Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

Бакалаврська кваліфікаційна робота на тему:

«Автоматизація процесу запису клієнтів на станції технічного обслуговування з використанням технологій чат-ботів»

Виконав: студент IV курсу,
групи 1АКІТ-21б
спеціальності 151 – Автоматизація та
комп'ютерно-інтегровані технології
(шифр і назва спеціальності)
Жовнір Володимир Олександрович
(прізвище та ініціали)

Керівник: к.т.н., доцент кафедри АІІТ
Софина Ольга Юріївна
(прізвище та ініціали)
«17» 06 2025 р.

Рецензент: д.т.н., професор каф. КСУ
Юхимчук Марія Сергіївна
(прізвище та ініціали)
«18» 06 2025 р.

Допущено до захисту
завідувач кафедри АІІТ
«19» 06 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій
Рівень вищої освіти перший (бакалаврський)
Галузь знань 15 – Автоматизація та приладобудування
Спеціальність 151 – Автоматизація та комп'ютерно-інтегровані техгології
Освітньо-професійна програма Інтелектуальні комп'ютерні системи управління

ЗАТВЕРДЖУЮ
завідувач кафедри АІІТ
д.т.н., проф. Бісікало О. В.
«19» 06 2025 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Жовніру Володимиру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Автоматизація процесу запису клієнтів на станції технічного обслучовування з використанням технологій чат-ботів»
керівник роботи Софина Ольга Юріївна, к.т.н, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «20» березня 2025 року №97
2. Термін подання студентом роботи 12.06.2025
3.Вихідні дані до роботи: Автоматизована система запису на станції технічного обслуговування, аналіз і порівняння існуючих рішень, використання Telegram Bot API, Python 3.10+, фреймворк aiogram, SQLite.

4. Зміст текстової частини (перелік питань, які потрібно розробити) провести детальний аналіз предметної області, вивчити існуючі підходи та технології для розробки чат-ботів, провести їх порівняльний аналіз та обгрунтувати вибір оптимального програмного стеку, розробити архітектуру чат-бота, включаючи схему взаємодії з базою даних та логіку обробки запитів,реалізувати функціонал чат-бота з використанням обраних технологій (Python та aiogram v3), провести тестування розробленої системи.

5. Перелік ілюстративного матеріалу Схеми алгоритму модуля роботи системи автоматизованого запису клієнта, алгоритму модуля тестування системи автоматизованого запису клієнтів, UML діаграма, що показує взаємодію клієнта та адміністратора з системо

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Софина О.Ю., доц. каф. АІІТ		

7. Дата видачі завдання 20.03.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Термін виконання		Примітка
		початок	закінчення	
1	Вибір, узгодження та затвердження теми БКР	01.03.2025	20.03.2025	виконан
2	Аналіз предметної області та конкурентних рішень	21.03.2025	23.03.2025	виконан
3	Постановка задачі для вирішення в БКР	25.03.2025	27.03.2025	виконан
4	Проектування архітектури системи, вибір технологій розробки	01.04.2025	15.04.2025	виконан
5	Розробка та тестування системи	16.04.2025	30.04.2025	виконан
6	Оформлення БКР	02.05.2025	16.05.2025	виконан
7	Доопрацювання БКР	01.06.2025	12.06.2025	виконан
8	Захист БКР	19.06.2025	20.06.2025	

Студент _____ Володимир ЖОВНІР
(підпис) (прізвище та ініціали)

Керівник роботи _____ Ольга СОФИНА
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 86 сторінок формату А4, включаючи 22 рисунків та 4 таблиць. Список використаних джерел містить 20 найменувань.

Дана бакалаврська робота присвячена розробці автоматизованої системи запису на станції технічного обслуговування з використанням технологій чат-ботів. Основна увага приділена реалізації рішення на платформі Telegram з використанням відповідних програмних засобів для забезпечення зручного та ефективного взаємодії з клієнтами.У роботі проведено детальний аналіз предметної області, виявлено існуючі проблеми традиційних методів запису на СТО та визначено актуальність впровадження інноваційних рішень. Досліджено сучасні ІТ-рішення для автоматизації бізнес-процесів та виконано огляд існуючих систем автоматизованого запису. На основі проведеного аналізу було розроблено архітектуру чат-бота**,** що включає модулі для діалогової взаємодії, організації збереження та обробки даних клієнтів у базі даних, а т акож алгоритми валідації даних, бронювання та підтвердження запису.

Ключові слова: автоматизація, станція технічного обслуговування, СТО, чат-бот, Telegram, запис клієнтів, інтелектуальні технології, автоматизована система.

ABSTRACT

This Bachelor's Qualification Thesis comprises 86 pages in A4 format, including 22 figures and 4 tables. The list of references contains 20 entries.

This bachelor's thesis is dedicated to the development of an automated appointment booking system for car service stations (CSS) using chatbot technologies. The main focus is on implementing the solution on the Telegram platform, leveraging relevant software tools to ensure convenient and efficient interaction with clients.

The work involved a detailed analysis of the subject area**,** identifying existing problems with traditional CSS appointment booking methods and highlighting the relevance of implementing innovative solutions. Modern IT solutions for automating business processes were explored, and an overview of existing automated booking systems was conducted. Based on this analysis, a chatbot architecture was developed that includes modules for dialogue interaction, managing client data storage and processing in a database, and algorithms for data validation, booking, and appointment confirmation. The developed chatbot ensures 24/7 service availability and minimizes staff involvement.

Keywords: automation, car service station, CSS, chatbot, Telegram, client appointments, intelligent technologies, automated system.

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі ефективна автоматизація взаємодії з клієнтами є критично важливою для успіху у сфері послуг. Для станцій технічного обслуговування (СТО) оперативне управління записами клієнтів безпосередньо впливає на якість обслуговування, завантаженість персоналу та прибутковість.

Традиційні методи запису, такі як телефонні дзвінки або особисте відвідування, супроводжуються значними недоліками: обмежений час роботи, високе навантаження на персонал, ризик людських помилок та неможливість цілодобового обслуговування. Це призводить до втрати потенційних замовлень та зниження ефективності. Відповіддю на ці виклики є впровадження інноваційних цифрових рішень, зокрема чат-ботів, здатних автоматизувати рутинні операції та забезпечити безперебійну взаємодію. Вибір Telegram як основної платформи ґрунтується на його значній популярності (понад 900 мільйонів активних користувачів щомісяця) та широкому функціоналі для розробки ботів, що дозволяє легку інтеграцію з зовнішніми сервісами.

Актуальність теми даної роботи полягає в нагальній потребі оптимізації бізнес-процесів у сфері послуг. Автоматизація процесу запису за допомогою чат-бота значно підвищує ефективність роботи СТО та покращує клієнтський досвід, надаючи цілодобовий доступ до запису, швидкі відповіді та персоналізовані нагадування. Таке рішення сприяє зменшенню операційних витрат, підвищенню пропускної спроможності та збільшенню прибутковості підприємства.

Метою дослідження є автоматизація процесу запису клієнтів на станції технічного обслуговування з використанням технології чат-ботів, що забезпечить зручний, цілодобовий доступ до послуг та підвищить ефективність управління розкладом.

Завдання дослідження:

- Провести детальний аналіз предметної області для виявлення існуючих проблем та чіткого формулювання вимог до майбутньої автоматизованої системи запису.
- Вивчити існуючі підходи та технології для розробки чат-ботів, провести їх порівняльний аналіз та обґрунтувати вибір оптимального програмного стеку.
- Розробити архітектуру чат-бота, включаючи схему взаємодії з базою даних та логіку обробки запитів.
- Реалізувати функціонал чат-бота з використанням обраних технологій (Python та aiogram v3).
- Провести тестування розробленої системи для виявлення та усунення помилок, а також перевірки її відповідності поставленим вимогам.
- Здійснити впровадження чат-бота у хмарне середовище для забезпечення його цілодобової доступності та безперебійної роботи.

Об’єктом дослідження є процес автоматизації взаємодії з клієнтами на станції технічного обслуговування з використанням чат-ботів.

Предметом дослідження є технології розробки чат-ботів на платформі Telegram, які застосовуються для автоматизації процесу запису на СТО.

Практична цінність роботи полягає в тому, що розроблений чат-бот дозволяє скоротити навантаження на персонал, мінімізувати помилки при записі, забезпечити цілодобове обслуговування клієнтів та збільшити загальну ефективність роботи СТО. Створене рішення може бути адаптоване до інших сфер, де потрібен запис на послуги, сприяючи зменшенню операційних витрат, підвищенню пропускної спроможності сервісу та, як наслідок, збільшенню прибутковості підприємства. Накопичення даних про записи також дозволить проводити подальший аналіз та оптимізацію бізнес-процесів СТО.

1 ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗАЦІЇ ОБСЛУГОВУВАННЯ КЛІЄНТІВ СТО

1.1 Особливості діяльності станцій технічного обслуговування

Діяльність станцій технічного обслуговування є ключовою ланкою в життєвому циклі автомобіля, забезпечуючи його надійність, безпеку та довговічність через комплекс діагностичних, ремонтних та профілактичних робіт. Ця сфера має низку унікальних характеристик, які безпосередньо впливають на всі аспекти управління, включаючи взаємодію з клієнтами та необхідність автоматизації.

СТО функціонують в умовах високої технологічності та постійного розвитку. Сучасні автомобілі оснащені складною електронікою та інноваційними системами, що вимагає від СТО наявності спеціалізованого обладнання, такого як діагностичні сканери, стенди розвал-сходження, тестери гальмівних систем, та висококваліфікованого персоналу. Майстри повинні постійно підвищувати свою кваліфікацію, а обладнання потребує регулярного оновлення та калібрування. Це обумовлює значні капітальні вкладення, що впливає на ціноутворення та конкурентоспроможність.

Суттєвим аспектом є різновид та складність послуг. Спектр послуг СТО надзвичайно широкий: від стандартних, як-от заміна мастила, фільтрів чи шиномонтаж, до складних, таких як капітальний ремонт двигуна, відновлення кузова після ДТП, ремонт автоматичних коробок передач чи діагностика та ремонт електричних систем. Кожна послуга має свою тривалість, потребує специфічних інструментів та витратних матеріалів, а також певного рівня кваліфікації майстра. Це створює складну задачу для планування ресурсів та формування розкладу, оскільки необхідно враховувати не лише час, а й доступність відповідного обладнання та спеціалістів. Також можна зазначити, що діяльність СТО характеризується високою чутливістю до сезону та зовнішніх

факторів. Наприклад піком попиту на шиномонтаж припадає осінній та весняний періоди, а після ожеледиці зростає кількість звернень щодо кузовного ремонту або проблем з підвіскою. Непередбачувані поломки автомобілів також створюють нерівномірне навантаження. Це призводить до періодів пікового завантаження, коли майстри працюють на межі можливостей, і періодів простою, що негативно впливає на прибутковість. Ефективне управління записом може допомогти оптимізувати завантаження.

Критично важливим є орієнтованість на клієнта та якість сервісу. В умовах високої конкуренції саме якість обслуговування, швидкість реагування, точність діагностики та рівень комунікації визначають лояльність клієнтів. Зручний процес запису, прозорість цін, інформування про статус ремонту та можливість швидкого вирішення питань значно підвищують задоволеність клієнтів та сприяють формуванню позитивної репутації.

Отже управління матеріальними та людськими ресурсами на СТО є складним завданням. Необхідно контролювати наявність тисяч найменувань запасних частин та витратних матеріалів, враховувати їхню специфіку, наприклад, оригінальні запчастини чи аналоги, їх сумісність з конкретними марками авто. Крім того, ефективне планування графіків роботи майстрів, розподіл завдань між ними з урахуванням їхніх компетенцій та завантаження обладнання є запорукою успіху.

Усі ці аспекти підкреслюють, що процес запису клієнтів на СТО не є просто адміністративною функцією. Це стратегічний елемент управління, який безпосередньо впливає на операційну ефективність, фінансові показники та клієнтську лояльність. Неоптимізований запис призводить до втрачених можливостей, простоїв, незадоволених клієнтів та, в кінцевому результаті, до зниження прибутковості.

1.2 Аналіз процесу запису клієнтів: проблеми та типові рішення

Традиційний процес запису клієнтів на станціях технічного обслуговування часто є недосконалий, що створює неефективність та незадоволеність як для персоналу, так і для клієнтів. Детальне розуміння цих проблем є основою для розробки ефективних автоматизованих рішень.

Типові проблеми традиційного процесу запису включають високе операційне навантаження на персонал. Значна частина записів здійснюється по телефону, що вимагає від адміністраторів або майстрів-приймальників постійного відволікання на дзвінки. Вони витрачають значний час на уточнення марки та моделі авто, виду послуги, побажань клієнта щодо дати та часу, а також перевірку доступності майстра та обладнання. Це не лише знижує продуктивність, а й може призвести до помилок при одночасному виконанні кількох завдань. Навіть якщо клієнт залишає запит через форму на сайті, його необхідно вручну обробити, перевірити доступність і зв'язатися з клієнтом для підтвердження. Дзвінки для нагадування про візит або обробка запитів на скасування чи перенесення також займають час персоналу.

Крім того, існує людський фактор та високий ризик помилок. Запис інформації у паперові журнали, електронні таблиці, наприклад, Excel, або навіть у неавтоматизовані внутрішні системи схильний до людських помилок, таких як неправильно записаний номер телефону, ім'я, марка авто, некоректно вибраний час або послуга. При одночасному обслуговуванні кількох каналів запису або недостатній координації між співробітниками можуть виникати накладки, коли кілька клієнтів записані на один і той самий час або до одного й того ж майстра. Це призводить до черг, затримок та незадоволеності клієнтів.

Також значною проблемою є обмежена доступність та незручність для клієнта. Клієнти можуть записатися лише в робочий час СТО, що незручно для тих, хто зайнятий у робочий час і не може здійснити дзвінок. Необхідність дзвонити, очікувати на лінії, а потім чекати підтвердження може дратувати клієнтів. Клієнти не завжди мають миттєвий доступ до актуального розкладу,

вільних слотів, повного переліку послуг та їх орієнтовної вартості без прямого контакту з СТО.

Для подолання цих проблем застосовуються типові рішення. Найпростішим і найдешевшим способом, що все ще використовується в невеликих СТО, є паперові журнали та блокноти, хоча вони мають високий ризик помилок та низьку інформативність. Електронні таблиці, такі як Excel або Google Sheets, є покращеною версією, що дозволяє зберігати дані в цифровому вигляді, але все ще страждають від відсутності автоматизації та проблем з одночасним доступом.

Онлайн-форми на веб-сайтах дозволяють клієнту заповнити форму на сайті СТО, що забезпечує цілодобову доступність та збір структурованих даних, але часто вимагає ручної обробки та не завжди інтегрується з внутрішнім розкладом у реальному часі. Спеціалізовані програмні комплекси та CRM-системи для СТО є комплексними рішеннями, що охоплюють облік клієнтів, склад, фінанси та модуль запису. Вони централізують дані, автоматизують багато процесів та надають розширену аналітику. Однак їх впровадження є дорогим і складним, і вони часто не забезпечують зручного самостійного запису для клієнта через звичні для нього канали, такі як месенджери.

Крім того, варто враховувати, що багато клієнтів очікують простого інтуїтивного досвіду спілкування з бізнесом через канали, до яких вони звикли – месенджери, соціальні мережі, мобільні додатки. У цьому контексті навіть найсучасніші CRM-системи можуть виглядати технічно досконалими, але недостатньо зручними з точки зору користувача.

Незважаючи на наявність усіх вищезгаданих рішень, багато з них не усувають ключової проблеми – відсутності повністю автоматизованого, інтерактивного та цілодобового каналу запису, який би мінімізував участь персоналу та був максимально зручним для клієнта. Враховуючи стрімкий розвиток технологій штучного інтелекту та популярність месенджерів, найперспективнішим напрямком є впровадження чат-ботів – інтелектуальних агентів, здатних самостійно вести діалог, уточнювати дані, бронювати послуги

та інтегруватися з внутрішніми системами розкладу. Саме ця технологія дає можливість створити ефективну, економну та клієнтоорієнтовану систему запису, яка враховує потреби як бізнесу, так і кінцевого користувача[1; 18].

1.3 Сучасні IT-рішення для автоматизації бізнес-процесів

Ефективність сучасного бізнесу значною мірою залежить від ступеня автоматизації його ключових процесів. Впровадження інформаційних технологій дозволяє оптимізувати операції, знизити витрати, підвищити продуктивність та найважливіше - покращити взаємодію з клієнтами. Розглянемо основні категорії сучасних IT-рішень, що застосовуються для автоматизації бізнес-процесів:

1. Системи планування ресурсів підприємства (ERP). Це комплексні, інтегровані програмні пакети, які автоматизують основні бізнес-функції організації, такі як фінанси, бухгалтерський облік, управління ланцюгами поставок, виробництво, управління персоналом, управління проектами та взаємодія з клієнтами. Вони об'єднують дані з усіх департаментів у єдину базу, забезпечуючи централізований контроль та прозорість бізнес-процесів, підвищуючи ефективність, покращуючи координацію між відділами та оптимізуючи витрати. Прикладами є SAP ERP, Oracle ERP Cloud, Microsoft Dynamics 365, проте вони відзначаються високою вартістю впровадження та підтримки (рис. 1.1).

Рисунок 1.1 – Найпопулярніші ERP системи.

2. Системи управління взаємовідносинами з клієнтами (CRM). Ці рішення допомагають компаніям управляти взаємовідносинами з поточними та потенційними клієнтами, охоплюючи продажі, маркетинг та обслуговування. CRM-системи зберігають усю інформацію про клієнтів, автоматизують маркетингові кампанії, відстежують статус угод та управляють зверненнями, що призводить до покращення якості обслуговування та підвищення лояльності клієнтів. Приклади включають Salesforce, HubSpot CRM, Zoho CRM, а для СТО існують спеціалізовані CRM, такі як RemOnline та АвтоПрофі (рис. 1.2).

Рисунок 1.2 – Найпопулярніші CRM системи.

3. Системи управління бізнес-процесами (BPM). Вони надають інструменти та методології для моделювання, автоматизації, виконання, моніторингу та оптимізації бізнес-процесів. BPM-системи визначають та стандартизують послідовність дій, автоматизують їх, відстежують прогрес та дозволяють виявляти недоліки, що підвищує ефективність, зменшує операційні витрати та покращує якість результатів. Приклади – Appian, Bizagi, Camunda BPM (рис. 1.3).

