і моделювання взаємодії користувачів із системою (розділ 2, п.2.1, п.2.3);

– методи моделювання для створення сценаріїв взаємодії користувачів, побудови діаграм варіантів використання та логічної структури чат-бота (розділ 2, п.2.2);

– методи програмної інженерії для реалізації функціональних модулів чат-бота, розробки бази даних та інтеграції системи з Telegram API (розділ 2, п.2.4);

– експертний аналіз для оцінки функціональності, зручності використання та економічної ефективності розробленого рішення (розділ 3, п.3.3, п.3.5).

Наукова новизна даної роботи полягає у розробці інтерактивного чат-бота для опитування здобувачів вищої освіти, який поєднує автоматизацію збору зворотного зв’язку з інтеграцією бази даних для аналізу отриманих даних.

Вперше запропоновано методику створення чат-бота для освітньої сфери, яка враховує специфіку опитувань студентів, включаючи інтерактивні сценарії взаємодії та можливість оперативного оновлення запитань. Система забезпечує не лише збирання відповідей, але й генерацію статистичних звітів для вдосконалення навчального процесу.

Практичне значення одержаних результатів полягає у створенні інтерактивного чат-бота, який автоматизує процес збору зворотного зв’язку від здобувачів вищої освіти. Розроблене рішення дозволяє значно підвищити ефективність проведення опитувань завдяки інтеграції з платформою Telegram, що забезпечує зручність доступу для студентів. Бот надає можливість швидко отримувати відповіді на запитання, зберігати їх у базі даних та аналізувати результати для подальшого вдосконалення освітнього процесу.

Розроблений інструмент може бути впроваджений у закладах вищої освіти для регулярного моніторингу думок студентів щодо якості навчання, освітніх програм і інфраструктури. Це сприятиме підвищенню якості управлінських рішень, зокрема шляхом оперативного виявлення проблемних аспектів та визначення потреб студентів.

Система також має потенціал для адаптації в інших сферах, де потрібний збір даних або проведення анкетувань, таких як організація подій, опитування співробітників чи маркетингові дослідження. Застосування чат-бота в широкому спектрі завдань сприятиме вдосконаленню процесів прийняття рішень та підвищенню продуктивності роботи організацій.

РОЗДІЛ 1.
АНАЛІЗ ТЕМАТИЧНОЇ СФЕРИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз сучасних технологій опитування студентів

Збір зворотного зв’язку від студентів є важливим процесом, який дозволяє закладам вищої освіти отримувати об’єктивну інформацію про якість освітніх послуг та потреби студентів. Завдяки цьому можна своєчасно реагувати на виклики освітнього середовища, оптимізувати програми навчання, а також створювати комфортні умови для навчання. Такий підхід забезпечує більш ефективне управління освітнім процесом і сприяє підвищенню задоволеності студентів.

Сучасні підходи до організації опитувань студентів можна умовно поділити на дві категорії: традиційні та цифрові. Традиційні методи включають використання паперових анкет, усних опитувань і телефонних інтерв’ю, тоді як цифрові методи зосереджені на інтерактивних інструментах, таких як онлайн-анкети, чат-боти та інструменти, інтегровані в навчальні платформи. Кожен із цих підходів має свої особливості, які визначають їхнє застосування залежно від конкретних цілей і умов.

Дослідження існуючих методів збору зворотного зв’язку є необхідним для вибору оптимального рішення, яке дозволить забезпечити ефективність і зручність опитувань у сучасних умовах. У цьому підрозділі розглянемо основні технології, що застосовуються для збору даних від студентів, та їхню специфіку. Це дозволить глибше зрозуміти ключові аспекти впровадження таких рішень у контексті освітньої діяльності.

Паперові анкети – це інструмент для збору зворотного зв’язку, який передбачає використання друкованих форм із запитаннями, що заповнюються вручну [3]. У контексті опитувань студентів паперові анкети часто застосовуються для збору даних про їхнє задоволення освітнім процесом, якість викладання або умови навчання. Такий формат підходить для проведення

опитувань у середовищі, де студенти перебувають у безпосередньому контакті з організаторами, наприклад, у межах лекцій, семінарів чи інших заходів.

У табл. 1.1 наведено основні характеристики паперових анкет.

Таблиця 1.1.

Характеристики паперових анкет

Параметр	Опис	Приклад застосування	Особливості реалізації
Формат анкетування	Друкована форма, яку роздають студентам для заповнення вручну.	Поширення анкет на лекціях чи після навчальних занять.	Необхідність підготовки та друку анкет.
Тип даних	Відповіді, що можуть бути як текстовими, так і вибором із запропонованих варіантів.	Опитування щодо оцінки викладача або навчального курсу.	Передбачає наявність полів для коментарів і варіантів.
Обробка результатів	Відповіді збираються та аналізуються вручну.	Складання звіту за результатами наприкінці семестру.	Потребує часу та залучення відповідальних осіб.
Середовище проведення	Навчальні аудиторії чи інші фізичні місця, де зібрані студенти.	Збір анкет після освітнього заходу.	Важливо забезпечити комфортні умови для заповнення.

Паперові анкети застосовуються для опитувань у ситуаціях, коли важливо забезпечити доступність форми для всіх студентів, незалежно від рівня їхньої цифрової грамотності чи наявності електронних пристроїв. Вони зручні для використання в освітніх закладах під час організованих заходів, де всі учасники знаходяться фізично в одному місці. Такий підхід дозволяє організаторам отримувати відгуки від студентів у безпосередньому контакті.

Опитування є одним із найпоширеніших інструментів для збору даних, який дозволяє отримувати структуровану інформацію безпосередньо від респондентів [4]. У сфері освіти опитування широко застосовуються для вивчення думок студентів щодо навчальних програм, якості викладання, умов навчання та інших аспектів, пов’язаних із діяльністю навчальних закладів. Цей

метод передбачає формування серії запитань, на які учасники мають дати відповіді в заданій формі: вибором із запропонованих варіантів, відкритими текстовими відповідями або оцінкою за шкалою.

Опитування може проводитися як у традиційній, так і в цифровій формі, залежно від мети, технічних можливостей і потреб організаторів. Такий підхід забезпечує можливість охоплення великої кількості респондентів, що є особливо важливим для аналізу освітніх процесів у масштабах закладу. Основні характеристики цього методу представлено в табл. 1.2.

Таблиця 1.2.

Характеристики опитувань

Параметр	Опис	Приклад застосування	Особливості реалізації
Формат проведення	Паперовий або цифровий (онлайн-форми, мобільні додатки).	Онлайн-анкети для оцінки дисциплін.	Вибір формату залежить від технічних можливостей.
Структура запитань	Може включати відкриті, закриті або комбіновані запитання.	Питання щодо оцінки викладача, предмета або навчального процесу.	Складання запитань має відповідати меті опитування.
Цільова аудиторія	Студенти всіх рівнів та спеціальностей.	Анкетування групи студентів для оцінки якості занять.	Охоплення залежить від методів поширення анкети.
Частота проведення	Може бути одноразовим, періодичним або регулярним (щомісячно/ щосеместру).	Підсумкове анкетування наприкінці навчального року.	Планується заздалегідь відповідно до графіку навчання.
Результати	Дані, отримані в результаті опитування, використовуються для аналізу та прийняття рішень.	Аналіз задоволеності студентів навчальними програмами.	Результати можуть зберігатися у базі даних.

Опитування є гнучким та універсальним методом збору даних, який широко застосовується в освітній сфері. Воно дозволяє організаторам отримувати зворотний зв’язок безпосередньо від студентів, що є важливим для вдосконалення навчальних програм і поліпшення умов навчання. Проведення опитувань вимагає ретельної підготовки, включаючи складання запитань, визначення формату проведення та організацію обробки результатів. Такий підхід сприяє створенню інформаційної основи для стратегічного планування та прийняття рішень.

Телефонні інтерв’ю – це метод збору даних, який передбачає проведення опитування через телефонні дзвінки. У сфері освіти цей підхід використовується для отримання зворотного зв’язку від студентів щодо окремих аспектів навчального процесу, таких як якість викладання, організація занять чи рівень задоволеності навчальною інфраструктурою [5]. Телефонні інтерв’ю дають змогу безпосередньо взаємодіяти зі студентами, отримуючи більш розгорнуті відповіді на запитання, які потребують пояснень або уточнень [6].

Цей метод є особливо зручним у випадках, коли респондентам важко надати відповіді через інші канали, наприклад, через обмежений доступ до інтернету. Організація телефонних інтерв’ю вимагає наявності попередньо сформованого списку контактів, сценарію розмови та підготовленого персоналу для проведення опитувань. Характеристики цього методу наведено в таблиці 1.3.

Таблиця 1.3.

Характеристики телефонних інтерв’ю

Параметр	Опис	Приклад застосування	Особливості реалізації
1	2	3	4
Формат спілкування	Жива розмова між інтерв’юером і респондентом через телефонний дзвінок.	Дзвінки студентам для оцінки якості викладання дисципліни.	Вимагає навичок комунікації від інтерв’юера.

Продовження табл. 1.3.

1	2	3	4
Тип запитань	Відкриті та закриті запитання, часто з уточненнями від інтерв'юера.	Питання про загальну задоволеність навчальним процесом.	Можливість адаптувати питання в процесі розмови.
Цільова аудиторія	Студенти, які надали свої контактні дані для зв'язку.	Залучення випускників для оцінки довгострокових результатів навчання.	Необхідність попередньої згоди респондентів.
Час проведення	Узгоджується індивідуально з кожним респондентом.	Опитування студентів у позанавчальний час.	Потребує гнучкості у плануванні дзвінків.
Результати	Дані, отримані під час інтерв'ю, записуються вручну або за допомогою запису розмов.	Аналіз відгуків щодо роботи адміністрації закладу освіти.	Потребує додаткових ресурсів для обробки записів.

Телефонні інтерв'ю є важливим інструментом для збору якісного зворотного зв'язку, особливо у випадках, коли потрібно отримати розгорнуті відповіді чи уточнити деталі. Цей метод забезпечує живий контакт із респондентами, що сприяє кращому розумінню їхньої точки зору. Проведення телефонних інтерв'ю вимагає ретельної підготовки, включаючи створення сценарію розмови та підготовку персоналу, що дозволяє отримати максимально інформативні результати.

Розвиток інформаційних технологій значно змінив підходи до збору зворотного зв'язку, зробивши процес більш автоматизованим, зручним та економічно ефективним. Основними цифровими методами є: онлайн-анкети, чат-боти, соціальні мережі та інтерактивні опитування.

Онлайн-анкети – це сучасний інструмент для збору даних, який використовує цифрові платформи для створення та розповсюдження запитань серед респондентів [7]. У сфері освіти онлайн-анкети дозволяють студентам

швидко й зручно надати зворотний зв’язок щодо якості навчання, викладання чи інших аспектів освітнього процесу. Цей підхід ґрунтується на використанні спеціалізованих платформ, таких як Google Forms, Microsoft Forms, SurveyMonkey та інші, що забезпечують доступ до анкет через веббраузер або мобільні пристрої.

Такий формат опитувань особливо актуальний у сучасних умовах, коли значна частина освітнього процесу переведена в онлайн-формат. Онлайн-анкети дозволяють автоматизувати процес збору та обробки даних, а також охопити значну кількість респондентів у короткі терміни. Основні характеристики онлайн-анкет наведено в таблиці 1.4.

Таблиця 1.4.

Характеристики онлайн-анкет

Параметр	Опис	Приклад застосування	Особливості реалізації
Формат проведення	Запитання надаються у вебінтерфейсі або через мобільні додатки.	Онлайн-опитування студентів щодо задоволеності навчанням.	Потребує доступу до інтернету.
Тип запитань	Закриті, відкриті, комбіновані, з можливістю завантаження файлів.	Вибір варіанту відповіді або додавання коментарів.	Інтуїтивно зрозумілий інтерфейс важливий для залучення.
Цільова аудиторія	Студенти з доступом до цифрових пристроїв.	Опитування магістрантів щодо якості проведення лекцій.	Поширення посилань через пошту або месенджери.
Автоматизація	Відповіді автоматично збираються та обробляються.	Формування звітів для адміністрації закладу освіти.	Можливість експорту даних у різні формати.
Час проведення	Респонденти можуть заповнювати анкету у зручний для них час.	Анкетування протягом тижня після завершення семестру.	Необхідно встановлювати чіткі терміни.

Онлайн-анкети є гнучким і зручним методом для збору зворотного зв’язку, що відповідає вимогам сучасного освітнього середовища. Завдяки доступності цифрових платформ вони дозволяють охоплювати значну кількість респондентів та автоматизувати процес аналізу даних. Організація онлайн-анкетування вимагає попередньої підготовки, яка включає створення зрозумілих і структурованих запитань, а також вибір платформи, що найкраще відповідає потребам опитування. Це забезпечує інформативність та зручність для всіх учасників процесу.

Чат-боти – це програмні додатки, які імітують спілкування з користувачами через текстові чи голосові інтерфейси, використовуючи платформи обміну повідомленнями, такі як Telegram, Facebook Messenger або WhatsApp. У сфері освіти чат-боти все частіше застосовуються для проведення опитувань студентів, оскільки вони забезпечують інтерактивний та зручний формат взаємодії [8]. Студенти можуть відповідати на запитання в реальному часі, а бот автоматично зберігає та обробляє їхні відповіді.

Чат-боти дозволяють персоналізувати опитування, адаптувати їх до поведінки та потреб користувача, а також нагадувати про необхідність завершити опитування. Вони ідеально підходять для студентської аудиторії, яка активно використовує мобільні пристрої та месенджери. Основні характеристики використання чат-ботів для опитувань наведено в табл. 1.5.

Таблиця 1.5.

Характеристики чат-ботів для опитувань

Параметр	Опис	Приклад застосування	Особливості реалізації
1	2	3	4
Формат проведення	Інтерактивна текстова взаємодія через месенджери.	Опитування через Telegram-бот щодо якості викладання.	Необхідність підключення до платформи API.

Продовження табл. 1.5.

1	2	3	4
Цільова аудиторія	Студенти, які користуються мобільними пристроями та месенджерами.	Опитування першокурсників через Telegram після лекцій.	Важливо забезпечити простоту та інтуїтивність інтерфейсу.
Автоматизація	Відповіді збираються та обробляються автоматично в режимі реального часу.	Формування звітів для адміністрації закладу освіти.	Інтеграція з базою даних для збереження інформації.
Час проведення	Опитування може відбуватися будь-коли, за зручністю користувача.	Анкетування студентів протягом двох тижнів після іспитів.	Гнучке налаштування таймінгу нагадувань.

Чат-боти є сучасним та інтерактивним інструментом для збору зворотного зв’язку, який відмінно підходить для студентської аудиторії. Їхня здатність автоматично обробляти відповіді, надавати персоналізовані запитання та працювати в режимі реального часу робить їх зручним рішенням для закладів освіти. Впровадження чат-ботів у процес опитування сприяє підвищенню залученості студентів та забезпечує можливість отримувати дані без додаткових витрат часу на їх обробку вручну. Такий підхід відкриває нові можливості для покращення освітнього процесу.

Традиційні методи, хоча й прості у впровадженні, значно поступаються цифровим за ефективністю. Вони є затратними, повільними та менш гнучкими. Цифрові методи, особливо чат-боти, надають більшу інтерактивність, автоматизацію та можливість адаптації до потреб користувачів.

У контексті освітнього середовища важливим є створення рішень, які інтегрують функції збору зворотного зв’язку в повсякденні інструменти, що використовуються студентами. Чат-боти в цьому сенсі мають значний потенціал

завдяки своїй доступності, простоті інтеграції з базами даних і можливості гнучкої адаптації до змін у структурі опитувань.

Отже, аналіз сучасних технологій опитування показує, що найбільш перспективними є цифрові рішення, які забезпечують автоматизацію та інтерактивність. Використання чат-ботів у цій сфері створює нові можливості для збору та аналізу зворотного зв’язку, адаптуючи опитування до сучасних потреб студентів і закладів освіти.

1.2 Види чат-ботів і особливості їх застосування

Чат-боти є ефективним інструментом для автоматизації комунікацій у різних сферах, включаючи освіту. Їх популярність зумовлена здатністю забезпечувати швидку та зручну взаємодію з користувачами через текстові або голосові інтерфейси. Щоб визначити можливості чат-ботів у контексті освітніх опитувань нижче розглянуто основні їхні види та особливості застосування, зокрема:

1. Розважальні чат-боти

Розважальні чат-боти розроблені для створення цікавого та інтерактивного спілкування з користувачами. Вони часто використовуються для залучення уваги, ігор, вікторин або організації інтерактивних заходів [9]. У контексті опитувань студентів такі боти можуть бути інтегровані в процес навчання для організації неформальних анкетувань або розважальних опитувань, які стимулюють залученість студентів.

Основними особливостями застосування розважальних чат-ботів є:

– створення дружньої та невимушеної атмосфери серед студентів;

– залучення студентів до участі в опитуваннях за допомогою ігрових елементів;

– обмежена функціональність у зборі даних для детального аналізу.

2. Інформаційні чат-боти

Інформаційні чат-боти спеціалізуються на наданні користувачам швидкого доступу до необхідної інформації. Вони можуть надавати відповіді на поширені

запитання, нагадувати про події або забезпечувати доступ до навчальних матеріалів [10]. Для опитувань такі боти можуть використовуватися для інформування студентів про цілі опитування, методику його проведення та інші деталі.

Особливостями застосування інформаційних чат-ботів є:

– надання студентам актуальної інформації про процес опитування;

– формування рекомендацій щодо правильного заповнення анкет;

– використання як додаткового інструменту комунікації між студентами та організаторами опитувань.

3. Інтерактивні чат-боти

Інтерактивні чат-боти є найбільш підходящими для реалізації процесу опитувань. Вони дозволяють користувачам відповідати на запитання, вибирати варіанти відповідей і отримувати зворотний зв’язок у реальному часі. Такі боти зазвичай інтегруються з базами даних, що забезпечує автоматизацію збору та обробки даних.

Основні особливості застосування інтерактивних чат-ботів:

– забезпечення динамічної взаємодії з користувачами, що підвищує їхню залученість;

– підтримка інтеграції з базами даних для автоматизованого збору та обробки результатів;

– адаптація запитань відповідно до відповідей користувача для більш персоналізованого підходу.

4. Голосові чат-боти

Голосові чат-боти взаємодіють із користувачами за допомогою голосових команд. Вони особливо корисні для користувачів, які віддають перевагу голосовій взаємодії, наприклад, через обмежені можливості використання текстового інтерфейсу [11]. У контексті опитувань студентів такі боти можуть використовуватися для збору відгуків у неформальному середовищі.

Основними особливостями застосування голосових чат-ботів є:

– спрощення доступу до опитувань для людей із обмеженими можливостями;

– необхідність складнішої технологічної реалізації для забезпечення роботи з голосовими командами;

– здатність збирати відкриті відповіді, які можуть бути конвертовані в текст для подальшого аналізу.

Чат-боти, залежно від їх типу, пропонують широкий спектр функцій для виконання різних завдань. Вони можуть слугувати як для залучення уваги через інтерактиви, так і для надання структурованої інформації, організації опитувань або роботи з голосовими запитами. Кожен вид чат-ботів має свої специфічні особливості, які визначають доцільність їхнього застосування в різних контекстах. Для порівняння функціоналу і можливостей різних видів чат-ботів, наведено їхні основні характеристики в таблиці 1.6.

Таблиця 1.6.

Порівняння функціоналу та можливостей чат-ботів

Тип чат-бота	Функціонал	Приклад застосування	Особливості реалізації
Розважальні чат-боти	Організація інтерактивів, ігор та вікторин.	Проведення неформальних опитувань серед студентів.	Вимагають інтеграції елементів гейміфікації.
Інформаційні чат-боти	Надання інформації та відповіді на поширені запитання.	Інформування студентів про процес опитування.	Інтеграція з базами знань.
Інтерактивні чат-боти	Проведення опитувань із динамічними запитаннями.	Збір відповідей на навчальні анкети.	Автоматизація з використанням баз даних.
Голосові чат-боти	Збір даних через голосову взаємодію.	Проведення опитувань через голосові команди.	Потребують інтеграції технологій розпізнавання мови.

Отже, чат-боти є багатофункціональними інструментами, які можуть бути адаптовані до різних потреб у сфері освіти. Кожен із видів чат-ботів має свої особливості, які визначають його застосування залежно від мети опитування. Розробка інтерактивних чат-ботів для проведення опитувань є перспективним напрямом, оскільки вони забезпечують ефективну комунікацію, автоматизацію збору даних і зручність для студентів.

1.3 Аналіз чат-ботів для проведення опитувань

Чат-боти стають дедалі популярнішим інструментом для проведення опитувань завдяки своїй інтерактивності, доступності та автоматизації. У сфері освіти вони використовуються для збору зворотного зв’язку від студентів, що дозволяє оперативно отримувати дані про якість навчального процесу, викладання та інші аспекти освітньої діяльності. Їхня інтеграція в платформи обміну повідомленнями, такі як Telegram, Facebook Messenger або WhatsApp, робить опитування зручними та доступними для молодіжної аудиторії.

Основною перевагою використання чат-ботів у процесі опитування є можливість адаптації запитань до відповідей респондентів, що забезпечує більш персоналізований підхід і підвищує якість зібраних даних. Окрім того, автоматизація збору та обробки відповідей значно зменшує час і ресурси, необхідні для проведення опитувань традиційними методами.

Аналіз функціональності, технологічних особливостей та прикладів успішного застосування чат-ботів у контексті проведення опитувань дозволяє оцінити їхній потенціал в освітній сфері.

SurveyBot – це чат-бот, призначений для створення та проведення інтерактивних опитувань через Facebook Messenger (рис. 1.1). Його основна мета полягає у спрощенні процесу збору зворотного зв’язку шляхом інтеграції з платформою, якою активно користуються респонденти, зокрема студенти. SurveyBot дозволяє налаштовувати запитання, відстежувати відповіді в реальному часі та автоматизувати обробку даних [11].

Архітектура SurveyBot включає серверну частину, що відповідає за обробку даних, інтеграцію з API Facebook Messenger та базою даних для збереження відповідей респондентів [12]. Інтерфейс користувача реалізований у вигляді текстових повідомлень, що відображаються безпосередньо в месенджері, з підтримкою кнопок для вибору варіантів відповідей. Чат-бот використовує вебхуки для отримання повідомлень від користувачів та динамічного формування відповідей. База даних слугує для збереження питань, відповідей та статистичних звітів, що дозволяє організаторам опитувань ефективно аналізувати отримані результати. Така архітектура забезпечує зручність використання та високу швидкість роботи.

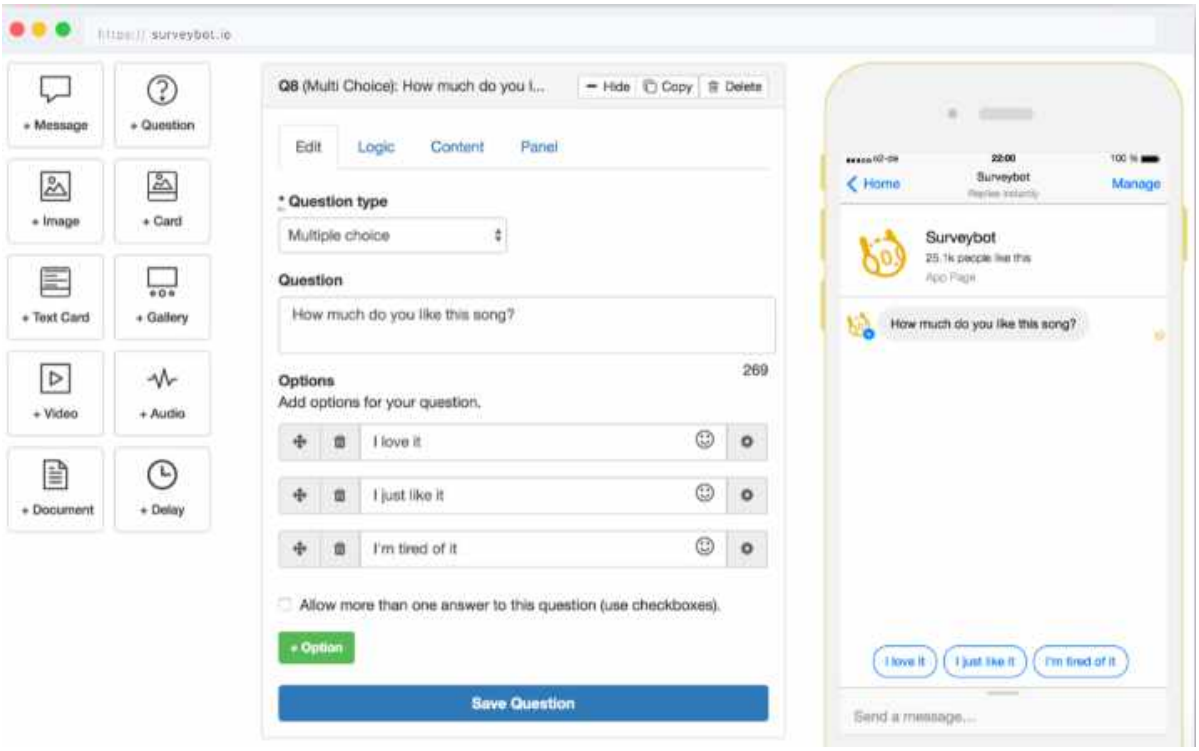


Рис. 1.1 Приклад використання «SurveyBot» [13]

До переваг SurveyBot відносяться:

– інтеграція з Facebook Messenger, що забезпечує доступність для широкої аудиторії [14];

– простота налаштування та запуску опитувань без необхідності технічних знань;

– автоматичний збір та обробка відповідей у реальному часі [15];

– підтримка інтерактивного інтерфейсу з можливістю додавання кнопок для відповідей.

