ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка програмних засобів для організації онлайн-продажів через чат-бот»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Владислав КАЛОМБЕТ
(підпис)

Виконав: здобувач вищої освіти групи ПД-42

Владислав КАЛОМБЕТ

Керівник: Ірина ЩЕРБИНА
к.т.н., доцент

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

_____Каломбету Владиславу Станіславовичу_____

1. Тема кваліфікаційної роботи: «Розробка програмних засобів для організації онлайн-продажів через чат-бот»
керівник кваліфікаційної роботи к.т.н., доцент Ірина ЩЕРБИНА,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56.

2. Строк подання кваліфікаційної роботи «02» червня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про автоматизацію електронної комерції, принципи роботи чат-ботів, документація Telegram Bot API, Node.js, Express, Mongoose та MongoDB.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз технологій автоматизації електронних продажів, включаючи веб-платформи, маркетплейси, соціальні мережі та використання Telegram-ботів для онлайн-продажів.
2. Вибір засобів для реалізації Telegram-бота, визначення функціональних і нефункціональних вимог, а також проектування зручного інтерфейсу.

3. Проектування архітектури, структури даних і бази даних Telegram-бота для ефективної автоматизації продажів цифрових товарів.
4. Реалізація Telegram-бота з інтеграцією платіжних систем, основних функцій, завантаження на GitHub та повне тестування з аналізом результатів.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Програмні засоби реалізації.
4. Загальна діаграма діяльності системи.
5. Діаграма процесу оплати та доставки товар.
6. Екранні форми.
7. Апробація результатів дослідження.

6. Дата видачі завдання «25» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	25.02.-04.03.2025	
2	Аналіз та дослідження існуючих аналогів	05.03-13.03.2025	
3	Огляд технологій автоматизації електронної комерції та архітектури чат-ботів	14.03-19.03.2025	
4	Проектування Telegram-бота для автоматизованого продажу цифрових товарів	20.03-26.03.2025	
5	Програмна реалізація Telegram-бота (Node.js, MongoDB, API інтеграції)	27.03-17.04.2025	
6	Тестування функціоналу бота та платіжних систем	17.04-01.05.2025	
7	Оформлення роботи: вступ, висновки, реферат	01.04-12.05.2025	
8	Розробка демонстраційних матеріалів	12.05-18.05.2025	
9	Попередній захист роботи	19.05-30.05.2025	

Здобувач вищої освіти _____ Владислав КАЛОМБЕТ
(підпис)

Керівник
кваліфікаційної роботи _____ Ірина ЩЕРБИНА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 48 стор., 1 табл., 15 рис., 11 джерел.

Мета роботи – автоматизація процесу продажу цифрових товарів з підтримкою криптовалютних оплат та адмініструванням через внутрішню панель керування за допомогою Telegram-бота.

Об'єкт дослідження – процес електронної комерції з використанням месенджерів як інструменту продажу цифрових продуктів.

Предмет дослідження – архітектура, функціональність і технічні особливості Telegram-бота для продажу цифрових товарів, включаючи механізм оплати, систему управління товарами та взаємодію з користувачем.

Короткий зміст роботи: У роботі проаналізовано сучасні підходи та технології автоматизації електронної комерції, а також засоби розробки Telegram-ботів для здійснення онлайн-продажів. Розглянуто особливості інтеграції платіжних систем, зокрема криптовалютних сервісів та API банківських карток. Розроблено алгоритм роботи Telegram-бота та програмно реалізовані ключові функціональні можливості, зокрема: каталог товарів, система оплати, баланс користувача, історія замовлень, панель адміністратора для керування товарами та замовленнями. Проведено функціональне тестування, тестування на коректність обробки платежів та перевірку захищеності даних користувачів.
У роботі використано Node.js як серверне середовище, бібліотеку node-telegram-bot-api для взаємодії з Telegram API, Express та Mongoose для побудови REST-сервісу та взаємодії з базою даних MongoDB.
Сферою використання розробленого Telegram-бота є автоматизація процесів онлайн-продажу цифрових товарів у малому та середньому бізнесі.

КЛЮЧОВІ СЛОВА: Telegram-бот, електронна комерція, Node.js, MongoDB, автоматизація продажів, чат-бот, інтеграція платіжних систем, цифрові товари.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API - Application Programming Interface

UI - User Interface

CRM - Customer Relationship Management

AI - Artificial Intelligence

SEO - Search Engine Optimization

HTTPS - HyperText Transfer Protocol Secure

MongoDB - Mongo Database

Node.js - Node JavaScript

Web3 - Web 3.0

USDT – Tether (Crypto)

LiqPay - Liquid Payments

CryptoBot - Cryptocurrency Bot

EasyPay - Easy Payment

NowPayment - Now Payment Solutions

Google Analytics - Google Analytical Tools

ВСТУП

Актуальність: організація онлайн-продажів є важливим аспектом сучасного бізнесу, що сприяє підвищенню доступності товарів і послуг у цифровому середовищі. З розвитком технологій чат-боти стають ефективним інструментом для автоматизації продажів, забезпечуючи зручність для користувачів і оптимізуючи бізнес-процеси. Telegram, як одна з найпопулярніших платформ для комунікації, надає широкі можливості для створення ботів, які дозволяють користувачам купувати товари в зручний час і в комфортних умовах. Чат-бот для продажів забезпечує швидкий доступ до товарів, персоналізовану взаємодію та автоматизоване управління замовленнями, що робить його важливим інструментом для сучасного e-commerce.

Розробка Telegram-бота для автоматизації продажів дозволяє об’єднати в одному рішенні функції управління товарами, обробки платежів, аналітики даних і взаємодії з користувачами. Такий бот забезпечує гнучкість, дозволяючи користувачам переглядати товари, оформлювати замовлення та отримувати підтримку в реальному часі, а адміністраторам — ефективно управляти асортиментом і клієнтською базою.

Об’єктом дослідження є процес електронної комерції з використанням месенджерів як інструменту продажу цифрових продуктів.

Предметом дослідження є архітектура, функціональність і технічні особливості Telegram-бота для продажу цифрових товарів, включаючи механізм оплати, систему управління товарами та взаємодію з користувачем.

Метою роботи є автоматизація процесу продажу цифрових товарів з підтримкою криптовалютних оплат та адмініструванням через внутрішню панель керування за допомогую телеграм бота.

Методи дослідження: На першому етапі було проаналізовано існуючі рішення для автоматизації продажів через чат-боти для формування

функціональних і нефункціональних вимог до програмного забезпечення. Далі проведено огляд технологічного стеку, який відповідає вимогам, і обрано Node.js з бібліотекою node-telegram-bot-api для взаємодії з Telegram API[1], Express.js для серверної частини, MongoDB з Mongoose для роботи з базою даних. Розробка проводилася з використанням модульної архітектури, асинхронного програмування та методів обробки природної мови для аналізу запитів користувачів. Прототипування інтерфейсу бота виконувалося з урахуванням принципів зручності використання.

Наукова новизна роботи: Розроблено Telegram-бот, який поєднує автоматизацію продажів, інтеграцію з платіжними системами (Crypto, EasyPay, CryptoBot[12]), систему станів для управління діалогом, а також аналітику даних для оптимізації бізнес-процесів.

Практична значущість результатів: Реалізований Telegram-бот може використовуватися для автоматизації продажів цифрових товарів, забезпечуючи зручний інтерфейс, безпечну обробку транзакцій і ефективне управління користувачами та асортиментом на платформі Telegram/

1 АНАЛІЗ ТЕХНОЛОГІЙ АВТОМАТИЗАЦІЇ ЕЛЕКТРОНИХ ПРОДАЖІВ

1.1 Аналіз існуючих технологій автоматизації електронних продажів

Автоматизація електронних продажів відіграє ключову роль у розвитку сучасного бізнесу, сприяючи оптимізації процесів, зниженню операційних витрат і підвищенню рівня задоволеності клієнтів. Цей розділ аналізує різні технології, такі як веб-платформи, кастомні рішення, маркетплейси, соціальні мережі та месенджер-боти, з урахуванням їхніх основних функцій, унікальних особливостей та переваг.

Способи автоматизації електронних продажів представлені на рисунку 1.1

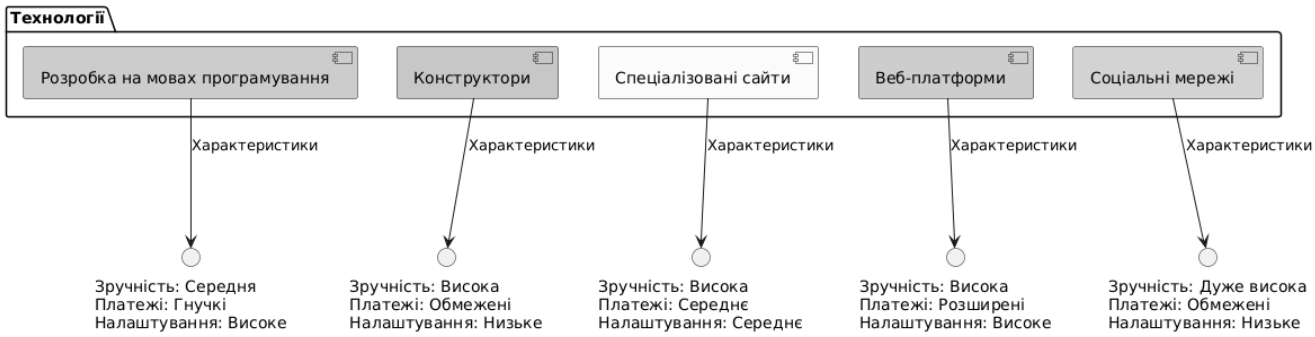


Рис. 1.1 Способи автоматизації електронних продажів

Веб-платформи є одним із найпоширеніших інструментів для організації онлайн-продажів, надаючи широкі можливості для презентації товарів, обробки замовлень і взаємодії з клієнтами. Конструктори, такі як Wix, Squarespace чи український Tilda, дозволяють швидко створювати сайти з готовими шаблонами, інтеграцією платіжних систем (PayPal, Stripe) та базовою автоматизацією, наприклад, надсиланням сповіщень про статус замовлення. Інтернет-магазини на базі WooCommerce (плагін для WordPress) чи Magento пропонують гнучкі каталоги, системи знижок і персоналізовані рекомендації. Системи управління контентом (CMS), такі як Joomla чи Drupal, забезпечують складніші рішення з інтеграцією CRM[3], аналітикою та модулями для email-розсилок.

Основні функції веб-платформ для продажів**:**

– Каталог товарів: Структурований перелік із фільтрами за ціною, категоріями та характеристиками.

– Кошик і оформлення замовлень: Додавання товарів до кошика з вибором доставки та оплати.

– Інтеграція з платіжними системами: Підтримка LiqPay, PayPal, Stripe для безпечних транзакцій.

– Особистий кабінет: Профіль із історією замовлень і бонусними програмами.

– SEO[5]-оптимізація: Інструменти для просування в пошукових системах.

– Аналітика: Збір даних через Google Analytics[15] про відвідуваність і конверсії.

– Система відгуків: Можливість залишати оцінки для підвищення довіри.

– Адаптивний дизайн: Оптимізація для смартфонів і планшетів.

Однією з унікальних особливостей є використання штучного інтелекту для персоналізації. Наприклад, Magento інтегрує AI[4] для аналізу поведінки покупців і пропозиції товарів, а Tilda пропонує "динамічні блоки", які автоматично оновлюють контент (акції, новинки), підвищуючи актуальність сайту. Також деякі платформи підтримують технологію PWA (Progressive Web Apps), що забезпечує швидку роботу й доступ офлайн.

Переваги**:**

– Гнучкість дизайну: Можливість створення унікального інтерфейсу під бренд.

– Широкий функціонал: Реалізація складних логік, наприклад, програм лояльності.

– Довіра клієнтів: Професійний вигляд підвищує репутацію.

– Масштабованість: Легке додавання нових функцій, як чат-боти.

– Аналітичні можливості: Детальна статистика для оптимізації стратегій.

Недоліки**:**

– Висока вартість розробки: Значні витрати на створення та підтримку.

– Складність адміністрування: Потреба в технічних знаннях або спеціалістах.

– Залежність від хостингу: Проблеми з сервером можуть призвести до простоїв.

– Необхідність маркетингу: Без SEO сайт залишається непоміченим.

Кастомні веб-рішення, розроблені з використанням фреймворків, таких як React чи Vue.js, та бібліотек, як jQuery чи Axios, надають унікальні можливості для створення індивідуальних платформ. Такі системи дозволяють реалізовувати автоматичну обробку великих обсягів даних, інтеграцію з API обмінних курсів і реальну синхронізацію складів, що забезпечує високу швидкість роботи й адаптивність.

Основні функції кастомних рішень**:**

– Динамічні інтерфейси: Адаптивні сторінки з миттєвим оновленням.

– Інтеграція API: Підключення до зовнішніх сервісів (платежі, курси валют).

– Управління контентом: Гнучке редагування товарів і акцій.

– Аналітика в реальному часі: Відстеження продажів і поведінки користувачів.

– Кросплатформність: Підтримка десктопів і мобільних пристроїв.

Кастомні рішення вирізняються можливістю інтеграції з Web3[9]-технологіями, наприклад, блокчейном для безпечних транзакцій через смарт-контракти, що особливо актуально для цифрових товарів. Також можна використовувати бібліотеку TensorFlow.js для прогнозування попиту на основі історичних даних, що робить платформу інноваційною й адаптивною до ринкових тенденцій.

Переваги:

– Повний контроль над функціоналом: Можливість реалізації унікальних ідей.

– Адаптація під потреби: Налаштування під специфічні бізнес-процеси.

– Висока швидкість: Оптимізований код для миттєвих відповідей.

Недоліки:

– Висока вартість: Значні ресурси на розробку й підтримку.

– Потреба в експертах: Вимагає кваліфікованих розробників.

– Тривалий час реалізації: Розробка займає місяці, а не дні.

Маркетплейси, такі як українські Rozetka та Prom.ua, і міжнародні Shopify та Amazon, пропонують готову інфраструктуру для автоматизації продажів, забезпечуючи швидкий доступ до аудиторії та автоматизовану обробку замовлень. Rozetka інтегрує логістику й маркетинг, Prom.ua підтримує малий бізнес, Shopify дозволяє створювати глобальні магазини, а Amazon надає доступ до мільйонів покупців.

Основні функції маркетплейсів**:**

– Каталог товарів: Централізований перелік із фільтрами.

– Автоматизована логістика: Координація доставки та повернень.

– Платіжні системи: Інтеграція з LiqPay, PayPal, Amazon Pay.

– Аналітика продажів: Статистика про популярність товарів.

– Рекламні інструменти: Таргетована реклама й просування.

