ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНИХ СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів на мові програмування PHP»

на здобуття освітнього ступеня бакалавра

зі спеціальності 126 «Інформаційні системи та технології»

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

_____ Назар БОЖОК

(підпис)

Виконав: здобувач вищої освіти гр.ІСД-41
Назар БОЖОК

Керівник: Каміла СТОРЧАК

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення автоматизованих систем

Ступінь вищої освіти - бакалавр

Спеціальність «126 Інформаційні системи та технології»

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ

Завідувач кафедрою ІПЗАС
Каміла Сторчак
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
Божку Назару Юрійовичу

(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: Розробка телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів на мові програмування PHP

керівник кваліфікаційної роботи: д.т.н., проф. Каміла СТОРЧАК

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024 р. № 36

2. Строк подання кваліфікаційної роботи «___» _____ 20__ р.

3. Вихідні дані до кваліфікаційної роботи:

1. Архітектура телеграм-боту для криптопослуг.
2. Технології інтеграції з платіжними системами.

3. Наукова література та веб-джерела.

4. Документація Telegram Bot API.

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

1. Дослідити Telegram як платформу для надання цифрових послуг.
2. Проаналізувати мови програмування та бази даних.
3. Розробити програмне рішення.

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання: «___» _____ 20__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз науково-технічної літератури	11.04.2024 р.	виконано
2	Загальна концепція телеграм-боту	15.04.2024 р.	виконано
3	Аналіз особливостей розробки та технологій	17.04.2024 р.	виконано
4	Дослідження мови програмування PHP	19.04.2024 р.	виконано
5	Дослідження баз даних	21.04.2024 р.	виконано
6	Вибір платіжних систем	30.04.2024 р.	виконано
7	Розробка програмного рішення	01.05.2024р.	виконано
8	Оформлення дипломної роботи	05.05.2024р.	виконано

Здобувач вищої освіти Назар БОЖОК

(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи Каміла СТОРЧАК

(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 80 стор., 40 рис., 16 джерел.

Мета дослідження - розробка та інтеграція системи криптоплатежів у телеграм-боті. Проаналізувати можливості автоматизації процесів продажу та обробки платежів за допомогою бота для забезпечення ефективності та зручності для користувачів. Визначити переваги та недоліки використання телеграм-боту для обробки криптоплатежів у порівнянні з іншими методами оплати.

Об'єкт дослідження **-** процес розробки та використання телеграм-боту для автоматизації продажів та обробки криптоплатежів.

Предмет дослідження - телеграм-бот для надання криптопослуг з автоматичною обробкою криптоплатежів.

У рамках дослідження було розроблено телеграм-бота для надання криптопослуг, який включає автоматичну обробку криптоплатежів, використовуючи мову програмування PHP. Розробка бота здійснювалася через інтеграцію з блокчейн технологіями та використання API. Процес включав налаштування бази даних в PHPMyAdmin для зберігання та управління даними, розробку інтерфейсу бота, функціоналу для проведення платежів у криптовалюті. Розглянуто ключові аспекти та значення криптовалют та блокчейн технологій, а також роль Telegram. Вибір інструментів та технологій охопив аналіз мов програмування, баз даних та специфіку використання платіжних систем для автоматизації криптоплатежів.

КЛЮЧОВІ СЛОВА: TELEGRAM BOT, КРИПТОВАЛЮТА, КРИПТОПЛАТЕЖІ, БАЗА ДАНИХ, PHP, CRYPTO BOT, API, ТОКЕН, БЛОКЧЕЙН.

ABSTRACT

Textual part of the qualification work for obtaining a bachelor's degree: 80 pages, 40 figures, 16 sources.

The purpose of the study is to develop and integrate a crypto payment system in a Telegram bot. To analyze the possibilities of automating sales and payment processing processes using a bot to ensure efficiency and convenience for users. Identify the advantages and disadvantages of using a telegram bot to process crypto payments compared to other payment methods.

Object of research - the process of developing and using a Telegram bot for automating sales and processing crypto payments.

The subject of the study is a telegram bot for the provision of crypto services with automatic processing of crypto payments.

As part of the study, a telegram bot was developed to provide crypto services, including automatic processing of crypto payments, using the PHP programming language. The bot was developed through integration with blockchain technologies and the use of APIs. The process included setting up a database in PHPMyAdmin for storing and managing data, developing a bot interface, and functionality for making payments in cryptocurrency. The key aspects and significance of cryptocurrencies and blockchain technologies, as well as the role of Telegram, were considered. The choice of tools and technologies included the analysis of programming languages, databases, and the specifics of using payment systems to automate crypto payments.

KEYWORDS: TELEGRAM BOT, CRYPTOCURRENCY, CRYPTO PAYMENTS, DATABASE, PHP, CRYPTO BOT, API, TOKEN, BLOCKCHAIN

ЗМІСТ

ВСТУП

Актуальність теми. Тема дипломної роботи - “Розробка телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів на мові програмування PHP” є вкрай актуальною в сучасному світі цифрової економіки та фінансів. Останнім часом спостерігається значний ріст інтересу до криптовалют, які стають все більш популярним інвестиційним та платіжним інструментом. У такому контексті розвиток інноваційних фінансових сервісів, що використовують криптовалюти, є ключовим напрямком.

Телеграм-боти, як зручний та доступний інструмент комунікації, стають ефективним засобом для автоматизації процесу продажу та обробки криптоплатежів. Розробка бота для продажу з використанням криптовалют дозволить не лише спростити та прискорити процес оплати для клієнтів, але й підвищить конкурентоспроможність бізнесу на ринку. Крім того, такий проект дозволить студенту не лише поглибити свої знання у сфері програмування та фінансів, але й реалізувати свої навички у сфері розробки прогресивних інформаційних технологій на практиці. Таким чином, тема розробки телеграм-бота для продажу з обробкою криптоплатежів є важливою і перспективною як для академічного дослідження, так і для практичного застосування у сфері бізнесу та фінансів.

Об’єкт дослідження - процес розробки та використання телеграм-боту для автоматизації продажів та обробки криптоплатежів.

Предмет дослідження - Розробки телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів.

Мета дослідження - Дослідження можливостей та ефективності використання телеграм-боту для організації продажів товарів або послуг з використанням криптовалютних платежів. Вивчення технологічних аспектів розробки та інтеграції системи криптоплатежів у телеграм-боті. Аналіз

можливостей автоматизації процесів продажу та обробки платежів за допомогою бота для забезпечення ефективності та зручності для користувачів. Визначення переваг та недоліків використання телеграм-боту для обробки криптоплатежів у порівнянні з іншими методами оплати.

Відповідно до мети було поставлено наступні **завдання:**

- Охарактеризувати явище та поняття технології блокчейн;
- Дослідити процес і можливості створення телеграм-ботів;
- Проаналізувати доступні мови програмування;
- Проаналізувати доступні системи баз даних;
- Розробити програмне рішення;
- Протестувати розроблену програму;

Структура роботи. Робота складається з переліку позначень, трьох розділів, дванадцяти підрозділів, висновків та списку використаних джерел. Загальний обсяг роботи - 80 сторінок.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API - Application Programming Interface.

API-відповідь - Відповідь від API.

API-запит - Запит до API.

API-ключ - Application Programming Interface Key.

БД - База Даних.

CSV - Comma-Separated Values.

DB - Database.

ERC-20 - Ethereum Request for Comment 20.

ETH - Ethereum.

GUI - Graphical User Interface.

HTTP - Hypertext Transfer Protocol.

HTTPS - Hypertext Transfer Protocol Secure.

HTTPS-запит - Запит, відправлений за допомогою захищеного протоколу передачі гіпертексту.

JSON - JavaScript Object Notation.

SQL - Structured Query Language.

SSL - Secure Sockets Layer.

TLS - Transport Layer Security.

ТГ-бот - Телеграм-бот.

Токен - Token.

USD - United States Dollar.

XML - eXtensible Markup Language.

РОЗДІЛ 1. ТЕОРЕТИЧНИЙ ОГЛЯД

1.1 Огляд криптовалют та їхнє значення у сучасному світі

Криптовалюта - це вид цифрової валюти, яка використовує криптографію для забезпечення безпеки транзакцій та контролю за створенням нових одиниць. Цей інноваційний фінансовий інструмент виник у 2009 році з появою Bitcoin, який був представлений у вигляді децентралізованої електронної готівки Сатоші Накамото. Розвиток Bitcoin та інших криптовалют кардинально змінив уявлення про гроші та фінансові взаємовідносини, впровадивши концепцію фінансової системи без центрального регулюючого органу [1].

Криптовалюти відіграють важливу роль у модернізації глобальних фінансових систем. Вони сприяють створенню більш інклюзивної фінансової системи, доступної людям без традиційних банківських рахунків. Крім того, криптовалюти можуть зменшити вартість і час трансграничних переказів, що робить їх особливо привабливими для глобальної економіки [2].

Для аналізу криптовалют важливо розглядати різні типи цих активів. Окрім Bitcoin, який часто називають "золотим стандартом" криптовалют, існують альтернативні криптовалюти (altcoins), такі як Ethereum, Toncoin та Litecoin. Кожна з цих криптовалют має унікальні функціональні характеристики, наприклад, Ethereum пропонує розширені можливості через смарт-контракти, що дозволяє автоматизувати багато фінансових операцій [3].

Новітні блокчейни, такі як Solana, Ethereum та Cosmos, використовують інноваційні консенсусні механізми, які дозволяють значно збільшити пропускну спроможність мережі, зменшуючи час обробки транзакцій та забезпечуючи високу швидкість виконання операцій. Наприклад, використання протоколу доведення частки (Proof of Stake) та його варіантів може зменшити енергетичні

витрати та покращити ефективність в порівнянні з традиційним механізмом доведення роботи (Proof of Work), що використовується Bitcoin [4].

Економічний вплив криптовалют спостерігається не тільки у фінансовому секторі, але й у багатьох інших аспектах сучасної економіки. Їхня здатність працювати в рамках децентралізованої мережі пропонує нові можливості для створення цифрових економік, що базуються на принципах прозорості, безпеки та ефективності [5].

1.2 Принципи роботи блокчейн технологій

Блокчейн технологія є фундаментальною основою для криптовалют і представляє собою розподілену базу даних, що зберігає всі транзакції у вигляді "блоків", які хронологічно зв'язані один з одним і захищені за допомогою криптографічних методів. Ця технологія була вперше використана в Bitcoin, але з часом її потенціал почав використовуватись у різноманітних секторах економіки [6].

Однією з ключових характеристик блокчейну є його здатність до децентралізації. Відмовляючись від централізованого контролю, така структура дозволяє усім учасникам мережі мати доступ до однакової інформації, що значно ускладнює несанкціоновані зміни даних. Блокчейн забезпечує високий рівень безпеки та надійності, оскільки змінити інформацію в одному блоку потребує змін у всіх попередніх блоках, що вимагає значних обчислювальних ресурсів [7].

Крім того, блокчейн пропонує високий рівень прозорості. Оскільки кожен блок містить унікальну інформацію та посилання на попередній блок, вся історія транзакцій є доступною і перевіриною для всіх учасників мережі. Це створює безпечне середовище, де відсутній ризик подвійних витрат або фальсифікацій [8].

Блокчейн технологія поділяється на кілька типів, залежно від способу доступу до мережі: публічні, приватні та консорціумні блокчейни. Публічні блокчейни, такі як Bitcoin і Ethereum, дозволяють будь-якому користувачу приєднатися до мережі та участі в процесі верифікації транзакцій (Antonopoulos, 2014). Приватні блокчейни використовуються в корпоративних цілях, де доступ до мережі мають лише певні особи. Консорціумні блокчейни управляються групою організацій, що надають баланс між децентралізацією публічних та контрольованим доступом приватних блокчейнів [9].

Завдяки своїм унікальним властивостям, блокчейн знайшов застосування не тільки в фінансових сервісах, але й у таких сферах, як ланцюги постачання, охорона здоров'я, державне управління та багато інших, де важлива надійність та прозорість обміну інформацією [10].

1.3 Роль і місце Telegram у наданні цифрових послуг

Telegram значно посилив свою роль у сфері цифрових послуг, інтегрувавши власний гаманець у користувацький інтерфейс, що спростило управління фінансовими операціями для користувачів. Ця інновація не лише зробила платформу більш зручною для щоденного використання, але й значно розширила можливості для проведення криптовалютних транзакцій, підвищуючи інтеграцію цифрових валют у повсякденне життя користувачів [11].

Крім того, на платформі Telegram активно функціонує велика кількість ботів, призначених для здійснення криптоплатежів, таких як Crypto Bot. Цей бот використовує функцію Crypto Pay, яка дозволяє інтегрувати оплату криптовалютою в комерційні продукти через API. Така можливість надає бізнесам простий інструмент для приймання платежів у криптовалюті, що є особливо корисним у контексті глобальної економіки, де криптовалюта займає все більш вагому позицію.

Інновації Telegram у сфері цифрових послуг не лише сприяють зростанню використання криптовалют, але й відкривають нові можливості для розвитку цифрової економіки. Інтеграція криптовалютних транзакцій, поєднана з високим рівнем безпеки та приватності, який забезпечує Telegram, створює надійну платформу для електронної комерції та фінансових інновацій [12].

Ця розширена інтеграція технологічних новацій робить Telegram важливим інструментом у сфері надання цифрових послуг, підвищуючи його популярність і корисність для користувачів і бізнесу на міжнародному рівні.

1.4 Історія і розвиток ботів у Telegram

Telegram боти з'явилися як важливий елемент платформи не випадково. З самого початку своєї появи в 2013 році, Telegram активно розширював свої функціональні можливості, прагнучи забезпечити користувачам інноваційні інструменти для комунікації та взаємодії. Введення ботів у 2015 році стало значним кроком у розвитку платформи, дозволяючи розробникам та користувачам створювати спеціалізовані інтерфейси для автоматизації взаємодій.

Боти в Telegram можуть виконувати широкий спектр завдань, від простого надсилання повідомлень до складних операцій, таких як обробка платежів, управління задачами, резервування та інші комерційні функції. Вони програмуються через спеціальний API, що дозволяє Telegram стати не просто месенджером, а потужною платформою для розробки.

Особливе місце серед ботів займає використання криптовалютних технологій. Наприклад, Crypto Bot та інші схожі інструменти забезпечують користувачів можливістю проводити криптовалютні транзакції безпосередньо через інтерфейс Telegram. Використання функції Crypto Pay через API цих ботів спрощує інтеграцію криптовалютних платежів у різноманітні продукти та

сервіси, відкриваючи нові горизонти для електронної комерції та цифрових фінансів.

Завдяки своїм можливостям, боти в Telegram суттєво вплинули на спосіб взаємодії користувачів з цифровими послугами. Вони забезпечують значну автоматизацію процесів і знижують поріг вхідних бар'єрів для бізнесів, які хочуть використовувати новітні технології. Це допомагає підприємствам у всьому світі збільшувати свою конкурентоспроможність та надавати користувачам вищий рівень сервісу [13].

РОЗДІЛ 2. ВИБІР ІНСТРУМЕНТІВ І ТЕХНОЛОГІЙ

2.1 Аналіз мов програмування

Python - це високорівнева, інтерпретована, динамічна мова програмування, яка здатна вирішувати різноманітні завдання, від простих сценаріїв до складних веб-додатків та наукових обчислень.