Недоліки SurveyBot:

– обмежена функціональність для кастомізації складних опитувань;

– залежність від платформи Facebook Messenger, що зменшує охоплення користувачів інших месенджерів [16];

– відсутність можливості інтеграції з нестандартними базами даних без додаткових модифікацій [17];

– обмежена підтримка для роботи з відкритими текстовими відповідями.

SurveyBot є зручним і простим у використанні інструментом для організації опитувань, особливо в рамках платформ соціальних мереж. Він добре підходить для стандартних опитувань із закритими запитаннями, де важлива швидкість збору відповідей. Проте його обмеження у функціональності та залежність від Facebook Messenger можуть стати перешкодою для використання у складніших сценаріях або для більш широкої аудиторії.

SurveySparrow – це чат-бот, розроблений для проведення інтерактивних опитувань через різні платформи, включаючи месенджери, вебсайти та мобільні додатки (рис. 1.2) [18]. Його основне призначення – забезпечення зручного, персоналізованого та інтерактивного збору зворотного зв’язку. SurveySparrow підтримує опитування у вигляді розмови, що підвищує залученість респондентів і сприяє отриманню більш точних даних.

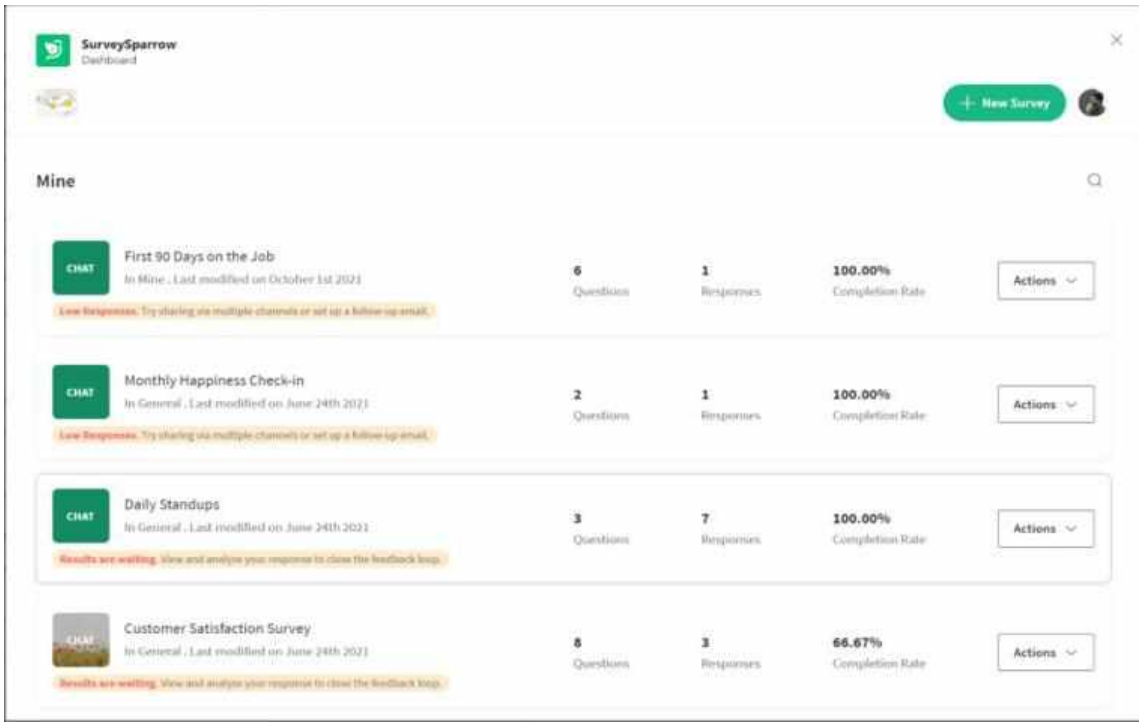


Рис. 1.2 Приклад використання «SurveySparrow» [19]

Архітектура SurveySparrow включає серверну частину, що обробляє запити та відповіді користувачів, а також модулі інтеграції з різними платформами через API [20]. Бот використовує центральну базу даних для збереження запитань, відповідей та аналітичних даних. Інтерфейс чат-бота забезпечує динамічну взаємодію з респондентами, адаптуючи наступні запитання залежно від їхніх відповідей. Для аналізу зібраних даних передбачено інструменти звітності та візуалізації, що інтегруються з іншими аналітичними платформами. Така архітектура робить SurveySparrow гнучким рішенням, придатним для різних сценаріїв застосування, включаючи освітню сферу.

Переваги SurveySparrow:

– підтримка опитувань у форматі розмови, що підвищує залученість респондентів [21];

– мультиплатформеність, яка дозволяє використовувати бот у месенджерах, на вебсайтах та в мобільних додатках [22];

– гнучкість у налаштуванні запитань і адаптація до відповідей респондентів у реальному часі;

– вбудовані аналітичні інструменти для створення звітів та інтеграції з іншими платформами [23].

Недоліки SurveySparrow:

– відносно висока вартість для використання розширеного функціоналу;

– обмеження для роботи з великими обсягами даних або складними сценаріями опитувань [24];

– залежність від стабільного інтернет-з’єднання для проведення опитувань у режимі реального часу [25];

– потреба в інтеграції додаткових модулів для специфічних потреб, що може ускладнити впровадження.

SurveySparrow є універсальним рішенням для проведення інтерактивних опитувань, яке відзначається зручністю у використанні та можливістю залучення користувачів через різні платформи. Його здатність адаптувати запитання до відповідей респондентів і наявність аналітичних інструментів роблять цей бот ідеальним для освітньої та бізнес-сфери. Однак для використання в масштабних проєктах або специфічних сценаріях може знадобитися додаткове налаштування та фінансові вкладення. Це робить SurveySparrow оптимальним вибором для середніх і невеликих проєктів із фокусом на персоналізації опитувань.

PollBot – це простий і зручний чат-бот, створений для організації швидких опитувань у месенджері Telegram (рис. 1.3). Його основне призначення полягає в оперативному зборі відповідей на прості запитання з попередньо визначеними варіантами [26]. Завдяки мінімалістичному дизайну та легкості використання, PollBot підходить для повсякденного застосування, включаючи освітні опитування, командні голосування чи збори думок під час подій.

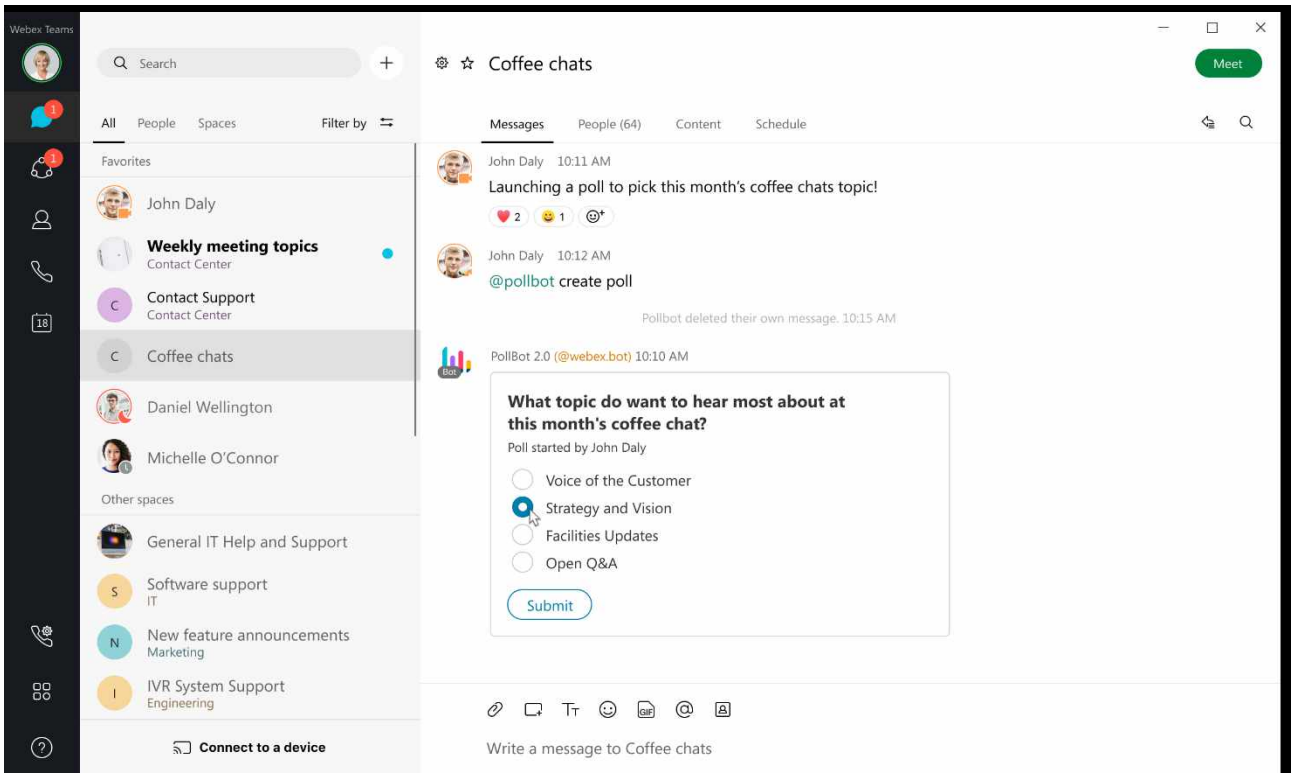


Рис. 1.3 Приклад використання «Pollbot» [27]

Архітектура PollBot орієнтована на швидкодію та простоту. Бот працює через Telegram Bot API, що забезпечує взаємодію з користувачами в месенджері. Серверна частина обробляє команди користувачів, створює опитування та зберігає результати. Дані про опитування зберігаються у базі даних, що дозволяє відслідковувати результати в реальному часі. Інтерфейс бота базується на текстових командах і кнопках, які полегшують створення та участь в опитуваннях. Така архітектура забезпечує простоту використання та стабільність роботи навіть при великій кількості учасників.

Переваги PollBot:

– простота у використанні завдяки текстовим командам і мінімалістичному інтерфейсу [28];

– швидке створення та проведення опитувань без складних налаштувань;

– інтеграція з Telegram, що забезпечує доступність для широкої аудиторії;

– можливість отримання результатів у реальному часі.

Недоліки PollBot:

– обмежений функціонал, який не підходить для складних або динамічних опитувань [29];

– відсутність інтеграції з базами даних для автоматичного аналізу результатів [30];

– орієнтація лише на Telegram, що звужує коло потенційних користувачів;

– обмежена підтримка кастомізації для спеціалізованих сценаріїв опитувань.

Отже, PollBot є чудовим рішенням для швидких і простих опитувань у Telegram. Його легкість використання та інтеграція з популярним месенджером роблять його популярним серед освітніх закладів, командних груп та інших організацій, які потребують оперативного збору відповідей. Проте обмежений функціонал і вузька спеціалізація можуть стати перешкодою для використання цього бота у масштабних або складних опитуваннях. PollBot найкраще підходить для невеликих, локальних ініціатив, які потребують простого інструмента для збору думок.

Проведений аналіз існуючих чат-ботів для опитувань, таких як SurveyBot, SurveySparrow і PollBot, виявив низку обмежень, які можна подолати шляхом розробки нового рішення. Основний акцент нового чат-бота буде зроблено на простішому функціоналі, що дозволить студентам легко взаємодіяти з ботом без необхідності спеціальних знань. Легкість впровадження досягається завдяки інтеграції з популярними платформами, такими як Telegram, використанню інтуїтивного інтерфейсу та готових шаблонів опитувань. Важливою перевагою є безкоштовна базова версія, що робить рішення доступним для освітніх установ з обмеженим бюджетом. Також нове рішення враховуватиме зручну організацію роботи з базами даних і автоматичну генерацію звітів, що вже частково реалізовано в наданому коді. Це дозволить створити універсальний і ефективний інструмент для збору зворотного зв’язку.

1.4 Постановка задачі

Для розробки системи керування чат-ботом і самого чат-бота для опитування студентів необхідно створити функціональне рішення, яке забезпечить інтерактивність, автоматизацію процесів та простоту в управлінні. Система повинна підтримувати дві ролі: адміністратор і користувач, кожна з яких матиме чітко визначений набір функцій. Адміністратор отримає доступ до інструментів управління ботом, базами даних, опитуваннями, статистикою та подіями, тоді як користувач зможе брати участь у опитуваннях, залишати відгуки та переглядати інформацію, надану ботом.

Функціонал програми повинен включати такі основні компоненти:

– запуск Telegram-бота. Користувач повинен мати змогу швидко налаштовувати та запускати чат-бота для взаємодії з користувачами через платформу Telegram;

– теми опитувань. Система повинна дозволяти користувачу створювати та керувати темами опитувань. Теми є структурною одиницею, яка об’єднує питання за спільною тематикою;

– запитання для кожної теми. Користувач повинен мати можливість додавати запитання для кожної теми, включаючи варіанти відповідей. Запитання можуть бути закритими (з фіксованими варіантами) або відкритими;

– статистика. Необхідно реалізувати збір і відображення статистичних даних. Система має надавати аналітичні звіти за темами, запитаннями та відгуками користувачів, а також загальну кількість проведених опитувань і зібраних відповідей;

– система управління даними. Адміністратор повинен мати доступ до управління базами даних, зокрема користувачами, подіями та результатами опитувань. Це дозволить забезпечити зручну інтеграцію та підтримку системи.

Чат-бот для користувачів повинен забезпечувати:

– вибір теми опитування та динамічне відображення відповідних запитань;

– проведення опитування до обраної теми з можливістю послідовного заповнення відповідей;

– можливість залишати текстові відгуки та оцінки;

– інтерактивну взаємодію з користувачем, включаючи відповіді на команди та навігацію в головному меню.

Система керування та сам чат-бот повинні бути розроблені з урахуванням масштабованості, щоб у майбутньому забезпечити можливість додавання нових функцій та інтеграції з іншими сервісами. Це дозволить ефективно використовувати рішення для підвищення якості освітнього процесу та оперативного реагування на потреби студентів.

1.5 Висновок

У рамках даного розділу проведено всебічний аналіз сучасних технологій опитування студентів, що включав як традиційні методи (паперові анкети, опитування, телефонні інтерв'ю), так і цифрові підходи (онлайн-анкети та чат-боти). Виявлено основні особливості, переваги й обмеження кожного з методів, що дозволило обґрунтувати доцільність використання цифрових рішень у сучасному освітньому середовищі.

Детально розглянуто види чат-ботів, серед яких виділено розважальні, інформаційні, інтерактивні та голосові боти. Проведений аналіз дав змогу визначити ключові функціональні особливості кожного типу та їхню відповідність завданням опитування студентів. Особливу увагу приділено інтерактивним чат-ботам, як найбільш ефективному рішенню для автоматизації процесу збору зворотного зв'язку.

Проаналізовано існуючі чат-боти для проведення опитувань, включаючи SurveyBot, SurveySparrow і PollBot. Визначено їх переваги, недоліки та обмеження, що дозволило обґрунтувати необхідність розробки нового рішення з більш простим функціоналом, легкістю впровадження та доступністю безкоштовної версії.

На основі проведеного аналізу сформульовано постановку задачі, яка охоплює розробку системи керування чат-ботом із двома ролями (адміністратор і користувач) та забезпеченням функціональності, необхідної для проведення опитувань, збору відгуків, аналізу статистичних даних і управління базами даних. Результати розгляду, отримані в цьому розділі, створюють основу для наступного етапу роботи, де будуть розроблені архітектура, програмна реалізація та проведено тестування нового чат-бота для опитувань.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ РОЗРОБКА ТА ТЕСТУВАННЯ ЧАТ-БОТУ ДЛЯ ОПИТУВАННЯ

2.1 Вибір платформи та інструментів для розробки

Розробка чат-бота для опитування студентів вимагає ретельного вибору платформи та інструментів, які забезпечать гнучкість, масштабованість і зручність у використанні. Правильний вибір платформи визначає не лише ефективність реалізації основного функціоналу, а й можливість подальшої інтеграції бота з іншими системами та сервісами, зокрема базами даних для збереження відповідей і статистичних звітів. У цьому підрозділі проаналізуємо три популярні платформи для створення чат-ботів: Telegram Bot API, Dialogflow і Chatfuel, порівняємо їх основні характеристики та обґрунтуємо вибір платформи, що використовується в рамках цієї роботи.

Telegram Bot API є одним із найпоширеніших інструментів для створення чат-ботів завдяки своїй простоті, безкоштовності та широким можливостям кастомізації [31]. Ця платформа дозволяє створювати боти з будь-якою логікою, використовуючи різні мови програмування, і забезпечує доступ до багатофункціонального API для інтеграції з базами даних, аналітичними системами та зовнішніми сервісами. Telegram Bot API має великий потенціал для налаштування, але вимагає певних навичок програмування для реалізації складних сценаріїв.

Dialogflow – це платформа, розроблена Google, яка спеціалізується на створенні чат-ботів із можливістю обробки природної мови (NLP). Вона дозволяє швидко створювати боти, які розуміють текстові й голосові запити, а також інтегрувати їх із популярними месенджерами [32]. Однак, для реалізації бота для освітніх опитувань Dialogflow може бути надлишковим через свою складність і акцент на NLP, що не є критично необхідним для реалізації завдань цієї теми. Крім того, деякі функції платформи є платними, що може бути обмеженням у контексті даного проєкту.

Chatfuel – це конструктор чат-ботів, орієнтований на створення рішень для Facebook Messenger [33]. Платформа відзначається простотою у використанні та дозволяє створювати боти без програмування, використовуючи блокову структуру. Однак Chatfuel має суттєві обмеження у функціональності та інтеграції з іншими сервісами, що робить його менш придатним для завдань, пов'язаних із гнучким налаштуванням і використанням в інших месенджерах, таких як Telegram.

Вибір платформи для розробки чат-бота залежить від низки характеристик, таких як гнучкість у налаштуванні, доступність для розробників, можливості інтеграції та вартість використання. Щоб порівняти Telegram Bot API, Dialogflow та Chatfuel, наведено основні характеристики кожної платформи в табл. 2.1.

Таблиця 2.1.

Порівняння платформ для розробки чат-ботів

Характеристика	Telegram Bot API	Dialogflow	Chatfuel
Гнучкість у налаштуванні	Висока: підтримка будь-якої логіки через API.	Середня: обмежена готовими модулями NLP.	Низька: створення бота через блокову структуру.
Підтримка інтеграцій	Широка: можливість роботи з базами даних, API.	Середня: інтеграції через Google сервіси.	Обмежена: інтеграція здебільшого з Messenger.
Вартість	Безкоштовно.	Безкоштовний базовий функціонал, є платні функції.	Обмежена безкоштовна версія, додаткові функції платні.
Платформи підтримки	Telegram.	Багато месенджерів (Telegram, WhatsApp, ін.).	Facebook Messenger.
Рівень програмування	Потребує знань програмування.	Мінімальний: доступні готові модулі.	Не потребує програмування.

Вибір Telegram Bot API для розробки даної системи обґрунтований його гнучкістю у налаштуванні, що дозволяє реалізувати будь-яку логіку чат-бота

через доступний API. Завдяки широким можливостям інтеграції Telegram Bot API забезпечує зручну роботу з базами даних, зовнішніми сервісами та інструментами для автоматизації збору й аналізу даних. Важливою перевагою є повна безкоштовність використання платформи, що робить її доступною для освітніх установ із обмеженим бюджетом. Окрім того, підтримка Telegram як популярного месенджера забезпечує зручність і доступність для студентів, що є критично важливим для ефективного проведення опитувань.

Вибір мови програмування залежить від багатьох факторів, включаючи її продуктивність, зручність у розробці, можливості інтеграції з API та підтримку масштабованості системи. В табл. 2.2 проведено порівняння мов програмування за їхніми ключовими характеристиками, які впливають на швидкість розробки, стабільність роботи та легкість інтеграції із Telegram Bot API.

Таблиця 2.2.

Порівняння мов програмування для розробки чат-ботів

Характеристика	Python	Java	C#
Продуктивність	Середня, залежить від реалізації [34]	Висока, завдяки JVM і оптимізації коду	Висока, завдяки екосистемі .NET
Можливості інтеграції	Багато бібліотек для роботи з API	Інтеграція через сторонні бібліотеки	Тісна інтеграція з API через .NET
Інструменти розробки	Простий набір інструментів, широкий вибір IDE	Широкий вибір професійних інструментів	Потужна підтримка в Visual Studio
Стабільність і масштабованість	Стабільна, підходить для середніх проєктів	Висока, підходить для великих систем	Висока, добре працює у великих системах
Продуктивність	Середня, залежить від реалізації	Висока, завдяки JVM і оптимізації коду [35]	Висока, завдяки екосистемі .NET та компіляції [36]
Можливості інтеграції	Широкі: багато бібліотек для роботи з API	Підтримує інтеграцію через сторонні бібліотеки	Тісна інтеграція з API через .NET

Вибір мови програмування C# для розробки системи обґрунтований її високою продуктивністю, що забезпечується компіляцією в середовищі .NET, а також стабільністю і масштабованістю для створення складних систем. Завдяки потужній інтеграції з Visual Studio C# забезпечує зручність розробки, що дозволяє швидко створювати, тестувати та розгортати чат-бота. Тісна інтеграція з Telegram Bot API через бібліотеки .NET дозволяє реалізувати весь необхідний функціонал з мінімальними витратами часу. Окрім того, використання C# забезпечує безкоштовний доступ до екосистеми .NET Core, що робить розробку економічно вигідною для освітніх проєктів. Завдяки цим перевагам C# є оптимальним вибором для реалізації даної системи.

2.2 Сценарії взаємодії з користувачем, діаграми варіантів використання

Взаємодія користувачів із системою є ключовим аспектом розробки, який визначає зручність використання та ефективність виконання основних завдань. Для забезпечення інтуїтивного та продуктивного процесу роботи з системою, необхідно спроектувати сценарії взаємодії, які охоплюють усі доступні функції як для адміністратора, так і для кінцевих користувачів. У контексті чат-бота для проведення опитувань ці сценарії мають включати дії користувачів, такі як вибір теми, участь в опитуванні, перегляд результатів і залишення відгуків, а також дії адміністратора, зокрема управління базою даних, створення опитувань і генерацію статистики.

Для структуризації сценаріїв взаємодії використовується підхід з описом варіантів використання системи. Це дозволяє визначити основні функціональні можливості системи, описати, як користувачі взаємодіють із нею, та відобразити логіку роботи через діаграми варіантів використання [37]. Табл. 2.3 містить узагальнені варіанти використання системи, які описують ключові дії, доступні адміністраторам і кінцевим користувачам.

Таблиця 2.3.

Варіанти використання системи

№	Ім’я	Опис
1	2	3
UC1	Реєстрація користувача	Дозволяє користувачам створювати власний профіль у системі для доступу до функціоналу чат-бота.
UC2	Автентифікація користувача	Забезпечує перевірку особи користувача для входу до системи з метою захисту даних.
UC3	Перегляд списку користувачів	Надає адміністраторам можливість переглядати детальну інформацію про всіх зареєстрованих користувачів.
UC4	Додати користувача	Дозволяє адміністраторам додавати нових користувачів до системи із зазначенням основної інформації.
UC5	Оновлення даних користувача	Забезпечує адміністраторам можливість редагувати профілі користувачів у базі даних.
UC6	Перегляд каталогу тем	Дозволяє користувачам переглядати доступні теми для опитувань у системі.
UC7	Створення нової теми	Надає адміністраторам можливість створювати нові теми опитувань із додаванням супутньої інформації.
UC8	Редагування існуючої теми	Забезпечує можливість оновлення вже наявних тем опитувань, їх структури або супутніх даних.
UC9	Перегляд запитань	Дозволяє адміністраторам або користувачам переглядати список запитань у межах вибраної теми.
UC10	Додавання нового запитання	Дозволяє адміністраторам додавати нові запитання до теми, забезпечуючи гнучкість опитувань.
UC11	Видалення запитання	Забезпечує можливість видалення неактуальних або помилкових запитань із теми.
UC12	Активація чату в Telegram	Забезпечує запуск чат-бота в Telegram для проведення опитувань і взаємодії з користувачами.
UC13	Вибір теми для опитування	Надає користувачам можливість обирати тему опитування, яка їх цікавить.
UC14	Проходження опитування	Дозволяє користувачам відповідати на запитання обраної теми та завершувати опитування.
UC15	Перегляд правил користування	Дозволяє користувачам ознайомитися з умовами використання системи.

Продовження табл. 2.3.

1	2	3
UC16	Перегляд інформації про навчальний заклад	Надає користувачам інформацію про навчальний заклад або організацію, яка використовує систему.
UC17	Залишення відгуку	Дозволяє користувачам залишати відгуки про роботу системи, теми опитувань або функціональність.
UC18	Перегляд відгуків користувачів	Забезпечує адміністраторам можливість переглядати всі відгуки, залишені користувачами.
UC19	Формування статистики про опитування	Дозволяє адміністраторам генерувати статистичні звіти за результатами опитувань.
UC20	Перегляд історії подій	Дозволяє адміністраторам переглядати журнал дій та подій у системі для моніторингу її роботи.

Згідно з вимогами до функціональності системи, було визначено три ключові ролі: «Адміністратор», «Аналітик» та «Користувач Telegram-бота». Кожна роль виконує певний набір завдань, які спрямовані на забезпечення ефективної роботи системи та спрощення взаємодії з нею. Для ролі «Адміністратор» передбачено широкий спектр функцій, включаючи керування користувачами, створення й редагування тем та запитань, а також моніторинг активності системи і формування статистичних звітів (рис. 2.1). Роль «Аналітик» зосереджена на поглибленому аналізі даних, оптимізації змісту опитувань і оцінці ефективності проведених опитувань (рис. 2.2). «Користувач Telegram-бота» виконує сценарії, які включають вибір теми опитування, участь у ньому, отримання інформації та залишення відгуків про використання системи (рис. 2.3). Така структура ролей дозволяє системі забезпечувати гнучкість у роботі, задовольняючи потреби різних категорій користувачів.