Rozetka вирізняється програмою лояльності з балами для знижок, а Prom.ua пропонує "аукціон" для конкуренції за топові позиції. Amazon використовує алгоритм A9 для оптимізації пошуку, а Shopify інтегрує AI для персоналізованих рекомендацій, що підвищує залученість клієнтів.

Переваги**:**

– Швидкий доступ до аудиторії: Готові канали для продажів.

– Автоматизація платежів: Зручна обробка транзакцій.

– Аналітика: Детальні звіти для оптимізації.

Недоліки:

– Висока комісія: До 15–20% на Amazon.

– Залежність від правил: Обмеження платформи.

– Конкуренція: Труднощі з виділитися серед інших продавців.

Rozetka це найбільший український маркетплейс із асортиментом від електроніки до одягу. Платформа автоматично обробляє замовлення, інтегрує LiqPay[11] і "Нову Пошту", пропонує рекламу на головній сторінці та програму лояльності. Комісія становить 5–10% залежно від категорії, а для старту потрібен

внесок (близько 1000 грн). Також є підтримка локальних акцій, наприклад, "Чорна п’ятниця".

Prom.ua орієнтований на малий і середній бізнес, дозволяє швидко додати товари. Є інтеграція з LiqPay, аналітика й модуль для створення акцій. Аудиторія менша (близько 10 млн відвідувачів на місяць проти 30 млн у Rozetka), але платформа пропонує гнучкі пакети просування (від 500 грн), що робить її доступною для початківців.

Соціальні мережі, такі як Instagram і TikTok, стають популярними платформами для автоматизованих продажів завдяки інтеграції торгових функцій. Instagram пропонує магазини в профілі з тегами товарів, а TikTok інтегрує товари в відео з алгоритмами рекомендацій.

Основні функції соціальних мереж**:**

– Вітрина товарів: Каталог у профілі чи відео.

– Прямі повідомлення: Консультації з клієнтами.

– Реклама: Таргетовані кампанії.

– Візуальний контент: Фото, відео, історії.

– Аналітика: Дані про охоплення та взаємодії.

Instagram вирізняється функцією "Shoppable Posts" для тегування товарів у постах і Stories, що скорочує шлях до покупки. TikTok використовує комп’ютерне зір для розпізнавання товарів у відео, створюючи унікальний маркетинговий підхід із вірусним потенціалом.

Переваги**:**

– Візуальна привабливість: Якісний контент залучає увагу.

– Широка аудиторія: Доступ до мільйонів користувачів.

– Простота просування: Хештеги й реклама працюють швидко.

Недоліки**:**

– Обмежений функціонал: Відсутність кошика чи фільтрів.

– Залежність від платформи: Зміни алгоритмів впливають на охоплення.

– Ручна обробка: Потребує втручання для замовлень.

Месенджер-боти, зокрема Telegram-боти, є інноваційним рішенням для автоматизації продажів у месенджерах. Вони дозволяють створювати інтерактивні меню, інтегрувати платіжні системи (CryptoBot, NowPayments[14]) і обробляти замовлення в реальному часі за допомогою Node.js[8] та бібліотеки node-telegram-bot-api.

Основні функції месенджер-ботів**:**

– Інтерактивне меню: Вибір товарів і дій.

– Платежі: Інтеграція з платіжними системами.

– Сповіщення: Повідомлення про статус замовлення.

– Адмін-панель: Управління товарами й користувачами.

–Підтримка: Автоматичні відповіді на запитання.

Telegram-боти вирізняються можливістю інтеграції з криптовалютними платежами, наприклад, USDT[10] через CoinGecko API, що забезпечує міжнародні транзакції. Також боти можуть використовувати чат-функції для автоматизованих опитувань, збираючи зворотний зв’язок від клієнтів.

Переваги**:**

– Низька вартість розробки: Економічний стартовий поріг.

– Висока швидкість відповіді: Миттєва взаємодія з клієнтами.

– Персоналізація: Індивідуальний підхід до кожного користувача.

Недоліки:

– Обмеження API: Відсутність складного UI[2].

– Потреба в оновленнях: Регулярне вдосконалення для уникнення блокувань.

Кожна технологія має унікальні можливості, що робить їх придатними для різних сценаріїв. Веб-платформи забезпечують гнучкість і довіру, кастомні рішення — інновації, маркетплейси — швидке розширення, соціальні мережі — залучення молоді, а месенджер-боти — економічний і прямий контакт із клієнтами. Комбінація кількох підходів може стати оптимальною стратегією для максимізації ефективності.

1.2 Використання Telegram-бота для автоматизації продажів

Telegram-бот для автоматизації продажів — це спеціалізований програмний засіб, який функціонує в месенджері Telegram і призначений для спрощення та оптимізації процесу онлайн-продажів цифрових товарів. Такі боти забезпечують користувачам зручний доступ до товарів, автоматизоване оформлення замовлень, обробку платежів і підтримку, а адміністраторам — ефективне управління асортиментом і клієнтською базою. Основна мета Telegram-бота полягає в забезпеченні швидкого, безпечного та доступного способу здійснення покупок, що дозволяє користувачам взаємодіяти з магазином у реальному часі з будь-якого пристрою, підключеного до інтернету.

Основні компоненти Telegram-бота для продажів:

– Інтерфейс користувача: Меню та інлайн-кнопки для перегляду товарів, вибору категорій, оформлення замовлень і звернення до підтримки.

– Система управління товарами: Функціонал для додавання, редагування та видалення товарів, включаючи ціни, описи та зображення.

– Платіжні модулі: Інтеграція з платіжними системами (Crypto, EasyPay, CryptoBot) для обробки транзакцій.

– Система станів: Управління діалогом через стани (наприклад, головне меню, вибір кількості товару, очікування платежу).

– Аналітика та статистика: Збір даних про покупки, транзакції та поведінку користувачів для оптимізації продажів.

– Адміністративний інтерфейс: Функції для управління користувачами, розсилок і блокування/розблокування акаунтів.

Основні цілі Telegram-бота для автоматизації продажів:

– Забезпечення зручного доступу до товарів.

– Автоматизація процесу оформлення замовлень.

– Інтеграція з платіжними системами для безпечних транзакцій.

– Надання аналітичних даних для адміністраторів.

– Підвищення ефективності взаємодії з клієнтами.

– Забезпечення безпеки та захисту даних користувачів.

– Спрощення управління асортиментом і клієнтською базою.

Переваги та недоліки Telegram-бота як засобу автоматизації продажів:

Переваги:

– Доступність: Бот доступний у будь-якій точці світу через Telegram, що робить його зручним для користувачів із різних регіонів.

– Гнучкість: Користувачі можуть здійснювати покупки в будь-який час, використовуючи лише смартфон або комп’ютер.

– Автоматизація: Бот зменшує потребу в ручному управлінні продажами, скорочуючи час на обробку замовлень.

– Інтеграція з платіжними системами: Підтримка різних методів оплати (Crypto, EasyPay, CryptoBot) підвищує зручність для користувачів.

– Аналітика: Збір даних про покупки та поведінку клієнтів допомагає оптимізувати бізнес-процеси.

– Низька вартість: Розробка та підтримка бота зазвичай дешевші, ніж створення повноцінного вебсайту чи мобільного додатку.

Недоліки:

– Обмеження платформи: Функціонал бота залежить від можливостей Telegram API, що може обмежувати складні інтеграції.

– Технічні проблеми: Нестабільне інтернет-з’єднання або збої в роботі Telegram можуть впливати на доступність бота.

– Обмеження взаємодії**:** Відсутність прямого людського контакту може ускладнювати вирішення складних запитів користувачів.

– Вимоги до безпеки: Необхідність ретельної валідації даних і захисту від зловмисних дій, що потребує додаткових зусиль у розробці.

– Обмеження інтерфейсу: Інтерфейс Telegram-бота менш гнучкий порівняно з вебсайтами чи мобільними додатками, що може впливати на користувацький досвід.

Основні функції телеграм-бота зображено на рисунку 1.2.

Рис. 1.2 Основні функції Telegram-бота

1.3 Аналіз сучасних методів розробки телеграм ботів для автоматизації продажів

Telegram-боти є ефективним інструментом для автоматизації продажів, дозволяючи створювати зручні онлайн-магазини безпосередньо в месенджері. Вони забезпечують швидкий доступ до товарів, автоматизоване оформлення замовлень, обробку платежів і підтримку клієнтів, а також спрощують управління асортиментом для адміністраторів. Існує кілька способів створення Telegram-ботів, кожен із яких має свої особливості, переваги та недоліки: розробка на мовах програмування, використання конструкторів, створення через спеціалізовані сайти тощо. У цьому розділі розглядаються основні підходи до створення ботів, їх функціональні можливості та обґрунтування вибору конкретного рішення.

Один із найпоширеніших способів створення Telegram-ботів — це розробка з використанням мов програмування. Найпопулярнішими мовами для цього є Python, JavaScript (зокрема Node.js), Java, PHP і Ruby. Цей підхід передбачає використання Telegram Bot API для створення бота з нуля або з використанням бібліотек, таких як python-telegram-bot (Python), node-telegram-bot-api (Node.js), telegraf (Node.js), java-telegram-bot-api (Java) чи php-telegram-bot (PHP).

Основні функції ботів, створених на мовах програмування**:**

– Інтерактивний інтерфейс: Реалізація меню, інлайн-кнопок і команд для перегляду товарів і оформлення замовлень.

– Управління станами: Логіка діалогів (наприклад, вибір товару, введення кількості, очікування платежу).

– Інтеграція з платіжними системами: Підключення до CryptoBot, Stripe чи NowPayments.

– Аналітика: Збір і обробка даних про транзакції через бази даних (MongoDB[7], PostgreSQL).

– Адмін-функції: Управління товарами, користувачами, розсилками.

Розробка на мовах програмування дозволяє створювати унікальні функції, наприклад, інтеграцію з Web3 для криптовалютних платежів або використання машинного навчання для аналізу поведінки користувачів. Наприклад, Python із бібліотекою pandas може аналізувати продажі, а Node.js забезпечує високу швидкість обробки запитів у реальному часі.

Переваги:

– Повна кастомізація: Можливість реалізувати будь-який функціонал без обмежень платформи.

– Висока продуктивність: Оптимізований код забезпечує швидку роботу навіть при великому навантаженні.

– Гнучкість інтеграцій: Легке підключення до баз даних, API та інших сервісів.

– Безпека: Можливість впровадження складних механізмів захисту даних.

Недоліки:

– Висока складність: Потребує знання програмування та розуміння Telegram API.

– Тривалий час розробки: Створення бота з нуля займає більше часу порівняно з конструкторами.

– Потреба в технічній підтримці: Для оновлень і виправлення помилок потрібні розробники.

Конструктори дозволяють створювати Telegram-ботів без глибоких знань програмування, використовуючи графічний інтерфейс для налаштування функціоналу. Популярні конструктори включають BotFather (офіційний інструмент Telegram), ManyChat, Chatfuel і Botsify. Ці платформи пропонують шаблони для створення ботів, які можна налаштувати за допомогою drag-and-drop інтерфейсу.

Основні функції ботів, створених через конструктори**:**

– Меню та кнопки: Налаштування команд для перегляду товарів і оформлення замовлень.

– Автоматизовані відповіді: Реакція на ключові слова чи запити користувачів.

– Інтеграція з платежами: Підключення до Stripe, PayPal (залежить від платформи).

– Сповіщення: Автоматичні повідомлення про замовлення.

– Базова аналітика: Статистика взаємодій із ботом.

Конструктори, такі як ManyChat, пропонують вбудовані інструменти для маркетингу, наприклад, автоматичні розсилки або ретаргетинг, що дозволяє швидко залучати клієнтів. Chatfuel має функцію AI-розпізнавання запитів, що робить взаємодію з користувачами більш природною.

Переваги:

– Простота використання: Не потрібні навички програмування.

– Швидке створення: Бот можна налаштувати за кілька годин.

– Доступність: Багато конструкторів мають безкоштовні плани для базового функціоналу.

Недоліки:

– Обмежений функціонал: Неможливо реалізувати складні бізнес-логіки чи інтеграції.

– Обмежена кастомізація: Дизайн і функції залежать від шаблонів платформи.

– Залежність від платформи: Обмеження в масштабуванні та інтеграціях через правила конструктора.

Спеціалізовані сайти, такі як Tilda (з модулем для створення ботів), Zapier (для автоматизації через Telegram) і Botmother, дозволяють створювати ботів через веб-інтерфейс із розширеними можливостями порівняно з конструкторами. Ці платформи орієнтовані на інтеграцію з іншими сервісами та автоматизацію бізнес-процесів.

Основні функції ботів, створених через сайти**:**

– Інтеграція з CRM: Підключення до HubSpot, Salesforce.

– Автоматизація: Налаштування сценаріїв через Zapier (наприклад, надсилання замовлення в Google Sheets).

– Управління товарами: Додавання асортименту через веб-інтерфейс.

– Платежі: Інтеграція з платіжними системами через API.

– Аналітика: Відстеження дій користувачів через вбудовані інструменти.

Zapier дозволяє створювати складні автоматизації, наприклад, синхронізацію замовлень із Google Calendar для логістики, а Botmother підтримує створення чат-ботів із вбудованим AI для обробки природної мови, що підвищує якість взаємодії з клієнтами.

Переваги**:**

– Розширені інтеграції: Легке підключення до зовнішніх сервісів.

– Зручність: Інтуїтивний веб-інтерфейс для налаштування.

– Автоматизація: Можливість створення складних сценаріїв без програмування.

Недоліки:

– Вартість: Більшість платформ вимагають платну підписку для повного функціоналу.

– Обмеження кастомізації: Менша гнучкість порівняно з програмуванням.

– Залежність від сервісу: Проблеми з платформою можуть зупинити роботу бота.

Існують також інші методи, наприклад, використання готових рішень, таких як шаблони ботів на GitHub, або платформ із відкритим кодом, як Botpress. Botpress дозволяє створювати ботів із модульною архітектурою, де можна додавати функції

через плагіни, а шаблони на GitHub дають змогу адаптувати вже готові рішення під власні потреби.

Основні функції таких ботів**:**

– Модульність: Додавання функцій через плагіни (Botpress).

– Базовий функціонал: Каталог, платежі, сповіщення (GitHub-шаблони).

– Кастомізація: Адаптація коду під власні потреби.

Botpress підтримує створення ботів із функціями обробки природної мови (NLP), що дозволяє відповідати на складні запити користувачів. GitHub-шаблони часто включають інноваційні рішення, наприклад, інтеграцію з блокчейном для транзакцій.

Переваги**:**

– Гнучкість: Можливість адаптації під власні потреби.

– Безкоштовність: Більшість шаблонів і Botpress мають безкоштовні версії.

– Спільнота: Підтримка від спільноти розробників.

Недоліки**:**

– Технічні навички: Потреба в базовому розумінні коду для адаптації.