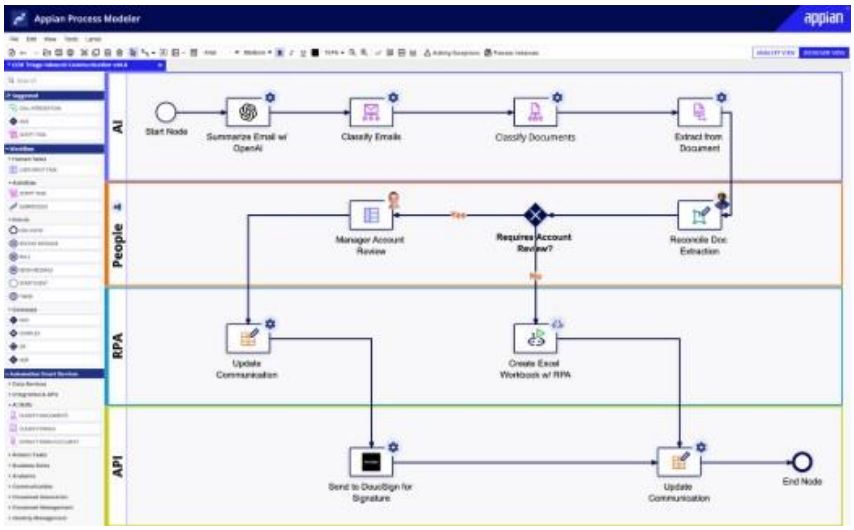

Рисунок 1.3 – Інтерфейс популярної BPM системи Appian.

4. Системи для електронної комерції, призначені для створення та управління онлайн-магазинами. Вони забезпечують функціонал для каталогу товарів, кошика, платіжних систем, управління замовленнями та доставки, розширюючи ринок збуту та автоматизуючи продажі. Серед них – Shopify, WooCommerce, Magento (рис. 1.4).

Рисунок 1.4 – Найпопулярніші системи для електронної комерції.

5. Чат-боти та віртуальні помічники. Це програмні рішення, що імітують розмову з людиною для надання інформації, виконання завдань або підтримки. Вони взаємодіють з користувачами через текстові або голосові інтерфейси, використовуючи заздалегідь визначені сценарії або алгоритми обробки природної мови. Їхні переваги включають цілодобову доступність, швидкість реагування, зниження навантаження на персонал та масштабованість, що покращує клієнтський досвід.

6. Хмарні обчислення. Це модель надання обчислювальних послуг через інтернет, що дозволяє компаніям використовувати ресурси за потребою без значних інвестицій у власну інфраструктуру, забезпечуючи гнучкість, масштабованість та високу доступність. Прикладами є Amazon Web Services, Microsoft Azure, Google Cloud Platform (рис. 1.5).

Рисунок 1.5 – Найпопулярніші системи хмарних обчислень.

7. Аналітичні системи. Вони призначені для збору, обробки, аналізу та візуалізації великих обсягів даних для підтримки прийняття управлінських рішень, перетворюючи "сирі" дані в корисні звіти та інсайти. Приклади – Tableau, Microsoft Power BI, Qlik Sense.

Для автоматизації процесу запису клієнтів на СТО, особливий інтерес становлять чат-боти, оскільки вони є доступним, інтерактивним та ефективним інструментом для взаємодії з клієнтами в повсякденних каналах комунікації. Їх можна інтегрувати з CRM-системами та календарями, що дозволяє поєднати переваги автоматизованої комунікації з централізованим управлінням даними [5].

1.4 Технології чат-ботів: архітектура, платформи, сфери застосування

Чат-боти є однією з найбільш динамічно розвиваючихся технологій у сфері автоматизації комунікацій та обслуговування. Це програмні системи які імітують живе спілкування з користувачем у режимі реального часу через текстові або голосові повідомлення. Їх основне завдання полягає у швидкому наданні інформації відповіді на запити користувача або виконанні певної дії наприклад бронювання запис створення заявки тощо. Сфера застосування чат-ботів стрімко розширюється охоплюючи такі галузі як технічна підтримка електронна комерція освіта медицина банківські послуги громадський транспорт і сфера послуг включаючи станції технічного обслуговування автомобілів.

Чат-бот функціонує за певною архітектурною моделлю яка включає кілька ключових компонентів:

Клієнтська частина яка взаємодіє з користувачем через один із популярних месенджерів або спеціальний веб-інтерфейс.

Серверна частина яка обробляє повідомлення користувача формує відповіді та здійснює логіку роботи.

База даних яка зберігає інформацію про користувачів історію взаємодій записи на послуги розклад роботи компанії тощо (рис. 1.6).

Рисунок 1.6 – Блок-схема архітектурної моделі чат-боту.

Також сучасні чат-боти можуть інтегруватися з зовнішніми системами наприклад CRM системами Google Calendar електронною поштою або платіжними сервісами що значно розширює їх функціональні можливості. У більш просунутих реалізаціях чат-боти можуть містити модулі обробки природної мови що дозволяє їм краще розуміти неконкретні або розмовні запити користувача та відповідати на них більш гнучко

Щодо інструментів і платформ для розробки чат-ботів то сьогодні розробники мають широкий вибір. Однією з найпоширеніших є Telegram Bot API яка пропонує простий у використанні інтерфейс для створення ботів із підтримкою текстових кнопок інлайн-клавіатур повідомлень з геолокацією та інших функцій. Популярністю користується також платформа Dialogflow від Google яка підтримує обробку природної мови й може інтегруватися з великою кількістю месенджерів. Іншими прикладами є Microsoft Bot Framework який підходить для масштабних проєктів і підтримує безліч каналів а також Rasa – фреймворк з відкритим кодом для створення гнучких кастомізованих ботів із власною логікою. Крім того існують візуальні конструктори ManyChat чи Chatfuel які дозволяють створювати чат-ботів без програмування що є зручним для малого бізнесу.

Чат-боти активно використовуються для автоматизації рутинних процесів і зменшення навантаження на персонал. У сфері обслуговування автомобілів зокрема на СТО чат-бот може замінити телефонного оператора виконуючи запис

клієнта на обслуговування відповідно до графіка роботи майстрів нагадувати про візит уточнювати попередні заявки або інформувати про акції та знижки. Така автоматизація забезпечує швидкість обробки запитів зменшує кількість помилок пов'язаних із людським фактором і підвищує рівень задоволеності клієнтів.

Таким чином впровадження чат-ботів у бізнес-процеси особливо в контексті СТО є актуальним і економічно доцільним кроком. Завдяки широкому спектру доступних технологій і платформ компанії можуть обирати саме ті інструменти які найкраще відповідають їхнім технічним можливостям та бізнес-цілям а також забезпечують зручну й ефективну взаємодію з клієнтами [16].

1.5 Огляд існуючих рішень автоматизованого запису клієнтів, переваги використання чат- ботів.

Ринок автоматизованих рішень для запису клієнтів постійно розвивається, пропонуючи різноманітні підходи, що різняться за функціональністю, вартістю та рівнем інтеграції. Кожен з цих підходів має свої унікальні переваги та недоліки, які необхідно враховувати при виборі оптимального рішення для конкретного бізнесу.

Системи онлайн-запису на веб-сайтах це найбільш поширений і базовий рівень автоматизації, що вже став стандартом для багатьох галузей. Клієнт, зайшовши на веб-сайт СТО або інтегрований модуль стороннього сервісу, отримує можливість самостійно обрати потрібну послугу, переглянути доступні дати та часові слоти, а потім заповнити свої контактні дані для підтвердження запису. Такі системи можуть бути як вбудованими модулями в існуючі CRM-системи, що використовуються СТО, наприклад, RemOnline чи АвтоПрофі, так і окремими спеціалізованими онлайн-сервісами для бронювання, які можуть бути адаптовані під специфічні потреби автосервісу. Головна перевага цього підходу полягає у його широкій доступності та простоті у використанні для клієнтів, які активно шукають послуги в інтернеті. Вони не потребують встановлення

додаткового програмного забезпечення, що робить процес запису швидким і зрозумілим. Проте, суттєвим обмеженням є пасивність цього методу: клієнту необхідно свідомо шукати веб-сайт та переходити на нього, а також відсутність проактивного спілкування та можливості швидкого уточнення деталей в реальному часі.

Мобільні застосунки представляють собою більш функціональне та інтегроване рішення. Деякі великі мережеві СТО, дилерські центри або агрегатори автосервісів розробляють власні мобільні додатки, які значно розширюють можливості взаємодії з клієнтами. Через ці додатки клієнти можуть не лише записатися на обслуговування, а й переглядати повну історію своїх візитів, отримувати персоналізовані нагадування про майбутні ТО або акційні пропозиції, а також відстежувати статус ремонту свого автомобіля. Це забезпечує високий рівень функціональності та глибоку персоніфікацію, а також дозволяє використовувати push-повідомлення для миттєвого інформування. Однак, головним недоліком цього підходу є значна складність у залученні користувачів до встановлення мобільних додатків. Більшість клієнтів неохоче встановлюють окремий додаток для кожного сервісу, яким вони користуються, що призводить до низької конверсії на встановлення. Крім того, розробка, постійна підтримка та оновлення мобільного додатку вимагають значних фінансових та часових витрат.

CRM та ERP-системи з функціоналом запису представляють собою потужні інструменти для централізованого управління бізнес-процесами СТО. Окрім управління клієнтами, складськими запасами та фінансовими операціями, ці системи часто включають модулі для планування розкладу роботи майстрів та запису на послуги. Вони переважно призначені для внутрішнього використання персоналом СТО, дозволяючи централізувати всі бізнес-процеси та надавати детальну аналітику щодо ефективності роботи, завантаженості персоналу та фінансових показників. Прикладами таких систем є RemOnline, АвтоДілер, АвтоПрофі. Незважаючи на їхню функціональність та аналітичні можливості, основний недолік полягає у високій вартості впровадження та налаштування.

Крім того, їхній основний фокус зосереджений на внутрішньому обліку та оптимізації процесів для персоналу, а не на зручності зовнішнього клієнта для самостійного запису через звичні для нього канали комунікації.

Чат-боти є сучасним та інноваційним підходом до автоматизації запису, що набуває все більшої популярності завдяки своїй зручності та ефективності. Чат-боти інтегруються безпосередньо у найпопулярніші месенджери, такі як Telegram, Viber, Facebook Messenger, що дозволяє клієнтам записатися на СТО, спілкуючись з ботом у звичному для них середовищі. Цей підхід відрізняється високою інтерактивністю: бот веде діалог з користувачем, уточнюючи необхідні деталі – марку автомобіля, тип послуги, бажану дату та час, після чого перевіряє доступність слотів та підтверджує запис. Існують як сценарійні боти, що працюють за заздалегідь визначеними алгоритмами та питаннями, так і більш складні боти з використанням технологій NLU, які здатні розуміти вільну мову користувача, що робить взаємодію більш природною. Деякі CRM-системи також пропонують власні інтеграції з чат-ботами, що забезпечує синхронізацію даних [9; 17].

Переваги чат-ботів для СТО є численними та значними**:**

- Цілодобова доступність: Клієнти можуть записатися на послуги в будь-який час доби, незалежно від робочого графіка СТО, що значно підвищує зручність та доступність сервісу.
- Висока зручність для клієнта: Взаємодія відбувається у звичному та повсякденному середовищі месенджера. Клієнтам не потрібно завантажувати нові додатки, запам'ятовувати паролі або шукати веб-сайти.
- Зниження навантаження на персонал: Чат-боти автоматично обробляють більшість рутинних запитів, звільняючи час співробітників СТО для більш складних завдань, що вимагають людського втручання.
- Автоматичні нагадування: Боти можуть надсилати автоматичні нагадування про майбутній візит, що суттєво зменшує кількість неявок та підвищує ефективність планування.

- Персоналізація: Чат-боти здатні "запам'ятовувати" клієнта та інформацію про його автомобіль (марку, модель, історію попередніх візитів), що дозволяє пропонувати персоналізовані послуги та покращувати клієнтський досвід.
- Економічна ефективність: Розробка та впровадження чат-бота часто є дешевшою та швидшою, ніж створення повноцінного мобільного додатку, а його підтримка є більш гнучкою та менш витратною.
- Широке охоплення аудиторії: Використання популярних месенджерів дозволяє охопити значну частину потенційних клієнтів, які вже активно користуються цими платформами.

Однак їхні недоліки включають потенційну обмеженість у розумінні дуже складних або нестандартних запитів, які можуть вимагати втручання людини. Також, для забезпечення актуальності даних та ефективної роботи, необхідна надійна інтеграція чат-бота з внутрішніми системами СТО, такими як CRM або система обліку зайнятості майстрів. Ефективність функціонування бота також безпосередньо залежить від якості розробленого діалогового сценарію або точності моделі NLU [19; 16; 11].

1.6 Висновок до розділу 1

Аналізуючи існуючі рішення, можна дійти висновку, що кожне з них має свої сильні та слабкі сторони. Системи онлайн-запису на веб-сайтах прості, але пасивні. Мобільні застосунки пропонують глибоку інтеграцію, але вимагають значних інвестицій та мають низький рівень встановлення. CRM/ERP-системи є потужними для внутрішнього управління, але не завжди орієнтовані на зручність зовнішнього клієнта.

Саме чат-боти пропонують унікальне поєднання зручності для клієнта, цілодобової доступності та значного зниження операційного навантаження на персонал СТО. Вони мінімізують складність для клієнта, дозволяючи здійснити запис у звичному та комфортному середовищі месенджера, що є ключовим фактором у сучасному цифровому світі. Ця комбінація переваг робить чат-боти

особливо привабливими для автоматизації процесу запису та визначає фокус даної дипломної роботи на їх розробці та впровадженні, з метою створення ефективного та клієнтоорієнтованого рішення для сучасних станцій технічного обслуговування.

2 ВИБІР ОПТИМАЛЬНОГО ПІДХОДУ ДЛЯ РЕАЛІЗАЦІЇ ВЛАСНОГО РІШЕННЯ

2.1 Аналіз існуючих підходів до розробки Telegram-ботів

Для реалізації Telegram-бота сьогодні доступний широкий спектр технологій та платформ, кожна з яких має свої особливості. Було проведено ретельний аналіз трьох основних категорій: хмарних NLP-платформ та серверної розробки на JavaScript і Python [4].

Хмарна NLP-платформа Google Dialogflow, являє собою потужну хмарну платформу, спеціально розроблену для створення розмовних інтерфейсів. Вона дозволяє розробляти віртуальних агентів, які здатні ефективно розуміти природну мову користувачів. Основними компонентами Dialogflow є інтенти (наміри користувача, наприклад, "записатися на візит" або "скасувати запис"), сутності (ключові слова або фрази, такі як "дата", "час", "послуга") та контексти, що зберігають стан діалогу. Принцип роботи полягає в тому, що розробник визначає можливі запити користувача та формулює типові фрази, за якими бот може розпізнати їх. Dialogflow автоматично обробляє вхідний текст, визначає найвірогідніший намір та витягує з нього необхідні дані. Для реалізації складнішої бізнес-логіки, такої як перевірка доступності конкретного часу в базі даних або інші унікальні бізнес-процеси, Dialogflow може викликати зовнішні вебхуки, які обробляються вашим власним сервером.

До переваг Dialogflow можна віднести надзвичайно швидку розробку простих діалогів завдяки інтуїтивно зрозумілому візуальному інтерфейсу, що дозволяє створювати ботів без глибоких знань програмування. Він також володіє потужними можливостями NLP, забезпечуючи відмінне розуміння природної мови та підтримку багатьох мов. Крім того, Dialogflow легко інтегрується з різними месенджерами, включаючи Telegram, а також з голосовими помічниками, такими як Google Assistant. Однак, існують і значні недоліки**.** Його функціональність може бути обмеженою для реалізації дуже складної та унікальної бізнес-логіки, оскільки для глибокої інтеграції з власною базою даних

або впровадження специфічних бізнес-процесів доводиться писати значний обсяг власного коду у вебхуках, що ускладнює архітектуру системи. Крім того, Dialogflow є хмарним сервісом, що обумовлює залежність від нього та потенційні витрати: безкоштовний тариф має обмеження, а зі зростанням кількості запитів вартість може суттєво зрости (рис.2.1).

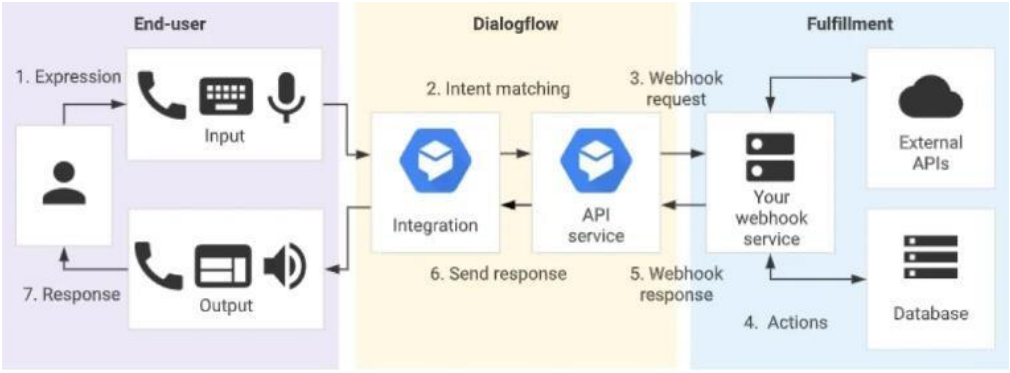

Рисунок 2.1 – Приклад роботи Google Dialogflow

Python, як мова програмування, здобула широку популярність завдяки своєму простому синтаксису, інтуїтивно зрозумілому коду та надзвичайно багатій екосистемі бібліотек, що робить її ідеальною для швидкої та ефективної розробки. Серед фреймворків для розробки Telegram-ботів на Python, aiogram та python-telegram-bot є двома провідними рішеннями, які підтримують асинхронну роботу. Aiogram особливо вирізняється своєю повністю асинхронною природою, вбудованою підтримкою Finite State Machine (FSM) та архітектурними особливостями, які роблять його високоефективним для взаємодії з Telegram API. Принцип роботи полягає в тому, що Python-додаток, використовуючи asyncio, отримує оновлення від Telegram, а aiogram ефективно маршрутизує ці оновлення до відповідних обробників. Завдяки асинхронності, бот може одночасно обробляти кілька запитів, не блокуючи основний потік виконання під час очікування відповіді від Telegram API або бази даних. Вбудована FSM дозволяє легко керувати станами діалогу, забезпечуючи послідовність та логічність взаємодії з користувачем (рис.2.2).