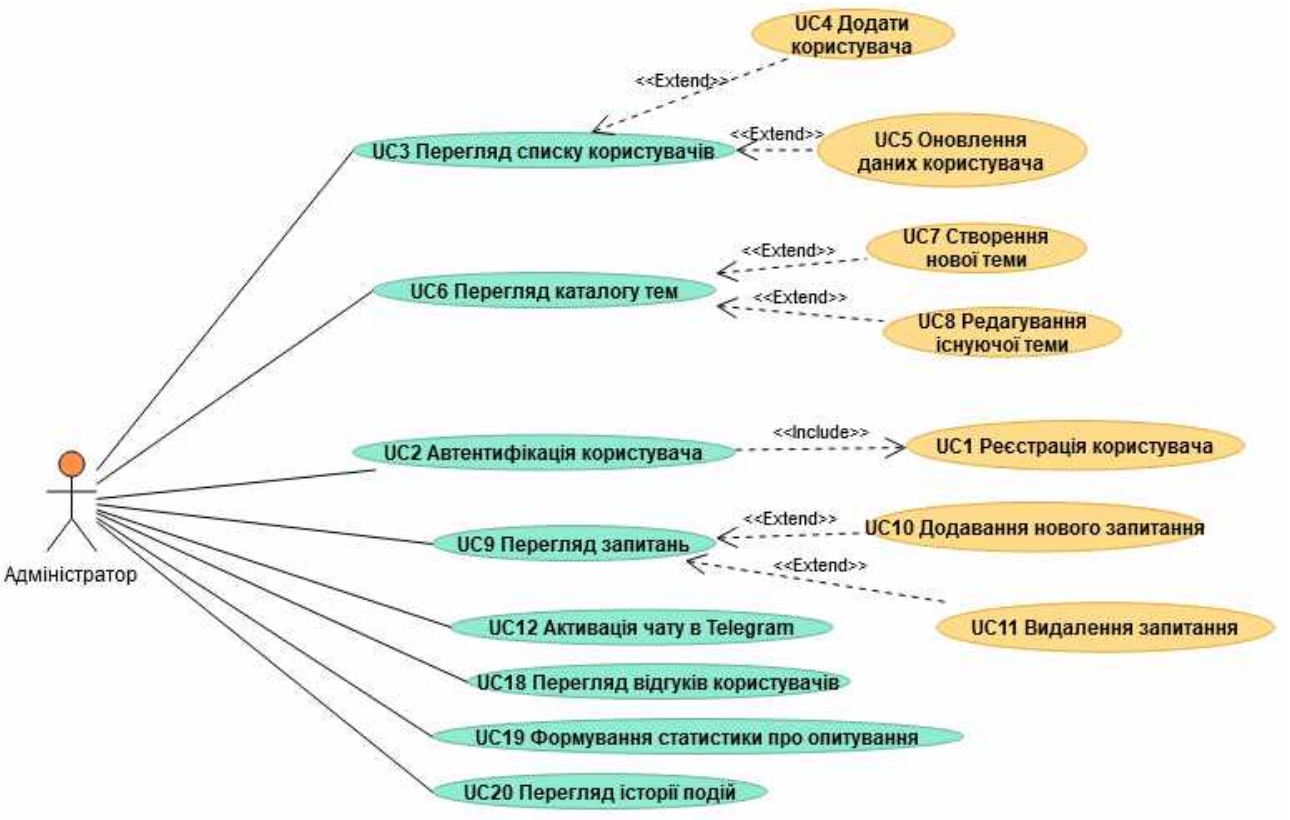


Рис. 2.1 Діаграма прецедентів для ролі «Адміністратор»

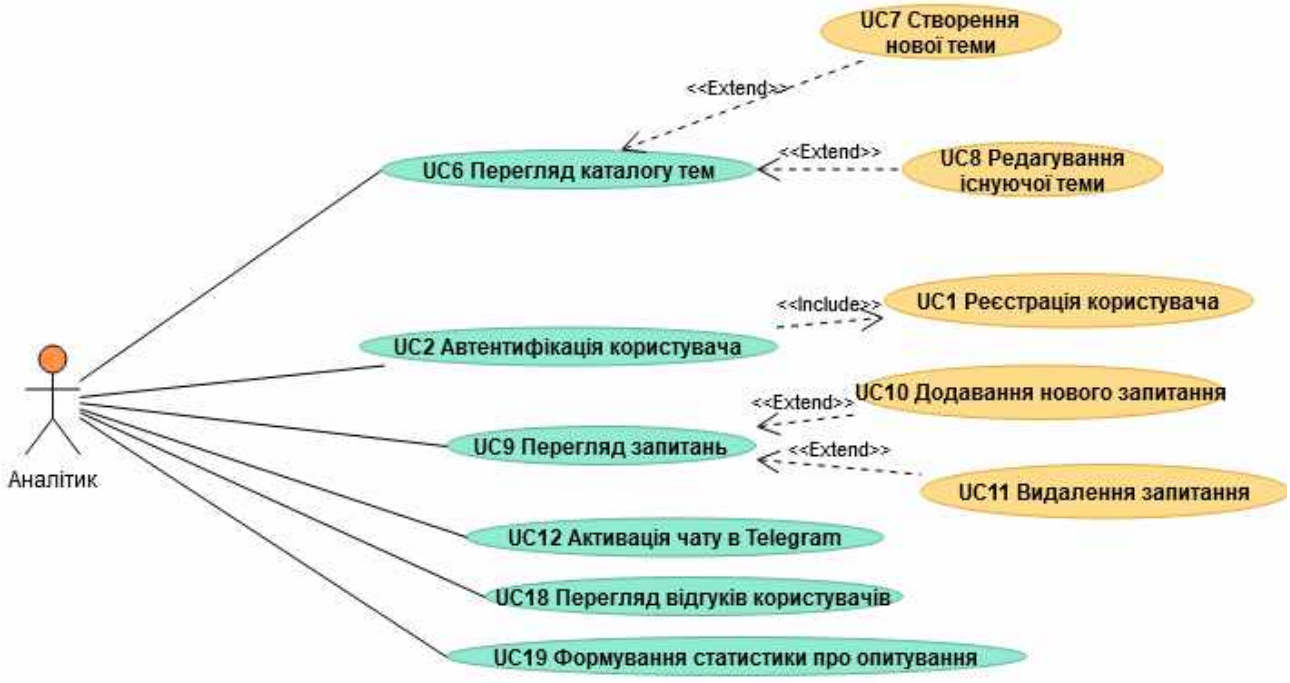


Рис. 2.2 Діаграма прецедентів для ролі «Аналітик»



Рис. 2.3 Діаграма прецедентів для ролі «Користувач ТГ бота»

На наведених діаграмах варіантів використання системи показано основні взаємозв'язки між ролями та функціями системи. Діаграми ілюструють, як кожен прецедент забезпечує різні можливості для виконання операцій у системі, що включає реєстрацію, обробку даних та доступ до тематичного контенту.

2.3 Проєктування основних бізнес-процесів

Розробка системи взаємодії з користувачами через Telegram-бот вимагає ретельного проєктування бізнес-процесів, які охоплюють усі ключові етапи функціонування системи та забезпечують її стабільність і ефективність. Проєктування передбачає детальне визначення всіх дій, які виконуються як користувачами, так і системою у відповідь на запити, починаючи від обробки команд і завершуючи збереженням результатів у базі даних. Головною метою цього підходу є створення чіткої послідовності виконання задач, таких як збір відгуків, проведення опитувань і збереження даних для подальшого аналізу. Для цього використовуються діаграми послідовностей, які допомагають візуалізувати кожен етап процесу, зокрема взаємодію між користувачем, ботом і базою даних [38, 39]. Такий підхід дозволяє забезпечити ефективну інтеграцію системи з іншими компонентами, уникнути помилок у функціонуванні та досягти високого рівня автоматизації бізнес-процесів.

На рис. 2.4 зображено процес взаємодії користувача із Telegram-ботом, який ілюструє основні етапи роботи системи, починаючи від надсилання повідомлення до збереження результатів у базі даних.

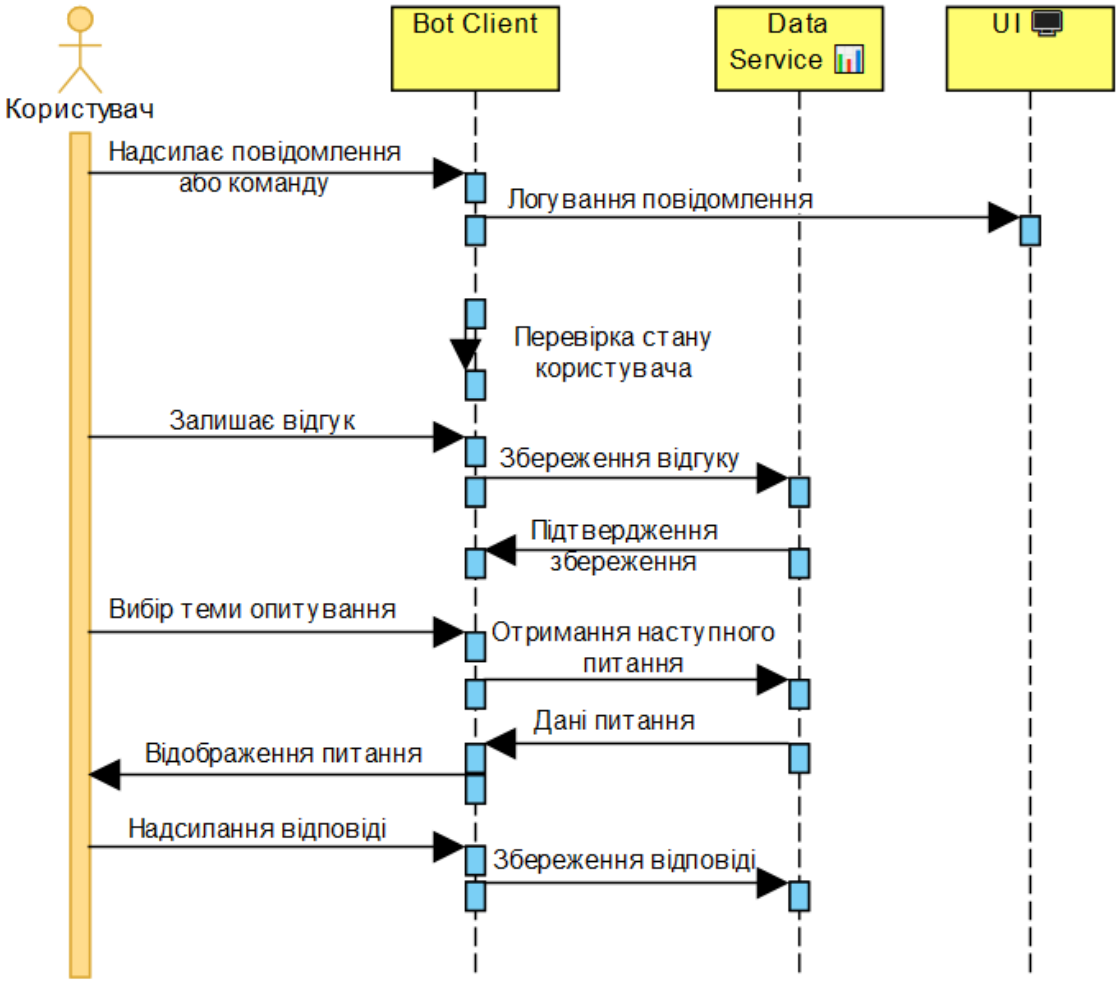


Рис. 2.4 Процес взаємодії користувача із ТГ-ботом

Процес взаємодії користувача із Telegram-ботом починається з надсилання користувачем повідомлення або команди. На цьому етапі повідомлення реєструється через функцію логування, яка відповідає за фіксацію кожної дії в системі для забезпечення прозорості та відстежуваності. Далі здійснюється перевірка стану користувача, яка визначає, чи є він зареєстрованим і чи має доступ до функціоналу системи.

Якщо користувач залишає відгук, бот передає дані у службу збереження, де відгук зберігається в базі даних, а користувач отримує підтвердження про успішне виконання операції. У разі вибору теми опитування бот запитує наступне питання, отримує відповідь і зберігає її в базі даних. Циклічність цього процесу дозволяє автоматично обробляти всі запитання обраної теми, надаючи користувачу можливість відповідати послідовно, без перевантаження інтерфейсу.

Усі питання відображаються користувачу через інтерфейс Telegram-бота, де система забезпечує чіткий і зрозумілий формат подання. Після надсилання відповіді бот передає її до служби обробки даних, яка відповідає за збереження відповідей і формування бази для подальшого аналізу. Такий підхід гарантує, що всі дані обробляються надійно, а користувач отримує плавний досвід взаємодії з системою.

Процес додавання нової теми починається з дій користувача, який відкриває форму для управління темами. На цьому етапі виконується ініціалізація форми, яка включає завантаження всіх доступних тем через запит до бази даних. Після обробки запиту список тем повертається та відображається у формі, що дозволяє користувачу бачити доступні теми та приймати рішення щодо додавання нової.

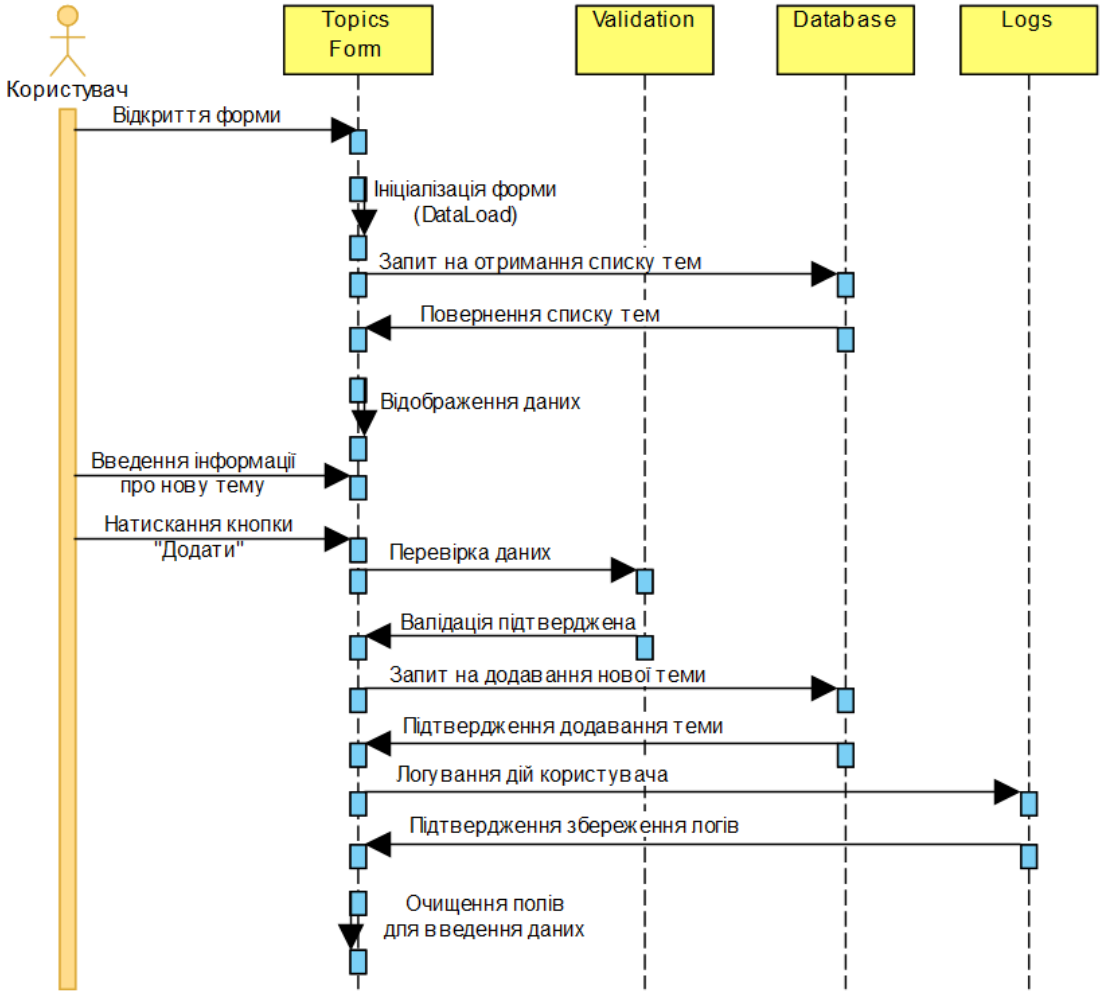


Рис. 2.5 Процес додавання нової теми для проведення опитування

Користувач вводить інформацію про нову тему, після чого натискає кнопку «Додати». В цей момент система виконує перевірку введених даних, яка забезпечує правильність і повноту внесеної інформації. Після перевірки дані проходять етап валідації, що підтверджує їх відповідність встановленим правилам. Якщо валідація успішна, формується запит до бази даних на додавання нової теми.

База даних обробляє запит і повертає підтвердження про успішне додавання теми. Після цього система логує дії користувача, зокрема додавання нової теми, для забезпечення прозорості та моніторингу активності. Збереження цих логів підтверджується, що дозволяє адміністраторам контролювати історію змін у системі. Завершальним етапом є очищення полів форми для введення даних, що дозволяє користувачу знову вводити інформацію без необхідності ручного видалення попередніх даних. Така структура процесу забезпечує надійність додавання нових тем, інтегровану перевірку даних і зручність для користувачів, що сприяє ефективному управлінню опитуваннями.

Процес формування статистики, що наведено на рис. 2.6 починається з дій користувача, який відкриває відповідну форму у системі. На першому етапі ініціалізується функція генерації статистики (GenerateStatistics), яка відповідає за підготовку запитів до бази даних для отримання необхідної інформації. Система послідовно формує запити для визначення кількості проведених опитувань, кількості отриманих відповідей, а також даних за окремими темами.

Після кожного запиту база даних повертає відповідні результати, які обробляються у формі. Перший запит отримує загальну кількість проведених опитувань у системі, що дозволяє визначити масштаб активності користувачів. Наступний запит повертає кількість відповідей, що були надані користувачами, і забезпечує оцінку рівня залученості. Далі система надсилає запит на отримання даних за темами, що дозволяє визначити популярність окремих тематичних опитувань. Після цього формується запит на отримання популярних відповідей, які найбільш часто обирали користувачі. Ці дані дозволяють аналізувати тенденції та пріоритети респондентів у межах кожної теми.

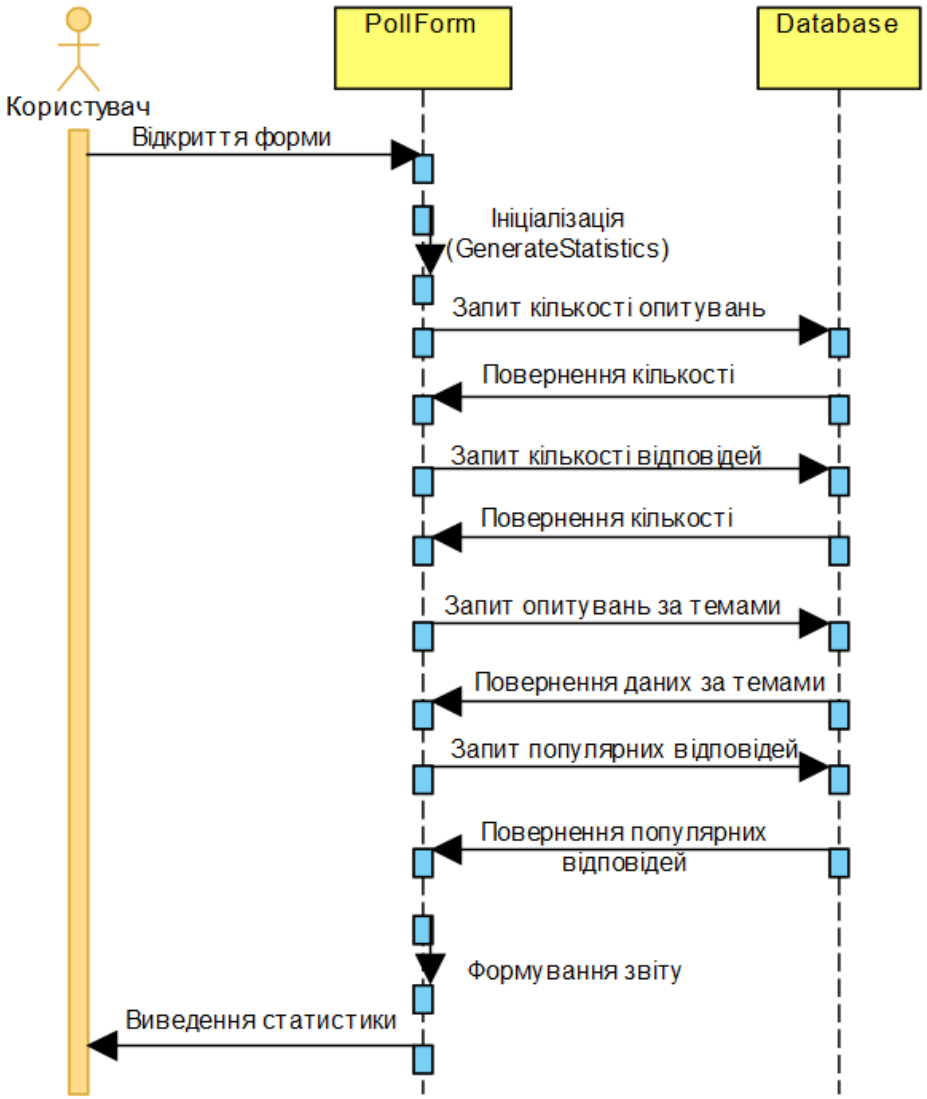


Рис. 2.6 Процес формування статистики

На заключному етапі всі отримані дані об'єднуються, і система формує підсумковий звіт. Цей звіт виводиться у зручному форматі для перегляду користувачем, що забезпечує доступність ключових показників і результатів опитувань. Такий підхід до формування статистики забезпечує глибокий аналіз діяльності системи, допомагаючи адміністраторам оцінювати ефективність роботи та приймати обґрунтовані рішення.

2.4 Розробка основних модулів системи

Розробка Telegram-бота для взаємодії зі студентами вимагає не лише детального проєктування бізнес-процесів, але й створення модулів, які забезпечують коректне функціонування бота. Проєктування бізнес-процесів

охоплює моделювання сценаріїв взаємодії користувачів із системою, побудову діаграм послідовностей і визначення ключових компонентів, які дозволяють забезпечити плавний і надійний обмін даними між користувачем, ботом і базою даних. Основним етапом є розробка модуля, що реалізує функціонування Telegram-бота, який забезпечує обробку запитів користувачів, зберігання їхніх відповідей і взаємодію з базою даних.

Модуль для функціонування Telegram-бота побудований на основі клієнта ITelegramBotClient, який забезпечує взаємодію з платформою Telegram (рис. 2.7). Для налаштування бота використано такі дані, як назва бота (_botName), його унікальний ідентифікатор (_botUserName) і токен доступу (_token), отриманий із технічного завдання. Ці параметри дозволяють Telegram-боту бути ідентифікованим у системі та підтримувати безпечний доступ до його функціоналу.

```
// Дані бота (назва, username та токен з технічного завдання)
private string _botName = "EduSurvey";
private string _botUserName = "EduSurveyBot";

// Токен бота (з технічного завдання)
private string _token = "7153237731:AAFyT2gqtx9qsV9cl2vIdEIvUf_F_GjC8sY";

// Основний клієнт Telegram
private ITelegramBotClient _botClient;

// Словник станів користувачів (chatId => стан користувача)
private Dictionary<long, BotUserState> userStates = new Dictionary<long, BotUserState>();

// Сервіс для доступу до бази даних (Topics, Questions, Responses, UserSurvey, LeavingFeedback)
private DataService _dataService;
```

Рис. 2.7 Ініціалізація змінних

Для управління станами користувачів використовується словник userStates, у якому кожен користувач ідентифікується за своїм унікальним chatId. Це дозволяє боту зберігати та обробляти інформацію про поточний стан кожного користувача, наприклад, обрану тему опитування чи відповідь на запитання. Словник забезпечує персоналізовану взаємодію між ботом і кожним користувачем, гарантуючи, що всі дії виконуються в межах контексту поточного діалогу.

Модуль також містить компонент _dataService, який відповідає за доступ до бази даних. Він забезпечує зберігання та отримання інформації про теми опитувань (Topics), запитання (Questions), відповіді користувачів (Responses), активність опитувань (UserSurvey) та залишені відгуки (LeavingFeedback). Інтеграція з базою даних через _dataService дозволяє Telegram-боту оперативно зчитувати й оновлювати дані, необхідні для проведення опитувань і обробки результатів.

Метод на рис. 2.8 відповідає за ініціалізацію Telegram-бота під час завантаження форми в графічному інтерфейсі користувача. Основним завданням є налаштування клієнта Telegram-бота (_botClient) із використанням токена доступу, який гарантує ідентифікацію бота на платформі Telegram. Після створення клієнта бот підписується на події отримання повідомлень (OnMessage) і обробки callback-запитів (OnCallbackQuery), що дозволяє реагувати на взаємодію користувачів, зокрема повідомлення чи натискання кнопок.

```
private void TelegramBotForm_Load(object sender, EventArgs e) {
  // Створюємо клієнт бота
  _botClient = new TelegramBotClient(_token);

  // Підписуємося на події
  _botClient.OnMessage += BotClient_OnMessage;
  _botClient.OnCallbackQuery += BotClient_OnCallbackQuery;

  // Запускаємо отримання повідомлень
  _botClient.StartReceiving();

  // Для прикладу: виведемо у вікно форми інформацію про запуск
  RaportTBox.Text = $"Бот '{_botName}' (@{_botUserName}) запущено...\r\n";
}
```

Рис. 2.8 Ініціалізація Telegram-бота під час завантаження форми

Після підключення обробників подій викликається метод StartReceiving, який забезпечує постійне отримання та обробку вхідних даних від користувачів. Завдяки цьому бот може працювати в режимі реального часу, реагуючи на дії користувачів без затримок. Для наочності передбачено відображення у текстовому полі форми (RaportTBox) інформації про успішний запуск бота, включаючи його ім’я (_botName) та username (_botUserName). Це дозволяє адміністраторам системи переконатися в коректному старті бота та отримати

базову інформацію про його стан. Код забезпечує зручну інтеграцію з Telegram API та дозволяє гнучко налаштовувати поведінку бота під час його роботи.