– Обмежена підтримка: Відсутність офіційної технічної підтримки.

– Ризики безпеки: Шаблони можуть містити вразливості, якщо не перевірені.

Для створення Telegram-бота для організації онлайн-продажів через чат-бот, було обрано підхід із розробкою на мові програмування, а саме Node.js. Основною причиною цього вибору стала необхідність повної кастомізації та гнучкості, які забезпечує програмування. Конструктори та спеціалізовані сайти, хоча й дозволяють швидко створити бота, мають обмеження у функціоналі та інтеграціях, що не відповідало вимогам проєкту, зокрема щодо інтеграції з кількома платіжними системами (CryptoBot, NowPayments, EasyPay) і створення складної логіки управління станами. Розробка на Node.js із бібліотекою node-telegram-bot-api дозволила реалізувати унікальні функції, такі як криптовалютні платежі через CoinGecko API, а також забезпечити високу швидкість обробки запитів завдяки асинхронній природі JavaScript. Крім того, програмування дало змогу впровадити

надійні механізми безпеки, такі як валідація даних і захист від зловмисних дій, що було б складно зробити в конструкторах чи на сайтах.

Створення Telegram-ботів для автоматизації продажів можливе різними способами, кожен із яких має свої сильні та слабкі сторони. Розробка на мовах програмування забезпечує максимальну гнучкість і продуктивність, конструктори — швидкість і простоту, спеціалізовані сайти — розширені інтеграції, а інші методи, як шаблони, пропонують економію часу. Вибір конкретного підходу залежить від потреб проєкту, бюджету та технічних навичок.

Таблиця 1.1 зведена для аналізу існуючих технологій побудови телеграм-ботів для автоматизації продажу

Таблиця 1.1

Аналіз існуючих технологій побудови телеграм-ботів для автоматизації продажу

Показник	**Розробка на мовах програмування**	**Конструктори**	**Спеціалізовані сайти**	**Веб-платформи**	**Соціальні мережі**
Каталог товарів	Гнучкий каталог із можливістюстворення складних структур, фільтрів і описів через код (наприклад, MongoDB для даних).	Базовий каталог із категоріями, обмежений шаблонами конструктора, без глибокої кастомізації	Організований перелік із можливістю інтеграції з зовнішніми сервісами (наприклад, Google Sheets через Zapier).	Вітрина товарів через Instagram Shopping із тегами	Вітрина товарів через Instagram Shopping або TikTok Shop із тегами у відео чи постах.
Оформлення замовлень	Повністю автоматизоване оформлення з кошиком, вибором кількості, інтеграцією платежів через код.	Автоматизоване через шаблони, але обмежене (вибір товару, базовий кошик через кнопки).	Автоматизоване з можливістю інтеграції з CRM для складніших сценаріїв, але залежить від платформи	Кошик із вибором доставки, оплати, автоматичним підтвердженням.	Ручне оформлення через прямі повідомлення або перехід на зовнішні платіжні сервіси.

Продовження таблиці 1.1

Аналіз існуючих технологій побудови телеграм-ботів для автоматизації продажу

Показник	**Розробка на мовах програмування**	**Конструктори**	**Спеціалізовані сайти**	**Веб-платформи**	**Соціальні мережі**
Платіжні системи	Інтеграція з будь-якими системами (CryptoBot, NowPayments, Stripe) через API, гнучке налаштування.	Обмежена інтеграція (Stripe, PayPal), залежить від конструктора.	Підтримка через API (Stripe, PayPal), але залежить від можливостей платформи.	LiqPay, PayPal, Stripe, із можливістю додавання локальних методів.	Зовнішні сервіси (LiqPay, PayPal) або прямі перекази, інтеграція обмежена
Управління асортиментом	Повноцінне управління через код (додавання, редагування, видалення товарів із бази даних).	Базове управління через інтерфейс конструктора, обмежене шаблонами.	Управління через веб-інтерфейс із можливістю інтеграції з зовнішніми сервісами.	Повноцінна адмін- панель із гнучким редагуванням асортименту.	Ручне додавання через публікації, Stories чи відео, без автоматизації.
Аналітика	Детальна аналітика через бази даних (MongoDB) і бібліотеки (наприклад, pandas у Python), аналіз транзакцій, поведінки.	Базова статистика (взаємодії, кліки), обмежена можливостями конструктора.	Аналітика через вбудовані інструмент и або інтеграцію з Google Analytics, залежить від платформи.	Google Analytics, детальна статистика відвідувань, конверсій, поведінки.	Instagram Insights, TikTok Analytics, базові дані охоплення та взаємодій.
Доступність на платформах	Telegram (Web, iOS, Android), але потребує хостингу для серверної частини.	Telegram (Web, iOS, Android), працює через сервери конструктора.	Telegram (Web, iOS, Android), залежить від інфраструктури сайту.	Web, адаптивний дизайн для всіх пристроїв (десктоп, смартфони, планшети).	Instagram, TikTok (Web, iOS, Android), обмежено платформою соцмережі.

Продовження таблиці 1.1

Аналіз існуючих технологій побудови телеграм-ботів для автоматизації продажу

Показник	**Розробка на мовах програмування**	**Конструктори**	**Спеціалізовані сайти**	**Веб-платформи**	**Соціальні мережі**
Інтерфейс	Інтуїтивний, обмежений Telegram API	Інтуїтивний, адаптований до Telegram	Інтуїтивний, із можливістю кастомізації через веб-інтерфейс, але залежить від платформи	Гнучкий, адаптивний дизайн із можливістю створення унікального UI.	Візуальний, але обмежений форматом соцмереж і (пости, Stories, відео).
Цільова аудиторія	Покупці, адміністратори, розробники, бізнеси з унікальними потребами.	Покупці, малі бізнеси, які не мають технічних навичок.	Покупці, бізнеси, які потребують інтеграцій із зовнішніми сервісами.	Широка аудиторія, бізнеси всіх рівнів, що прагнуть професійного вигляду.	Молодь, малі бізнеси, блогери, які орієнтовані на візуальний контент.
Особливості	Web3-інтеграція, AI-аналітика, висока швидкість через Node.js.	Швидке налаштування, маркетингові розсилки, AI-відповіді (ManyChat).	Складні автоматизації (Zapier), AI-обробка запитів (Botmother).	SEO-оптимізація, AI-персоналізація, PWA-технології.	Shoppable Posts (Instagram), комп’ютерне зір(TikTok).
Недоліки	Складність розробки, потреба в технічних навичках, час на створення.	Обмежена кастомізація, залежність від платформи, базовий функціонал.	Вартість платних планів, обмежена кастомізація, залежність від сервісу.	Висока вартість розробки, складність адміністрування, потреба в SEO.	Ручна обробка, обмежений функціонал, залежність від алгоритмів.

1.4 Особливості автоматизації продажів через Telegram-боти

Автоматизація продажів через Telegram-боти має унікальні особливості, які відрізняють її від інших платформ e-commerce. Розглянемо їх.

Технічні характеристики:

– Використання Telegram API для створення інтерактивного інтерфейсу з підтримкою текстових команд, інлайн-кнопок і колбеків.

– Асинхронна обробка запитів користувачів для швидкої реакції на дії.

– Інтеграція з базами даних (наприклад, MongoDB) для зберігання інформації про товари, користувачів і транзакції.

– Підтримка платіжних систем (Crypto, EasyPay, CryptoBot) для обробки транзакцій у реальному часі.

Функціональні особливості:

– Система станів: управління діалогом через стани (наприклад, вибір товару, оформлення замовлення, очікування оплати).

– Автоматизована обробка замовлень: від вибору товару до підтвердження платежу.

– Реферальна система: залучення нових клієнтів через унікальні посилання.

Інтерфейс і взаємодія:

– Обмежений, але інтуїтивний інтерфейс: меню та кнопки, адаптовані до платформи Telegram.

– Обробка текстових команд користувачів із застосуванням регулярних виразів для аналізу запитів.

– Можливість масових розсилок для інформування клієнтів про акції чи оновлення.

Безпека:

– Перевірка статусу бану користувачів перед виконанням дій.

– Валідація вхідних даних для захисту від атак.

– Захист адміністративних функцій через обмеження доступу.

2 ВИБІР ТЕХНІЧНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ ЗАСТОСУНКУ

2.1 Визначення функціональних та нефункціональних вимог до застосунку

Проаналізувавши специфіку Telegram-ботів, їхню роль в автоматизації продажів, цільову аудиторію та аналоги, було визначено наступні вимоги до Telegram-бота для продажу цифрових товарів:

Функціональні вимоги

– Авторизація: Реєстрація нових користувачів через Telegram ID, автоматичний вхід у систему при взаємодії з ботом.

– Каталог товарів: Зберігання та управління інформацією про товари (назва, ціна, опис, зображення) у базі даних MongoDB.

– Система пошуку та фільтрації товарів**:** Пошук товарів за назвою чи категорією, фільтрація за ціною або доступністю.

– Оформлення замовлень**:** Інтерактивний процес вибору товарів, вказівки кількості, додавання до кошика та підтвердження замовлення через інлайн-кнопки.

– Інтеграція з платіжними системами: Підключення до Crypto, EasyPay[13] і CryptoBot для обробки транзакцій у реальному часі.

– Система станів: Управління діалогом через стани (головне меню, вибір товару, очікування оплати тощо) для забезпечення логічної взаємодії.

– Адміністративний інтерфейс: Можливість додавання, редагування та видалення товарів, управління користувачами, створення розсилок і перегляд аналітики.

– Реферальна система: Генерація унікальних реферальних посилань для залучення нових клієнтів із бонусами.

– Аналітика продажів**:** Збір даних про транзакції, поведінку користувачів і статистику покупок для оптимізації асортименту.

– Безпека: Перевірка статусу бану користувачів, валідація вхідних даних, захист адміністративних функцій через обмеження доступу.

– Швидкий та інтуїтивний інтерфейс: Відповідь бота менш ніж за 1 секунду, інтуїтивне меню з інлайн-кнопками, адаптоване до Telegram.

Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики якості, продуктивності та зручності Telegram-бота, що забезпечують його ефективну роботу та відповідність потребам користувачів і адміністраторів.

Продуктивність**:**

– Час відповіді бота не перевищує 1 секунду навіть при одночасному зверненні до 1000 користувачів.

– Максимальна пропускна здатність — обробка 5000 транзакцій на день без затримок.

– Наявність кешування для швидкого доступу до популярних товарів і даних користувачів.

Надійність:

– Гарантована доступність бота на рівні 99,9% протягом місяця, з автоматичним перезапуском у разі збоїв.

– Збереження даних про замовлення та транзакції навіть при відключенні сервера завдяки резервному копіюванню бази даних MongoDB кожні 24 години.

– Стійкість до навантажень під час пікових періодів (наприклад, акцій чи розсилок).

Безпека:

– Використання шифрування HTTPS[6] для всіх API-запитів до Telegram і платіжних систем.

– Захист від DDoS-атак через інтеграцію з сервісами типу Cloudflare.

– Регулярне оновлення бібліотек і залежностей для усунення вразливостей (щомісяця).

– Двофакторна авторизація для доступу до адмін-панелі через унікальні коди, надіслані в Telegram.

Масштабованість:

– Можливість розширення бази даних для підтримки до 10 000 активних користувачів без втрати продуктивності.

– Підтримка горизонтального масштабування через розподілені сервери або хмарні платформи (наприклад, AWS або Heroku).

– Гнучка архітектура, що дозволяє додавати нові платіжні системи чи функції (наприклад, інтеграцію з Web3) без переписування основного коду.

Зручність використання:

– Інтерфейс адаптований до різних екранів (мобільні пристрої, десктопи), із чіткими інлайн-кнопками та короткими текстовими повідомленнями (до 200 символів).

– Підтримка багатомовності (українська, англійська, російська) із автоматичним визначенням мови користувача через Telegram API.

– Документація та інструкції всередині бота для користувачів і адміністраторів, доступні через команду /help.

Ефективність підтримки:

– Час реагування на технічні збої — не більше 2 годин із моменту повідомлення адміністратора.

– Модульна структура коду для спрощення оновлень і виправлення помилок без зупинки бота.

Компатибільність**:**

– Сумісність із усіма версіями Telegram (Web, iOS, Android) без додаткових налаштувань.

– Підтримка інтеграції з популярними CRM-системами (наприклад, HubSpot) через API для аналізу даних.

– Адаптація до змін у Telegram API (оновлення кожні 6 місяців із тестуванням на тестовому сервері).

Функціональні вимоги визначають основні можливості Telegram-бота, такі як каталог товарів, оформлення замовлень і аналітика, тоді як нефункціональні вимоги забезпечують його стабільність, безпеку та зручність. Ці вимоги

враховують специфіку роботи в Telegram, потреби цільової аудиторії (покупці, адміністратори, маркетологи) і забезпечують конкурентоспроможність на ринку автоматизації продажів.

2.2 Вибір мови програмування та фреймворків для автоматизації продажів через чат боти

В дипломній роботі була обрана мова JavaScript — високорівнева, інтерпретована мова програмування, яка широко використовується для створення інтерактивних веб-додатків і серверних систем. Завдяки асинхронному виконанню через механізми, такі як Promise і async/await, JavaScript ідеально підходить для розробки чат-ботів, зокрема Telegram-ботів.

У контексті Telegram-бота для автоматизації продажів JavaScript використовується з бібліотекою node-telegram-bot-api для обробки повідомлень і взаємодії з Telegram API. Node.js, як середовище виконання, дозволяє створювати серверні додатки, що обробляють запити в реальному часі. Простота синтаксису та велика екосистема бібліотек, таких як Express.js для створення веб-серверів і Mongoose для роботи з MongoDB, роблять JavaScript оптимальним вибором для реалізації модульних і масштабованих систем. Асинхронне програмування забезпечує швидку обробку запитів користувачів і транзакцій.

JavaScript має активну спільноту, яка регулярно оновлює бібліотеки та інструменти, що сприяє швидкому вирішенню проблем і адаптації до нових вимог. Завдяки кросплатформності JavaScript-код може виконуватися на будь-якому сервері чи пристрої, що підтримує Node.js, забезпечуючи універсальність для розробки Telegram-ботів.

Для структурування та передачі даних використовувався JSON (JavaScript Object Notation). Хоча JSON не є мовою програмування в класичному розумінні, він відіграє ключову роль у розробці Telegram-ботів, побудованих на Node.js, завдяки своїй простоті та універсальності.

У проєкті Telegram-бота для автоматизації продажів JSON використовується для зберігання та обробки даних у MongoDB, яка є нереляційною базою даних із документо-орієнтованою структурою. Наприклад, інформація про користувачів (userId, balance, purchaseStats) і товари (назва, ціна, опис) зберігається у форматі JSON-документів. JSON також застосовується для обміну даними між ботом і Telegram API, наприклад, при обробці вхідних повідомлень чи відправленні відповідей.