Рисунок 2.2 – Логотип та візуальний вигляд коду

Основними перевагами цієї зв'язки є простота та швидкість розробки. Простий синтаксис Python дозволяє швидко писати, тестувати та вносити зміни до коду. Повна асинхронність aiogram забезпечує високу продуктивність та ефективне використання системних ресурсів, що критично важливо для додатків, які інтенсивно працюють з операціями введення та виведення. Вбудована FSM є потужним інструментом для зручного управління послідовними діалогами, що значно спрощує розробку складних сценаріїв. Величезна кількість готових бібліотек та інструментів в екосистемі Python для різноманітних завдань, таких як робота з базами даних (SQLite, PostgreSQL), обробка дат та часу, валідація даних, суттєво прискорює розробку та спрощує інтеграцію. Крім того, Python та aiogram мають велику та надзвичайно активну спільноту розробників, що забезпечує легкий доступ до документації, прикладів та допомоги [14].

2.2 Обґрунтування вибору середовища розробки

Прийняття рішення щодо вибору програмного стеку для створення Telegram чат-бота, призначеного для автоматизації запису клієнтів на СТО, було ретельно обмірковано та базується на основі глибокого аналізу доступних технологій. Фінальний вибір – поєднання мови програмування Python 3.10+ та потужного фреймворку python-telegram-bot – є оптимальним рішенням, що відповідає найважливішим вимогам проєкту щодо продуктивності, гнучкості, масштабованості та простоти підтримки.

Однією з ключових переваг Python є його інтуїтивно зрозумілий синтаксис та висока читабельність коду, що значно прискорює процес розробки. Це дозволяє команді швидко створювати функціональні прототипи, ефективно тестувати гіпотези та оперативно вносити зміни, що є критично важливим для ітераційного підходу до розробки програмного забезпечення. Велика кількість готових бібліотек та модулів, що доступні в розгалуженій екосистемі Python (таких як для роботи з базами даних, обробки дат, логування), додатково скорочує час, необхідний для реалізації типових функцій, дозволяючи розробникам зосередитися безпосередньо на унікальній бізнес-логіці бота. Більше того, універсальність Python дозволяє використовувати його для широкого спектру завдань, що робить його гнучким вибором для майбутнього розширення функціоналу або інтеграції з іншими системами.

Фреймворк python-telegram-bot, своєю чергою, розроблений спеціально для зручної та ефективної роботи з Telegram Bot API. Він надає зручні та ефективні інструменти для обробки вхідних повідомлень, команд, натискань кнопок та інших типів оновлень, а також для керування станом діалогу. Його архітектура, що базується на Updater та Dispatcher, забезпечує високу швидкість обробки запитів та дозволяє боту реагувати на безліч одночасних звернень без блокування, що є життєво важливим для безперебійної роботи чат-бота у системі, яка потенційно обслуговуватиме велику кількість користувачів. Однією з найсильніших сторін python-telegram-bot є ConversationHandler, який суттєво

спрощує реалізацію складних, багатоетапних сценаріїв діалогу, таких як процес запису, шляхом чіткого визначення станів та обробників для кожного кроку. Фреймворк також надає потужні можливості для створення інтерактивних елементів інтерфейсу, таких як InlineKeyboardMarkup та ReplyKeyboardMarkup, що робить взаємодію з ботом максимально інтуїтивною та зручною для кінцевого користувача.

Вибір Python та python-telegram-bot забезпечує виняткову гнучкість та широкі можливості кастомізації. Це дозволяє адаптувати чат-бота під найрізноманітніші та унікальні бізнес-вимоги сервісу СТО. Наприклад, можна легко інтегрувати складну логіку для вибору послуг з динамічним ціноутворенням, враховувати завантаженість окремих майстрів або спеціалізованого обладнання, реалізувати гнучкий графік запису, що враховує вихідні та святкові дні, а також налаштувати безшовну синхронізацію з існуючими CRM-системами або календарями. Python дозволяє реалізовувати нестандартні алгоритми та ефективну обробку даних, а python-telegram-bot надає всі необхідні інструменти для створення інтерактивних меню, клавіатур, валідації введених даних та обробки різноманітних типів повідомлень, що робить взаємодію з ботом максимально зручною та персоналізованою для кінцевого користувача. Ця гнучкість також дозволяє легко впроваджувати нові функції та масштабувати можливості бота в майбутньому без архітектурних обмежень.

Python має одну з найбільших та найактивніших спільнот розробників у світі. Це означає доступ до величезної кількості документації, навчальних матеріалів, прикладів коду, готових бібліотек та рішень для практично будь-якого завдання. У разі виникнення проблем або потреби в консультації, відповідь можна швидко знайти на спеціалізованих форумах, у тематичних групах, на GitHub або в офіційній документації. Спільнота навколо python-telegram-bot також є досить розвиненою, що забезпечує регулярні оновлення фреймворку, своєчасні виправлення помилок, які враховують зміни в Telegram API, та постійну підтримку. Наявність такої широкої та активної екосистеми значно

спрощує процес розробки та супроводу проєкту, мінімізуючи потенційні труднощі та ризики, а також полегшує пошук кваліфікованих кадрів для підтримки та розвитку системи.

Таким чином, вибір Python 3.10+ та python-telegram-bot як середовище розробки є цілком обґрунтованим і стратегічно правильним рішенням. Ця комбінація забезпечить створення функціонального, надійного, продуктивного та масштабованого Telegram чат-бота, який ефективно автоматизуватиме процес запису клієнтів на СТО, відповідаючи всім сучасним вимогам до подібних систем та закладаючи міцний фундамент для майбутнього розвитку [6; 12].

2.3 Інтеграція чат-бота з базою даних та зовнішніми сервісами

Ефективна робота Telegram чат-бота для запису клієнтів на СТО неможлива без повноцінної інтеграції з надійною базою даних для постійного зберігання інформації, а також, у перспективі, з зовнішніми сервісами, які суттєво розширюють функціональність системи та забезпечують значно кращий користувацький досвід та операційну ефективність.

Для реалізації локального зберігання даних у даному проєкті було обрано SQLite – легку, файлову реляційну систему управління базами даних. Цей вибір є ідеальним для невеликих за обсягом автономних додатків, де не потрібен потужний сервер баз даних і де пріоритетом є простота розгортання та адміністрування. Основними перевагами SQLite є її простота у використанні, мінімальні вимоги до ресурсів системи, відсутність потреби в окремому сервері (база даних зберігається як єдиний файл на тому ж пристрої, що й додаток), та зручна інтеграція з Python завдяки стандартній вбудованій бібліотеці sqlite3. Це дозволяє швидко і ефективно виконувати SQL-запити без додаткових залежностей. У базі даних зберігаються всі критично важливі дані, необхідні для процесу запису: інформація про клієнта (ім’я, номер телефону, що є унікальним ідентифікатором), обрана послуга (наприклад, "Заміна масла", "Діагностика"),

бажана дата та час запису, а також додаткові деталі, такі як номерний знак автомобіля. У майбутньому може бути додано статус запису (активний, скасований, завершений) для більш детальної аналітики.

З точки зору архітектури, база даних функціонує як окремий, незалежний компонент, до якого чат-бот надсилає запити (за допомогою SQL-команд) при створенні нового запису, оновленні існуючої інформації або отриманні даних про поточні та минулі записи. Усі запити до бази даних є параметризованими, що означає, що значення передаються окремо від SQL-команди. Це є важливою мірою безпеки, що мінімізує ризик SQL-ін'єкцій та підвищує загальний рівень безпеки системи, запобігаючи виконанню довільного шкідливого коду (рис. 2.4).

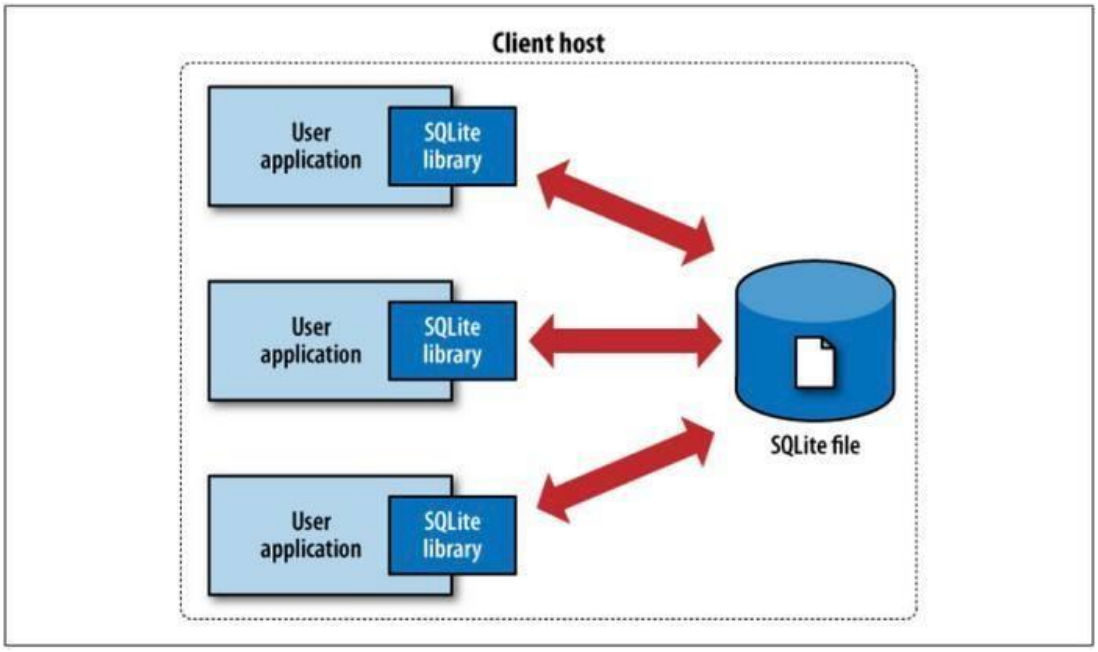

Рисунок 2.4 – Принцип роботи SQLite

У межах подальшого розвитку функціоналу чат-бота, передбачено інтеграцію з різноманітними зовнішніми сервісами, що значно розширить його можливості:

- Google Calendar – для автоматичної синхронізації записів клієнтів із календарем адміністратора або окремих майстрів СТО. Це дозволить уникнути конфліктів у графіку, автоматично блокувати зайнятий час та надавати клієнтам

лише доступні слоти. Інтеграція забезпечить централізований перегляд розкладу для всього персоналу.

- CRM-системи (наприклад, SalesDrsve, Asteril CRM або інші популярні галузеві рішення) – для автоматичного збереження контактів нових клієнтів, детальної історії їхніх візитів, переліку наданих послуг, історії платежів та подальшого ефективного маркетингу. Це дозволить сегментувати клієнтську базу, запускати цільові рекламні кампанії, надсилати персоналізовані пропозиції та будувати довгострокові відносини (рис. 2.5), (рис. 2.6).

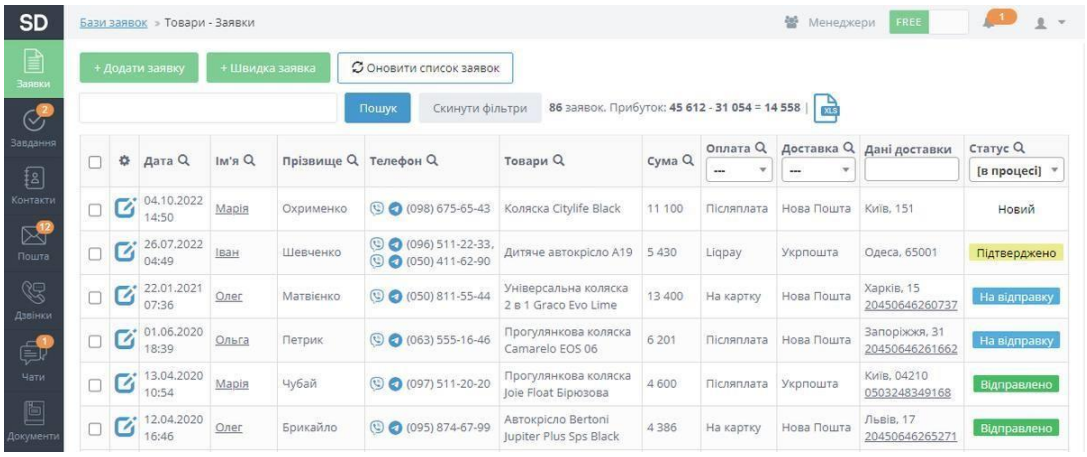

Рисунок 2.5 – Інтерфейс SalesDrsve

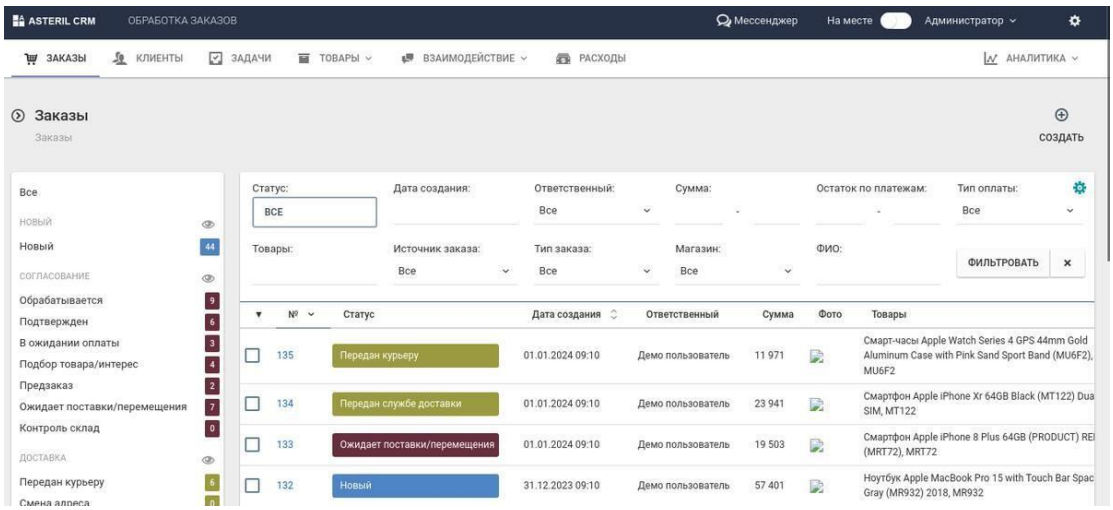

Рисунок 2.6 – Інтерфейс Asteril CRM

- Сервіси електронної пошти або SMS (наприклад, SendGrid, Twilio) – для надсилання додаткових підтверджень запису та нагадувань про майбутній візит

через альтернативні канали комунікації. Це підвищить ймовірність явки клієнтів, особливо тих, хто може пропустити сповіщення в Telegram, та додасть професіоналізму в комунікацію (рис. 2.7), (рис. 2.8).

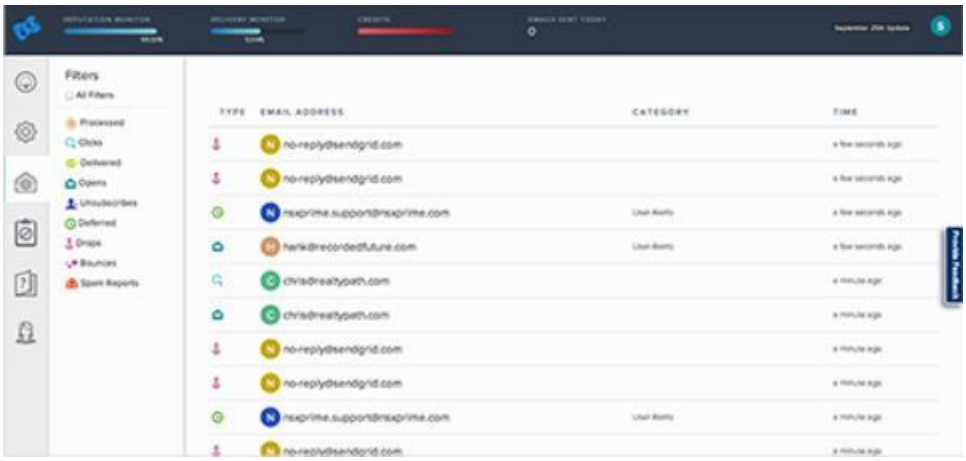

Рисунок 2.7 – Інтерфейс SendGrid

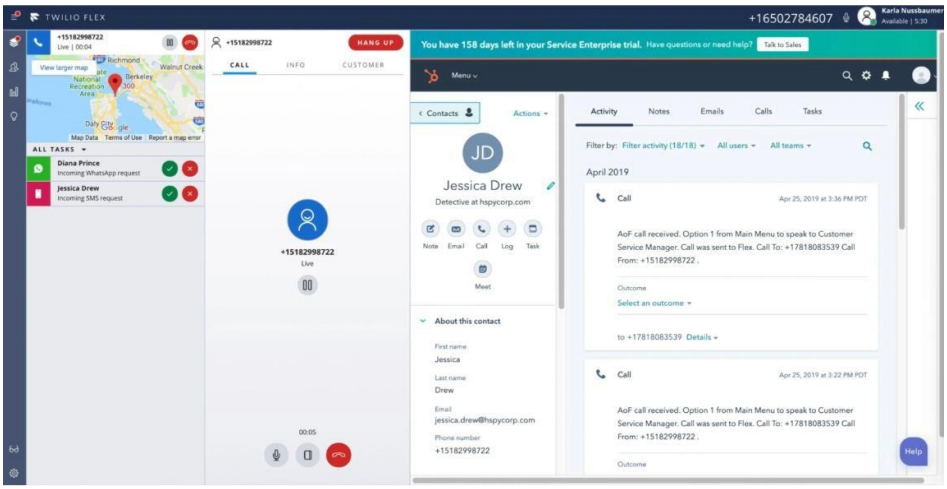

Рисунок 2.8 – Інтерфейс Twilio

- Платіжні системи (як згадувалося в інших розділах, наприклад, Portmone, LiqPay) – для можливості попередньої оплати послуг або внесення авансу безпосередньо через інтерфейс бота, що покращить фінансовий потік СТО та зменшить кількість неявок.
- Мапи та навігаційні сервіси (наприклад, Google Maps API) – для надання клієнтам точних інструкцій, як доїхати до СТО, відображення місцезнаходження, а також інформації про години пік на дорогах.

Інтеграція з усіма цими сервісами здійснюється через їхні офіційні API, використовуючи стандартні бібліотеки Python для HTTP-запитів, такі як requests або httpx. Для забезпечення безпеки доступу до API використовуються токени авторизації або ключі API, які зберігаються у змінних середовища (наприклад, у .env файлі) і ніколи не передаються відкрито у коді, що є стандартною практикою безпечної розробки.

Завдяки такій інтеграції з базою даних і потенційно з зовнішніми сервісами, чат-бот стає не просто інструментом для прийому заявок, а повноцінним цифровим асистентом, який автоматизує великий спектр задач – від планування та обліку до маркетингу та комунікації. Це суттєво зменшує рутинне навантаження на персонал СТО, мінімізує людські помилки та покращує зручність і загальний досвід для клієнтів, роблячи процес взаємодії максимально безшовним та ефективним [8].