Код на рис. 2.8 реалізує обробник подій для отримання текстових повідомлень, надісланих користувачами в Telegram-бот. Коли бот отримує повідомлення, спочатку перевіряється, чи містить воно текст. Якщо повідомлення не текстове або порожнє, обробка завершується. Ідентифікатор чату (chatId) і текст повідомлення (userMessage) зберігаються для подальшої роботи. Для наочності повідомлення разом із ідентифікатором чату записується у текстове поле інтерфейсу (RaportTBox), що дозволяє адміністраторам спостерігати за вхідними даними в режимі реального часу.

```
private async void BotClient_OnMessage(object sender, MessageEventArgs e) {
  // Перевірка, чи є хоч якесь текстове повідомлення
  if (e.Message?.Text == null)
    return;
  long chatId = e.Message.Chat.Id;
  string userMessage = e.Message.Text;
  // Лог у вікні
  RaportTBox.Invoke((MethodInvoker)delegate {
    RaportTBox.Text += $"[{chatId}]: {userMessage}\r\n";
  });
  // Якщо вперше бачимо цього користувача – ініціалізуємо його стан
  if (!userStates.ContainsKey(chatId)) {
    userStates[chatId] = new BotUserState();
  }
  var userState = userStates[chatId];
  // Якщо користувач у режимі введення відгуку (IsLeavingFeedback = true)
  if (userState.IsLeavingFeedback) {
    // Зберігаємо відгук у базу
    await SaveFeedback(chatId, e.Message);
    // Скидаємо прапорець
    userState.IsLeavingFeedback = false;
    // Повертаємось у головне меню
    await SendMainMenu(chatId);
    return;
  }
```

Рис. 2.8 Обробник подій для отримання текстових повідомлень

Якщо бот вперше взаємодіє з користувачем (відсутній запис у словнику userStates), створюється новий запис із початковим станом для цього користувача. Це забезпечує можливість персоналізації взаємодії, оскільки бот зберігає інформацію про поточний стан кожного користувача.

Коли користувач перебуває в режимі введення відгуку (властивість IsLeavingFeedback активна), текст повідомлення розпізнається як відгук. Бот викликає асинхронний метод SaveFeedback, який зберігає відгук у базу даних.

Після цього стан користувача оновлюється, скидаючи режим введення відгуку, і бот відправляє головне меню, завершуючи обробку запиту. Така структура забезпечує чітке управління станами користувачів і зручну роботу з відгуками.

На рис. 2.9 наведено код, що обробляє ситуацію, коли користувач бере участь в опитуванні (активний прапорець IsSurveyInProgress у стані користувача). У цьому режимі бот отримує текст відповіді від користувача та зберігає її в базу даних через метод StoreUserAnswer, асоціюючи відповідь із поточним питанням у опитуванні. Якщо відповідь не зберігається (наприклад, через її некоректність або порожній зміст), бот відправляє користувачу повідомлення із запитом надати коректну відповідь або обрати варіант із запропонованих (1–4). Це гарантує, що відповіді, які зберігаються, є релевантними та придатними для подальшого аналізу.

```
// Якщо користувач у розпалі опитування (IsSurveyInProgress = true)
// – обробляємо відповіді на запитання
if (userState.IsSurveyInProgress) {
  // Записуємо відповідь користувача у Responses (за поточним питанням)
  bool isStored = await StoreUserAnswer(chatId, userMessage);
  if (!isStored) {
    // Наприклад, якесь повідомлення користувачу, що відповідь порожня або некоректна
    await _botClient.SendTextMessageAsync(chatId,
      "Виберіть, будь ласка, один із варіантів (1-4) або введіть коректну відповідь.");
    return;
  }
```

Рис. 2.9 Обробник подій для проведення опитування

У разі успішного збереження відповіді бот переходить до наступного запитання, збільшуючи індекс запитання (QuestionIndex) у стані користувача. Такий підхід забезпечує послідовну та контрольовану взаємодію в процесі опитування, дозволяючи користувачеві відповідати на запитання одне за одним у чітко визначеній послідовності. Це також дозволяє зберігати дані про прогрес опитування та динамічно адаптуватися до відповідей користувача.

Код на рис. 2.10 завершує логіку роботи бота в опитуванні, перевіряючи, чи користувач відповів на всі запитання, пов’язані з обраною темою. Бот отримує список запитань для поточної теми через метод _dataService.GetQuestionsByTopicId, а потім порівнює поточний індекс питання (QuestionIndex) у стані користувача з кількістю запитань у темі. Якщо всі

запитання пройдено, опитування завершується: прапорець IsSurveyInProgress змінюється на false, а користувачу відправляється повідомлення з подякою за участь. Після цього бот повертає користувача до головного меню, надаючи можливість обрати інші дії або теми.

```
    // Якщо питання закінчилися, завершуємо опитування
    var questions = _dataService.GetQuestionsByTopicId(userState.CurrentTopicId);
    if (userState.QuestionIndex >= questions.Count) {
        userState.IsSurveyInProgress = false;
        await _botClient.SendTextMessageAsync(chatId, "Опитування завершено, дякуємо за участь!");
        // Повертаємо користувача до головного меню
        await SendMainMenu(chatId);
    } else {
        // Показуємо наступне питання
        await SendNextQuestion(chatId, userState);
    }
    return;
}
```

Рис. 2.10 Завершення логіки роботи бота в опитуванні

Якщо ж запитання ще залишилися, бот викликає метод SendNextQuestion, щоб відобразити користувачу наступне запитання. Цей підхід забезпечує динамічну взаємодію з користувачем, автоматизуючи послідовний перехід між запитаннями та гарантуючи завершення опитування після останнього запитання. Така логіка сприяє зручності використання бота та дозволяє системі точно контролювати процес опитування.

Рис. 2.11 відображає код, який відповідає за формування та відправку головного меню користувачеві у вигляді інлайн-кнопок. Головне меню містить основні дії, які користувач може виконати, такі як перегляд доступних тем, ознайомлення з інформацією про заклад, перегляд правил користування та можливість залишити відгук.

```
private async System.Threading.Tasks.Task SendMainMenu(long chatId) {
  var inlineKeyboard = new InlineKeyboardMarkup(new[]
  {
            // Перший ряд
            new[]
            {
                InlineKeyboardButton.WithCallbackData("Перегляд тем", "topics"),
                InlineKeyboardButton.WithCallbackData("Інформація", "info"),
                InlineKeyboardButton.WithCallbackData("Правила", "rules")
            },
            // Другий ряд
            new[]
            {
                InlineKeyboardButton.WithCallbackData("Залишити відгук", "feedback")
            },
        });
  string welcomeText = "Вітаємо у боті EduSurvey!\nОберіть дію:";
  await _botClient.SendTextMessageAsync(
      chatId: chatId,
      text: welcomeText,
      replyMarkup: inlineKeyboard
  );
}
```

Рис. 2.11 Формування головного меню користувачеві у вигляді інлайн-кнопок

Для створення кнопок використовується InlineKeyboardMarkup, який дозволяє розташувати їх у вигляді рядків. Перший ряд містить кнопки для перегляду тем, інформації та правил, а другий – кнопку для залишення відгуку.

Після створення клавіатури бот формує вітальний текст, який повідомляє користувача про доступні дії та запрошує обрати одну з них. Потім викликається метод SendTextMessageAsync, який відправляє повідомлення з текстом і клавіатурою користувачу, визначеному за chatId. Це забезпечує інтуїтивну та зручну взаємодію з ботом, дозволяючи користувачам легко орієнтуватися в його функціоналі та швидко виконувати потрібні дії.

Код на рис. 2.12 відповідає за збереження відгуків користувачів у базу даних, використовуючи таблицю LeavingFeedback. Після отримання тексту відгуку від користувача через повідомлення (msg), бот зчитує текст і формує об’єкт LeavingFeedback, який включає ключову інформацію: ідентифікатор чату (ChatId), ім’я (FirstName) та прізвище (LastName) користувача (якщо вони доступні), дату створення відгуку (FeedbackDate) та сам текст повідомлення (UserMessage). Цей об’єкт представляє запис, який буде додано до бази даних.

```
private async System.Threading.Tasks.Task SaveFeedback(long chatId, Telegram.Bot.Types.Message msg) {
  string feedbackText = msg.Text;
  // Зберігаємо відгук
  var feedback = new LeavingFeedback {
    ChatId = chatId,
    FirstName = msg.Chat.FirstName ?? "", // Якщо потрібно
    LastName = msg.Chat.LastName ?? "",
    FeedbackDate = DateTime.Now,
    UserMessage = feedbackText
  };
  _dataService.InsertLeavingFeedback(feedback);
  await _botClient.SendTextMessageAsync(chatId, "Дякуємо! Ваш відгук збережено.");
}
```

Рис. 2.12 Збереження відгуків користувачів у базу даних

Для внесення відгуку в базу використовується метод InsertLeavingFeedback, який викликається через сервіс доступу до даних (_dataService). Після успішного збереження бот відправляє користувачу повідомлення подяки, підтверджуючи, що його відгук було збережено.

Модуль для формування звітності є важливою частиною системи, яка забезпечує аналіз і узагальнення даних, отриманих під час роботи Telegram-бота. Основна мета цього модуля – створити зрозумілі звіти на основі проведених опитувань, отриманих відповідей і зібраної статистики. Така звітність допомагає адміністраторам системи оцінювати активність користувачів, популярність тем опитувань, а також визначати тренди у відповідях. Модуль використовує запити до бази даних для збору необхідної інформації, обробляє ці дані та формує звіт у вигляді текстового або іншого зручного для перегляду формату.

Фрагмент методу «GenerateStatistics» наведеного на рис. 2.13 забезпечує формування текстового звіту, використовуючи SQL-запити для отримання ключових статистичних даних із бази. Запит queryTotalSurveys підраховує загальну кількість проведених опитувань у таблиці UserSurvey, що дає змогу визначити загальну активність користувачів. Запит queryTotalResponses отримує кількість відповідей, зібраних у таблиці Responses, що показує загальну залученість респондентів.

```
public void GenerateStatistics() {
  string report = "";
  string queryTotalSurveys = "SELECT COUNT(*) AS TotalSurveys FROM UserSurvey;";
  string queryTotalResponses = "SELECT COUNT(*) AS TotalResponses FROM Responses;";
  string querySurveysByTopic = @"
    SELECT T.TopicTitle, COUNT(US.SurveyId) AS SurveyCount
    FROM UserSurvey US
    JOIN Topics T ON US.TopicId = T.TopicId
    GROUP BY T.TopicTitle;";
  string queryPopularAnswers = @"
    SELECT Q.QuestionsText, R.Answer, COUNT(R.Answer) AS AnswerCount
    FROM Responses R
    JOIN Questions Q ON R.QuestionId = Q.QuestionId
    GROUP BY Q.QuestionsText, R.Answer
    ORDER BY Q.QuestionsText, AnswerCount DESC;";
```

Рис. 2.13 Формування статистичного звіту по проведенню опитування

Запит querySurveysByTopic повертає дані про кількість опитувань, проведених за кожною темою, шляхом об'єднання таблиць UserSurvey та Topics. Цей запит дозволяє визначити, які теми користуються найбільшою популярністю. Запит queryPopularAnswers аналізує частоту відповідей на кожне питання, використовуючи таблиці Responses і Questions. У результаті цього запиту формується ранжований список популярних відповідей за кожним запитанням, що може бути використано для виявлення ключових тенденцій і поведінкових патернів.

Метод поєднує отримані дані для формування звіту, який включає загальну статистику, деталізацію за темами та відповідями, а також дає адміністраторам та аналітикам цілісне уявлення про ефективність роботи системи та її вплив на користувачів. Даний підхід дозволяє автоматизувати аналіз великих обсягів даних і швидко отримувати релевантну інформацію для прийняття рішень.

2.5 Тестування функціоналу чат-боту

Тестування функціоналу чат-боту є важливим етапом розробки, який забезпечує перевірку коректності роботи всіх компонентів системи. Основна увага приділяється перевірці таких аспектів, як взаємодія користувачів із ботом, коректність передачі даних до бази, надійність відправки відповідей і зручність інтерфейсу. Для забезпечення повноцінного функціонування системи тестуються не лише основні модулі, але й додаткові елементи, такі як авторизація

користувачів у системі. На рисунку 2.14 зображено форму авторизації, яка дозволяє користувачам підтвердити свої облікові дані для доступу до системи.

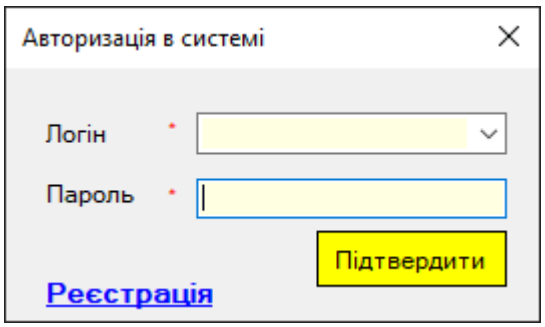


Рис. 2.15 Форма авторизації користувача у системі

Форма авторизації включає поля для введення логіна та пароля, а також кнопку підтвердження. Для перевірки її функціональності виконувалися тестові сценарії, що охоплювали введення коректних і некоректних даних. Тестування показало, що система успішно ідентифікує зареєстрованих користувачів і відмовляє у доступі при неправильному введенні логіна чи пароля. Крім того, перевірено коректність повідомлень про помилки та функцію переходу до реєстрації нових користувачів через посилання внизу форми. Результати тестування підтвердили надійність форми авторизації та її здатність інтегруватися з основним функціоналом чат-бота.

Функціонал додавання та управління темами опитувань також був ретельно протестований для забезпечення його коректної роботи. Як показано на рис. 2.15, інтерфейс дозволяє адміністраторам додавати нові теми, редагувати їх опис і переглядати список існуючих тем. Під час тестування перевірялися такі аспекти, як валідність введених даних, коректне додавання тем до списку, функціональність кнопки "Очистити" для швидкого видалення введеної інформації та коректне завершення роботи за допомогою кнопки "Вихід".

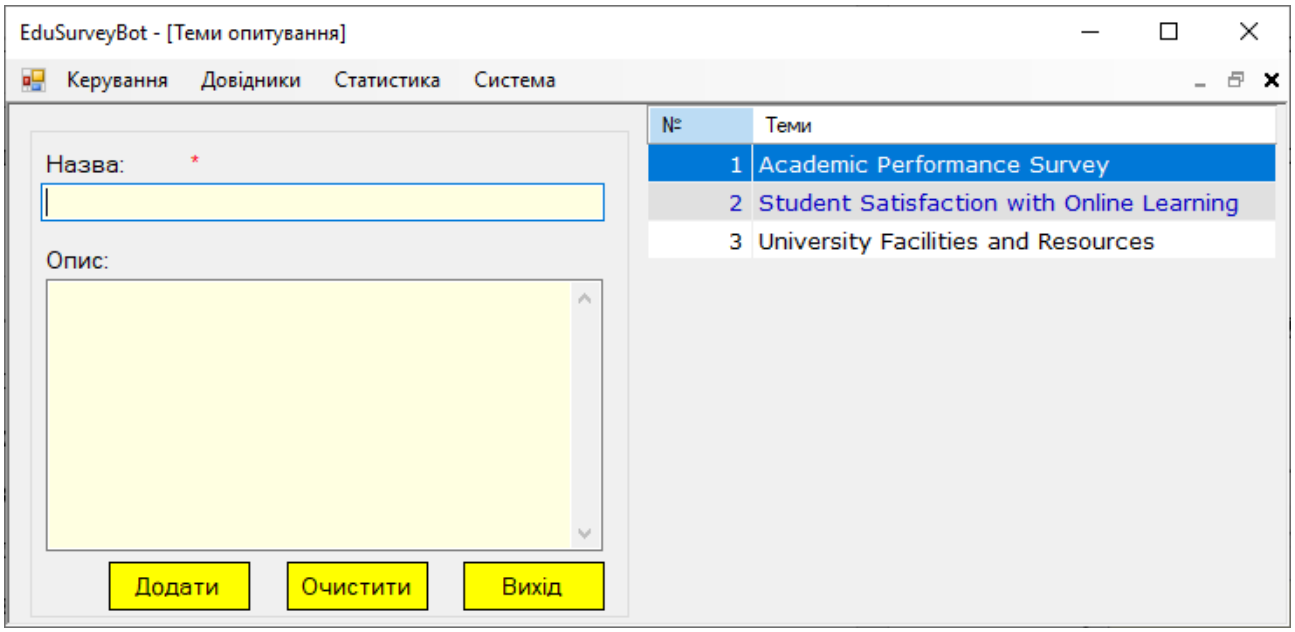


Рис. 2.15 Форма додавання та управління темами опитувань

Тестування показало, що система успішно реагує на введення валідних даних і повідомляє про помилки у випадку пропуску обов’язкових полів, таких як назва теми. Також було підтверджено, що нові теми відображаються у списку праворуч одразу після додавання, а база даних оновлюється відповідно до змін. Завдяки цим перевіркам було забезпечено стабільність і зручність роботи з функціоналом управління темами.

Функціонал додавання та редагування запитань до опитувань, зображений на рис. 2.16, був протестований для перевірки його коректності та стабільності. Інтерфейс дозволяє адміністраторам вводити текст запитання, додавати чотири варіанти відповідей і прив’язувати запитання до обраної теми опитування. Під час тестування було перевірено валідність введення даних, зокрема обов’язковість заповнення всіх полів, а також можливість очистити форму за допомогою кнопки "Очистити".

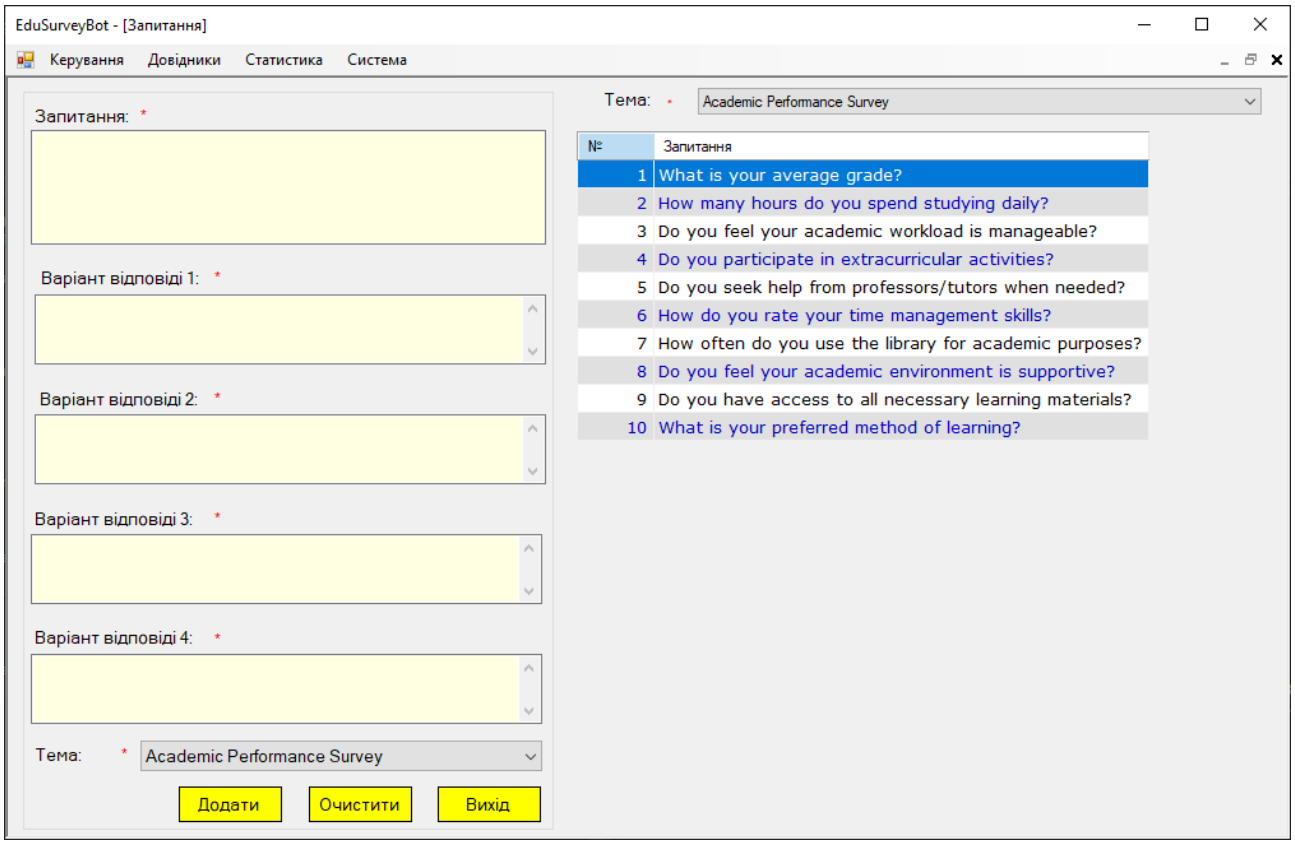


Рис. 2.16 Форма додавання та редагування запитань до опитувань

Було підтверджено, що нові запитання успішно додаються до списку праворуч і зберігаються в базі даних із відповідною прив'язкою до теми. Функція вибору теми також була протестована, що гарантує правильну прив'язку запитань до обраних опитувань. Тестування підтвердило надійність цього модуля та його готовність до інтеграції з іншими компонентами системи, такими як чат-бот і модуль статистики.

Модуль статистики, представлений на рис. 2.17, був протестований для перевірки його здатності генерувати звіти на основі даних, зібраних у процесі опитувань. Інтерфейс надає зведену інформацію, таку як загальна кількість опитувань і відповідей, а також деталізацію по темах із зазначенням популярних відповідей. Тестування охоплювало перевірку правильності виконання SQL-запитів до бази даних, а також точності відображення результатів у текстовому полі.

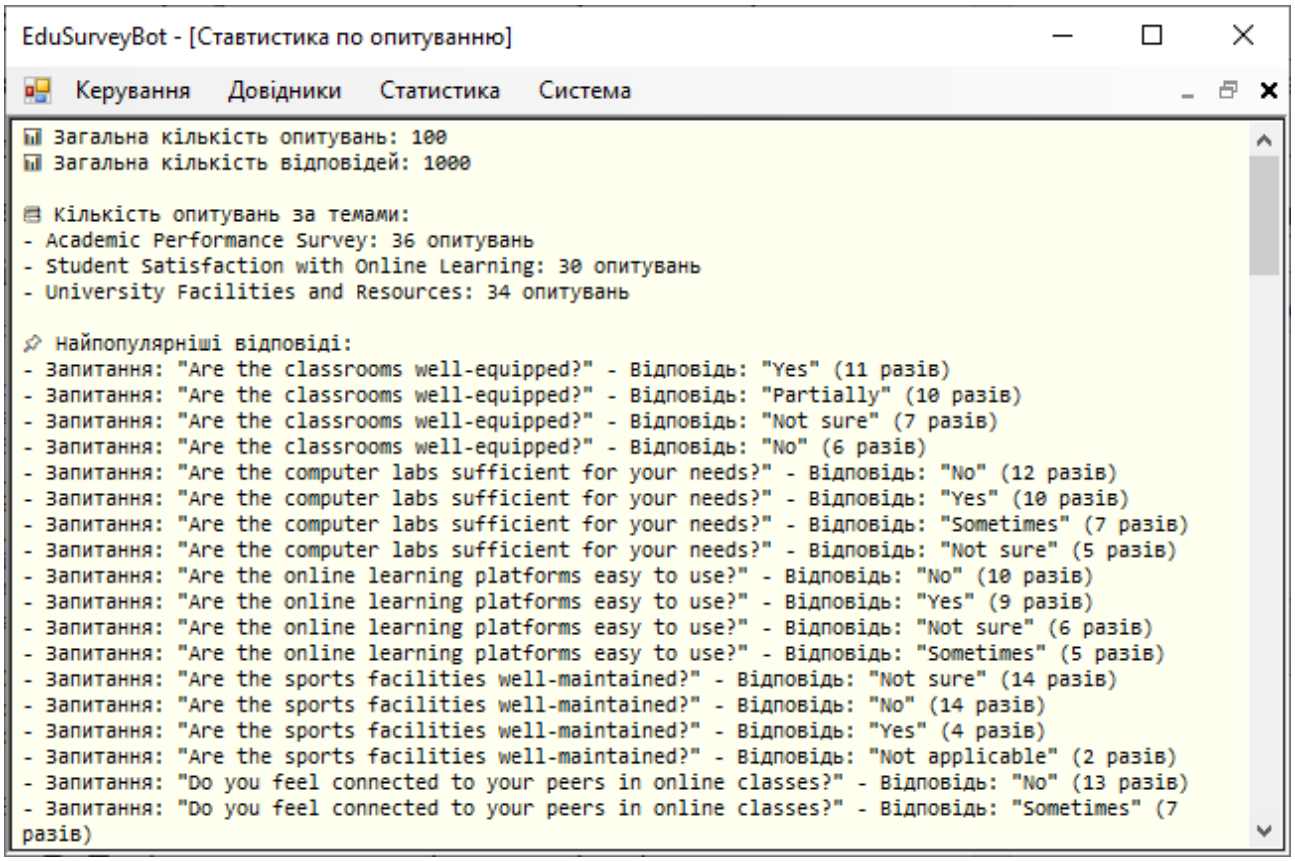


Рис. 2.17 Форма статистики

Було підтверджено, що всі дані генеруються коректно, включаючи кількість проведених опитувань за темами та найбільш популярні відповіді респондентів. Особливу увагу приділили тестуванню обробки великих обсягів даних, що гарантує стабільність роботи навіть при значній кількості записів. Результати показали, що модуль формує зручний для аналізу звіт, який можна використовувати для оцінки ефективності системи та виявлення ключових тенденцій у відповідях.