Розглянемо переваги Python:

Простий синтаксис - Python має зрозумілий і простий для вивчення синтаксис, що робить його ідеальним вибором для початківців та професіоналів.

Широка підтримка спільноти - Python має велику та активну спільноту, яка регулярно розвиває бібліотеки та фреймворки, що полегшують розробку програм.

Багато бібліотек - Python має велику екосистему бібліотек для різних цілей, таких як робота з базами даних, обробка даних, машинне навчання, веб-розробка тощо.

Портативність - Python підтримується на багатьох платформах, включаючи Windows, macOS і різні дистрибутиви Linux, що робить його універсальним і доступним для розробників.

Швидкість розробки - Python дозволяє розробникам швидко прототипувати та впроваджувати програми завдяки своїй простоті та зручному синтаксису.

Недоліки Python:

Виконання на великих об'ємах даних - у порівнянні з деякими іншими мовами програмування, такими як C++ або Java, Python може бути повільним у виконанні на великих об'ємах даних.

Неідеальна підтримка паралельного програмування - хоча Python має модулі для паралельного програмування, такі як multiprocessing та threading, вони не завжди ефективні через особливості інтерпретатора Python.

Не підходить для деяких задач високої продуктивності - у деяких областях, де вимагається висока продуктивність, таких як вбудовані системи або ігри, Python може бути не найкращим вибором через свою інтерпретовану природу.

Незважаючи на ці недоліки, Python залишається однією з найпопулярніших мов програмування завдяки своїм перевагам у сфері швидкості розробки, доступності та потужності [14].

PHP (Hypertext Preprocessor) - це мова програмування, яка використовується для розробки веб-додатків та веб-сайтів. Ось детальний огляд переваг і недоліків використання PHP:

Простота вивчення - PHP має простий і зрозумілий синтаксис, що робить його досить легким для вивчення, навіть для початківців.

Широке використання - PHP є однією з найпоширеніших мов програмування для веб-розробки і використовується для створення різних типів веб-додатків, від простих блогів до великих електронних комерційних платформ.

Багатофункціональність - PHP має велику кількість вбудованих функцій та розширень, що полегшує роботу з базами даних, обробкою форм, створенням зображень тощо.

Зручність інтеграції - PHP добре інтегрується з різними СУБД, такими як MySQL, PostgreSQL, SQLite тощо, що дозволяє розробникам легко працювати з базами даних.

Велика спільнота - PHP має велику та активну спільноту розробників, що забезпечує широкий доступ до документації, форумів підтримки та багато інших ресурсів.

Недоліки PHP:

Безпека - PHP має деякі вразливості, такі як вразливості типу переповнення буфера та вразливості SQL-ін'єкції, що можуть стати причиною атак на веб-додатки.

Низька продуктивність - У порівнянні з деякими іншими мовами програмування, такими як Java або C++, PHP може мати меншу продуктивність та швидкодію, особливо для великих та складних додатків.

Неявна типізація - PHP має слабку підтримку статичної типізації, що може призвести до непередбачуваного поведінки програми та помилок.

Нестабільність функціоналу - історично PHP мала проблеми зі стабільністю деяких функцій та розширень, що може призвести до проблем при оновленні або міграції додатків на нові версії PHP.

Хоча PHP має свої недоліки, вона залишається однією з найпопулярніших мов програмування для веб-розробки завдяки своїм перевагам у сфері доступності, багатофункціональності та широкому використанню.

У виборі мови програмування PHP замість Python для розробки Telegram боту є кілька ключових причин, які відображаються у специфіці цієї мови. Мова PHP була спеціально розроблена для веб-розробки, що робить її ідеальною для створення ботів, які взаємодіють через HTTP-протоколи, легше інтегрувати веб-сервери та використовувати веб-специфічні функції. PHP-додатки легко розгортати на більшості веб-серверів, що робить запуск Telegram боту швидким процесом, що є важливим для проєктів із обмеженими ресурсами та строгими термінами. Легка робота з Webhook у PHP є великою перевагою для інтеграції платіжних систем, таких як Crypto Bot, дозволяючи ефективно налаштовувати та управляти платіжними операціями через Telegram бот. Ця мова легко інтегрується з різними базами даних та іншими веб-технологіями, що є важливим для ботів, котрі потребують взаємодії з різними веб-ресурсами та базами даних для зберігання стану, користувацьких даних тощо. Ці особливості

роблять PHP привабливим вибором для розробки Telegram боту, особливо коли потрібно швидко розробити і запустити бота [15].

2.2 Аналіз баз даних

MySQL - це одна з найпопулярніших відкритих систем управління базами даних (СУБД), яка широко використовується для зберігання та керування реляційними базами даних. Ось детальний огляд особливостей MySQL:

1. Надійність та швидкість:

MySQL відома своєю високою надійністю та швидкодією, що робить її популярним вибором для великих веб-сайтів, систем управління контентом та корпоративних додатків.

Вона оптимізована для роботи з великим обсягом даних та високими навантаженнями, що дозволяє їй ефективно обробляти тисячі запитів за короткий час.

2. Розширюваність:

MySQL підтримує різні методи розширення, включаючи горизонтальне та вертикальне масштабування, реплікацію та розподілені транзакції.

Це дозволяє розгортати бази даних MySQL на серверах з великою кількістю ядер та пам'яті для забезпечення високої продуктивності та доступності.

3. Функціональність:

MySQL має широкий набір функцій, включаючи підтримку транзакцій, індексацію даних, зберігані процедури, тригери, представлення, операції з JSON, регулярні вирази та інші.

Вона підтримує різні типи даних, такі як числа, рядки, дати, часи, зображення, географічні дані, що робить її універсальним рішенням для різних типів додатків.

4. Легкість використання:

MySQL має простий та зрозумілий SQL-синтаксис, який робить легкою роботу з базою даних для розробників усіх рівнів.

Вона підтримує велику кількість інструментів адміністрування, таких як phpMyAdmin, MySQL Workbench, що полегшує управління базою даних.

5. Відкритість та спільнота:

MySQL - це відкрита СУБД з активною спільнотою розробників, яка надає регулярні оновлення, виправлення помилок та нові функції.

Вона має широку документацію та велику кількість ресурсів для навчання та підтримки.

В цілому, MySQL є потужним, надійним та широко використовуваним рішенням для зберігання та управління реляційними базами даних у різних типах програм [16].

PostgreSQL - це потужна об'єктно-реляційна система управління базами даних (СУБД), яка відкрита, гнучка та розширювана. Ось детальний огляд особливостей PostgreSQL [17].

1. Надійність та стабільність:

PostgreSQL відома своєю високою надійністю та стабільністю. Вона має вбудовану підтримку транзакцій, журналізації та відновлення після збоїв, що робить її ідеальним рішенням для критичних застосунків.

2. Розширюваність:

PostgreSQL підтримує різні методи розширення, включаючи горизонтальне та вертикальне масштабування, реплікацію, розподілені транзакції та розподілені обробки запитів.

Це дозволяє гнучко налаштовувати та розгортати бази даних для відповіді на різні потреби та обсяги даних.

3. Функціональність:

PostgreSQL має багатий набір функцій, включаючи підтримку тригерів, збережених процедур, виразів, віконних функцій, географічних типів даних, операцій з JSON та інші.

Вона підтримує також різні типи даних, такі як рядки, числа, дати, часи, зображення, географічні дані тощо.

4. Сумісність:

PostgreSQL підтримує стандарт SQL ANSI, що робить її сумісною з багатьма іншими СУБД і дозволяє легко переносити додатки з однієї системи на іншу.

Вона також підтримує різні рівні ізоляції транзакцій, включаючи серіалізовану ізоляцію, що забезпечує високий рівень консистентності даних.

5. Розвиток та спільнота:

PostgreSQL має активну та велику спільноту розробників, яка надає регулярні оновлення, виправлення помилок та нові функції.

Вона має докладну документацію, різноманітні ресурси для навчання та підтримки, а також різноманітні розширення та допоміжні інструменти.

Узагальнюючи, PostgreSQL є потужною та надійною СУБД, яка підходить для різних типів додатків та завдань. Вона забезпечує широкий набір функцій, високу продуктивність та надійність, що робить її відмінним вибором для великих та критичних застосунків.

MongoDB - це документо-орієнтована система управління базами даних (NoSQL), яка використовує JSON-подібні документи для зберігання даних. Ось детальний огляд особливостей MongoDB:

1. Схема документів:

MongoDB використовує гнучку схему документів, що дозволяє зберігати дані у вигляді JSON-подібних об'єктів, які можуть мати різні поля та структури.

Це дозволяє легко змінювати структуру даних та додавати нові поля без необхідності зміни схеми бази даних.

2. Горизонтальне масштабування:

MongoDB підтримує горизонтальне масштабування, що дозволяє розгортати бази даних на кількох серверах та автоматично розподіляти навантаження між ними.

Це дозволяє підтримувати великі обсяги даних та високу доступність в розподілених середовищах.

3. Швидкодія та продуктивність:

MongoDB відома своєю високою швидкодією та продуктивністю завдяки використанню індексації даних, кешування та іншим оптимізаціям.

Вона також підтримує запити до бази даних у форматі мови запитів MongoDB (MongoDB Query Language), яка дозволяє швидко та ефективно отримувати дані.

4. Гнучкість та розширюваність:

MongoDB має широкий набір функцій, таких як підтримка транзакцій, текстового пошуку, геопросторових операцій, операцій з масивами та інші.

Вона також підтримує різні формати даних, такі як документи, масиви, геодані та інші, що дозволяє зберігати різнорідні дані.

5. Спільнота та інструменти:

MongoDB має велику та активну спільноту розробників, яка надає регулярні оновлення, виправлення помилок та нові функції.

Вона має також широкий вибір інструментів для адміністрування, моніторингу та розробки, таких як MongoDB Compass, MongoDB Atlas, MongoDB Shell та інші.

Узагальнюючи, MongoDB є потужною та гнучкою системою управління базами даних, яка підходить для різних типів додатків та завдань. Вона надає широкий набір функцій, високу швидкодію та надійність, що робить її відмінним вибором для розробки сучасних додатків.

Redis - це швидка та потужна система управління даними типу ключ-значення, яка часто використовується для кешування, сесій та повідомлень у веб-додатках. Ось детальний огляд особливостей Redis:

1. Простота та ефективність:

Redis має простий та легкий у використанні інтерфейс команд, який дозволяє швидко зберігати та отримувати дані у форматі ключ-значення.

Вона працює у пам'яті, що дозволяє отримувати швидкий доступ до даних та виконувати операції з ними з великою швидкодією.

2. Типи даних:

Redis підтримує різні типи даних, включаючи рядки, хеші, списки, множини та сортовані множини.

Це дозволяє зберігати різнорідні дані та використовувати їх для різних цілей, від кешування до обробки потоків даних.

3. Висока доступність:

Redis підтримує реплікацію та шардування, що дозволяє розгортати кластери Redis на кількох серверах та забезпечувати високу доступність та надійність.

Вона також має вбудовану підтримку механізму моніторингу та автоматичного відновлення, що дозволяє швидко виявляти та виправляти збої.

4. Підтримка транзакцій:

Redis підтримує транзакції, що дозволяє виконувати групу команд атомарно та забезпечувати консистентність даних.

Вона також підтримує різні операції над множинами даних, такі як об'єднання, перетин, різниця тощо.

5. Гнучкість та розширюваність:

Redis має широкий вибір розширень та модулів, які дозволяють розширювати його функціональність та використовувати його для різних цілей.

Вона також має велику спільноту розробників, яка надає різноманітні інструменти та бібліотеки для роботи з Redis.

Узагальнюючи, Redis є потужним та ефективним інструментом для кешування, сесій та повідомлень у веб-додатках. Вона надає швидку доступність до даних, гнучкість та високу доступність, що робить її відмінним вибором для великих та складних проектів [18].

Одним із найважливіших інструментів для управління базами даних MySQL є PHPMyAdmin, веб-інтерфейс, що дозволяє адмініструвати MySQL бази даних з веб-браузера. PHPMyAdmin надає широкий спектр функцій для роботи з базами даних, включаючи створення, модифікацію, видалення баз даних і таблиць, управління полями, виконання SQL запитів, а також управління користувачами та правами доступу. Ці можливості роблять PHPMyAdmin вибором багатьох розробників для швидкого розгортання та легкого тестування проектів, оскільки він спрощує процес управління базами даних, особливо в середовищах, де потрібна швидка ітерація і часте оновлення даних.

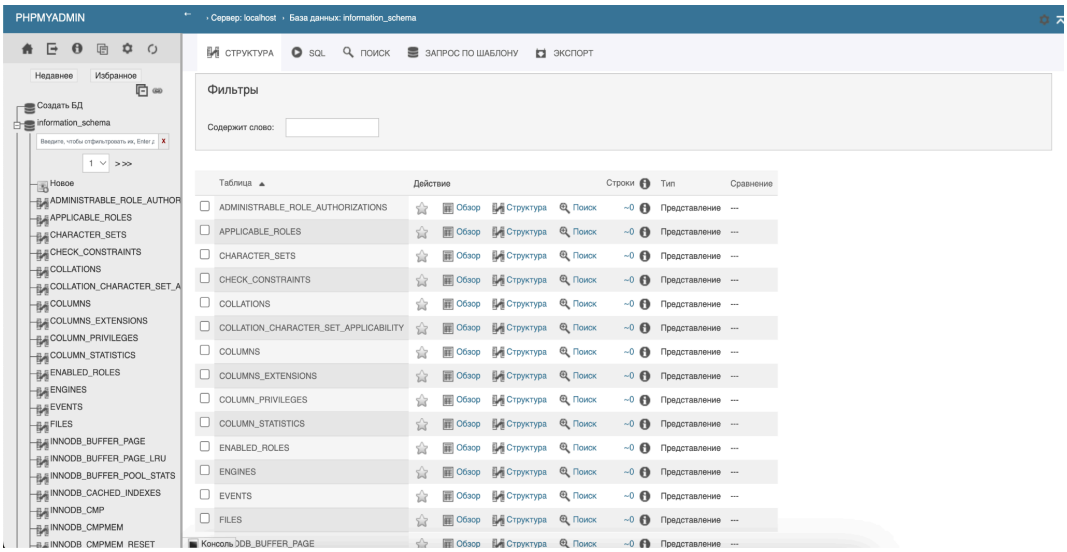

Рисунок 2.1 – Інтерфейс PHPMyAdmin

Вибір PHPMyAdmin як інструменту для управління базами даних в проекті розробки Telegram-бота на PHP було зумовлено його зручністю,

інтуїтивно зрозумілим інтерфейсом та потужним функціоналом. Цей веб-інструмент дозволяє розробникам ефективно керувати структурою бази даних, проводити необхідні зміни та адаптації, а також виконувати складні запити без потреби заходити в консоль. Така гнучкість є критично важливою у процесі швидкої розробки і налагодження ботів, що мають взаємодіяти з користувачами в реальному часі. Також, використання PHPMyAdmin допомагає уникнути помилок, що можуть виникнути при ручній обробці даних, і забезпечує більш стабільне та надійне середовище для роботи додатків, які залежать від баз даних.

Загалом, PHPMyAdmin є незамінним інструментом для розробників, які використовують MySQL у своїх проектах, надаючи їм можливість легко керувати базами даних через графічний інтерфейс, що сприяє підвищенню продуктивності та ефективності розробки.