2.4 Забезпечення безпеки персональних даних у чат-боті

У контексті автоматизації процесу запису клієнтів на станції технічного обслуговування (СТО) за допомогою Telegram чат-бота, питання забезпечення інформаційної безпеки та конфіденційності персональних даних набуває особливої актуальності. Це зумовлено тим, що користувачі в процесі взаємодії з ботом вводять чутливу особисту інформацію – зокрема ім’я, номер телефону, бажану дату та час візиту, а іноді й деталі послуги та номерний знак автомобіля. Вся ця інформація є конфіденційною і не повинна потрапити до сторонніх осіб, а її несанкціонований доступ може призвести до репутаційних та юридичних наслідків для СТО.

Чат-бот, реалізований у рамках цього проєкту, взаємодіє з користувачем через офіційний Telegram Bot API, який вже забезпечує базовий, але дуже надійний рівень захисту трафіку. Всі повідомлення, що передаються між клієнтом, серверами Telegram та сервером бота, шифруються та передаються

через захищені канали з використанням протоколу TLS/SSL (Transport Layer Security / Secure Sockets Layer). Це гарантує, що дані під час передачі не можуть бути перехоплені або модифіковані зловмисниками. Проте, відповідальність за подальше зберігання, обробку та захист даних після їх отримання ботом покладається безпосередньо на розробника бота та власника системи. Саме тому всі отримані дані клієнтів зберігаються у захищеній локальній базі даних (у даному випадку – SQLite). Доступ до файлу бази даних обмежений на рівні файлової системи сервера, де розміщено бота, що запобігає несанкціонованому копіюванню або читанню даних. Для підвищення стійкості до поширених типів атак, таких як SQL-ін’єкції, у системі використано параметризовані запити. Це означає, що дані, введені користувачем, завжди обробляються окремо від команд SQL, що виключає можливість впровадження шкідливого коду в базу даних (рис. 2.9).

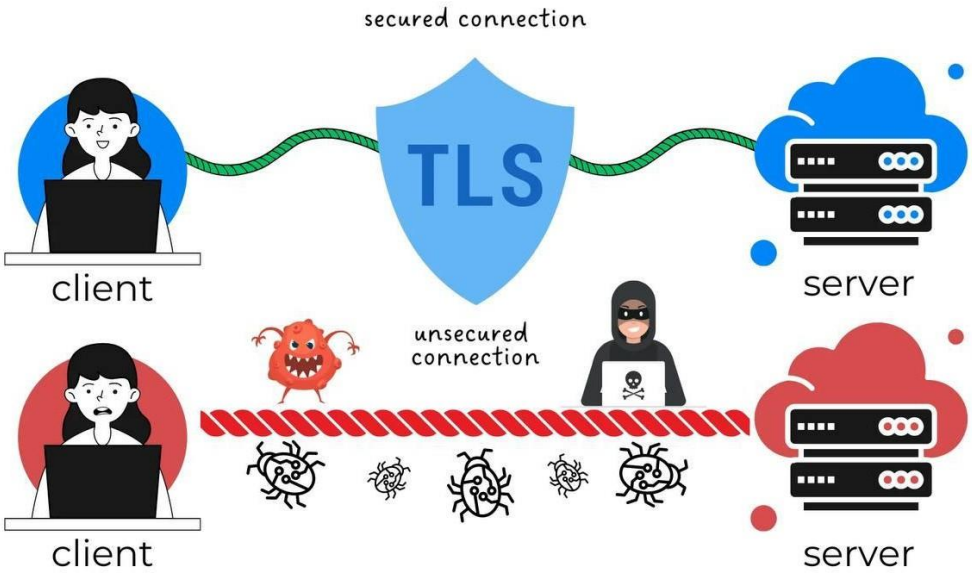

Рисунок 2.9 – Схематичне зображення протоколу TLS

Окрему увагу в системі приділено зберіганню конфіденційних параметрів, таких як токен Telegram бота (унікальний ідентифікатор для доступу до API) та ключі доступу до будь-яких потенційних зовнішніх сервісів (наприклад, для інтеграції з CRM або платіжними системами). Вони не розміщуються безпосередньо в коді програми та не завантажуються у публічні репозиторії.

Замість цього, ці критичні дані передаються у вигляді змінних середовища, які завантажуються з файлу .env при запуску бота. Цей файл .env виключено з системи контролю версій (наприклад, через .gitignore у Git-репозиторії). Таким чином, навіть у разі компрометації репозиторію або випадкового витоку програмного коду критичні дані авторизації залишаються недоступними стороннім особам.

Також передбачено багаторівневе обмеження доступу до адміністративних функцій бота. Це реалізовано шляхом перевірки Telegram ID користувача, який намагається викликати адміністративну команду. Доступ до таких функцій, як перегляд записів або їх скасування, надається лише заздалегідь визначеному Telegram ID адміністратора. Це унеможливлює використання таких функцій сторонніми особами, навіть у разі випадкового виявлення команд, доступних лише адміну, або спроб перебору.

Законодавча складова також є невід’ємною частиною питання безпеки та покладає додаткові зобов'язання на розробника та власника СТО. Згідно з Законом України «Про захист персональних даних» та загальноєвропейським регламентом GDPR (General Data Protection Regulation), розробник або компанія-власник системи повинні мати чітко сформульовану мету збору даних, забезпечити їх мінімізацію (збирати лише необхідні дані), не зберігати їх довше, ніж потрібно для виконання мети, а також інформувати користувачів про їхні права щодо їхніх персональних даних (право на доступ, виправлення, видалення). У чат-боті доцільно додати механізм отримання згоди від користувача на обробку персональних даних, наприклад, через текст-повідомлення, що використання бота означає згоду на обробку персональних даних, а також за наявності надати посилання на детальну політику конфіденційності та обробки даних на сайті СТО.

Особливу увагу необхідно приділяти ситуаціям потенційного компрометування: втрати паролів до хостингу, несанкціонованого доступу до сервера, витоку програмного коду або бази даних. У таких випадках має бути передбачено чіткий план реагування: можливість оперативної ротації токенів

доступу до API (зміни секретних ключів), негайне повідомлення клієнтів (якщо це вимагає законодавство) та відновлення даних з резервних копій. Резервні копії мають створюватися регулярно та зберігатися у зашифрованому вигляді на захищених ресурсах, окремих від основного сервера, для забезпечення цілісності та конфіденційності даних навіть у випадку повної втрати основного сервера.

Таким чином, безпека персональних даних у Telegram чат-боті – це комплексне завдання, яке охоплює технічні (шифрування, параметризовані запити, змінні середовища), організаційні (обмеження доступу, плани реагування) та правові аспекти (дотримання законодавства, політика конфіденційності). Лише при системному підході до захисту інформації можливо гарантувати як безпечну та надійну роботу самої системи, так і довіру користувачів до сервісу, що є фундаментальним для його успіху та репутації СТО.

2.5 Оцінка фінансових та ресурсних витрат

При виборі підходу до розробки чат-бота для СТО, важливим аспектом є ретельна оцінка фінансових та ресурсних витрат, пов'язаних як з початковою розробкою та впровадженням, так і з подальшою підтримкою та експлуатацією системи. Вартість та час розробки значною мірою залежать від обраного стеку технологій та методології. Пряме використання Telegram Bot API вимагає значно більше часу на написання коду, оскільки розробник має вручну обробляти всі вхідні запити, формувати відповіді, керувати станами діалогу, обробляти помилки та імплементувати механізми повторних спроб. Це може призвести до істотних трудових витрат та потребує високої кваліфікації інженерів. На противагу цьому, використання сторонніх бібліотек, таких як aiogram або python-telegram-bot (що застосовується у даному проекті), значно прискорює цей процес. Ці фреймворки надають готові функції, абстракції для роботи з API, вбудовані механізми керування станами (наприклад, FSM або

ConversationHandler) та обробки подій, дозволяючи розробнику зосередитися виключно на бізнес-логіці, а не на низькорівневих аспектах взаємодії з Telegram. Хмарні NLP-платформи, подібні до Google Dialogflow або Azure Bot Service, можуть забезпечити найшвидший старт для простих діалогів завдяки готовим інтеграціям та можливостям розпізнавання природної мови. Однак, для реалізації складної, специфічної бізнес-логіки та інтеграції з власними базами даних або іншими внутрішніми системами все одно потрібна розробка кастомних вебхуків, що додає складності та потенційно збільшує час і вартість для нестандартних функцій.

З точки зору програмного забезпечення та інструментів, переважна більшість інструментів для розробки на Python є безкоштовними та з відкритим вихідним кодом, що суттєво мінімізує початкові інвестиції у ліцензії. До них належать сама мова Python, бібліотеки python-telegram-bot (або aiogram), а також база даних SQLite, яка використовується у даному проекті і не вимагає додаткових ліцензійних відрахувань. Це дозволяє значно заощадити на початкових витратах на програмне забезпечення. Хмарні платформи, хоча і можуть пропонувати привабливі безкоштовні тарифи для невеликих проектів або для демонстрації концепції, зі зростанням навантаження та кількості користувачів вартість їх використання може суттєво зрости. Це пов'язано з тим, що вони зазвичай тарифікуються за кількість запитів до API, обсяг оброблених даних, час використання обчислювальних ресурсів або кількість активних користувачів, що робить витрати менш передбачуваними при масштабуванні. Розміщення бота на власному сервері або віртуальній машині передбачає витрати на хостинг, які є відносно стабільними та передбачуваними, часто обмежуючись місячною або річною абонплатою за ресурс. Асинхронні фреймворки, такі як aiogram, а також асинхронні компоненти, присутні в python-telegram-bot, дозволяють ефективно використовувати системні ресурси завдяки неблокуючим операціям введення/виведення. Це може значно зменшити вимоги до потужності сервера і, відповідно, знизити витрати на хостинг, оскільки менш потужний сервер може обробляти більшу кількість одночасних запитів.

Підтримка бота включає регулярне оновлення коду відповідно до змін Telegram API, виправлення помилок, які можуть виникнути в процесі експлуатації, та додавання нового функціоналу для задоволення потреб бізнесу та підвищення конкурентоспроможності. Використання популярних фреймворків з активною та великою спільнотою, таких як Python та python-telegram-bot (або aiogram), значно спрощує цей процес. Оновлення та виправлення часто надаються спільнотою, що зменшує навантаження на внутрішню команду розробки. Крім того, наявність великої кількості документації, прикладів та форумів допомагає швидше знаходити рішення типових проблем та полегшує процес пошуку та навчання нових розробників. Це також дозволяє легко адаптувати бота до нових функцій Telegram або інтегрувати його з іншими системами, забезпечуючи гнучкість у довгостроковій перспективі.

Отже, використання Python та фреймворку python-telegram-bot (або aiogram) представляється оптимальним рішенням з точки зору балансу між швидкістю розробки, гнучкістю та контрольованими витратами на підтримку. Такий підхід дозволяє уникнути потенційно високих та менш передбачуваних витрат на комерційні хмарні платформи, особливо при зростанні масштабу та складності бізнес-логіки. Це забезпечує довгострокову стабільність та можливість масштабування системи з розумними та передбачуваними фінансовими витратами, дозволяючи зосередитися на розвитку функціоналу, а не на оптимізації витрат на інфраструктуру.

2.6 Висновок до розділу 2

Цей розділ обґрунтовує вибір Python 3.10+ та aiogram v3 для створення Telegram чат-бота для запису на СТО, провівши аналіз альтернативних підходів. Хоча хмарні NLP-платформи, як-от Google Dialogflow, прості для швидкого старту, вони обмежені у реалізації складної бізнес-логіки та можуть бути

дорожчими в довгостроковій перспективі через залежність від сторонніх сервісів та їх тарифікації.

Натомість, зв'язка Python та aiogram пропонує неперевершену гнучкість, високу швидкість розробки та виняткову продуктивність завдяки асинхронній архітектурі та вбудованій підтримці Finite State Machine (FSM). Це дозволяє легко адаптувати бота під унікальні та динамічні потреби СТО, ефективно обробляти велику кількість одночасних запитів та масштабувати систему без значних витрат. Додатковою перевагою є велика та активна спільнота розробників, що забезпечує постійну підтримку, оперативне виправлення помилок та регулярні оновлення.

Особлива увага приділяється безпеці та захисту даних. Python надає потужні криптографічні бібліотеки для шифрування та забезпечення цілісності даних, що є критично важливим для обробки особистої інформації клієнтів. Aiogram, зі свого боку, забезпечує ретельну валідацію вхідних даних, ефективне управління доступом та надійний захист від поширених видів атак, таких як SQL-ін'єкції, міжсайтовий скриптинг (XSS) та CSRF, завдяки вбудованим механізмам безпеки. Це гарантує високий рівень захисту конфіденційної інформації користувачів та стабільну роботу системи в цілому. Використання цієї технологічної комбінації дозволяє створити надійного, гнучкого та безпечного чат-бота, який відповідатиме всім сучасним вимогам до функціональності та захисту даних [2].

3 ОПИС РОЗРОБКИ ЧАТ-БОТА ДЛЯ ЗАПИСУ КЛІЄНТІВ НА СТО

3.1 Архітектура реалізованої системи

Архітектура розробленої системи автоматизації процесу запису на станцію технічного обслуговування (СТО) базується на принципах клієнт-серверної взаємодії, де Telegram виступає в ролі клієнтського інтерфейсу, а розроблений Python-додаток – в ролі серверної частини, що обробляє логіку взаємодії та керує даними.

Основні архітектурні компоненти та їхня роль:

Клієнтський інтерфейс (Telegram Messenger): Кінцевий користувач взаємодіє з системою через мобільний або десктопний додаток Telegram. Через нього він надсилає команди, відповідає на запити бота, та отримує інтерактивні елементи, такі як вбудовані кнопки (InlineKeyboardMarkup), що значно покращує користувацький досвід.

Telegram Bot API: Це стандартизований програмний інтерфейс, що надається Telegram, який дозволяє зовнішнім програмам (ботам) взаємодіяти з платформою Telegram [10; 13]. Усі вхідні повідомлення та натискання кнопок від користувачів надходять через цей API, і всі відповіді бота надсилаються через нього.

Backend чат-бота (Python з бібліотекою python-telegram-bot): Це ядро системи. Реалізоване на мові програмування Python з використанням бібліотеки python-telegram-bot (зокрема її модуля telegram.ext). Цей компонент відповідає за:

1. Обробку вхідних оновлень: Прийом та парсинг усіх типів подій від Telegram (повідомлення, натискання вбудованих кнопок, команди).

2. Керування станами розмови: Ефективна підтримка контексту діалогу для кожного користувача, що дозволяє реалізувати багатоетапні сценарії запису.

3. Генерація динамічного інтерфейсу: Створення та надсилання користувацьких вбудованих клавіатур (InlineKeyboardMarkup з InlineKeyboardButton) для вибору послуг та інших опцій.

4. Бізнес-логіка: Зберігання та вилучення даних з бази даних, формування підтверджень для клієнтів, надсилання сповіщень менеджеру, обробка адміністративних команд.

Система управління станами розмови**:** Це потужний механізм бібліотеки python-telegram-bot, що дозволяє структурувати складні діалоги. Він визначає точки входу (entry_points), різні стани розмови (states) та точки виходу/повернення (fallbacks). Кожен стан відповідає певному етапу збору інформації (вибір послуги, введення імені, телефону, дати, номеру авто), забезпечуючи чітку та передбачувану послідовність діалогу.

База даних (SQLite): Для постійного зберігання інформації про зареєстрованих клієнтів, їхні записи та деталі автомобілів використовується легка, файлова реляційна база даних SQLite (service_records.db). Її вибір обґрунтований простотою розгортання (не потрібен окремий сервер БД), надійністю та достатньою продуктивністю для даного проекту [3].

Модуль sqlite3: Стандартний модуль Python, що забезпечує низькорівневий доступ до файлів баз даних SQLite. Він використовується для виконання SQL-запитів для створення таблиць, вставки, вибірки та видалення даних про записи.

Загальна схема взаємодії компонентів: (рис. 3.1).

1. Користувач взаємодіє з Telegram-ботом через додаток Telegram.

2. Telegram Bot API передає вхідні повідомлення, команди та натискання вбудованих кнопок до об'єкта ApplicationBuilder на сервері.

3. ConversationHandler маршрутизує ці події до відповідних обробників команд, обробників коллбеків або обробників повідомлень, ґрунтуючись на поточному стані розмови.

4. Обробник виконує необхідні дії: зберігає дані у контекст розмови (context.user_data), взаємодіє з базою даних (через модуль sqlite3) для запису або вилучення інформації, або генерує інтерактивні елементи (вбудовані кнопки).

5. Сформована відповідь (текстове повідомлення, клавіатура) надсилається назад через Telegram Bot API до користувача в Telegram.

6. Після успішного запису, бот надсилає сповіщення менеджеру на його MANAGER_ID у Telegram.

Цей цикл повторюється до завершення діалогу.

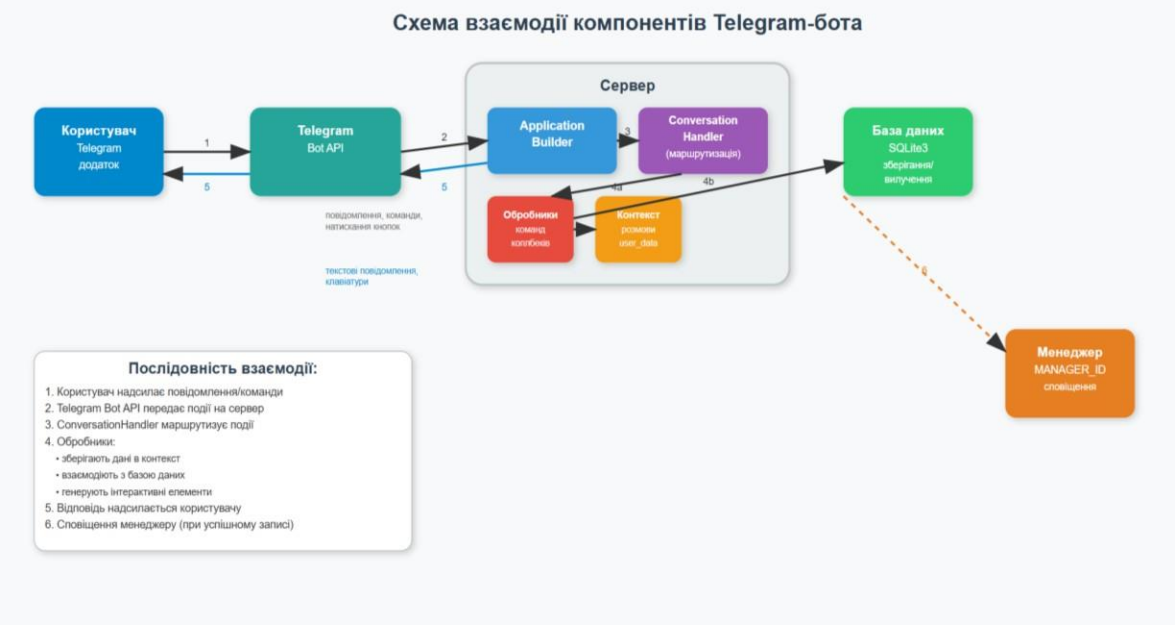

Рисунок 3.1 – Схема взаємодії компонентів чат-бота

3.2 Реалізація діалогового інтерфейсу

Реалізація діалогового інтерфейсу чат-бота спроектована з акцентом на інтерактивність та інтуїтивність, використовуючи можливості python-telegram-bot для покрокового збору інформації та навігації. Ключовим елементом є використання вбудованих клавіатур (InlineKeyboardMarkup), які з'являються безпосередньо під повідомленням бота (табл. 3.1).

