Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

БАКАЛАВРСЬКА ДИПЛОМНА РОБОТА

на тему:

«Розробка захищеного Telegram– бота оброблення замовлень та резервувань»

Виконав: студент 4 курсу,
групи 2КІТС–20б
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем

Молчан Є. В.

Керівник: доцент каф. МБІС
Грицак А. В.

«6» червня 2024 р.

Рецензент: к.т.н., доцент каф. ОТ
Городецька О. С.
«6» червня 2024 р.

Допущено до захисту
Голова секції УБ, кафедра МБІС
д.т.н., проф. Яремчук Ю. Є.
“6” червня 2024 р.

Вінниця ВНТУ – 2024

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Ступінь – бакалавр
Спеціальність 125 – «Кібербезпека»

ЗАТВЕРДЖУЮ
Голова секції УБ кафедри МБІС
_____ д.т.н., проф. Яремчук Ю. Є.
«22» березня 2024р.

З А В Д А Н Н Я
на бакалаврську дипломну роботу студенту

Молчану Єгору Віталійовичу
(прізвище та ініціали)

1. Тема роботи: «Розробка захищеного Telegram-бота оброблення замовлень та резервувань»
2. Керівник роботи: к.т.н., доц. каф. МБІС Грицак А. В.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ректора закладу вищої освіти від «22» березня 2024 року № 80

2. Строк подання студентом роботи за тиждень до захисту
3. Вихідні дані до роботи:
Метою роботи є розробка захищеного Telegram-бота для автоматизації процесів оброблення замовлень та резервувань з використанням механізмів шифрування даних та аутентифікації користувачів для забезпечення безпечного обміну інформацією та високого рівня користувацького сервісу.
4. Зміст текстової частини:
У першому розділі дипломної роботи проводиться аналіз аналіз існуючих рішень для обробки замовлень та резервувань, проводиться аналіз використання платформи Telegram, їх переваг та недоліків, досліджуються відомі загрози та існуючі програмні рішення в даній сфері. У другому розділі обґрунтовується розроблення підходу захищеного Telegram-бота, описується його взаємодія з базою даних, проектуються компоненти архітектури бота. Третій розділ присвячений безпосередньо програмній реалізації захищеного Telegram-бота, обґрунтуванню вибору мови програмування та середовища розробки, а також інструкції користувачу. У висновках підсумовуються результати виконаної роботи.
5. Перелік графічного матеріалу:
У першому розділі бакалаврської дипломної роботи наявно 6 рисунків, у другому розділі – 6 рисунків, у третьому розділі – 12 рисунків.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	Грицак А. В., к.т.н., доц. каф. МБІС		
Розділ II	Грицак А. В., к.т.н., доц. каф. МБІС		
Розділ III	Грицак А. В., к.т.н., доц. каф. МБІС		

7. Дата видачі завдання «22» березня 2024р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів бакалаврської дипломної роботи		Примітка
		початок	закінчення	
1	Визначення напрямку бакалаврської роботи, формулювання теми	23.03.2024	25.03.2024	
2	Аналіз предметної області обраної теми	26.03.2024	01.04.2024	
3	Розробка алгоритму роботи	02.04.2024	16.04.2024	
4	Написання бакалаврської роботи на основі розробленої теми	17.04.2024	05.05.2024	
5	Передзахист бакалаврської дипломної роботи	06.05.2024	24.05.2024	
6	Виправлення, уточнення, корегування бакалаврської дипломної роботи	25.05.2024	05.06.2024	
7	Захист бакалаврської дипломної роботи	12.06.2024	13.06.2024	

Студент _____ Молчан Є. В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Грицак А. В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська дипломна робота складається із 69 сторінок тексту формату А4, включаючи 2 рисунка, 5 таблиць, посилання на 35 літературних джерел та 4 додатки.

Дипломна робота присвячена розробці захищеного Telegram–бота для обробки замовлень та резервувань. У сучасному світі значну увагу приділяють автоматизації комунікацій між бізнесом та клієнтами, а також забезпеченню високого рівня безпеки передачі даних. Telegram, як платформа з високим ступенем шифрування та конфіденційності, надає відмінні можливості для створення сервісних ботів.

Метою роботи є створення бота, який дозволить користувачам ефективно взаємодіяти зі службою підтримки, робити замовлення або бронювати товари та послуги безпосередньо через інтерфейс Telegram. Основні завдання включають аналіз вимог до системи, проектування архітектури бота, розробку модулів безпеки для захисту даних користувачів та інтеграцію з зовнішніми системами бронювання та обробки замовлень.

Для досягнення цієї мети використовувалась методологія агільної розробки, що дозволило ефективно адаптуватися до змін у вимогах та швидко впроваджувати нові функціональні можливості. Було проведено аналіз безпеки, на основі якого розроблені та імплементовані механізми шифрування даних та аутентифікації користувачів. Також було реалізовано інтерфейс для зручного управління замовленнями та резервуваннями, інтегрований з популярними платформами електронної комерції.

Результатом роботи став повнофункціональний Telegram–бот, що забезпечує безпечний обмін даними та високий рівень користувацького сервісу. Бот демонструє значні переваги в швидкості та зручності обслуговування замовлень та резервувань, а також високу стійкість до зовнішніх загроз та атак.

Ключові слова: telegram–бот, обробка замовлень, резервування, захист даних, шифрування, аутентифікація, безпека інформації, цифрове обслуговування, користувацький сервіс, стійкість до атак.

ABSTRACT

The bachelor's thesis consists of 69 pages of A4 text, including 2 figures, 5 tables, references to 35 literary sources and 4 appendices.

Thesis is dedicated to the development of a secure Telegram bot for processing orders and reservations. In today's world, significant attention is paid to automating communications between businesses and customers, as well as ensuring a high level of data transmission security. Telegram, as a platform with a high degree of encryption and confidentiality, provides excellent opportunities for creating service bots.

The goal of this thesis is to create a bot that allows users to effectively interact with customer service, place orders, or reserve products and services directly through the Telegram interface. The main tasks include analyzing system requirements, designing the bot's architecture, developing security modules to protect user data, and integrating with external booking and order processing systems.

To achieve this goal, agile development methodology was used, which allowed for effective adaptation to changes in requirements and the rapid implementation of new features. A security analysis was conducted, based on which data encryption mechanisms and user authentication were developed and implemented. An interface was also developed for convenient management of orders and reservations, integrated with popular e–commerce platforms.

The result is a fully functional Telegram bot that provides secure data exchange and a high level of user service. The bot demonstrates significant advantages in the speed and convenience of order and reservation processing, as well as high resistance to external threats and attacks.

Keywords: telegram bot, order processing, reservations, data protection, encryption, authentication, information security, digital service, user service, attack resistance.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі месенджери стали невід'ємною частиною повсякденного життя людей та активно використовуються для спілкування, обміну інформацією та навіть ведення бізнесу. Одним із популярних месенджерів є Telegram, який відрізняється високим рівнем захисту даних та зручним функціоналом для розробки власних ботів. Однак, при розробці Telegram–ботів для обробки замовлень та резервувань виникають певні виклики щодо забезпечення безпеки конфіденційних даних користувачів та захисту від різноманітних загроз.

Обробка замовлень та резервувань через месенджери стає все більш поширеною практикою, оскільки дозволяє компаніям надавати зручний та швидкий сервіс для своїх клієнтів. Проте, передача конфіденційної інформації, такої як платіжні дані, персональні дані клієнтів або деталі замовлень, вимагає належного рівня захисту для запобігання витоку або неправомірного доступу до цих даних.

Розробка захищеного Telegram–бота для обробки замовлень та резервувань є актуальним завданням, адже воно дозволяє поєднати зручність використання месенджера з високим рівнем безпеки та захисту конфіденційних даних користувачів. Такий бот може забезпечити надійну обробку замовлень, резервувань та платежів, захистити персональні дані клієнтів від несанкціонованого доступу та запобігти можливим загрозам, таким як хакерські атаки або витік інформації.

Мета роботи полягає у розробці захищеного Telegram–бота для обробки замовлень та резервувань, який забезпечить високий рівень безпеки та захисту конфіденційних даних користувачів.

Було поставлено наступні завдання для досягнення вище згаданої мети роботи:

– проаналізувати існуючі рішення для обробки замовлень та резервувань у месенджерах, їх переваги та недоліки.

– спроектувати та обґрунтувати архітектуру захищеного Telegram–бота з урахуванням вимог безпеки.

– спроєктувати базу даних для зберігання замовлень, резервувань та персональних даних користувачів.

– впровадити заходи безпеки та захисту даних у розроблений бот.

– реалізувати програмну частину Telegram–бота з використанням обраних технологій та інструментів розробки.

– підготувати інструкцію для користувачів бота.

Об'єктом дослідження є процес обробки замовлень та резервувань у месенджері Telegram.

Новизна роботи полягає у розробці унікального підходу до створення захищеного Telegram-бота для обробки замовлень та резервувань, який поєднує в собі зручність використання месенджера з високим рівнем безпеки та захисту конфіденційних даних користувачів.

Предметом дослідження є методи та засоби розробки захищеного Telegram–бота для обробки замовлень та резервувань з високим рівнем безпеки та захисту конфіденційних даних користувачів.

Практична цінність роботи полягає у можливості використання розробленого захищеного Telegram–бота компаніями різних сфер діяльності для зручної та безпечної обробки замовлень і резервувань від клієнтів з дотриманням високих стандартів захисту конфіденційних даних.

РОЗДІЛ 1. АНАЛІЗ ТЕОРЕТИЧНИХ ОСНОВ РОЗРОБКИ TELEGRAM–БОТІВ ДЛЯ ОБРОБКИ ЗАМОВЛЕНЬ ТА РЕЗЕРВУВАНЬ У МЕСЕНДЖЕРАХ

1.1 Аналіз особливостей використання платформи Telegram для обробки замовлень та резервувань

Telegram, хмарна платформа обміну миттєвими повідомленнями, була запущена у 2013 році братами Миколою та Павлом Дуровими. Вона виникла як альтернатива домінуючим комунікаційним сервісам, наголошуючи на швидкості та безпеці. На відміну від багатьох своїх попередників, Telegram пропонує наскрізну зашифровану передачу повідомлень і децентралізовану структуру завдяки унікальному використанню протоколу MTProto [1].

Месенджер Telegram став одним із найпопулярніших інструментів для спілкування та обміну інформацією в сучасному цифровому світі. Завдяки своїй швидкості, зручності та високому рівню безпеки, Telegram перетворився на привабливу платформу не лише для особистого спілкування, але й для ведення бізнесу та надання різноманітних послуг. Зокрема, використання Telegram для обробки замовлень та резервувань стає все більш поширеною практикою серед компаній різних галузей.

Важливою перевагою використання Telegram для обробки замовлень та резервувань є високий рівень безпеки та захисту конфіденційних даних, який забезпечується месенджером. Telegram використовує потужне end–to–end шифрування, що гарантує захист передачі даних від стороннього доступу та підслуховування [2]. Крім того, месенджер дотримується політики конфіденційності, яка забороняє доступ до особистих даних користувачів без їхньої згоди.

Ще однією перевагою платформи Telegram є її кросплатформеність. Месенджер доступний на різних операційних системах, включаючи Android, iOS, Windows та веб–версію, що забезпечує зручність для користувачів та можливість обробляти

замовлення та резервування з різних пристроїв. Крім того, Telegram підтримує різні типи файлів та мультимедіа, що дозволяє компаніям надавати детальну інформацію про свої продукти чи послуги, а клієнтам – надсилати супровідні документи або фотографії під час оформлення замовлень (рис.1.1) [3].

Рисунок 1.1 – Вид програмного застосунку Telegram на ОС Android та iOS

Однією з ключових особливостей платформи Telegram, яка робить її привабливою для обробки замовлень та резервувань, є можливість створення власних ботів. Боти – це спеціальні програми, які можуть автоматизувати певні процеси, надавати інформацію та взаємодіяти з користувачами у зручному інтерфейсі месенджера. Завдяки ботам компанії можуть надавати своїм клієнтам зручний та швидкий сервіс для оформлення замовлень, здійснення резервувань, отримання інформації про продукти чи послуги та навіть проведення оплати [4].

Однак, незважаючи на свої розширені можливості, екосистема розробки ботів для Telegram має кілька недоліків. Одним з таких обмежень є відсутність повних офіційних бібліотек для певних мов програмування, що може стати на заваді розробникам–новачкам у цьому середовищі. Крім того, документація, хоча й обширна, іноді може бути нечіткою щодо певних розширених функціональних можливостей, що призводить до стрімкої кривої навчання.

Для порівняння, інші платформи, такі як Slack або Facebook Messenger, також підтримують розробку ботів, але суттєво відрізняються за своїм підходом. Slack зосереджується на інтеграції в професійне середовище, пропонуючи широкі функціональні можливості для спілкування на робочому місці, які не так яскраво виражені в Telegram. Боти для Facebook Messenger глибоко інтегровані в екосистему Facebook, надаючи доступ до величезної бази користувачів, але ціною меншої гнучкості щодо конфіденційності даних (табл. 1.1) [5].

Таблиця 1.1 – Порівняння відомих аналогів, що схожі на Telegram

Платформа	Telegram	Slack	Facebook Messenger
Фокус	Конфіденційність та безпека	Професійне середовище	Інтеграція в Facebook екосистему
Користувачі	Понад 900 млн активних користувачів	Корпоративні клієнти	Величезна база користувачів Facebook
Шифрування	Опціональне наскрізне шифрування	Немає даних	Немає повного наскрізного шифрування
Приватність	Не ділиться даними користувачів	Немає даних	Збирає дані користувачів
Можливості ботів	Розширені можливості через API	Зосереджено на професійних інструментах	Обмежені можливості
Відкритий код	Закритий код	Закритий код	Закритий код

З точки зору конфіденційності та безпеки даних користувачів, Telegram виділяється завдяки опціональному наскрізному шифруванню та обіцянці не ділитися даними з третіми сторонами, що є значною перевагою над Facebook Messenger. Однак шифрування не ввімкнене за замовчуванням, на відміну від таких сервісів, як Signal, який також забезпечує прозорість завдяки відкритому коду на стороні клієнта [6].

Набір функцій платформи продовжував розширюватися з появою голосових дзвінків у 2017 році та відеодзвінків у 2020 році, збагачуючи взаємодію користувачів завдяки більш особистим та цікавим формам спілкування. Станом на березень 2024 року Telegram налічує понад 900 мільйонів активних користувачів щомісяця і випередив WhatsApp [7]. На цьому еволюція платформи не зупинилася: вона постійно впроваджувала нові функції, такі як реакції на повідомлення, розширені можливості ботів та анімовані емодзі для преміум–користувачів. Ці постійні інновації, узгоджені з потребами користувачів і технологічним прогресом, стали ключовим фактором зростання і незмінної популярності Telegram.

Незважаючи на те, що Telegram надає надійні засоби безпеки та універсальні інструменти комунікації, він стикається з жорсткою конкуренцією з боку інших платформ, які надають пріоритет іншим аспектам, таким як інтеграція в певні цифрові екосистеми або покращення користувацького інтерфейсу. Його основною відмінною рисою з самого початку була надзвичайна увага до конфіденційності та безпеки обміну повідомленнями. Використання шифрування кінця–в–кінець (end–to–end encryption) у "секретних чатах" дозволяє користувачам відчувати себе в безпеці, знаючи, що їхні повідомлення можуть прочитати тільки вони та адресат.

Telegram виділяється на тлі інших додатків для обміну повідомленнями завдяки поєднанню безпеки, швидкості та універсальності. Розроблений як хмарний сервіс обміну повідомленнями, він забезпечує безперебійну синхронізацію між усіма пристроями користувача. Платформа підтримує текстове, голосове та відеозв'язок, а також групові чати, які можуть включати до 200 000 учасників в одній групі, що робить її потужним інструментом для широкомасштабного спілкування [8].

Отримання та відправка повідомлень – базова функція будь–якого бота — це взаємодія з користувачем через повідомлення. Telegram API дозволяє ботам не тільки отримувати текстові повідомлення від користувачів, але й відправляти їм текст, зображення, відео, аудіофайли, документи та навіть стікери. Також можливо відправляти повідомлення з форматуванням, використовуючи Markdown або HTML, що робить текст більш зручним та інформативним для користувача.

Розробка Telegram–ботів потребує від розробника не лише знання основ програмування, але й розуміння специфіки взаємодії з API месенджера. Важливим аспектом є також вміння гарантувати безпеку даних, особливо коли мова йде про обробку персональної інформації та замовлень. У цьому контексті, Telegram надає розробникам потужну платформу, що дозволяє створювати безпечні, швидкі та надійні боти для широкого спектру застосувань (рис.1.2) [9].

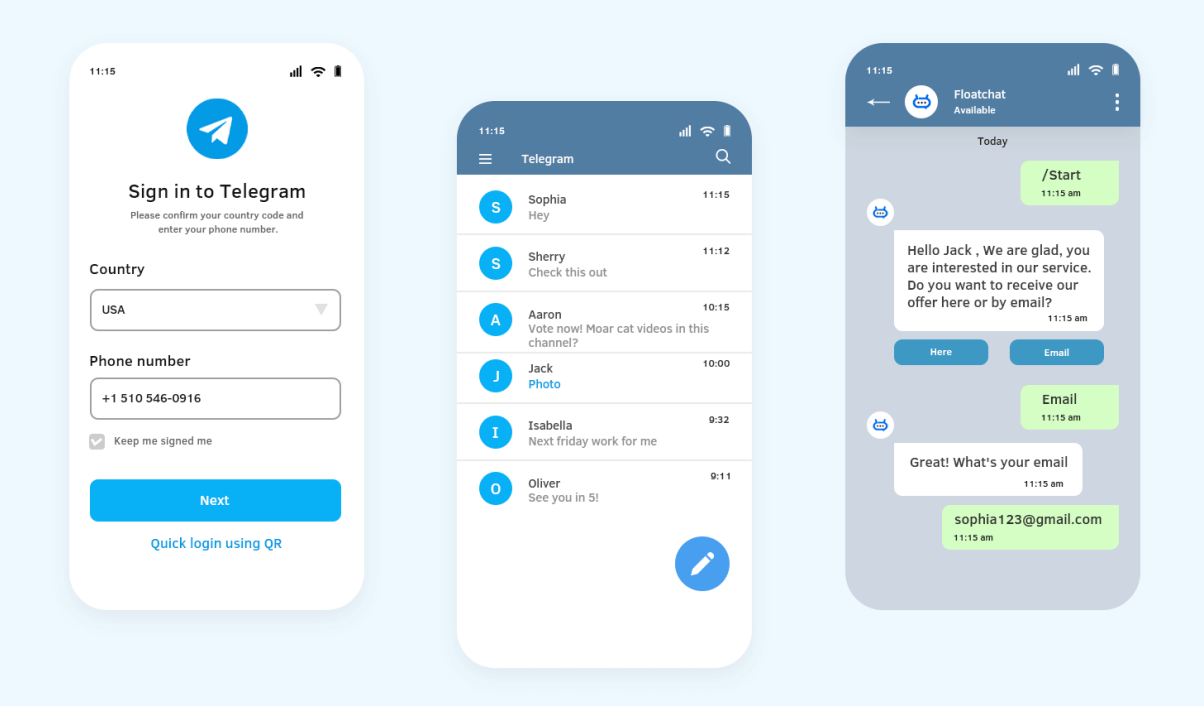


Рисунок 1.2 – Загальний вигляд платформи Telegram на телефоні

Однак, незважаючи на численні переваги, використання Telegram для обробки замовлень та резервувань також має певні виклики та обмеження. По–перше, безпека та конфіденційність даних значною мірою залежать від правильної конфігурації та налаштувань захисту бота, оскільки неналежний рівень безпеки може призвести до витоку конфіденційних даних клієнтів. По–друге, обробка великої кількості замовлень та резервувань може вимагати додаткових ресурсів та оптимізації роботи бота для забезпечення належної швидкодії та відгуку.

Ще одним викликом є інтеграція платіжних систем для проведення оплат через Telegram–бота. Хоча Telegram підтримує деякі платіжні шлюзи, проте їх кількість обмежена, і компаніям може знадобитися додаткова інтеграція з іншими

платіжними системами або створення власних рішень для безпечної обробки платежів [10].

Незважаючи на ці виклики, використання Telegram для обробки замовлень та резервувань залишається привабливим рішенням для багатьох компаній. Завдяки своїй зручності, швидкості та високому рівню безпеки, платформа Telegram дозволяє надавати клієнтам якісний сервіс та забезпечувати належний захист їхніх конфіденційних даних.

Розвиток месенджер–ботів є значною еволюцією у взаємодії між цифровими платформами та їхніми користувачами. Боти–месенджери, програмно автоматизовані інтерфейси в додатках для обміну повідомленнями, призначені для імітації розмовної взаємодії та виконання різноманітних завдань – від обслуговування клієнтів до доставки контенту та обробки транзакцій.