Переваги JSON включають читабельність, компактність і підтримку в усіх сучасних мовах програмування, що спрощує інтеграцію з Node.js, Express.js і Mongoose. Формат дозволяє легко масштабувати структуру даних, додаючи нові поля, такі як реферальні посилання чи статистика покупок, без зміни схеми бази даних. JSON є критично важливим для забезпечення швидкої та ефективної взаємодії між компонентами бота, платіжними системами та базою даних.

Для бекенду мною був обраний Node.js — це середовище виконання JavaScript, яке дозволяє створювати серверні додатки поза браузером. Запущене у 2009 році Райаном Далем, Node.js використовує асинхронну модель на основі подій, що забезпечує високу продуктивність при обробці великої кількості запитів. У контексті Telegram-бота для автоматизації продажів Node.js є основою для реалізації серверної логіки, обробки повідомлень користувачів і інтеграції з платіжними системами.

Node.js підтримує бібліотеку node-telegram-bot-api, яка забезпечує взаємодію з Telegram API, дозволяючи обробляти текстові команди, інлайн-кнопки та колбеки. Завдяки асинхронному програмуванню через механізми Promise і async/await, Node.js забезпечує швидку реакцію бота навіть при інтенсивному потоці запитів. Інтеграція з MongoDB через Mongoose дозволяє зберігати дані про користувачів (userId, balance, purchaseStats), товари та транзакції у структурованому форматі JSON. Node.js також підтримує Axios для виконання HTTP-запитів до зовнішніх сервісів, таких як платіжні системи Crypto чи EasyPay.

Переваги Node.js включають швидкість, масштабованість і велику екосистему бібліотек через npm, що спрощує додавання функціоналу, наприклад,

реферальної системи чи аналітики продажів. Активна спільнота забезпечує регулярні оновлення та підтримку, що дозволяє швидко адаптувати проєкт до нових вимог. Кросплатформність Node.js гарантує виконання коду на будь-якому сервері, що робить його універсальним для розробки Telegram-ботів.

Axios — це легка та потужна бібліотека JavaScript для виконання HTTP-запитів, яка широко використовується в Node.js-додатках для взаємодії з API. Axios підтримує асинхронні операції через Promise, що робить її ідеальною для інтеграції Telegram-бота з зовнішніми сервісами, такими як платіжні системи чи Telegram API.

У проєкті Telegram-бота для автоматизації продажів Axios використовується для надсилання запитів до Telegram API (наприклад, для відправки повідомлень, оновлення статусів чи обробки колбеків) і для інтеграції з платіжними системами, такими як CryptoBot чи EasyPay. Бібліотека спрощує обробку JSON-відповідей, що необхідно для роботи з даними про транзакції, статуси платежів чи конфігурацію бота. Axios підтримує налаштування заголовків, тайм-аутів і обробку помилок, що забезпечує надійність і безпеку при взаємодії з зовнішніми сервісами.

Переваги Axios включають простоту синтаксису, можливість перехоплення запитів і відповідей, а також підтримку паралельних запитів, що дозволяє одночасно обробляти кілька операцій, наприклад, перевірку платежів і оновлення бази даних. Легка вага бібліотеки не перевантажує проєкт, а її універсальність забезпечує сумісність із Node.js і MongoDB, що критично для швидкої та ефективної роботи бота.

2.3 Вибір бібліотеки для роботи з телеграм апі

В дипломній роботі використовується node-telegram-bot-api — це бібліотека для Node.js, яка спрощує створення Telegram-ботів шляхом надання зручного інтерфейсу для роботи з Telegram API. Вона дозволяє розробникам обробляти вхідні повідомлення, створювати інтерактивні меню, обробляти колбеки від інлайн-кнопок і надсилати відповіді користувачам.

У проєкті Telegram-бота для автоматизації продажів node-telegram-bot-api є основним інструментом для реалізації взаємодії з користувачами. Бібліотека забезпечує обробку текстових команд (наприклад, /start, /menu), створення динамічних меню для перегляду товарів і оформлення замовлень, а також підтримку системи станів (main_menu, awaiting_quantity тощо). Вона дозволяє інтегрувати бот із платіжними системами через відправку запитів про статус транзакцій і повідомлення користувачів про успішні покупки.

Переваги node-telegram-bot-api включають простоту налаштування, підтримку асинхронних операцій і гнучкість у створенні складних діалогів. Бібліотека має активну спільноту та регулярні оновлення, що забезпечує сумісність із новими функціями Telegram API. Вона також дозволяє обробляти помилки, наприклад, при некоректних командах, і забезпечує швидку реакцію бота, що критично для користувацького досвіду.

2.4 Вибір баз даних та систем управління базами даних

Mongoose — це бібліотека для Node.js, яка забезпечує об’єктно-орієнтовану взаємодію з MongoDB, нереляційною базою даних. Mongoose спрощує роботу з MongoDB, надаючи інструменти для створення схем, валідації даних і виконання запитів у зрозумілому форматі.

У дипломній роботі Telegram-бота Mongoose використовується для управління даними користувачів (userId, username, balance, purchaseStats), товарів (назва, ціна, опис) і транзакцій у MongoDB. Бібліотека дозволяє створювати чіткі схеми даних, наприклад, модель User для зберігання інформації про клієнтів і їхню історію покупок. Mongoose підтримує асинхронні запити, що забезпечує швидке оновлення даних, наприклад, при зміні балансу користувача чи додаванні нового товару.

Переваги Mongoose включають зручну роботу з JSON-подібними документами, вбудовану валідацію даних і підтримку складних запитів, таких як агрегація для аналітики продажів. Бібліотека забезпечує захист від помилок,

наприклад, при некоректному введенні даних, і спрощує інтеграцію з Node.js і Axios. Mongoose також дозволяє масштабувати базу даних, додаючи нові поля, наприклад, для реферальної системи чи статистики, без зміни структури проєкту.

MongoDB — це відкрита нереляційна система управління базами даних (СУБД), яка використовує документо-орієнтовану модель для зберігання даних у форматі JSON-подібних документів (BSON). MongoDB вирізняється своєю гнучкістю, масштабованістю та здатністю обробляти великі обсяги даних, що робить її ідеальною для веб-додатків, таких як Telegram-боти для автоматизації продажів.

MongoDB забезпечує високу продуктивність завдяки горизонтальному масштабуванню та підтримці асинхронних операцій, що дозволяє ефективно обробляти запити в реальному часі. У контексті Telegram-бота MongoDB використовується для зберігання інформації про користувачів (userId, username, balance, purchaseStats), товари (назва, ціна, опис, категорія) та транзакції. Гнучка структура документів дозволяє легко додавати нові поля, наприклад, для реферальної системи чи аналітики продажів, без зміни схеми бази даних.

MongoDB підтримує складні запити, агрегацію даних і індексацію, що спрощує аналіз продажів і поведінки користувачів. Бібліотека Mongoose, яка інтегрується з Node.js, забезпечує зручну роботу з MongoDB, надаючи схеми для валідації даних і об’єктно-орієнтований підхід до створення запитів. MongoDB також підтримує JSON-формат, що спрощує інтеграцію з Axios для обробки API-запитів і node-telegram-bot-api для збереження даних, отриманих від Telegram.

Переваги MongoDB включають гнучкість структури даних, високу швидкість роботи з великими наборами даних і підтримку хмарних рішень, таких як MongoDB Atlas, що полегшує розгортання та резервне копіювання. Це робить MongoDB оптимальним вибором для проєктів, де потрібна швидка адаптація до змін у вимогах, наприклад, додавання нових функцій до бота.

Використання Node.js**:** Забезпечує асинхронну обробку запитів, що важливо для Telegram-бота з великою кількістю одночасних користувачів. MongoDB і Mongoose: MongoDB дозволяє гнучко працювати з нереляційними даними, а

Mongoose спрощує моделювання та валідацію. Система станів: Використання state і tempData у моделі User дає змогу ефективно керувати багатоступеневими діалогами. Інтеграція платіжних систем**:** Підтримка кількох методів (CryptoBot, NowPayments, EasyPay) підвищує доступність для користувачів. Логування: Використання console.log і console.error для відстеження помилок під час розробки та тестування.

2.5 Система контролю версій

GitHub — це веб-платформа для хостингу коду, яка базується на системі контролю версій Git. GitHub стала стандартом для спільної розробки програмного забезпечення, надаючи інструменти для управління кодом, співпраці в командах і автоматизації робочих процесів. У проєкті Telegram-бота для автоматизації продажів GitHub використовується для зберігання коду, відстеження змін і забезпечення версійності.

GitHub дозволяє створювати репозиторії (публічні чи приватні) для зберігання коду, де розробники можуть фіксувати зміни через коміти, створювати гілки для паралельної розробки та об’єднувати їх через пул-реквести. Це забезпечує контроль над змінами в коді Telegram-бота, наприклад, при додаванні нових функцій, таких як реферальна система чи інтеграція з платіжними сервісами (Crypto, EasyPay). GitHub також підтримує інтеграцію з CI/CD інструментами, що спрощує автоматизацію тестування та розгортання бота.

Ключові можливості GitHub включають відстеження проблем (Issues) для управління задачами, рецензування коду через пул-реквести та спільну роботу над проєктом. Для Telegram-бота GitHub забезпечує безпечне зберігання коду, включаючи файли для Node.js, Axios, node-telegram-bot-api та Mongoose, а також конфігураційні файли для MongoDB. Платформа підтримує документацію через README-файли, що полегшує опис проєкту та його залежностей.

2.6 Інструменти проектування інтерфейсу користувача

Figma — це веб-базований інструмент для дизайну інтерфейсів і створення прототипів, який широко використовується для розробки зручних і візуально привабливих користувацьких інтерфейсів. Запущена у 2012 році, Figma вирізняється своєю хмарною архітектурою, що дозволяє працювати над дизайном із будь-якого пристрою без встановлення додаткового програмного забезпечення.

У проєкті Telegram-бота Figma використовується для створення прототипів інтерфейсу користувача, зокрема для дизайну меню, інлайн-кнопок і структури діалогів. Наприклад, у Figma можна змоделювати, як виглядатиме головне меню бота, екран вибору товарів чи підтвердження оплати. Інструмент дозволяє створювати інтерактивні прототипи з переходами між станами (наприклад, від вибору товару до введення кількості), що допомагає протестувати користувацький досвід перед реалізацією.

Figma підтримує спільну роботу, дозволяючи кільком розробникам або дизайнерам одночасно працювати над прототипом, що спрощує узгодження дизайну меню чи структури бота. Інструмент також пропонує бібліотеки компонентів, які забезпечують консистентність дизайну, наприклад, при створенні однакових стилів для кнопок чи повідомлень. Figma інтегрується з інструментами розробки, дозволяючи експортувати активи для використання в коді Telegram-бота.

Переваги Figma включають доступність через браузер, підтримку командної роботи та можливість швидкого створення прототипів. Це дозволяє ефективно планувати інтерфейс Telegram-бота, забезпечуючи зручність для користувачів і відповідність вимогам проекту.

3 ПРОЕКТУВАННЯ ЗАСТОСУНКУ ДЛЯ АВТОМАТИЗАЦІЇ ПРОДАЖІВ

3.1 Проектування архітектури застосунку

Проектування Telegram-бота для автоматизації продажів є ключовим етапом розробки, що визначає архітектуру системи, її модульність і взаємодію між компонентами. У цьому розділі розглядаються структура клієнтської та серверної частин, а також архітектура бази даних, ілюстровані відповідними діаграмами.

Застосунок Telegram-бота для автоматизації продажів реалізовано за моделлю клієнт-сервер, що дозволяє розділити обробку даних між сервером і клієнтом. Такий підхід сприяє оптимізації виконання завдань, покращує масштабованість і ефективність системи.

Клієнтська частина (фронтенд) забезпечує взаємодію з користувачами через Telegram-інтерфейс, розроблений за допомогою бібліотеки node-telegram-bot-api. Вона відправляє запити до сервера через API Telegram, отримує відповіді та динамічно оновлює вміст (наприклад, меню чи повідомлення про оплату), використовуючи JSON. Це створює зручний досвід, схожий на роботу з інтерактивними додатками.

Серверна частина (бекенд), побудована на Node.js, виконує основну логіку обробки даних. Вона приймає запити від клієнта, обробляє їх відповідно до бізнес-логіки (наприклад, перевірка балансу, обробка платежів), звертається до бази даних MongoDB для зберігання або вилучення інформації та надсилає відповіді назад. Бекенд також інтегрується з платіжними системами (CryptoBot, NowPayments, EasyPay) через API для забезпечення транзакцій.

Сервер додатково підтримує взаємодію з зовнішніми сервісами, наприклад, CoinGecko API для отримання курсів криптовалют, що дозволяє конвертувати платежі в реальному часі.

Діаграма розгортання клієнт-серверної моделі представлена на рисунку 3.1

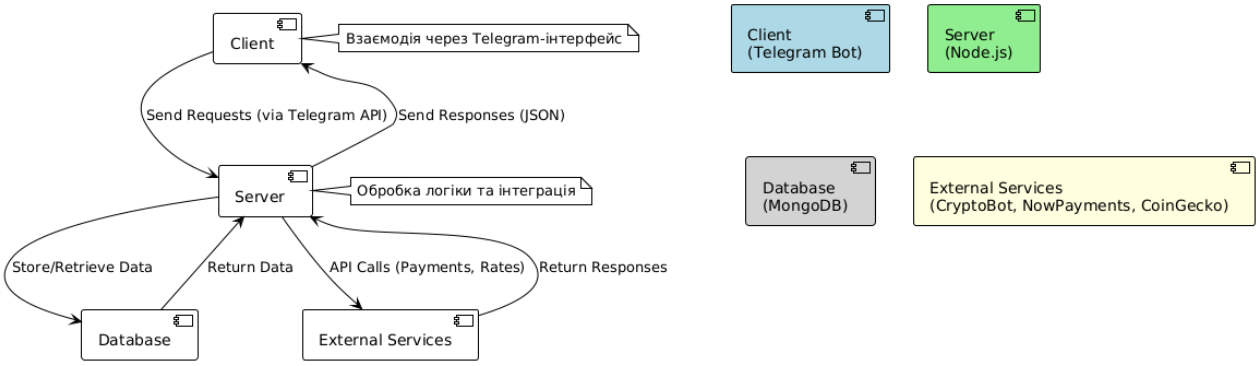


Рис. 3.1 Діаграма розгортання клієнт-серверної моделі

3.2 Архітектура клієнтської частини застосунку

Структура Telegram-бота організована у три основні частини: компоненти, сторінки та сервіси. Цей підхід забезпечує чітке розділення відповідальностей, сприяє кращій організації коду та полегшує його тестування й обслуговування.