Функціонал Telegram-бота, зображений на рис. 2.18, був протестований для перевірки його здатності забезпечувати інтерактивну взаємодію з користувачами. Бот успішно відображає головне меню з інлайн-кнопками, що пропонують користувачам обрати одну з доступних дій: перегляд тем, ознайомлення з інформацією, правилами або можливість залишити відгук. Після вибору дії бот належним чином реагує, переходячи до відповідного сценарію.

Особлива увага приділялася тестуванню обробки вибору тем для опитування. Бот коректно відображав список тем, дозволяючи користувачам

зробити вибір і розпочати опитування. Крім того, перевірено поведінку кнопки "Повернутися у меню", яка успішно повертає користувача до головного інтерфейсу без затримок чи збоїв. Тестування показало, що бот забезпечує інтуїтивний і зрозумілий досвід для користувачів, підтримуючи стабільну роботу на кожному етапі взаємодії.

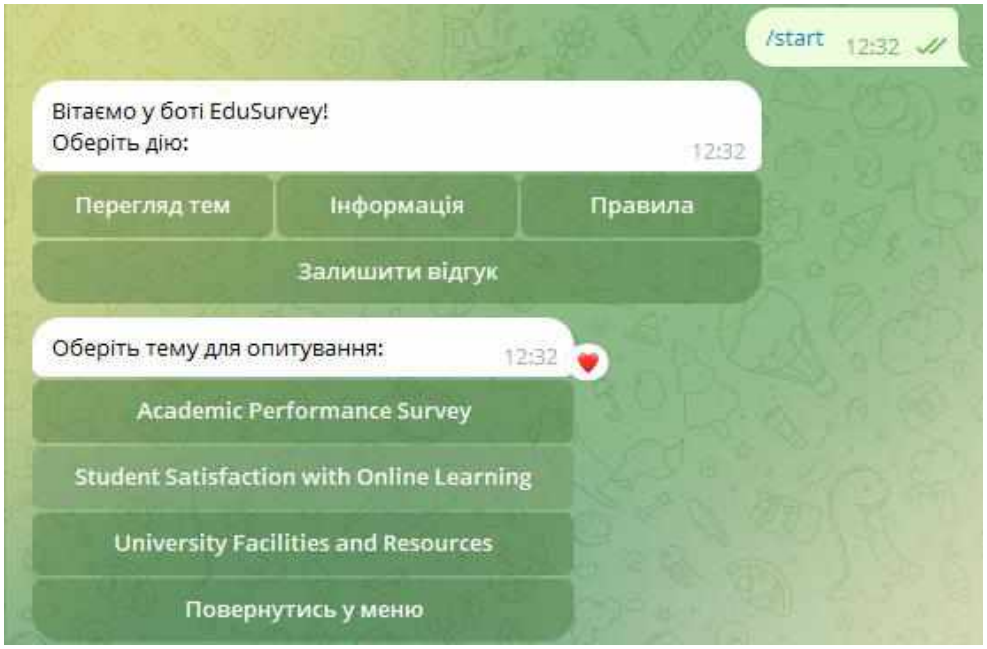


Рис. 2.18 Головне меню ТГ бота

Процес проведення опитування через Telegram-бот, зображений на рис. 2.19, був протестований для перевірки коректності взаємодії з користувачами. Після вибору теми бот розпочинає опитування, надсилаючи перше запитання з варіантами відповідей.

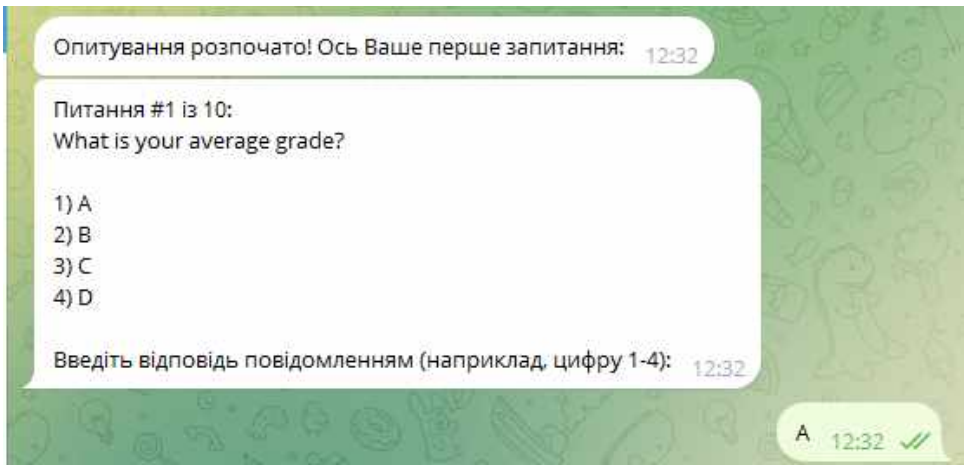


Рис. 2.19 Проведення опитування через Telegram-бот

Тестування показало, що бот успішно розпізнає введену відповідь, зберігає її в базі даних і автоматично переходить до наступного запитання. Перевірено, що у випадку некоректної або порожньої відповіді бот надсилає відповідне повідомлення із проханням повторно надати правильну відповідь. Це забезпечує безперервний і контрольований процес опитування, дозволяючи користувачам проходити його без ускладнень. Загальна взаємодія була стабільною, а відповіді на запитання оброблялися швидко, що підтверджує високу надійність реалізованого функціоналу.

На заключних етапах опитування, як показано на рис. 2.20, бот продовжує надсилати запитання відповідно до обраної теми, зберігаючи відповіді користувача після кожного введення. Користувачам пропонується кілька варіантів відповідей, зокрема текстові та числові, що дозволяє забезпечити зручність і гнучкість під час вибору. Після кожної відповіді бот миттєво переходить до наступного запитання, підтримуючи динамічність процесу.

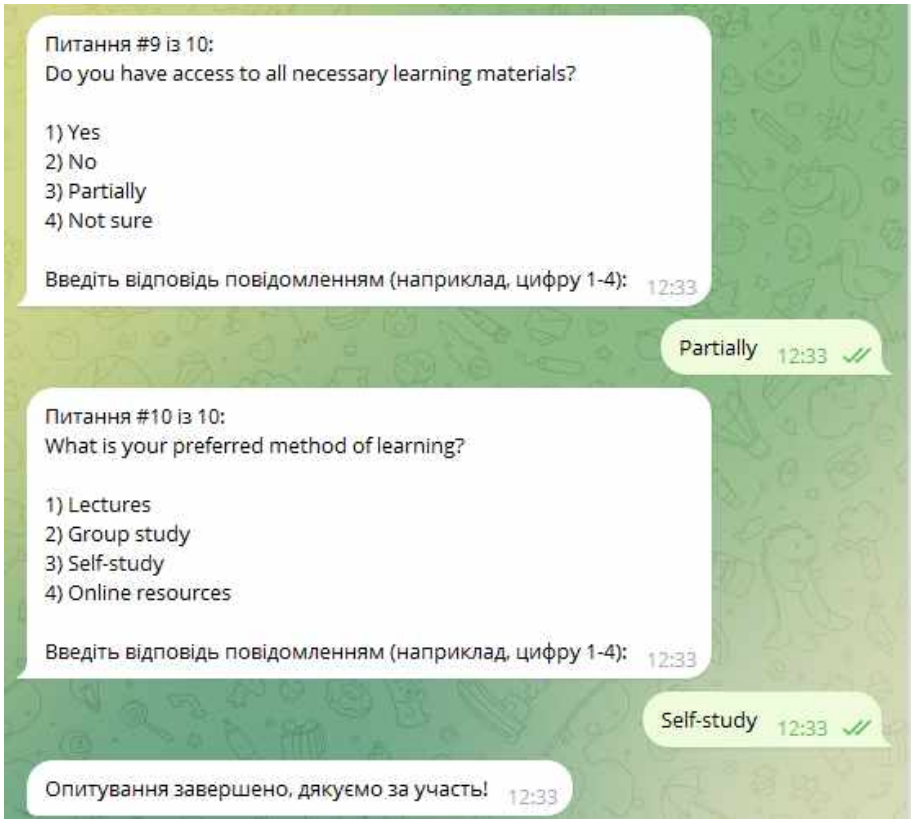


Рис. 2.20 Заключний етап опитування

Коли користувач відповідає на останнє запитання, бот автоматично завершує опитування та відправляє повідомлення з подякою за участь.

Тестування показало, що цей функціонал працює без збоїв навіть при великій кількості питань і відповідей. Завершення опитування відбувається плавно, а користувач отримує підтвердження про успішну участь, що створює позитивний досвід взаємодії з системою.

Функціонал надання інформації про заклад вищої освіти, зображений на рис. 2.21, був протестований для перевірки його здатності надавати чітку, структуровану та зрозумілу інформацію користувачам. Бот формує текстове повідомлення, яке містить основні переваги закладу, такі як широкий вибір програм, професійний викладацький склад і використання інноваційних технологій у навчанні. Інформація структурована за допомогою іконок, що робить її більш зручною для сприйняття.

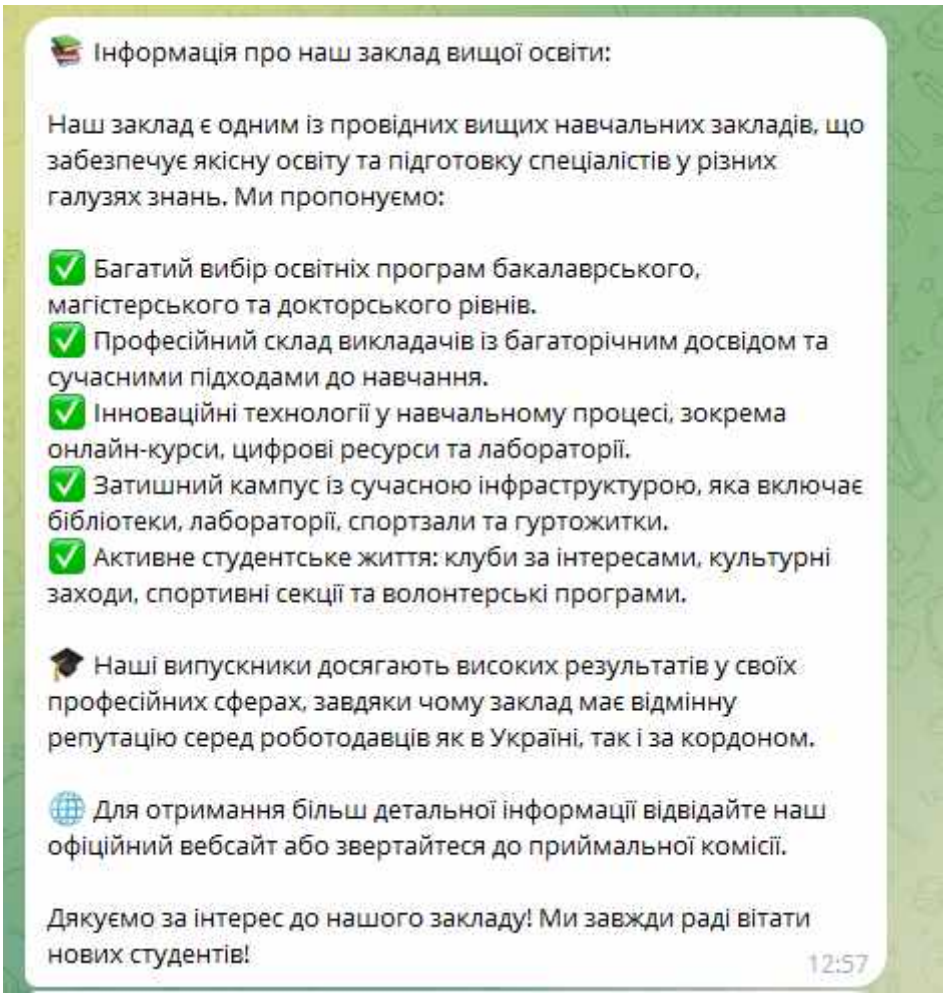


Рис. 2.21 Надання інформації про заклад вищої освіти

Тестування підтвердило коректність відображення всього тексту та перевірило інтерактивність посилання на офіційний вебсайт, яке коректно

перенаправляє користувачів на ресурс для отримання додаткової інформації. Цей функціонал працює стабільно, забезпечуючи студентам доступ до необхідних відомостей про заклад і підвищуючи довіру до системи. Загальний досвід використання був позитивним, а інтерфейс зручним і зрозумілим для користувачів.

Функціонал залишення відгуків, як показано на рис. 2.22, був перевірений на зручність взаємодії та стабільність роботи. Бот успішно запрошує користувача ввести свій відгук одним повідомленням, після чого зберігає отриманий текст у базі даних. Під час тестування перевірялася обробка різних типів повідомлень, включаючи довгі тексти та випадки відсутності введених даних.

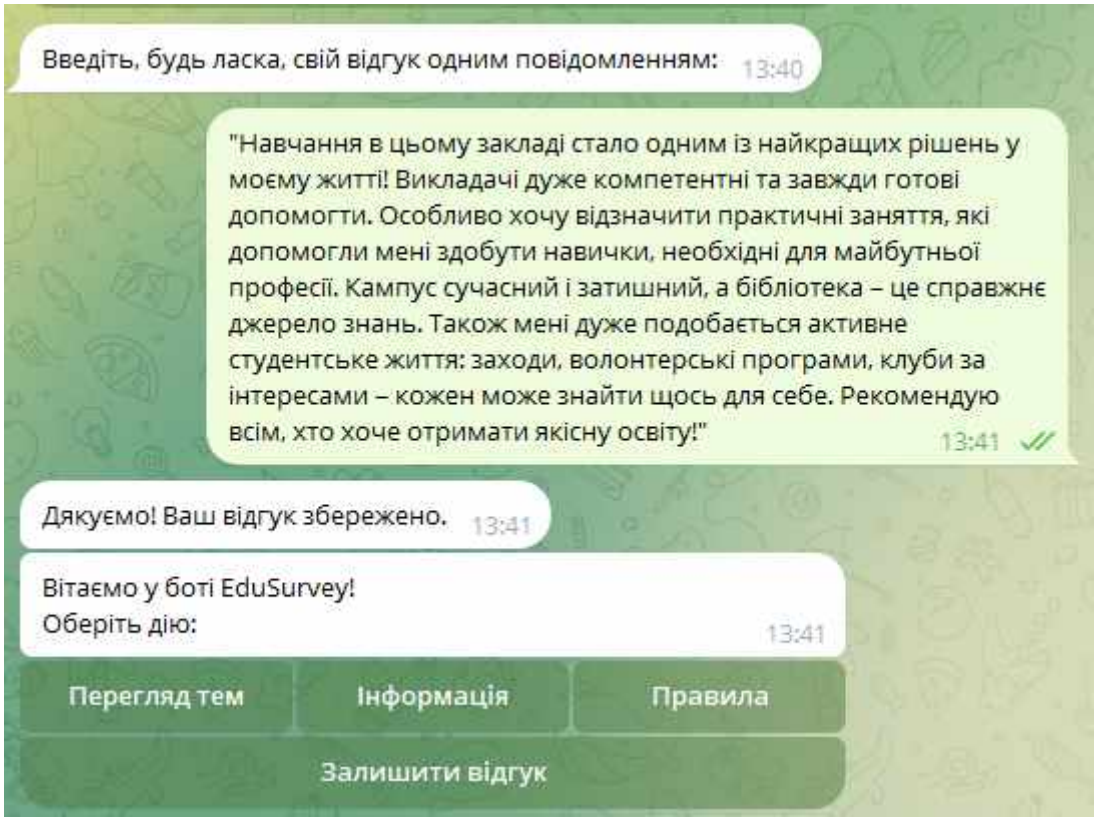


Рис. 2.22 Залишення відгуку про заклад

Для перевірки коректності роботи програмного забезпечення було використано MSTest – інструмент для тестування, який дозволяє автоматизовано перевірити функціональні можливості системи, забезпечуючи надійність та стабільність її роботи. Проведення MSTest тестування дозволило перевірити широкий набір методів і функцій, що використовуються в системі, зокрема функціонал, пов'язаний з обробкою даних чат-бота для медичної допомоги.

2.6 Висновок

У рамках даного розділу було проведено комплексне проектування, розробку та тестування чат-бота для проведення опитувань, що дозволило створити основу для функціональної та ефективної роботи системи. Було порівняно популярні платформи для розробки чат-ботів, серед яких Telegram Bot API, Dialogflow та Chatfuel, і обрано Telegram Bot API завдяки його гнучкості та інтеграції з C#. Також було порівняно мови програмування Python, Java та C#, і на основі вимог системи обрано C# як найоптимальніший варіант для розробки.

На основі визначених функціональних вимог системи сформовано сценарії взаємодії користувачів із системою, які охоплюють три основні ролі: «Адміністратор», «Аналітик» та «Користувач Telegram-бота». Ці сценарії відображено у вигляді діаграм варіантів використання, що деталізують взаємодію кожної ролі з системою. Проєктування бізнес-процесів включало створення діаграм послідовності для ключових процесів, таких як взаємодія з ботом, додавання тем для опитувань і формування статистичних даних.

Розроблено основні модулі системи, включаючи ініціалізацію Telegram-бота, обробку текстових повідомлень, проведення опитувань, формування головного меню з інлайн-кнопками, а також збереження відгуків і відповідей у базі даних. Кожен модуль реалізовано з урахуванням вимог до надійності, гнучкості та зручності для користувачів.

Для перевірки стабільності роботи системи виконано тестування основних компонентів, таких як форма авторизації користувачів, модулі додавання та редагування тем і запитань, інтерфейс для роботи зі статистикою, головне меню бота, процес проведення опитувань, а також функціонал надання інформації про заклад і залишення відгуків. Результати тестування підтвердили коректність роботи системи та готовність до подальшого використання.

Отримані результати розробки та тестування створюють основу для проведення аналізу функціональності та вартості програмного забезпечення в наступному розділі, що дозволить оцінити ефективність створеної системи та визначити шляхи її вдосконалення.

РОЗДІЛ 3.
АНАЛІЗ ФУНКЦІОНАЛЬНОСТІ ТА ВАРТОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Визначення обґрунтованих функцій програмного забезпечення

У процесі розробки чат-бота для опитування здобувачів вищої освіти використовується метод функціонально-вартісного аналізу (ФВА) для оцінки ефективності та корисності функцій програмного забезпечення [40]. Даний метод дозволяє визначити ключові можливості чат-бота, які забезпечують його практичну цінність для кінцевого користувача та підвищують ефективність опитувань.

Функціональність бота визначається технічними вимогами, які передбачають автоматизацію збору та аналізу відповідей користувачів. Важливими аспектами є доступність, інтерактивність та безпека, що забезпечують зручність користування для студентів та адміністрації закладу освіти.

Основні технічні вимоги до чат-бота:

– програмний продукт реалізовано у вигляді Telegram-бота, що забезпечує доступність на мобільних пристроях та комп’ютерах без необхідності встановлення додаткового ПЗ;

– бот має інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам легко орієнтуватися в меню, проходити опитування та залишати зворотний зв’язок;

– забезпечується оперативна обробка запитів завдяки інтеграції з базою даних та використанню асинхронних запитів;

– результати опитувань мають зберігатися у базі даних для подальшого аналізу та формування звітності;

– програмне забезпечення повинно гарантувати безпечне зберігання даних, дотримуючись вимог конфіденційності.

Головна функція програмного продукту:

F_0 – розробка чат-бота для автоматизації проведення опитувань, збору та аналізу відповідей.

На основі технічних вимог виділено три ключові функції:

F_1 – інтерактивний інтерфейс для взаємодії з користувачами:

а) Використання текстових команд /start, /help, що дозволяють розпочати роботу з ботом та отримати довідкову інформацію;

б) Реалізація inline-кнопок, які забезпечують швидку навігацію між основними функціями (перегляд тем, початок опитування, залишення відгуку).

F_2 – проведення та адміністрування опитувань:

а) Надання можливості користувачеві обрати тему опитування із запропонованого списку;

б) Послідовне надсилання питань із варіантами відповідей або відкритими полями для введення.

F_3 – обробка відповідей користувачів та збереження результатів:

а) Запис відповідей у таблицю бази даних, що дозволяє зберігати відповіді для подальшого аналізу;

б) Формування підтвердження про завершення опитування та відправка подяки користувачу.

Розглянувши можливі варіанти реалізації кожної функції, обґрунтовано наступні рішення:

– для F_1 (Інтерактивний інтерфейс): використано inline-кнопки, оскільки вони підвищують зручність навігації порівняно з текстовими командам;.

– для F_2 (Проведення опитувань): реалізовано інтеграцію з базою даних, що дозволяє гнучко змінювати перелік питань без зміни коду;

– для F_3 (Обробка відповідей): передбачено автоматичне збереження у таблиці Responses, що забезпечує структуроване збирання даних.

Виділені функції чат-бота забезпечують ефективну взаємодію з користувачем, автоматизують процес проведення опитувань та дозволяють зберігати дані для подальшого аналізу.

Для розробки чат-бота, призначеного для проведення опитувань здобувачів вищої освіти, необхідно визначити найоптимальніші способи реалізації основних функцій. Для цього використовується морфологічна карта, яка дозволяє проаналізувати можливі комбінації технічних рішень і вибрати найкращий варіант для кожної функції (рис. 3.1).

На рис. 3.1 наведено можливі варіанти реалізації функцій, розглянуті в рамках даної розробки.

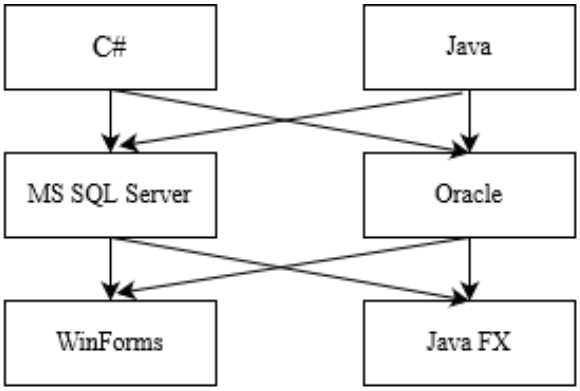


Рисунок 3.1 – Морфологічна карта реалізації функцій чат-бота

Для визначення найбільш ефективного способу реалізації кожної функції використано позитивно-негативну матрицю, яку зображено у табл. 3.1.

Таблиця 3.1 – Позитивно-негативна матриця реалізації функцій чат-бота

Основні функції	Варіанти реалізації	Переваги	Недоліки
F_1 – Інтерактивний інтерфейс	а	Простота реалізації, сумісність з усіма клієнтами Telegram	Менш зручне управління для користувача
	б	Інтуїтивне керування, підвищений UX	Обмежена кастомізація у Telegram
F_2 – Проведення опитувань	а	Простота реалізації, мінімальні затрати ресурсів	Відсутня гнучкість у зміні запитань
	б	Гнучкість у додаванні нових опитувань	Вимагає підтримки структури БД
F_3 – Обробка відповідей	а	Не потребує БД, проста розробка	Ускладнена обробка даних та аналіз результатів
	б	Можливість аналітики, централізоване збереження	Залежність від сервера та СУБД

Аналізуючи варіанти реалізації основних функцій, було визначено найбільш ефективні підходи:

– для F_1 (інтерактивний інтерфейс) обрано варіант б (Inline-кнопки), оскільки він забезпечує зручну взаємодію та швидку навігацію;

– для F_2 (проведення опитувань) обрано варіант б (динамічне завантаження питань з БД), що дозволяє змінювати питання без редагування коду;

– для F_3 (обробка відповідей) обрано варіант б (збереження у базі даних), який дає змогу ефективно аналізувати отримані результати.

Таким чином, обрані варіанти дозволяють створити гнучку, інтерактивну та масштабовану систему, що задовольняє потреби студентів і адміністраторів закладу освіти.

3.2 Встановлення оптимальних параметрів програмного забезпечення

На основі технічних вимог, визначених для чат-бота з проведення опитувань здобувачів вищої освіти, а також ключових функцій програмного продукту, встановлено головні параметри продуктивності. Вони будуть використані для оцінки ефективності програмного рішення та визначення коефіцієнта його технічного рівня.

Для аналізу оптимальних параметрів програмного забезпечення використовуються наступні критерії:

– X1 – швидкість обробки запитів користувачів, що визначається ефективністю реалізації бота та вибором програмної архітектури;

– X2 – обсяг пам’яті, необхідний для зберігання даних, включаючи базу даних з опитуваннями, відповідями користувачів та аналітичними звітами;

– X3 – час відповіді бота, тобто середній час, що витрачається на отримання, обробку та відправку відповіді користувачу;

– X4 – обсяг програмного коду, який необхідний для реалізації основних функцій чат-бота.

Для кожного із визначених параметрів (X_1 – X_4) встановлено граничні значення, що відповідають найгіршим, середнім та найкращим можливим показникам. Ці значення формуються на основі аналізу продуктивності аналогічних рішень, специфікацій замовника та реальних умов експлуатації чат-бота.

У табл. 3.2 наведено характеристику основних параметрів програмного продукту з визначенням їхніх граничних значень.

Таблиця 3.2.

Характеристики основних параметрів програмного продукту

Назва параметра	Умовне позначення	Одиниці виміру	Значення параметра		
			Гірші	Середні	Кращі
Швидкість обробки запитів	X1	Запитів/сек	5	15	50
Обсяг пам’яті для збереження даних	X2	МБ	256	128	64
Час відповіді бота	X3	мс	200	80	20
Обсяг програмного коду	X4	Кількість рядків	2538	2100	2600

За даними табл. 3.2 побудовано графічні характеристики параметрів (рис. 3.2 – рис. 3.5).