Фундаментальні принципи розробки ботів включають декілька ключових аспектів [11]. Перш за все, дизайн взаємодії з користувачем вимагає чіткого розуміння обробки природної мови (NLP), щоб точно інтерпретувати вхідні дані користувача і забезпечувати розмовний досвід, який відображає людську взаємодію, пропонуючи інтуїтивно зрозумілі та контекстно–залежні відповіді. Далі, інтеграція та масштабованість є критично важливими, оскільки боти повинні легко інтегруватися з існуючими системами і масштабуватися відповідно до потреб користувачів без погіршення продуктивності.

Це вимагає надійної внутрішньої архітектури, здатної витримувати значні навантаження і коливання у взаємодії. Безпека і конфіденційність також мають велике значення, оскільки боти часто обробляють конфіденційну особисту та фінансову інформацію [12]. Шифрування передачі даних і дотримання законів і правил про конфіденційність є обов'язковими. Безперервне навчання та адаптація через алгоритми машинного навчання дозволяють ботам вчитися на основі взаємодії та вдосконалювати свої реакції з часом, що підтримує їх актуальність та ефективність.

При створенні месенджер–ботів розробники стикаються з викликом створення такого інструменту, який був би не тільки технологічно продвинутим, але й легким

у використанні, зрозумілим і корисним для кінцевого користувача. Ефективність месенджер–бота вимірюється не лише його функціональністю, але й здатністю забезпечити приємний і зручний досвід користувача, водночас гарантуючи високий рівень безпеки і надійності. Основні принципи розробки месенджер–ботів повинні охоплювати як технічні аспекти, так і аспекти взаємодії з людиною (рис. 1.3) [14].

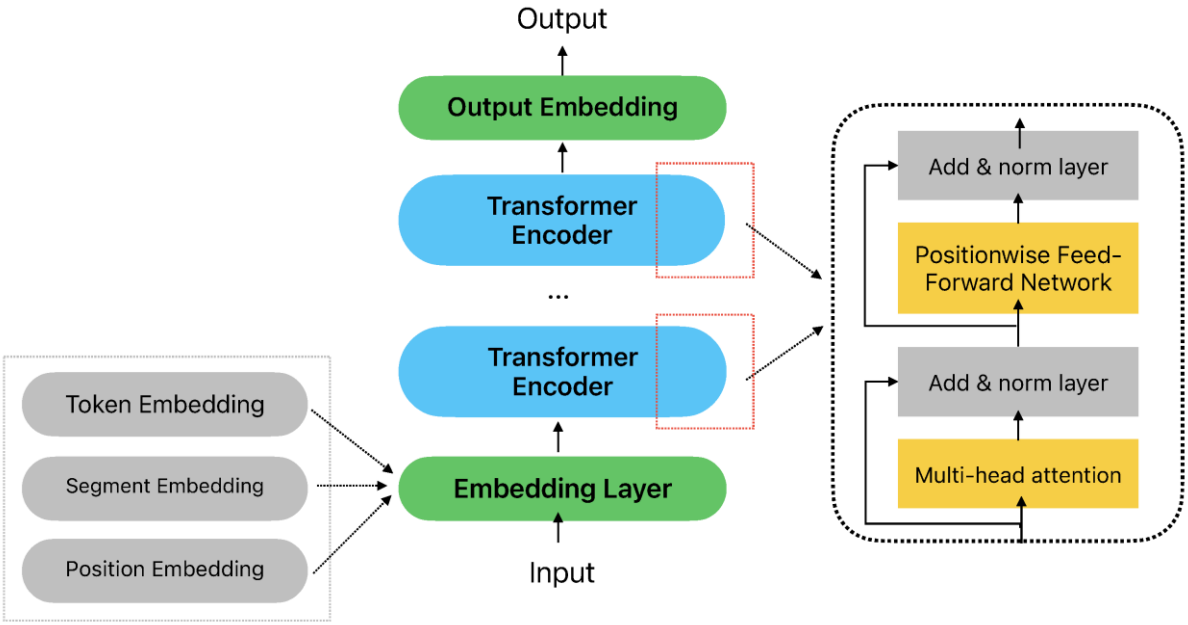


Рисунок 1.3 – Вигляд архітектури звичайного Telegram–бота

Персоналізація взаємодії може значно підвищити цінність бота для кінцевого користувача. Адаптація комунікації і функціональності до індивідуальних потреб і переваг користувачів робить взаємодію більш особистою і ефективною, збільшуючи лояльність до бренду або сервісу.

У сучасному цифровому світі забезпечення безпеки месенджер–ботів є критично важливим аспектом, що вимагає особливої уваги розробників. Проблематика безпеки охоплює ряд викликів, які необхідно вирішувати на всіх етапах життєвого циклу розробки та експлуатації ботів. Забезпечення безпеки ботів–месенджерів пов'язане з кількома проблемами, які вкрай важливо вирішити, враховуючи зростаючу залежність від цих ботів для обробки конфіденційної інформації та взаємодії [15]. Основні проблеми стосуються конфіденційності даних, несанкціонованого доступу та цілісності роботи ботів.

Боти часто передають конфіденційну інформацію, таку як особисті дані, платіжна інформація та конфіденційні повідомлення. Без надійних протоколів шифрування ці дані можуть бути перехоплені сторонніми особами. Також боти зберігають великі обсяги даних, до яких, якщо вони не захищені належним чином, може статися доступ або витік [16]. Забезпечення шифрування даних у стані спокою та безпечного управління ними має вирішальне значення для захисту від зловмисників.

Підхід Telegram відрізняється високим рівнем безпеки та конфіденційності даних, що робить його чудовим вибором для розробки ботів для обробки замовлень та резервувань. Опціональне наскрізне шифрування, політика неподілення даних користувачів та потужний API для розробки ботів з розширеними можливостями є значними перевагами Telegram порівняно з іншими платформами.

Хоча Slack та Facebook Messenger мають свої переваги у корпоративному середовищі та інтеграції з великими базами користувачів відповідно, вони поступаються Telegram у питаннях безпеки та приватності даних. Для обробки конфіденційної інформації, такої як замовлення та резервування, захист даних користувачів є критично важливим, і Telegram забезпечує найкращі можливості в цьому аспекті [17].

Таким чином, підхід з використанням Telegram та розробки власного бота є чудовим рішенням для створення безпечної та зручної системи обробки замовлень та резервувань завдяки поєднанню високого рівня безпеки, конфіденційності даних та потужного функціоналу для розробників ботів.

1.2 Аналіз відомих загроз, що виникають при розробці Telegram–ботів для обробки замовлень та резервувань

Розробка захищеного Telegram–бота для обробки замовлень та резервувань вимагає ретельного аналізу та врахування широкого спектру потенційних загроз безпеці. Ця сфера характеризується підвищеними ризиками, адже бот оперує конфіденційними даними користувачів, такими як персональна інформація, платіжні дані та деталі замовлень. Крім того, бот взаємодіє з користувачами через

відкритий канал зв'язку, що створює додаткові можливості для зловмисників здійснювати різноманітні атаки.

Загрози безпеці даних відіграють критичну роль у забезпеченні надійності та довіри до будь–якої системи, що обробляє конфіденційну інформацію, включаючи Telegram–боти. Однією з головних проблем у цьому аспекті є несанкціонований доступ до даних користувачів. Це означає, що зловмисники можуть отримати доступ до особистої чи фінансової інформації користувачів без їх відома або згоди, що може призвести до крадіжки ідентичності, фінансових втрат або інших негативних наслідків (рис. 1.4) [18].

Рисунок 1.4 – Типи загроз безпеки даних користувачів Telegram–ботів

Інша значна загроза пов'язана з вразливостями у коді самого бота. Навіть незначні помилки в програмному забезпеченні можуть відкрити шлях для витоку важливої інформації, якщо зловмисники виявлять і експлуатують ці вразливості [15]. Це може включати недоліки в реалізації протоколів безпеки, слабкі місця в алгоритмах шифрування або прогалини, що дозволяють обходити процедури аутентифікації.

Одна з найсерйозніших загроз, що постає перед розробниками захищеного Telegram–бота, – це несанкціонований доступ до конфіденційних даних користувачів. Зловмисники можуть використовувати різноманітні методи, такі як злам облікових записів, експлуатація вразливостей у програмному забезпеченні або

перехоплення трафіку, щоб отримати доступ до персональної інформації, платіжних даних чи деталей замовлень [19]. Наслідки такого витоку даних можуть бути катастрофічними, включаючи крадіжку особистих даних, фінансові втрати, шахрайство та втрату репутації компанії.

Іншою серйозною загрозою є маніпуляції з повідомленнями, які надсилаються між користувачами та ботом. Зловмисники можуть здійснювати атаки типу "людина посередині", перехоплюючи та модифікуючи повідомлення під час передачі. Це може призвести до відправлення користувачами некоректних даних або виконання небажаних дій, таких як оформлення замовлень з помилковими деталями або здійснення платежів на рахунки зловмисників.

Маніпуляції з повідомленнями включають сценарії, коли зловмисники намагаються перехопити, змінити або імітувати повідомлення між користувачами та ботом [48]. Це може бути здійснено через атаки "людина посередині", де зловмисник перехоплює трафік між користувачем і ботом, або шляхом використання вразливостей в системі аутентифікації бота, що дозволяє надсилати від його імені шкідливі або маніпулюючі повідомлення. Такі дії можуть призвести не тільки до витоку конфіденційної інформації, але й до відправлення користувачами некоректних даних або виконання небажаних дій, що може призвести до фінансових втрат або інших неприємностей (рис. 1.5) [20].

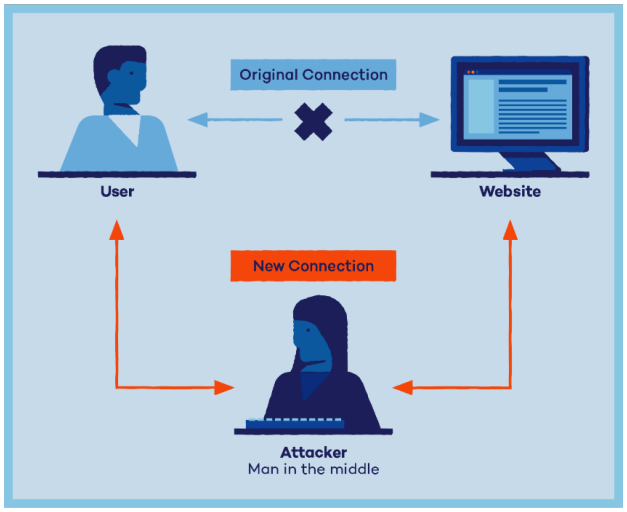


Рисунок 1.5 – Загальна структура атаки Man–in–Middle

Для захисту від маніпуляцій з повідомленнями розробники мають забезпечити належний рівень шифрування трафіку між користувачем і ботом, а також реалізувати ефективні методи аутентифікації та верифікації повідомлень. Важливим є також моніторинг активності бота на предмет незвичайних або підозрілих запитів, що можуть вказувати на спробу втручання.

Атаки, спрямовані на відмову в обслуговуванні (DoS та DDoS), становлять ще одну значну загрозу для стабільної роботи Telegram–бота. Ці атаки передбачають перевантаження системи величезною кількістю запитів або трафіком, що призводить до її зупинки або уповільнення роботи. Як наслідок, легітимні користувачі не можуть отримати доступ до послуг бота, що негативно впливає на репутацію компанії та може призвести до фінансових втрат (рис.1.6) [21].

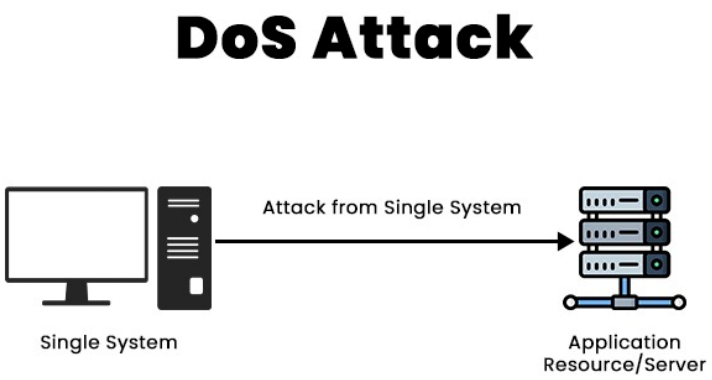


Рисунок 1.6 – Загальна структура DoS атаки

Для захисту від DoS атак слід використовувати механізми обмеження швидкості запитів, автоматичне виявлення та блокування підозрілих IP–адрес, а також застосування рішень на рівні інфраструктури, таких як розподілені системи обробки запитів і використання хмарних служб з вбудованими можливостями захисту від DDoS [22]. Враховуючи постійний розвиток методів нападу, важливо постійно оновлювати та адаптувати системи безпеки для ефективного протистояння новим загрозам.

Фішинг та соціальна інженерія базуються на використанні психологічних прийомів для обману користувачів з метою отримання до них конфіденційної інформації [23]. Зловмисники можуть створювати фейкові боти, які на вигляд не відрізняються від справжнього бота, але при цьому збирають введену користувачами інформацію. Також зловмисники можуть надсилати маніпулятивні повідомлення від імені легітимного бота, спонукаючи користувачів надати особисту інформацію або фінансові дані. Ці методи можуть бути використані не тільки для крадіжки інформації, але й для ширення шкідливого ПЗ або виконання інших шахрайських дій.

Захист від фішингу та соціальної інженерії вимагає комплексного підходу, що включає освітні кампанії для користувачів, інформуючи їх про потенційні загрози та безпечні практики використання ботів. Важливо також впровадити додаткові методи верифікації та аутентифікації, щоб ускладнити зловмисникам маніпулювання користувачами.

Вразливості програмного забезпечення – це технічний аспект безпеки, що включає баги або слабкі місця у коді бота, які можуть бути використані зловмисниками для виконання шкідливого коду, отримання несанкціонованого доступу до системи або здійснення інших неправомірних дій [24]. Наприклад, через вразливості в коді бота зловмисник може отримати доступ до бази даних користувачів, змінювати логіку роботи бота або використовувати бота як частину ботнету для здійснення DDoS–атак. Такі вразливості часто виникають через помилки у процесі розробки, недоліки тестування або використання застарілого програмного забезпечення.

Для запобігання вразливостям у програмному забезпеченні критично важливою є регулярна перевірка та оновлення коду, використання сучасних інструментів для автоматичного виявлення вразливостей, а також застосування принципів безпечного програмування на всіх етапах розробки. Враховуючи швидкість розвитку технологій та методів кібератак, неперервне оновлення знань та умінь у сфері кібербезпеки є ключовим для захисту ботів від цих загроз [25].

Для захисту від цих загроз, розробники мають зосередити свої зусилля на ретельній перевірці безпеки коду, імплементації сучасних методів аутентифікації та шифрування, регулярному оновленні програмного забезпечення та проведенні інформаційних кампаній з освіти користувачів щодо потенційних загроз і методів їх уникнення. Також важливим є впровадження систем моніторингу та аудиту діяльності бота для своєчасного виявлення та реагування на безпекові інциденти.

Вимоги до захисту інформації та даних користувачів в рамках розробки та впровадження Telegram–бота для оброблення замовлень та резервувань вимагають комплексного підходу, що включає застосування найкращих практик і стандартів кібербезпеки. Забезпечення безпеки даних не лише допомагає уникнути потенційних загроз і втрати інформації, але й підтримує довіру користувачів та забезпечує дотримання нормативних вимог.

Основною вимогою є шифрування всіх конфіденційних даних, які передаються чи зберігаються. Це стосується не лише інформації, яка пересилається між користувачем та ботом у реальному часі, але й даних, що зберігаються на серверах або в хмарних сховищах. Використання сучасних алгоритмів шифрування та захищених протоколів з'єднання, таких як TLS, є обов'язковим для захисту інформації від перехоплення та несанкціонованого доступу [26].

Іншою критично важливою вимогою є реалізація ефективних механізмів аутентифікації та авторизації. Користувачі повинні проходити надійну аутентифікацію перед взаємодією з ботом, що дозволяє мінімізувати ризики несанкціонованого доступу. Авторизація дій у межах системи допомагає контролювати, що користувачі мають доступ лише до тих операцій, які дозволені згідно з їхніми правами доступу.

Також необхідно регулярно оновлювати програмне забезпечення, використовувати патчі безпеки для відомих вразливостей і проводити аудити безпеки. Це допомагає забезпечити, що система захищена від останніх відомих загроз та що всі компоненти системи дотримуються сучасних вимог безпеки. Важливо забезпечити відповідність всім регуляторним вимогам і стандартам захисту даних, таким як GDPR у Європейському Союзі, що вимагає від компаній

забезпечувати високий рівень захисту особистих даних користувачів та інформувати їх про будь–які інциденти, які можуть вплинути на їхню конфіденційність [27].

Удосконалення існуючих методів розробки та впровадження ботів для месенджерів є необхідним з огляду на обмеження сучасних технологій у контексті обробки природної мови, інтеграції з різними системами, забезпечення безпеки та масштабованості.

По–перше, покращення обробки природної мови (NLP) є критичним для підвищення точності розуміння контексту та намірів користувача. Використання сучасних моделей машинного навчання та глибокого навчання, таких як трансформери, дозволяє ботам забезпечувати більш адекватні відповіді на запити користувачів [28]. Це зменшує ймовірність виникнення помилок у розумінні тексту та підвищує загальну якість взаємодії.

По–друге, оптимізація структур баз даних спрямована на зменшення часу відповіді на запити та покращення ефективності обробки даних. Ефективне структурування даних допомагає уникати дублювання та знижувати ризик виникнення помилок. Однак надмірна нормалізація може ускладнити структуру бази даних і зменшити продуктивність через необхідність виконання великої кількості JOIN–запитів.

По–третє важливо також враховувати, що загрози безпеці не є статичними – вони постійно еволюціонують, і зловмисники розробляють нові, все більш складні методи атак. Тому розробники мають бути готові до постійного вдосконалення систем безпеки, відстежувати нові тенденції та загрози, а також своєчасно оновлювати програмне забезпечення для усунення виявлених вразливостей.

Важливо пам'ятати, що захист від загроз безпеці – це не разовий захід, а безперервний процес. Розробники мають постійно відстежувати нові тенденції та вразливості, оновлювати системи безпеки та навчати користувачів принципам безпечного використання Telegram–бота. Тільки комплексний підхід, що поєднує технічні рішення та освітні заходи, може забезпечити належний рівень безпеки та

захисту конфіденційних даних при обробці замовлень та резервувань через Telegram–бота.

1.3 Аналіз існуючих програмних рішень обробки замовлень та резервувань

На сучасному ринку існує широкий спектр програмних рішень для обробки замовлень та резервувань, розроблених різними компаніями та призначених для різних галузей. Ці системи відрізняються за своїми функціональними можливостями, інтерфейсом, масштабованістю та рівнем безпеки. Проаналізуємо деякі з найбільш поширених рішень та виділимо їхні переваги та недоліки.

Одним з популярних рішень є система управління взаємовідносинами з клієнтами CRM від компанії Salesforce. Вона пропонує комплексний підхід до управління замовленнями, резервуваннями, клієнтськими даними та взаємодією з клієнтами. Salesforce CRM має потужні аналітичні інструменти, які дозволяють відстежувати тенденції продажів та оптимізувати бізнес–процеси [29]. Однак, ця система може бути дорогою для малих та середніх підприємств, а також вимагає значних зусиль для налаштування та інтеграції з іншими системами.

Іншим поширеним рішенням є платформа управління ресторанним бізнесом Toast. Вона спеціалізується на обробці замовлень, резервувань та управлінні операціями в ресторанах та закладах харчування. Toast пропонує зручний інтерфейс для персоналу, можливість онлайн–резервувань для клієнтів, інтеграцію з системами доставки їжі та аналітичні інструменти. Проте, вона може не підходити для інших галузей, крім ресторанного бізнесу, та має обмежені можливості для налаштування під специфічні потреби.

Ще одним цікавим рішенням є платформа обробки замовлень OrderEase, яка фокусується на електронній комерції та обробці замовлень для онлайн–магазинів. Вона пропонує функції автоматизації процесу обробки замовлень, інтеграцію з різними платіжними шлюзами, управління інвентарем та аналітику продажів. Однак, OrderEase може не підходити для оффлайн–бізнесу або компаній, які надають послуги замість продажу товарів.

У сфері туризму та готельного бізнесу популярним рішенням є система управління власністю PMS від Opera. Вона дозволяє ефективно управляти резервуваннями номерів, тарифами, персоналом та іншими аспектами готельного бізнесу. Opera PMS має потужні інструменти для аналізу даних та управління доходами, але може бути складною для впровадження та вимагати значних інвестицій (табл. 1.2) [30].