Компоненти є будівельними блоками клієнтської частини. Вони включають логіку управління даними та шаблони для інтерактивних елементів, таких як меню, кнопки чи кошик. Наприклад, компонент menu генерує головне меню з командами, а buttons створює інлайн-кнопки для вибору товарів. Кожен компонент повторно використовується в різних частинах бота.

Сторінки формують окремі стани взаємодії, об’єднуючи компоненти. Наприклад, сторінка product_selection включає компоненти для вибору категорій і товарів, а payment_process — для оформлення покупки. Сторінки забезпечують логічну послідовність дій користувача.

Сервіси виокремлюють бізнес-логіку з компонентів, роблячи код чистішим і тестопридатним. Сервіс command_handler обробляє команди, state_manager керує станами (наприклад, очікування введення кількості), а api_integration інтегрує платіжні системи.

Така організація сприяє модульності та повторному використанню коду. Компонентно-орієнтований підхід дозволяє легко оновлювати окремі частини бота без впливу на інші, що важливо для його розширення.

Діаграма архітектури клієнтської частини зображена на рис 3.2

Рис. 3.2 Діаграма архітектури клієнтської частини

3.3 Архітектура серверної частини застосунку

Проєкт серверної частини Telegram-бота вирізняється шаруватою структурою, побудованій на Node.js. На верхньому рівні розміщені основні конфігурації (наприклад, підключення до MongoDB), а нижче — модулі, кожен із чітко визначеними завданнями.

– Моделі: Визначають структуру бази даних через Mongoose (наприклад, User, Category, Position). Це дозволяє ефективно управляти даними об’єктно-орієнтованим способом.

– Обробники (Handlers): Обробляють запити від клієнта, взаємодіють із моделями для отримання чи модифікації даних (наприклад, command_handler.js, callbackHandler.js).

– Утиліти: Містять допоміжні функції, такі як fileHelpers.js (обробка зображень) і positionManager.js (керування позиціями).

– API-інтеграція: Забезпечує зв’язок із зовнішніми сервісами (CryptoBot, NowPayments, CoinGecko) через axios.

Перевагами цієї архітектури є:

– Модульність: Чітке відокремлення модулів дозволяє масштабувати проект і розширювати його.

– Легкість управління: Інтеграція з MongoDB забезпечує ефективну обробку даних.

– Швидка розробка: В икористання Node.js спрощує створення асинхронних додатків.

На рисунку 3.3 зображена діаграма компонентів.

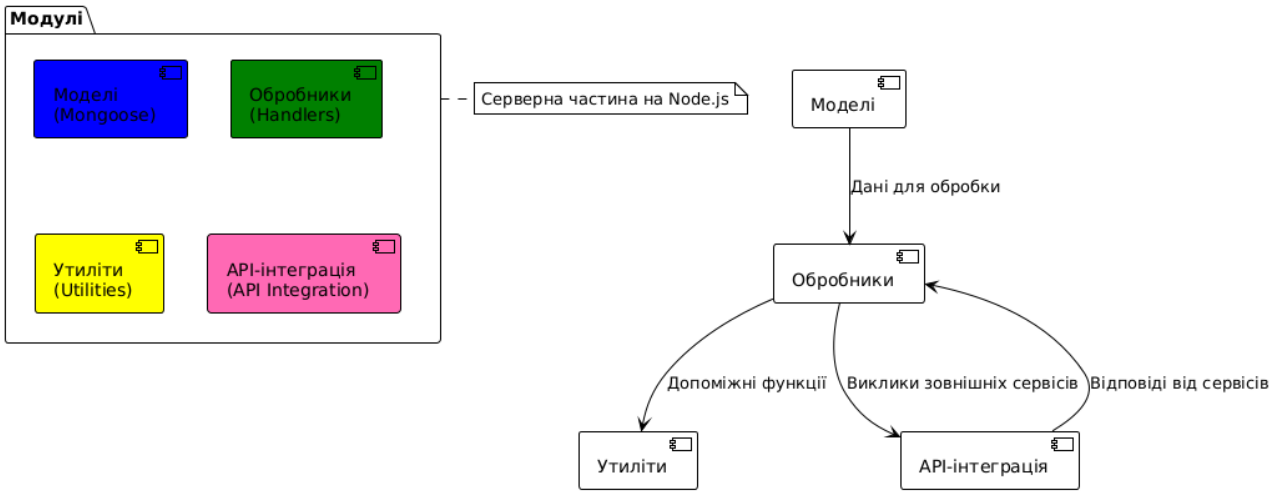


Рис. 3.3 Діаграма компонентів

3.4 Проектування баз даних застосунку

Архітектура бази даних побудована на MongoDB, що використовує нереляційну модель для гнучкого зберігання даних. База складається з колекцій, які містять дані про користувачів, товари та статистику.

Колекції:

– users: Профілі та баланс користувачів.

– categories: Категорії товарів.

– positions: Позиції з цінами та зображеннями.

– productitems: Конкретні товари з інформацією про покупця.

– botstats: Статистика бота.

– supports: Дані про підтримку.

– addedproducts: Лог додавання товарів.

Відносини:

– positions пов’язані з categories через category_id.

– productitems пов’язані з positions (positionId), categories (categoryId) та users (adminId, buyerId).

– addedproducts пов’язані з users (userId) і positions (positionId).

Оптимізація: Індекси на userId, positionId, categoryId для швидкого доступу.

Переваги:

– Гнучкість: Легке додавання нових полів.

– Швидкість: Індекси оптимізують запити.

– Масштабованість: Підходить для зростання кількості даних.

Діаграма класів бази даних зображена на рис 3.4

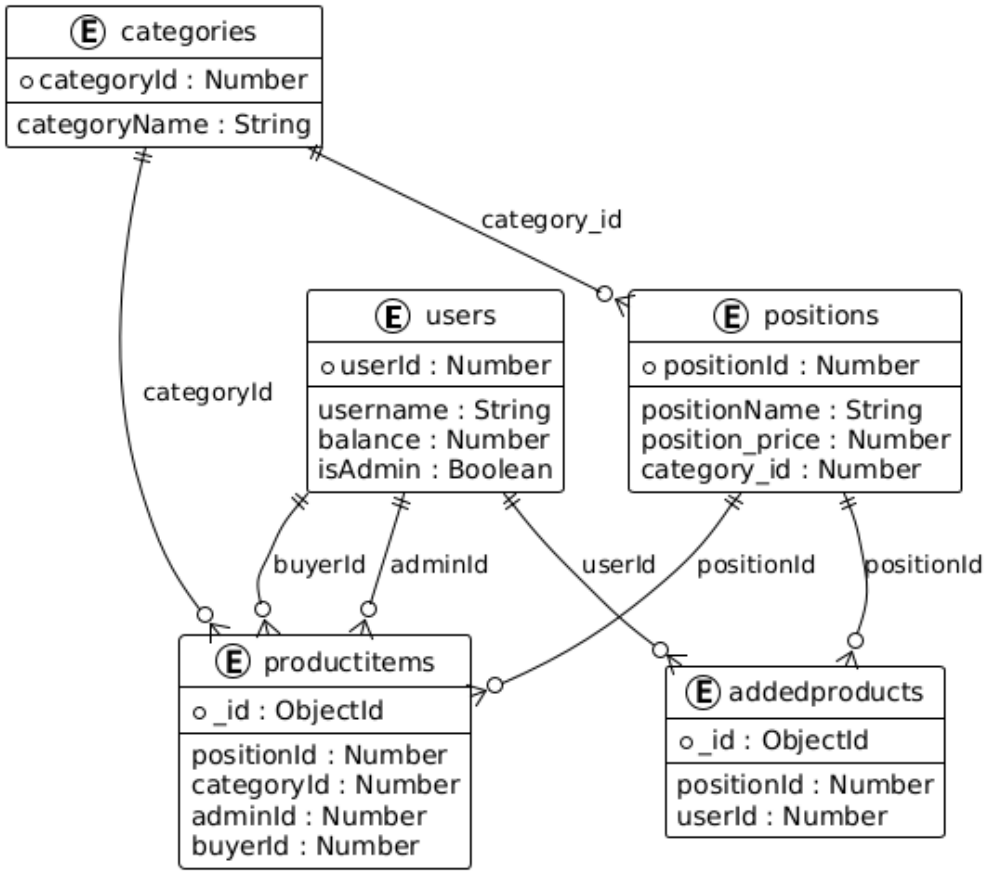


Рис. 3.4 Діаграма класів

Серверна частина використовує MongoDB через бібліотеку Mongoose для моделювання даних. Основні моделі включають:

– User: Зберігає інформацію про користувачів (userId, username, balance, isAdmin, isBanned, state, totalDeposited, purchaseStats, tempData, createdAt).

– Category: Містить категорії товарів (categoryName, categoryId).

– Position**:** Описує позиції товарів (positionName, positionId, position_price, position_discription, position_image, category_id).

– ProductItem: Зберігає конкретні товари (positionId, categoryId, adminId, content, isUsed, buyerId, usedAt).

– BotStats**:** Статистика бота (totalUsers, totalSales24h, totalDeposits24h тощо).

– Support: Інформація про підтримку та бота (support, information).

– AddedProduct**:** Лог додавання товарів (positionId, userId, products).

На рисунку 3.5 зображена модель користувача в базі даних.

Рис. 3.5 Модель користувача в базі даних

Підключення до бази даних**:** Модуль connect.js забезпечує підключення до MongoDB через змінні середовища (MONGO_URI). У разі помилки підключення процес завершується.

API-інтеграція**:**

– CryptoBot: Створення інвойсів і перевірка платежів через API (pay.crypt.bot).

– CoinGecko**:** Отримання актуальних курсів криптовалют для конверсії UAH у USDT, BTC тощо.

– NowPayments: Створення платежів через USDT (TRC20) із підтримкою тестового режиму.

– EasyPay: Заглушка для інтеграції (ручна перевірка платежів із номером платежу).

Обробка файлів**:** Модуль fileHelpers.js генерує унікальні імена для зображень (pos_{timestamp}_{random}.jpg) і зберігає їх у директорії uploads.

Утиліти:

– helpers.js: Містить функції для форматування дати (formatDate), перевірки команд меню (isMenuCommand).

– positionManager.js: Керує станом створення позицій через Map.

На рисунку 3.6 зображена діаграма серверної послідовності покупки товару у боті.

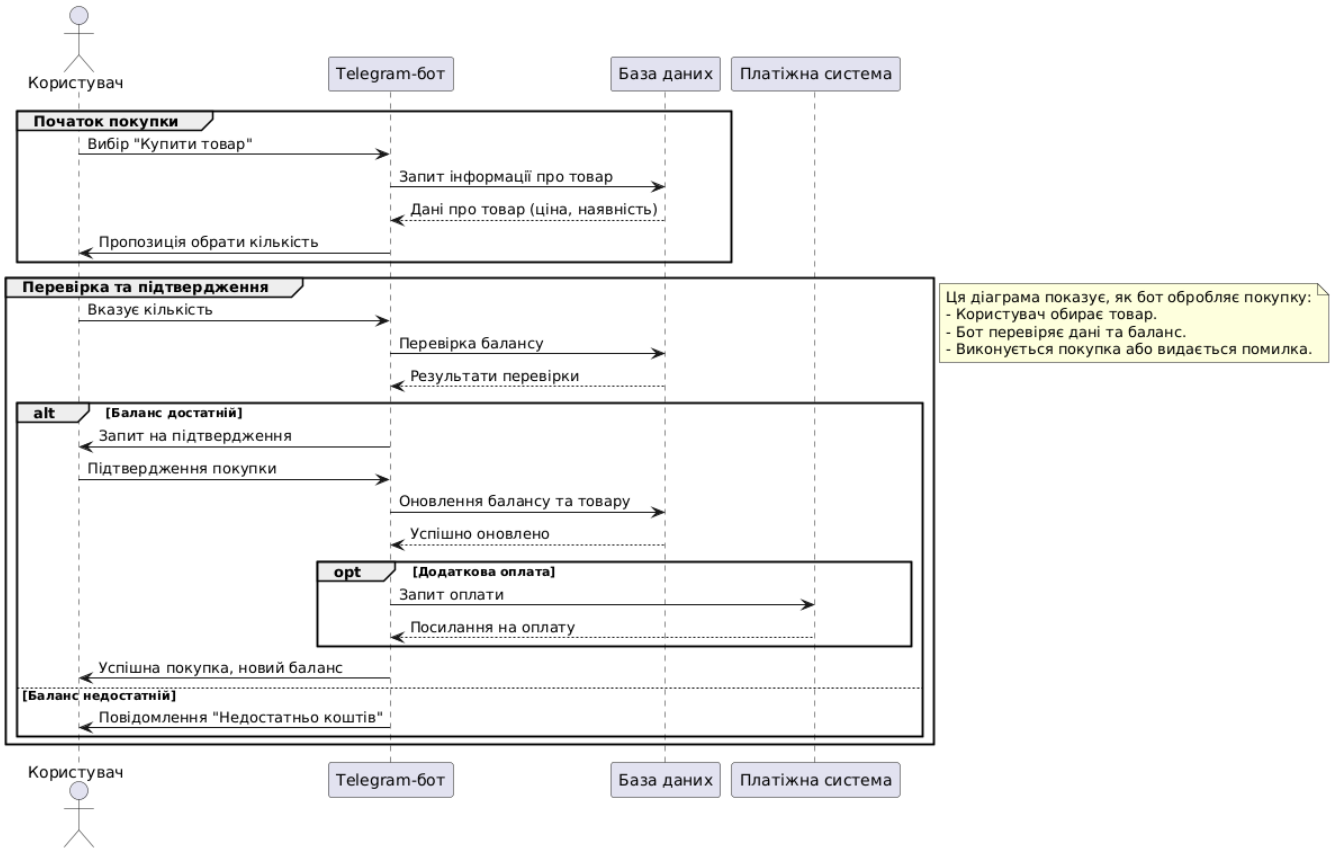


Рис. 3.6 Діаграма серверної послідовності покупки товару у боті.

Серверна частина забезпечує:

– Масштабованість**:** Завдяки модульній структурі та MongoDB легко додавати нові моделі даних чи інтеграції.

– Надійність**:** Обробка помилок через try-catch і логи (console.error) допомагають уникати збоїв.

– Гнучкість**:** Інтеграція з кількома платіжними системами (CryptoBot, NowPayments) дозволяє розширювати функціонал.

Для зображення взаємодії користувача з ботом наведено діаграму варіантів використання. Актор (користувач) взаємодіє з системою через основні функції: перегляд товарів, покупка, поповнення балансу та адмін-панель.