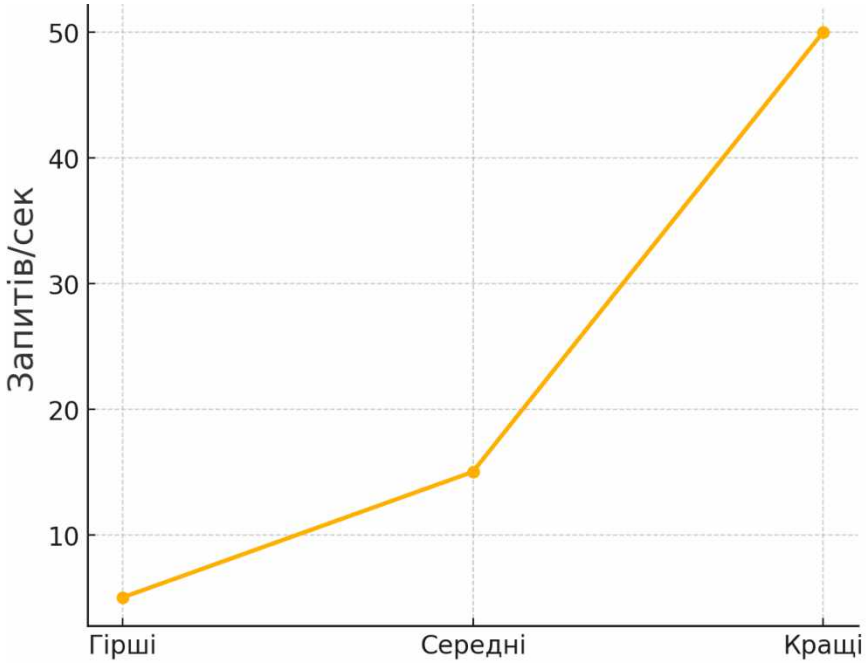


Рисунок 3.2 – Швидкість обробки запитів

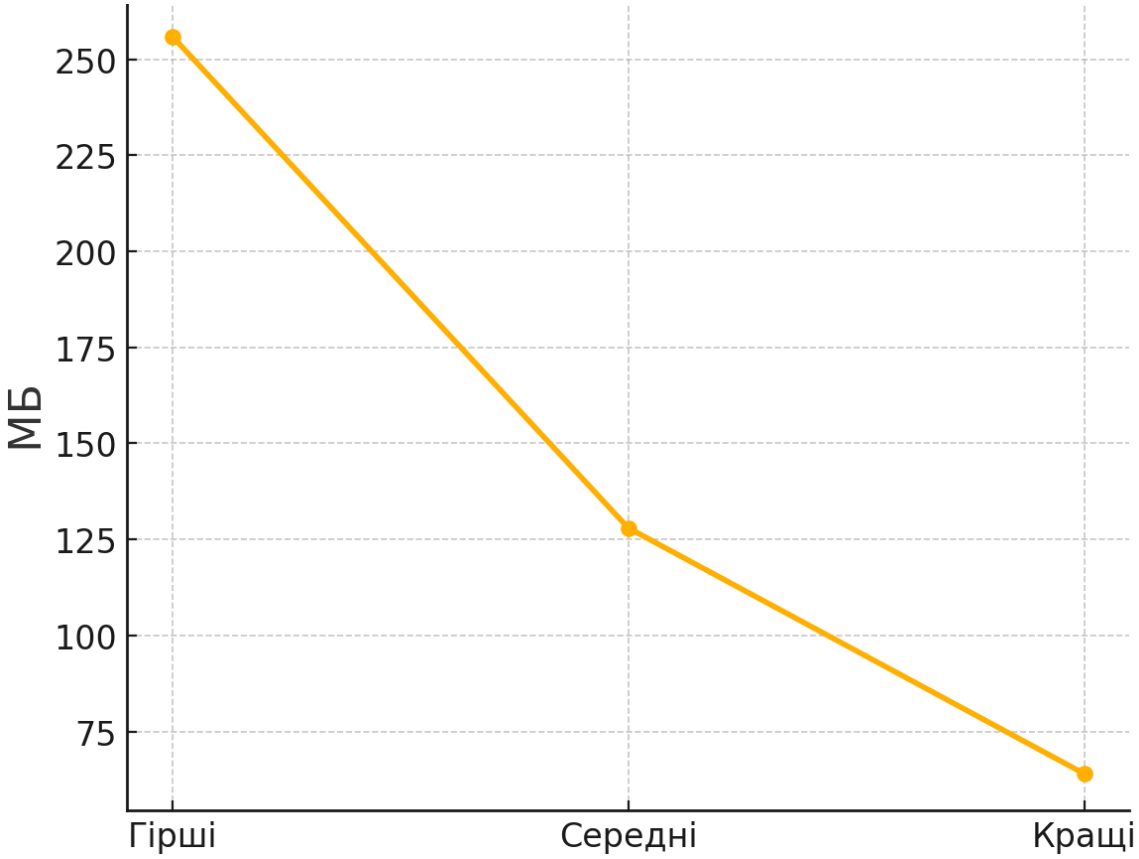


Рисунок 3.3 – Обсяг пам’яті для збереження даних

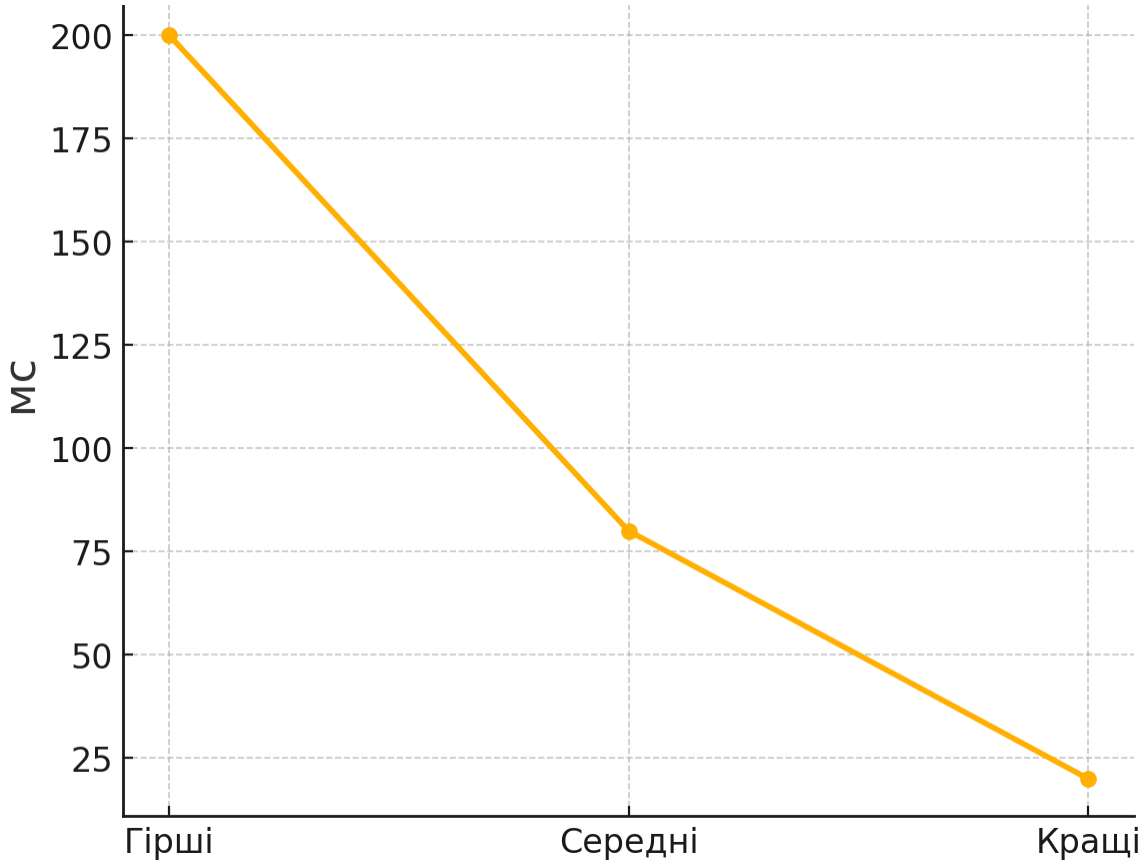


Рисунок 3.4 – Час відповіді бота

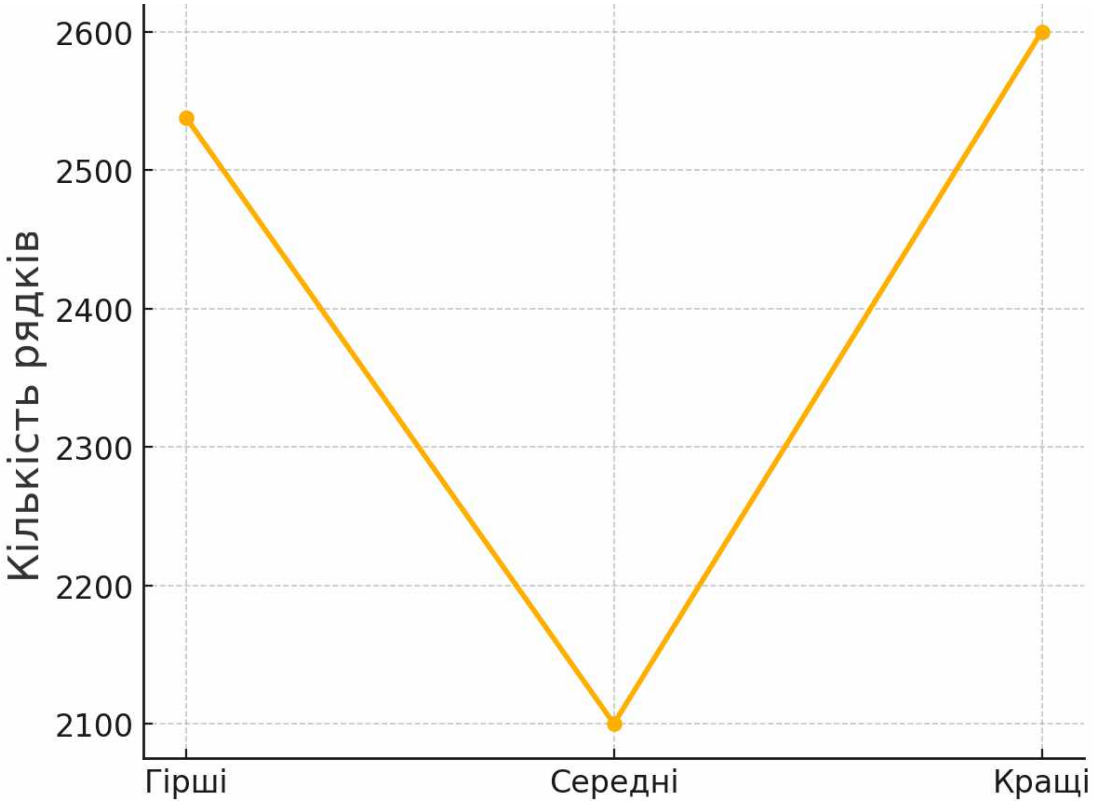


Рисунок 3.5 – Обсяг програмного коду

Швидкість обробки запитів безпосередньо залежить від ефективності взаємодії бота з базою даних, обробки повідомлень у Telegram API та продуктивності сервера. Оптимальним вважається значення 50 запитів за секунду, що забезпечує миттєву реакцію бота на дії користувачів. Обсяг пам’яті для збереження даних визначається форматом збереження відповідей та використаними методами оптимізації. Використання ефективної структури бази даних дозволяє досягти мінімального використання пам’яті в межах 64 МБ, що є оптимальним показником. Час відповіді бота є критичним параметром для забезпечення комфортного користувацького досвіду. У найкращому випадку бот повинен реагувати менш ніж за 20 мс, що досягається за рахунок асинхронної обробки запитів і кешування популярних даних. Обсяг програмного коду впливає на підтримуваність та масштабованість системи. Оптимізація коду шляхом рефакторингу, використання модульної архітектури та повторного використання компонентів дозволяє зменшити кількість рядків до 2000, що є бажаним результатом для ефективного управління розробкою. Встановлені

параметри дозволяють оцінити ефективність роботи чат-бота та забезпечити його оптимальну продуктивність. Досягнення найкращих значень у наведених характеристиках сприятиме стабільній роботі програмного забезпечення та зручності використання для здобувачів вищої освіти.

3.3 Експертний аналіз параметрів та їх оцінка

Для визначення важливості ключових параметрів програмного забезпечення було проведено експертний аналіз. До складу експертної комісії увійшли спеціалісти у сфері розробки програмних систем, аналізу даних та впровадження інформаційних технологій у навчальний процес. Метою дослідження було визначення вагомості кожного параметра у контексті розробки чат-бота, що забезпечує автоматизацію процесу опитувань та ефективне збереження й обробку отриманих даних.

Для оцінки значущості параметрів використано метод попарного порівняння, що дозволяє встановити відносну вагу кожного параметра шляхом безпосереднього порівняння їх між собою. Процедура оцінювання складалася з кількох етапів:

– визначення рангів параметрів відповідно до їх важливості для досягнення цілей проєкту;

– перевірка узгодженості експертних оцінок з метою їх подальшого використання в аналізі;

– розрахунок пріоритетних значень для кожного параметра за результатами попарного порівняння;

– аналіз отриманих даних для визначення коефіцієнтів значущості кожного параметра у загальній оцінці якості програмного забезпечення.

Результати експертного аналізу дозволяють виявити найбільш критичні параметри, які впливають на продуктивність та ефективність чат-бота, та визначити оптимальні підходи до їх коригування з метою покращення функціональності системи.

Результати цього експертного оцінювання представлені в табл. 3.3.

Таблиця 3.3.

Отримані результати експертного ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкість обробки запитів	Запитів /с	1	2	3	4	5	6	7	27	0,75	0,56
X2	Об'єм пам'яті	Мб.	4	3	4	4	4	4	4	25	–1,25	1,56
X3	Час відповіді бота	мс	4	4	4	3	4	3	3	12	–14,25	203
X4	Обсяг програм-ного коду	К-сть рядків	2	2	1	2	1	2	2	41	14,75	217
Разом			15	15	15	15	15	15	15	105	0	420

Для перевірки достовірності отриманих результатів експертного аналізу було проведено розрахунок кількох критичних параметрів. Це дозволяє оцінити узгодженість думок експертів та визначити коректність ранжування параметрів.

а) Обчислення суми рангів кожного параметра та загальної суми рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 105, \quad (3.1)$$

де:

N – кількість експертів,

n – кількість оцінюваних параметрів.

б) Середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 26{,}25. \quad (3.2)$$

в) Визначення відхилення суми рангів кожного параметра від середнього значення:

$$\Delta_i = R_i - T. \quad (3.3)$$

Сума всіх відхилень повинна дорівнювати нулю, що підтверджує правильність розрахунків.

г) Загальна сума квадратів відхилень:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 420{,}75. \quad (3.4)$$

д) Обчислення коефіцієнта узгодженості:

$$SW = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 420{,}75}{7^2(5^3-5)} = 1{,}03 > W_k = 0{,}67. \quad (3.5)$$

Отриманий коефіцієнт узгодженості перевищує нормативний рівень 0,67, що свідчить про високий рівень узгодженості експертних оцінок. Це означає, що отримане ранжування параметрів є достовірним, а прийняті рішення щодо пріоритетності параметрів можуть бути використані для подальшого аналізу та оптимізації програмного забезпечення.

На основі результатів експертного ранжування проведено попарне порівняння всіх параметрів, що дозволяє визначити їхню відносну значущість та вплив на функціональність чат-бота. Отримані дані представлені у табл. 3.4.

Таблиця 3.4.

Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	=	>	=	<	=	<	<	<	0,5
X1 і X3	<	<	<	<	<	<	<	<	0,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	<	<	<	<	<	<	<	<	0,5
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	>	>	>	>	>	>	1,5

Для кількісного визначення рівня переваги одного параметра над іншим використано індикатор порівняльної ваги a_{ij}, який визначається за такою формулою:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j. \\ 0.5 \text{ при } X_i < X_j \end{cases} \quad (3.6)$$

Отримані числові значення переваги параметрів дозволяють сформувати матрицю порівняння $A=\|a_{ij}\|$, яка використовується для подальших розрахунків.

На основі матриці переваг кожного параметра обчислено його ваговий коефіцієнт $K_{ві}$ за формулою:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \quad (3.7)$$

де b_i – сумарне значення коефіцієнтів переваги для кожного параметра, що обчислюється як:

$$b_i = \sum_{i=1}^{N} a_{ij}. \quad (3.8)$$

Отримані значення відносних ваг параметрів уточнювалися декілька разів, доки зміни у нових значеннях не стали меншими за 2%.

На другому та подальших етапах відносні ваги параметрів визначалися за допомогою уточненої формули:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \quad (3.9)$$

де b_i' – уточнене значення, яке розраховується наступним чином:

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j. \quad (3.10)$$

На основі проведених розрахунків встановлено, що різниця між значеннями коефіцієнтів вагомості на кожній ітерації не перевищує 2%, що свідчить про достатню точність отриманих результатів та узгодженість параметрів. Розрахунок вагомості параметрів на кожній ітерації представлено в табл. 3.5.

Таблиця 3.5.

Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер.	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1	0,5	0,5	1,5	3,5	0,291	22,25	0,216	100	0,215
X2	1,5	1	0,5	1,5	4,5	0,281	27,25	0,282	124,25	0,283

X3	1,5	1,5	1	1,5	5,5	0,344	34,25	0,347	156	0,348
X4	0,5	0,5	0,5	1	2,5	0,156	14,25	0,155	64,7	0,154
Всього:					16	1	98	1	445	1

На кожній ітерації вагові коефіцієнти параметрів поступово уточнювалися, і на третій ітерації зміни стали меншими за 2%, що підтверджує стабільність результатів, зокрема:

– найбільшу вагу отримав параметр X3, що свідчить про його критичну важливість у роботі системи;

– параметр X2 також має значний вплив, що пояснюється необхідністю ефективного збереження великого обсягу даних;

– параметр X1 отримав середню оцінку, оскільки він суттєво впливає на продуктивність, але має певний рівень оптимізації на рівні програмного коду;

– найменшу вагу отримав параметр X4 (обсяг програмного коду), що свідчить про те, що загальний розмір коду хоч і є важливим фактором підтримки проєкту, але не має такого критичного впливу, як інші параметри.

Отримані результати свідчать про високу узгодженість розрахунків та дозволяють зробити висновок про ключові аспекти оптимізації програмного забезпечення. Основним напрямком покращення продуктивності є зменшення часу відповіді бота та ефективне управління ресурсами пам’яті, що сприятиме підвищенню якості взаємодії користувачів із системою.

3.4 Оцінка рівня якості реалізації різних функціональних варіантів

Для визначення рівня відповідності кожного варіанта реалізації основних функцій програмного забезпечення проведено аналіз абсолютних значень вибраних технічних параметрів. Це дозволяє оцінити, наскільки кожен із варіантів реалізації відповідає технічним вимогам та оптимальним критеріям продуктивності.

Аналіз показав, що параметри *X1*, *X3* та *X4* повністю відповідають встановленим технічним вимогам для даного програмного продукту. Значення параметра *X1*, хоча й не є найкращим із можливих варіантів, було обрано так,

щоб забезпечити збалансоване співвідношення між продуктивністю та ресурсозатратністю.

Для оцінки технічного рівня кожного варіанта реалізації чат-бота використовується наступна формула:

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}\,, \tag{3.11}$$

де:

n – кількість оцінюваних параметрів,

$K_{вi}$ – коефіцієнт вагомості іі-го параметра для j-го варіанта,

B_i – оцінка параметра іі у балах для j-го варіанта.

Розрахунок технічного рівня дозволяє об'єктивно оцінити ефективність кожного варіанта реалізації, визначивши найбільш оптимальне рішення для подальшого впровадження. Усі розрахунки та оцінки наведеного у табл. 3.6.

Таблиця 3.6.

Розрахунок рівня якості реалізації функцій чат-бота

Основні функції	Варіант реалізації	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	*А*	*X1*	50 зап/сек	3.6	0.215	0.774
F2	*А*	*X2*	64 МБ	3.4	0.283	0.962
F3	*Б*	*X3*	20 мс	2.4	0.348	0.835
F4	*А*	*X4*	2000	1.0	0.154	0.154

На основі даних, наведених у табл. 3.6, за допомогою формули:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \tag{3.12}$$

було визначено загальний рівень якості для кожного з варіантів реалізації функцій чат-бота.

Розрахунки дали наступні результати:

$$K_{K1} = 0{,}724 + 0{,}862 + 0{,}876 = 20{,}4\,;$$

$$K_{K2} = 0{,}724 + 0{,}862 + 0{,}144 = 16{,}08\,.$$

Як видно з проведених розрахунків, кращим є перший варіант, оскільки для нього коефіцієнт рівня якості має вище значення. Це підтверджує ефективність обраного підходу до реалізації ключових функцій чат-бота.

Таким чином, використання inline-кнопок для інтерактивного інтерфейсу, динамічного завантаження питань з бази даних та централізованого збереження відповідей у БД є найбільш оптимальним рішенням для даної системи.

3.5 Економічне обґрунтування варіантів розробки ПЗ

Для визначення вартості розробки чат-бота, що використовується для проведення опитувань здобувачів вищої освіти, необхідно оцінити трудомісткість проєкту. Вона включає витрати на виконання ключових завдань, пов’язаних із реалізацією функціональності бота, що дозволяє оцінити часові та ресурсні затрати.

Процес розробки складається з двох основних завдань:

– проектування архітектури та функціональної концепції чат-бота;

– розробка програмної оболонки, включаючи реалізацію алгоритмів взаємодії з користувачем та обробки відповідей.

Перше завдання (група А) є складнішим, оскільки передбачає проектування структури бази даних, визначення логіки обробки запитів користувачів та інтеграцію з Telegram API. З точки зору алгоритмічної складності, воно належить до групи 1 (найвищий рівень складності).

Друге завдання (група Б) пов’язане із реалізацією коду бота, включаючи впровадження основних команд, обробку запитів та підключення до бази даних. Це менш складний процес, що належить до групи 3.

Розрахунок загальної трудомісткості проєкту здійснюється за формулою:

$$\mathrm{Т_О} = \mathrm{Т_Р} \cdot \mathrm{К_П} \cdot \mathrm{К_{СК}} \cdot \mathrm{К_М} \cdot \mathrm{К_{СТ}} \cdot \mathrm{К_{СТ.М}}, \quad (3.13)$$

де:

T_P – базова трудомісткість розробки програмного забезпечення;

$K_П$ – поправочний коефіцієнт, що враховує специфіку проєкту;

$K_{СК}$ – коефіцієнт, що враховує складність обробки вхідної інформації;

K_M – коефіцієнт, що залежить від рівня мови програмування та використовуваних технологій;

K_{CT} – коефіцієнт застосування стандартних модулів та готових бібліотек;

$K_{CT.M}$ – коефіцієнт використання стандартного математичного забезпечення.

Для визначення загальної трудомісткості процесу розробки чат-бота проведено розрахунки з урахуванням рівня складності завдань, рівня їхньої новизни та застосування стандартних модулів.

Розрахунок трудомісткості першого завдання

Перше завдання включає розробку архітектури чат-бота, що включає проектування бази даних, визначення основних алгоритмів обробки запитів та інтеграцію з Telegram API. Це складний процес, що потребує високої кваліфікації, тому воно належить до групи А (високий рівень новизни) та групи 1 (найскладніші алгоритми).

Відповідно до нормативних часових витрат, базова трудомісткість визначена як T_P =90 людино-днів. Враховуючи, що при виконанні завдання використовуються нормативно-довідкові матеріали, вводиться поправочний коефіцієнт $K_П$ = 1.7. Оскільки складність обробки інформації оцінена як середня, встановлено $K_{СК}$ = 1. Застосування стандартних бібліотек та програмних модулів дає змогу зменшити трудомісткість, використовуючи коефіцієнт K_{CT} = 0.8.

Загальна трудомісткість першого завдання визначається як:

$$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122{,}40 \text{ людино–днів.}$$

Друге завдання включає реалізацію програмної частини бота, що передбачає написання коду, обробку користувацьких команд, формування відповідей та роботу з базою даних. Це завдання має середню складність (група Б) і відноситься до третьої групи алгоритмічної складності.

Відповідно, базова трудомісткість становить T_P = 27 людино-днів. Поправочний коефіцієнт для цієї групи $K_П$ = 0.9, а коефіцієнт застосування стандартних програмних модулів K_{CT} = 0.8.

$$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44 \text{ людино–днів.}$$

Агрегуючи трудомісткість усіх завдань, отримуємо загальну оцінку трудовитрат у людино-годинах для кожного варіанта реалізації:

$$T_I = (122{,}4 + 19.44 + 4.8 + 19.44) \cdot 8 = 1228{,}6 \text{ людино–годин.}$$

$$T_{II} = (122{,}4 + 19.44 + 6.91 + 19.44) \cdot 8 = 1322{,}52 \text{ людино–годин.}$$

Розрахунки показують, що перший варіант реалізації чат-бота (з використанням inline-кнопок, динамічного завантаження запитань з БД та централізованого збереження відповідей) є менш трудомістким. Він дозволяє зменшити витрати на розробку та спростити підтримку програмного продукту у майбутньому.

Розробка програмного забезпечення потребує залучення фахівців відповідної кваліфікації. У процесі створення чат-бота беруть участь два програмісти з місячними окладами 15 000 грн та 14 000 грн, а також аналітик даних з окладом 18 500 грн.