Таблиця 1.2 – Порівняння існуючих програмних рішень для обробки замовлень та резервувань

Існуючі програмні рішення	Особливості	Переваги	Недоліки
Salesforce CRM	Управління замовленнями, резервуваннями, клієнтськими даними, аналітика	Потужні аналітичні інструменти, масштабованість	Висока вартість, складність налаштування та інтеграції
Toast	Обробка замовлень, резервувань, управління операціями	Зручний інтерфейс, інтеграція з сервісами доставки	Орієнтоване лише на ресторанну сферу
OrderEase	Автоматизація обробки замовлень, інтеграція платіжних шлюзів, управління інвентарем	Зосереджене на онлайн–замовленнях	Не підходить для офлайн–бізнесу або послуг
Opera PMS	Управління резервуваннями номерів, тарифами, персоналом	Потужні інструменти аналітики та управління доходами	Складність впровадження, високі інвестиції

Незважаючи на різноманітність існуючих програмних рішень для обробки замовлень та резервувань, більшість з них мають певні недоліки та обмеження. Одним із основних недоліків є відсутність інтегрованої системи взаємодії з клієнтами через месенджери, такі як Telegram. Більшість цих рішень зосереджені на веб–інтерфейсах, мобільних додатках або традиційних каналах зв'язку, таких як електронна пошта або телефон.

Крім того, багато існуючих систем не приділяють належної уваги питанням безпеки та захисту конфіденційних даних користувачів. Часто заходи безпеки реалізовані на базовому рівні, що може створювати ризики для компаній, які обробляють великі обсяги конфіденційної інформації, такої як платіжні дані або персональні дані клієнтів.

Ще одним недоліком є обмежена гнучкість та можливості налаштування під специфічні потреби бізнесу. Більшість програмних рішень пропонують стандартний набір функцій, який може не повністю відповідати вимогам певних галузей або компаній з унікальними бізнес–процесами.

Таким чином, існуючі програмні рішення для обробки замовлень та резервувань мають певні переваги та багато недоліки. Хоча вони пропонують різноманітні функції та інструменти, їм часто бракує інтегрованої системи взаємодії з клієнтами через месенджери, належного рівня безпеки та гнучкості для налаштування під специфічні потреби бізнесу. Це відкриває можливості для розробки нових рішень, таких як захищений Telegram–бот, який може усунути ці недоліки та запропонувати зручний, безпечний та гнучкий спосіб обробки замовлень та резервувань.

У контексті обробки замовлень та резервувань через месенджер Telegram, варто розглянути деякі існуючі рішення, що використовують цю платформу [31].

Один із прикладів – Telegram–бот від сервісу онлайн–бронювання житла Booking.com. Цей бот дозволяє користувачам шукати та резервувати готелі, апартаменти чи інші варіанти проживання прямо в Telegram. Він має зручний інтерфейс з кнопками та меню для навігації, але відсутність додаткових заходів безпеки може становити ризик при передачі конфіденційних даних.

Інше рішення – Telegram–бот від сервісу доставки їжі Uber Eats. Він дозволяє переглядати меню ресторанів, робити замовлення та відстежувати статус доставки прямо в месенджері. Однак цей бот орієнтований лише на замовлення їжі та не підходить для інших видів послуг чи товарів [32].

Компанія Travelline також має Telegram–бот для бронювання квитків на різні види транспорту, включаючи авіа, залізничний та автобусний. Бот пропонує пошук

маршрутів, вибір дат та класів обслуговування, але може бути обмеженим у функціоналі для складних бронювань або спеціальних вимог [33].

Незважаючи на наявність цих рішень, більшість існуючих Telegram–ботів для обробки замовлень та резервувань мають ряд недоліків. Часто вони зосереджені на вузькій ніші або типі послуг, не маючи універсального характеру. Крім того, питання безпеки та захисту конфіденційних даних користувачів часто не розглядаються належним чином, що може становити ризики при обробці платіжної інформації чи персональних даних (табл 1.3) [34].

Таблиця 1.3 – Порівняння існуючих Telegram–ботів для обробки замовлень та резервувань

Бот	Особливості	Переваги	Недоліки
Booking.com	Пошук та бронювання житла	Зручний інтерфейс з кнопками та меню	Відсутність додаткових заходів безпеки
Uber Eats	Замовлення їжі та відстеження доставки	Інтеграція з сервісом доставки	Обмежений лише замовленням їжі
Travelline	Бронювання квитків на різні види транспорту	Пошук маршрутів та класів обслуговування	Обмежений функціонал для складних бронювань

Ще одним недоліком є обмежена гнучкість та можливості налаштування під специфічні потреби бізнесу. Більшість існуючих ботів пропонують стандартний набір функцій, який може не повністю відповідати вимогам певних галузей або компаній з унікальними бізнес–процесами [35].

Таким чином, хоча деякі компанії вже використовують Telegram для обробки замовлень та резервувань, ця сфера все ще залишається відносно новою та недостатньо розвиненою. Існуючі рішення часто мають обмежений функціонал, недостатній рівень безпеки або не враховують специфічні потреби різних бізнесів. Це підкреслює необхідність розробки нового, ефективного та захищеного Telegram–боту, здатного забезпечити зручну, безпечну та гнучку обробку замовлень та резервувань для компаній різних галузей.

1.4 Висновки та постановка задач

Проведений аналіз особливостей використання платформи Telegram, потенційних загроз безпеки та існуючих програмних рішень для обробки замовлень та резервувань дозволяє зробити наступні висновки, крім того, проаналізовано існуючі рішення та найкращі підходи у цій галузі.

Існуючі програмні рішення для обробки замовлень та резервувань, такі як CRM–системи та галузеві платформи, не повною мірою враховують переваги використання месенджерів та часто мають обмежений функціонал або недостатній рівень безпеки. Таким чином месенджер Telegram є привабливою платформою для розробки бота з обробки замовлень та резервувань завдяки зручному функціоналу, високому рівню безпеки, великій користувацькій базі та можливості створення власних ботів.

Розглянуті наявні відомі Telegram–боти для обробки замовлень та резервувань зосереджені на вузьких нішах, не пропонують універсального рішення та не приділяють належної уваги питанням безпеки та захисту конфіденційних даних користувачів.

Виходячи з цих висновків, можна сформулювати наступні завдання для розробки захищеного Telegram–бота оброблення замовлень та резервувань:

– спроектувати архітектуру та обґрунтувати підхід захищеного Telegram–бота;

– спроєктувати базу даних для зберігання замовлень, резервувань та персональних даних користувачів з належним рівнем захисту;

– реалізувати програмну частину Telegram–бота з використанням обраних технологій та інструментів розробки, впровадивши заходи безпеки та захисту даних.

Успішне виконання поставлених завдань дозволить розробити універсальне рішення у вигляді захищеного Telegram–бота для обробки замовлень та резервувань, який поєднає зручність використання месенджера з високим рівнем безпеки та захисту конфіденційних даних користувачів.

РОЗДІЛ 2. РОЗРОБЛЕННЯ ПІДХОДУ ЗАХИЩЕНОГО TELEGRAM–БОТА ТА ПРОЄКТУВАННЯ КОМПОНЕНТІВ АРХІТЕКТУРИ ОБРОБЛЕННЯ ЗАМОВЛЕНЬ ТА РЕЗЕРВУВАНЬ

2.1 Обґрунтування розроблення підходу захищеного Telegram–бота оброблення замовлень та резервувань

Виходячи з проведеного у розділі 1 аналізу особливостей використання платформи Telegram, потенційних загроз безпеки та недоліків існуючих програмних рішень для обробки замовлень та резервувань, стає очевидною необхідність розроблення нового підходу у вигляді захищеного Telegram–бота. Такий підхід дозволить поєднати зручність та популярність месенджера Telegram з високим рівнем безпеки та захисту конфіденційних даних користувачів, що є критично важливим при обробці замовлень та резервувань.

Перш за все, розроблення захищеного Telegram–бота обґрунтовується потребою у сучасному та зручному способі взаємодії з клієнтами під час оформлення замовлень та резервувань. Месенджери, такі як Telegram, стали невід'ємною частиною повсякденного життя людей, тож їх використання для цих цілей є логічним та зручним рішенням [36]. Бот забезпечить швидкий та інтуїтивний доступ до послуг компанії, дозволяючи клієнтам здійснювати необхідні дії в звичному для них інтерфейсі месенджера.

Крім зручності, Telegram також пропонує високий рівень безпеки та захисту даних завдяки використанню потужного шифрування та суворої політики конфіденційності. Це є критично важливим для захищеного Telegram–бота, оскільки він буде оперувати конфіденційною інформацією, такою як персональні дані клієнтів, платіжні дані та деталі замовлень. Розроблений підхід має забезпечити належні заходи безпеки та конфіденційності, щоб запобігти витокам даних, несанкціонованому доступу чи іншим загрозам.

Ще одним важливим обґрунтуванням для розробки захищеного Telegram–бота є необхідність подолання недоліків існуючих програмних рішень для обробки

замовлень та резервувань. Як було показано в аналізі, більшість цих рішень не враховують переваги використання месенджерів, мають обмежений функціонал або недостатній рівень безпеки. Розроблений підхід має усунути ці недоліки, пропонуючи універсальне та гнучке рішення з високим рівнем захисту конфіденційних даних.

Крім того, розробка захищеного Telegram–бота є актуальною з огляду на зростаючі вимоги до безпеки та захисту даних у сучасному цифровому середовищі. Компанії все частіше стикаються з різноманітними кіберзагрозами, такими як хакерські атаки, фішинг та витоки даних. Впровадження захищеного Telegram–бота дозволить компаніям відповідати найвищим стандартам безпеки та захисту конфіденційних даних клієнтів, підвищуючи довіру та репутацію серед споживачів.

Нарешті, розроблення підходу захищеного Telegram–бота обґрунтовується його потенційною економічною вигодою. Автоматизація процесів обробки замовлень та резервувань через бота може значно скоротити витрати на обслуговування клієнтів, підвищити ефективність бізнес–процесів та збільшити прибутки компанії. Крім того, впровадження захищеного рішення дозволить уникнути можливих збитків, пов'язаних з витоками даних чи кіберінцидентами.

Таким чином, розроблення підходу захищеного Telegram–бота оброблення замовлень та резервувань є обґрунтованим з точки зору зручності та сучасних способів взаємодії з клієнтами, забезпечення високого рівня безпеки та захисту конфіденційних даних, подолання недоліків існуючих рішень, відповідності сучасним вимогам до безпеки даних та потенційної економічної вигоди для компаній [37]. Цей підхід дозволить створити універсальне та гнучке рішення, здатне забезпечити надійну, зручну та безпечну обробку замовлень і резервувань для компаній різних галузей.

Для наочної демонстрації підходу захищеного Telegram–бота оброблення замовлень та резервувань наведено блок–схему алгоритму його роботи (рис. 2.1), який демонструє покроковий процес взаємодії з користувачем, обробки замовлень та резервувань, а також впровадження заходів безпеки для захисту конфіденційних даних.

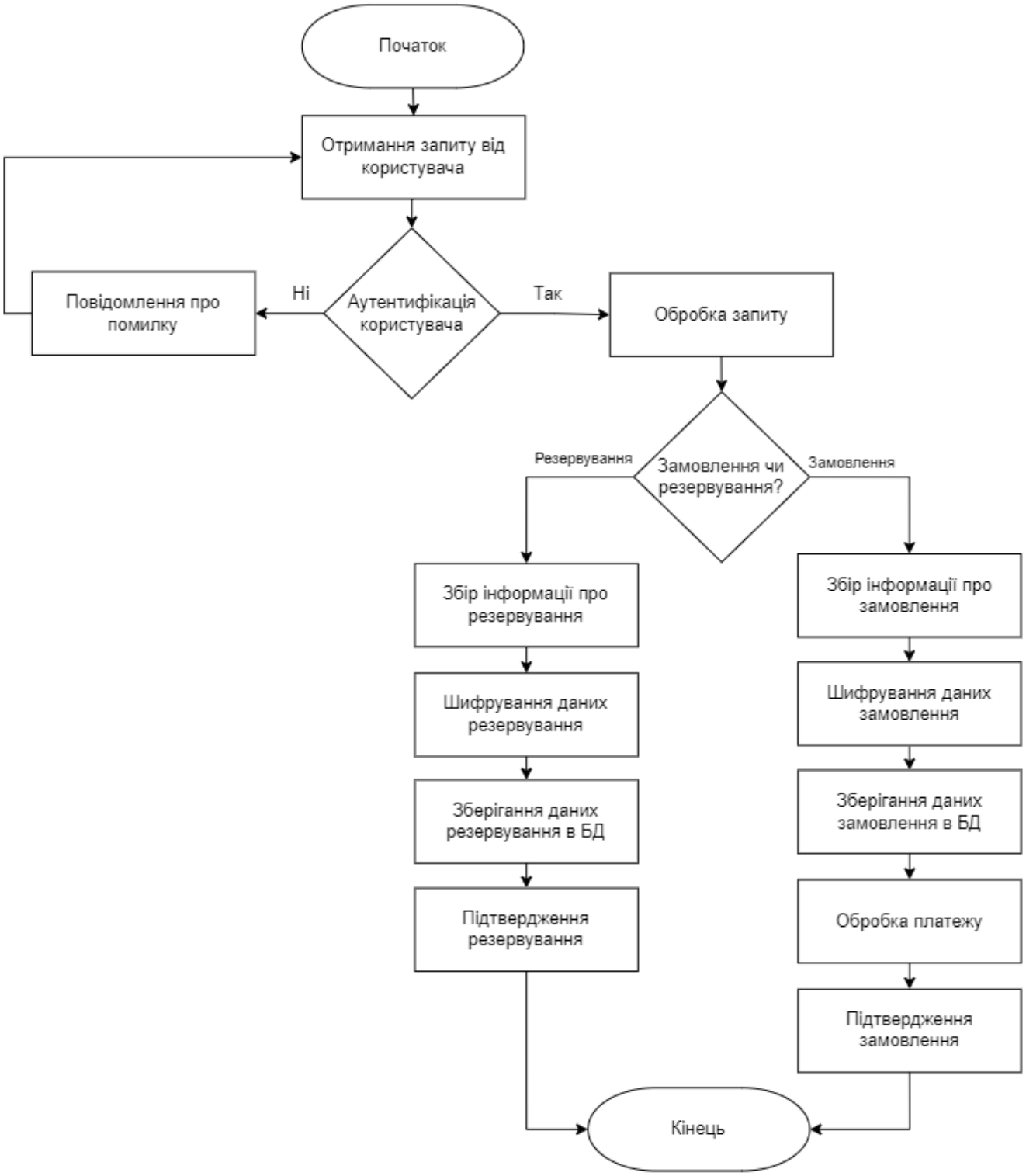


Рисунок 2.1 – Блок–схема алгоритму роботи алгоритму захищеного Telegram–бота оброблення замовлень та резервувань

Блок–схема візуалізує основні етапи функціонування захищеного Telegram–бота, починаючи від ініціалізації взаємодії з користувачем та завершуючи успішним оформленням замовлення чи резервування із забезпеченням належного рівня

безпеки. Вона демонструє, як бот обробляє вхідні запити користувачів, збирає необхідну інформацію, взаємодіє з базою даних для зберігання замовлень та резервувань, а також впроваджує різноманітні заходи безпеки, такі як шифрування даних, аутентифікація користувачів та захист від потенційних загроз.

Крок 1. Початок. Блок–схема починається з отримання запиту від користувача через Telegram–бот.

Крок 2. Отримання запиту від користувача. Бот отримує запит від користувача, який може бути пов'язаний з замовленням чи резервуванням.

Крок 3. Аутентифікація користувача. Після отримання запиту бот перевіряє автентифікацію користувача. Якщо аутентифікація пройшла успішно, бот переходить до обробки запиту. Якщо аутентифікація не вдалася, бот повідомляє про помилку та повертається до отримання запиту від користувача.

Крок 4. Повідомлення про помилку. У випадку невдалої аутентифікації бот повідомляє користувача про помилку та очікує нового запиту.

Крок 5. Обробка запиту. Якщо аутентифікація пройшла успішно, бот обробляє запит користувача.

Крок 6. Перевірка на тип запиту (замовлення чи резервування). Бот визначає, чи запит користувача стосується замовлення чи резервування.

Крок 7. Збір інформації про резервування. Якщо запит стосується резервування, бот збирає необхідну інформацію про резервування від користувача.

Крок 8. Шифрування даних резервування. Після збору інформації бот шифрує дані резервування для забезпечення безпеки.

Крок 9. Зберігання даних резервування в базі даних (БД). Зашифровані дані резервування зберігаються в базі даних.

Крок 10. Підтвердження резервування. Бот підтверджує успішне резервування для користувача.

Крок 11. Збір інформації про замовлення. Якщо запит стосується замовлення, бот збирає необхідну інформацію про замовлення від користувача.

Крок 12. Шифрування даних замовлення. Після збору інформації бот шифрує дані замовлення для забезпечення безпеки.

Крок 13. Зберігання даних замовлення в БД. Зашифровані дані замовлення зберігаються в базі даних.

Крок 14. Обробка платежу. Для замовлень бот обробляє платіж користувача.

Крок 15. Підтвердження замовлення. Після успішної обробки платежу бот підтверджує замовлення для користувача.

Крок 16. Кінець. Блок–схема завершується після успішного оформлення замовлення або резервування.

Ця блок–схема демонструє основний робочий процес захищеного Telegram–бота для обробки замовлень та резервувань, включаючи аутентифікацію користувача, збір інформації, шифрування даних, взаємодію з базою даних, обробку платежів (для замовлень) та підтвердження операцій.

Таким чином, комплескний підхід із захисту Telegram–бота включає у себе [38]:

– фільтрація трафіку – на сервері, де розгорнуто Telegram–бот, буде налаштовано брандмауер (firewall) з встановленими правилами фільтрації трафіку. Ці правила дозволять приймати запити лише з відомих IP–адрес серверів Telegram, блокуючи при цьому весь інший вхідний трафік. Це запобігатиме спробам атак на бот з невідомих або зловмисних джерел.

– використання вебхуків – для безпечної взаємодії з Telegram буде використано механізм вебхуків. URL вебхука буде захищено за допомогою SSL–сертифіката та не розкриватиметься публічно. Доступ до вебхука матимуть лише сервери Telegram, що унеможливить несанкціонований доступ до бота.

– шифрування даних – всі конфіденційні дані, такі як платіжна інформація, персональні дані клієнтів чи деталі замовлень, передаються між ботом і серверами Telegram у зашифрованому вигляді за допомогою HTTPS, що захищає від перехоплення.

– аутентифікація користувачів – для управління ботом та доступу до конфіденційної інформації реалізована багаторівнева система аутентифікації користувачів. Це включає використання унікальних ідентифікаторів користувачів Telegram, таких як токени доступу [39]. Лише авторизовані користувачі з

відповідними дозволами зможуть керувати ботом або отримувати доступ до захищених даних.

Такий комплекс захисту даних під час обробки замовлень та резервувань за допомогою Telegram–бота наведено у вигляді блок–схеми (рис.2.2)

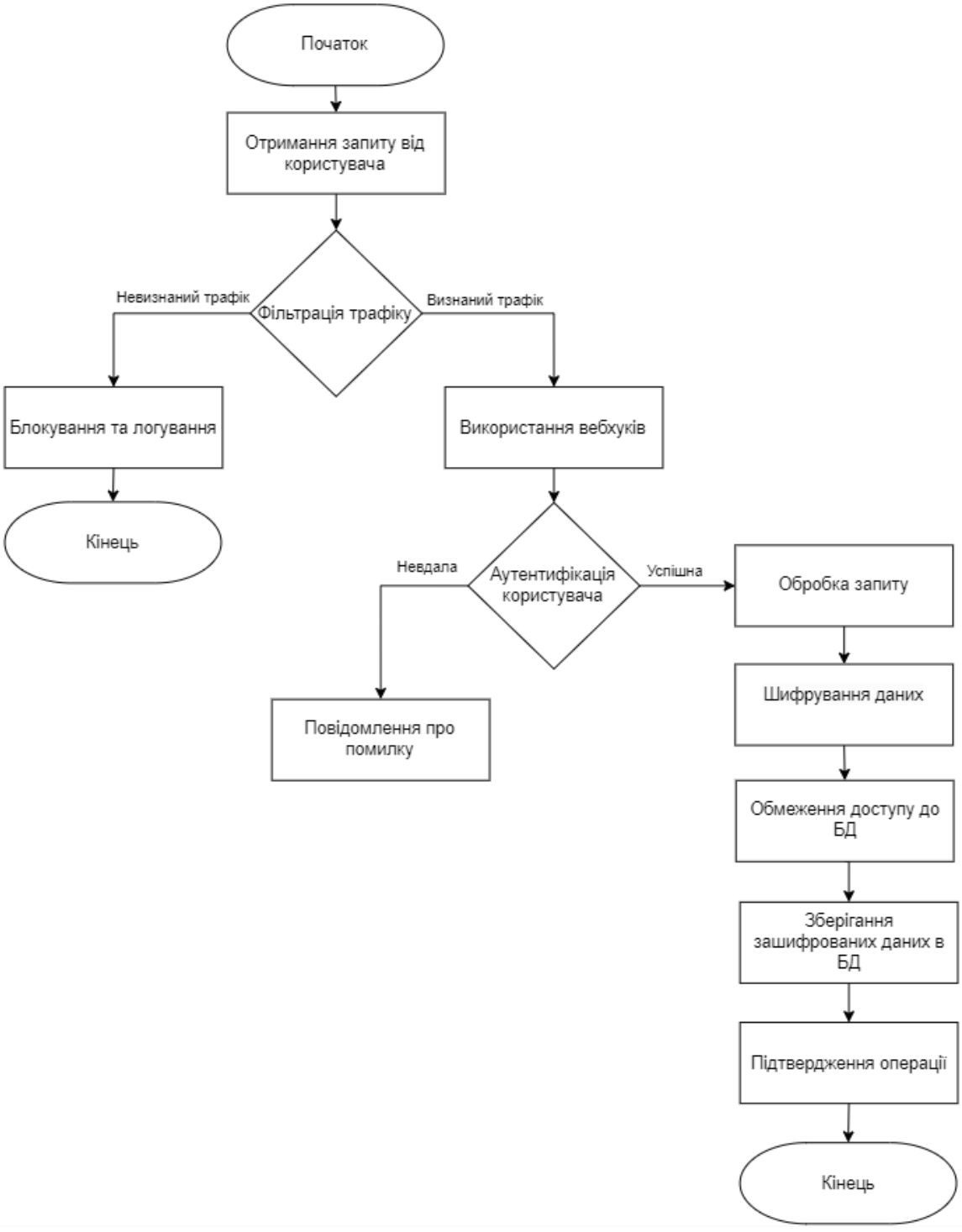


Рисунок 2.2 – Блок–схема алгоритму роботи підходу захищеного Telegram–бота оброблення замовлень та резервувань

Крок 1. Початок. Процес починається, коли користувач взаємодіє із захищеним Telegram–ботом для здійснення замовлення або резервування.