Однією з ключових переваг PHPMyAdmin є його здатність спрощувати процеси імпорту та експорту даних. Завдяки графічному інтерфейсу користувача, PHPMyAdmin дозволяє розробникам легко завантажувати та вивантажувати дані з баз даних у різних форматах, таких як SQL, CSV, XML та інші. Цей процес є інтуїтивно зрозумілим: користувачам просто потрібно вибрати базу даних або таблицю, натиснути на кнопку "Експорт" чи "Імпорт" і вказати необхідний формат файлу та інші параметри, які можуть включати кодування, типи даних та обмеження на розмір файлу.

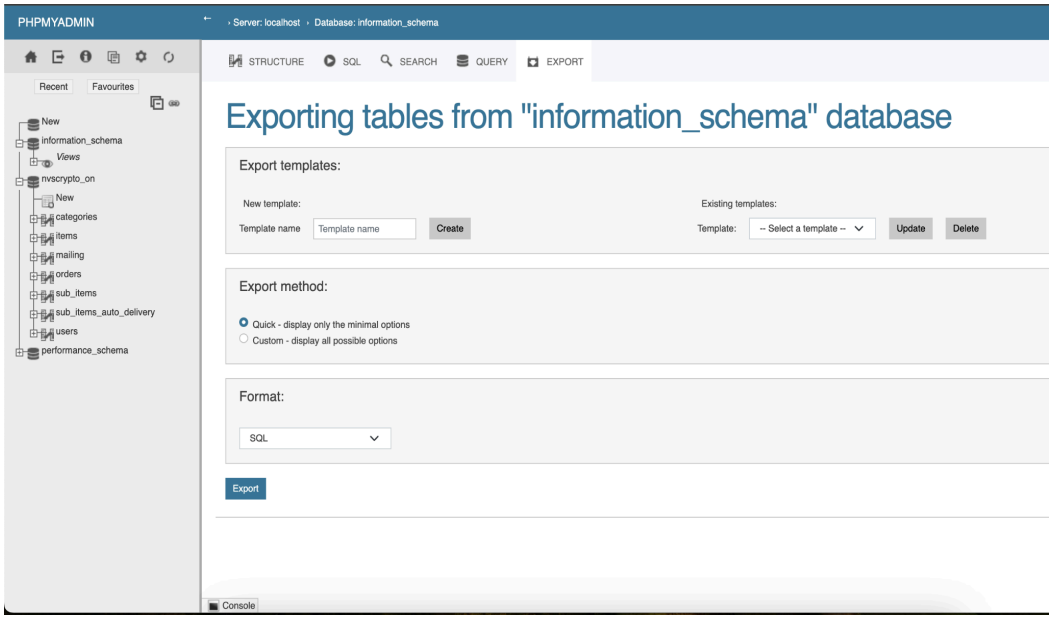

Рисунок 2.2 – Функція експорту в PHPMyAdmin

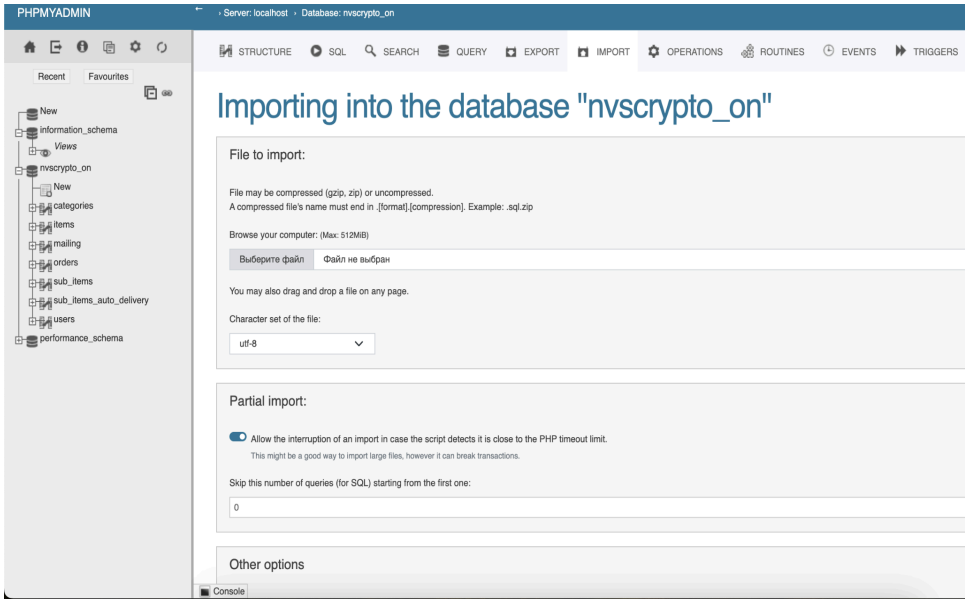

Рисунок 2.3 - Функція імпорту в PHPMyAdmin

Ця функціональність робить PHPMyAdmin незамінним інструментом для тих, хто часто працює з переносом даних між різними системами або потребує резервного копіювання та відновлення баз даних. Операції імпорту та експорту

можуть виконуватися без прямого втручання розробника в структуру бази даних, що знижує ризик помилок і забезпечує високу точність обробки даних.

PHPMyAdmin надає розширені можливості для роботи з таблицями баз даних, зокрема для сортування даних за різними критеріями. Ця функціональність є особливо корисною, коли користувачам потрібно аналізувати або візуалізувати великі обсяги інформації за певним порядком, забезпечуючи швидкий доступ до необхідних даних.

У PHPMyAdmin сортування таблиць може бути здійснене за будь-яким стовпчиком, який містить дані, такі як дата, кількість або номер. Користувачі можуть легко вибрати потрібний стовпчик та вказати напрямок сортування (за зростанням або за спаданням). Процес сортування відбувається в реальному часі, і результати можуть бути відразу візуалізовані.

Така можливість сортування особливо корисна, коли потрібно швидко організувати дані для подальшого аналізу або звітності. Наприклад, сортування по даті може допомогти відслідковувати зміни або тенденції протягом часу, сортування по кількості може відобразити обсяги продажів або запасів, а сортування по номеру може бути використане для легкого доступу до конкретних записів або транзакцій.

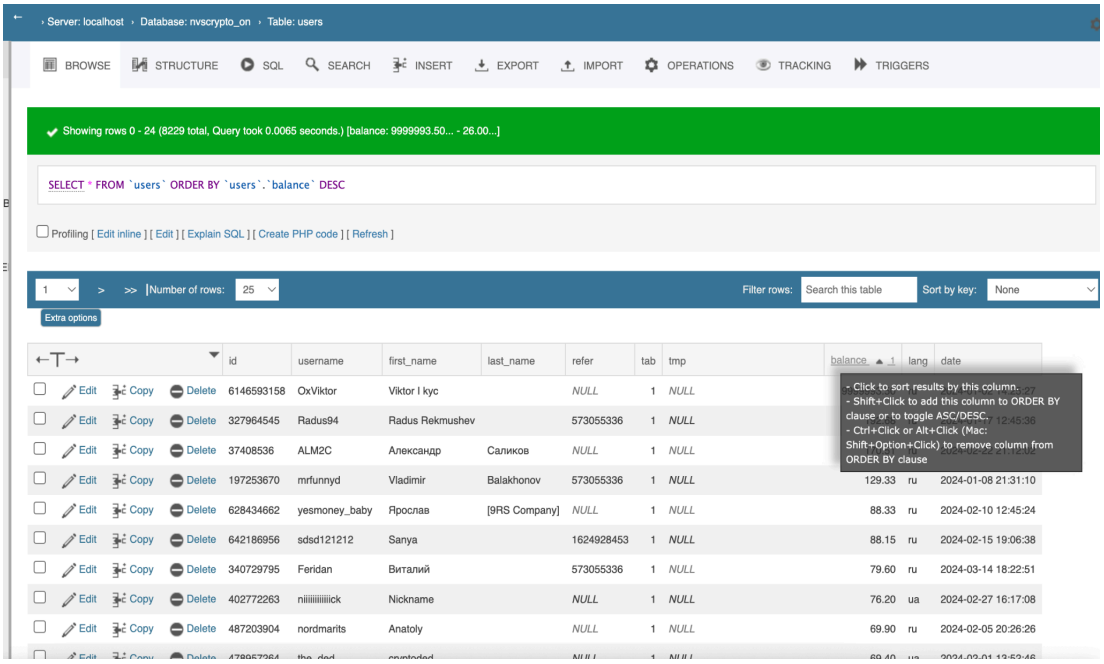

Рисунок 2.4 – Функція сортування таблиць в PHPMyAdmin, на прикладі сортування за балансом користувача

PHPMyAdmin підтримує широкий спектр кодувань, що дозволяє користувачам управляти базами даних у різноманітних мовах і форматах. Однією з важливих особливостей PHPMyAdmin є його підтримка сучасних кодувань, таких як UTF-8, яке є стандартом для кодування символів у веб-застосунках. UTF-8 дозволяє користувачам вводити та відображати символи з будь-якої мови, включаючи ті, що містять спеціальні символи та емоджі [19].

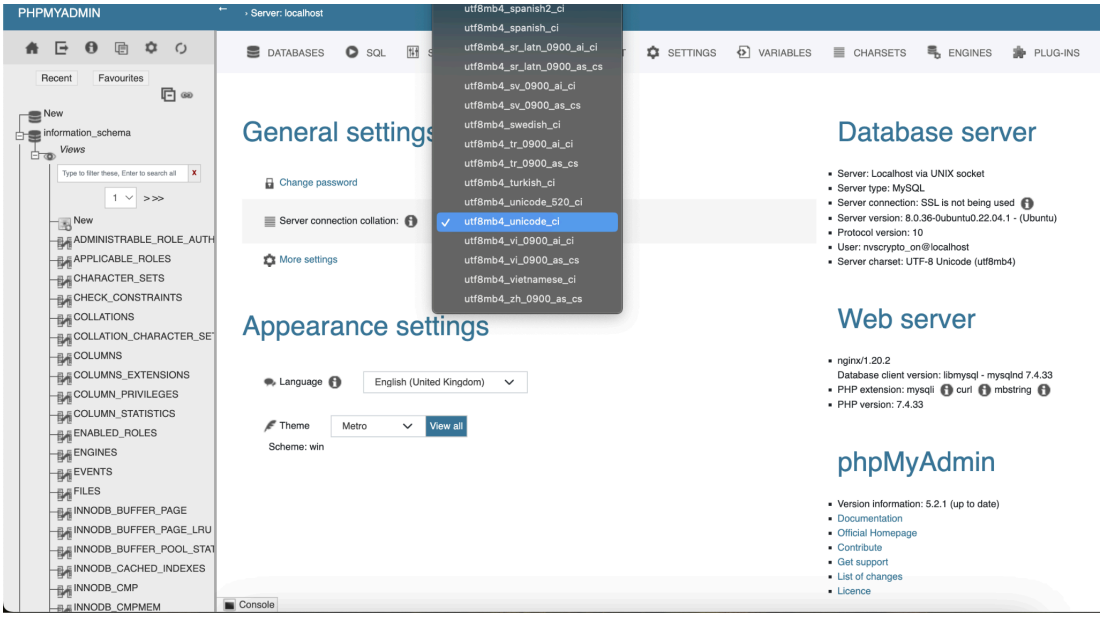

Рисунок 2.5 – Зміна кодування в PHPMyAdmin

2.3 Crypto Bot для автоматизації криптоплатежів

Crypto Bot — це автоматизований інструмент, який дозволяє користувачам виконувати різноманітні операції з криптовалютами, такі як торгівля, обмін та управління портфелем. Використання таких ботів сприяє ефективному аналізу ринку, автоматизації торгових стратегій та забезпеченню своєчасного реагування на ринкові зміни без постійного втручання користувача. Боти можуть бути програмовані для автоматичного виконання торгових операцій на основі визначених алгоритмів аналізу та прогнозування цін, що дозволяє користувачам оптимізувати свої інвестиції та збільшувати прибуток.

Crypto Pay є частиною функціоналу Crypto Bot, що забезпечує прийом платежів у криптовалютах. Ця система значно спрощує процес інтеграції криптоплатежів для бізнесів, дозволяючи їм легко приймати оплату за товари та послуги в криптовалютах. Інтеграція Crypto Pay відкриває нові можливості для міжнародної торгівлі, оскільки криптовалюта не залежить від національних валют і дозволяє здійснювати швидкі та безпечні міжнародні транзакції. Така

функціональність зокрема корисна для онлайн-магазинів, геймінгових платформ та інших цифрових сервісів, які прагнуть розширити свою клієнтську базу за рахунок прийому універсальних криптовалютних платежів.

Технічно, для інтеграції Crypto Pay необхідно отримати доступ до API, налаштувати вебхуки для обробки транзакцій та належно настроїти систему безпеки для захисту фінансових даних користувачів. Забезпечення високого рівня безпеки є критично важливим, адже криптовалютні транзакції є незворотними, і будь-які помилки або зловмисні дії можуть призвести до втрати коштів. SSL/TLS шифрування для захисту даних, регулярне оновлення програмного забезпечення та використання надійних методів аутентифікації і авторизації є стандартними практиками для захисту системи [20].

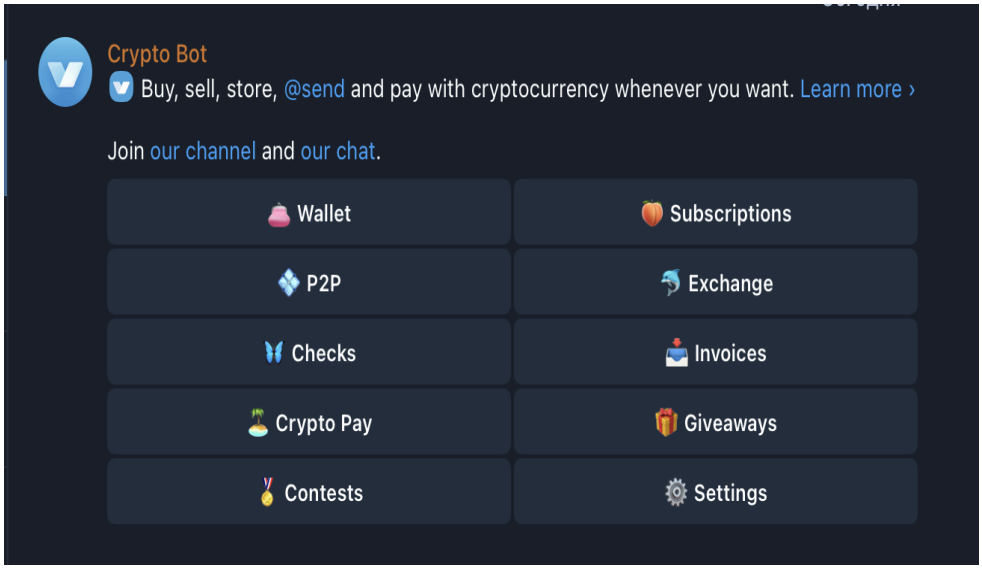

Рисунок 2.6 – Інтерфейс Crypto Bot

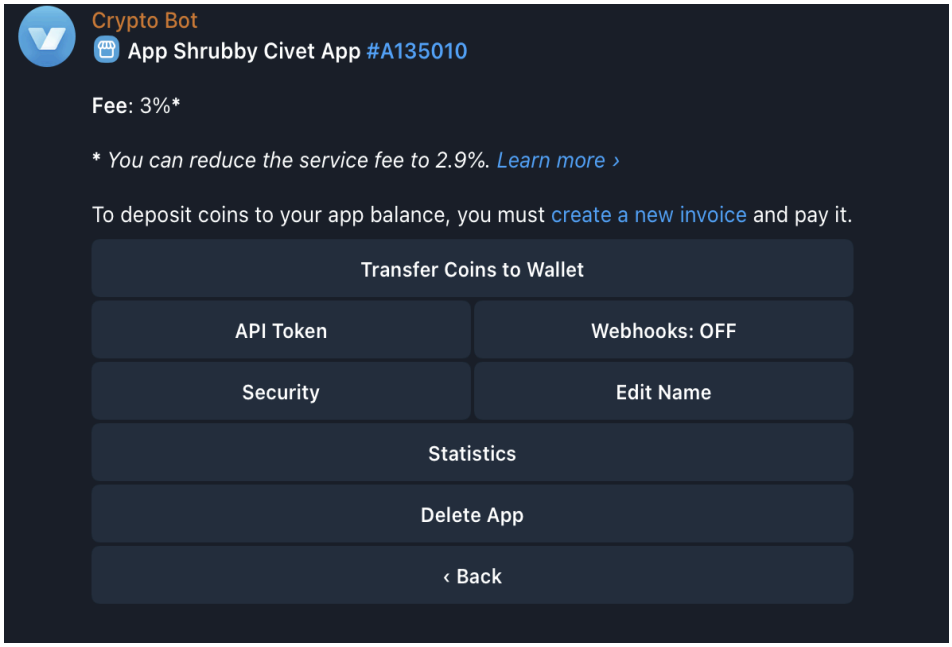

Рисунок 2.7 – Інтерфейс Crypto Pay

Таким чином, для розробки програмного рішення було обрано мову програмування PHP, БД MySQL з графічним інтерфейсом PHPMyAdmin та для обробки криптоплатежів Crypto Bot.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО РІШЕННЯ

3.1 Створення API токену для взаємодії з ботом та Telegram

Для початку роботи необхідно встановити Telegram та отримати в ньому обліковий запис.