Розрахунок погодинної ставки

Для визначення годинної заробітної плати використовується формула:

$$\text{С}_\text{ч} = \frac{\text{М}}{T_m \cdot t} \text{ грн.}, \tag{3.14}$$

де:

M – місячний оклад спеціаліста,

T_m – кількість робочих днів на тиждень (5 днів),

t – тривалість робочого дня (8 годин).

Розрахуємо середню погодинну оплату для залучених спеціалістів:

$$\text{С}_\text{ч} = \frac{14000 + 13000 + 18800}{3 \cdot 21 \cdot 8} = 90{,}87 \text{ грн.}$$

Загальні витрати на заробітну плату розраховуються за формулою:

$$\text{С}_\text{ЗП} = \text{С}_\text{ч} \cdot \text{Т}_i \cdot \text{К}_\text{Д}, \tag{3.15}$$

де:

$\text{С}_\text{ч}$ – погодинна ставка;

Т_i – трудомісткість завдання (у год.);

$\text{К}_\text{Д}$ – коефіцієнт врахування додаткових виплат.

Розрахуємо заробітну плату для двох варіантів реалізації:

I. $С_{ЗП} = 90,87 \cdot 1228,64 \cdot 1.2 = 133975,82$ грн.

II. $С_{ЗП} = 90,87 \cdot 1322.52 \cdot 1.2 = 14421287,09$ грн.

Відрахування на єдиний соціальний внесок (ЄСВ) становлять 22% від суми заробітної плати:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 133975,82 \cdot 0.22 = 29474,68$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 144212,87 \cdot 0.22 = 31726,83$ грн.

Для оцінки витрат, пов'язаних із використанням комп'ютерної техніки під час розробки чат-бота, проводиться розрахунок витрат на одну машино-годину.

Для визначення витрат на оплату машино-години використовується програміст із місячним окладом 14 000 грн та коефіцієнтом зайнятості комп'ютера 0.2.

$$C_Г = 12 \cdot M \cdot K_З = 12 \cdot 14000 \cdot 0,2 = 33600 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$C_{ЗП} = C_Г \cdot (1 + K_З) = 33600 \cdot \cdot (1 + 0.2) = 40320 \text{ грн.}$$

Відрахування на соціальний внесок (22% від заробітної плати):

$$C_А = K_{ТМ} \cdot K_А \cdot Ц_{ПР} = 1.4 \cdot 0.25 \cdot 10000 = 3500 \text{ грн.},$$

Вартість комп'ютера – 10 000 грн, коефіцієнт витрат на транспортування та монтаж $K_{ТМ}$ =1.4, річна норма амортизації $K_А$ =0.25:

$$С_А = K_{ТМ} \cdot K_А \cdot Ц_{ПР} = 1.4 \cdot 0.25 \cdot 10000 = 3500 \text{ грн.},$$

Витрати на поточний ремонт:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 10000 \cdot 0.25 = 575 \text{ грн.},$$

Загальна кількість робочих годин комп'ютера за рік розраховується за формулою:

$$Т_{ЕФ} = (Д_К - Д_В - Д_С - Д_Р) \cdot t_З \cdot К_В, \quad (3.16)$$

де:

$Д_К$ – загальна кількість календарних днів у році;

$Д_В$ – кількість вихідних днів (104);

$Д_С$ – святкові дні (8);

$Д_Р$ – дні ремонту (16);

t – кількість робочих годин у день (8);

$К_В$ – коефіцієнт використання обладнання (0.9).

$$T_{ЕФ} = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4 \text{ години.}$$

Розрахунок витрат на електроенергію:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_C \cdot К_З \cdot Ц_{ЕН}, \quad (3.17)$$

де:

N_C – середньоспоживча потужність обладнання;

$К_З$ – коефіцієнт зайнятості обладнання;

$Ц_{ЕН}$ – тариф за 1 кВт·год.

$$C_{ЕЛ} = = 1706{,}4 \cdot 0{,}2 \cdot 0.3\ 5{,}23 = 535{,}47 \text{ грн.},$$

Розрахунок накладних витрат:

$$C_Н = Ц_{ПР} \cdot 0.67 = 10000 \cdot 0{,}67 = 6700 \text{ грн.}$$

Розрахунок щорічних експлуатаційних витрат:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н, \quad (3.18)$$

$$C_{ЕКС} = 40320+8870{,}4+3500+575+535{,}47+6700=60497{,}8 \text{ грн.}$$

Оскільки всі роботи ведуться на комп’ютері, витрати на оплату машинного часу визначаються за формулою:

$$С_М = С_{М-Г} \cdot Т, \quad (3.19)$$

I. $C_M = 35{,}45 \cdot 1228{,}64 = 43555{,}29$ грн.

II. $C_M = 35{,}45 \cdot 1322{,}52 = 46883{,}33$ грн.

Накладні витрати складають 67% від заробітної плати:

I. $C_Н = 133975{,}82 \cdot 0{,}67 = 89763{,}8$ грн.

II. $C_Н = 144212{,}87 \cdot 0{,}67 = 96622{,}62$ грн.

Загальна вартість розробки програмного продукту визначається як:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н, \quad (3.20)$$

I. $C_{ПП} = 133975{,}82 + 29474{,}68 + 43555{,}29 + 89763{,}8 = 296769{,}59$ грн.

II. $C_{ПП} = 144212{,}87 + 31726{,}83 + 46883{,}33 + 96622{,}62 = 319445{,}65$ грн.

Отримані результати дозволяють оцінити загальну вартість розробки чат-бота для кожного з розглянутих варіантів. Перший варіант є більш економічно вигідним, оскільки загальна вартість його розробки менша, ніж у другого варіанта. Це підтверджує доцільність його вибору, оскільки він дозволяє оптимізувати фінансові витрати без значного впливу на якість розробки.

3.6 Обрання оптимального варіанту ПЗ на основі техніко-економічних показників

Для визначення ефективності кожного варіанта реалізації програмного забезпечення розраховується коефіцієнт техніко-економічного рівня (КТЕР). Він визначається як відношення рівня якості розробленого продукту до його фінансової вартості. Розрахунок здійснюється за формулою:

$$К_{ТЕР j} = К_{Кj} / С_{Фj}, \quad (3.21)$$

де:

$К_{ТЕР j}$ – коефіцієнт техніко-економічного рівня для jj-го варіанта,

$К_{Кj}$ – коефіцієнт рівня якості для jj-го варіанта,

$С_{Фj}$ – загальна вартість розробки програмного забезпечення для jj-го варіанта.

Розрахуємо значення для кожного варіанта:

$$К_{ТЕР1} = 2{,}57 / 296769{,}59 = 8{,}65 \cdot 10^{-6},$$

$$К_{ТЕР2} = 1{,}89 / 319445{,}65 = 5{,}91 \cdot 10^{-6}.$$

Як видно з розрахунків, найбільш ефективним є перший варіант реалізації чат-бота, оскільки його коефіцієнт техніко-економічного рівня є вищим, що свідчить про оптимальне співвідношення між якістю реалізації та фінансовими витратами. Це підтверджує доцільність вибору першого варіанта для подальшого впровадження, оскільки він забезпечує вищу продуктивність та економічну ефективність.

3.7 Висновок

У даному розділі проведено аналіз та оцінку варіантів реалізації чат-бота для опитування здобувачів вищої освіти, визначено оптимальні параметри програмного забезпечення, оцінено трудомісткість розробки, розраховано фінансові витрати та виконано техніко-економічне обґрунтування вибору найбільш ефективного варіанта.

Розрахунок ключових параметрів системи показав, що швидкість обробки запитів, час відповіді бота та ефективність використання пам’яті є найбільш значущими для продуктивності чат-бота. Попарне порівняння параметрів та експертне оцінювання дозволило визначити найкращі технологічні рішення, які забезпечують високу якість розробки.

На основі аналізу трудовитрат було розраховано загальну трудомісткість проєкту, що становить 1228,6 людино-годин для першого варіанта та 1322,52 людино-годин для другого варіанта. Розрахунок витрат на заробітну плату, оплату машинного часу та накладних витрат показав, що загальна вартість розробки першого варіанта (296769,59 грн) є нижчою, ніж у другого (319445,65 грн).

Фінальним етапом дослідження стало визначення коефіцієнта техніко-економічного рівня (КТЕР), який вказує на оптимальне співвідношення між якістю програмного забезпечення та його вартістю. Перший варіант отримав вищий коефіцієнт K_{TEP1}= 8,65 $\cdot 10^{-6}$, що свідчить про його економічну ефективність та доцільність впровадження.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було проведено комплексне дослідження, спрямоване на розробку та аналіз чат-бота для опитування здобувачів вищої освіти. В результаті було визначено сучасні технології збору зворотного зв’язку в освітньому середовищі, досліджено існуючі рішення у сфері опитувань та виявлено їхні переваги й недоліки. Аналіз існуючих платформ для розробки чат-ботів дозволив обґрунтовано обрати Telegram Bot API як найбільш зручний та гнучкий інструмент, що відповідає вимогам доступності, продуктивності та інтеграції з базою даних.

Було проведено проектування чат-бота з урахуванням особливостей взаємодії користувачів. Розроблено діаграми варіантів використання, які визначають функціональні можливості системи, та бізнес-процеси, що відображають послідовність дій для адміністрування та проведення опитувань. Реалізовано програмні модулі, що забезпечують коректну роботу бота, включаючи ініціалізацію, обробку команд користувачів, збереження відповідей у базі даних та формування аналітичної звітності. Проведено тестування функціональності, яке підтвердило відповідність реалізованого рішення встановленим вимогам.

З метою оцінки ефективності розробленого програмного забезпечення виконано аналіз функціональних параметрів чат-бота, що включав визначення ключових характеристик, таких як швидкість обробки запитів, використання пам’яті, час відповіді та складність коду. Виконано експертне оцінювання параметрів, яке дозволило визначити найбільш значущі аспекти, що впливають на якість роботи бота. Проведено аналіз альтернативних варіантів реалізації основних функцій, на основі чого було вибрано найбільш оптимальне рішення, що забезпечує високу швидкість роботи, зручність взаємодії користувачів та ефективність збору даних.

Розраховано техніко-економічні показники двох варіантів реалізації чат-бота, що дозволило оцінити фінансові витрати на розробку та порівняти їх із

рівнем якості програмного продукту. Визначено, що перший варіант реалізації, який включає використання inline-кнопок для взаємодії, динамічне завантаження запитань із бази даних та централізоване збереження відповідей, є більш економічно вигідним і має вищий коефіцієнт техніко-економічного рівня. Це підтверджує доцільність його вибору для подальшого впровадження.

Отримані результати можуть бути використані для подальшого вдосконалення систем автоматизованого збору зворотного зв’язку в освітньому середовищі. Запропонований підхід до проектування чат-бота може бути адаптований для інших сфер, що потребують швидкого та зручного збору даних від користувачів. В перспективі можливим є розширення функціональності бота за рахунок впровадження машинного навчання для аналізу відповідей та автоматичної генерації рекомендацій на основі отриманих даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Жихорська, О. ВИКОРИСТАННЯ ЧАТ-БОТІВ У НАВЧАННІ СТУДЕНТІВ. Вісник Київського національного університету імені Тараса Шевченка. Серія «Педагогіка». 2024. Вип. 2, №18.

2. Role of AI chatbots in education: systematic literature review / Education and Information Technologies. – 2023. . URL: (дата звернення: 15.01.2025).

3. Грицюк Ю. В., Грицюк О. В. Онлайн-анкетування як елемент зворотного зв'язку при побудові хмарного освітнього середовища // Матеріали конференції. – 2014. – С. 1–4.

4. Шульга А., Дідух В., Гатрич І. Використання засобів зворотного зв'язку під час дистанційного навчання студентів закладів вищої освіти // Науковий вісник Вінницької академії безперервної освіти. Серія «Педагогіка. Психологія». – 2023. – №3. – С. 15.

5. Гоманюк М. А. Інтерв'юер у масовому опитуванні: Кишеньковий навчально-методичний посібник. – Херсон: Гілея, 2013. – 84 с.

6. Internet-mediated phone interviews for remote qualitative educational research / International Journal of Social Research Methodology. – 2024. URL: https://www.tandfonline.com/doi/pdf/10.1080/13645579.2024.2421766 (дата звернення: 15.01.2025).

7. Використання онлайн-сервісу Google Forms в освітньому процесі закладу загальної середньої освіти. URL: https://www.academia.edu/91095992/Використання_онлайн_сервісу_Google_forms_в_освітньому_процесі_закладу_загальної_середньої_освіти (дата звернення: 15.01.2025).

8. Чаплінська Ю. С. Можливості використання чат-ботів в освітній сфері // Медіаосвіта. URL: https://mediaosvita.org.ua/wp-content/uploads/2020/07/CHaplinska_YUS_Vykorystannya_chat-botiv_v_osviti.pdf (дата звернення: 15.01.2025).

9. Hasan M. R., Chowdhury N. I., Rahman M. H., Syed M. A. B., Ryu J. Analysis of the User Perception of Chatbots in Education Using A Partial Least Squares Structural Equation Modeling Approach // arXiv preprint arXiv:2311.03636. – 2023. URL: https://arxiv.org/abs/2311.03636 (дата звернення: 15.01.2025).

10. Belda-Medina J., Kokošková V. Integrating chatbots in education: insights from the Chatbot-Human Interaction Satisfaction Model (CHISM) // International Journal of Educational Technology in Higher Education. – 2023. URL: https://educationaltechnologyjournal.springeropen.com/articles/10.1186/s41239-023-00432-3 (дата звернення: 15.01.2025).

11. Surveybot: Facebook Messenger Chatbot for Facebook Survey. URL: https://surveybot.io/ (дата звернення: 15.01.2025).

12. About Surveybot. URL: https://sourceforge.net/software/product/Surveybot/ (дата звернення: 15.01.2025).

13. SurveyBot. URL: https://roi4cio.com/catalog/de/product/surveybot (дата звернення: 15.01.2025).

14. Reviews of SurveyBot. URL: https://www.capterra.com/p/152563/SurveyBot/reviews/ (дата звернення: 15.01.2025).

15. Surveybot Reviews 2024. URL: https://slashdot.org/software/p/Surveybot/ (дата звернення: 15.01.2025).

16. Key Features of Surveybot. URL: https://www.softwaresuggest.com/surveybot (дата звернення: 15.01.2025).

17. Surveybot Reviews & Product Details. URL: https://www.g2.com/products/surveybot/reviews (дата звернення: 15.01.2025).

18. SurveySparrow. URL: https://www.glassdoor.com/Reviews/SurveySparrow-Reviews-E2194468.htm (дата звернення: 15.01.2025).

19. SurveySparrow App. URL: https://appsource.microsoft.com/en-us/product/office/wa200002257?tab=overview (дата звернення: 15.01.2025).

20. SurveySparrow Reviews. URL: https://www.softwareadvice.com/survey/surveysparrow-profile/reviews/ (дата звернення: 15.01.2025).

21. SurveySparrow. URL: https://www.trustpilot.com/review/surveysparrow.com (дата звернення: 15.01.2025).

22. What is SurveySparrow? URL: https://www.trustradius.com/products/surveysparrow/reviews (дата звернення: 15.01.2025).

23. SurveySparrow Reviews & Product Details. URL: https://www.g2.com/products/surveysparrow/reviews (дата звернення: 15.01.2025).

24. Reviews of SurveySparrow. URL: https://www.capterra.com/p/168823/SurveySparrow/reviews/ (дата звернення: 15.01.2025).

25. About SurveySparrow. URL: https://www.softwareadvice.com/survey/surveysparrow-profile/ (дата звернення: 15.01.2025).

26. About Poll Bot. URL: https://www.softwareadvice.com/polling/poll-bot-profile/ (дата звернення: 15.01.2025).

27. Pollbot | Webex App Hub. URL: https://apphub.webex.com/applications/pollbot-cisco-systems-12150-78220-99857 (дата звернення: 15.01.2025).

28. Poll Bot. It’s easy to create functional WhatsApp surveys by Poll Bot. URL: https://www.producthunt.com/products/poll-bot (дата звернення: 15.01.2025).

29. Poll Bot Overview. URL: https://www.softwareworld.co/software/poll-bot-reviews/ (дата звернення: 15.01.2025).

30. Pollbot. URL: https://www.techimply.com/reviews-add/pollbot (дата звернення: 15.01.2025).

31. Modrzyk, N. Building Telegram Bots: Develop Bots in 12 Programming Languages using the Telegram Bot API. Apress, 2018. 512 p.

32. Sajjapanroj, S., Longpradit, P., & Polanunt, K. A Prototype of Google Dialog Flow for School Teachers' Uses in Conducting Classroom Research. Asian Journal of Distance Education. 2021, Vol. 15, No 2, pp. 133–142.

33. Chatfuel Review - 2024. URL: https://www.thesamur.ai/learn/chatfuel-review---2024 (дата звернення: 15.01.2025).

34. Балабанов О.І., Павленко А.П. PyCharm для розробників Python: Навчальний посібник. - Львів: Видавництво Львівської політехніки, 2018. - 300 с.

35. Карапецький В. П. Побудова графічного контенту додатків з використанням JavaFX і Swing компонентів і даних, взятих із баз даних / В. П. Карапецький. – Науковий вісник НЛТУ. – 2015. – 418 с.

36. Albahari B., Albahari J. C# 6. 0 in a Nutshell: The Definitive Reference. O'Reilly Media, Incorporated, 2015. 1136 p.

37. Томашевський О. М., Цегелик Г. Г., Вітер М. Б., Дубук В. І. Інформаційні технології та моделювання бізнес-процесів. Київ : «Центр учбової літератури», 2012. 296 с.

38. Табунщик Г. В. Проектування, моделювання та аналіз інформаційних систем: Навчальний посібник / Г.В. Табунщик, Р.К. Кудерметов, А. В. Притула. – Запоріжжя : ЗНТУ, 2011. – 292 с.

39. Томашевський О. М. Інформаційні технології та моделювання бізнес-процесів : навч. посіб. / О. М. Томашевський, Г. Г. Цигелик, М. Б. Вітер, В. І. Дудук. – К. : Центр учбової літератури, 2012. – 296 с.

40. Охріменко О.О. Функціонально-вартісний аналіз: навчальний посібник. Київ: Освіта України, 2013. 208 с.

ДОДАТКИ

Додаток А. Скрипти бази даних

```
CREATE TABLE [dbo].[LeavingFeedback]
(
[LeavingFeedbackId] INT NOT NULL PRIMARY KEY IDENTITY,
   [ChatId] BIGINT NULL,
   [FirstName] NVARCHAR(150) NULL,
   [LastName] NVARCHAR(150) NULL,
   [FeedbackDate] DATETIME NULL,
   [UserMessage] NVARCHAR(MAX) NULL,
   CONSTRAINT FK_LeavingFeedback_Users FOREIGN KEY ([ChatId]) REFERENCES [dbo].[Users]([UsersId]) ON DELETE SET NULL
);

CREATE TABLE [dbo].[Topics]
(
[TopicId] INT NOT NULL PRIMARY KEY IDENTITY,
   [TopicTitle] NVARCHAR(500) NULL,
   [Description] NVARCHAR(MAX) NULL,
   [CreatedDate] DATETIME NULL
);

CREATE TABLE [dbo].[Questions]
(
[QuestionId] INT NOT NULL PRIMARY KEY IDENTITY,
   [TopicId] INT NULL,
   [QuestionsText] NVARCHAR(MAX) NULL,
   [Option1] NVARCHAR(MAX) NULL,
   [Option2] NVARCHAR(MAX) NULL,
   [Option3] NVARCHAR(MAX) NULL,
   [Option4] NVARCHAR(MAX) NULL,
```

```
    CONSTRAINT FK_Questions_Topics FOREIGN KEY ([TopicId]) REFERENCES [dbo].[Topics]([TopicId]) ON DELETE SET NULL
    );

    CREATE TABLE [dbo].[UserSurvey]
    (
    [SurveyId] INT NOT NULL PRIMARY KEY IDENTITY,
      [TopicId] INT NULL,
      [ChatId] BIGINT NULL,
      [FirstName] NVARCHAR(150) NULL,
      [LastName] NVARCHAR(150) NULL,
      [SurveyDate] DATETIME NULL,
      CONSTRAINT FK_UserSurvey_Topics FOREIGN KEY ([TopicId]) REFERENCES [dbo].[Topics]([TopicId]) ON DELETE SET NULL,
      CONSTRAINT FK_UserSurvey_Users FOREIGN KEY ([ChatId]) REFERENCES [dbo].[Users]([UsersId]) ON DELETE SET NULL
    );

    CREATE TABLE [dbo].[Responses]
    (
    [ResponseId] INT NOT NULL PRIMARY KEY IDENTITY,
      [SurveyId] INT NULL,
      [QuestionId] INT NULL,
      [Answer] NVARCHAR(MAX) NULL,
      [ResponseDate] DATETIME NULL,
      CONSTRAINT FK_Responses_UserSurvey FOREIGN KEY ([SurveyId]) REFERENCES [dbo].[UserSurvey]([SurveyId]) ON DELETE CASCADE,
      CONSTRAINT FK_Responses_Questions FOREIGN KEY ([QuestionId]) REFERENCES [dbo].[Questions]([QuestionId]) ON DELETE CASCADE
    );

    CREATE TABLE [dbo].[Users] (
      [UsersId]    INT          IDENTITY (1, 1) NOT NULL,
      [FirstName]   NVARCHAR (60)  NULL,
      [LastName]    NVARCHAR (60)  NULL,
```

```
    [UsersName]     NVARCHAR (60)   NULL,
    [UsersPassword] NVARCHAR (250)  NULL,
    [RoleId]        INT             NULL,
    [Description]   NVARCHAR (1200) NULL,
    [Email]         NVARCHAR (150)  NULL,
    PRIMARY KEY CLUSTERED ([UsersId] ASC)
);

CREATE TABLE [dbo].[Logs] (
    [LogsId]        INT             IDENTITY (1, 1) NOT NULL,
    [UsersId]       INT             NULL,
    [EventNameShow] NVARCHAR (MAX) NULL,
    [EventDate]     DATETIME        NULL,
    PRIMARY KEY CLUSTERED ([LogsId] ASC),
    CONSTRAINT FK_Logs_Users FOREIGN KEY ([UsersId]) REFERENCES
[dbo].[Users]([UsersId]) ON DELETE SET NULL
);
```

Додаток Б. Лістинги програми

Лістинг 1. Код класу «QuestionsProvider.cs»

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SurveyOfAcquirersApp.AppCode;

namespace SurveyOfAcquirersApp.Providers {
  internal class QuestionsProvider {
    private string _ConnString =
System.Configuration.ConfigurationSettings.AppSettings["CONNECT"];

    public void InsertQuestions(int TopicId, string QuestionsText, string Option1, string Option2,
string Option3, string Option4) {
      string SqlString = "INSERT INTO Questions (TopicId, QuestionsText, Option1, Option2,
Option3, Option4) " +
                    "VALUES (@TopicId, @QuestionsText, @Option1, @Option2, @Option3,
@Option4)";

      using (SqlConnection conn = new SqlConnection(_ConnString)) {
        using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
          cmd.CommandType = CommandType.Text;

          // Додавання параметрів
          cmd.Parameters.AddWithValue("@TopicId", TopicId);
          cmd.Parameters.AddWithValue("@QuestionsText", QuestionsText);
          cmd.Parameters.AddWithValue("@Option1", Option1);
          cmd.Parameters.AddWithValue("@Option2", Option2);
          cmd.Parameters.AddWithValue("@Option3", Option3);
          cmd.Parameters.AddWithValue("@Option4", Option4);

          conn.Open();
          cmd.ExecuteNonQuery();
          conn.Close();
        }
      }
    }


    public List<Questions> GetAllQuestions() {
      int num = 0;
      string SqlString = "SELECT * FROM Questions ORDER BY QuestionId";

      List<Questions> listAllQuestions = new List<Questions>();
```

```
    using (SqlConnection conn = new SqlConnection(_ConnString)) {
      using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
        conn.Open();
        using (SqlDataReader reader = cmd.ExecuteReader()) {
          while (reader.Read()) {
            Questions oneQuestion = new Questions();

            // Заповнення полів об'єкта
            oneQuestion.Number = ++num;
            oneQuestion.QuestionId = Convert.ToInt32(reader["QuestionId"]);
            oneQuestion.TopicId = Convert.ToInt32(reader["TopicId"]);
            oneQuestion.QuestionsText = reader["QuestionsText"].ToString();
            oneQuestion.Option1 = reader["Option1"].ToString();
            oneQuestion.Option2 = reader["Option2"].ToString();
            oneQuestion.Option3 = reader["Option3"].ToString();
            oneQuestion.Option4 = reader["Option4"].ToString();

            listAllQuestions.Add(oneQuestion);
          }
        }
        conn.Close();
      }
    }

    // Якщо немає даних, додаємо повідомлення
    if (listAllQuestions.Count == 0) {
      Questions noQuestion = new Questions();
      noQuestion.QuestionId = 0;
      noQuestion.Message = NamesMy.NoDataNames.NoDataInQuestions;
      listAllQuestions.Add(noQuestion);
    }

    return listAllQuestions;
  }



  public List<Questions> GetAllQuestionsByTopicId(int TopicId) {
    int num = 0;
    string SqlString = "SELECT * FROM Questions WHERE TopicId = @TopicId ORDER BY
QuestionId";

    List<Questions> listAllQuestions = new List<Questions>();

    using (SqlConnection conn = new SqlConnection(_ConnString)) {
      using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
        cmd.Parameters.AddWithValue("@TopicId", TopicId); // Додавання параметра для
фільтрації

        conn.Open();
        using (SqlDataReader reader = cmd.ExecuteReader()) {
          while (reader.Read()) {
```

```
            Questions oneQuestion = new Questions();