Крок 2. Отримання запиту від користувача. Бот отримує запит від користувача через месенджер Telegram.

Крок 3. Фільтрація трафіку. Система фільтрації трафіку перевіряє джерело запиту. Якщо джерело невизнане (не від серверів Telegram), запит блокується та логується. Якщо джерело визнане, процес переходить до наступного кроку.

Крок 4. Блокування та логування. У випадку невизнаного трафіку система блокує запит та записує інформацію про інцидент у логи для подальшого аналізу. Після цього процес завершується.

Крок 5. Використання вебхуків. Для взаємодії з Telegram використовуються вебхуки, що забезпечує авторизований доступ лише з визнаних джерел.

Крок 6. Аутентифікація користувача. Відбувається процес аутентифікації користувача для перевірки його дозволів та доступу до системи. Якщо аутентифікація не вдалася, користувачеві надсилається повідомлення про помилку, і процес повертається до кроку 2.

Крок 7. Повідомлення про помилку. У випадку невдалої аутентифікації користувач отримує повідомлення про помилку та запрошується повторити спробу.

Крок 8. Обробка запиту. Якщо аутентифікація пройшла успішно, бот обробляє запит користувача на здійснення замовлення або резервування.

Крок 9. Шифрування даних. Конфіденційні дані замовлення чи резервування шифруються для забезпечення безпеки та захисту інформації.

Крок 10. Обмеження доступу до бази даних. Реалізовано механізм обмеження доступу до бази даних, який дозволяє лише авторизованим користувачам взаємодіяти з нею та запобігає несанкціонованому доступу.

Крок 11. Зберігання зашифрованих даних у базі даних. Зашифровані дані замовлення чи резервування безпечно зберігаються у базі даних системи.

Крок 12. Підтвердження операції. Після успішного збереження даних користувачеві надсилається підтвердження про успішне здійснення замовлення чи резервування.

Крок 13. Кінець. Процес обробки замовлення чи резервування завершується.

Ця блок–схема демонструє, як система забезпечує безпеку та захист конфіденційних даних на різних етапах процесу обробки замовлень та резервувань.

Комплексне впровадження цих заходів безпеки забезпечить надійний захист Telegram–бота від різноманітних загроз, гарантуватиме конфіденційність та цілісність даних клієнтів, а також дозволить безпечно обробляти замовлення та резервування через месенджер [40].

Таким чином, запропонований метод захисту для захищеного Telegram–бота є актуальним та необхідним для забезпечення конфіденційності даних, підвищення безпеки взаємодії, відповідності нормативним вимогам, зміцнення довіри клієнтів, гнучкості та масштабованості, а також отримання конкурентної переваги на ринку.

2.2 Обґрунтування взаємодії захищеного Telegram–бота із базою даних для зберігання замовлень та резервувань

Важливим аспектом є також впровадження заходів безпеки на рівні бази даних, включаючи шифрування конфіденційних даних користувачів, щоб забезпечити їх захист від несанкціонованого доступу або витоку. Це вимагає вибору технологій, які надають вбудовані інструменти для шифрування або дозволяють легко інтегрувати зовнішні рішення безпеки [41].

Для забезпечення оптимальної роботи бота необхідно правильно вибрати бази даних, які будуть використовуватися для зберігання і обробки даних. У цьому контексті були вибрані наступні типи баз даних:

– реляційна база даних PostgreSQL – обрана для зберігання структурованих даних завдяки її потужним можливостям для роботи з транзакціями та забезпеченням цілісності даних [42]. Вона підтримує складні запити та надає широкий набір інструментів для роботи з даними. Ця база даних підходить для зберігання інформації про користувачів, запити, логи та інші структуровані дані, що вимагають суворого контролю цілісності.

– нереляційна база даних MongoDB –вибрана для обробки великих обсягів неструктурованих даних [43]. Вона забезпечує високу масштабованість та

продуктивність, що дозволяє ефективно працювати з динамічними та часто змінюваними даними, такими як журнали подій або кешування сесій користувачів. MongoDB зберігає дані у форматі JSON, що спрощує роботу з гнучкими структурами даних та прискорює процеси читання та запису (табл.2.1) [44].

Таблиця 2.1 – Обгрунтування вибору баз даних для забезпечення оптимальної роботи із розробленим Telegram–ботом

Критерій	PostgreSQL (реляційна)	MongoDB (нереляційна)
Структура даних	Строга схема, таблиці	Гнучка, без схеми
Цілісність даних	Висока, завдяки транзакціям та обмеженням	Низька, відсутні вбудовані механізми забезпечення цілісності
Продуктивність для операцій читання/запису	Висока для структурованих даних	Вища для неструктурованих даних
Масштабованість	Вертикальна (додавання потужностей серверу)	Горизонтальна (розподіл між серверами)
Забезпечення безпеки	Потужні механізми безпеки та контролю доступу	Обмежені можливості безпеки
Підтримка транзакцій	Повна підтримка ACID транзакцій	Обмежена підтримка транзакцій
Схема даних	Фіксована, зміна схеми складна	Схема даних динамічна, легко змінюється

Таким чином, комбінація реляційної бази даних PostgreSQL для зберігання структурованих даних замовлень та резервувань і нереляційної бази даних MongoDB для обробки неструктурованих даних забезпечить оптимальне рішення для захищеного Telegram–бота, поєднуючи безпеку, цілісність, продуктивність та масштабованість.

UML діаграми візуалізують структуру даних для двох баз даних, що використовуватимуться у розробці захищеного Telegram–бота. Реляційна база даних PostgreSQL містить наступні сутності (рис. 2.3):

- USERS – зберігає інформацію про користувачів, включаючи їх ідентифікатори, імена користувачів, паролі, електронні адреси та номери телефонів.

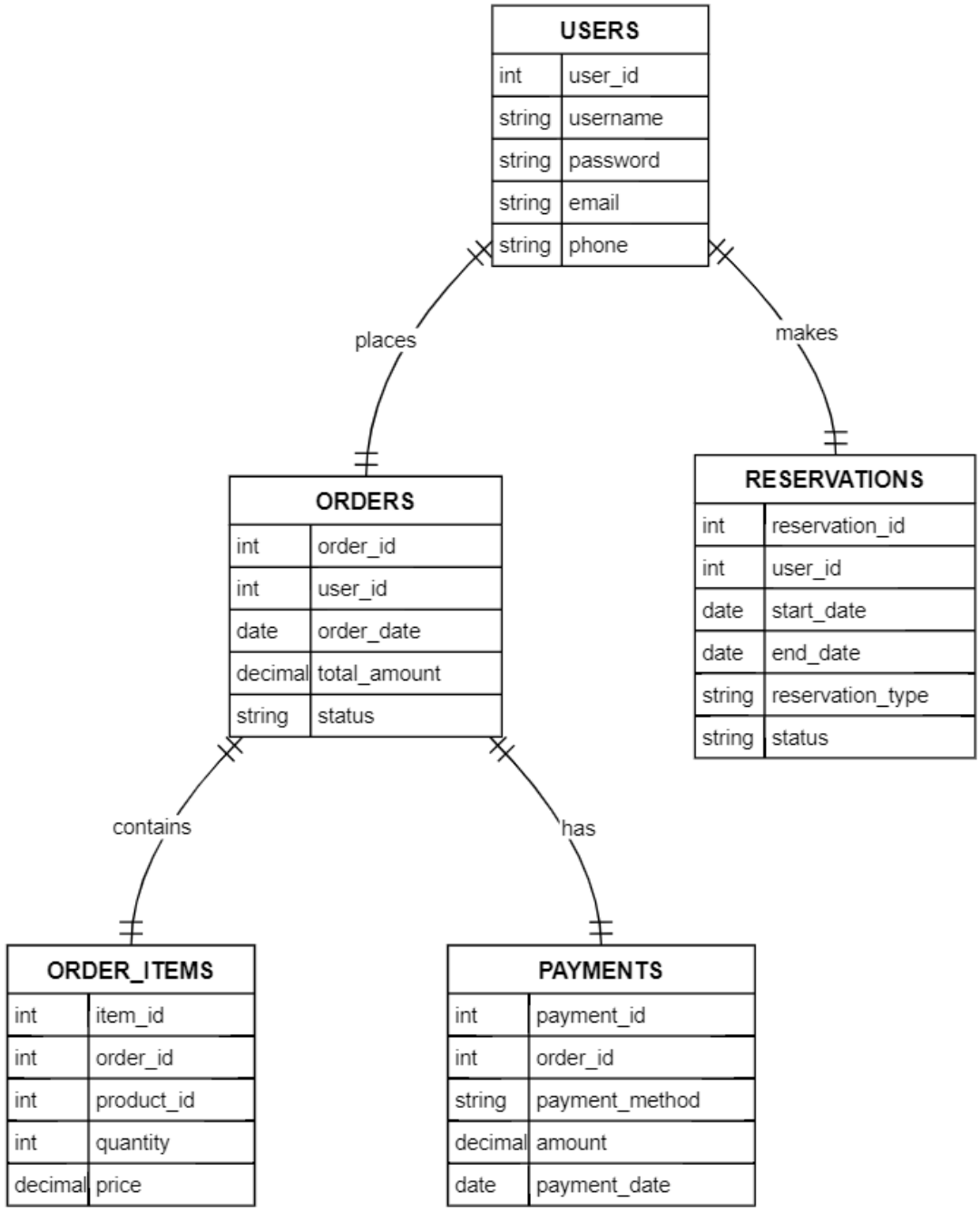


Рисунок 2.3 – UML–діаграма реляційної бази даних PostgreSQL захищеного Telegram–бота оброблення замовлень та резервувань

- ORDERS – зберігає інформацію про замовлення, включаючи ідентифікатор замовлення, ідентифікатор користувача, який зробив замовлення, дату замовлення, загальну суму та статус замовлення.
- ORDER_ITEMS – зберігає інформацію про окремі товари в замовленні, включаючи ідентифікатор товару, кількість та ціну.

- PAYMENTS – зберігає інформацію про платежі, пов'язані з замовленнями, включаючи ідентифікатор платежу, ідентифікатор замовлення, спосіб оплати, суму та дату платежу.

- RESERVATIONS – зберігає інформацію про резервування, включаючи ідентифікатор резервування, ідентифікатор користувача, який зробив резервування, дати початку та завершення, тип резервування та статус.

Нереляційна база даних MongoDB містить дві сутності (рис. 2.4):

– EVENT_LOGS: Зберігає журнали подій, включаючи унікальний ідентифікатор запису, часову позначку, ідентифікатор користувача, тип події та дані події у форматі JSON.

– USER_SESSIONS: Зберігає інформацію про сесії користувачів, включаючи унікальний ідентифікатор запису, ідентифікатор користувача, часові позначки початку та завершення сесії, а також дані сесії у форматі JSON.

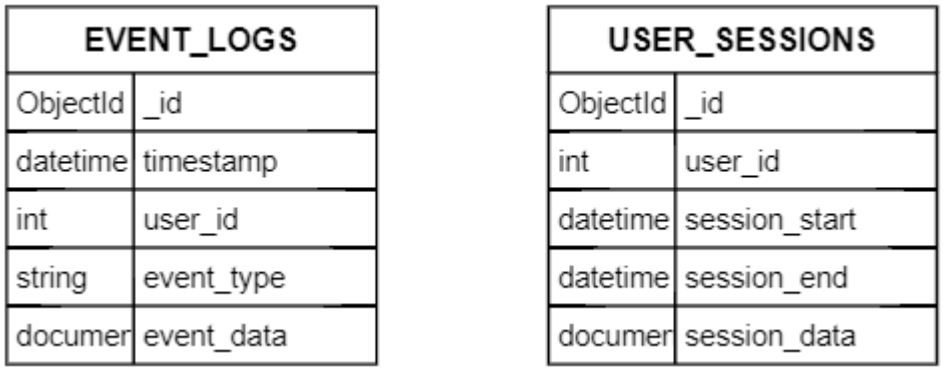


Рисунок 2.4 – UML–діаграма Нереляційна база даних MongoDB захищеного Telegram–бота оброблення замовлень та резервувань

Загалом, використання комбінації реляційної PostgreSQL та нереляційної MongoDB бази даних забезпечить оптимальне зберігання та обробку різних типів даних, що виникають під час роботи захищеного Telegram–бота для обробки замовлень та резервувань.

Це дозволить поєднати переваги кожного типу бази даних, такі як цілісність даних, безпека, продуктивність та масштабованість, що є критично важливим для забезпечення ефективної та надійної роботи розроблюваного рішення.

2.3 Проектування компонентів архітектури захищеного Telegram–бота оброблення замовлень та резервувань

Проектування архітектури бота для оброблення замовлень та резервувань у Telegram є фундаментальним етапом, що визначає якість і ефективність майбутнього рішення. Цей процес включає вибір технологічного стеку і мови програмування, розробку структури та архітектури бота, а також проектування інтерфейсу користувача [45]. Детально розглянемо кожен з цих аспектів.

Структура і архітектура захищеного Telegram–бота для оброблення замовлень та резервувань є ключовими аспектами, які забезпечують його ефективну роботу, масштабованість та безпеку [46]. Розглянемо детальніше, як може бути організована така архітектура, виходячи з концепції мікросервісів (рис. 2.5).

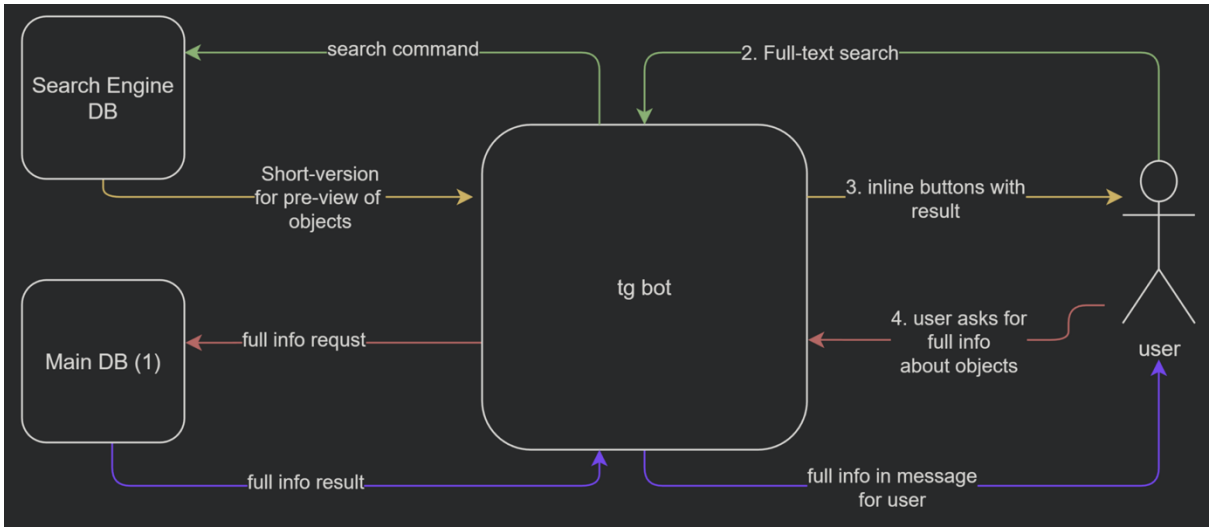


Рисунок 2.5 – Архітектура компонентів захищеного Telegram–бота

Архітектура бота будується навколо кількох ключових компонентів, кожен з яких відповідає за свій сегмент функціональності:

Інтерфейс користувача (UI): відповідає за взаємодію з користувачем через чат–бота у Telegram. Цей компонент управляє прийомом команд від користувачів, відображенням меню, форм і відповідей.

Модуль обробки замовлень: процесує замовлення від користувачів, включаючи їх валідацію, зберігання в базі даних і обробку платежів, якщо це необхідно.

Модуль аутентифікації та авторизації: забезпечує встановлення ідентифікатора користувача та перевірку його прав на здійснення певних дій у системі.

Сервіс взаємодії з Telegram API: централізовано управляє всіма запитами до Telegram API, що дозволяє іншим компонентам системи комунікувати з користувачами без необхідності безпосереднього звернення до API [47].

База даних: центральне сховище для зберігання всієї інформації, необхідної для роботи бота, включаючи дані користувачів, замовлень і транзакцій.

Використання мікросервісної архітектури надає кілька значущих переваг для проекту:

Масштабованість: кожен мікросервіс може бути масштабований незалежно від інших, що дозволяє ефективно управляти ресурсами системи та забезпечувати високу доступність сервісів.

Гнучкість розробки: розділення функціоналу на окремі сервіси дозволяє командам розробників працювати над різними аспектами системи паралельно, спрощуючи процес розробки та тестування.

Висока доступність: у разі збою одного мікросервісу, інші компоненти системи продовжують працювати, що знижує ризики виникнення повної недоступності системи.

Безпека: розділення відповідальності між сервісами дозволяє точніше налаштувати політики безпеки для кожного з них, забезпечуючи більш ефективний захист від зовнішніх та внутрішніх загроз.

Структура і архітектура захищеного Telegram–бота, організовані за мікросервісним принципом, забезпечують високу масштабованість, доступність та безпеку системи. Це створює міцну основу для ефективної та надійної роботи бота, здатного задовольнити вимоги сучасних користувачів і бізнесу. Проектування інтерфейсу користувача для Telegram–бота, що спеціалізується на обробленні замовлень та резервуваннях, є важливим етапом, який вимагає детального підходу для створення інтуїтивно зрозумілого та легкого у використанні інтерфейсу.

Завдання ускладнюється обмеженими можливостями візуалізації у месенджерах, що вимагає від розробників знаходження креативних рішень для ефективної взаємодії з користувачем. Візуалізація основних компонентів мікросервісної архітектури захищеного Telegram–бота наведено у вигляді схеми (рис.2.6).