Можна завантажити програму під необхідну операційну систему з офіційного сайту https://telegram.org/.

Необхідно зайти в пошук та набрати "BotFather". Знайдеться кілька схожих, але нам потрібен офіційний із синьою галочкою (https://t.me/BotFather).

Вибираємо правильного "BotFather" і натискаємо у чаті, що відкрилося, "start".

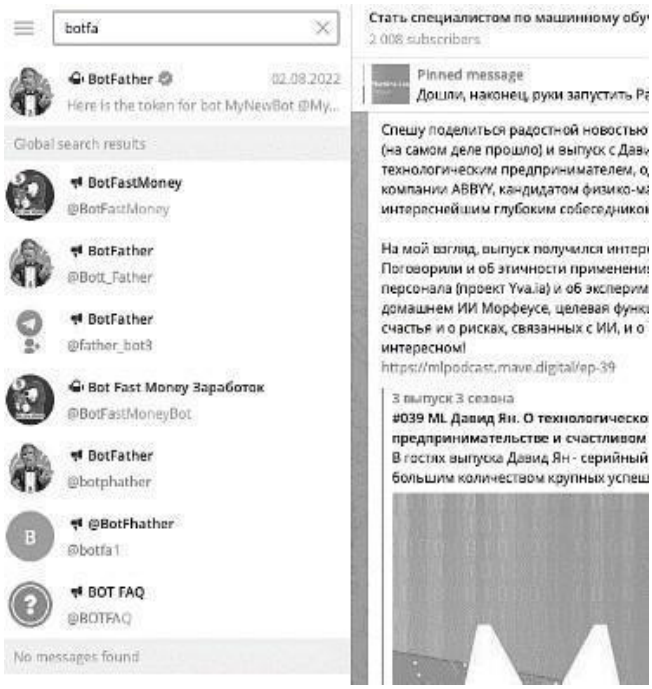

Рисунок 3.1 – BotFather у Telegram

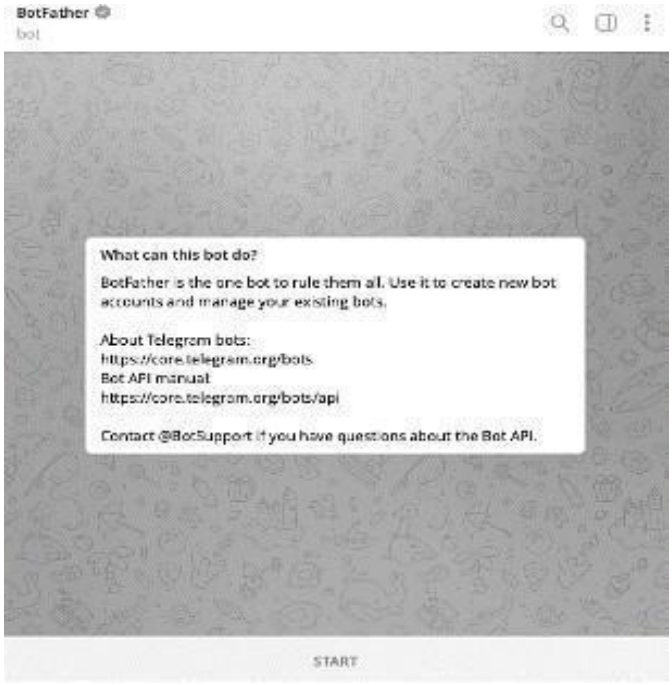

Рисунок 3.2 – Чат із BotFather телеграм-ботів

З'явиться повідомлення зі списком команд, які можна надіслати батькові ботів. Є багато різних команд. Зараз нас цікавить команда "/newbot". Або клацаємо прямо по ній, чи-бо ще в лівому нижньому кутку можна натиснути кнопку "Menu" і там теж відкриється можливість вибрати потрібну команду.

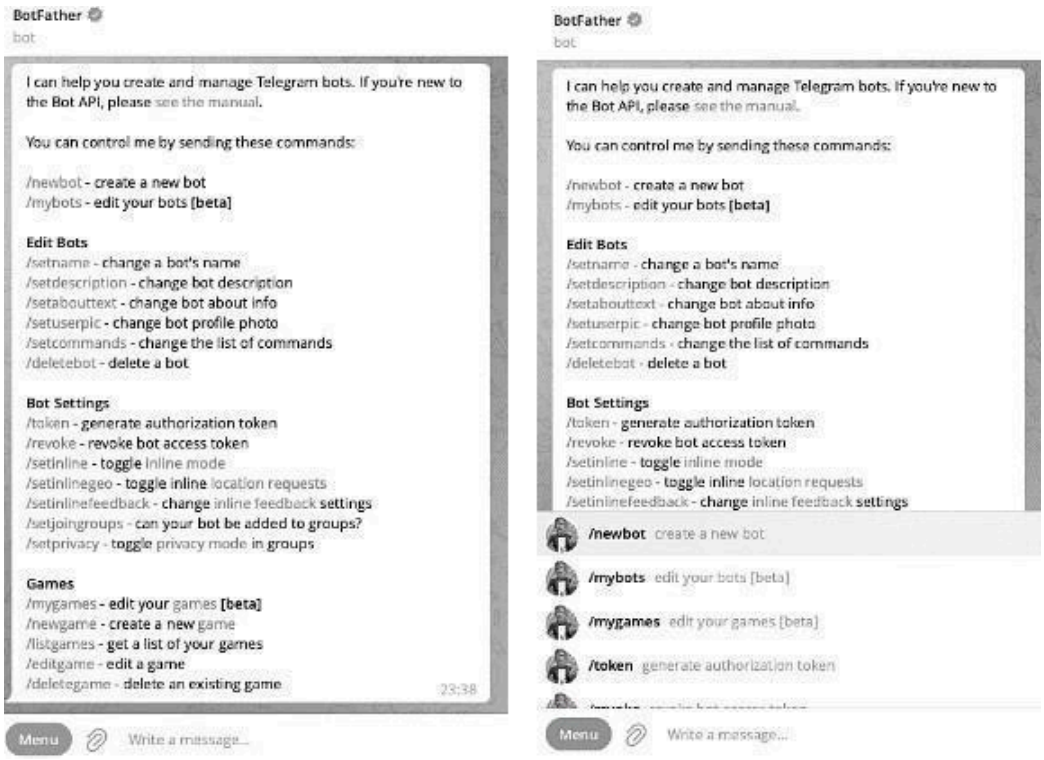

Рисунок 3.3 – Команди для налаштування BotFather

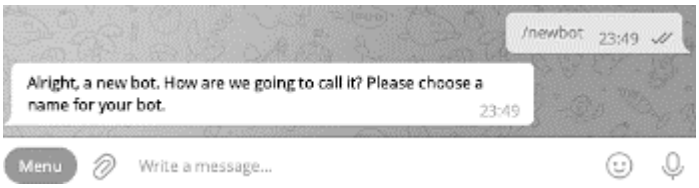

Рисунок 3.4 – Створення імені нашого боту

BotFather запропонує вибрати ім'я нашого нового роботу.

Далі потрібно задати username нашому боту - унікальне ім'я, яке не можна буде змінювати, і яке обов'язково має закінчуватися на "bot", причому регістр букв не має значення.

Пишемо будь-яке ім'я. Його, надалі, можна буде змінити за бажання. довжиною від 5 до 32 символи. Можна використовувати латинські літери, цифри та підкреслення, проте з цифри або підкреслення не можна починати username бота.

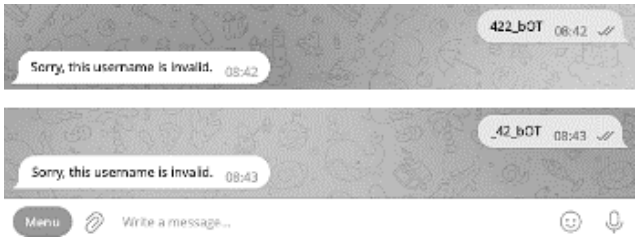

Рисунок 3.5 – Помилка у створенні імені боту

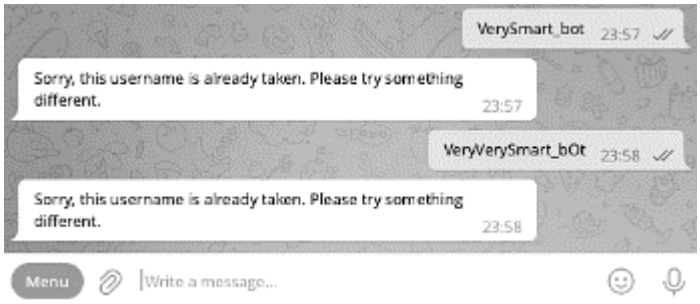

Рисунок 3.6 – Зайнятість імені

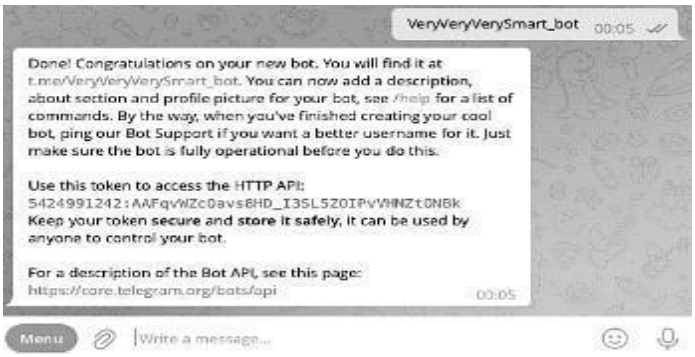

Рисунок 3.7 – Успішне створення бота та отримання токена

Таким чином, username має бути довжиною від 5 до 32 символів, серед яких можуть бути літери, цифри та підкреслення, але починатися повинен обов'язково з літери, а закінчуватися обов'язково словом "bot", причому регістр значення не має.

Через те, що юзернейм має бути унікальним, підібрати його може бути не дуже просто - багато з них вже зайняті.

При виборі не зайнятого username для бота вас привітає BotFather.

Для подальшої роботи нам знадобиться токен. Він має бути недоступний широкому колу користувачів, щоб уникнути зловживань. За потреби токен боту можна змінити. Також через @BotFather.

В рамках одного телеграм акаунту допускається керувати максимум 20 ботами.

Щоб на будь-якому етапі взаємодії з @BotFather отримати від нього список доступних команд, необхідно надіслати в чат з ним команди /start або /help. Після всіх процедур із створенням самого бота у просторі телеграм нам потрібно вибрати IDE, віртуальне оточення, мову програмування та метод створення самого бота.

Рисунок 3.8 – Токен телеграм-бот

3.2 Створення Crypto Pay застосунку та отримання API токену

Спочатку потрібно відкрити пошук у Telegram і ввести "CryptoBot". З'явиться декілька подібних результатів, проте важливо обрати офіційного CryptoBot, який позначено синьою галочкою (https://t.me/CryptoBot). Після цього перейдіть у чат з правильним "CryptoBot" і натисніть "start".

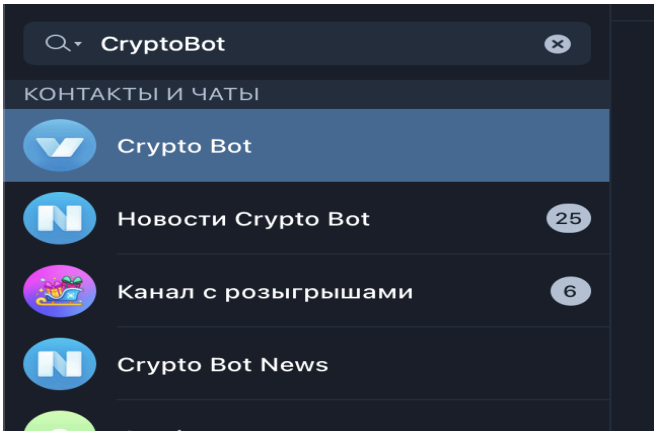

Рисунок 3.9 – Пошук CryptoBot у Telegram

В головному меню Crypto Bot переходимо в пункт Crypto Pay та створюємо застосунок.

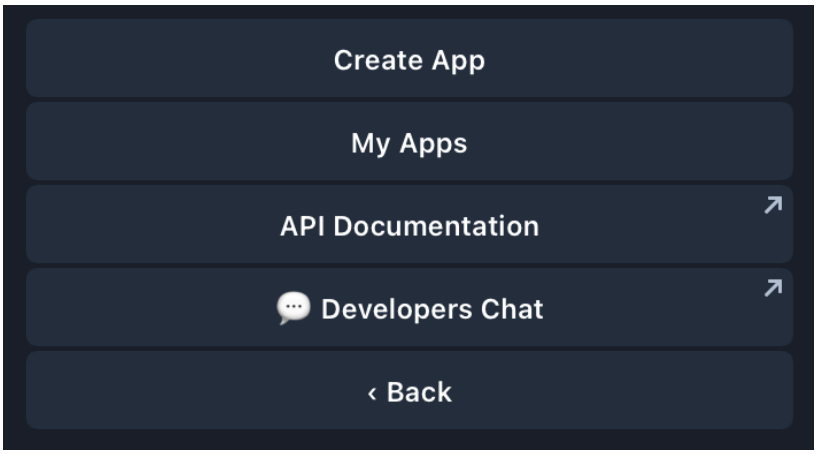

Рисунок 3.10 – Створення застосунку Crypto Pay

Тепер необхідно взяти токен застосунка, щоб інтегрувати його в бота. Для цього потрібно обрати наш застосунок та перейти в меню “API TOKEN”.