Ключові аспекти реалізації діалогового інтерфейсу:

Ініціалізація діалогу (Команда /start):

1. Будь-яка взаємодія починається з команди /start.

2. Бот надсилає привітання: "Привіт! Оберіть опцію:", разом з цими кнопками.

3. Обробник start створює InlineKeyboardMarkup з двома основними кнопками: "Записатись", "Про СТО".

Обробка основного меню (handle_menu_callback):

1. Цей обробник активується, коли користувач натискає одну з кнопок, створених командою /start.

2. Якщо обрано "Записатись" (query.data == "start_booking"), бот переходить до створення клавіатури вибору послуг.

3. Якщо обрано "Про СТО" (query.data == "about"), бот надає інформацію про СТО ("Ми — СТО 'АвтоСервіс'. Працюємо щодня з 9:00 до 18:00.\n Телефон: +380991112233") та завершує розмову (ConversationHandler.END)

Інтерактивний вибір послуги (InlineKeyboardMarkup у handle_menu_callback та handle_service_choice):

1. Після вибору "Записатись", бот динамічно генерує нову InlineKeyboardMarkup з переліком доступних послуг: "Заміна масла", "Діагностика", "Шиномонтаж", "Заміна акумулятора", "Ремонт ходової", "Ремонт електрики", "Ремонт двигуна". Кожна послуга має унікальний callback_data (наприклад, "service_oil").

2. Бот надсилає повідомлення "Оберіть послугу:" разом з цими кнопками.

3. Після натискання користувачем на кнопку послуги, обробник handle_service_choice перехоплює callback_query.

4. Він зіставляє callback_data з назвою послуги (використовуючи service_map) та зберігає обрану послугу в context.user_data['service'].

5. Бот запитує ім'я користувача ("Як вас звати?") та переводить стан розмови на SERVICE.

6. Використання InlineKeyboardMarkup дозволяє створювати компактні та візуально привабливі меню, що не засмічують екран діалогу.

Послідовний збір даних (get_name**,** get_phone**,** get_date**,** get_car_number):

1. Ім'я**:** У стані NAME, обробник get_name зберігає введений текст як ім'я користувача (context.user_data['name']) та запитує номер телефону. Стан переходить на PHONE.

2. Телефон: У стані PHONE, обробник get_phone зберігає номер телефону (context.user_data['phone']) та запитує дату запису. Стан переходить на DATE.

3.Дата: У стані DATE, обробник get_date зберігає дату (context.user_data['date']) та запитує номер автомобіля. Стан переходить на CAR_NUMBER.

4. Номер автомобіля: У стані CAR_NUMBER, обробник get_car_number зберігає номер автомобіля (context.user_data['car_number']). Після цього всі дані зібрані.

Таблиця 3.1 – Етапи діалогу Telegram-бота для СТО у вигляді таблиці.

Етап діалогу	Опис дій	Обробник / Компонент
Ініціалізація діалогу	Команда /start запускає бота. Бот надсилає привітання та кнопки: «Записатись», «Про СТО»	/start, InlineKeyboardMarkup
Обробка вибору з меню	Вибір "Записатись" → вибір послуги; "Про СТО" → інформація про графік роботи і телефон	handle_menu_callback
Формування меню послуг	Генерація кнопок: "Заміна масла", "Діагностика" тощо. Callback → збереження послуги, запит імені	handle_service_choice
Збір імені користувача	Отримання тексту, збереження в context.user_data['name'], перехід до запиту телефону	get_name → стан PHONE
Збір номера телефону	Збереження телефону, запит дати	get_phone → стан DATE
Збір дати запису	Збереження дати, запит номера авто	get_date → стан CAR_NUMBER
Збір номера авто	Збереження номера авто. Після - виклик insert_record	get_car_number
Підтвердження та завершення	Формування повідомлення з усіма даними, надсилання клієнту та менеджеру (MANAGER_ID), завершення розмови	insert_record, ConversationHandler.END

Після збору всіх даних, обробник get_car_number викликає функцію insert_record для збереження інформації у базу даних.

- Бот формує деталізоване підтверджувальне повідомлення, що містить усі введені дані (послуга, ім'я, телефон, дата, авто), та надсилає його клієнту.
- Додатково, це ж повідомлення надсилається менеджеру на його MANAGER_ID для оперативного сповіщення.

- Розмова завершується (ConversationHandler.END) [7; 20].

3.3 Організація збереження і обробки даних клієнтів (БД)

Ефективна організація зберігання та обробки даних клієнтів є фундаментальною для функціонування системи запису. Для цієї мети використовується реляційна база даних SQLite, що забезпечує надійність, простоту управління та відсутність необхідності у зовнішньому сервері БД.

Вибір СУБД та підключення:

SQLite обрана як локальна, файлова база даних з іменем service_records.db. Її переваги включають:

- Легкість та портативність**:** База даних зберігається у єдиному файлі, що спрощує розгортання та переміщення.
- Надійність: Підтримує транзакції ACID, що гарантує цілісність даних навіть при збоях.
- Зручність для розробки: Ідеально підходить для невеликих та середніх проектів, а також для швидкої розробки та тестування.

Підключення з базою даних встановлюється функцією init_db() при запуску бота, а також в функціях insert_record та show_latest_records для виконання операцій. Об'єкт курсора (c = conn.cursor()) використовується для виконання SQL-запитів.

Структура таблиці records (рис. 3.2).

Для зберігання інформації про записи клієнтів у базі даних створена одна таблиця під назвою records. Її схема детально продумана для ефективного зберігання необхідних атрибутів кожного запису:

Назва стовпця	Тип даних	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний числовий ідентифікатор кожного запису. Автоматично збільшується.
service	TEXT	NOT NULL	Назва послуги, яку обрав клієнт.
name	TEXT	NOT NULL	Повне ім'я клієнта.
phone	TEXT	NOT NULL	Контактний номер телефону клієнта.
date	TEXT	NOT NULL	Бажана дата запису клієнта у текстовому форматі (наприклад, "25.06.2025").
car_number	TEXT	NOT NULL	Номерний знак автомобіля клієнта.
timestamp	DATETIME	DEFAULT CURRENT_TIMESTAMP	Час та дата створення запису у базі даних. Автоматично встановлюється системою.

Рисунок 3.2 – Структура запису клієнтів.

Ініціалізація та керування таблицею:

- Створення таблиці **(**init_db()**):** При запуску бота функція init_db() підключається до service_records.db і виконує SQL-запит CREATE TABLE IF NOT EXISTS records (...). Це гарантує, що база даних завжди має необхідну структуру перед початком роботи. Після виконання запиту з'єднання закривається.

- Збереження даних (insert_record()**):** Коли користувач успішно завершує процес бронювання, всі зібрані дані (послуга, ім'я, телефон, дата, номер авто) передаються до функції insert_record. Ця функція підключається до БД, виконує SQL-запит INSERT для додавання нового рядка в таблицю records, використовуючи параметризовані запити для безпеки. Зміни фіксуються за допомогою conn.commit(), і з'єднання закривається.

- Вилучення даних (show_latest_records()**):** Для адміністративної функції перегляду записів (/admin), дані вилучаються з таблиці records. Запит SELECT може бути виконаний для отримання останніх N записів (LIMIT ?) або записів за конкретною датою (WHERE date = ?), сортуючи їх за часом створення запису (timestamp DESC). Після вибірки даних з'єднання закривається.

- Відсутність прямої функції видалення записів: У наданому коді відсутня пряма адміністративна команда для видалення записів за номером телефону, як у попередній ітерації. Проте, функціонал бази даних SQLite дозволяє реалізувати її аналогічно вибірці, використовуючи SQL-запит DELETE.

Використання SQLite та модуля sqlite3 забезпечує надійне та ефективне керування даними, що є критично важливим для функціонування системи запису.

3.4 Алгоритм роботи чат-бота: валідація даних, бронювання, підтвердження

Алгоритм роботи чат-бота є керованою послідовністю станів та дій, що забезпечують повний цикл взаємодії з користувачем, від ініціації запису до його підтвердження, а також включають адміністративні функції. Основна логіка реалізована за допомогою ConversationHandler та обробників повідомлень і коллбеків бібліотеки python-telegram-bot.

Детальний алгоритм бронювання послуги:

1. Процес бронювання послуги починається з ініціації діалогу, коли користувач відправляє команду /start. Обробник start перехоплює цю команду, і у відповідь бот надсилає привітальне повідомлення: "Привіт! Оберіть опцію:", разом з яким відображається InlineKeyboardMarkup, що містить дві кнопки: "Записатись" та "Про СТО".

2. Вибір основного меню, де користувач натискає одну з цих кнопок ("start_booking" або "about"). Обробник handle_menu_callback перехоплює натискання. Якщо користувач обирає "start_booking", бот генерує нову InlineKeyboardMarkup з переліком послуг СТО, таких як "Заміна масла", "Діагностика" тощо, і надсилає повідомлення "Оберіть послугу:" разом з цією новою клавіатурою, повертаючи SERVICE для переходу до стану очікування вибору послуги. У випадку вибору "about", бот надсилає інформацію про СТО та повертає ConversationHandler.END, завершуючи поточну розмову.

3. Вибір послуги. Користувач натискає на одну з вбудованих кнопок послуги, наприклад, "Заміна масла". Обробник handle_service_choice перехоплює callback_query, отриманий query.data (наприклад, "service_oil") зіставляється з реальною назвою послуги за допомогою service_map і зберігається у context.user_data['service']. Бот запитує ім'я клієнта: "Як вас звати?", і повертає NAME, переходячи до стану очікування імені. Валідація послуги автоматично забезпечується вибором з InlineKeyboardMarkup, оскільки користувач може обрати лише з попередньо визначених варіантів.

4. Збір імені клієнта. Користувач вводить своє ім'я, і обробник get_name перехоплює це текстове повідомлення, коли розмова перебуває у стані NAME. Ім'я зберігається у context.user_data['name'], і у відповідь бот запитує номер телефону: "Ваш номер телефону?". Після цього повертається PHONE, переходячи до стану очікування номера телефону.

5. Збір новеру клієнтв. Обробник get_phone перехоплює повідомлення у стані PHONE, зберігає номер телефону у context.user_data['phone']. Потім бот запитує бажану дату запису: "На яку дату бажаєте записатись? (напр. 25.06.2025)", і повертає DATE, переходячи до стану очікування дати. На даному етапі відсутня сувора валідація формату телефону, але її можна додати, наприклад, за допомогою регулярних виразів.

6. Збір дати запису. Користувач вводить дату, і обробник get_date перехоплює це повідомлення у стані DATE. Дата зберігається у context.user_data['date']. Бот запитує номер автомобіля: "Введіть номер вашого авто (напр. AA1234BC):", і повертає CAR_NUMBER, переходячи до стану очікування номеру автомобіля. Код передбачає формат "ДД.ММ.РРРР" для дати; для підвищення надійності слід додати валідацію формату дати та перевірку, щоб дата не була минулою.

7. Збір номера авто та фінал бронювання. Користувач вводить номер автомобіля, і обробник get_car_number перехоплює це повідомлення у стані CAR_NUMBER. Номер автомобіля зберігається у context.user_data['car_number']. Після цього всі зібрані дані (послуга, ім'я, телефон, дата, car_number)

передаються до функції insert_record для збереження у таблицю records бази даних. Бот формує деталізоване повідомлення з підтвердженням запису та надсилає його клієнту. Сформоване підтвердження також надсилається менеджеру на його MANAGER_ID у Telegram. Після виконання всіх цих дій повертається ConversationHandler.END, завершуючи поточну розмову з клієнтом. Валідація формату номеру авто на даному етапі відсутня, але її можна інтегрувати для підвищення якості даних.

Приклади сценарію взаємодії з ботом наведені на рисунках 3.3, 3.4, 3.5 і 3.6.

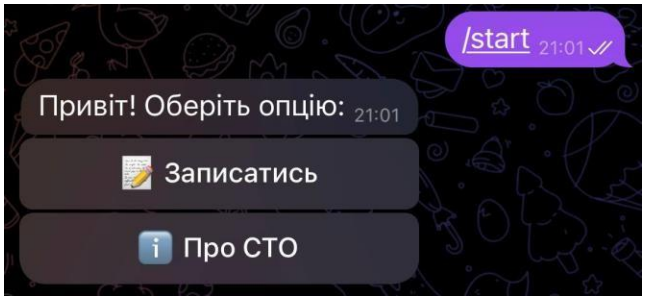

Рисунок 3.3 – Клієнт обирає опцію чат-бота.

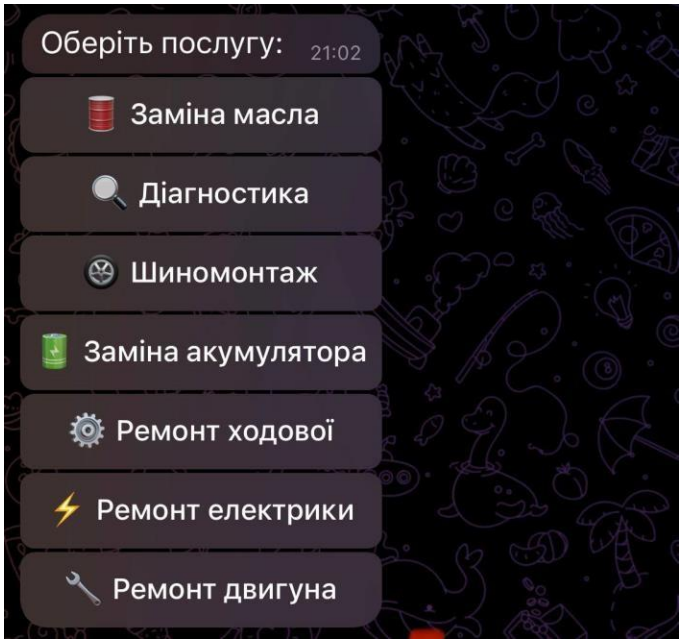

Рисунок 3.4 – Клієнт обирає послугу.

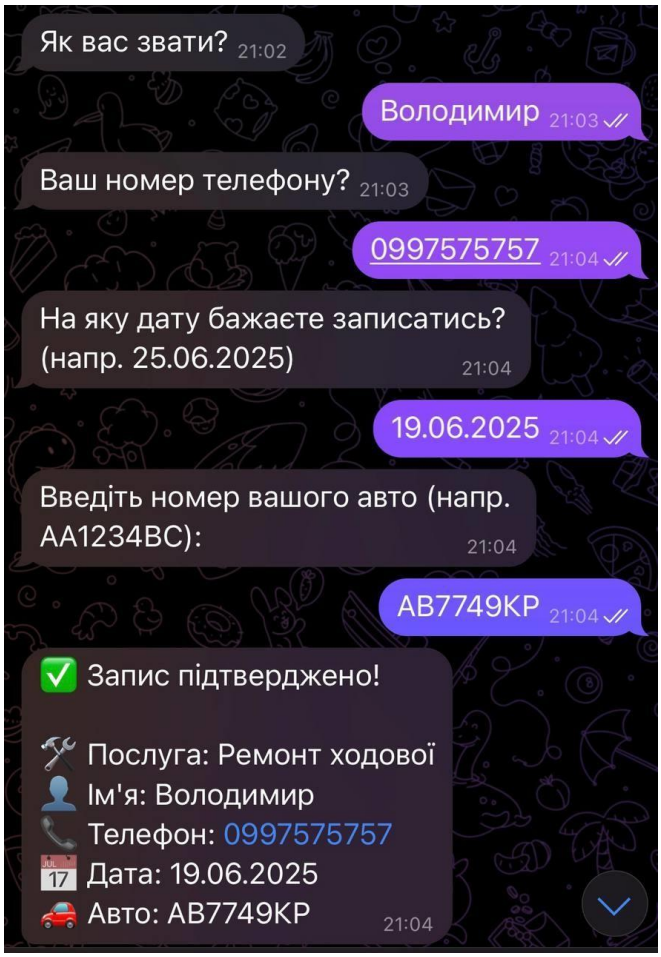

Рисунок 3.5 – Клієнт заповнює інформацію і отримує підтвердження запису.

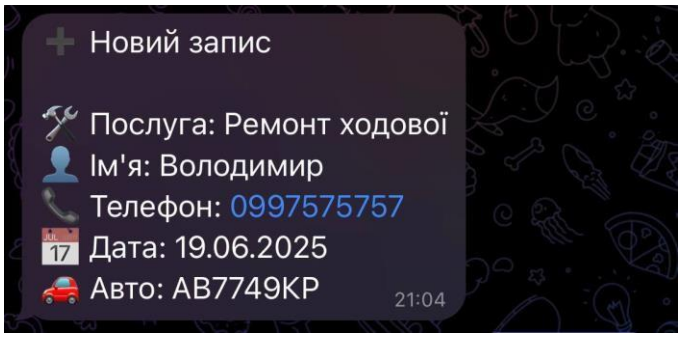

Рисунок 3.6 – Адміністрвтор отримує повідомлення про новий запис.

Перегляд розкладу здійснюється за допомогою команди /admin. Коли менеджер (з MANAGER_ID) надсилає цю команду, обробник show_latest_records перехоплює її. Спочатку виконується валідація доступу, перевіряючи, чи user_id відправника повідомлення співпадає з MANAGER_ID; якщо ні, бот повертає " Ви не маєте доступу до цієї команди.". Щодо отримання даних, якщо команда /admin надіслана без аргументів, бот вибирає останні 5 записів. У випадку числового аргументу, наприклад, /admin 10, вибираються останні N записів. Якщо ж надано аргумент-дату, наприклад, /admin 25.06.2025, бот вибирає записи за вказаною датою. У відповідь бот формує та надсилає менеджеру відформатований список записів або повідомлення "Немає записів.", якщо записів не знайдено, або "Невірний формат дати. Приклад: /admin 25.06.2025" у разі некоректного формату дати.