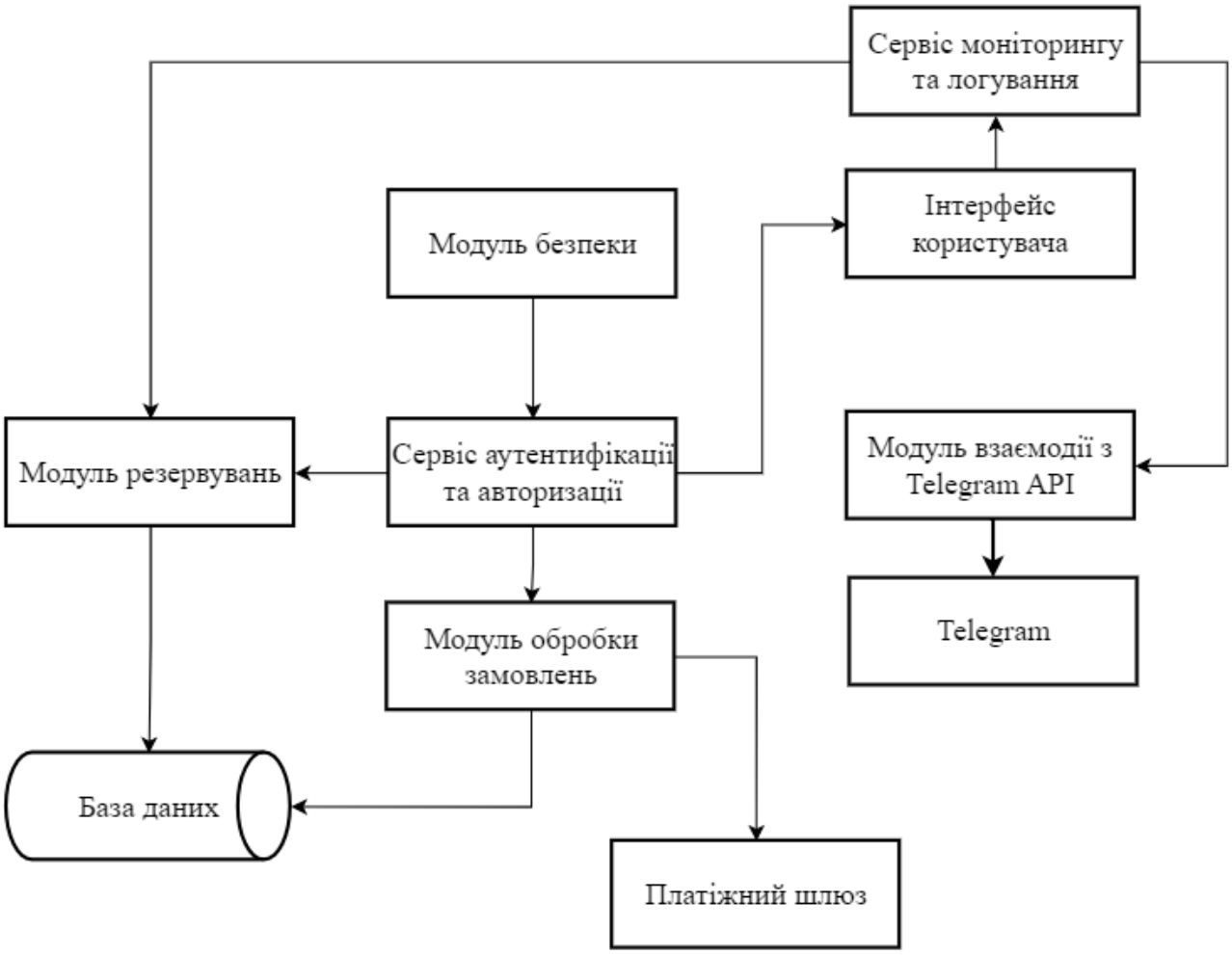


Рисунок 2.6 – Схема мікросервісної архітектури компонентів розробленого захищеного Telegram–бота

Інтерфейс користувача забезпечує взаємодію з користувачами через Telegram, отримує команди, відображає меню та форми. Інтерфейс користувача є головним компонентом, через який користувачі взаємодіють з ботом. Він відповідає за відображення меню, форм та інструкцій, а також отримує команди та дані від користувачів для подальшої обробки.

Сервіс аутентифікації та авторизації відповідає за встановлення ідентифікатора користувача та перевірку його прав доступу. Цей сервіс забезпечує процеси аутентифікації (встановлення достовірності користувачів) та авторизації (надання прав доступу) в системі. Він перевіряє ідентифікатори користувачів та їхні повноваження для доступу до певних функцій або даних бота [48].

Модуль обробки замовлень – цей ключовий компонент відповідає за весь процес обробки замовлень від користувачів. Він отримує дані замовлення від інтерфейсу користувача, валідує їх, здійснює необхідні перевірки та взаємодіє з базою даних для зберігання інформації про замовлення. Якщо замовлення вимагає оплату, модуль взаємодіє з платіжним шлюзом для безпечної обробки платежів.

Модуль резервування обробляє запити на резервування від користувачів та взаємодіє з базою даних [49]. Цей компонент призначений для обробки запитів на резервування від користувачів. Він отримує дані про резервування від інтерфейсу користувача, перевіряє їх коректність та наявність вільних ресурсів, а також взаємодіє з базою даних для зберігання інформації про резервування.

Модуль взаємодії з Telegram API централізовано управляє запитами до Telegram API. Цей модуль забезпечує зв'язок між компонентами системи та офіційним Telegram API. Він централізовано управляє всіма запитами до Telegram API, дозволяючи іншим компонентам взаємодіяти з користувачами через Telegram, не маючи прямого доступу до API [50].

Бази даних виконують роль центрального сховища для зберігання даних користувачів, замовлень, резервувань та транзакцій. БД є центральним сховищем для зберігання всієї інформації, необхідної для роботи бота, включаючи дані користувачів, замовлень, резервувань та транзакцій. База даних повинна забезпечувати високий рівень безпеки та цілісності даних за допомогою шифрування, контролю доступу та резервного копіювання.

Платіжний шлюз – використовується для безпечної обробки платежів під час замовлень. Платіжний шлюз використовується для безпечної обробки платежів під час оформлення замовлень. Модуль обробки замовлень взаємодіє з платіжним шлюзом для проведення транзакцій та отримання підтверджень про оплату.

Сервіс моніторингу та логування – цей компонент відповідає за відстеження та реєстрування діяльності інших компонентів системи. Він здійснює моніторинг та збирає журнали (логи) для виявлення потенційних проблем або загроз безпеці. Зібрані дані можуть використовуватися для аналізу та вдосконалення роботи бота.

Модуль безпеки – відповідає за виявлення та запобігання загроз безпеці, здійснює моніторинг та реагування на інциденти. Модуль безпеки відіграє ключову роль у забезпеченні захисту системи. Він відповідає за виявлення та запобігання загрозам безпеці, таким як хакерські атаки, шкідливе програмне забезпечення чи спроби несанкціонованого доступу. Цей модуль здійснює моніторинг активності та реагує на виявлені інциденти, вживаючи відповідних заходів безпеки.

Загалом, ця схема демонструє, як різні компоненти взаємодіють між собою для забезпечення безпечної та ефективної роботи захищеного Telegram–бота для обробки замовлень та резервувань.

Ця схема архітектури демонструє взаємозв'язок між різними компонентами захищеного Telegram–бота, їхні ролі та відповідальності. Вона візуалізує потоки даних та процеси, що відбуваються під час обробки замовлень та резервувань, враховуючи аспекти безпеки, аутентифікації, моніторингу та взаємодії з зовнішніми системами, такими як Telegram API та платіжний шлюз.

2.4 Висновки до розділу 2

У даному розділі було детально розглянуто процес проектування та розроблення підходу до створення захищеного Telegram–бота для обробки замовлень та резервувань. Цей процес включав кілька ключових етапів та прийнятих рішень.

По–перше, було обґрунтовано необхідність розробки такого захищеного Telegram–бота з точки зору зручності взаємодії з клієнтами, забезпечення належного рівня безпеки даних та подолання недоліків існуючих рішень. Представлено блок–схему алгоритму роботи бота, що демонструє покроковий процес обробки замовлень та резервувань з впровадженням заходів безпеки.

Далі було проаналізовано та обґрунтовано вибір реляційної бази даних PostgreSQL для зберігання структурованих даних замовлень та резервувань, а також нереляційної бази даних MongoDB для обробки неструктурованих даних, таких як журнали подій та сесії користувачів. Для наочності були представлені UML–діаграми, що візуалізують структуру та взаємозв'язки між сутностями в базах даних.

Ключовим етапом став процес проектування компонентів архітектури захищеного Telegram–бота. Було обрано мікросервісний підхід, що забезпечує гнучкість, масштабованість та високу доступність системи. Детально розглянуто роль та функціональність кожного компонента, включаючи інтерфейс користувача, модулі обробки замовлень та резервувань, сервіси аутентифікації, взаємодії з Telegram API та базу даних. Запропоновано схему архітектури у вигляді блок–схеми, що демонструє взаємозв'язки між компонентами.

Також було представлено конкретний підхід до забезпечення безпеки Telegram–бота, що включає фільтрацію трафіку, використання вебхуків, шифрування даних, аутентифікацію користувачів та обмеження доступу до бази даних. Ці заходи забезпечують комплексний захист від широкого спектру потенційних загроз, таких як перехоплення трафіку, хакерські атаки чи витоки даних.

Загалом, було ретельно спроектовану архітектуру захищеного Telegram–бота, яка поєднує функціональність, безпеку, масштабованість та зручність використання. Прийняті рішення та підходи забезпечують міцну основу для подальшої реалізації та впровадження рішення, що дозволить компаніям надавати своїм клієнтам зручний, безпечний та сучасний спосіб обробки замовлень та резервувань через месенджер Telegram.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАХИЩЕНОГО TELEGRAM–БОТА ОБРОБЛЕННЯ ЗАМОВЛЕНЬ ТА РЕЗЕРВУВАНЬ

3.1 Обгрунтування мови програмування та середовища розробки захищеного Telegram–бота

Вибір мови програмування та середовища розробки є критичним етапом у процесі створення будь–якого програмного забезпечення, включаючи захищений Telegram–бот для обробки замовлень та резервувань. Правильний вибір технологій може значно вплинути на продуктивність, масштабованість, безпеку та загальну ефективність розробленого рішення. На ринку існує широкий спектр мов програмування та фреймворків, кожен з яких має свої переваги та недоліки, тому необхідно ретельно розглянути всі можливі варіанти перед прийняттям рішення.

Однією з найпопулярніших мов програмування для розробки веб–додатків та ботів є Python [51]. Ця мова відрізняється своєю зрозумілістю, читабельністю та простотою вивчення, що робить її привабливою як для досвідчених розробників, так і для новачків. Python має потужні бібліотеки та модулі, що охоплюють широкий спектр задач, від веб–розробки до обробки природної мови та машинного навчання. Ще однією перевагою Python є його крос–платформеність, що дозволяє розробляти програми, які можуть працювати на різних операційних системах без необхідності значних модифікацій коду.

У контексті розробки захищеного Telegram–бота мова Python пропонує кілька потужних фреймворків, серед яких варто виділити Django та Flask. Django – це високорівневий фреймворк із багатим набором функцій та інструментів для створення веб–додатків. Він забезпечує зручну структуру проекту, вбудовану систему адміністрування, надійну систему безпеки та підтримку ORM (Object–Relational Mapping) для взаємодії з базами даних. З іншого боку, Flask є більш мінімалістичним фреймворком, який пропонує гнучкість та свободу у виборі інструментів та бібліотек. Він відрізняється своєю легкістю та швидкістю, що

робить його ідеальним вибором для створення невеликих та простих додатків або API [52].

Однак, Python не є єдиною мовою програмування, яка може бути використана для розробки Telegram–ботів. Існують і інші популярні мови, такі як JavaScript (Node.js), Java, C#, Ruby та інші. Кожна з цих мов має свої переваги та недоліки, тому варто розглянути їх детальніше та порівняти з Python (табл 3.1) [53].

Таблиця 3.1 – Порівняння мов програмування та фреймворків для розробки захищеного Telegram–бота.

Критерій	Python (Django, Flask)	Node.js (Express.js)	Java (Spring)	C# (.NET)
Продуктивність	Середня	Висока	Висока	Висока
Безпека	Висока	Середня	Висока	Висока
Підтримка бібліотек	Багата	Багата	Багата	Багата
Зрозумілість коду	Висока	Середня	Середня	Середня
Крос–платформеність	Висока	Висока	Середня	Середня
Вивчення та документація	Проста	Середня	Середня	Складна
Масштабованість	Середня	Висока	Висока	Висока
Співтовариство	Велике	Велике	Велике	Велике

Як видно з таблиці, Python у поєднанні з фреймворками Django та Flask пропонує збалансований набір переваг, що робить його привабливим вибором для розробки захищеного Telegram–бота. Хоча Node.js, Java та C# можуть забезпечити вищу продуктивність та масштабованість, Python відрізняється своєю простотою вивчення, читабельністю коду та потужними бібліотеками, що охоплюють різноманітні аспекти розробки, включаючи безпеку та обробку природної мови.

Крім того, Python має велику та активну спільноту розробників, що забезпечує постійну підтримку, оновлення та розширення функціоналу мови та її

бібліотек. Це є важливим фактором для довгострокової підтримки та розвитку програмного забезпечення, такого як захищений Telegram–бот.

Вибір Django або Flask як фреймворку для розробки Telegram–бота залежить від конкретних вимог проекту. Django забезпечує багатий функціонал та зручну структуру проекту, що може бути корисним для складних додатків з багатьма функціями та модулями. Однак, для більш простих додатків або API Flask може бути кращим вибором завдяки своїй легкості та гнучкості.

Тому вибір Python разом із Django та Flask забезпечує баланс між швидкістю розробки, функціональністю та безпекою. Це дозволяє створювати потужні, гнучкі та надійні боти для месенджерів, що відповідають сучасним вимогам до продуктивності та безпеки.

Підсумовуючи, з огляду на необхідність забезпечення безпеки, простоту розробки та наявність потужних бібліотек для взаємодії з Telegram API, обробки природної мови та захисту даних, мова програмування Python у поєднанні з фреймворком Django або Flask є оптимальним вибором для розробки захищеного Telegram–бота оброблення замовлень та резервувань. Цей вибір забезпечить ефективну реалізацію проекту, зручність розробки та підтримки, а також необхідний рівень безпеки та продуктивності для задоволення вимог користувачів.

3.2 Опис програмної реалізації захищеного Telegram–бота

Програмна реалізація захищеного Telegram–бота базується на модульній структурі, де кожен компонент відповідає за певну функціональність або аспект безпеки. Розглянемо детально основні модулі та їх роль у забезпеченні безпеки та функціональності бота. Наступний код відповідає за налаштування основних параметрів бота та забезпечення безпеки:

```
BOT_TOKEN = 'c7730b824:AAFq0bXak j 8oRvMDTv3Bc 07dRP6ul 1 w80l/X'
PROJECT_NAME = 'store–bot–example'

WEBHOOK_HOST = f"https://{PROJECT_NAME}.herokuapp.com"
WEBHOOK_PATH = '/webhook/' + BOT_TOKEN
WEBHOOK_URL = f'{WEBHOOK_HOST}{WEBHOOK_PATH}'
```

```
ADMINS = [000000000, 1234567890]
```

– BOT_TOKEN: це унікальний ідентифікатор бота, який використовується для авторизації та взаємодії з API Телеграм. Без правильного токена бот не зможе приймати команди від користувачів чи відповідати на них.

– PROJECT_NAME: назва проекту відіграє роль у формуванні адреси вебхуку, яка є необхідною для отримання оновлень від Телеграм через зовнішній сервер.

– WEBHOOK_HOST, WEBHOOK_PATH, WEBHOOK_URL: ці параметри визначають адресу, на яку Телеграм буде надсилати оновлення про нові повідомлення чи команди від користувачів. WEBHOOK_HOST комбінується з WEBHOOK_PATH для створення повної WEBHOOK_URL, яка використовується в налаштуваннях бота.

– ADMINS: список ідентифікаторів адміністраторів бота, які мають привілейований доступ до управління ботом та обробки замовлень.

Кожен з цих параметрів виконує свою функцію та є вирішальним для коректної роботи та безпеки бота. Важливість цього модулю полягає у тому, що він інкапсулює ключові параметри, які можуть бути імпортовані та використані в інших частинах системи.

Цей код відповідає за перевірку, чи є користувач адміністратором бота:

```
from aiogram.types import Message
from aiogram.dispatcher.filters import BoundFilter
from data.config import ADMINS

class IsAdmin(BoundFilter):
    async def check(self, message: Message):
        return message.from_user.id in ADMINS
```

Цей код імпортує необхідні класи та функції з бібліотеки aiogram та модуля config. Клас IsAdmin наслідує BoundFilter та перевіряє, чи ідентифікатор користувача міститься у списку ADMINS. Це дозволяє обмежити доступ до адміністративних функцій бота лише для авторизованих користувачів.

Цей фрагмент коду відповідає за обробку замовлень та взаємодію з базою даних:

```
from aiogram.types import Message
from loader import dp, db
from handlers.user.menu import orders
from filters import IsAdmin

@dp.message_handler(IsAdmin(), text='orders')
async def process_orders(message: Message):
   orders = db.fetchall('SELECT * FROM orders')

   if len(orders) == 0: await message.answer('You have no orders.')
   else: await order_answer(message, orders)

async def order_answer(message, orders):
   res = ''
   for order in orders:
      res += f'Order <b>{order[3]}</b>\n\n'
   await message.answer(res)
```

Цей код імпортує необхідні модулі та фільтри. Функція process_orders викликається за командою orders від адміністраторів і отримує всі замовлення з бази даних за допомогою методу fetchall. Якщо замовлень немає, бот повідомляє про це, інакше викликається функція order_answer, яка форматує та відправляє повідомлення з деталями замовлень. Ці фрагменти коду демонструють, як модульна структура бота забезпечує розділення відповідальності між різними компонентами, такими як конфігурація, фільтрація, обробка замовлень та взаємодія з базою даних.

```
from aiogram.types import ReplyKeyboardMarkup

def query(self, arg, value=None):
   if value is None:
      self.core.decode(arg)
   else:
      self.core.decode(arg, value)
   return self.core.fetchone()
```

Цей код виконує отримання даних з бази даних. Функція query приймає аргумент arg, який ймовірно містить SQL–запит, та необов'язковий аргумент value. Функція decode з модуля utils.helper виконує допоміжну роль для декодування бінарних даних з бази даних.

```
headers = {
   "Content–Type": "application/octet–stream",
   "Accept": "application/octet–stream"
}
```

Ці заголовки використовуються для передачі бінарних даних через протокол HTTPS. Вони вказують на те, що дані повинні бути інтерпретовані як бінарні, а не як текстові рядки. Наступний код відповідає за відстеження статусу доставки замовлень користувачів.

```
from aiogram.types import Message
from loader import dp, db
from .menu import delivery_status
from filters import IsUser

@dp.message_handler(IsUser(), text=delivery_status)
async def process_delivery_status(message: Message):
   orders = db.fetchall('SELECT * FROM orders WHERE c_id=?', (message.chat.id,))

   if len(orders) == 0: await message.answer('You have no active orders.')
   else: await delivery_status_answer(message, orders)

async def delivery_status_answer(message, orders):
   res = ''

   for order in orders:
      res += f'Order <b>{order[3]}</b>'
      answer = [
         'is in stock',
         'on the road',
         'arrived and is waiting for you at the post office!'
      ]

      res += answer[order[4]]
      res += '\n\n'

   await message.answer(res)
```

Спочатку імпортуються необхідні модулі та фільтр IsUser, який дозволяє обробляти повідомлення лише від користувачів, а не адміністраторів. Функція process_delivery_status викликається, коли користувач надсилає повідомлення "delivery_status". Вона отримує всі замовлення користувача з бази даних за допомогою виразу db.fetchall('SELECT * FROM orders WHERE c_id=?', (message.chat.id,)).

Якщо у користувача немає активних замовлень, бот повідомляє про це. Інакше викликається функція delivery_status_answer з переліком замовлень.

```
from aiogram.dispatcher.filters.state import StatesGroup, State

class CheckoutState(StatesGroup):
    check_cart = State()
    name = State()
    address = State()
    confirm = State()
```

Цей код визначає стани checkout (оформлення замовлення) для телеграм–бота. StatesGroup – це допоміжний клас із бібліотеки AIOGram, який дозволяє організувати процес замовлення як скінченний автомат.

CheckoutState успадковує від StatesGroup і визначає 4 стани: check_cart (перевірка кошика), name (введення імені), address (введення адреси) та confirm (підтвердження замовлення). Кожен стан описується як окремий об'єкт State().

```
import logging
from aiogram import Bot, Dispatcher, executor
from handlers import dp
from config import BOT_TOKEN, USE_WEBHOOK, WEBHOOK_PATH, WEBAPP_HOST, WEBAPP_PORT
from storage import create_tables
```

Імпорт необхідних модулів та налаштування логування: Спочатку імпортуються необхідні модулі з бібліотек aiogram та власних модулів проекту. Також встановлюється рівень логування.

```
logging.basicConfig(level=logging.INFO)

bot = Bot(token=BOT_TOKEN)
dp = Dispatcher(bot)
```

Код вище выдповыдаэ за створення об'єктыв Bot та Dispatcher з використанням токена бота та підключаються обробники повідомлень.