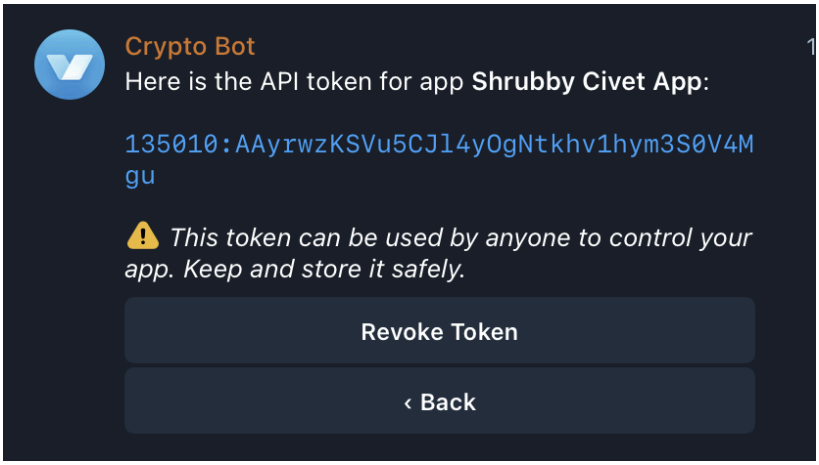

Рисунок 3.11 – Меню з API токеном Crypto Pay

Після отримання ключа API, наступним кроком є налаштування Webhook, який дозволяє отримувати автоматичні повідомлення від Crypto Bot про події, що відбуваються, такі як отримання нових платежів чи зміни стану платежів. Для цього потрібно вказати URL вашого сервера в налаштуваннях.

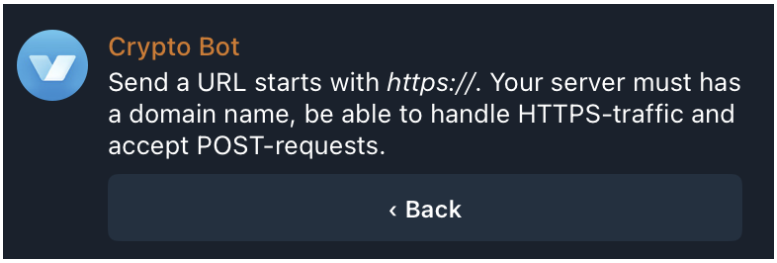

Рисунок 3.12 – Меню встановелння Webhook

3.3 Розробка бази даних та імпорт в PHPMyAdmin

Розробка бази даних та її імпорт у PHPMyAdmin відіграють вирішальну роль у підготовці функціональної системи для веб-додатку. Структура бази даних створюється з метою адекватного відображення потреб додатку. Створення таблиць таких як `users`, `items`, `sub_items`, `sub_items_auto_delivery`, `orders`, `mailing`, та `categories` дозволяє

систематично організувати і зберігати різноманітні дані, пов’язані з користувачами, товарними позиціями, транзакціями та замовленнями.

Конструкція кожної таблиці здійснюється за допомогою SQL команд, виконаних у середовищі розробки. Таблиця `users` містить поля для ідентифікації користувача, його імені, мови інтерфейсу та інших особистих даних. Таблиця `items` описує товари з їхніми категоріями та цінами, тоді як `sub_items` слугує для деталізації товарних позицій. Забезпечення інтегритету та взаємозв’язків між таблицями здійснюється через обмеження зовнішніх ключів, що гарантують логічні зв'язки між записами у різних таблицях.

CREATE TABLE - основна команда для створення таблиць у базі даних. Кожне визначення таблиці включає назви колонок, типи даних та інші параметри, такі як NOT NULL або DEFAULT. Це забезпечує структуру для зберігання даних згідно з вимогами системи. INSERT INTO - використовується для додавання нових записів до таблиць. Ця команда важлива для ініціалізації бази даних з початковими даними або для додавання нових даних у процесі використання системи. ALTER TABLE - застосовується для модифікації структури існуючих таблиць. За допомогою цієї команди можна додавати нові колонки, змінювати типи даних колонок, встановлювати первинні ключі (PRIMARY KEY), створювати індекси (ADD KEY) та визначати обмеження зовнішніх ключів (FOREIGN KEY). PRIMARY KEY - використовується для встановлення однієї або декількох колонок таблиці як унікального ідентифікатора для кожного запису в таблиці. Це забезпечує унікальність даних і є важливим для інтегритету бази даних. FOREIGN KEY - обмеження, яке забезпечує цілісність даних між таблицями, встановлюючи взаємозв’язки між колонками в різних таблицях. Це гарантує, що відносини між даними у таблицях є логічно консистентными. COMMIT - застосовується в кінці серії транзакцій для їх збереження в базі даних. Це означає, що всі модифікації, зроблені під час

транзакції, будуть постійно застосовані, забезпечуючи стійкість та відновлюваність даних.

Для створення таблиці `users` використовується наступна команда SQL:

```
sql
CREATE TABLE `users` (
  `id` varchar(512) NOT NULL,
  `username` varchar(512) DEFAULT NULL,
  `first_name` varchar(512) DEFAULT NULL,
  `last_name` varchar(512) DEFAULT NULL,
  `refer` varchar(512) DEFAULT NULL,
  `tab` int NOT NULL DEFAULT '1',
  `tmp` json DEFAULT NULL,
  `balance` decimal(65,2) NOT NULL DEFAULT '0.00',
   `lang` enum('ru','ua','en') CHARACTER SET utf8mb3 COLLATE
utf8mb3_general_ci DEFAULT NULL,
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.1 – Створення таблиці users

Ця команда створює таблицю з різними типами даних, включаючи `VARCHAR` для текстових рядків, `INT` для цілих чисел, `DECIMAL` для чисел з плаваючою комою, `JSON` для зберігання структурованих даних, та `ENUM` для обмеження значень до попередньо визначеного списку.

Таблиця `categories` використовується для категоризації товарів і може виглядати так:

```
sql
CREATE TABLE `categories` (
  `id` int NOT NULL,
     `name`  varchar(512)  CHARACTER  SET  utf8mb4  COLLATE
utf8mb4_unicode_ci NOT NULL,
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.2 – Створення таблиці categories

Ця таблиця містить три колонки: `id` як унікальний ідентифікатор для кожної категорії, `name` для назви категорії, яка підтримує Unicode для міжнародної підтримки через використання utf8mb4, і `date` для запису часу створення категорії, який автоматично встановлюється на поточний час і дату при створенні запису.

Після створення таблиці `categories`, треба додати дані:

```
sql
INSERT INTO `categories` (`id`, `name`, `date`) VALUES
(9, '👛Retrodrops👛', '2024-04-08 12:26:20'),
(10, '🧩Ready Retro Accs🧩', '2024-04-08 12:45:20');
```

Лістинг 3.3 – Додавання даних в таблицю categories

Ці записи забезпечують початкове наповнення таблиці з певними категоріями.

Також важливо встановити первинний ключ для таблиці, щоб забезпечити унікальність кожного запису:

```
sql
ALTER TABLE `categories`
  ADD PRIMARY KEY (`id`);
```

Лістинг 3.4 – Встановлення первинного ключа для таблиці categories

Це гарантує, що кожна категорія може бути однозначно ідентифікована через її `id`.

Таблиця `items` створюється за допомогою:

```
sql
CREATE TABLE `items` (
  `id` int NOT NULL,
  `category_id` int NOT NULL,
```

```
          `name`  varchar(512)  CHARACTER  SET  utf8mb4  COLLATE
utf8mb4_unicode_ci NOT NULL,
            `photo`  text  CHARACTER  SET  utf8mb3  COLLATE
utf8mb3_general_ci,
       `sub` int NOT NULL DEFAULT '1',
       `price` decimal(65,2) DEFAULT NULL,
       `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
     ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.5 – Створення таблиці items

Ця таблиця містить зв'язок з таблицею `categories` через поле `category_id`, що використовується для створення відносин між таблицями.

`sub_items` таблиця має наступну структуру:

```
     sql
     CREATE TABLE `sub_items` (
       `id` int NOT NULL,
       `item_id` int NOT NULL,
          `name`  varchar(512)  CHARACTER  SET  utf8mb4  COLLATE
utf8mb4_unicode_ci NOT NULL,
       `price` decimal(65,2) NOT NULL DEFAULT '0.00',
       `photo` text,
       `desc` text,
       `min` int NOT NULL DEFAULT '1',
       `delivery` enum('1','2','3') NOT NULL DEFAULT '1',
       `delivery_text` text,
       `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
     ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.6 – Створення таблиці sub_items

Вона також включає зовнішні ключі, що посилаються на `items`, та додаткові атрибути, які детально описують кожен підтовар.

Для `sub_items_auto_delivery`:

```
     sql
```

```
CREATE TABLE `sub_items_auto_delivery` (
  `id` int NOT NULL,
  `sub_item_id` int NOT NULL,
  `text` text,
  `status` int NOT NULL DEFAULT '0',
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.7 – Створення таблиці sub_items_auto_delivery

Ця таблиця використовується для автоматизації доставки підтоварів.

Таблиця `orders` використовується для зберігання інформації про замовлення:

```
sql
CREATE TABLE `orders` (
  `id` int NOT NULL,
  `bill`

 varchar(512) NOT NULL,
  `sum` decimal(65,6) NOT NULL,
  `payment` varchar(512) NOT NULL,
  `user_id` varchar(512) NOT NULL,
  `item_id` int DEFAULT NULL,
  `sub_item_id` int NOT NULL,
  `nums` int NOT NULL DEFAULT '1',
  `status` int NOT NULL DEFAULT '0',
  `msg_id` varchar(512) DEFAULT NULL,
    `send_date_item` text CHARACTER SET utf8mb3 COLLATE
utf8mb3_general_ci,
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.8– Створення таблиці orders

Ця таблиця також включає складніші типи даних і відносини.

Таблиця `mailing` для розсилок виглядає так:

```
sql
CREATE TABLE `mailing` (
  `id` int NOT NULL,
  `for` enum('all') NOT NULL DEFAULT 'all',
  `text` text,
  `status` int NOT NULL DEFAULT '0',
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

Лістинг 3.9 – Створення таблиці mailing

Це забезпечує зберігання інформації для розсилок користувачам.

Кожна з цих таблиць має індекси та обмеження, встановлені через `ALTER TABLE`, щоб забезпечити цілісність даних і оптимізувати швидкість запитів, включаючи первинні ключі (`ADD PRIMARY KEY`) та зовнішні ключі (`ADD CONSTRAINT`). Завершується весь процес командою `COMMIT;`, яка забезпечує збереження всіх змін, внесених у базу даних.

Імпорт розробленої бази даних до PHPMyAdmin являється завершальним етапом, що дозволяє адміністраторам систем легко керувати базою даних через графічний інтерфейс. Файл SQL, який містить всі таблиці та дані, можна імпортувати через інтерфейс PHPMyAdmin, вибравши відповідну базу даних і використовуючи функцію "Імпорт". Це забезпечує можливість швидкого відновлення або перенесення всієї бази даних у нове середовище, що підвищує ефективність та спрощує управління даними.

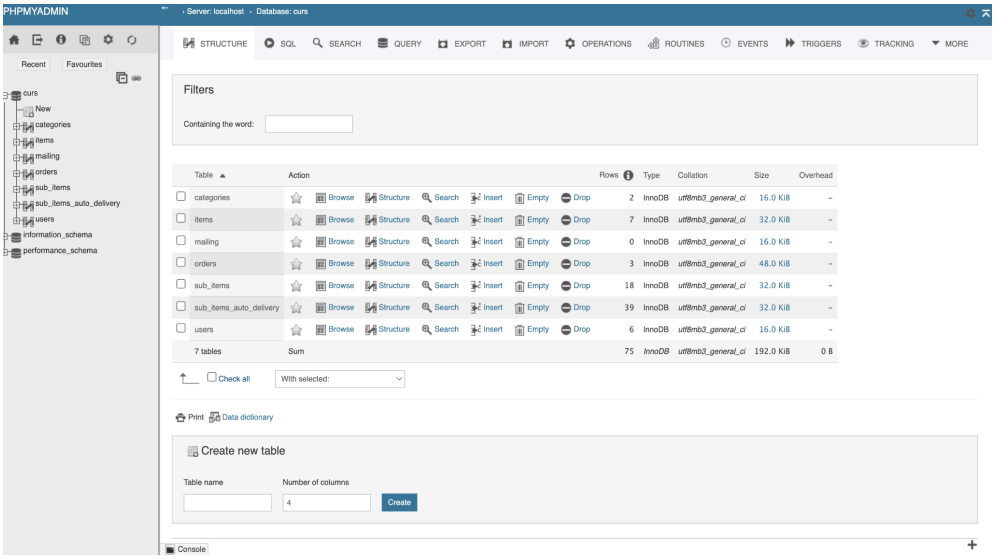

Рисунок 3.13 – Успішний імпорт БД в PHPMyAdmin

3.4 Розробка боту

При розробці програмного рішення для бота використано наступні файли:

1. `config.php`: Цей файл відповідає за зберігання конфігураційних налаштувань, зокрема токену доступу до API Telegram. Він дозволяє забезпечити зв'язок між сервером бота та платформою Telegram, необхідний для обміну повідомленнями та іншою взаємодією з користувачами.

2. `callback.php`: Цей файл містить функції, які використовуються для обробки повідомлень та відправлення відповідей. Він включає в себе різноманітні функції, такі як функція відправлення повідомлень, обробки команд користувача та інші допоміжні функції, необхідні для роботи бота.

3. `bot.php`: Цей файл представляє собою основний скрипт бота, який обробляє вхідні повідомлення, виконує необхідні дії та відправляє відповіді користувачам. Він взаємодіє з API Telegram, отримуючи та відправляючи дані через HTTP-запити.

4. `db.php`: Цей файл відповідає за з'єднання з базою даних та виконання різних операцій, таких як вибірка, оновлення чи видалення даних. Він містить функції для взаємодії з базою даних та виконання SQL-запитів.

5. `language.php` Цей файл використовується для локалізації бота, забезпечуючи текстові ресурси на кілька мов, що дозволяє адаптувати інтерфейс до мовних уподобань користувачів.

6. `users_tab_selector.php` відіграє ключову роль у навігації між різними вкладками користувацького інтерфейсу в боті. Він забезпечує динамічну взаємодію, дозволяючи користувачам переходити між різними частинами бота.

Ці файли утворюють основну структуру програмного рішення для бота, забезпечуючи його правильну роботу та взаємодію з користувачами платформи Telegram.

У рамках розробки телеграм-боту для надання криптопослуг важливим аспектом є організація взаємодії з базою даних. Підключення до бази даних дозволяє зберігати, обробляти та отримувати інформацію, необхідну для функціонування бота. Розглянемо файл `db.php`, який забезпечує інтеграцію з базою даних.

Першим кроком у файлі `db.php` є імпортування зовнішнього файлу налаштувань `config.php`, де зберігаються глобальні константи та змінні конфігурації. Цей рядок коду використовує конструкцію `include`, що дозволяє включити та виконати вказаний файл. Використання `__DIR__` забезпечує правильне знаходження шляху до файлу незалежно від поточної директорії виконання скрипта.

Далі, ініціалізується об'єкт `mysqli` для створення підключення до бази даних. Конструктор `mysqli` приймає параметри, необхідні для з'єднання:

```
php
$db = new mysqli(db_host, db_username, db_password, db_dbname);
```

Лістинг 3.10 – Підключення до бази даних

Змінні `db_host`, `db_username`, `db_password`, та `db_dbname` імпортуються з файлу `config.php` і представляють собою хост бази даних, ім'я користувача, пароль і назву бази даних відповідно. Це дозволяє гнучко керувати параметрами з'єднання через один конфігураційний файл.

Після створення з'єднання, проводиться налаштування параметрів з'єднання з базою даних для правильного відображення символів:

```
php
$db->query("SET NAMES 'utf8mb4'");
```

Лістинг 3.11– Встановлення кодування

Ця команда встановлює кодування символів `utf8mb4`, яке підтримує Unicode та забезпечує коректну роботу з емоджі та іншими символами, що важливо для мультикультурної підтримки.

На заключному етапі, скрипт встановлює часовий пояс і параметри обробки помилок:

```
php
date_default_timezone_set('Europe/Kyiv);
ini_set("memory_limit", "256M");
```

Лістинг 3.12– Встановлення часового поясу

Функція `date_default_timezone_set` задає часовий пояс, що важливо для коректної роботи з часовими мітками даних. `ini_set` змінює ліміт пам'яті для скрипта, забезпечуючи достатньо ресурсів для обробки запитів.

Ця розробка підключення до бази даних є критично важливою для забезпечення надійності, ефективності та масштабування.

Файл `bot.php` відіграє ключову роль у функціонуванні телеграм-бота, оскільки він відповідає за основну логіку обробки запитів від користувачів. Цей файл виконує інтеграцію з іншими компонентами системи, такими як база даних і модуль локалізації, та забезпечує обробку повідомлень, що надходять від користувачів.

На початку файлу здійснюється підключення до файлів, що містять необхідні налаштування та функції для роботи з базою даних та мовні налаштування:

```
php
include(__DIR__ . "/db.php");
include(__DIR__ . "/language.php");
```

Лістинг 3.13 – Підключення файлів до коду

Ці рядки коду включають зовнішні файли, що дозволяють використовувати вже визначені змінні та функції для роботи з базою даних і локалізації. Використання `__DIR__` гарантує, що шлях до файлів буде визначено коректно, незалежно від місця запуску скрипта.

Основна частина скрипта починається з обробки вхідних даних, що отримуються з запитів від Telegram:

```
php
$update = json_decode(file_get_contents('php://input'),
TRUE);
```

Лістинг 3.14 – Перетворення JSON-відповіді

Цей рядок перетворює JSON-відповідь, що надходить від сервера Telegram, в асоціативний масив PHP. Це дозволяє легко доступити до даних про повідомлення, які включають текст, фотографії, документи тощо.

Файл містить логіку для ідентифікації типу вхідного повідомлення – чи це текст, фотографія чи документ:

```
php
$is_photo = isset($update['message']['photo']);
$is_message = isset($update["message"]["text"]);
$is_document = isset($update['message']['document']);
```

Лістинг 3.15 – Перетворення JSON-відповіді

На основі отриманих даних, скрипт визначає тип повідомлення і ініціює відповідні обробники. Наприклад, якщо повідомлення містить текст, воно може бути оброблено за допомогою функцій, що аналізують і реагують на команди користувача.

Цей файл є фундаментальним для інтерактивності бота, оскільки він обробляє всі основні запити користувачів і реалізує логіку взаємодії. Використання цього файлу дозволяє забезпечити гнучке управління поведінкою бота залежно від потреб користувача.

`config.php` є центральним файлом налаштувань для телеграм-бота, що забезпечує криптопослуги. Цей файл містить визначення констант і змінних, які використовуються в інших частинах програми для доступу до ресурсів та конфігурації параметрів. Він включає налаштування для бази даних, API ключів, параметрів платежів та інших елементів, необхідних для роботи бота.

На початку файла встановлюються параметри для з'єднання з базою даних:

```
php
const db_host = "";
const db_username = "";
const db_password = "";
const db_dbname = "";
```

Лістинг 3.14 – Вказання даних для підключення до бази даних

Ці константи використовуються для встановлення з'єднання з базою даних через mysqli в інших скриптах. Значення хоста, імені користувача, паролю, та назви бази даних мають бути визначені для забезпечення здійснення з'єднань.

Далі, у файлі визначаються токени та ідентифікатори для інтеграції з API Telegram та іншими платежними системами:

```
php
const __TG_TOKEN = "";
const NOTIFY_CHAT_ID = "";
const CRYPTOBOT__TOKEN = "";
```

Лістинг 3.14 – Вказання даних для роботи з API

Ці константи несуть в собі значення для аутентифікації та взаємодії з сервісами, забезпечуючи можливість надсилання повідомлень та виконання фінансових операцій.

Окрема увага приділяється конфігурації платіжних методів та підтримуваних валют:

```
php
const CRYPTOBOT_PAYMENTS_METHODS = ["USDT", "TRX", "USDC"];
const PAYMENT_METHODS = ["CryptoBot"];
```

Лістинг 3.16– Підключення видів і типів платіжних методів

Ці параметри визначають доступні методи та валюти для проведення транзакцій, що забезпечує гнучкість в опціях платежів для користувачів бота.

Файл також включає визначення для локалізації, визначаючи підтримувані мови та відповідні текстові мітки:

```
php
const LANG_LIST_code = ['ua', 'ru', 'en'];
const LANG_LIST_txt = ['Українська', 'Русский', 'English'];
```

Лістинг 3.17 – Мови інтерфейсу

Ці змінні дозволяють програмі адаптуватися до мовних налаштувань користувачів, забезпечуючи більш доступний та зрозумілий інтерфейс.

Файл `language.php` використовується для локалізації бота, забезпечуючи текстові ресурси на кілька мов, що дозволяє адаптувати інтерфейс до мовних вподобань користувачів. Цей файл містить масиви текстів, які використовуються у різних частинах програми для забезпечення зручності і зрозумілості інтерфейсу.

На початку файла створюється масив `$_LANGUAGE`, який включає варіанти перекладів для кожної підтримуваної мови. Ось приклад з українськими перекладами:

```
php
$_LANGUAGE = [
    'ua' => [
          "START_TEXT_MESSAGE" => "Привіт, це бот retrodrops,
раді вітати! Оберіть послугу, яка вас цікавить",
            'START_BTNS_MESSAGE' => [
                "🔍 Каталог",
                "📱 Відгуки",
                "🏠 Профіль",
                "👩‍💻 Допомога",
                "📊 Партнерка"
            ],
```

```
                "BUY_ITEM_ERROR_NUMS" => "Неправильна кількість,
спробуйте ще раз 🔄",
                "BTN_RETURN" => "«Назад",
                "BTN_RETURN_MENU" => "«В меню",

        ],
```

Лістинг 3.18 – Локалізація

Цей розділ файла включає рядки для української мови, з перекладами текстів, які відображатимуться користувачам. Такий підхід забезпечує легку адаптацію текстових елементів інтерфейсу під різні мови, що покращує взаємодію з користувачами з різних культурних та лінгвістичних середовищ.

У файлах такого типу важливо підтримувати консистенцію ключів у масиві для кожної мови, щоб запобігти помилкам у відображенні текстів. Кожен ключ в масиві представляє собою окремий текстовий елемент інтерфейсу, який може бути легко викликаний у різних частинах програми.

`users_tab_selector.php` відіграє ключову роль у навігації між різними вкладками користувацького інтерфейсу в боті. Він забезпечує динамічну взаємодію, дозволяючи користувачам переходити між різними частинами бота, такими як покупка товарів, перегляд інформації або налаштування особистого профілю. Цей файл використовує умовні оператори для визначення поточної вкладки, яку обрав користувач, і відповідно змінює поведінку бота.

У коді файлу використовуються різні функції для обробки даних та взаємодії з користувачем. Ось приклад коду з поясненнями ключових функцій:

```
php
if ($tmp_type == 'buy_sub_item' && $current_tab == 2) {
    $sub_item = get_value_by_id($tmp->i, 'sub_items');
    $item = get_value_by_id($sub_item->item_id, 'items');
    if (!$item || !$sub_item) return;
```

```
        $nums = (int)$message;

                        if ($nums < 0) return
msg(bold($lang['BUY_ITEM_ERROR_NUMS']));

          if ($sub_item->min > $nums) return msg(bold("⚠️ " .
arr_replace($lang['PAGE_BUY_SUB_ITEM_MSG'][2], ["v" =>
$sub_item->min]) . " ⚠️"));

        if (!$item || !$sub_item) return;
        $txt = gen_text([
            bold($item->name . " " . $sub_item->name),
            "",
                    bold($lang['PAGE_BUY_SUB_ITEM_MSG_TAB'][0] .
code($nums) + " шт. "),
                    bold($lang['PAGE_BUY_SUB_ITEM_MSG_TAB'][1] +
code(round($sub_item->price * $nums, 2)) + "$"),
            "",
            bold($lang['PAGE_BUY_SUB_ITEM_MSG_TAB'][2])
        ]);
        $kb = [];
        foreach (PAYMENT_METHODS as $key => $payment) {
              $kb[] = ["text" => $payment, "callback_data" =>
json_encode(["t" => "pre_s_ip_c_buy", "i" => $key])];
        }
```

Лістинг 3.19 – Навігація боту

- get_value_by_id(): Забирає значення з бази даних за певним ID. У цьому контексті використовується для отримання деталей про товар та підтовар.

- msg(): Функція для відправлення повідомлень користувачу. Використовується для повідомлення про помилки або інші важливі повідомлення.

- bold(): Функція для форматування тексту у жирний шрифт, забезпечує кращу візуалізацію важливої інформації.

- code(): Функція для вставки тексту у форматі коду, часто використовується для підкреслення чисел або спеціальних значень у повідомленнях.

- gen_text(): Генерує відформатований текст на основі наданих аргументів, забезпечуючи структуроване і зрозуміле відображення інформації.

Файл callback.php обробляє зворотні виклики (callback queries), які є важливою частиною взаємодії у Telegram ботах, дозволяючи реагувати на дії користувача, такі як натискання кнопок. Зворотні виклики використовуються для забезпечення динамічної інтерактивності без необхідності відправляти нове повідомлення.

```
<?php
include(__DIR__ . "/db.php");
include(__DIR__ . "/language.php");
```

Лістинг 3.20 – Імпорт даних з файлів

Цей код включає два файли за допомогою функції include(). Давайте розглянемо його детальніше:

include(__DIR__ . "/db.php");: Цей рядок включає файл з назвою "db.php", що знаходиться в тій же директорії, що і файл, в якому викликається цей код. Функція __DIR__ повертає шлях до поточної директорії, а include() включає вміст файлу в цю точку виконання скрипту.

include(__DIR__ . "/language.php");: Аналогічно першому рядку, цей рядок включає файл з назвою "language.php", що також знаходиться в поточній директорії.

Такі файли, як "db.php" і "language.php", містять код, який організовує роботу з базою даних або мовними ресурсами, відповідно. Включення цих

файлів у звичайний PHP-скрипт дозволяє використовувати їх функціонал у поточному контексті.

```
$update = json_decode(file_get_contents('php://input'), TRUE);

bot_init();

$callback = $update['callback_query'];
$is_photo = isset($update['message']['photo']);
$is_message = isset($update["message"]["text"]);
$is_callback = isset($update['callback_query']);
$is_document = isset($update['message']['document']);
$tmp = (object)[];
$current_tab = 1;
$tmp_type = "";
```

Лістинг 3.21 – Обробка даних

$update = json_decode(file_get_contents('php://input'), TRUE);: Цей рядок отримує дані, які надходять від бота або іншого джерела через php://input, як JSON-рядок. Функція file_get_contents() отримує вміст вхідного потоку, а json_decode() розбирає JSON-рядок у масив PHP.

bot_init();: Цей рядок викликає функцію bot_init(), яка, очевидно, ініціалізує бота чи виконує якісь підготовчі дії для роботи з ботом.

$callback = $update['callback_query'];: В цьому рядку витягується дані зі змінної $update із ключем 'callback_query' та присвоюється змінній $callback.

$is_photo = isset($update['message']['photo']);: Цей рядок перевіряє, чи містить дані, отримані у змінну $update, ключ 'message', який, в свою чергу, містить ключ 'photo'.

$is_message = isset($update["message"]["text"]);: Тут перевіряється наявність текстового повідомлення у змінній $update.

$is_callback = isset($update['callback_query']);: Перевіряється наявність callback-запиту у змінній $update.

$is_document = isset($update['message']['document']);: Цей рядок перевіряє наявність документа у змінній $update.

$tmp = (object)[];: Створюється порожній об'єкт $tmp.

$current_tab = 1;: Змінній $current_tab присвоюється значення 1.

$tmp_type = "";: Змінній $tmp_type присвоюється порожній рядок.

Цей код в основному виконує обробку даних, які приходять від бота чи іншого джерела, і встановлює різні прапорці для подальшої роботи з ними.