На рисунку 3.7 зображена діаграма варіантів використання

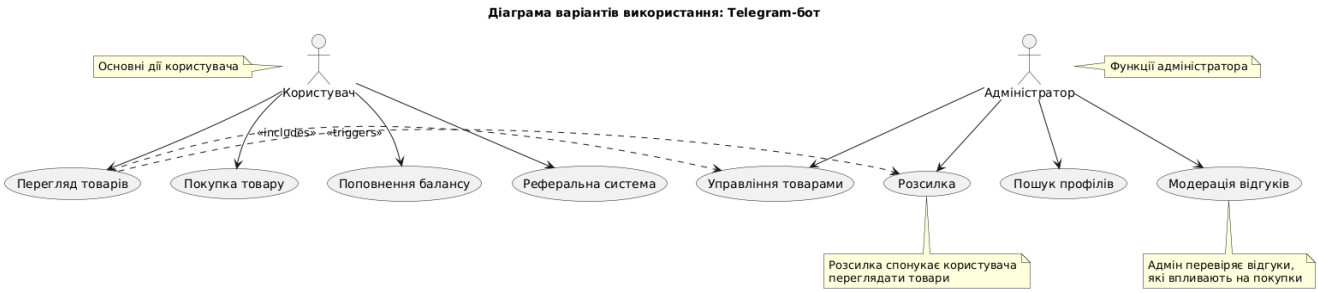


Рис. 3.7 Діаграма варіантів використання

4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

4.1 Дизайн інтерфесу

Telegram-бот для автоматизації продажів взаємодіє з користувачами через текстовий інтерфейс, який складається з повідомлень, меню та інлайн-кнопок. Дизайн інтерфейсу бота розроблявся з урахуванням зручності користувача, інтуїтивності навігації та швидкості доступу до основних функцій. Для цього використовувалася бібліотека node-telegram-bot-api, яка дозволяє створювати інтерактивні елементи, такі як кнопки та меню.

Основні кроки створення інтерфейсу:

Ініціалізація бота та створення головного меню:

– Бот створено через BotFather у Telegram, отримано токен API.

– У файлі start.js реалізовано команду /start, яка надсилає привітальне повідомлення та головне меню з кнопками: Товари, Профіль, Підтримка. Реалізовано через метод bot.sendMessage із параметром reply_markup для створення клавіатури.

На рисунку 4.1 зображено головне меню бота.

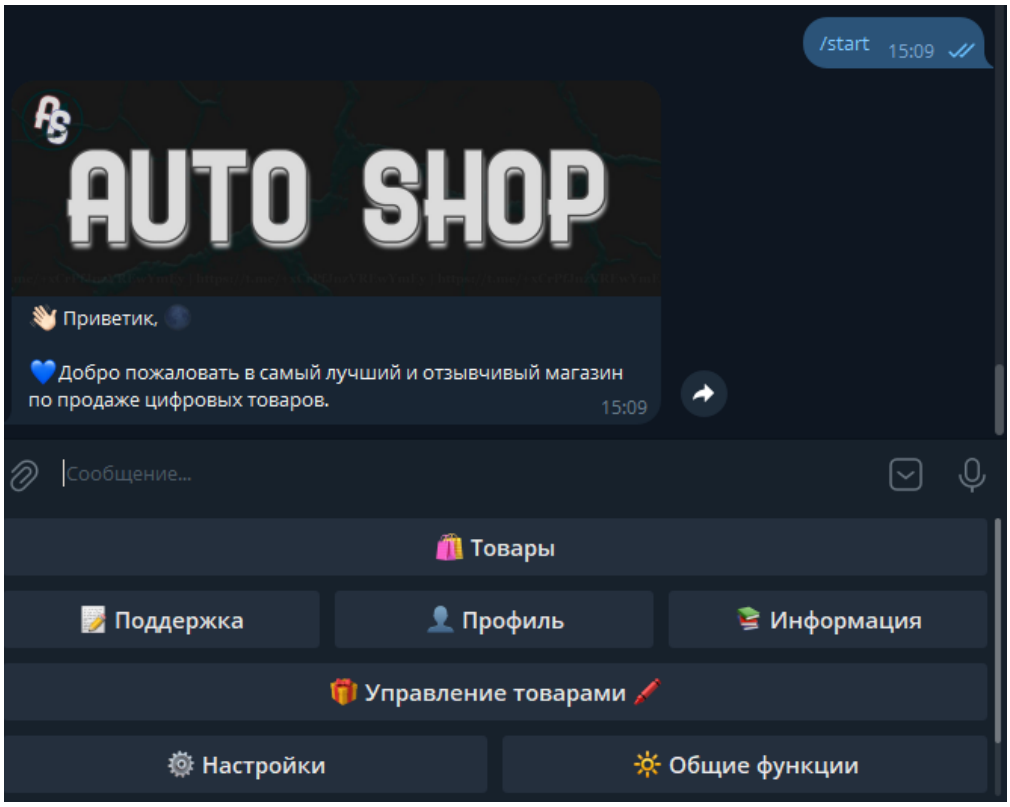


Рис. 4.1 Екрана форма головного меню.

Розробка інлайн-кнопок для навігації:

– У файлі products.js реалізовано вибір категорій і товарів через інлайн-кнопки. Наприклад, після вибору «Товари» бот відображає список категорій із inline_keyboard, де кожна кнопка має callback_data для обробки дії.

– Кнопки створено за допомогою методу bot.sendMessage із параметром reply_markup.

Створення повідомлень із зображеннями та текстом**:**

– Для відображення товарів у products.js використано метод bot.sendPhoto, який надсилає зображення позиції разом із описом (назва, ціна, кількість).

– Наприклад, при виборі товару:

На рисунку 4.2 зображена форма обраного товару для придбання.

Рис. 4.2 Екрана форма обраного товару для придбання

Розробка адмін-інтерфейсу:

– У файлі admin/index.js створено меню для адміністраторів із функціями: додавання товарів, розсилка, пошук профілів.

– Наприклад, після команди «управління товарами» бот надсилає:

На рисунку 4.3 зображене адмін-меню з управлінням товарів.

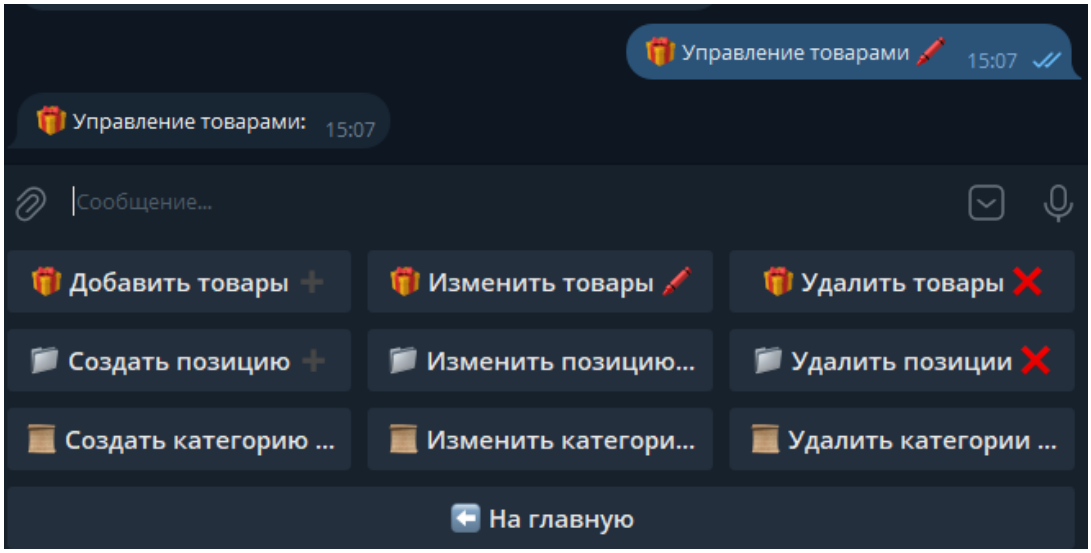


Рис. 4.3 Екрана форма адмін панелі

4.2 Реалізація основних функцій

Реалізація покупки товару.

Функція покупки (файл products.js) включає:

– Вибір категорії та позиції через інлайн-кнопки.

– Запит кількості товару з використанням стану awaiting_quantity у моделі User.

– Перевірка балансу через запит до MongoDB.

– Якщо баланс достатній, товар списується, баланс оновлюється, і бот надсилає:

На рисунку 4.4 зображено те як виглядає придбання товару.

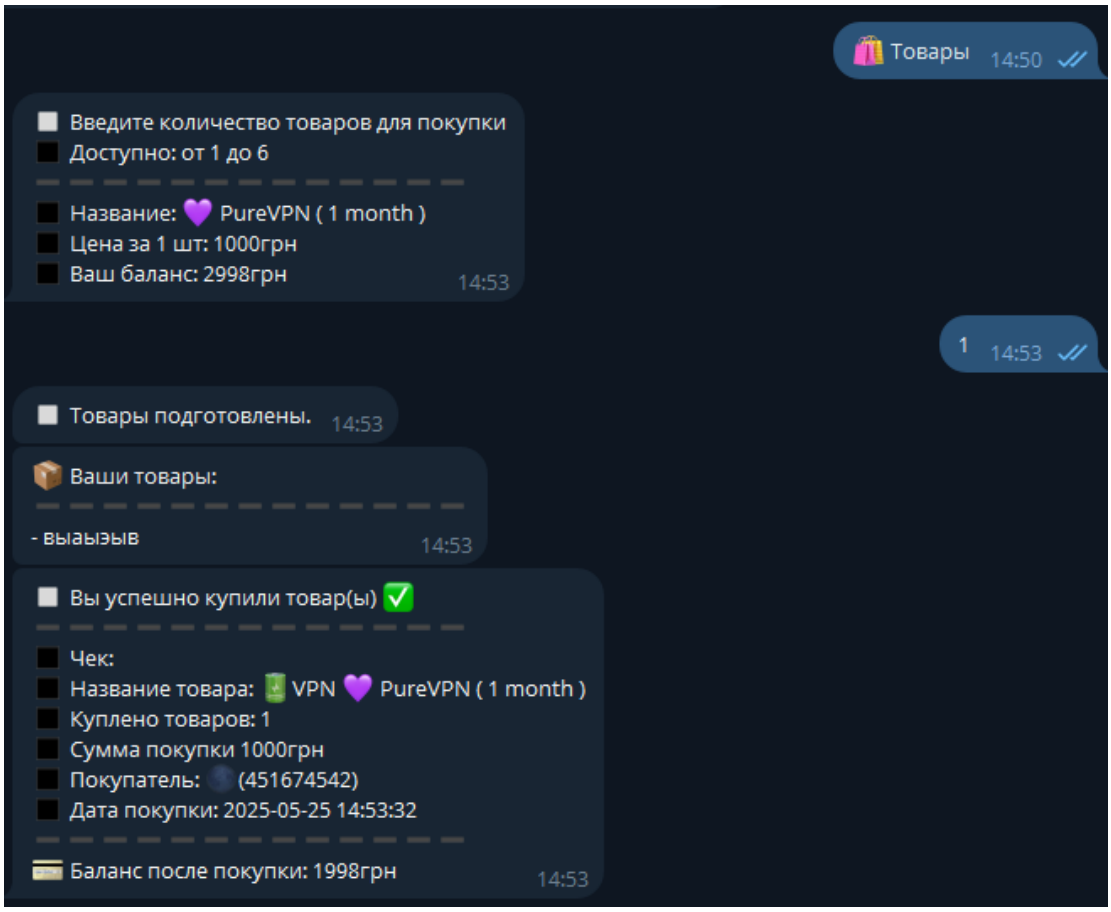


Рис. 4.4 Екрана форма придбання товару

Реалізація поповнення балансу

Поповнення (файли cryptoBotPayment.js, cryptoDeposit.js, easyPayDeposit.js) підтримує:

– CryptoBot**:** Генерація інвойсу для USDT, BTC через API CryptoBot із конверсією через CoinGecko.

– NowPayments**:** Підтримка USDT (TRC20) із тестовим режимом.

– EasyPay**:** Генерація номера платежу з комісією 2% для ручної оплати. Після оплати баланс оновлюється в реальному часі.

Реалізація адмін-функцій

Адмін-функції (файл admin/index.js) включають:

– Додавання категорій і товарів через діалоги (стани awaiting_category_name, adding_products).

– Розсилка повідомлень через команду /mailing.

– Пошук профілів за userId.

4.3 Реалізація серверної частини

Ініціалізація проєкту**:**

Проєкт створено з використанням Node.js. Структура включає файли:

– index.js: Точка входу для бота.

– connect.js: Підключення до MongoDB через mongoose.

– .env: Зберігає змінні середовища (токен, API-ключі).

Створення модулів:

– Кожен функціональний блок (наприклад, products.js, admin/index.js) є окремим модулем.

– Модулі підключено через require або import для модульності.

Реалізація моделей**:**

– У файлі models/User.js створено модель User із полями: userId, balance, isAdmin.

– Приклад моделі Position у models/Position.js:

На рисунку 4.5 зображено програмний код моделі з бд

```
const positionSchema = new mongoose.Schema({
    positionName: {
        type: String,
        required: true,
        unique: true,
        validate: {
            validator: function(v) {
                return v && v.trim().length > 0;
            },
            message: 'Название позиции не может быть пустым'
        }
    },
    positionId: { type: Number, unique: true },
    position_price: {
        type: Number,
        required: true,
        min: [0.01, 'Цена должна быть больше 0']
    },
    position_discription: {
        type: String,
        required: true,
        validate: {
            validator: function(v) {
                return v && v.trim().length > 0;
            },
            message: 'Описание не может быть пустым'
        }
    },
    position_image: {
        type: String,
        validate: {
            validator: function(v) {
                // Позволяем null или строку с расширением файла
                return v === null || /\.(jpg|jpeg|png|gif)$/i.test(v);
            },
            message: 'Неподдерживаемый формат изображения'
        }
    },
    position_date: { type: Date, default: Date.now },
    category_id: { type: Number, required: true }
});

positionSchema.pre('save', async function(next) {
    if (!this.isNew) return next();

    try {
        const lastPosition = await this.constructor.findOne({}, {}, { sort: { positionId: -1 } });
        this.positionId = lastPosition ? lastPosition.positionId + 1 : 1;
        next();
    } catch (error) {
        next(error);
    }
});
```

Рис. 4.5 Модель Position

Обробка запитів:

– У callbackHandler.js реалізовано обробку інлайн-кнопок (наприклад, покупка товару).

– У messageHandler.js — обробку текстових команд.

Інтеграція з API:

– У cryptoBotPayment.js реалізовано виклики API CryptoBot для створення інвойсів.

– У cryptoDeposit.js — інтеграція з NowPayments.

Запуск проєкту:

– Використано команду node index.js для запуску бота.

– Бот доступний через Telegram за адресою @YourBotName.

На рисунку 4.6 зображена структура проекту.