Залежно від значення змінної USE_WEBHOOK, бот запускається у режимі вебхука або полінгу. При запуску вебхука визначаються функції on_startup та on_shutdown, які виконуються при старті та зупинці роботи бота відповідно.

```
from handlers import register_handlers
register_handlers(dp)

if USE_WEBHOOK:
   executor.start_webhook(
      dispatcher=dp,
      webhook_path=WEBHOOK_PATH,
      on_startup=on_startup,
      on_shutdown=on_shutdown,
      host=WEBAPP_HOST,
      port=WEBAPP_PORT,
   )
else:
   executor.start_polling(dp, on_startup=on_startup)
```

Фрагмент коду нижче це функція виконується перед запуском бота. Тут створюються необхідні таблиці у базі даних за допомогою функції create_tables() з модуля storage.py. Також встановлюється вебхук для бота на вказаному URL.

```
async def on_startup(dp):
   await create_tables()
   await bot.set_webhook(WEBHOOK_URL)

async def on_shutdown(dp):
   logging.warning('Shutting down..')
   await bot.delete_webhook()
   await dp.storage.close()
   await dp.storage.wait_closed()
   logging.warning('Bye!')
```

Таким чином вище описаний код компонентів, є центральною точкою входу для запуску телеграм–бота. Він відповідає за налаштування логування, ініціалізацію бота та диспетчера, підключення обробників повідомлень, запуск бота у режимі вебхука або полінгу, а також за налаштування вебхука, створення таблиць у базі даних та коректне вивільнення ресурсів при зупинці бота.

3.3 Інструкція користувачу захищеного Telegram–бота оброблення замовлень та резервувань

Щоб почати роботу із захищеним Telegram–ботом для обробки замовлень та резервувань, необхідно відкрити чат з ботом у месенджері Telegram. На початку бесіди бот вітає користувача і пояснює, що він є ботом–магазином для постачання

товарів різних категорій. Бот також надає інструкції щодо використання меню команд для навігації та вибору потрібних опцій, як це ілюстровано на (рис. 3.1).

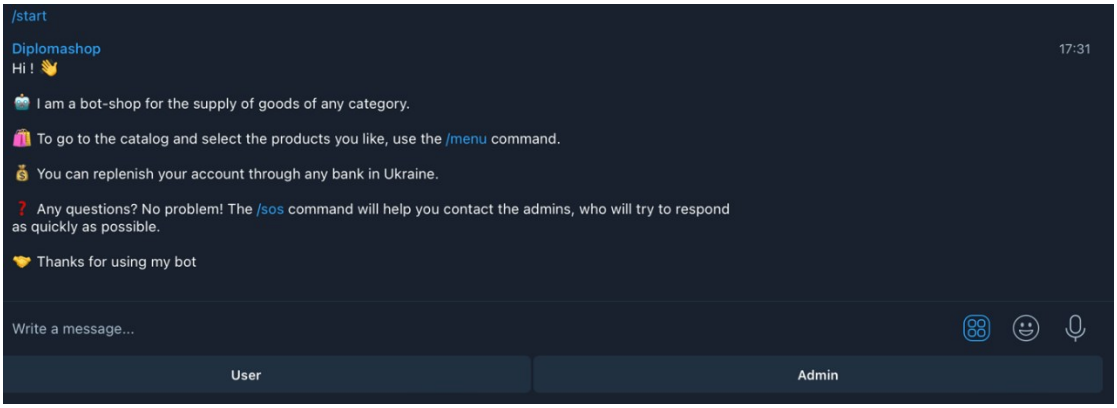


Рисунок 3.1 – Початкове привітання та інструкції вTelegram–бота при запуску чату.

У меню, яке з'являється після привітання, бот пропонує кілька опцій, включаючи перехід до каталогу для вибору товарів, перегляд балансу, кошика замовлень та статусу замовлення. Це дозволяє користувачу легко орієнтуватися та обирати потрібний пункт, як показано на (рис. 3.2).

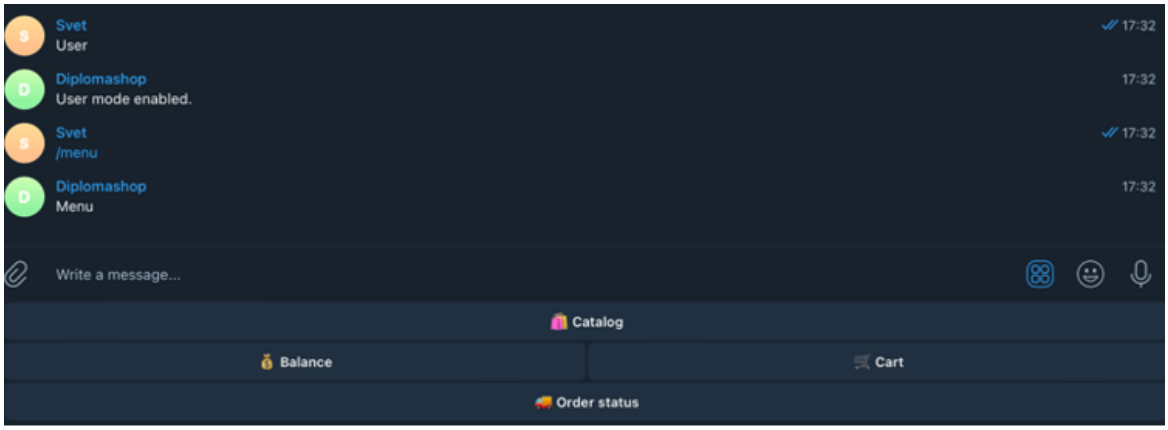


Рисунок 3.2 – Головне меню із опціями каталогу, балансу, кошика та статусу замовлення.

Обравши картинку товару в каталозі, бот відображає її у чаті, дозволяючи користувачу краще роздивитися. Під картинкою наводиться назва товару, його ціна, а також кнопка "Add to Basket" для додавання позиції до кошику покупок при бажанні придбати цей товар. Це забезпечує зручний та візуальний спосіб вибору і замовлення необхідних продуктів, що видно на рис. 3.3.

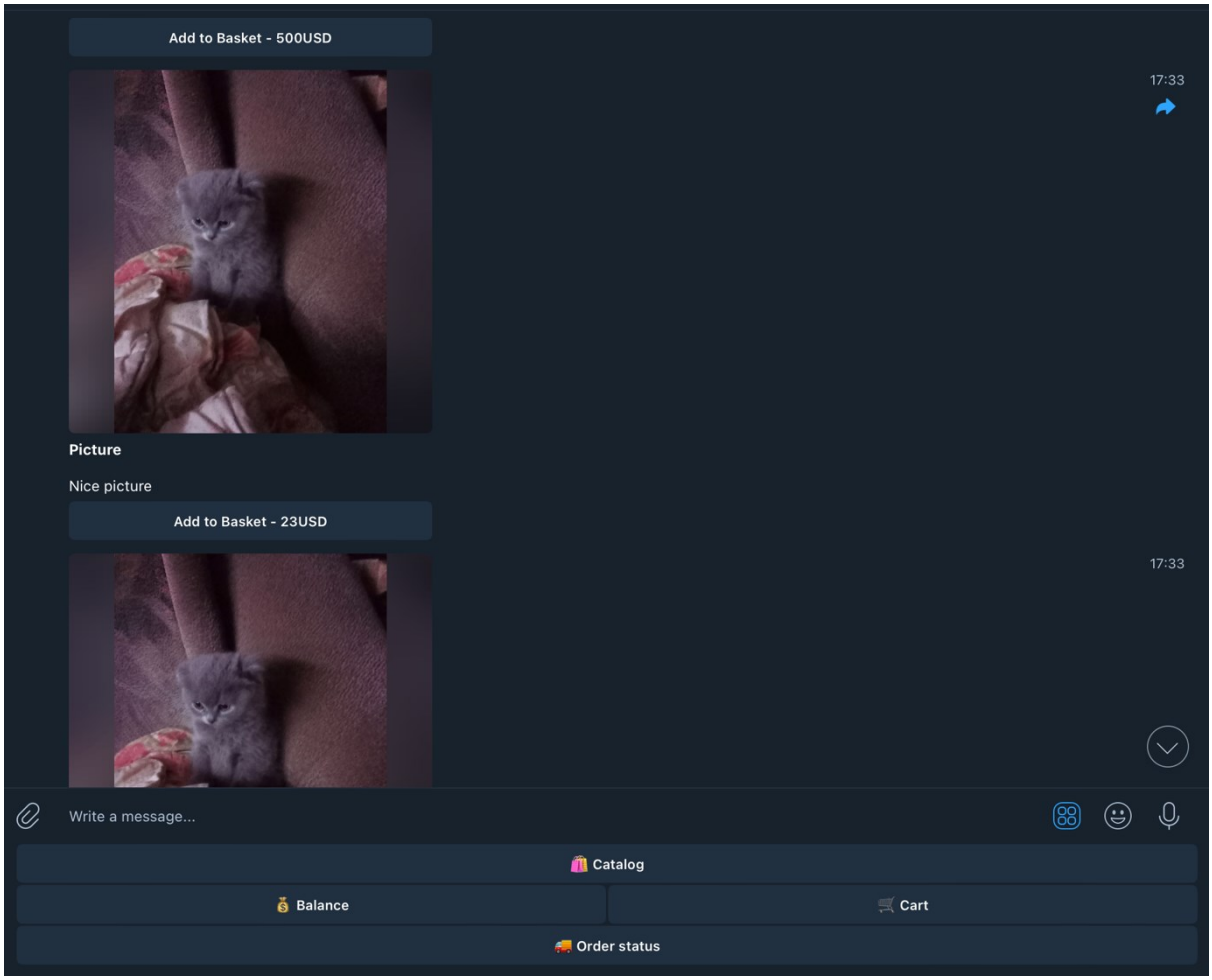


Рисунок 3.3 – Відображення обраного товару з картинкою, назвою, ціною та опцією додавання до кошику.

На наступному кроці, після додавання бажаного товару до кошика, бот відображає вміст кошика, куди потрапила обрана позиція. Показується назва доданого товару, його кількість – у даному випадку 1 шт, та загальна сума замовлення. Далі бот пропонує перейти до оформлення замовлення, натиснувши кнопку "Go to checkout?", як це видно на (рис. 3.4).

Рисунок 3.4 – Вміст кошика замовлення із доданим товаром та опцією переходу до оформлення.

На завершальному етапі оформлення замовлення бот просить користувача ввести його ім'я та адресу доставки у відповідні текстові поля. Після введення інформації бот просить перевірити коректність даних і підтвердити замовлення. У випадку успішного підтвердження, бот повідомляє, що замовлення знаходиться в процесі обробки і буде доставлене найближчим часом. Це фінальний крок процесу замовлення, відображений на (рис. 3.5).

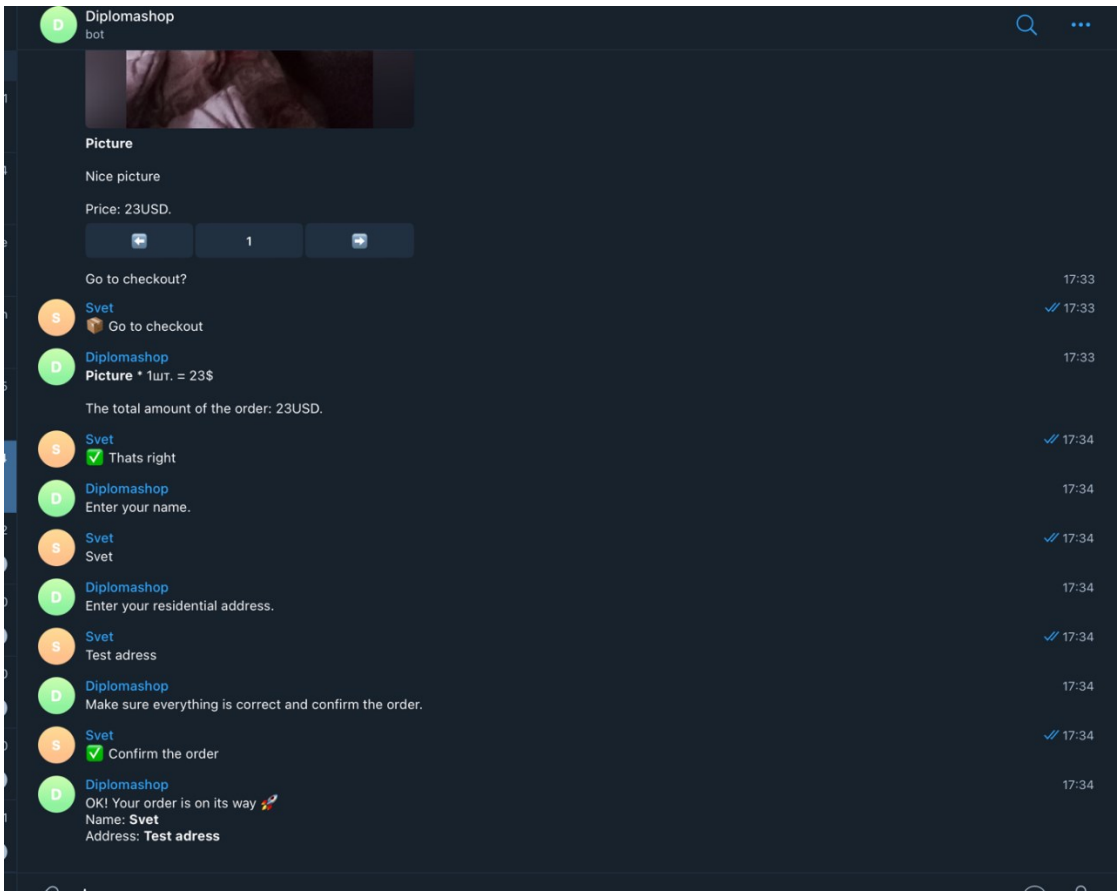


Рисунок 3.5 – Введення даних користувача, підтвердження замовлення та повідомлення про успішне оформлення.

Після успішного оформлення замовлення бот надсилає користувачеві повідомлення з деталями замовлення для остаточної перевірки. Повідомлення містить унікальний номер замовлення, який складається з випадкового набору цифр і букв, наприклад, "N°12038aa58f454cbe62f3de1f0a4de18d=1". Бот також

повідомляє, що вказаний товар є в наявності на складі, про що свідчить фраза "Order N°12038aa58f454cbe62f3de1f0a4de18d=1 is in stock." Це дає користувачу впевненість, що замовлення буде виконано і доставлено найближчим часом, як це видно на (рис. 3.6).

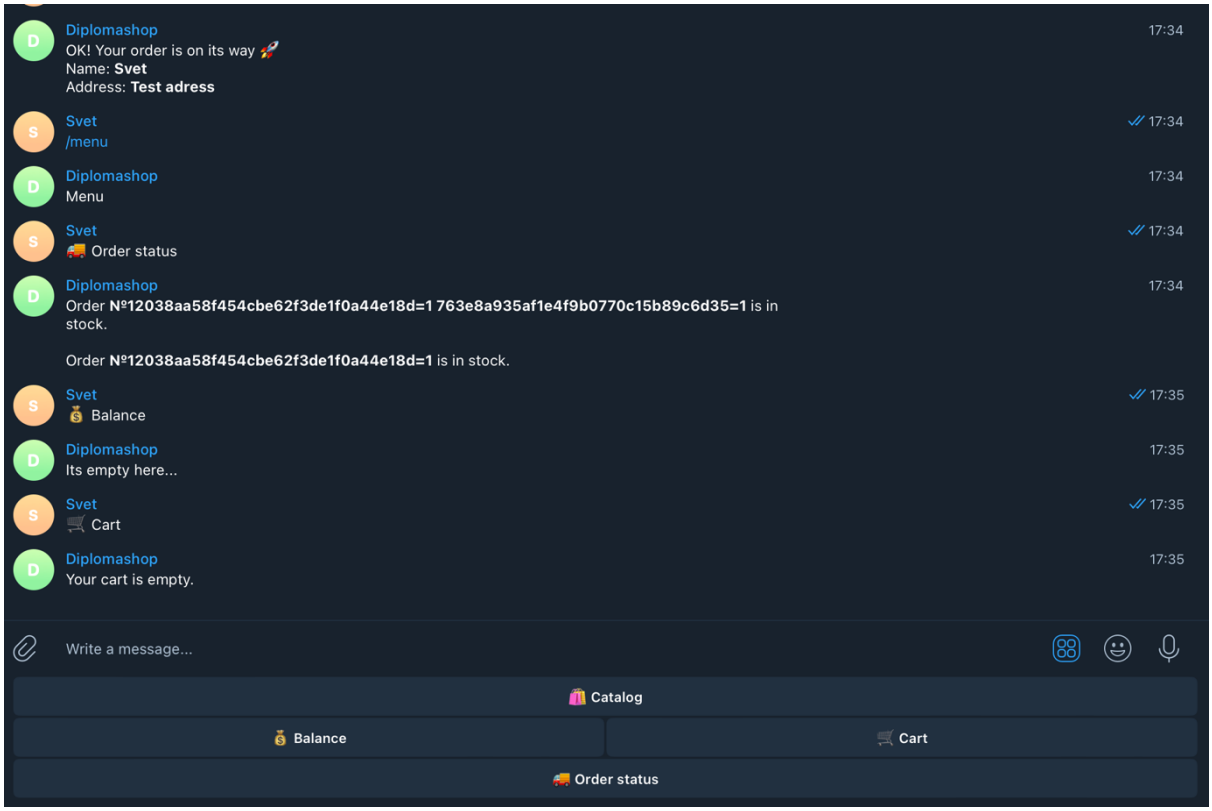


Рисунок 3.6 – Повідомлення з деталями замовлення та підтвердженням наявності товару на складі після оформлення.

Щоб отримати допомогу від адміністратора у випадку проблеми з замовленням, користувач може скористатися опцією "Questions" в меню бота. При виборі цього пункту, бот запитує детальний опис проблеми і просить користувача надати якомога більше подробиць для ефективної допомоги. Це дозволяє адміністратору краще зрозуміти суть проблеми і надати точну відповідь, як це ілюстровано на (рис. 3.7).

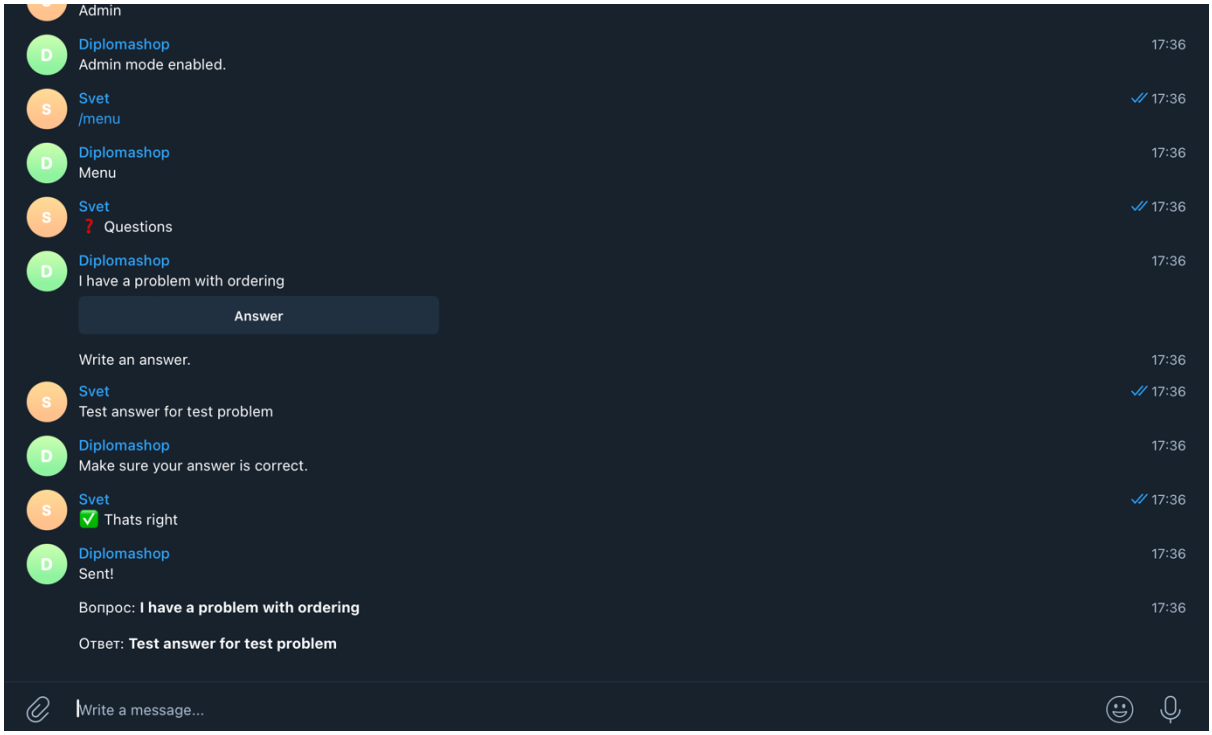


Рисунок 3.7 – Опція "Questions" для звернення до адміністратора з детальним описом проблеми замовлення.

Для тестування коректності роботи бота і його реакції на різні запити, передбачена можливість написати тестове повідомлення через опцію "Test answer for test problem". Після входу до режиму адміністрування захищеного Telegram–бота, користувач потрапляє до меню управління основними налаштуваннями та компонентами. Це меню містить розділи "Catalog setup", "Orders" і "Questions", кожен з яких відповідає за певний аспект роботи бота, як це видно на (рис. 3.8).