```
if ($is_photo || $is_message || $is_document) :
    $lower_message = mb_strtolower($update["message"]["text"]);
    $message = $update["message"]["text"];
    $type = $update["message"]['chat']['type'];
    $chat_arr = $update['message']['chat'];
    $chat_group_id = $chat_arr["id"];
    $msg_id = $update["message"]["message_id"];
    $chat_id = $update["message"]["chat"]["id"];
    $user_id = $update["message"]["from"]["id"];
    $username = $update["message"]['from']['username'];
    $first_name = $update["message"]['from']['first_name'];
    $last_name = $update["message"]['from']['last_name'];
    $clean_msg = str_replace(" ", "", $message);
  $user = get_user_by_id($user_id, true);
    $is_Admin = in_array($user_id, ADMINS_LIST);
    $lang = init_lang_file($user->lang);
    $refer = str_replace("/start ", "", $message);
    if ($type != "private") {
        if ($message == '/id' && $is_Admin) msg("Chat id: " .
$chat_group_id);
        if (find_str($message, "/send") && $is_Admin) {
            $reply_data = $update['message']['reply_to_message'];
            $reply_data_msg_id = $reply_data['message_id'];
            $reply_data_msg = $reply_data['text'];
            $reply_data_entities = $reply_data['entities'];
            $order = get_order_by_msg_id($reply_data_msg_id);
            if (!$order) return;
            $delivery_text = str_replace("/send ", "", $message);
            $txt =
format_telegram_entities_in_text($reply_data_msg . "\n\n✅ Товар
отриманий", $reply_data_entities);
            $kb = [];
            edite_inline_keyboard(NOTIFY_CHAT_ID,
$reply_data_msg_id);
            msg_to_chat(bold("📩 Ваш товар:\n") . $delivery_text,
$order->user_id);
        }
        return;
    }
```

Лістинг 3.22 – Обробка повідомлень

Ця частина коду виконується, якщо будь-яка змінна $is_photo, $is_message або $is_document має значення true. Давайте розглянемо кожний рядок цієї умови:

$lower_message = mb_strtolower($update["message"]["text"]);: Змінна $lower_message отримує текст повідомлення змінної $update, перетворений до нижнього регістру за допомогою функції mb_strtolower(). Це може бути корисно для подальшого порівняння тексту.

$message = $update["message"]["text"];: Змінна $message отримує текст повідомлення без будь-яких змін.

$type = $update["message"]['chat']['type'];: Змінна $type отримує тип чату (наприклад, "private" для приватного чату або "group" для групового).

$chat_arr = $update['message']['chat'];: Змінна $chat_arr отримує дані про чат змінної $update.

$chat_group_id = $chat_arr["id"];: Змінна $chat_group_id отримує ідентифікатор групи чату.

$msg_id = $update["message"]["message_id"];: Змінна $msg_id отримує ідентифікатор повідомлення.

$chat_id = $update["message"]["chat"]["id"];: Змінна $chat_id отримує ідентифікатор чату.

$user_id = $update["message"]["from"]["id"];: Змінна $user_id отримує ідентифікатор користувача, який надіслав повідомлення.

$username = $update["message"]['from']['username'];: Змінна $username отримує ім'я користувача (якщо воно встановлене).

$first_name = $update["message"]['from']['first_name'];: Змінна $first_name отримує ім'я користувача.

$last_name = $update["message"]['from']['last_name'];: Змінна $last_name отримує прізвище користувача (якщо воно встановлене).

$clean_msg = str_replace(" ", "", $message);: Змінна $clean_msg отримує текст повідомлення з видаленими пробілами за допомогою функції str_replace(). Це може бути корисно для обробки тексту, наприклад, для аналізу команд.

Отже, ці рядки отримують і обробляють дані про повідомлення, які надходять від користувача.

Далі, код містить обробку повідомлень у випадку, якщо тип чату не є "private" (тобто це груповий або канал), інші дії виконуються, якщо користувач є адміністратором чату. Давайте розглянемо кожен рядок:

$user = get_user_by_id($user_id, true);: Функція get_user_by_id() отримує дані користувача за його ідентифікатором.

$is_Admin = in_array($user_id, ADMINS_LIST);: Змінна $is_Admin встановлюється в true, якщо ідентифікатор користувача є в списку адміністраторів.

$lang = init_lang_file($user->lang);: Ініціалізується мовний файл для користувача на основі його мовних установок.

$refer = str_replace("/start ", "", $message);: Видаляється "/start" з повідомлення, якщо воно містить цю команду.

Блок умови перевіряє, чи не є тип чату "private" (тобто чи це не приватний чат). Якщо це так, виконуються додаткові дії:

Якщо повідомлення містить команду '/id' і користувач є адміністратором ($is_Admin), то виводиться повідомлення з ідентифікатором чату.

Якщо в повідомленні знаходиться команда '/send' і користувач є адміністратором ($is_Admin), виконуються деякі додаткові дії, пов'язані з відправленням повідомлення.

Якщо виконані всі необхідні дії у груповому чаті, виконується оператор return;, що завершує виконання функції і виходить з неї.

Цей фрагмент коду обробляє дії, які відбуваються у груповому чаті, зокрема, для адміністратора чату.

```
    if (strlen($user->lang) == 0) {
        if (!$user && find_str($message, "/start", 'l') &&
$tmp_type != "write_paid_data") insert_new_user($user_id,
$username, $first_name, $last_name, $refer);
        $txt = bold("?");
        $kb = [];
        foreach (LANG_LIST_txt as $key => $lang_name) {
            $kb[] =  ["text" => $lang_name, "callback_data" =>
json_encode(["t" => "ch_lang", "i" => $key])];
        }
        arr_chunk($kb, 2);
        return inline_keyboard($chat_id);
    }
```

Лістинг 3.23 – Перевірка мови

У цьому фрагменті коду виконується додаткова логіка, яка перевіряє, чи має користувач встановлену мову. Якщо мова користувача не встановлена, виконується наступне:

if (strlen($user->lang) == 0) {: Перевіряється, чи має користувач встановлену мову. Якщо мова користувача не встановлена ($user->lang - пустий рядок), виконується наступний блок коду.

if (!$user && find_str($message, "/start", 'l') && $tmp_type != "write_paid_data") insert_new_user($user_id, $username, $first_name, $last_name, $refer);: Ця умова перевіряє, чи користувач не існує в базі даних, чи повідомлення містить команду '/start', чи $tmp_type не дорівнює "write_paid_data". Якщо ці умови виконуються, викликається функція insert_new_user(), яка додає нового користувача до бази даних.

$txt = bold("?");: Змінній $txt присвоюється текст "?" в жирному форматі. Це, ймовірно, повідомлення, яке буде відображатися користувачу у відповідь.

Для кожної доступної мови створюється кнопка з текстом мови і відповідними даними зворотного виклику (callback_data), які містять інформацію про мову та тип події (t - тип, і - індекс). Згодом ця клавіатура буде відправлена користувачеві для вибору мови.

arr_chunk($kb, 2);: Цей рядок розділяє масив клавіш на підмасиви по 2 елементи, щоб створити дворядкову клавіатуру.

return inline_keyboard($chat_id);: Викликається функція inline_keyboard(), яка відправляє клавіатуру користувачеві у відповідь у груповому чаті.

```
    if (find_str($message, "/start", 'l') && $tmp_type != 
"write_paid_data") {
        if (!$user)  insert_new_user($user_id, $username, 
$first_name, $last_name, $refer);
        c_inline_keyboard($lang['START_TEXT_MESSAGE'], 
gen_main_buttons(), $chat_id, IMG_MAIN_PHOTO);
        return;
    }

    if (!$user) return;

    if ($user && $current_tab > 1) {
        include(__DIR__ . "/users_tab_selector.php");
        return;
    }
```

Лістинг 3.24 – Обробка повідомлень

У цьому фрагменті коду виконується додаткова логіка, що стосується обробки повідомлень, команд та дій користувача. Розглянемо кожен рядок:

if (find_str($message, "/start", 'l') && $tmp_type != "write_paid_data") {: Цей блок перевіряє, чи повідомлення містить команду '/start' (з регістром l), а також чи тип тимчасової змінної $tmp_type не дорівнює "write_paid_data". Якщо обидва умови виконуються, виконується наступний блок коду.

if (!$user) insert_new_user($user_id, $username, $first_name, $last_name, $refer);: Якщо користувач не існує, викликається функція insert_new_user(), щоб додати нового користувача до бази даних.

c_inline_keyboard($lang['START_TEXT_MESSAGE'], gen_main_buttons(), $chat_id, IMG_MAIN_PHOTO);: Викликається функція c_inline_keyboard(), яка генерує клавіатуру з основними кнопками для взаємодії з ботом. Повідомлення

$lang['START_TEXT_MESSAGE'] відображається в чаті, а зображення IMG_MAIN_PHOTO може використовуватися як фон для клавіатури.

return;: Виконання коду зупиняється, і функція завершується, коли ця умова виконується.

if (!$user) return;: Якщо користувач не існує, виконання функції припиняється і вона завершується.

if ($user && $current_tab > 1) {: Цей блок перевіряє, чи користувач існує, а також чи поточна вкладка більше 1. Якщо ці умови виконуються, викликається файл users_tab_selector.php, який містить логіку вибору вкладок користувачем.

include(__DIR__ . "/users_tab_selector.php");: Викликається файл users_tab_selector.php, який містить логіку вибору вкладок користувачем.

return;: Виконання коду зупиняється, і функція завершується, коли ця умова виконується.

```
if (find_str($lower_message, "/u") && $is_Admin &&
strlen(numbers_in_text($message)) > 3) {
        $u_id = numbers_in_text($clean_msg);
        $u = get_user_by_id($u_id);
        if (!$u) return msg(bold("👥 Користувач з ID " .
code($u_id) . " не знайдений !"));
        $txt = gen_text([
            bold("👥 Користувач @{$u->username} з ID" .
code($u->id) . " (/u{$u->id})"),
            "",
            bold("Баланс: " . code($u->balance) . " $"),
            bold("Інвайтер: ") . code($u->refer),
            bold("Рефералів: " . code(get_count_rows("users",
"WHERE `refer` = '{$u->id}'"))),
            bold("Дата першого воходу в бота: ") . code($u->date)
        ]);
        $kb = [
            [
                ["text" => 'Змінити баланс', "callback_data" =>
json_encode(["ad" => "chg_blc_u", "u" => $u->id])],
            ],
            [
                ["text" => Закрити, "callback_data" =>
json_encode(["t" => "del_msg"])],
            ]
        ];
```

```
            inline_keyboard($chat_id);
        }
        if (find_str($lower_message, "/help")) {
            $txt = bold($lang['HELP_RE_PAGE_MSG']);
            $kb = [
                [
                    ["text" => $lang['HELP_RE_PAGE_BTNS'][0],
"callback_data" => json_encode(["t" => "hlp_faq"])],
                ],
                [
                    ["text" => $lang['HELP_RE_PAGE_BTNS'][1],
"callback_data" => json_encode(["t" => "hlp_manuals"])],
                ],
                [
                    ["text" => $lang['HELP_RE_PAGE_BTNS'][2],
"callback_data" => json_encode(["t" => "hlp_tp_list"])],
                ],
                [
                    ["text" => $lang['BTN_RETURN'], "callback_data" =>
json_encode(["t" => "m_menu"])],
                ],
            ];
            inline_keyboard($chat_id);
        }
```

Лістинг 3.25 – Обробка команд

У цьому фрагменті коду відбувається обробка повідомлень, які містять певні команди. Розглянемо кожен блок коду:

if (find_str($lower_message, "/u") && $is_Admin && strlen(numbers_in_text($message)) > 3) {: Ця умова перевіряє, чи повідомлення містить команду "/u", чи користувач є адміністратором ($is_Admin), а також чи довжина числа, яке міститься у повідомленні, більше 3 символів.

$u_id = numbers_in_text($clean_msg);: Витягується ідентифікатор користувача з повідомлення.

$u = get_user_by_id($u_id);: Отримується інформація про користувача за його ідентифікатором.

if (!$u) return msg(bold("🫂 Користувач з ID " . code($u_id) . " Не Знайдений !"));: Якщо користувач не знайдений, відправляється повідомлення про помилку.

Формується текст повідомлення про користувача з деталями, такими як ім'я користувача, баланс, дата першого заходу в бота тощо.

Створюється клавіатура з кнопками для зміни балансу користувача та закриття повідомлення.

inline_keyboard($chat_id);: Викликається функція для відображення клавіатури в чаті.

if (find_str($lower_message, "/help")) {: Ця умова перевіряє, чи повідомлення містить команду "/help".

Формується текст повідомлення з інформацією про допомогу користувачеві.

Створюється клавіатура з кнопками для доступу до різних розділів допомоги та кнопка для повернення до головного меню.

inline_keyboard($chat_id);: Викликається функція для відображення клавіатури в чаті.

```
elseif (find_str($lower_message, "/profile")) {

        $txt = gen_text([
            bold($lang['USER_PROFILE_PAGE'][0]),
            "",
            bold("ID: " . tg_format_link_id($user_id)),
            bold("@: @" . $username),
            "",
            bold($lang['USER_PROFILE_PAGE'][1] . $user->balance .
" $"),
        ]);
        $kb = [];
        $kb[] =  [
            ["text" => $lang['BTN_CHANGE_LANGUAGE'],
"callback_data" =>   json_encode(["t" => "u_change_language"])],
        ];
        $kb[] =  [
            ["text" => $lang['BTN_RETURN'], "callback_data" =>
json_encode(["t" => "m_menu"])],
        ];
        inline_keyboard($chat_id, false, false, true);
    } elseif (find_str($lower_message, "/catalog")) {
        $txt = bold($lang['PAGE_CATALOG_MSG']);
        $kb = [];
```

```
        foreach (qrArrayTable("categories") as $key => $category)
{
            $kb[] =  ["text" => $category->name, "callback_data"
=>   json_encode(["t" => "ct_open", "i" => $category->id])];
        }
        arr_chunk($kb, 1);
        if ($is_Admin) {
            $kb[] =  [
                ["text" => '➕ Добавить ➕', "callback_data" =>
json_encode(["ad" => "add_new_ct"])],
            ];
        }
        $kb[] =  [
            ["text" => $lang['BTN_RETURN'], "callback_data" =>
json_encode(["t" => "m_menu"])],
        ];
        inline_keyboard($chat_id);
        user_set_tmp_data($user_id, 1, 'reset');
    }
```

Лістинг 3.26 – Обробка команд

У цьому фрагменті коду виконується обробка повідомлень, які містять певні команди. Розглянемо кожен блок коду:

elseif (find_str($lower_message, "/profile")) {: Ця умова перевіряє, чи повідомлення містить команду "/profile".

Формується текст повідомлення з інформацією про профіль користувача, такою як його ID, ім'я користувача та баланс.

Створюється клавіатура з кнопкою для зміни мови та кнопкою для повернення до головного меню.

inline_keyboard($chat_id, false, false, true);: Викликається функція для відображення клавіатури в чаті.

elseif (find_str($lower_message, "/catalog")) {: Ця умова перевіряє, чи повідомлення містить команду "/catalog".

Формується текст повідомлення зі списком категорій.

Створюється клавіатура з кнопками для вибору категорії та кнопкою для повернення до головного меню.

Якщо користувач є адміністратором, додається кнопка для додавання нової категорії.

inline_keyboard($chat_id);: Викликається функція для відображення клавіатури в чаті.

user_set_tmp_data($user_id, 1, 'reset');: Встановлюються тимчасові дані для користувача з параметром "reset".