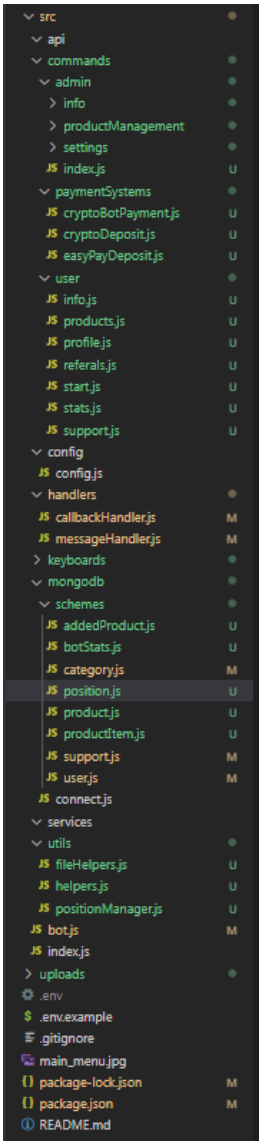


Рис. 4.6 Структура проекту

4.5 Завантаження проєкту на GitHub

Для розміщення проєкту Telegram-бота для автоматизації продажів у репозиторії на GitHub необхідно виконати кілька кроків, які включають ініціалізацію Git, додавання файлів до репозиторію та завантаження змін до GitHub.

Спочатку відкриваємо термінал у кореневому каталозі проєкту. Далі ініціалізуємо Git-репозиторій командою:

git init

Перед додаванням файлів створюємо файл .gitignore, щоб виключити завантаження чутливих даних, таких як файл .env із токеном бота та API-ключами. У файл .gitignore додаємо:

.env node_modules/ uploads/

Далі додаємо URL існуючого репозиторію GitHub як віддалений репозиторій. Наприклад:

git remote add origin https://github.com/YourUsername/Telegram-Sales-Bot.git

Наступним кроком додаємо всі файли проєкту до індексу Git:

git add .

Створюємо коміт із повідомленням, яке описує зміни:

git commit -m "Initial commit of Telegram Sales Bot"

Останнім кроком завантажуємо зміни до віддаленого репозиторію на GitHub:

git push -u origin main

4.6 Тестування застосунку

Для перевірки коректності роботи серверної частини Telegram-бота виконано модульне тестування. Це дозволило протестувати окремі компоненти системи (моделі, функції) ізольовано від інших, що забезпечило стабільність і надійність роботи.

Переваги модульного тестування

– Ізольованість: Тести перевіряють окремі функції чи модулі, що спрощує виявлення помилок.

– Автоматизація: Тести можна запускати автоматично, прискорюючи перевірку змін у коді.

– Покращення якості коду**:** Написання тестів сприяє створенню більш надійного коду.

– Запобігання регресії**:** Тести допомагають переконатися, що нові зміни не ламають існуючий функціонал.

Опис тестів

Для тестування використано фреймворк Mocha та бібліотеку Chai для асерцій. Тести перевіряють базові операції з моделями User, Category, Position та функцію покупки товару. Нижче наведено приклад тестів, реалізованих у файлі tests/models.test.js.

Пояснення тестів

before**:** Підключення до тестової бази MongoDB перед запуском тестів.

beforeEach: Очистка бази перед кожним тестом для ізоляції.

after: Відключення від бази після завершення тестів.

User Model: Перевірка створення користувача з коректними полями.

Category Model: Перевірка створення категорії.

Position Model**:** Перевірка створення позиції товару.

Purchase Function: Перевірка функції покупки — успішний випадок із достатнім балансом і невдалий із недостатнім.

Запуск тестів

Для запуску тестів використано команду:

mocha tests/models.test.js

Результат у терміналі:

Models and Functions Tests

User Model

✓ should create a user with correct fields

Category Model

✓ should create a category with correct fields

Position Model

✓ should create a position with correct fields

Purchase Function

✓ should successfully purchase a product if balance is sufficient

✓ should fail to purchase if balance is insufficient

Усі тести завершилися зі статусом "passing", що підтверджує коректність роботи моделей та функцій.

Результатом виконаної роботи стала розробка Telegram-бота для автоматизації продажів, що значно спрощує процес купівлі товарів через Telegram-інтерфейс. Бот дозволяє користувачам швидко переглядати товари, купувати їх та поповнювати баланс, а адміністраторам — ефективно керувати асортиментом і користувачами.

ВИСНОВКИ

У ході роботи проведено аналіз сучасних технологій автоматизації електронних продажів, включаючи веб-платформи, маркетплейси, соціальні мережі та месенджер-боти. Веб-платформи вирізняються гнучкістю, але потребують значних витрат, маркетплейси (Rozetka, Prom.ua) забезпечують швидкий доступ до аудиторії, але мають високу конкуренцію, а соціальні мережі (Instagram, TikTok) ефективні для візуального контенту з обмеженим функціоналом. Telegram-боти запропоновано як економічне рішення з персоналізованим підходом, попри обмеження API.

Досліджено методи розробки Telegram-ботів, включаючи мови програмування (Node.js), конструктори (BotFather, ManyChat) та сайти (Zapier). Обрано кастомний метод на Node.js через гнучкість і можливість інтеграції з платіжними системами (CryptoBot, NowPayments, EasyPay) та CoinGecko API для криптовалютних платежів, що долає обмеження конструкторів і сайтів.

Визначено функціональні вимоги (авторизація, каталог товарів, оформлення замовлень, аналітика, адмін-функції) та нефункціональні (швидкість, безпека, зручність, масштабованість), що сформували основу проєкту. Використано Node.js як мову програмування, бібліотеку node-telegram-bot-api для Telegram API, MongoDB для бази даних і GitHub для контNoролю версій.

Розроблено архітектуру з клієнтською (інтерфейс у Telegram) і серверною (обробка запитів на Node.js) частинами, а також базою даних для зберігання даних. Реалізовано дизайн інтерфейсу з меню та інлайн-кнопками, основні функції (покупка, кошик, реферальна система), інтеграцію з платіжними системами та адмін-панель. Проєкт завантажено на GitHub і протестовано на коректність роботи, усунувши дрібні помилки, як-от затримки при завантаженні.

Розроблений Telegram-бот відповідає вимогам, забезпечуючи зручність для користувачів і адміністраторів. Він має потенціал для розширення, наприклад, додаванням нових платіжних систем чи аналітики на базі AI, і може бути

застосований для автоматизації продажів у нішевих ринках, особливо для малого бізнесу.

Робота пройшла апробацію. За її результатами було опубліковано наступні тези доповідей:

1. Каломбет В.С., Щербина І.С. Роль телеграм ботів в процесі продажу цифрових товарів. VI Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». Збірник тез. 24 квітня 2025 р., м.Київ. К.:ДУІКТ., С.517-519.

ПЕРЕЛІК ПОСИЛАНЬ

1. Node.js documentation, Node.js. URL: https://nodejs.org/en/docs (date of access: 01.03.2025)

2. MongoDB documentation. MongoDB. URL: https://www.mongodb.com/docs/ (date of access: 05.03.2025).

3. Telegram Bot API documentation. Telegram. URL: https://core.telegram.org/bots/api (date of access: 10.03.2025).

4. Brown T. JavaScript: The definitive guide. O'Reilly Media.

5. Haverbeke M. Eloquent JavaScript: A modern introduction to programming. No Starch Press, 2020. URL: https://eloquentjavascript.net/ (date of access: 15.03.2025).

6. CryptoBot API documentation. CryptoBot. URL: https://help.crypt.bot/crypto-pay-api (date of access: 20.03.2025).

7. NowPayments API documentation. NowPayments. URL: https://docs.nowpayments.io/ (date of access: 20.03.2025).

8. CoinGecko API documentation. CoinGecko. URL: https://www.coingecko.com/en/api/documentation (date of access: 25.03.2025).

9. Mocha documentation. Mocha. URL: https://mochajs.org/ (date of access: 01.04.2025).

10. Chai documentation. Chai. URL: https://www.chaijs.com/ (date of access: 01.04.2025).

11. Express.js documentation. Express. URL: https://expressjs.com/ (date of access: 03.04.2025).

12. Mongoose documentation. Mongoose. URL: https://mongoosejs.com/docs/ (date of access: 05.04.2025).

13. JavaScript MDN Web Docs. Mozilla. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (date of access: 07.04.2025).

14. JSON Web Token (JWT) documentation. JWT.io. URL: https://jwt.io/introduction (date of access: 10.04.2025).

15. Postman Learning Center. Postman. URL: https://learning.postman.com/docs/getting-started/introduction/ (date of access: 12.04.2025).

16. Git documentation. Git-scm. URL: https://git-scm.com/doc (date of access: 15.04.2025).

17. GitHub Docs. GitHub. URL: https://docs.github.com/en (date of access: 17.04.2025).

18. Webhooks Guide. Stripe. URL: https://stripe.com/docs/webhooks (date of access: 20.04.2025).

19. REST API Tutorial. REST API Tutorial. URL: https://restfulapi.net/ (date of access: 22.04.2025).

20. OpenAPI Specification. Swagger. URL: https://swagger.io/specification/ (date of access: 25.04.2025)

ДОДАТОК А. ДЕМОНСТРАЦІЙНИЙ МАТЕРІАЛ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка програмних засобів для організації онлайн-продажів через чат-бот

Виконав: студент групи ПД-42 Каломбет Владислав Станіславович
Керівник: к.т.н, доцент кафедри Щербина Ірина Сергіївна

Київ – 2025

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи -** автоматизація процесу продажу цифрових товарів з підтримкою криптовалютних оплат та адмініструванням через внутрішню панель керування за допомогую телеграм бота.
- **Об’єкт дослідження -** процес електронної комерції з використанням месенджерів як інструменту продажу цифрових продуктів.
- **Предмет дослідження -** архітектура, функціональність і технічні особливості Telegram-бота для продажу цифрових товарів, включаючи механізм оплати, систему управління товарами та взаємодію з користувачем.

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати актуальні підходи до реалізації автоматизованих систем електронної комерції у месенджерах.
2. Визначити вимоги до функціональності Telegram-бота для продажу цифрових товарів.
3. Розробити архітектуру програмного забезпечення з урахуванням підтримки декількох платіжних систем.
4. Розробити Telegram-бота з можливістю створення та редагування товарів через адміністративну панель.
5. Забезпечити захищене збереження даних користувачів і транзакцій за допомогою бази даних MongoDB.
6. Провести тестування системи та оцінити її зручність використання, стабільність і ефективність.

3

АНАЛІЗ АНАЛОГІВ

Назва платформи/бота	Платформи оплати	Наявність адмін-панелі	Автоматична видача товару	Гнучкість керування товарами	Недоліки
ShopBot	Картки,PayPal	Так, веб-панель	Підтверджується вручну	Середня	Складна інтеграція
TranzzoBot	Картки,крипта	Лише через командний інтерфейс	Так	Низька	Відсутність повного контролю
AutoGoodsBot	Тільки крипта	Лише команди	Так	Середня	Немає масштабованості
EasyBot	**Crypto, CryptoBot, карта (в перспективу)**	**Вбудована телеграм адмінка**	**Так, повністю автоматизована**	**Висока (через адмінку)**	**Нереалізована оплата карткою**

4

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги:

1. Реалізація Telegram-бота з підтримкою обробки повідомлень та кнопок.
2. Автоматизована видача цифрових товарів після оплати.
3. Інтеграція з кількома платіжними системами: криптовалюта (через API), CryptoBot, банківські картки (у перспективі).
4. Адміністративна панель з функціями:
    1. Створення, редагування, видалення категорій, позицій і товарів.
    2. Перегляд статистики продажів.
    3. Контроль за балансом і платежами користувачів.
5. Збереження історії замовлень та управління товарами через базу даних MongoDB.

Нефункціональні вимоги:

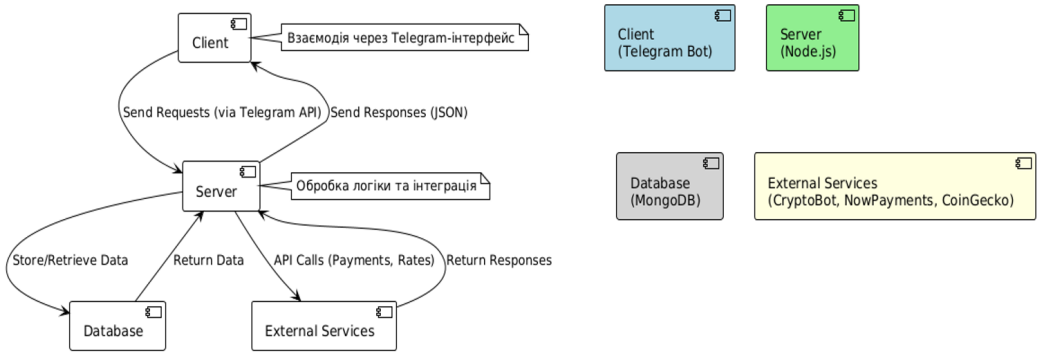
1. Швидкість відповіді бота не більше 1 сек.
2. Надійність обробки запитів при високому навантаженні (одночасне обслуговування ≥ 50 користувачів).
3. Захищене зберігання користувацьких даних і шифрування чутливої інформації.
4. Масштабованість архітектури — можливість додавання нових платіжних модулів без повної переробки системи.
5. Мінімізація втручання адміністратора в процес обробки замовлень.