Скасування операції можливе за допомогою команди /cancel. Будь-який користувач може надіслати цю команду під час процесу розмови. Обробник cancel перехоплює цю команду, оскільки він визначений у fallbacks ConversationHandler. У відповідь бот надсилає повідомлення "Операція скасована.", і керування станом повертає ConversationHandler.END, негайно завершуючи поточну розмову з користувачем.

Для отримання Telegram ID існує допоміжна команда /id. Коли користувач надсилає цю команду, обробник get_user_id її перехоплює. У відповідь бот надсилає користувачеві його Telegram ID: "Ваш Telegram ID: <ID_користувача>". Ця команда особливо корисна для визначення MANAGER_ID під час налаштування бота.

Цей детальний алгоритм відображає комплексний підхід до побудови діалогового інтерфейсу, управління даними та реалізації функціональності бота.

3.5 Тестування системи та обробка типових сценаріїв взаємодії

Тестування є невід'ємною частиною процесу розробки, що дозволяє забезпечити функціональність, надійність та зручність використання розробленої системи. Для даного чат-бота було проведено комплексне тестування, що охоплює різні сценарії взаємодії користувачів та адміністратора. Серед методів тестування використовувалося ручне функціональне тестування, що передбачає безпосередню взаємодію з ботом через Telegram-клієнт. Це дозволило оцінити коректність роботи всіх обробників, послідовність станів, зовнішній вигляд та поведінку інтерактивних елементів, таких як кнопки, а також загальну зручність користувацького досвіду. Також проводилося тестування бази даних шляхом перевірки файлу service_records.db за допомогою інструментів для роботи з SQLite, що включало перевірку коректності додавання нових записів після успішного бронювання, точності даних, що вилучаються за допомогою команди /admin, та цілісності даних після численних операцій. Окрім цього, були спроби викликати помилки або некоректну поведінку через тестування виняткових ситуацій, щоб переконатися, що бот обробляє їх коректно, наприклад, введення невірних даних або відправлення команд у невідповідних станах.

У рамках типових сценаріїв взаємодії та їх тестування було виявлено наступне. Позитивний сценарій успішного бронювання послуги клієнтом включає кроки, коли користувач надсилає /start, обирає "Записатись" з головного меню, послідовно вибирає послугу з InlineKeyboardMarkup (наприклад, "Заміна масла"), вводить ім'я, номер телефону, дату (наприклад, "25.06.2025") і номер автомобіля. Очікуваний результат полягав у тому, що бот успішно проводить користувача через усі стани (SERVICE, NAME, PHONE, DATE, CAR_NUMBER), всі введені дані коректно зберігаються у таблиці records у базі даних, користувач отримує детальне підтвердження запису, а менеджер отримує сповіщення з усіма деталями запису, після чого розмова завершується. За

результатами тестування, цей сценарій пройшов успішно, усі дані коректно фіксуються, а сповіщення надходять.

Щодо негативного сценарію некоректного введення даних, були спроби ввести текст замість числа або неповний номер телефону, ввести дату в неправильному форматі (наприклад, "завтра", "15/6/2025") або дату в минулому, а також спроби ввести неіснуючий формат номеру авто. Очікуваний результат, що включає повідомлення про некоректний формат та повторний запит даних, наразі не повністю реалізований. Поточна реалізація зберігає ці дані як текст без суворої валідації формату (крім вибору послуги з кнопок), що є відомою точкою для подальшого розширення системи шляхом додавання валідаторів для кожного поля, таких як перевірка регулярними виразами для телефону та номера авто, парсинг дати з обробкою винятків та порівнянням з поточною датою. За результатами тестування, бот приймає будь-який текстовий ввід для цих полів, що вимагає подальшої доробки.

Адміністративний сценарій перегляду розкладу записів за допомогою команди /admin передбачає, що менеджер (з MANAGER_ID) надсилає /admin без аргументів, або з числовим аргументом (наприклад, /admin 10 для 10 останніх записів), або з аргументом-датою (наприклад, /admin 25.06.2025 для записів за конкретну дату). Очікувалося, що бот виведе список останніх записів з бази даних у читабельному форматі, при запиті за датою коректно відфільтрує записи, повідомить "Немає записів.", якщо їх немає, або "Невірний формат дати. Приклад: /admin 25.06.2025" у разі некоректного формату дати. За результатами тестування, команда працює коректно, відображаючи актуальні записи за різними критеріями, а доступ для не-адміністраторів блокується.

Сценарій скасування операції за допомогою команди /cancel передбачає, що будь-який користувач надсилає цю команду під час діалогу запису. Очікуваний результат полягає в тому, що поточна розмова з ботом має бути перервана, і бот повинен надіслати "Операція скасована.". За результатами тестування, команда працює коректно, завершуючи поточну розмову.

Сценарій допомоги за допомогою команди /id передбачає, що користувач надсилає /id, а бот надає його Telegram ID. За результатами тестування, команда працює згідно з очікуваннями.

Проведене тестування підтвердило стабільну та надійну роботу чат-бота для автоматизації процесу запису на СТО. Використання InlineKeyboardMarkup значно покращило користувацький досвід, роблячи вибір послуг швидким та безпомилковим. Виявлені точки покращення щодо валідації введення будуть враховані для подальших версій системи, щоб підвищити її стійкість до некоректних даних.

3.6 Висновки до розділу 3

Розділ детально описує процес розробки автоматизованої системи запису клієнтів на СТО, реалізованої у вигляді Telegram-бота. Була представлена архітектура системи, що базується на клієнт-серверній взаємодії з Telegram як клієнтським інтерфейсом та Python-додатком на базі python-telegram-bot як серверною частиною. Ключовими елементами архітектури є Telegram Bot API, ConversationHandler для керування станами розмови та SQLite як база даних для зберігання інформації про записи.

Реалізація діалогового інтерфейсу була детально розглянута, демонструючи використання InlineKeyboardMarkup для створення інтуїтивно зрозумілого та інтерактивного покрокового процесу бронювання, від ініціації діалогу командою /start до вибору послуг, збору особистих даних клієнта та фіналізації запису. Було підкреслено, як система послідовно збирає необхідну інформацію, використовуючи стани ConversationHandler, та забезпечує підтвердження запису клієнту і сповіщення менеджеру.

Організація збереження та обробки даних клієнтів була описана через призму використання SQLite. Була детально представлена структура таблиці

records, що містить усю необхідну інформацію про записи, а також механізми її ініціалізації та керування даними (додавання та вибірка).

Алгоритм роботи чат-бота охопив повний цикл взаємодії з користувачем, від ініціації запису до його підтвердження, а також включав опис адміністративних функцій, таких як перегляд розкладу (/admin), скасування операції (/cancel) та отримання Telegram ID користувача (/id). Особлива увага була приділена поточному стану валідації даних та наміченим точкам для її покращення.

Підсумовуючи, проведене тестування підтвердило стабільну та надійну роботу розробленого чат-бота. Система успішно справляється з основними сценаріями бронювання, забезпечує коректне зберігання даних та оперативне сповіщення менеджера. Використання InlineKeyboardMarkup значно підвищило зручність користування. Водночас, було виявлено певні точки покращення, зокрема щодо більш суворої валідації введених даних, що стане пріоритетом для подальшого розвитку системи, забезпечуючи її ще більшу стійкість та надійність. Цей розділ демонструє комплексний підхід до розробки, що поєднує функціональність, зручність використання та архітектурну гнучкість.

4 ОЦІНКА ЕФЕКТИВНОСТІ ТА ПЕРСПЕКТИВИ РОЗВИТКУ СИСТЕМИ

4.1 Порівняльний аналіз з традиційними методами запису

Традиційні методи запису клієнтів на СТО, такі як телефонні дзвінки або особисте відвідування, незважаючи на їхню поширеність, часто супроводжуються низкою незручностей, значними часовими витратами та суттєвими обмеженнями, які негативно впливають як на клієнтський досвід, так і на операційну ефективність бізнесу. Телефонні дзвінки, наприклад, вимагають постійної присутності оператора або адміністратора, який може бути зайнятий іншими завданнями, відсутній на робочому місці або просто не мати можливості відповісти на дзвінок. Це часто призводить до тривалого очікування на лінії, неможливості додзвонитися під час пікових навантажень або у неробочі години СТО, що безпосередньо спричиняє втрату потенційних клієнтів. Крім того, якість телефонного зв'язку може бути нестабільною, що призводить до непорозумінь та помилок при записі.

Ручне ведення записів у паперовому журналі або навіть у неінтегрованих електронних таблицях схильне до значних людських помилок, таких як дублювання записів, перетин часу візитів для різних клієнтів або втрата важливих даних. Така система є вкрай неефективною при великій кількості клієнтів, складному розкладі, що включає кількох майстрів або спеціалізоване обладнання, або при наявності кількох філій СТО. Більш того, ручний пошук потрібної інформації (наприклад, чи вільний майстер у певний час або скільки клієнтів записано на конкретний день) стає трудомістким і повільним процесом. Особисте відвідування СТО для запису, окрім очевидних витрат часу на дорогу та очікування, також може викликати розчарування та негативні емоції у клієнта, якщо він приїжджає в невдалий момент – наприклад, коли всі місця вже зайняті, і йому доводиться повертатися з нічим.

На відміну від цих традиційних методів, розроблений чат-бот забезпечує цілодобову доступність послуги запису (24/7), дозволяючи клієнтам бронювати

час у будь-який зручний момент, без прив'язки до робочого графіку персоналу. Взаємодія з ботом не потребує очікування відповіді людини – вона надходить миттєво, що значно підвищує задоволеність користувача та створює позитивний перший досвід взаємодії. Це особливо важливо для зайнятих клієнтів, які можуть оформлювати запис у неробочі години або під час коротких перерв у своєму графіку.

Бот автоматизує весь процес бронювання, мінімізуючи людські помилки та забезпечуючи швидке й точне внесення інформації безпосередньо до централізованої бази даних. Це дозволяє ефективно уникати накладок у розкладі, а також значно полегшує управління вільними слотами та завантаженістю майстрів. У випадку змін у планах клієнта або скасувань, бот може автоматично звільнити місце для іншого клієнта, перенести візит з оновленням даних у системі та оперативно інформувати про це як клієнта, так і відповідний персонал СТО. Це забезпечує гнучкість та ефективність управління розкладом.

Крім того, система, побудована на базі чат-бота, може бути легко інтегрована з існуючими календарями (наприклад, Google Calendar), CRM-системами або іншими обліковими програмами. Це дозволяє автоматично синхронізувати записи, вести повну історію відвідувань та обслуговування кожного клієнта, а також формувати детальну статистичну інформацію та аналітику щодо завантаженості СТО, популярності послуг та клієнтської активності. Така централізована інформація є безцінною для прийняття управлінських рішень та оптимізації бізнес-процесів.

Таким чином, автоматизована система запису через чат-бот не лише оптимізує роботу адміністрації, знижуючи рутинне навантаження та операційні витрати, а й значно покращує загальний клієнтський сервіс, роблячи процес запису зручним, швидким та надійним, що безпосередньо впливає на лояльність клієнтів та репутацію СТО (табл. 4.1).

Для кількісної оцінки ефективності впровадження чат-бота порівняно з традиційними методами запису, можна використовувати наступну формулу ефективності:

$$E = ((T_c + A_c + C_c) - (T_b + A_b + C_b)) / (T_c + A_c + C_c) \times 100\%$$

де T – середній час обробки одного запису (в хвилинах);

A – доступність сервісу записів (у відсотках, наприклад, 50% означає, що сервіс доступний з 8:00 до 20:00, а 100% – цілодобово);

C – витрати на обслуговування (у грошовому еквіваленті);

індекс $_c$ – для традиційного каналу (телефонні дзвінки, особисті візит, онлайн-форма);

індекс $_b$ – для чат-бота.

Ця формула дозволяє об'єктивно оцінити, наскільки впровадження чат-бота покращує ефективність процесу запису, демонструючи відсоткове зменшення сукупних "витрат" (часу, недоступності, фінансових витрат) на кожен запис. Високе позитивне значення "E" свідчитиме про значну вигоду від автоматизації.

Таблиця 4.1 – Таблиця порівняння чат-боту з традиційними методами запису

Критерій	Традиційні методи	Чат-бот
Час обробки запису	Залежить від доступності оператора; може бути довгим	Миттєве реагування, автоматичне внесення
Доступність	Обмежена робочим часом	Цілодобова (24/7)
Людський фактор	Високий ризик помилок	Мінімізований
Формат запису	Паперові журнали або неінтегровані таблиці	Централізована база даних
Управління розкладом	Трудомістке, часто вручну	Автоматизоване управління слотами
Зміни/скасування	Вимагають дзвінка або візиту	Автоматичне оновлення та сповіщення
Інтеграція з системами	Обмежена або відсутня	Можлива з CRM, календарями тощо
Аналітика	Відсутня або ручна	Автоматизована, детальна статистика
Зручність для клієнта	Часто незручна, потребує часу	Висока, запис у будь-який момент
Операційні витрати	Високі	Низькі

4.2 Оцінка зручності та швидкості взаємодії з клієнтами

Використання чат-бота для запису на СТО значно підвищує рівень зручності та оперативності взаємодії з клієнтами, що є критично важливим фактором у сучасному сервісному секторі. Завдяки інтуїтивно зрозумілому діалоговому інтерфейсу, користувачі без зайвих зусиль можуть орієнтуватися в процесі запису, що значно знижує так званий "бар'єр входу" для потенційних клієнтів. Застосування популярного месенджера Telegram як основної платформи для взаємодії додає додатковий рівень комфорту: користувачам не потрібно завантажувати, встановлювати нові додатки, створювати додаткові облікові записи або реєструватися на сторонніх платформах. Вони можуть взаємодіяти з СТО у звичному для них цифровому середовищі, де вони вже проводять значну частину свого часу. Це особливо актуально для активних користувачів смартфонів, які звикли до швидкої, зручної та ефективної комунікації без зайвих кроків.

Процес запису розбитий на логічні, покрокові етапи, кожен з яких супроводжується зрозумілими інструкціями та зручними інтерактивними кнопками для вибору послуг, що з'являються безпосередньо в чаті. Цей підхід мінімізує когнітивне навантаження на користувача, усуваючи потребу запам'ятовувати команди чи вводити довгі текстові запити, та виключає можливість помилки через неправильне введення інформації, що є поширеною проблемою при традиційному телефонному записі або заповненні складних онлайн-форм. В результаті, клієнт може всього за 30-60 секунд обрати потрібну послугу, визначити бажану дату та час, а також швидко підтвердити свій запис.

Особливу увагу приділено простоті інтерфейсу, що робить чат-бот доступним для людей будь-якого віку та з різним рівнем цифрової грамотності. Чітка покроковість елементів управління, використання знайомих іконок та наявність миттєвого зворотного зв’язку у реальному часі створюють безпечне та

комфортне середовище для взаємодії навіть для користувачів, які не мають великого досвіду роботи з цифровими технологіями або раніше не користувалися подібними сервісами. Бот відповідає на кожному кроці, підтверджуючи отримання інформації та вказуючи на наступний необхідний крок.

Ще однією значною перевагою є миттєве, автоматизоване підтвердження запису. Після завершення процедури бронювання користувач одразу отримує структуроване повідомлення з усіма ключовими деталями: назвою обраної послуги, бажаною датою та часом, номером автомобіля та контактною інформацією СТО. Це усуває будь-яку невизначеність, пов'язану з успішністю запису, і сприяє формуванню довіри до сервісу. Оскільки вся інформація надається у текстовому вигляді безпосередньо в чаті Telegram, її легко зберегти в історії чату, переслати іншим особам (наприклад, членам сім'ї) або переглянути пізніше без потреби повторного звернення до СТО. Це також зменшує навантаження на адміністративний персонал, оскільки немає потреби у ручному підтвердженні кожного запису.

До того ж, чат-бот забезпечує асинхронну комунікацію, що є суттєвою перевагою порівняно з традиційним телефонним дзвінком або особистим візитом. Клієнт може розпочати процес запису у будь-який зручний момент, навіть у неробочі години СТО, перерватися на інші справи та повернутися до діалогу пізніше, не втрачаючи вже введених даних. Така гнучкість особливо важлива для зайнятих людей, які цінують свій час і прагнуть максимально ефективного обслуговування, не будучи прив'язаними до графіку роботи СТО. Це дозволяє клієнтам взаємодіяти з сервісом тоді, коли це справді зручно для них, що підвищує їхню лояльність.

У результаті, впровадження чат-бота дозволяє не лише значно оптимізувати та автоматизувати процес запису на послуги СТО, знижуючи операційні витрати, а й суттєво покращити загальний клієнтський досвід, забезпечуючи простоту, швидкість, надійність та гнучкість взаємодії. Це створює сучасний та орієнтований на клієнта сервіс, що може стати конкурентною перевагою на ринку.

4.3 Потенціал масштабування та впровадження нових функцій

Розроблена архітектура системи для запису клієнтів на СТО має значний потенціал для масштабування та подальшого розвитку, що є критично важливим для довгострокової життєздатності будь-якого програмного продукту. Вона спроектована таким чином, що може бути легко розширена для обслуговування значно більшої кількості клієнтів та одночасних запитів без необхідності повного переписування коду. Модульна структура, закладена в основу системи, дозволяє додавати нові послуги, розширювати графік роботи, впроваджувати додаткові філії або навіть підтримувати кілька СТО під одним обліковим записом без суттєвих змін в основній логіці ConversationHandler та обробників. Це забезпечує високу гнучкість та адаптивність до мінливих потреб бізнесу. Крім того, наявність відкритого коду та використання популярних бібліотек значно спрощує інтеграцію з іншими системами управління СТО, бухгалтерським програмним забезпеченням або CRM-системами, дозволяючи створити єдиний інформаційний простір для керування клієнтами та записами.

Перспективними напрямками для впровадження нових функцій, які значно розширять можливості бота та покращать клієнтський досвід, є наступні.

Інтеграція елементів штучного інтелекту (AI) для використання технологій обробки природної мови (NLP) дозволить чат-боту не лише розпізнавати типові команди та заздалегідь визначені варіанти відповідей, а й аналізувати складні або неоднозначні запити клієнтів, формулюючи відповіді, що максимально відповідають їхнім потребам. Завдяки NLP бот зможе краще розуміти контекст діалогу, враховувати попередні запити користувача в поточній сесії для більш персоналізованої взаємодії, а також адаптувати відповіді до стилю спілкування конкретного клієнта, створюючи відчуття живої розмови. Це особливо актуально при обробці запитань про специфічні послуги, типи автомобілів, технічні несправності, які клієнт може описати неформальною мовою, або для надання

рекомендацій щодо сервісу. У перспективі можливе впровадження більш складних діалогових моделей з машинним навчанням, які збиратимуть досвід з історії спілкувань, аналізуватимуть часто задавані питання та успішні сценарії для подальшої оптимізації сценаріїв обслуговування та підвищення ефективності комунікації.