            // Заповнення полів об'єкта
            oneQuestion.Number = ++num;
            oneQuestion.QuestionId = Convert.ToInt32(reader["QuestionId"]);
            oneQuestion.TopicId = Convert.ToInt32(reader["TopicId"]);
            oneQuestion.QuestionsText = reader["QuestionsText"].ToString();
            oneQuestion.Option1 = reader["Option1"].ToString();
            oneQuestion.Option2 = reader["Option2"].ToString();
            oneQuestion.Option3 = reader["Option3"].ToString();
            oneQuestion.Option4 = reader["Option4"].ToString();

            listAllQuestions.Add(oneQuestion);
          }
        }
        conn.Close();
      }
    }

    // Якщо немає даних, додаємо повідомлення
    if (listAllQuestions.Count == 0) {
      Questions noQuestion = new Questions();
      noQuestion.QuestionId = 0;
      noQuestion.Message = NamesMy.NoDataNames.NoDataInQuestions;
      listAllQuestions.Add(noQuestion);
    }
    return listAllQuestions;
  }



  public Questions SelectedQuestionByQuestionId(int QuestionId) {
    string SqlString = "SELECT * FROM Questions WHERE QuestionId=@QuestionId";

    Questions oneQuestion = new Questions();
    using (SqlConnection conn = new SqlConnection(_ConnString)) {
      using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
        cmd.Parameters.AddWithValue("@QuestionId", QuestionId);
        conn.Open();
        using (SqlDataReader reader = cmd.ExecuteReader()) {
          while (reader.Read()) {
            oneQuestion.QuestionId = Convert.ToInt32(reader["QuestionId"]);
            oneQuestion.TopicId = Convert.ToInt32(reader["TopicId"]);
            oneQuestion.QuestionsText = reader["QuestionsText"].ToString();
            oneQuestion.Option1 = reader["Option1"].ToString();
            oneQuestion.Option2 = reader["Option2"].ToString();
            oneQuestion.Option3 = reader["Option3"].ToString();
            oneQuestion.Option4 = reader["Option4"].ToString();
          }
        }
        conn.Close();
      }
```

```
      }
      return oneQuestion;
    }


    public void UpdateQuestion(int TopicId, string QuestionsText, string Option1, string Option2,
string Option3, string Option4, int QuestionId) {
      string SqlString = "UPDATE Questions SET TopicId=@TopicId,
QuestionsText=@QuestionsText, " +
                    "Option1=@Option1, Option2=@Option2, Option3=@Option3,
Option4=@Option4 WHERE QuestionId=@QuestionId";

      using (SqlConnection conn = new SqlConnection(_ConnString)) {
        using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
          cmd.CommandType = CommandType.Text;

          // Додавання параметрів
          cmd.Parameters.AddWithValue("@QuestionId", QuestionId);
          cmd.Parameters.AddWithValue("@TopicId", TopicId);
          cmd.Parameters.AddWithValue("@QuestionsText", QuestionsText);
          cmd.Parameters.AddWithValue("@Option1", Option1);
          cmd.Parameters.AddWithValue("@Option2", Option2);
          cmd.Parameters.AddWithValue("@Option3", Option3);
          cmd.Parameters.AddWithValue("@Option4", Option4);

          conn.Open();
          cmd.ExecuteNonQuery();
          conn.Close();
        }
      }
    }

    public void DeleteQuestionByQuestionId(int QuestionId) {
      string SqlString = "DELETE FROM Questions WHERE QuestionId=@QuestionId";

      using (SqlConnection conn = new SqlConnection(_ConnString)) {
        using (SqlCommand cmd = new SqlCommand(SqlString, conn)) {
          cmd.Parameters.AddWithValue("@QuestionId", QuestionId);
          conn.Open();
          cmd.ExecuteNonQuery();
          conn.Close();
        }
      }
    }


  }
}

// Клас Questions (Запитання)
public class Questions {
  // Номер питання (обов'язковий атрибут)
```

```
private int _Number;
// Унікальний ідентифікатор питання
private int _QuestionId;
// Ідентифікатор теми
private int _TopicId;
// Текст питання
private string _QuestionsText;
// Варіант відповіді 1
private string _Option1;
// Варіант відповіді 2
private string _Option2;
// Варіант відповіді 3
private string _Option3;
// Варіант відповіді 4
private string _Option4;
// Повідомлення (обов'язковий атрибут)
private string _Message;

// Конструктор за замовчуванням
public Questions() {
  _Number = 0;
  _QuestionId = 0;
  _TopicId = 0;
  _QuestionsText = string.Empty;
  _Option1 = string.Empty;
  _Option2 = string.Empty;
  _Option3 = string.Empty;
  _Option4 = string.Empty;
  _Message = string.Empty;
}

// Властивість Number
public int Number {
  set { _Number = value; }
  get { return _Number; }
}

// Властивість QuestionId
public int QuestionId {
  set { _QuestionId = value; }
  get { return _QuestionId; }
}

// Властивість TopicId
public int TopicId {
  set { _TopicId = value; }
  get { return _TopicId; }
}

// Властивість QuestionsText
public string QuestionsText {
  set { _QuestionsText = value; }
```

```
    get { return _QuestionsText; }
  }

  // Властивість Option1
  public string Option1 {
    set { _Option1 = value; }
    get { return _Option1; }
  }

  // Властивість Option2
  public string Option2 {
    set { _Option2 = value; }
    get { return _Option2; }
  }

  // Властивість Option3
  public string Option3 {
    set { _Option3 = value; }
    get { return _Option3; }
  }

  // Властивість Option4
  public string Option4 {
    set { _Option4 = value; }
    get { return _Option4; }
  }

  // Властивість Message
  public string Message {
    set { _Message = value; }
    get { return _Message; }
  }
}
```

Лістинг 2. Код класу «TelegramBotForm.cs»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Telegram.Bot;
using Telegram.Bot.Args;
using Telegram.Bot.Types.InputFiles;
using Telegram.Bot.Types.ReplyMarkups;

namespace SurveyOfAcquirersApp.Forms.Controls {
  public partial class TelegramBotForm : Form {
```

```
    // Дані бота (назва, username та токен з технічного завдання)
    private string _botName = "EduSurvey";
    private string _botUserName = "EduSurveyBot";

    // Токен бота (з технічного завдання)
    private string _token = "7153237731:AAFyT2gqtx9qsV9cl2vIdEIvUf_F_GjC8sY";

    // Основний клієнт Telegram
    private ITelegramBotClient _botClient;

    // Словник станів користувачів (chatId => стан користувача)
    private Dictionary<long, BotUserState> userStates = new Dictionary<long, BotUserState>();

    // Сервіс для доступу до бази даних (Topics, Questions, Responses, UserSurvey,
LeavingFeedback)
    private DataService _dataService;
    public TelegramBotForm() {
      InitializeComponent();
      // Ініціалізація сервісу БД(припустимо, що у Вас уже реалізований DataService)
          string connString =
System.Configuration.ConfigurationSettings.AppSettings["CONNECT"];
      _dataService = new DataService(connString);
    }

    private void TelegramBotForm_Load(object sender, EventArgs e) {
      // Створюємо клієнт бота
      _botClient = new TelegramBotClient(_token);

      // Підписуємося на події
      _botClient.OnMessage += BotClient_OnMessage;
      _botClient.OnCallbackQuery += BotClient_OnCallbackQuery;

      // Запускаємо отримання повідомлень
      _botClient.StartReceiving();

      // Для прикладу: виведемо у вікно форми інформацію про запуск
      RaportTBox.Text = $"Бот '{_botName}' (@{_botUserName}) запущено...\r\n";
    }


    /// <summary>
    /// Обробник звичайних повідомлень (текст, медіа тощо).
    /// </summary>
    private async void BotClient_OnMessage(object sender, MessageEventArgs e) {
      // Перевірка, чи є хоч якесь текстове повідомлення
      if (e.Message?.Text == null)
        return;
      long chatId = e.Message.Chat.Id;
      string userMessage = e.Message.Text;
      // Лог у вікні
      RaportTBox.Invoke((MethodInvoker)delegate {
        RaportTBox.Text += $"[{chatId}]: {userMessage}\r\n";
```

```
    });
    // Якщо вперше бачимо цього користувача — ініціалізуємо його стан
    if (!userStates.ContainsKey(chatId)) {
      userStates[chatId] = new BotUserState();
    }
    var userState = userStates[chatId];
    // Якщо користувач у режимі введення відгуку (IsLeavingFeedback = true)
    if (userState.IsLeavingFeedback) {
      // Зберігаємо відгук у базу
      await SaveFeedback(chatId, e.Message);
      // Скидаємо прапорець
      userState.IsLeavingFeedback = false;
      // Повертаємось у головне меню
      await SendMainMenu(chatId);
      return;
    }

    // Якщо користувач у розпалі опитування (IsSurveyInProgress = true)
    // — обробляємо відповіді на запитання
    if (userState.IsSurveyInProgress) {
      // Записуємо відповідь користувача у Responses (за поточним питанням)
      bool isStored = await StoreUserAnswer(chatId, userMessage);
      if (!isStored) {
        // Наприклад, якесь повідомлення користувачу, що відповідь порожня або некоректна
        await _botClient.SendTextMessageAsync(chatId,
          "Виберіть, будь ласка, один із варіантів (1-4) або введіть коректну відповідь.");
        return;
      }
      // Переходимо до наступного питання
      userState.QuestionIndex++;

      // Якщо питання закінчилися, завершуємо опитування
      var questions = _dataService.GetQuestionsByTopicId(userState.CurrentTopicId);
      if (userState.QuestionIndex >= questions.Count) {
        userState.IsSurveyInProgress = false;
        await _botClient.SendTextMessageAsync(chatId, "Опитування завершено, дякуємо за
участь!");
        // Повертаємо користувача до головного меню
        await SendMainMenu(chatId);
      } else {
        // Показуємо наступне питання
        await SendNextQuestion(chatId, userState);
      }
      return;
    }

    // Якщо нічого з вищезгаданого — обробляємо загальні команди
    switch (userMessage.ToLower()) {
      case "/start":
      case "старт":
        await SendMainMenu(chatId);
        break;
```

```
        default:
          // Можна зробити відповідь за замовчуванням або ігнорувати
          await _botClient.SendTextMessageAsync(chatId, "Я Вас не зрозумів. Введіть /start для
початку.");
          break;
      }
    }

    /// <summary>
    /// Обробка callback-даних від inline-кнопок (з головного меню або інших).
    /// </summary>
    private async void BotClient_OnCallbackQuery(object sender, CallbackQueryEventArgs e) {
      long chatId = e.CallbackQuery.Message.Chat.Id;
      string data = e.CallbackQuery.Data;

      // Лог у вікні
      RaportTBox.Invoke((MethodInvoker)delegate {
        RaportTBox.Text += $"[Callback {chatId}]: {data}\r\n";
      });

      // Перевіряємо, чи маємо стан
      if (!userStates.ContainsKey(chatId)) {
        userStates[chatId] = new BotUserState();
      }
      var userState = userStates[chatId];

      switch (data) {
        case "topics":
          // Натиснули "Перегляд тем" — показуємо список тем
          await ShowTopicsInline(chatId);
          break;

        case "info":
          // Інформація про заклад
          await _botClient.SendTextMessageAsync(chatId,
            "🏫 Інформація про наш заклад вищої освіти:\n\n" +
            "Наш заклад є одним із провідних вищих навчальних закладів, що забезпечує якісну
освіту " +
            "та підготовку спеціалістів у різних галузях знань. Ми пропонуємо:\n\n" +
            "✅ Багатий вибір освітніх програм бакалаврського, магістерського та докторського
рівнів.\n" +
            "✅ Професійний склад викладачів із багаторічним досвідом та сучасними
підходами до навчання.\n" +
            "✅ Інноваційні технології у навчальному процесі, зокрема онлайн-курси, цифрові
ресурси та лабораторії.\n" +
            "✅ Затишний кампус із сучасною інфраструктурою, яка включає бібліотеки,
лабораторії, спортзали та гуртожитки.\n" +
            "✅ Активне студентське життя: клуби за інтересами, культурні заходи, спортивні
секції та волонтерські програми.\n\n" +
```

```
            "🎓 Наші випускники досягають високих результатів у своїх професійних сферах, завдяки чому заклад має відмінну репутацію " +
            "серед роботодавців як в Україні, так і за кордоном.\n\n" +
            "🌐 Для отримання більш детальної інформації відвідайте наш офіційний вебсайт або звертайтеся до приймальної комісії.\n\n" +
            "Дякуємо за інтерес до нашого закладу! Ми завжди раді вітати нових студентів!");

          break;

        case "rules":
          // Правила користування
          // Правила користування
          await _botClient.SendTextMessageAsync(chatId,
            "Правила користування:\n" +
            "1) Використовуйте бот з повагою до інших користувачів та дотримуйтесь правил етики.\n" +
            "2) Уважно обирайте відповіді під час проходження опитувань, оскільки вони будуть збережені у базі даних.\n" +
            "3) Уникайте введення некоректної або нецензурної інформації в текстових полях, зокрема під час залишення відгуків.\n" +
            "4) У разі виникнення технічних проблем звертайтесь до адміністратора закладу через офіційні канали зв'язку.\n" +
            "5) Використовуйте функцію 'Повернутися в головне меню' для навігації в разі непорозумінь або необхідності почати з початку.\n" +
            "6) Ваші персональні дані (ім'я, прізвище, повідомлення) зберігаються виключно з метою аналізу та покращення обслуговування.\n" +
            "7) Усі ваші відповіді будуть оброблені конфіденційно відповідно до політики приватності закладу.\n" +
            "8) Не намагайтеся перевантажити бот через часті або некоректні запити.\n" +
            "9) Ознайомтесь з інформацією про заклад та використовуйте бот для отримання актуальних даних.\n" +
            "10) Залишайте чесні та конструктивні відгуки, щоб допомогти покращити роботу закладу.\n\n" +
            "Дякуємо за використання нашого Telegram-бота!");

          break;

        case "feedback":
          // Вмикаємо режим введення відгуку
          userState.IsLeavingFeedback = true;
          await _botClient.SendTextMessageAsync(chatId, "Введіть, будь ласка, свій відгук одним повідомленням:");
          break;

        case "back_main":
          // Повернення у головне меню
          await SendMainMenu(chatId);
          break;

        default:
          // Можливо, це клік по темі (callbackData = "topic_<id>")
          if (data.StartsWith("topic_")) {
```

```
        string[] parts = data.Split('_');
        if (parts.Length == 2 && int.TryParse(parts[1], out int topicId)) {
          // Користувач обрав тему => стартуємо опитування
          await StartSurvey(chatId, topicId);
        }
      }
      break;
  }

  // Закриємо "вікно" inline-кнопок у Telegram
  await _botClient.AnswerCallbackQueryAsync(e.CallbackQuery.Id);
}

/// <summary>
/// Відправляє головне меню у вигляді inline-кнопок:
///  - Перегляд тем
///  - Інформація про заклад
///  - Правила користування
///  - Залишити відгук
///  - Повернутися в головне меню (інколи)
/// </summary>
private async System.Threading.Tasks.Task SendMainMenu(long chatId) {
  var inlineKeyboard = new InlineKeyboardMarkup(new[]
  {
          // Перший ряд
          new[]
          {
             InlineKeyboardButton.WithCallbackData("Перегляд тем", "topics"),
             InlineKeyboardButton.WithCallbackData("Інформація", "info"),
             InlineKeyboardButton.WithCallbackData("Правила", "rules")
          },
          // Другий ряд
          new[]
          {
             InlineKeyboardButton.WithCallbackData("Залишити відгук", "feedback")
          },
      });
  string welcomeText = "Вітаємо у боті EduSurvey!\nОберіть дію:";
  await _botClient.SendTextMessageAsync(
     chatId: chatId,
     text: welcomeText,
     replyMarkup: inlineKeyboard
  );
}

/// <summary>
/// Формуємо список доступних тем (Topics) з БД і показуємо користувачу
/// у вигляді кнопок: TopicTitle -> callbackData = "topic_{TopicId}"
/// </summary>
private async System.Threading.Tasks.Task ShowTopicsInline(long chatId) {
  var allTopics = _dataService.GetAllTopics();
  if (allTopics == null || allTopics.Count == 0) {
```

```
        await _botClient.SendTextMessageAsync(chatId, "Наразі немає доступних тем.");
        return;
      }

      // Створюємо масив кнопок (по одній у рядку, або можна згрупувати)
      var buttons = new List<InlineKeyboardButton[]>();
      foreach (var topic in allTopics) {
        string buttonText = topic.TopicTitle;
        string callbackData = $"topic_{topic.TopicId}";
        buttons.Add(new[] { InlineKeyboardButton.WithCallbackData(buttonText, callbackData) });
      }

      // Додаємо кнопку повернення у головне меню
      buttons.Add(new[] { InlineKeyboardButton.WithCallbackData("Повернутись у меню",
"back_main") });

      var inlineKeyboard = new InlineKeyboardMarkup(buttons);

      await _botClient.SendTextMessageAsync(
          chatId: chatId,
          text: "Оберіть тему для опитування:",
          replyMarkup: inlineKeyboard
      );
    }

    /// <summary>
    /// Запускає опитування для вказаної теми:
    ///  1) Створює запис у UserSurvey
    ///  2) Перемикає користувача у стан IsSurveyInProgress = true
    ///  3) Показує перше питання
    /// </summary>
    private async System.Threading.Tasks.Task StartSurvey(long chatId, int topicId) {
      // Для користувача створюємо запис у таблиці UserSurvey
      // Припустимо, що FirstName / LastName можна взяти з
e.CallbackQuery.Message.Chat.FirstName тощо
      // Або зберегти як null, якщо немає
      int newSurveyId = _dataService.InsertUserSurvey(new UserSurvey {
        TopicId = topicId,
        ChatId = chatId,
        FirstName = "",  // або заповнити
        LastName = "",   // або заповнити
        SurveyDate = DateTime.Now
      });

      // Зберігаємо в userState
      var userState = userStates[chatId];
      userState.IsSurveyInProgress = true;
      userState.CurrentTopicId = topicId;
      userState.CurrentSurveyId = newSurveyId;
      userState.QuestionIndex = 0;
```

```
    await _botClient.SendTextMessageAsync(chatId, "Опитування розпочато! Ось Ваше перше запитання:");

    // Надсилаємо перше питання
    await SendNextQuestion(chatId, userState);
  }

  /// <summary>
  /// Надсилає користувачу поточне питання (за QuestionIndex) з варіантами відповідей (якщо є).
  /// </summary>
  private async System.Threading.Tasks.Task SendNextQuestion(long chatId, BotUserState userState) {
    var allQuestions = _dataService.GetQuestionsByTopicId(userState.CurrentTopicId);
    if (allQuestions == null || allQuestions.Count == 0) {
      await _botClient.SendTextMessageAsync(chatId, "У цій темі немає запитань.");
      // Вихід із опитування
      userState.IsSurveyInProgress = false;
      return;
    }

    if (userState.QuestionIndex >= allQuestions.Count) {
      // Перевірка на випадок, якщо вже закінчилися
      userState.IsSurveyInProgress = false;
      await _botClient.SendTextMessageAsync(chatId, "Опитування завершено. Дякуємо!");
      await SendMainMenu(chatId);
      return;
    }

    // Поточне питання
    var question = allQuestions[userState.QuestionIndex];

    // Формуємо текст запитання
    string textQuestion = $"Питання #{userState.QuestionIndex + 1} із {allQuestions.Count}:\n";
    textQuestion += question.QuestionsText + "\n\n";

    // Якщо є варіанти
    // (Наприклад, якщо Option1, Option2, Option3, Option4 не пусті, надсилаємо їх у тексті)
    // Так само можна зробити інлайн-кнопки з варіантами
    if (!string.IsNullOrEmpty(question.Option1))
      textQuestion += $"1) {question.Option1}\n";
    if (!string.IsNullOrEmpty(question.Option2))
      textQuestion += $"2) {question.Option2}\n";
    if (!string.IsNullOrEmpty(question.Option3))
      textQuestion += $"3) {question.Option3}\n";
    if (!string.IsNullOrEmpty(question.Option4))
      textQuestion += $"4) {question.Option4}\n";

    textQuestion += "\nВведіть відповідь повідомленням (наприклад, цифру 1-4):";

    await _botClient.SendTextMessageAsync(chatId, textQuestion);
  }
```

```
    /// <summary>
    /// Зберігає відповідь користувача для поточного питання (userState.QuestionIndex) у таблицю Responses.
    /// Повертає true, якщо збережено, false — якщо відповіді не було.
    /// </summary>
    private async System.Threading.Tasks.Task<bool> StoreUserAnswer(long chatId, string userAnswer) {
      var userState = userStates[chatId];
      var allQuestions = _dataService.GetQuestionsByTopicId(userState.CurrentTopicId);

      // Якщо вийшли за межі
      if (userState.QuestionIndex >= allQuestions.Count)
        return false;

      var question = allQuestions[userState.QuestionIndex];

      // Перевірка, чи не пустий рядок
      if (string.IsNullOrWhiteSpace(userAnswer))
        return false;

      // Припустимо, що нам не принципово зберігати тільки 1-4,
      // ми збережемо що завгодно

      var response = new Responses {
        SurveyId = userState.CurrentSurveyId,
        QuestionId = question.QuestionId,
        Answer = userAnswer,       // зберігаємо як є
        ResponseDate = DateTime.Now
      };

      _dataService.InsertResponses(response);
      return true;
    }

    /// <summary>
    /// Зберігає відгук у таблицю LeavingFeedback.
    /// </summary>
    private async System.Threading.Tasks.Task SaveFeedback(long chatId, Telegram.Bot.Types.Message msg) {
      string feedbackText = msg.Text;
      // Зберігаємо відгук
      var feedback = new LeavingFeedback {
        ChatId = chatId,
        FirstName = msg.Chat.FirstName ?? "", // Якщо потрібно
        LastName = msg.Chat.LastName ?? "",
        FeedbackDate = DateTime.Now,
        UserMessage = feedbackText
      };
      _dataService.InsertLeavingFeedback(feedback);
      await _botClient.SendTextMessageAsync(chatId, "Дякуємо! Ваш відгук збережено.");
    }
```

```
    /// <summary>
    /// Приклад надсилання файлу (якщо потрібно).
    /// </summary>
    private async System.Threading.Tasks.Task SendFile(long chatId, string filePath) {
      using (var fileStream = new FileStream(filePath, FileMode.Open, FileAccess.Read,
FileShare.Read)) {
        var fileName = Path.GetFileName(filePath);
        await _botClient.SendDocumentAsync(chatId, new InputOnlineFile(fileStream, fileName));
      }
    }
  }
}


public class BotUserState {
  public bool IsSurveyInProgress { get; set; }   // Чи йде зараз опитування
  public int CurrentTopicId { get; set; }        // Поточна обрана тема
  public int CurrentSurveyId { get; set; }       // Ідентифікатор запису в UserSurvey
  public int QuestionIndex { get; set; }         // Індекс поточного питання
  public bool IsLeavingFeedback { get; set; }    // Чи перебуває користувач у режимі введення
відгуку
}
```

Лістинг 3. Код класу «PollForm.cs»

```
using System;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace SurveyOfAcquirersApp.Forms.Statistics {
  public partial class PollForm : Form {
    private string _ConnString =
System.Configuration.ConfigurationSettings.AppSettings["CONNECT"];
    public PollForm() {
      InitializeComponent();
      GenerateStatistics();
    }
    public void GenerateStatistics() {
      string report = "";
      string queryTotalSurveys = "SELECT COUNT(*) AS TotalSurveys FROM UserSurvey;";
      string queryTotalResponses = "SELECT COUNT(*) AS TotalResponses FROM Responses;";
      string querySurveysByTopic = @"
        SELECT T.TopicTitle, COUNT(US.SurveyId) AS SurveyCount
        FROM UserSurvey US
        JOIN Topics T ON US.TopicId = T.TopicId
        GROUP BY T.TopicTitle;";
      string queryPopularAnswers = @"
```

```
      SELECT Q.QuestionsText, R.Answer, COUNT(R.Answer) AS AnswerCount
      FROM Responses R
      JOIN Questions Q ON R.QuestionId = Q.QuestionId
      GROUP BY Q.QuestionsText, R.Answer
      ORDER BY Q.QuestionsText, AnswerCount DESC;";

     using (SqlConnection conn = new SqlConnection(_ConnString)) {
      conn.Open();

      // Загальна кількість опитувань
      using (SqlCommand cmd = new SqlCommand(queryTotalSurveys, conn)) {
       int totalSurveys = (int)cmd.ExecuteScalar();
       report += $"📊 Загальна кількість опитувань: {totalSurveys}\r\n";
      }
      // Загальна кількість відповідей
      using (SqlCommand cmd = new SqlCommand(queryTotalResponses, conn)) {
       int totalResponses = (int)cmd.ExecuteScalar();
       report += $"📊 Загальна кількість відповідей: {totalResponses}\r\n";
      }
      // Кількість опитувань за темами
      report += "\r\n🗂 Кількість опитувань за темами:\r\n";
      using (SqlCommand cmd = new SqlCommand(querySurveysByTopic, conn)) {
       using (SqlDataReader reader = cmd.ExecuteReader()) {
        while (reader.Read()) {
         string topicTitle = reader["TopicTitle"].ToString();
         int surveyCount = (int)reader["SurveyCount"];
         report += $"- {topicTitle}: {surveyCount} опитувань\r\n";
        }
       }
      }

      // Популярні відповіді за запитаннями
      report += "\r\n📌 Найпопулярніші відповіді:\r\n";
      using (SqlCommand cmd = new SqlCommand(queryPopularAnswers, conn)) {
       using (SqlDataReader reader = cmd.ExecuteReader()) {
        while (reader.Read()) {
         string questionText = reader["QuestionsText"].ToString();
         string answer = reader["Answer"].ToString();
         int answerCount = (int)reader["AnswerCount"];
         report += $"- Запитання: \"{questionText}\" - Відповідь: \"{answer}\" ({answerCount}
разів)\r\n";
        }
       }
      }
      conn.Close();
     }
     // Вивід статистики у поле RaportTBox
     RaportTBox.Text = report;
    }
   }
}
```