5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

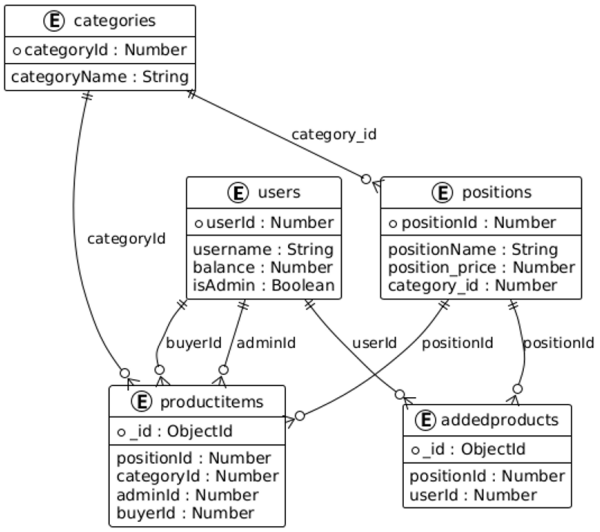
6

ДІАГРАМА РОЗГОРТАННЯ КЛІЄНТ-СЕРВЕРНОЇ МОДЕЛІ



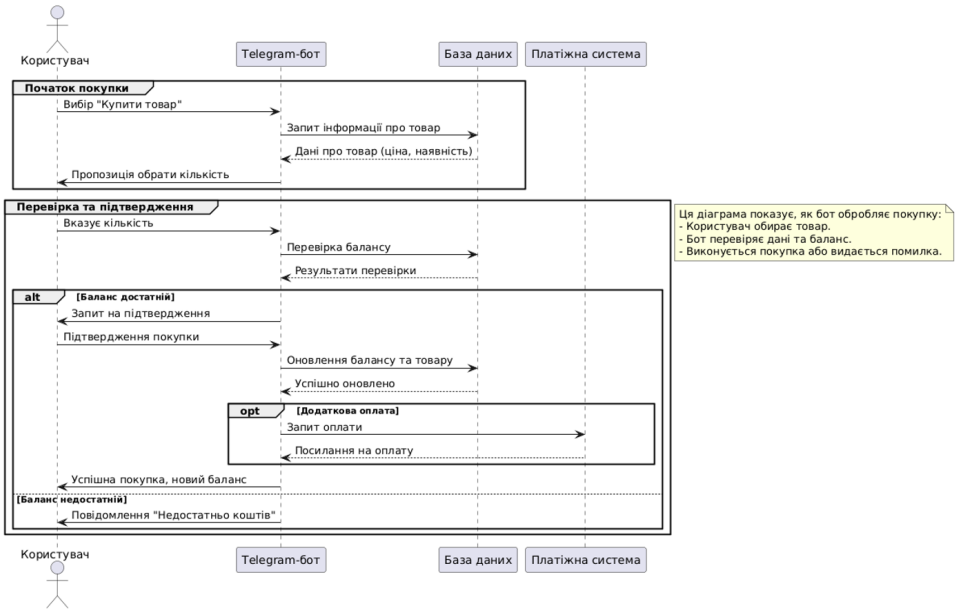
7

ДІАГРАМА КЛАСІВ



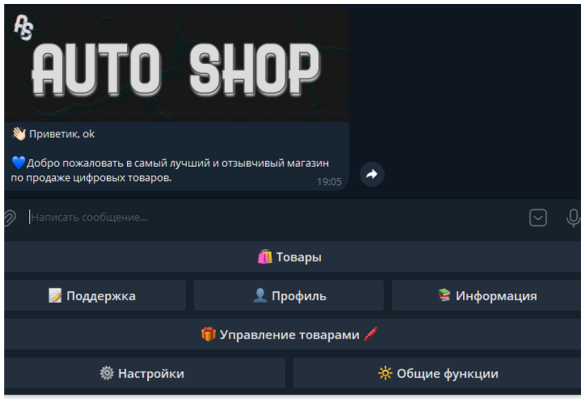
8

ДІАГРАМА СЕРВЕРНОЇ ПОСЛІДОВНОСТІ ПОКУПКИ ТОВАРУ У БОТІ



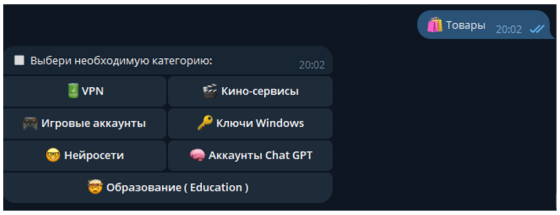
ОСНОВНІ КОМПОНЕНТИ ТЕЛЕГРАМ БОТУ

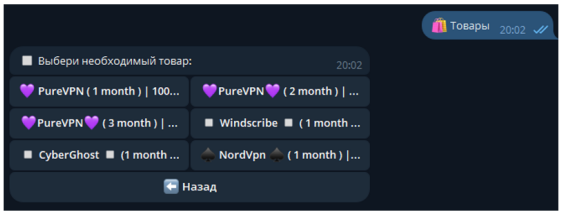
ЕКРАННІ ФОРМИ

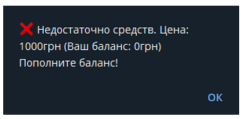


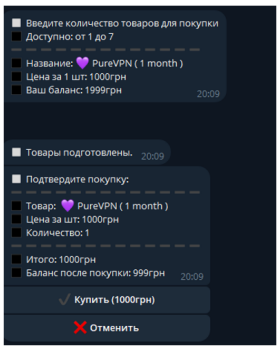
11

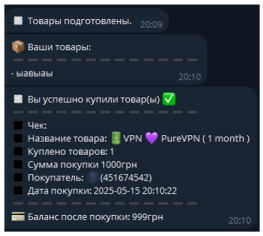
ЕКРАННІ ФОРМИ





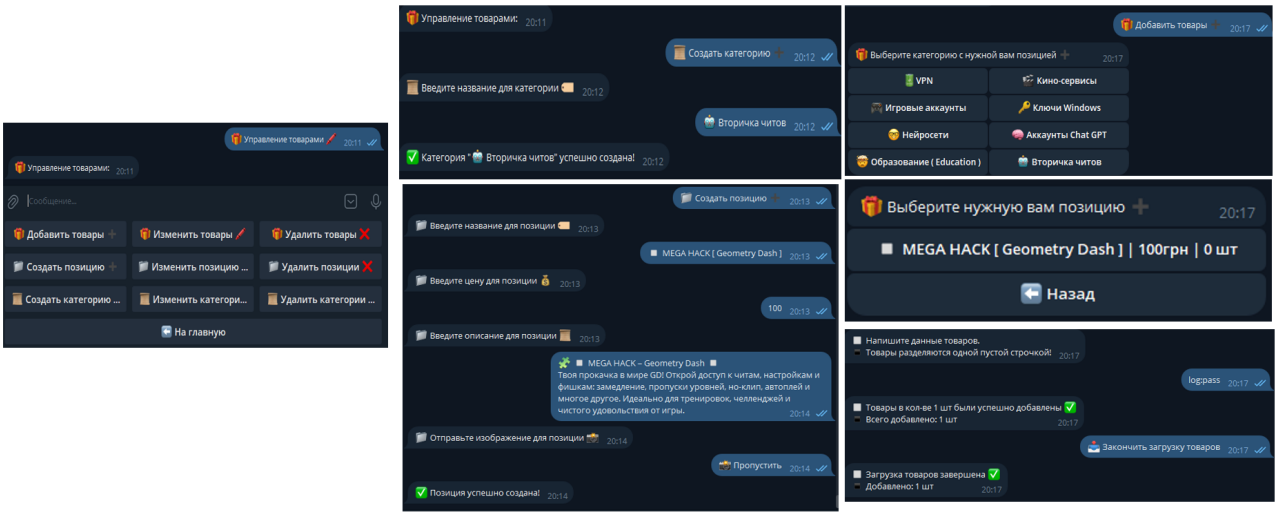






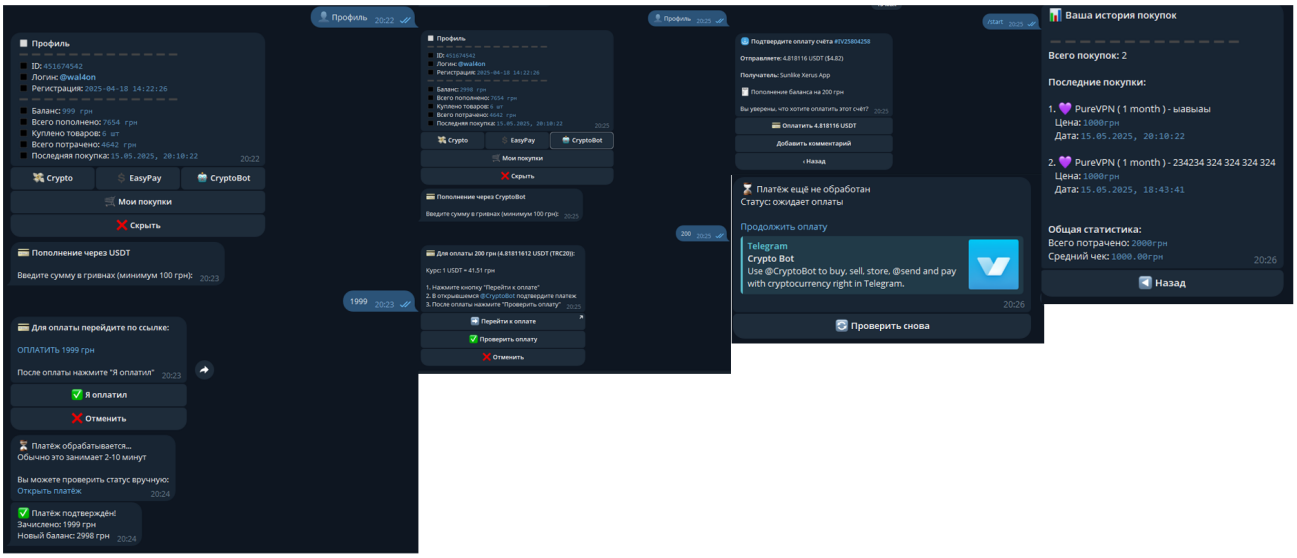
12

ЕКРАННІ ФОРМИ



13

ЕКРАННІ ФОРМИ



14

ВИСНОВКИ

1. Проаналізовано сучасні підходи до реалізації систем електронної комерції у месенджерах
Під час практики було вивчено специфіку реалізації електронної комерції за допомогою чат-ботів. Проаналізовано приклади існуючих Telegram-ботів, їх функціональні можливості, підходи до обробки платежів та взаємодії з користувачами. Особливу увагу приділено забезпеченню автоматизації продажів і зручності інтерфейсу. Зроблено висновок, що Telegram є ефективною платформою для впровадження комерційних рішень завдяки простоті, популярності та доступу до API. Отримані результати стали основою для побудови власного рішення.
2. Визначено вимоги до функціональності Telegram-бота для продажу цифрових товарів
На основі аналізу предметної галузі сформовано перелік обов'язкових функцій для бота. До них увійшли: перегляд каталогу товарів, покупка товару, інтеграція з платіжними системами, миттєва доставка файлів після оплати, облік замовлень та адмін-панель. Визначено вимоги до безпеки, стабільності, швидкодії та простоти використання. Це дозволило перейти до наступного етапу — розробки архітектури програмного продукту. Усі функції було реалізовано згідно з цими вимогами.
3. Розроблено архітектуру Telegram-бота з підтримкою кількох платіжних систем
Спроєктовано структуру програмного забезпечення з урахуванням масштабованості, модульності та інтеграції з різними API. Бот підтримує оплату через криптовалютні сервіси (Cryptomus, NowPayments) та банківські картки. Архітектура дозволяє легко додавати нові способи оплати у майбутньому. Усі платежі перевіряються через API без використання вебхуків, що підвищує безпеку. Передбачено обробку помилок, перевірку транзакцій та автоматичну доставку товарів після підтвердження оплати.
4. Реалізовано Telegram-бот з адмін-панеллю для керування товарами
Створено функціональну Telegram-адмінку, яка дозволяє адміністратору додавати, редагувати та видаляти товари, переглядати замовлення та вручну видавати товари у разі потреби. Адмін-панель побудована на inline-кнопках та меню, що робить її зручною у користуванні без додаткових інтерфейсів. Забезпечено авторизацію для обмеження доступу до адмінських функцій. Система дозволяє повністю керувати магазином через мобільний телефон. Це робить рішення універсальним для малого та середнього бізнесу.
5. Забезпечено безпечне зберігання даних і проведено тестування системи
Усі дані користувачів, транзакцій та товарів зберігаються у MongoDB. Реалізовано захист від несанкціонованого доступу, валідацію даних та резервування ключових операцій. Проведено тестування всіх сценаріїв роботи бота — від додавання товарів до реальних оплат та доставки. Система показала стабільну роботу при великій кількості взаємодій. Отримано позитивні відгуки щодо зручності використання, швидкодії та інтуїтивного інтерфейсу.

15

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Каломбет В.С., Щербина І.С. Роль телеграм ботів в процесі продажу цифрових товарів. VI Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». (24 квітня 2025 р.). – Київ: ДУІКТ, 2021 – 2025.

16

ДОДАТОК Б. ЛІСТИНГ ПРОГРАМНОГО КОДУ

src\bot.js:

```
import dotenv from "dotenv";
import TelegramBot from "node-telegram-bot-api";
import connectDB from './mongodb/connect.js';
import messageHandler from "./handlers/messagehandler.js";
import callbackHandler from "./handlers/callbackHandler.js";
import User from "./mongodb/schemes/user.js";

dotenv.config();

const bot = new TelegramBot(process.env.TOKEN, { polling: true });

connectDB()
  .then(() => console.log("MongoDB connected"))
  .catch((err) => console.log("MongoDB connection error:", err));

bot.on("message", async (msg) => {
  try {
    if (!msg.from || !msg.from.id) return;

    const user = await User.findOne({ userId: msg.from.id });
    console.log(user);
    if (user?.isBanned && !msg.text?.startsWith('/start')) {
      return bot.sendMessage(
        msg.chat.id,
        "🚫 Ваш аккаунт заблокирован. Обратитесь к администратору."
      );
    }

    await messageHandler(bot, msg);
  } catch (error) {
    console.error('ГЛОБАЛЬНАЯ ОШИБКА:', error.stack);
  }
});

bot.on("callback_query", async (query) => {
  try {
```

```
        const user = await User.findOne({ userId: query.from.id });
        console.log(user);
        if (user?.isBanned) {
            await bot.answerCallbackQuery(query.id, {
                text: "🚫 Ваш аккаунт заблокирован",
                show_alert: true
            });
            return;
        }

        await callbackHandler(bot, query);
    } catch (error) {
        console.error('ГЛОБАЛЬНАЯ ОШИБКА callback:', error.stack);
    }
});

export { bot };
```

src\commands\admin\index.js:

```
import { handleProductManagement } from "./productManagement/index.js";
import { adminProductManagementMenu } from "../../keyboards/menu.js";
import showBotInfo from "./info/info.js";
import { handleSettings } from "./settings/index.js";
import { handleProfileSearchCommand } from "./info/searchProfile.js";
import { handleMailingCommand } from "./info/mailing.js";

export async function handleAdminCommands(bot, msg, user) {
    const text = msg.text;
    const chatId = msg.chat.id;

    switch (text) {
        case "🎁 Управление товарами ✏️":
            await bot.sendMessage(chatId, "🎁 Управление товарами:", {
                reply_markup: {
                    keyboard: adminProductManagementMenu,
                    resize_keyboard: true,
                },
            });
            break;

        case "📰 Информация о боте":
```

```
    await showBotInfo(bot, chatId);
    break;
  case "⚙ Настройки":
    await handleSettings(bot, msg, user);
    break;
  case "❁ Общие функции":
    await bot.sendMessage(chatId, "❁ Выберите нужную функцию.", {
      reply_markup: {
        keyboard: [["📱 Поиск профиля 🔍", "📣 Рассылка"]],
        resize_keyboard: true
      }
    });
    break;
  case "📱 Поиск профиля 🔍":
    await handleProfileSearchCommand(bot, msg, user);
    break;
  case "📣 Рассылка":
    await handleMailingCommand(bot, msg, user);
    break;

  default:
    await handleProductManagement(bot, msg, user);
    break;
```