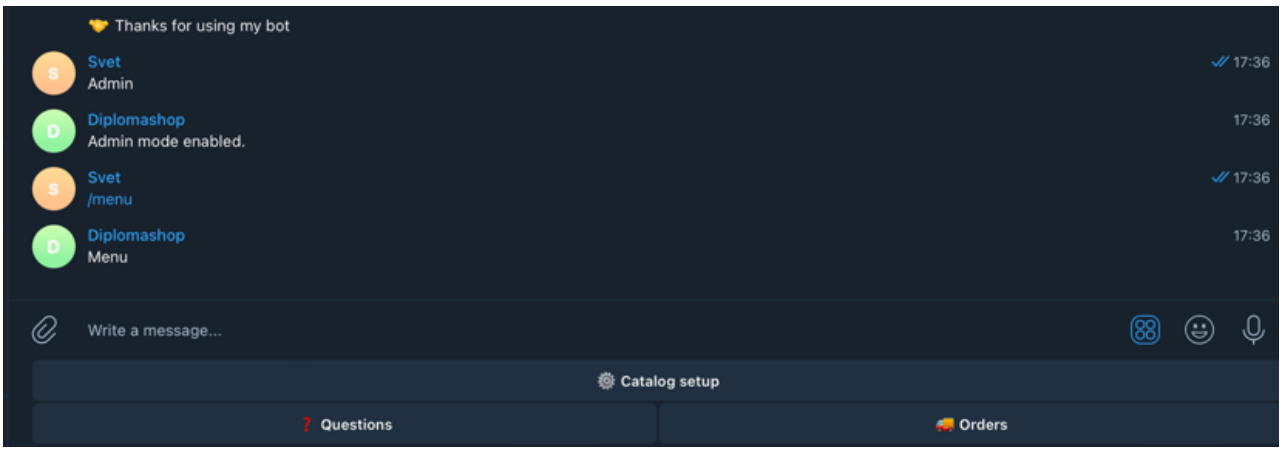


Рисунок 3.8 – Меню адміністрування з розділами налаштування каталогу, замовлень та запитань.

Отримавши відповідь від бота на тестове повідомлення, адміністратор може оцінити, наскільки вона релевантна і чи коректно бот зрозумів суть проблеми. Якщо відповідь влаштовує, адміністратор відправляє боту команду "That's right", підтверджуючи правильність реакції. В іншому випадку, він може продовжити тестування, ввівши "I have a problem with ordering", щоб відправити інший тестовий запит і отримати нову відповідь бота для оцінки. Ця взаємодія адміністратора з ботом зображена на (рис. 3.9).

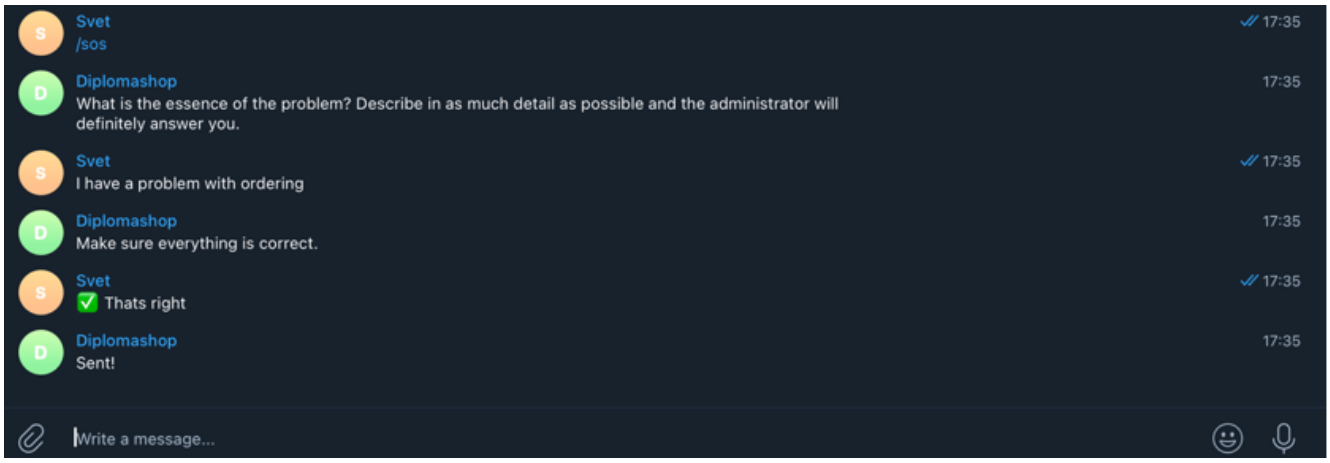


Рисунок 3.9 – Оцінка адміністратором відповіді бота на тестову проблему і подальше тестування при потребі.

Для зручного додавання нових товарів у каталог бот пропонує форму, де потрібно вказати назву товару, завантажити його фотографію та ввести ціну. Ці поля дозволяють детально описати товарну позицію перед її збереженням.

Обравши пункт "Catalog setup", адміністратор переходить до налаштування товарного каталогу. На цьому екрані відображаються наявні категорії товарів, такі як "Paintings" та "Books", а також опція додавання нової категорії "+ Add category".

Наявна клавіатура з цифрами спрощує введення ціни, а кнопка "✓ Thats right" використовується для підтвердження правильності введених даних. У нижній частині форми відображаються поточні категорії каталогу "Paintings" і "Books", а

також опція додавання нової категорії. Весь процес додавання товару з описом його властивостей показаний на (рис. 3.10).

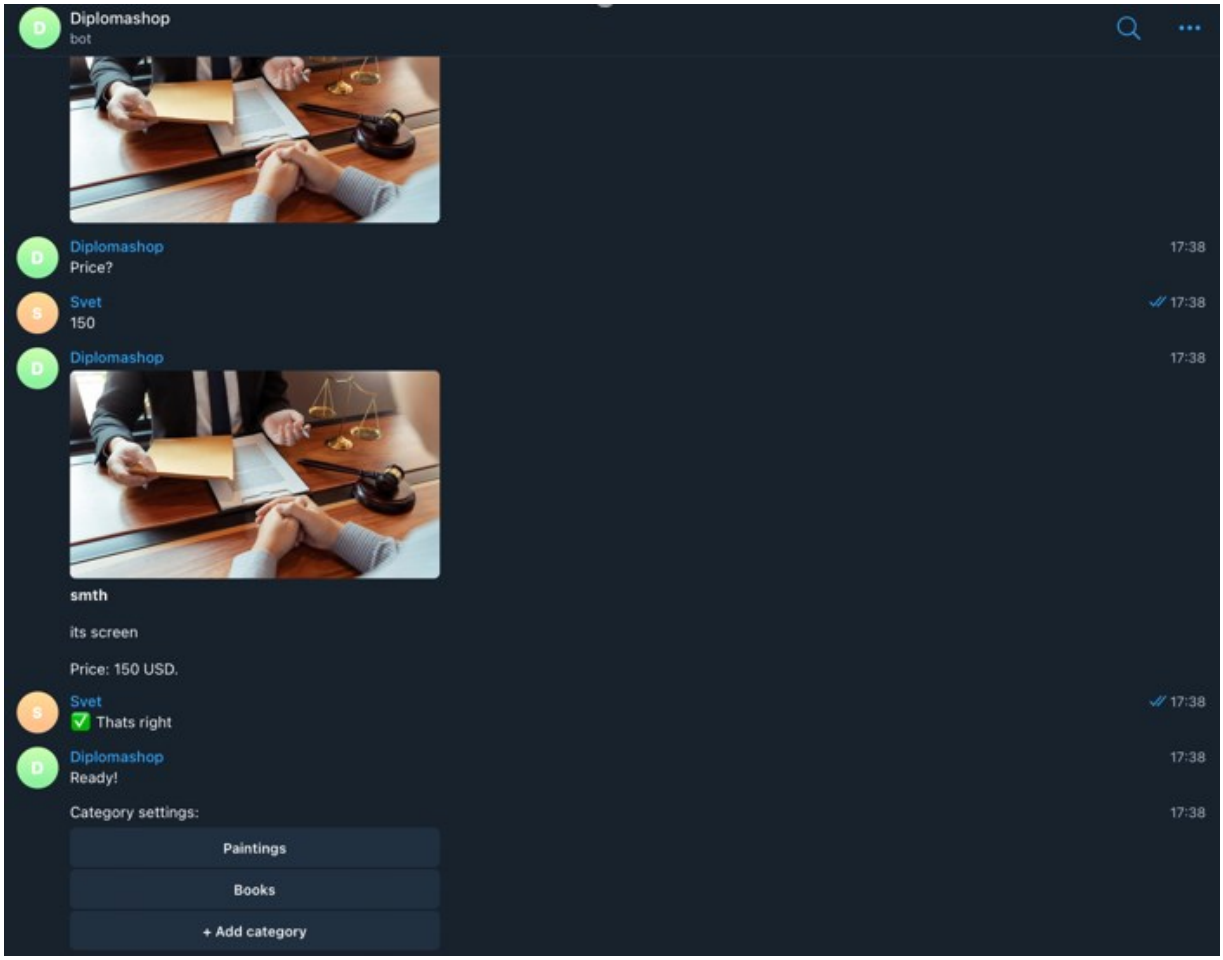


Рисунок 3.10 – Форма додавання нового товару з полями назви, фото, ціни та опцією вибору категорії.

Перейшовши до управління товарами всередині певної категорії, адміністратор бачить список доданих товарних позицій. Для кожного товару відображається його фотографія, назва та ціна. Це дає змогу швидко оцінити наповнення категорії та отримати загальне уявлення про представлені товари. Фотографії допомагають візуально ідентифікувати продукти, а назви та ціни надають ключову інформацію для адміністрування асортименту.

Обравши пункт "Catalog setup", адміністратор переходить до налаштування товарного каталогу. На цьому екрані відображаються наявні категорії товарів, такі як "Paintings" та "Books", а також додавання нової категорії "+ Add category". Це

дозволяє структурувати асортимент та товари за групами, що спрощує навігацію для покупців та управління каталогом для адміністратора. (рис. 3.11).

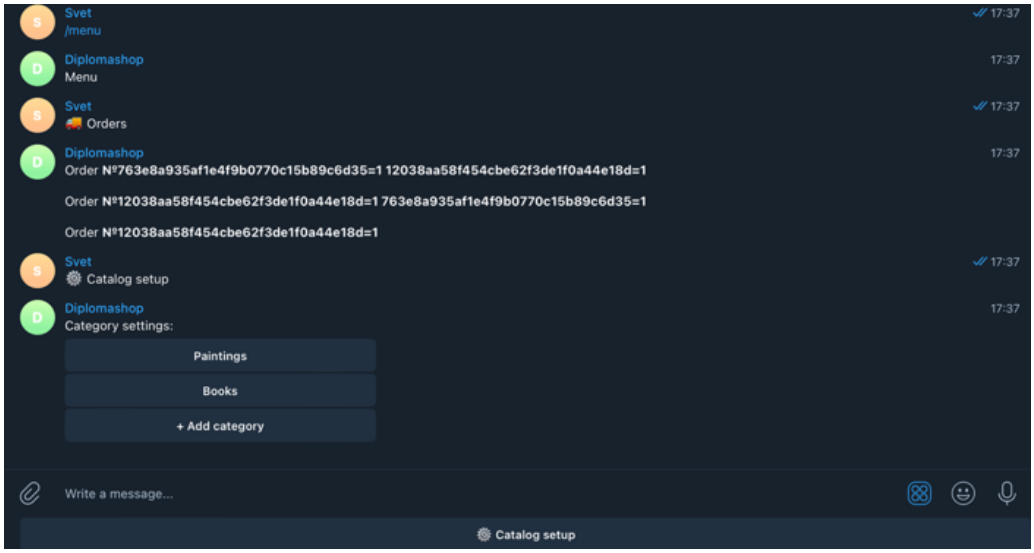


Рисунок 3.11 – Налаштування каталогу товарів з категоріями та опцією додавання нової категорії.

Обравши окремий товар для редагування чи видалення, бот відображає картинку товару, його назву та ціну. Під цією інформацією наявні кнопки "Svet" для редагування та "Delete" для видалення даного товару з каталогу. Це зручний спосіб управління асортиментом та властивостями товарних позицій, що показано на рис. 3.12.

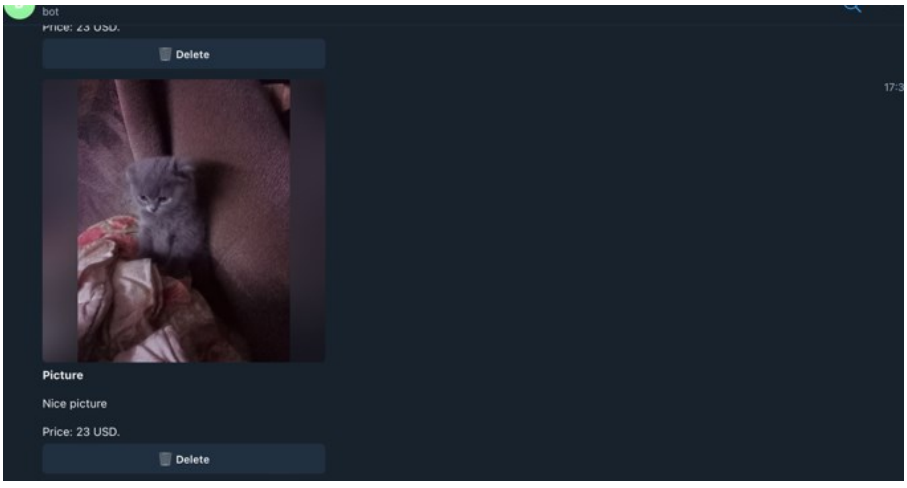


Рисунок 3.12 – Редагування окремого товару з опціями зміни властивостей чи видалення з каталогу.

Отже, захищений Telegram–бот забезпечує зручний та інтуїтивно зрозумілий інтерфейс для вибору товарів, додавання їх до кошика та оформлення замовлення з доставкою. Він супроводжує користувача на кожному кроці, надаючи чіткі інструкції та опції через текстові повідомлення та інтерактивні кнопки. Це дозволяє ефективно та безпечно здійснювати замовлення товарів прямо в месенджері Telegram.

3.4 Висновки до розділу 3

Щоб почати роботу із захищеним Telegram–ботом для обробки замовлень та резервувань, необхідно відкрити чат з ботом у месенджері Telegram. На початку бесіди бот вітає користувача і пояснює, що він є ботом–магазином для постачання товарів різних категорій. Бот також надає інструкції щодо використання меню

У даному підрозділі було представлено детальну інструкцію користувача для роботи із захищеним Telegram-ботом оброблення замовлень та резервувань. Розглянуто основні етапи взаємодії користувача з ботом, від початкового привітання та вибору товарів з каталогу до оформлення замовлення та отримання підтвердження.

Було продемонстровано, як бот надає зручне меню з опціями каталогу, балансу, кошика та статусу замовлення, що спрощує навігацію. Візуальне представлення товарів з картинками, назвами та цінами робить процес вибору інтуїтивним. Додавання товарів до кошика, введення даних користувача та підтвердження замовлення відбувається через чіткі кроки з підказками бота.

Адміністративна частина бота дозволяє налаштовувати каталог товарів, додавати нові категорії та товарні позиції, редагувати їх властивості та видаляти за потреби.

Загалом, інструкція користувача демонструє, що захищений Telegram-бот оброблення замовлень та резервувань має зручний та зрозумілий інтерфейс, який забезпечує ефективну взаємодію як для клієнтів, так і для адміністраторів системи. Бот спрощує процес замовлення товарів, надає швидкий доступ до важливої інформації та опцій, а також дозволяє гнучко керувати каталогом продуктів.

ВИСНОВКИ

В ході бакалаврської роботи було розроблено захищений Telegram-бот для оброблення замовлень та резервувань, який забезпечує зручний, ефективний та безпечний спосіб взаємодії компаній зі своїми клієнтами через популярний месенджер.

У першому розділі було проаналізовано теоретичні основи розробки Telegram-ботів для обробки замовлень та резервувань. Розглянуто особливості використання платформи Telegram, потенційні загрози безпеки та існуючі програмні рішення в цій галузі. Встановлено, що месенджер Telegram є привабливою платформою для розробки бота завдяки зручному функціоналу, високому рівню безпеки та великій користувацькій базі. Проте, існуючі рішення часто мають обмежений функціонал або недостатній рівень безпеки. На основі цих висновків було сформульовано завдання для розробки захищеного Telegram-бота.

У другому розділі детально розглянуто процес проектування та розроблення підходу до створення захищеного Telegram-бота. Обґрунтовано необхідність розробки такого бота з точки зору зручності взаємодії з клієнтами та забезпечення належного рівня безпеки даних. Проаналізовано та обґрунтовано вибір реляційної бази даних PostgreSQL та нереляційної бази даних MongoDB для зберігання даних. Спроектовано компоненти архітектури бота з використанням мікросервісного підходу, що забезпечує гнучкість, масштабованість та високу доступність системи. Запропоновано комплексний підхід до забезпечення безпеки, включаючи фільтрацію трафіку, використання вебхуків, шифрування даних та аутентифікацію користувачів.

У третьому розділі описано програмну реалізацію захищеного Telegram-бота. Обґрунтовано вибір мови програмування Python та середовища розробки PyCharm. Детально описано функціонал бота, включаючи процес оформлення замовлень, додавання товарів до кошика, вибір способу доставки та оплати. Представлено інструкцію користувачу, яка демонструє зручність та інтуїтивність інтерфейсу бота.

В результаті проведеної роботи було успішно розроблено захищений Telegram-бот для оброблення замовлень та резервувань. Бот забезпечує зручний та безпечний спосіб взаємодії компаній з клієнтами, дозволяючи здійснювати замовлення товарів та послуг прямо в месенджері Telegram. Завдяки ретельно спроектованій архітектурі та впровадженим заходам безпеки, бот гарантує належний захист конфіденційних даних користувачів та стійкість до потенційних загроз.

Завдяки ретельному аналізу теоретичних основ, проектуванню архітектури та впровадженню заходів безпеки, розроблений бот гарантує належний захист конфіденційних даних користувачів та стійкість до потенційних загроз. Він має значний потенціал для практичного застосування в різних галузях, дозволяючи компаніям надавати своїм клієнтам сучасний та зручний спосіб взаємодії.

Розроблений захищений Telegram-бот має значний потенціал для практичного застосування в різних галузях, таких як електронна комерція, сфера послуг, туризм тощо. Він дозволяє компаніям надавати своїм клієнтам сучасний, зручний та безпечний спосіб взаємодії, що може підвищити лояльність клієнтів та збільшити обсяги продажів.

Подальші напрямки розвитку проекту можуть включати розширення функціоналу бота, інтеграцію з додатковими платіжними системами, впровадження персоналізованих рекомендацій та розширення можливостей аналітики даних. Також доцільно розглянути можливість адаптації бота для використання на інших платформах месенджерів, таких як Viber або WhatsApp.

Загалом, розроблений захищений Telegram-бот є ефективним та інноваційним рішенням для оброблення замовлень та резервувань, що відповідає сучасним вимогам безпеки, зручності та функціональності. Його впровадження може допомогти компаніям покращити взаємодію з клієнтами, оптимізувати бізнес-процеси та підвищити ефективність продажів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Гуменюк К. П. Розробка чат–боту в Telegram для вивчення мови програмування Java [Електронний ресурс] : thesis / Гуменюк К. П. – [Б. м.], 2021. – Режим доступу: http://ir.stu.cn.ua/123456789/23018 (дата звернення: 27.05.2024). – Назва з екрана.

2. Кваснецький А. Б. Розробка конструктора чат–ботів для месенджера Telegram [Електронний ресурс] : thesis / Кваснецький А. Б., Безменов Микола Іванович. – [Б. м.], 2019. – Режим доступу: http://repository.kpi.kharkov.ua/handle/KhPI–Press/48381 (дата звернення: 27.05.2024). – Назва з екрана.

3. Сидоренко В. В. Особливості захисту інформації в інформаційних системах з використанням баз даних [Електронний ресурс] : thesis / Сидоренко В. В. – [Б. м.], 2014. – Режим доступу: http://dspace.kntu.kr.ua/jspui/handle/123456789/2994 (дата звернення: 27.05.2024). – Назва з екрана.

4. Chaple–Gil A. M. Telegram messenger: a suitable tool for teledentistry. [Electronic resource] / Alain Manuel Chaple–Gil, Kelvin Ian Afrashtehfar // Journal of oral research. – 2020. – Vol. 9, no. 1. – P. 4–6. – Mode of access: https://doi.org/10.17126/joralres.2020.001 (date of access: 27.05.2024). – Title from screen.