Інтеграція з Google Maps дозволить клієнтам безпосередньо з чат-бота отримати вказівки, як доїхати до СТО, відобразити найкоротший маршрут, а також показати поточну ситуацію на дорогах. Крім того, через Google Maps можна буде показувати години пік на дорогах, приблизну тривалість маршруту та навіть завантаження найближчих вулиць, що допоможе клієнтам спланувати свій візит. Для мережі СТО така інтеграція відкриває додаткові можливості, як-от підключення інформації про наявність вільних місць на парковці біля кожної філії, поточні рейтинги сервісу на картах або відгуки клієнтів з Google, що підвищить прозорість та привабливість сервісу.

Покращення функціоналу нагадувань про запис є критично важливим для зниження кількості неявок. Чат-бот може надсилати автоматичні повідомлення не лише напередодні візиту (за 24 години), а й за кілька годин до нього (наприклад, за 2-4 години), що є оптимальним для нагадування. Індивідуальні налаштування дозволять кожному клієнту обрати зручний час нагадування або навіть тип сповіщення (текст, беззвучне тощо). Додатково можна реалізувати функціонал «розумних» нагадувань: бот аналізує інтервали між минулими візитами, типи наданих послуг та навіть пробіг автомобіля (якщо ці дані будуть додані до профілю клієнта), і нагадує про планове обслуговування (наприклад, заміну масла, шин, перевірку гальмівної системи), враховуючи специфіку авто або минулі дії. Така інтеграція з внутрішньою CRM-системою СТО створить відчуття персонального підходу до кожного клієнта і значно підвищить якість сервісу та лояльність.

Реалізація системи оцінок та відгуків надасть цінний зворотний зв'язок. Після завершення візиту чат-бот може автоматично звернутися до клієнта з проханням оцінити надану послугу за шкалою від 1 до 5, а також залишити

текстовий коментар щодо свого досвіду. Зібрані дані можна зберігати в базі даних для подальшого детального аналізу – наприклад, для вивчення частоти негативних оцінок, виявлення слабких місць у сервісі (наприклад, тривале очікування, проблеми з комунікацією), оцінки ефективності окремих майстрів або визначення найбільш затребуваних послуг. Це дозволяє не тільки постійно покращувати якість обслуговування, а й вчасно реагувати на проблеми, перетворюючи негативний досвід на можливість для зростання.

Інтеграція з платіжними системами надасть клієнтам можливість здійснити попередню оплату послуг або внести аванс безпосередньо через чат- бот, що значно підвищує рівень довіри клієнта до СТО, оскільки це підтверджує серйозність намірів та гарантує бронювання. Крім того, це суттєво скорочує кількість скасованих бронювань та неявок, оскільки клієнт має фінансове зобов'язання. Застосування популярних платіжних платформ, таких як Portmone, LiqPay або Fondy, забезпечить швидкість, надійність та безпечність транзакцій, використовуючи вже перевірені та захищені інфраструктури. У майбутньому можливе впровадження повноцінної системи лояльності, наприклад, накопичувальних знижок, бонусних балів або кешбеку для постійних клієнтів при оплаті через бот, що стимулюватиме повторні звернення та зміцнить взаємодію з СТО.

Всі ці напрямки розширення функціоналу ґрунтуються на поточній модульній архітектурі, що робить їх реалізацію технічно здійсненною та економічно виправданою, забезпечуючи СТО потужний інструмент для подальшого розвитку та покращення сервісу.

4.4 Вплив автоматизації на лояльність клієнтів і навантаження персоналу

Впровадження автоматизованої системи запису через чат-бот має значний і багатогранний позитивний вплив як на лояльність клієнтів, так і на

ефективність роботи та навантаження персоналу СТО. З боку клієнтів, ключовими факторами, що формують позитивний досвід взаємодії зі СТО та підвищують їхню задоволеність послугою, є цілодобова доступність сервісу, що дозволяє записатися у будь-який зручний час доби, швидкість обробки запиту та зручність самого процесу. У сучасному світі, де час є цінним ресурсом, можливість швидко і без зусиль оформити запис без телефонних дзвінків чи очікування відповіді, є значною перевагою. Можливість самостійно керувати своїми записами – переглядати деталі, змінювати дату або час, а також скасовувати бронювання – дає клієнтам відчуття контролю та гнучкості, що є ключовим фактором у формуванні довгострокової лояльності до бренду. Крім того, високий рівень персоналізації, наприклад, потенційне збереження історії обслуговування автомобіля клієнта або надсилання індивідуальних нагадувань про планові візити (наприклад, про заміну масла за пробігом), сприяє формуванню більш глибокого та довготривалого зв’язку з клієнтом, перетворюючи його на постійного відвідувача.

Більш ефективне обслуговування, мінімізація часу очікування, а також чіткість та стандартизація комунікації, яку забезпечує бот, сприяють підвищенню загальної задоволеності сервісом. Автоматизація значно знижує ймовірність виникнення конфліктних ситуацій, які часто пов’язані з людським фактором: помилками в розкладі, випадковим дублюванням записів, втратою інформації про бронювання або непорозуміннями при передачі даних. Кожне підтвердження запису, яке миттєво отримує клієнт, є чітким документом, що усуває будь-яку невизначеність.

Для персоналу СТО автоматизація процесу запису суттєво зменшує адміністративне навантаження. Операторам та адміністраторам більше не потрібно витрачати значну частину свого робочого часу на рутинні, повторювані операції, такі як прийом численних телефонних дзвінків для запису, ручне заповнення графіків, координація змін у розкладі або багаторазове пояснення базової інформації про послуги та графік роботи. Це вивільняє їхні ресурси та час для виконання більш складних, критичних і важливих для бізнесу завдань,

які вимагають людської взаємодії та аналітичного мислення. Наприклад, персонал може зосередитися на безпосередній роботі з клієнтами на місці, надаючи консультації, вирішуючи нестандартні питання, контролюючи якість наданих послуг, ефективніше управляти внутрішніми ресурсами майстерні (наприклад, розподіл робіт між майстрами, замовлення запчастин) або більш глибоко аналізувати дані для планування розкладу та оптимізації процесів на основі отриманої аналітики.

Крім того, завдяки автоматичному збору та систематизації даних, керівництво СТО може отримувати цінну статистику та аналітичні звіти щодо завантаженості майстрів, часу простою, кількості пропущених записів, популярності окремих послуг або пікових годин. Ця інформація створює широкі можливості для подальшої оптимізації бізнес-процесів, більш ефективного розподілу ресурсів та прийняття обґрунтованих управлінських рішень, спрямованих на підвищення прибутковості та операційної ефективності.

Автоматизація також позитивно впливає на емоційний стан працівників, значно зменшуючи рівень рутини та кількість стресових ситуацій, пов’язаних із перевантаженням, багатозадачністю або непорозуміннями з клієнтами. Це, своєю чергою, сприяє покращенню загальної атмосфери в колективі, підвищенню мотивації працівників та, як наслідок, поліпшенню загальної якості обслуговування, оскільки персонал може приділяти більше уваги кожному клієнту, який звертається за допомогою до бота. Таким чином, чат-бот стає не лише інструментом автоматизації, а й важливим фактором у створенні позитивного середовища як для клієнтів, так і для співробітників СТО [19].

4.5 Пропозиції щодо подальшого вдосконалення чат-бота

Для забезпечення подальшого розвитку функціоналу чат-бота та підвищення його ефективності у роботі зі СТО, доцільно реалізувати низку стратегічних покращень, орієнтованих як на розширення можливостей для

клієнтів, так і на оптимізацію внутрішніх процесів підприємства. Ці вдосконалення допоможуть перетворити бота на повноцінний цифровий помічник.

Одним із найважливіших напрямків є глибша інтеграція чат-бота з внутрішніми системами СТО. Зокрема, йдеться про інтеграцію з CRM-системою для ведення повної історії взаємодії з кожним клієнтом, а також з обліком складських запасів. Це дозволить боту не тільки автоматично формувати замовлення-наряд після успішного запису, але й забезпечувати клієнтів актуальною інформацією про наявність необхідних запчастин для їхнього автомобіля або конкретної послуги. На основі історії обслуговування бот зможе налаштовувати персоналізовані рекомендації, пропонувати планові технічні огляди або необхідну заміну витратних матеріалів, значно підвищуючи рівень сервісу та лояльності.

Варто також впровадити систему автоматичних сповіщень для персоналу СТО. Це можуть бути як повідомлення у внутрішньому інтерфейсі управлінської системи, так і більш оперативні сповіщення через Telegram, наприклад, у спеціально створеному каналі для адміністраторів або майстрів. Такі сповіщення інформуватимуть працівників про нові записи, скасування, зміни в розкладі, а також про особливі запити або побажання клієнтів. Це дозволить команді СТО оперативніше реагувати на зміни, ефективніше розподіляти ресурси та оптимальніше організовувати робочий процес, мінімізуючи простої та максимізуючи завантаженість.

Не менш важливим є розширення функціоналу зворотного зв'язку. Чат-бот повинен надавати клієнтам можливість після завершення обслуговування автоматично залишити оцінку та розгорнутий коментар щодо якості послуг. Це стане цінним джерелом неструктурованих даних для подальшого аналізу задоволеності клієнтів, швидкого виявлення проблемних ділянок у сервісі, ідентифікації слабких сторін та підвищення загального рівня обслуговування. Можливість оперативного реагування на негативні відгуки може перетворити незадоволеного клієнта на лояльного.

Доцільним також є впровадження механізму динамічного ціноутворення, який враховуватиме різні фактори, такі як час доби (наприклад, знижки у непікові години), поточний рівень завантаженості СТО (спеціальні пропозиції для залучення клієнтів у періоди спаду) чи популярність конкретної послуги (збільшення ціни на послуги, що мають високий попит, та зниження на менш популярні). Крім того, через чат-бот можна ефективно запускати акційні пропозиції, надавати індивідуальні знижки постійним клієнтам або створювати промокоди, що сприятиме залученню нових клієнтів, стимулюванню повторних візитів та збільшенню середнього чека.

Для ефективного централізованого адміністрування системи критично важливо створити повноцінну адміністративну панель (веб-інтерфейс або окремий Telegram-бот для адміністраторів). Вона має забезпечувати зручне управління переліком послуг (додавання, редагування, видалення), гнучке налаштування розкладу роботи СТО та окремих майстрів, доступ до детальної аналітики записів (кількість, тип послуг, завантаженість), повної історії взаємодії з клієнтами та можливість оперативного налаштування параметрів бота (вітальні повідомлення, шаблони підтверджень) без потреби прямого технічного втручання у код. Це дозволить менеджменту СТО швидко адаптуватися до змін попиту, впроваджувати нові бізнес-стратегії та постійно покращувати якість сервісу, роблячи систему самодостатньою та керованою без залучення розробників для рутинних операцій.

Запропоновані вдосконалення дадуть змогу зробити чат-бот не просто засобом автоматизації запису, а й повноцінним, багатофункціональним інструментом цифрової трансформації бізнесу СТО, що покращить взаємодію з клієнтами та оптимізує внутрішні процеси.

4.6 Висновок до розділу 4

Проведений аналіз свідчить про високу ефективність розробленої системи автоматизації запису на СТО за допомогою чат-бота. У порівнянні з

традиційними методами, такими як телефонні дзвінки чи особисте відвідування, чат-бот забезпечує суттєве підвищення зручності, швидкості та точності взаємодії з клієнтами. Цілодобова доступність, мінімізація людського фактору, зменшення адміністративного навантаження та покращення клієнтського досвіду є ключовими перевагами запропонованого рішення.

Завдяки модульній архітектурі, платформа має значний потенціал до масштабування, інтеграції з іншими цифровими системами, а також подальшого розвитку через впровадження сучасних технологій, таких як NLP, інтеграція з картами, платіжними системами, CRM, і аналітичними інструментами.

Таким чином, впровадження чат-бота не лише вирішує актуальні проблеми традиційних методів запису, а й відкриває нові можливості для цифрової трансформації СТО, підвищуючи його конкурентоспроможність на ринку.

ВИСНОВКИ

У ході виконання даної бакалаврська кваліфікаційна робота успішно вирішила актуальне завдання автоматизації процесу запису на станції технічного обслуговування (СТО) за допомогою чат-ботів. Проведене дослідження та подальша практична реалізація підтвердили як доцільність, так і високу ефективність такого підходу в умовах сучасних вимог до швидкості та зручності клієнтського сервісу.

На початковому етапі були проаналізовані теоретичні засади функціонування СТО, що дозволило виявити ключові проблеми, притаманні традиційним методам запису клієнтів. До них належать значне операційне навантаження на персонал, підвищений ризик людських помилок, а також обмежена доступність та загальна незручність для клієнтів. Ці висновки чітко вказали на нагальну потребу у впровадженні інноваційних IT-рішень для оптимізації цих процесів. Далі, було досліджено різноманітні сучасні IT-рішення, що застосовуються для автоматизації бізнес-процесів, включаючи ERP- та CRM-системи, мобільні додатки та онлайн-форми. В результаті цього аналізу було чітко визначено, що саме чат-боти є найбільш перспективним інструментом для автоматизації взаємодії з клієнтами у звичних для них каналах комунікації, завдяки їхній доступності, інтерактивності та загальній ефективності.

Особливу увагу було приділено вивченню технологій чат-ботів: їх архітектури, доступних платформ та сфер застосування. Обґрунтований вибір платформи Telegram для практичної реалізації чат-бота базувався на її значній популярності в Україні, широкому та гнучкому функціоналу для розробки ботів, а також високому рівні безпеки та конфіденційності, що є критично важливим для захисту персональних даних клієнтів.

В рамках практичної частини роботи була розроблена та успішно реалізована архітектура Telegram-бота**,** який повноцінно автоматизує процес запису клієнтів на СТО. Детально описано процес створення діалогового інтерфейсу, принципи організації збереження та обробки даних клієнтів у

відповідній базі даних, а також розроблений алгоритм роботи чат-бота, що включає валідацію введених даних, безпосереднє бронювання послуг та автоматичне підтвердження запису. Після розробки було проведено всебічне тестування системи, яке дозволило виявити та усунути потенційні помилки, а також підтвердити коректність функціонування чат-бота у всіх типових сценаріях взаємодії з користувачами. Забезпечення цілодобової доступності та безперебійної роботи системи стало можливим завдяки її успішному впровадженню у хмарне середовище, що повністю відповідає сучасним вимогам клієнтів до сервісу 24/7.

На завершальному етапі була оцінена ефективність розробленої системи шляхом проведення порівняльного аналізу з традиційними методами запису. Результати цього аналізу однозначно показали, що впровадження чат-бота значно підвищує зручність та швидкість взаємодії з клієнтами, істотно мінімізує навантаження на адміністративний персонал СТО, сприяє суттєвому зменшенню кількості помилок та, як наслідок, покращує лояльність клієнтів. Надалі були сформульовані конкретні пропозиції щодо подальшого вдосконалення чат-бота, що включають розширення його функціоналу, інтеграцію з додатковими сервісами та поліпшення можливостей обробки природної мови. Ці пропозиції відкривають широкі перспективи для подальшого масштабування системи та її адаптації до динамічних потреб ринку.

Таким чином, розроблений та впроваджений Telegram-бот є ефективним, інноваційним та практично цінним інструментом автоматизації. Він дозволяє СТО не тільки значно підвищити якість наданих послуг, але й суттєво оптимізувати внутрішні операційні процеси, що, в свою чергу, зміцнює їхні конкурентні позиції на ринку. Ця робота має вагоме практичне значення і може слугувати основою для подальшого розвитку та масштабування аналогічних автоматизованих систем у різних сферах обслуговування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Використання чат-ботів як альтернативи стандартним сервісам // DUT. – URL: https://con.dut.edu.ua/index.php/communication/article/view/2844 (дата звернення 02.05.2025)

2. Python documentation . – URL: https://www.python.org/ (дата звернення 18.04.2025)

3. SQLite documentation . – URL: https://www.sqlite.org/docs.html (дата звернення 15.04.2025)

4. Telegram Bot API. Офіційна документація . – URL: https://core.telegram.org/bots/api (дата звернення 18.04.2025)

5. Ковальов В. В. Сучасні електронні сервіси як засіб комунікації експертних установ із замовниками експертних послуг // Інформаційне право. – 2021. – № 2(34). – С. 57–65. – DOI: http://dx.doi.org/10.37025/1992-4437/2020-34-2-57 (дата звернення 03.05.2025)

6. Бісікало О. В., Юрчук М. С. Класифікація чат-ботів для електронної комерції // Сучасна молодь в світі інформаційних технологій: матеріали конф. – 2022. – 164 с. (дата звернення 23.03.2025)

7. Вступ до асинхронного програмування на Python . – URL: https://devzone.org.ua/post/vstup-do-asinhronnogo-programuvannya-napython (дата звернення 18.04.2025)

8. Основні відомості про бази даних . – URL: https://support.microsoft.com/uk-ua/office/основні-відомості-про-бази-даних-a849ac16-07c7-4a31-9948-3c8c94a7c204 (дата звернення 23.03.2025)

9. Огляд чат-ботів і автоматизації для бізнесу . – URL: https://megasite.ua/blog/chto-takoe-chatbot-i-zachem-on-nuzhen-biznesu(дата звернення 22.03.2025)

10. Інструменти для створення Telegram-ботів . – URL: https://1b.app/ua/crm-for-telegram-bot/ (дата звернення 16.04.2025)

11. Чат-боти для бізнесу: як автоматизація змінює взаємодію . – URL: https://jobitt.com/uk/blog/chatbots-for-business-how-automation-is-changing-customer-interactions (дата звернення 01.06.2025)

12. Створення чат-бота у Telegram для підтримки . – URL: https://share.google/XMVU7nY7zgdvs683r (дата звернення 17.04.2025)

13. Khosravi H. Mastering Bots: Develop, Deploy, and Monetize Your Bots. – 2022. (дата звернення 05.04.2025)

14. Що таке чат-бот? . – URL: https://mc.today/uk/shho-take-chat-boti/ (дата звернення 23.03.2025)

15. Як чат-бот допомагає підвищити продажі для СТО . – URL: https://freelancehunt.com/blog/iak-za-dopomoghoiu-chat-botu-pidvishchiti-prodazhi-dlia-sto/ (дата звернення 22.03.2025)

16. Чат-боти для обслуговування клієнтів: приклади та переваги . – URL: https://claspo.io/ua/blog/chatbots-for-customer-service-examples-benefits-and-uses/ (дата звернення 23.03.2025)

17. Основні відмінності між Viber і Telegram . – URL: https://smsclub.mobi/.../viber-vs-telegram/ (дата звернення 23.03.2025)

18. Lytvynenko O. V., Nafieiev R. K. Використання чат-ботів як альтернативи сервісам . – URL: https://con.dut.edu.ua/index.php/communication/article/view/2844 (дата звернення 22.03.2025)

19. Casadei F. et al. Chatbots for Robotic Process Automation: Investigating Perceived Trust and User Satisfaction. – 2023. (дата звернення 27.03.2025)

20. Путівник мовою Python . – URL: https://pythonguide.rozh2sch.org.ua/ (дата звернення 16.04.2025)

ДОДАТКИ

Додаток А

(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

АВТОМАТИЗАЦІЯ ПРОЦЕСУ ЗАПИСУ КЛІЄНТІВ НА СТАНЦІЇ ТЕХНІЧНОГО ОБСЛУГОВУВАННЯ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ ЧАТ-БОТІВ

Алгоритм модуля роботи системи

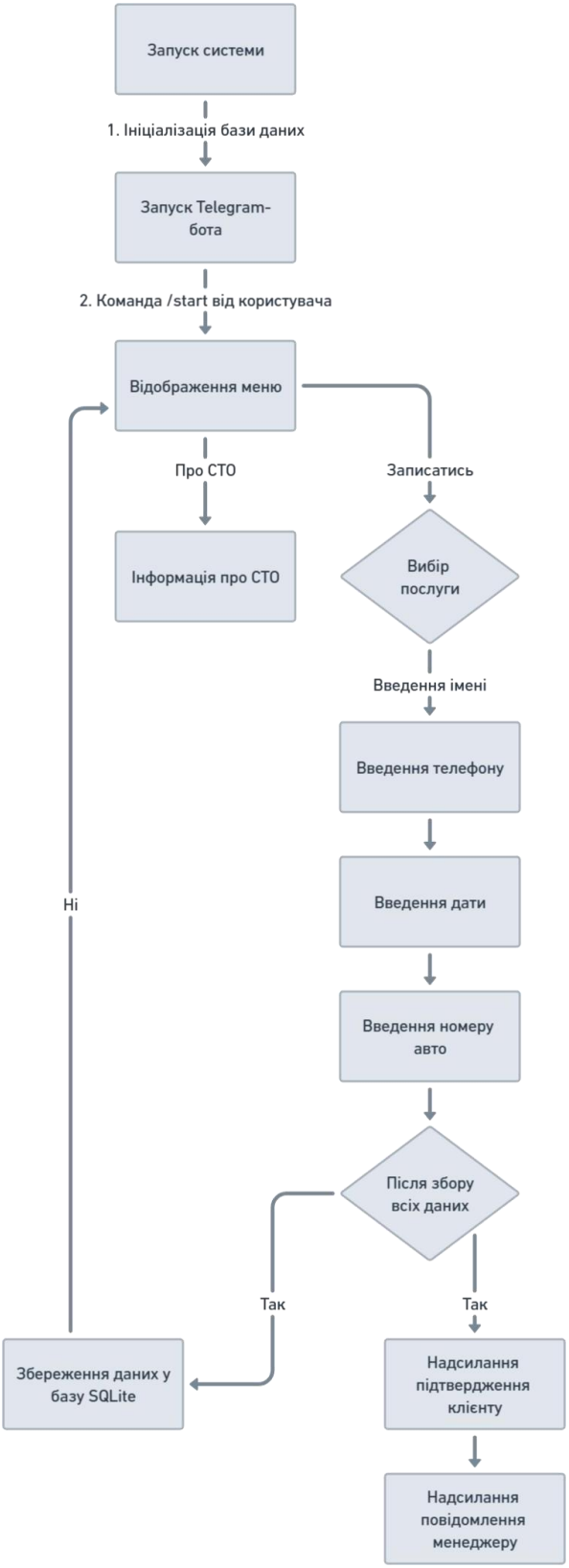

Рисунок А.1 – Алгоритм модуля роботи системи автоматизованого запису клієнта

Алгоритм модуля тестування системи

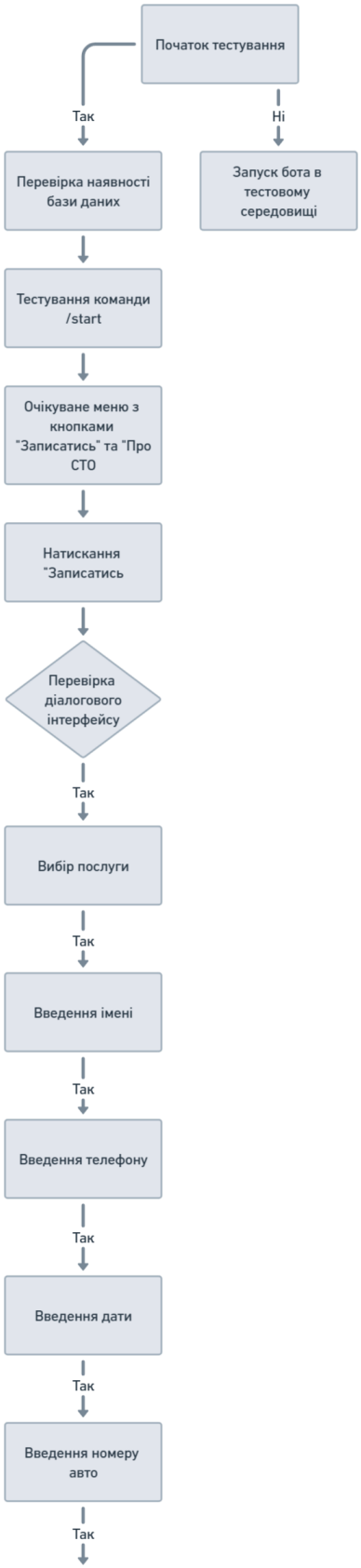

Рисунок А.2 – Алгоритм модуля тестування системи автоматизованого запису клієнтів

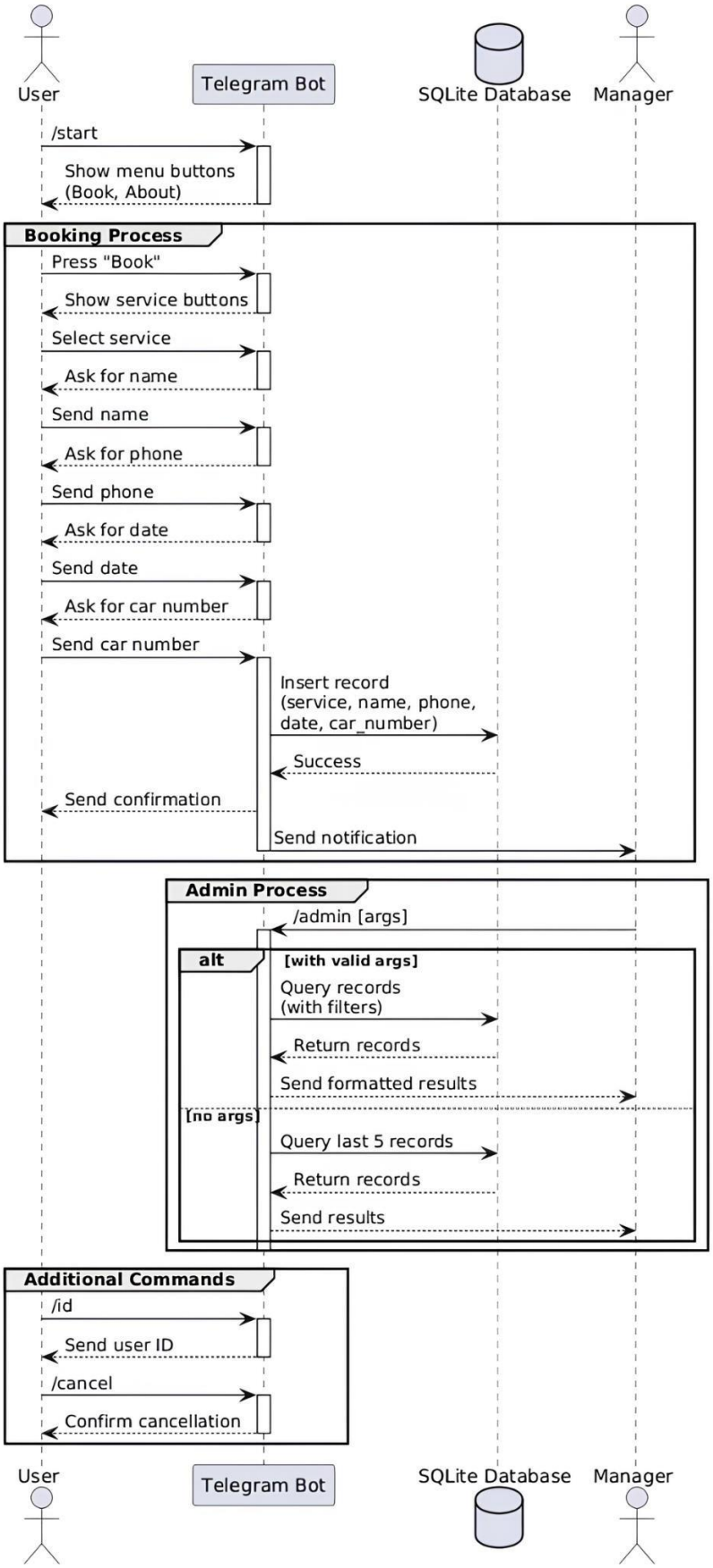

Рисунок А.3 – UML діаграма, що показує взаємодію клієнта та адміністратора з системою

Додаток Б

(обов’язковий)

Лістинг основних функцій програми

```
import sqlite3
from datetime import datetime
from telegram import (
    Update,
    InlineKeyboardMarkup,
    InlineKeyboardButton
)
from telegram.ext import (
    ApplicationBuilder,
    CommandHandler,
    MessageHandler,
    CallbackQueryHandler,
    ConversationHandler,
    ContextTypes,
    filters
)

# Стан розмови
SERVICE, NAME, PHONE, DATE, CAR_NUMBER = range(5)

# Заміни на свій Telegram ID
MANAGER_ID = 799798585

# Ініціалізація бази даних
def init_db():
    conn = sqlite3.connect("service_records.db")
```

```
    c = conn.cursor()
    c.execute("""
        CREATE TABLE IF NOT EXISTS records (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            service TEXT NOT NULL,
            name TEXT NOT NULL,
            phone TEXT NOT NULL,
            date TEXT NOT NULL,
            car_number TEXT NOT NULL,
            timestamp DATETIME DEFAULT CURRENT_TIMESTAMP
        )
    """)
    conn.commit()
    conn.close()

# Додавання запису
def insert_record(service, name, phone, date, car_number):
    conn = sqlite3.connect("service_records.db")
    c = conn.cursor()
    c.execute("""
        INSERT INTO records (service, name, phone, date, car_number)
        VALUES (?, ?, ?, ?, ?)
    """, (service, name, phone, date, car_number))
    conn.commit()
    conn.close()

# Команда /start
async def start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    keyboard = [
        [InlineKeyboardButton("📝 Записатись", callback_data="start_booking")],
```

```
        [InlineKeyboardButton("ℹ️ Про СТО", callback_data="about")]
    ]
    await update.message.reply_text("Привіт! Оберіть опцію:", reply_markup=InlineKeyboardMarkup(keyboard))

# Обробка меню
async def handle_menu_callback(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    if query.data == "start_booking":
        keyboard = [
            [InlineKeyboardButton("🔧 Заміна масла", callback_data="service_oil")],
            [InlineKeyboardButton("🛠 Діагностика", callback_data="service_diagnostics")],
            [InlineKeyboardButton("🛞 Шиномонтаж", callback_data="service_tires")],
            [InlineKeyboardButton("🔋 Заміна акумулятора", callback_data="service_battery")]
        ]
        await query.message.reply_text("Оберіть послугу:", reply_markup=InlineKeyboardMarkup(keyboard))
        return SERVICE

    elif query.data == "about":
        await query.message.reply_text("🔧 Ми — СТО 'АвтоСервіс'. Працюємо щодня з 9:00 до 18:00.\n📞 Телефон: +380991112233")
        return ConversationHandler.END
```

```
# Вибір послуги
async def handle_service_choice(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    service_map = {
        "service_oil": "Заміна масла",
        "service_diagnostics": "Діагностика",
        "service_tires": "Шиномонтаж",
        "service_battery": "Заміна акумулятора"
    }

    service = service_map.get(query.data, "Невідома послуга")
    context.user_data["service"] = service
    await query.message.reply_text("Як вас звати?")
    return NAME

# Збір даних користувача
async def get_name(update: Update, context: ContextTypes.DEFAULT_TYPE):
    context.user_data['name'] = update.message.text
    await update.message.reply_text("Ваш номер телефону?")
    return PHONE

async def get_phone(update: Update, context: ContextTypes.DEFAULT_TYPE):
    context.user_data['phone'] = update.message.text
    await update.message.reply_text("На яку дату бажаєте записатись? (напр. 25.06.2025)")
    return DATE
```

```
async def get_date(update: Update, context: ContextTypes.DEFAULT_TYPE):
    context.user_data['date'] = update.message.text
    await update.message.reply_text("Введіть номер вашого авто (напр. AA1234BC):")
    return CAR_NUMBER

async def get_car_number(update: Update, context: ContextTypes.DEFAULT_TYPE):
    context.user_data['car_number'] = update.message.text

    service = context.user_data['service']
    name = context.user_data['name']
    phone = context.user_data['phone']
    date = context.user_data['date']
    car_number = context.user_data['car_number']

    # Збереження в базу
    insert_record(service, name, phone, date, car_number)

    # Повідомлення клієнту
    msg = (
        f"✅ Запис підтверджено!\n\n"
        f"🛠 Послуга: {service}\n"
        f"👤 Ім'я: {name}\n"
        f"📞 Телефон: {phone}\n"
        f"📅 Дата: {date}\n"
        f"🚗 Авто: {car_number}"
    )
    await update.message.reply_text(msg)
```

```
    # Повідомлення менеджеру
    try:
        await context.bot.send_message(chat_id=MANAGER_ID, text=msg)
    except Exception as e:
        print(f"⚠ Помилка надсилання менеджеру: {e}")

    return ConversationHandler.END

# Команда /admin
async def show_latest_records(update: Update, context: ContextTypes.DEFAULT_TYPE):
    user_id = update.effective_user.id
    if user_id != MANAGER_ID:
        await update.message.reply_text("⛔ Ви не маєте доступу до цієї команди.")
        return

    args = context.args
    conn = sqlite3.connect("service_records.db")
    c = conn.cursor()

    # /admin 10
    if len(args) == 1 and args[0].isdigit():
        limit = int(args[0])
        c.execute("""
            SELECT service, name, phone, date, car_number, timestamp
            FROM records
            ORDER BY timestamp DESC
            LIMIT ?
```

```
        """, (limit,))
        rows = c.fetchall()
        title = f"📋 Останні {limit} записів:\n\n"

    # /admin 25.06.2025
    elif len(args) == 1:
        try:
            datetime.strptime(args[0], "%d.%m.%Y")
            search_date = args[0]
            c.execute("""
                SELECT service, name, phone, date, car_number, timestamp
                FROM records
                WHERE date = ?
                ORDER BY timestamp DESC
            """, (search_date,))
            rows = c.fetchall()
            title = f"📆 Записи за {search_date}:\n\n"
        except ValueError:
            await update.message.reply_text("❌ Невірний формат дати. Приклад: /admin 25.06.2025")
            conn.close()
            return

    # /admin
    else:
        limit = 5
        c.execute("""
            SELECT service, name, phone, date, car_number, timestamp
            FROM records
```

```
            ORDER BY timestamp DESC
            LIMIT ?
        """, (limit,))
        rows = c.fetchall()
        title = "📋 Останні 5 записів:\n\n"

    conn.close()

    if not rows:
        await update.message.reply_text("📭 Немає записів.")
        return

    text = title
    for i, row in enumerate(rows, start=1):
        service, name, phone, date_str, car_number, ts = row
        text += (
            f"🔹 {i}) {service}\n"
            f"👤 {name} | 📞 {phone}\n"
            f"📅 {date_str} | 🚗 {car_number}\n"
            f"🕒 {ts}\n\n"
        )

    await update.message.reply_text(text)

# Команда /cancel
async def cancel(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await update.message.reply_text("Операція скасована.")
    return ConversationHandler.END
```

```
# Команда /id (допоміжна)
async def get_user_id(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await update.message.reply_text(f"Ваш Telegram ID: {update.effective_user.id}")

# Запуск
def main():
    init_db()

    app = ApplicationBuilder().token("7748000508:AAFut3e9_2_bz-IrCG9ZNlWwAaF35iuBNOk").build()

    conv_handler = ConversationHandler(
        entry_points=[
            CommandHandler("start", start),
            CallbackQueryHandler(handle_menu_callback)
        ],
        states={
            SERVICE: [CallbackQueryHandler(handle_service_choice)],
            NAME: [MessageHandler(filters.TEXT & ~filters.COMMAND, get_name)],
            PHONE: [MessageHandler(filters.TEXT & ~filters.COMMAND, get_phone)],
            DATE: [MessageHandler(filters.TEXT & ~filters.COMMAND, get_date)],
            CAR_NUMBER: [MessageHandler(filters.TEXT & ~filters.COMMAND, get_car_number)],
        },
        fallbacks=[CommandHandler("cancel", cancel)]
    )

    app.add_handler(conv_handler)
    app.add_handler(CommandHandler("admin", show_latest_records))
```

```
    app.add_handler(CommandHandler("id", get_user_id))

    print("✅ Бот працює. Натисніть Ctrl+C для зупинки.")
    app.run_polling()

if __name__ == "__main__":
    main()
```

86

Додаток В
(обов'язковий)
Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

Назва роботи: Автоматизація процесу запису клієнтів на станції технічного обслуговування з використанням технологій чат-ботів

Тип роботи: _____ бакалаврська кваліфікаційна робота _____
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ _____ АІІТ, ФІІТА, 1АКІТ-21б _____
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 0 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

☑ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Олег БІСІКАЛО, зав. кафедри АІІТ _____ (підпис)
(прізвище, ініціали, посада)

Любов ЯНЧУК, секретар кафедри АІІТ _____ (підпис)
(прізвище, ініціали, посада)

Особа, відповідальна за перевірку _____ Костянтин ОВЧИННИКОВ
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Софина О.Ю., к.т.н. доц. кафедри
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Жовнір В.О.
(підпис) (прізвище, ініціали)