5. Дудикевич В. Б. Протоколи і механізми безпеки інформації в комп'ютерних системах і мережах [Електронний ресурс] / В. Б. Дудикевич, Б. В. Томашевський, Р. В. Сергієнко // Ukrainian information security research journal. – 2009. – Т. 11, № 1(42). – Режим доступу: https://doi.org/10.18372/2410–7840.11.5373 (дата звернення: 27.05.2024). – Назва з екрана.

6. Кваша А. О. Використання чат–ботів для потреб маркетингу [Електронний ресурс] : thesis / Кваша А. О. – [Б. м.], 2019. – Режим доступу: https://er.knutd.edu.ua/handle/123456789/12804 (дата звернення: 27.05.2024). – Назва з екрана.

7. Telegram [Electronic resource] // Archiwa, biblioteki i muzea kościelne. – 2020. – Vol. 79. – P. 25. – Mode of access: https://doi.org/10.31743/abmk.9387 (date of access: 27.05.2024). – Title from screen.

8. Vieirais M. Securing web applications: techniques and challenges [Electronic resource] / Marco Vieirais // Journal of information security and cryptography (enigma). – 2015. – Vol. 1, no. 1. – P. 65. – Mode of access: https://doi.org/10.17648/enig.v1i1.21 (date of access: 27.05.2024). – Title from screen.

9. Vijayakumar T. API security [Electronic resource] / Thurupathan Vijayakumar // Practical API architecture and development with azure and AWS. – Berkeley, CA, 2018. – P. 97–132. – Mode of access: https://doi.org/10.1007/978–1–4842–3555–3_5 (date of access: 27.05.2024). – Title from screen.

10. Wallace M. Building telegram bots: the telegram bot api to develop bots by 12 programming languages / Marina Wallace. – [S. l.] : Independently Published, 2022.

11. West P. G. Introduction to cybersecurity for busy people [Electronic resource] / Paul G. West // Journal of learning for development. – 2020. – Vol. 7, no. 2. – P. 261–263. – Mode of access: https://doi.org/10.56059/jl4d.v7i2.399 (date of access: 27.05.2024). – Title from screen.

12. Secure software development best practices [Electronic resource] / Muhammad Firdaus Fauzi [et al.] // International journal of emerging multidisciplinaries: computer science & artificial intelligence. – 2023. – Vol. 2, no. 1. – Mode of access: https://doi.org/10.54938/ijemdcsai.2023.02.1.256 (date of access: 27.05.2024). – Title from screen.

13. Shahrul A. Choosing an instant messaging app: security or convenience? Comparison between whatsapp and telegram [Electronic resource] / Azzhan Shahrul, Aji Prasetya Wibawa // Buletin ilmiah sarjana teknik elektro. – 2021. – Vol. 3, no. 2. – P. 115–121. – Mode of access: https://doi.org/10.12928/biste.v3i2.2784 (date of access: 27.05.2024). – Title from screen.

14. Popov V. A. Telegram messenger data import models [Electronic resource] / V. A. Popov, A. A. Chepovskiy // Vestnik NSU. series: information technologies. – 2022.

– Vol. 20, no. 2. – P. 60–71. – Mode of access: https://doi.org/10.25205/1818–7900–2022–20–2–60–71 (date of access: 27.05.2024). – Title from screen.

15. Gollmann D. Securing Web applications [Electronic resource] / Dieter Gollmann // Information security technical report. – 2008. – Vol. 13, no. 1. – P. 1–9. – Mode of access: https://doi.org/10.1016/j.istr.2008.02.002 (date of access: 27.05.2024). – Title from screen.

16. Graff M. Secure coding: principles & practices / Mark Graff. – Beijing : O'Reilly, 2003. – 202 p.

17. Huang D. Practical cyber security for extremely busy people: protect yourself, your family, and your career from online exploitation / Daniel Huang. – [S. l.] : Independently Published, 2020.

18. Ismawati D. The development of telegram BOT through short story [Electronic resource] / Dwi Ismawati, Iis Prasetyo // Brawijaya international conference on multidisciplinary sciences and technology (BICMST 2020), Malang, Indonesia, 2–3 January 2020. – Paris, France, 2020. – Mode of access: https://doi.org/10.2991/assehr.k.201021.049 (date of access: 27.05.2024). – Title from screen.

19. Quick reference guide [Electronic resource] // Nursing standard. – 1999. – Vol. 13, no. 42. – P. 29. – Mode of access: https://doi.org/10.7748/ns.13.42.29.s50 (date of access: 27.05.2024). – Title from screen.

20. Rozga S. Introduction to chat bots [Electronic resource] / Szymon Rozga // Practical bot development. – Berkeley, CA, 2018. – P. 1–28. – Mode of access: https://doi.org/10.1007/978–1–4842–3540–9_1 (date of access: 27.05.2024). – Title from screen.

21. Scarfone K. NIST cybersecurity framework 2.0: [Electronic resource] / Karen Scarfone. – Gaithersburg, MD : National Institute of Standards and Technology, 2024. – Mode of access: https://doi.org/10.6028/nist.sp.1302.ipd (date of access: 27.05.2024). – Title from screen.

22. Advanced XSS attack vectors [Electronic resource] / J. Grossman [et al.] // Cross site scripting attacks. – [S. l.], 2007. – P. 191–218. – Mode of

access: https://doi.org/10.1016/b978–159749154–9/50006–8 (date of access: 27.05.2024). – Title from screen.

23. Armando Y. Penetration testing tangerang city web application with implementing OWASP top 10 web security risks framework [Electronic resource] / Yoel Armando, Rosalina Rosalina // JISA(Jurnal informatika dan sains). – 2023. – Vol. 6, no. 2. – P. 105–109. – Mode of access: https://doi.org/10.31326/jisa.v6i2.1656 (date of access: 27.05.2024). – Title from screen.

24. De B. API security [Electronic resource] / Brajesh De // API management. – Berkeley, CA, 2017. – P. 111–142. – Mode of access: https://doi.org/10.1007/978–1–4842–1305–6_7 (date of access: 27.05.2024). – Title from screen.

25. Fox D. Cross site scripting (XSS) [Electronic resource] / Dirk Fox // Datenschutz und Datensicherheit – DuD. – 2012. – Vol. 36, no. 11. – P. 840. – Mode of access: https://doi.org/10.1007/s11623–012–0284–2 (date of access: 27.05.2024). – Title from screen.

26. Jyothi Salibindla. Microservices API security [Electronic resource] / Jyothi Salibindla // International journal of engineering research and. – 2018. – Vol. V7, no. 01. – Mode of access: https://doi.org/10.17577/ijertv7is010137 (date of access: 27.05.2024). – Title from screen.

27. Kissoon T. Securing web applications / Tara Kissoon. – [S. l.] : Taylor & Francis Group, 2012. – 300 p.

28. Kovács L. National cybersecurity strategy framework [Electronic resource] / László Kovács // Academic and applied research in military and public. – 2019. – Vol. 18, no. 2. – P. 65–76. – Mode of access: https://doi.org/10.32565/aarms.2019.2.9 (date of access: 27.05.2024). – Title from screen.

29. Machine learning use cases in cybersecurity [Electronic resource] / S. M. Avdoshin [et al.] // Proceedings of the institute for system programming of the RAS. – 2019. – Vol. 31, no. 5. – P. 191–202. – Mode of access: https://doi.org/10.15514/ispras–2019–31(5)–15 (date of access: 27.05.2024). – Title from screen.

30. Mayes K. Information security best practices [Electronic resource] / Keith Mayes, Konstantinos Markantonakis // Secure smart embedded devices, platforms and applications. – New York, NY, 2013. – P. 119–144. – Mode of access: https://doi.org/10.1007/978–1–4614–7915–4_6 (date of access: 27.05.2024). – Title from screen.

31. Möller D. P. F. Machine learning and deep learning [Electronic resource] / Dietmar P. F. Möller // Cybersecurity in digital transformation. – Cham, 2020. – P. 77–87. – Mode of access: https://doi.org/10.1007/978–3–030–60570–4_5 (date of access: 27.05.2024). – Title from screen.

32. Morgan D. Securing high availability Web applications [Electronic resource] / David Morgan // Network security. – 2004. – Vol. 2004, no. 5. – P. 8–11. – Mode of access: https://doi.org/10.1016/s1353–4858(04)00079–0 (date of access: 27.05.2024). – Title from screen.

33. Ul Haq M. N. Mastering cloud security: techniques and best practices [Electronic resource] / Mohd Naved Ul Haq, Mohit Kumar Sharma // Emerging trends in cloud security and intelligent agents. – [S. l.], 2023. – Mode of access: https://doi.org/10.52458/9788196869434.2023.eb.grf.ch–07 (date of access: 27.05.2024). – Title from screen.

34. Silaban F. A. Build smart home controls using wemos microcontroller–based telegram app [Electronic resource] / Freddy Artadima Silaban, Rendy Elmianto, Lukman Madriavin Silalahi // CCIT journal. – 2021. – Vol. 14, no. 1. – P. 1–12. – Mode of access: https://doi.org/10.33050/ccit.v14i1.802 (date of access: 27.05.2024). – Title from screen.

35. Two factor for internet end–users [Electronic resource] // Two–Factor authentication. – [S. l.], 2015. – P. 68–77. – Mode of access: https://doi.org/10.2307/j.ctt15hvwnz.12 (date of access: 27.05.2024). – Title from screen.

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції
“Управління інформаційною
безпекою” кафедри МБІС
д.т.н., професор
_____Юрій ЯРЕМЧУК
“ 22 ” березня 2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до бакалаврської дипломної роботи на тему:

Розробка захищеного Telegram– бота оброблення замовлень та резервувань

08–72.БДР.015.00.069.ТЗ

Керівник бакалаврської дипломної роботи
к.т.н., доц. каф. МБІС
_____ Грицак А. В.
“____” _____

Вінниця – 2024 р.

1. Найменування та область застосування

Захищений Telegram–бот для оброблення замовлень та резервувань. Область застосування: надання послуг захищеного оброблення замовлень та резервування через розроблений Telegram–бот.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № 80 від 11.03.2024 р.

3. Мета та призначення розробки

3.1 Мета розробки: розробка захищеного Telegram-бота для оброблення замовлень та резервувань з використанням методів шифрування та захисту конфіденційних даних користувачів.

3.2 Призначення: розроблений Telegram-бот забезпечує безпечне оброблення замовлень та резервувань, захищаючи персональні дані та конфіденційну інформацію користувачів від несанкціонованого доступу та витоку.

4. Джерела розробки

4.1. Кваснецький А. Б. Розробка конструктора чат–ботів для месенджера Telegram [Електронний ресурс] : thesis / Кваснецький А. Б., Безменов Микола Іванович. – [Б. м.], 2019.

4.2. Кваша А. О. Використання чат–ботів для потреб маркетингу [Електронний ресурс] : thesis / Кваша А. О. – [Б. м.], 2019.

4.3. Розробка веб–додатків та сервісів на платформі node.js [Електронний ресурс] / О. П. Кошова [та ін.] // Таврійський науковий вісник. Серія: технічні науки. – 2023. – № 2. – С. 78–89.

4.4. Сидоренко В. В. Особливості захисту інформації в інформаційних системах з використанням баз даних [Електронний ресурс] : thesis / Сидоренко В. В. – [Б. м.], 2014.

4.5 Слишинська В. О. Аналіз даних за допомогою штучного інтелекту [Електронний ресурс] : thesis / Слишинська В. О. – [Б. м.], 2017.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний засіб повинен виконувати процес автентифікації користувачів у системі.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Intel Core i5–5500U 2.40 GHz і подібні до них;

– оперативна пам’ять – не менше 4Gb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко–економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської дипломної роботи	Початок	Закінчення
1.	Визначення напрямку бакалаврської роботи, формулювання теми	14.02.2024	19.02.2024
2.	Аналіз предметної області обраної теми	20.02.2024	11.03.2024
3.	Розробка алгоритму роботи	12.03.2024	05.04.2024
4.	Написання бакалаврської роботи на основі розробленої теми	06.04.2024	02.05.2024
5.	Передзахист бакалаврської дипломної роботи	06.05.2024	24.05.2024
6.	Виправлення, уточнення, корегування бакалаврської дипломної роботи	02.06.2024	07.06.2024
7.	Захист бакалаврської дипломної роботи	12.06.2024	13.06.2024

10. Порядок контролю та прийому

10.1 До приймання бакалаврської дипломної роботи надається:

– ПЗ до бакалаврської дипломної роботи;

- демонстрація результату бакалаврської дипломної роботи;
- презентація;
- відзив керівника роботи;
- відзив рецензента

Технічне завдання до виконання прийняв _____ Молчан Є. В.

Додаток Б. Лістинг коду розроблюваного захищеного Telegram–бота

```
import os
import handlers
from aiogram import executor, types
from aiogram.types import ReplyKeyboardMarkup, ReplyKeyboardRemove
from data import config
from loader import dp, db, bot
import filters
import logging

filters.setup(dp)

WEBAPP_HOST = "0.0.0.0"
WEBAPP_PORT = int(os.environ.get("PORT", 5000))
user_message = 'User'
admin_message = 'Admin'


@dp.message_handler(commands='start')
async def cmd_start(message: types.Message):

    markup = ReplyKeyboardMarkup(resize_keyboard=True)

    markup.row(user_message, admin_message)

    await message.answer('''Hi ! 👋

🤖 I am a bot–shop for the supply of goods of any category.

🛍️ To go to the catalog and select the products you like, use the /menu command.

💰 You can replenish your account through any bank in Ukraine.

❓ Any questions? No problem! The /sos command will help you contact the admins, who will try to respond as quickly as possible.

🤝 Thanks for using my bot
    ''', reply_markup=markup)


@dp.message_handler(text=user_message)
async def user_mode(message: types.Message):

    cid = message.chat.id
    if cid in config.ADMINS:
        config.ADMINS.remove(cid)

    await message.answer('User mode enabled.', reply_markup=ReplyKeyboardRemove())


@dp.message_handler(text=admin_message)
async def admin_mode(message: types.Message):

    cid = message.chat.id
    if cid not in config.ADMINS:
```

```
        config.ADMINS.append(cid)

    await message.answer('Admin mode enabled.', reply_markup=ReplyKeyboardRemove())


async def on_startup(dp):
    logging.basicConfig(level=logging.INFO)
    db.create_tables()

    await bot.delete_webhook()
    await bot.set_webhook(config.WEBHOOK_URL)


async def on_shutdown():
    logging.warning("Shutting down..")
    await bot.delete_webhook()
    await dp.storage.close()
    await dp.storage.wait_closed()
    logging.warning("Bot down")


if __name__ == '__main__':

    if "HEROKU" in list(os.environ.keys()):

        executor.start_webhook(
            dispatcher=dp,
            webhook_path=config.WEBHOOK_PATH,
            on_startup=on_startup,
            on_shutdown=on_shutdown,
            skip_updates=True,
            host=WEBAPP_HOST,
            port=WEBAPP_PORT,
        )

    else:

        executor.start_polling(dp, on_startup=on_startup, skip_updates=False)

import sqlite3 as lite

class DatabaseManager(object):

    def __init__(self, path):
        self.conn = lite.connect(path)
        self.conn.execute('pragma foreign_keys = on')
        self.conn.commit()
        self.cur = self.conn.cursor()

    def create_tables(self):
        self.query('CREATE TABLE IF NOT EXISTS products (idx text, title text, body text, photo blob, price int, tag text)')
        self.query('CREATE TABLE IF NOT EXISTS orders (cid int, usr_name text, usr_address text, products text)')
        self.query('CREATE TABLE IF NOT EXISTS cart (cid int, idx text, quantity int)')
        self.query('CREATE TABLE IF NOT EXISTS categories (idx text, title text)')
        self.query('CREATE TABLE IF NOT EXISTS wallet (cid int, balance real)')
        self.query('CREATE TABLE IF NOT EXISTS questions (cid int, question text)')

    def update_category(self, old_title, new_title):
```

```
        self.query("UPDATE categories SET title = ? WHERE title = ?", (new_title, old_title))

    def update_product_tag(self, old_tag, new_tag):
        self.query("UPDATE products SET tag = ? WHERE tag = ?", (new_tag, old_tag))


    def query(self, arg, values=None):
        if values == None:
            self.cur.execute(arg)
        else:
            self.cur.execute(arg, values)
        self.conn.commit()

    def fetchone(self, arg, values=None):
        if values == None:
            self.cur.execute(arg)
        else:
            self.cur.execute(arg, values)
        return self.cur.fetchone()

    def fetchall(self, arg, values=None):
        if values == None:
            self.cur.execute(arg)
        else:
            self.cur.execute(arg, values)
        return self.cur.fetchall()

    def __del__(self):
        self.conn.close()
```

products: idx text, title text, body text, photo blob, price int, tag text

orders: cid int, usr_name text, usr_address text, products text

cart: cid int, idx text, quantity int ==> product_idx

categories: idx text, title text

wallet: cid int, balance real

questions: cid int, question text

Додаток В. Ілюстративний матеріал

Розробка захищеного Telegram-бота оброблення замовлень та резервувань

Автор: Єгор Молчан

Вінницький національний технічний університет

2024

Вступ

- Актуальність розробки захищеного Telegram-бота для автоматизації процесів оброблення замовлень та резервувань з використанням механізмів шифрування даних та аутентифікації користувачів.

Мета роботи

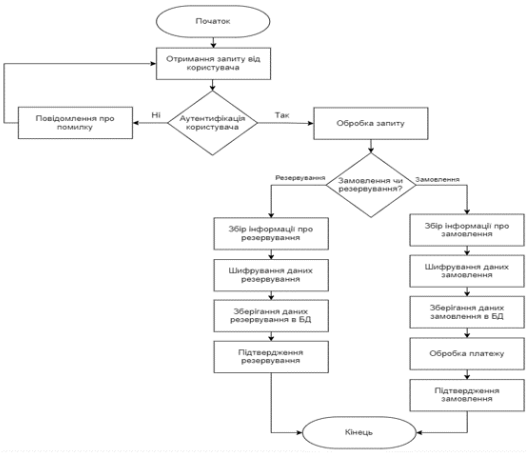
- Метою роботи є розробка захищеного Telegram-бота для автоматизації процесів оброблення замовлень та резервувань з використанням механізмів шифрування даних та аутентифікації користувачів для забезпечення безпечного обміну інформацією та високого рівня користувацького сервісу.

Аналіз існуючих рішень

- Аналіз існуючих рішень для обробки замовлень та резервувань, їх переваг та недоліків, дослідження відомих загроз та існуючих програмних рішень в даній сфері.

Проектування архітектури

- Проектування компонентів архітектури Telegram-бота, взаємодія з базою даних, механізми шифрування та аутентифікації користувачів.

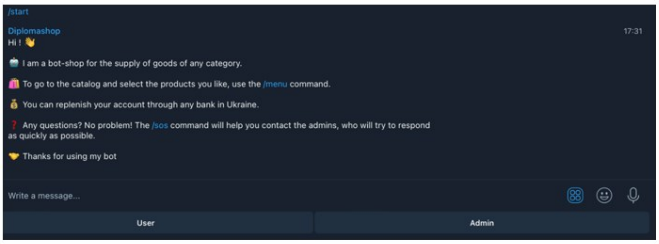


Блок–схема алгоритму роботи підходу захищеного Telegram–бота оброблення замовлень та резервувань

Реалізація бота

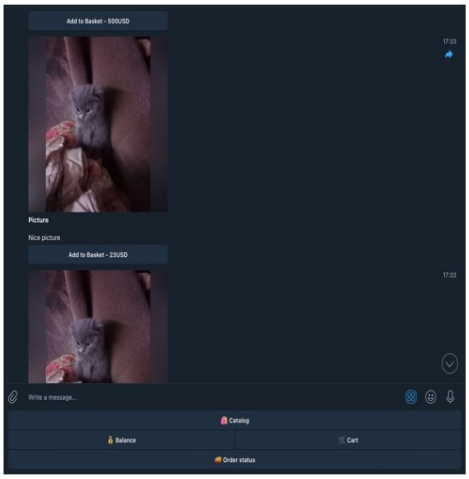
- Програмна реалізація захищеного Telegram-бота, вибір мови програмування та середовища розробки, інструкція користувачу.

Інтерфейс Telegram-бота



Початкове привітання та інструкції вTelegram–бота при запуску чату.

Головне меню із опціями каталогу, балансу, кошика та статусу замовлення.



Відображення обраного товару з картинкою, назвою, ціною та опцією додавання до кошику.

Висновки

- Результати виконаної роботи: створення захищеного Telegram-бота, що забезпечує безпечний обмін даними та високий рівень користувацького сервісу.

Додаток Г. Звіт на перевірку антиплагіату

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка захищеного Telegram-бота оброблення замовлень та резервувань

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 98 % Схожість 2 %

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ (підпис) Коваль Н.П. (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ (підпис) Молчан Є.В. (прізвище, ініціали)

Керівник роботи _____ (підпис) Грицак А.В. (прізвище, ініціали)