```
endif;
if ($callback) :
    include(__DIR__ . "/callback.php");
endif;
```

Лістинг 3.27– Обробка

Цей фрагмент коду закриває попередні умови та переходить до обробки колбеків, якщо вони присутні. Розглянемо його:

endif;: Цей рядок закриває попередні умови (if) і вказує на завершення блоку коду, який містить різні умови для обробки повідомлень користувача.

if ($callback) :: Ця умова перевіряє, чи присутній колбек (запит від користувача, що містить додаткову інформацію про подію, таку як натискання на кнопку в клавіатурі). Якщо такий запит присутній, виконується наступний блок коду.

include(__DIR__ . "/callback.php");: Включається файл callback.php, який, ймовірно, містить логіку обробки колбеків.

endif;: Цей рядок закриває блок коду для обробки колбеків (if ($callback) :) і вказує на завершення цього блоку коду.

Загальною метою цієї частини коду є обробка повідомлень та взаємодія з користувачами через Telegram-бота. Основні дії, які виконуються в цьому коді, включають:

Обробка вхідних повідомлень від користувачів за допомогою вбудованих функцій Telegram API.

Перевірка та обробка команд, які надходять від користувачів.

Відправлення відповідей користувачам та взаємодія з клавіатурою.

Код використовує функції для роботи з базою даних та з клавіатурою, щоб надавати різноманітні функції для користувачів бота. Додатково, він має механізм для обробки колбеків (відповідей на натискання кнопок у клавіатурі).

Загалом, цей код представляє собою робочий фрагмент Telegram-бота, який може виконувати різні функції для взаємодії з користувачами, забезпечуючи їм доступ до різних послуг або інформації через чатовий інтерфейс.

3.5 Тестування програми

Після запуску бота бачимо наступне повідомлення:

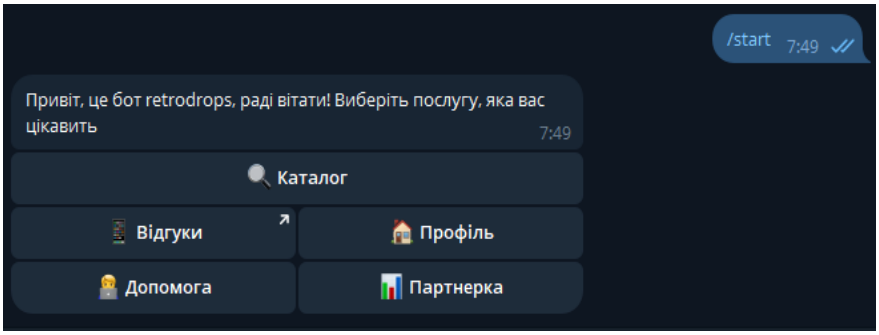

Рисунок 3.14 – Початок роботи з Ботом

Обираємо послугу «Каталог»:

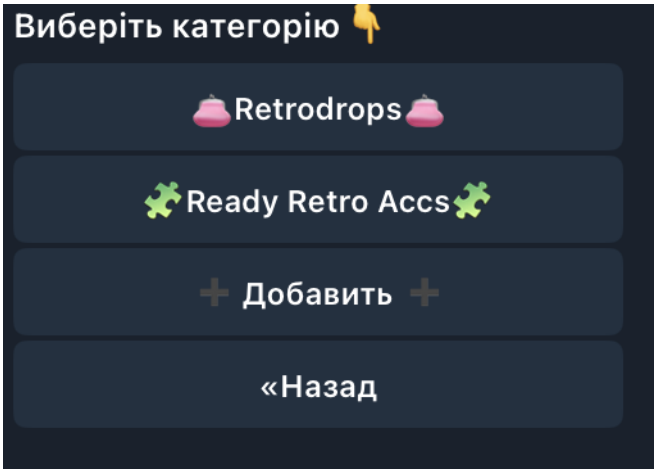

Рисунок 3.15 – Розділ «Каталог»

Обравши категорію «Retrodrops» бачимо наступне меню:

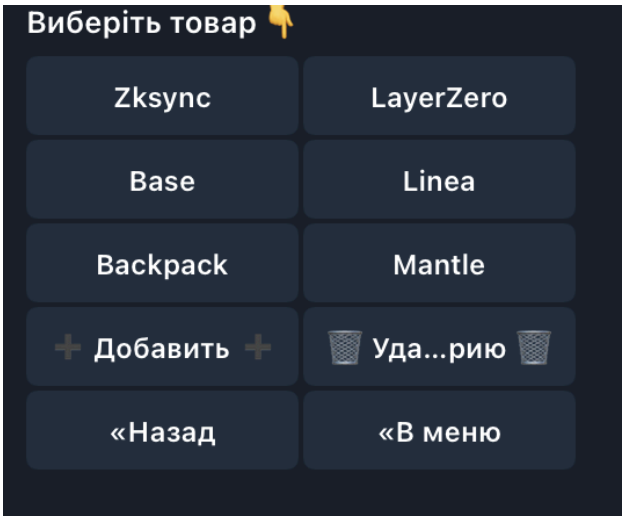

Рисунок 3.16 – Категорія «Retrodrops»

Відповідно, можна обрати послугу, оберемо один з варіантів:

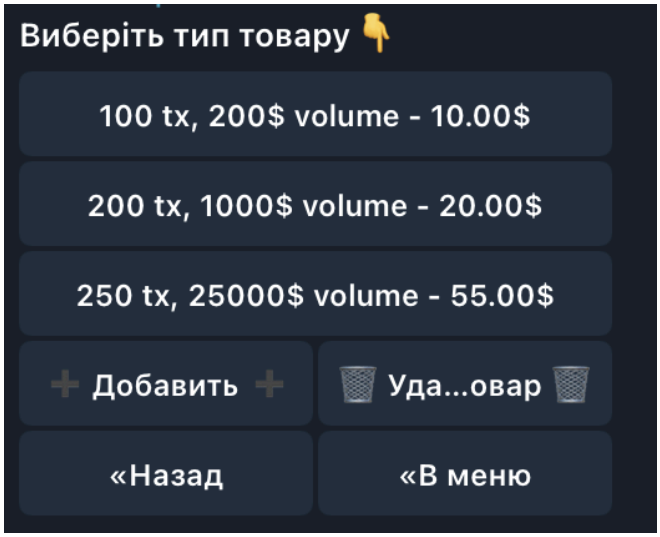

Рисунок 3.17 – Вибір типу товару

Далі, програма нам запропонує кілька варіантів на вибір.

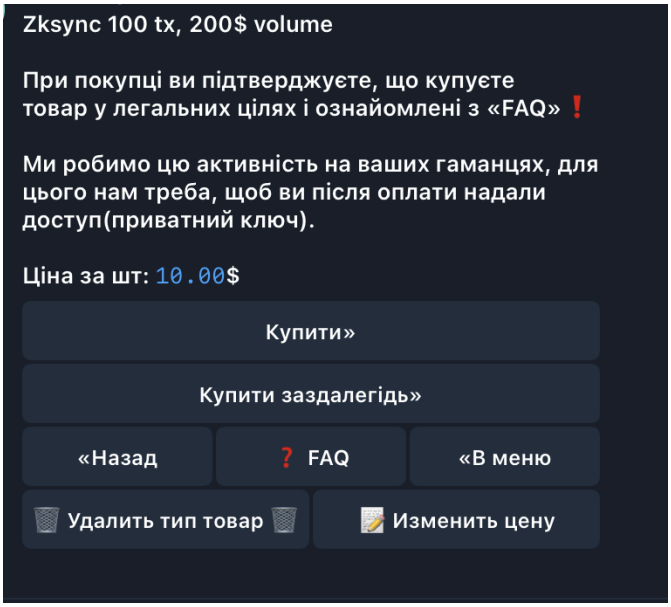

Рисунок 3.18 – Покупка «Zksync»

Ознайомившись з повідомленнями, тиснемо кнопку «Купити». Ввівши необхідну кількість, переходимо до оплати покупки:

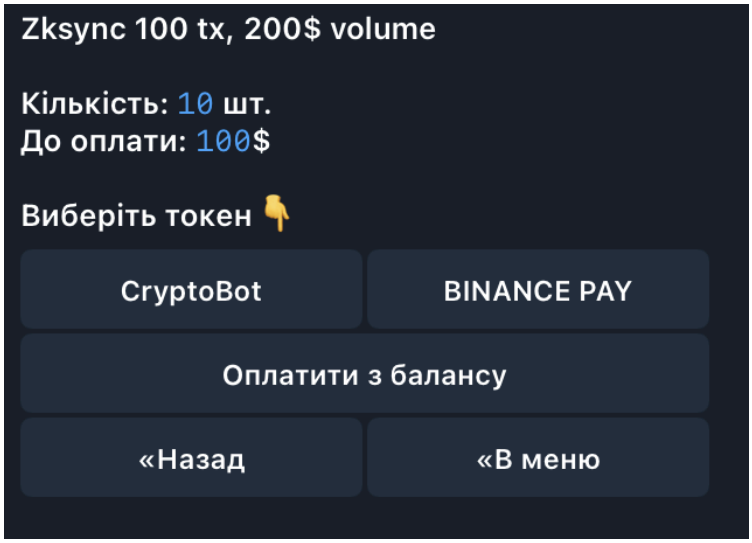

Рисунок 3.19 – Вибір методу оплати

Вибравши оплату через платіжну систему Crypto Bot, нас перенаправляє в @CryptoBot, для оплати.

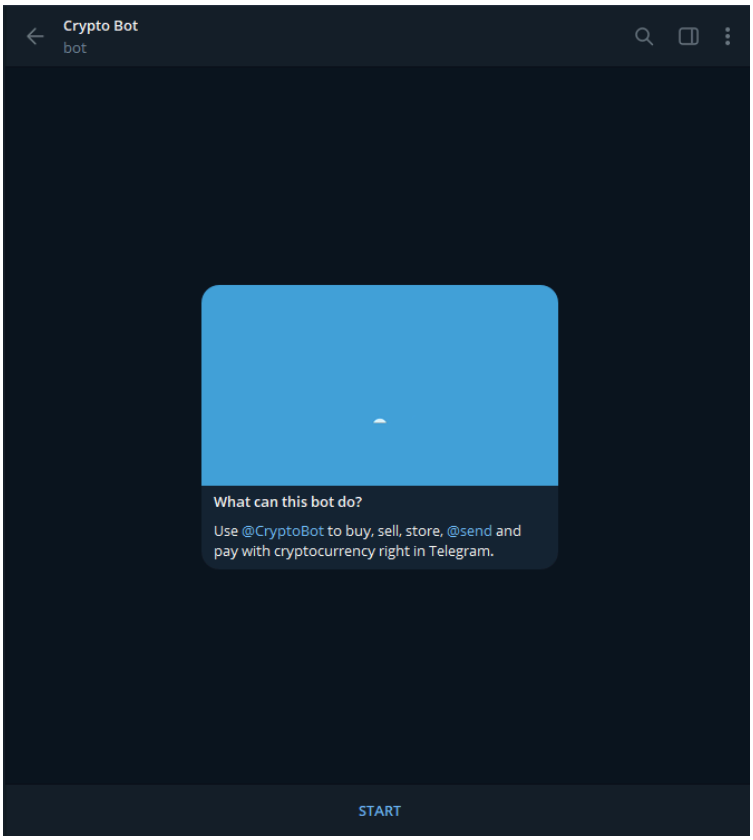

Рисунок 3.20 – Оплата через «CryptoBot»

Далі насикаємо «start» і нам даються реквізити для оплати.

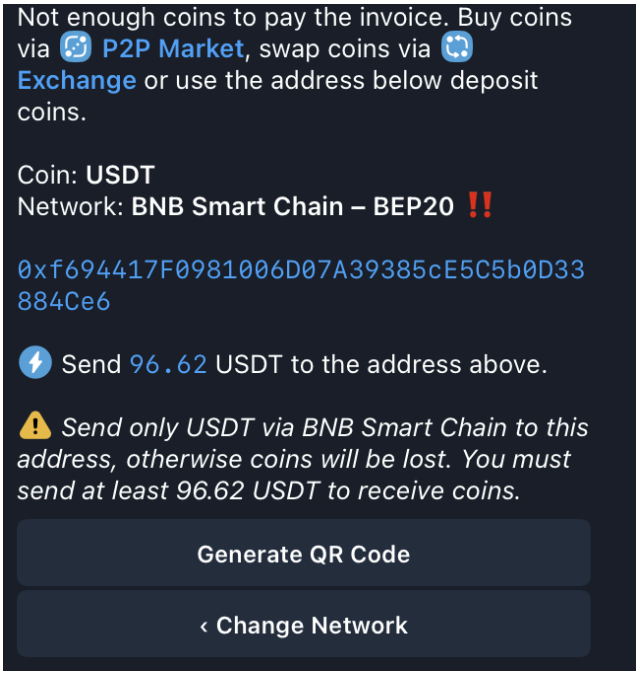

Рисунок 3.21 – Реквізити для сплати рахунку

Після сплати рахунку повертаємося в нашого бота та натискаємо кнопку «Сплатив»

Далі бот пропонує вам вказати дані, для того щоб надати послугу.

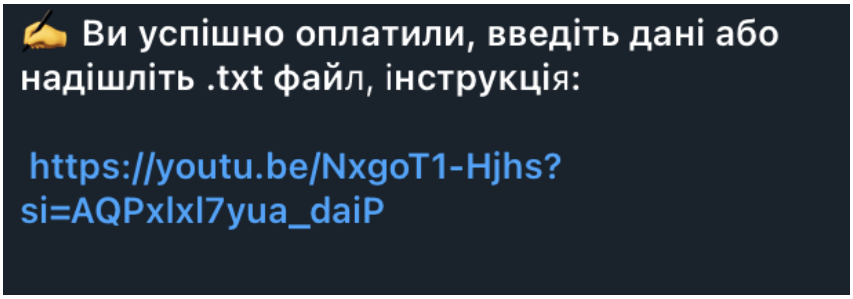

Рисунок 3.22 – Сповіщення після успішної сплати

Надавши дані, власнику магазину приходить сповіщення про те, що клієнт здійснив замовлення.

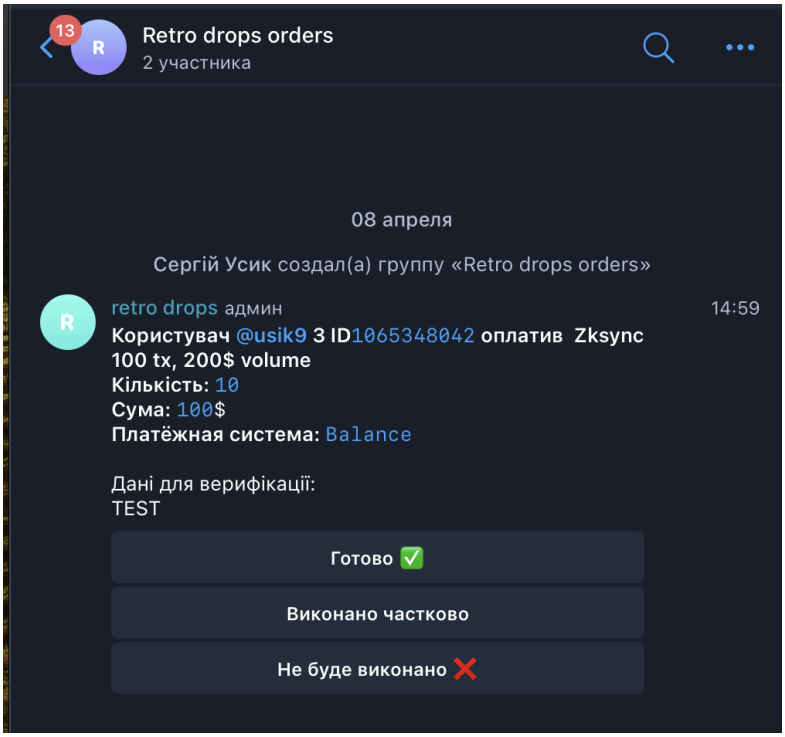

Рисунок 3.23 – Сповіщення власнику, про те що клієнт здійснив покупку

Також, можемо обрати категорію «ReadyRetroAccs»:

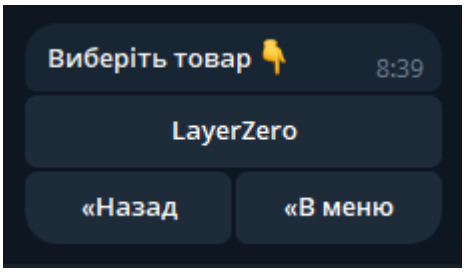

Рисунок 3.24 – Вибір товару для покупки

Обираємо запропонований варіант:

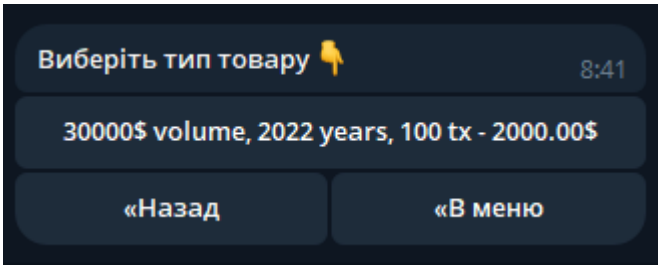

Рисунок 3.25 – Вибір типу товару

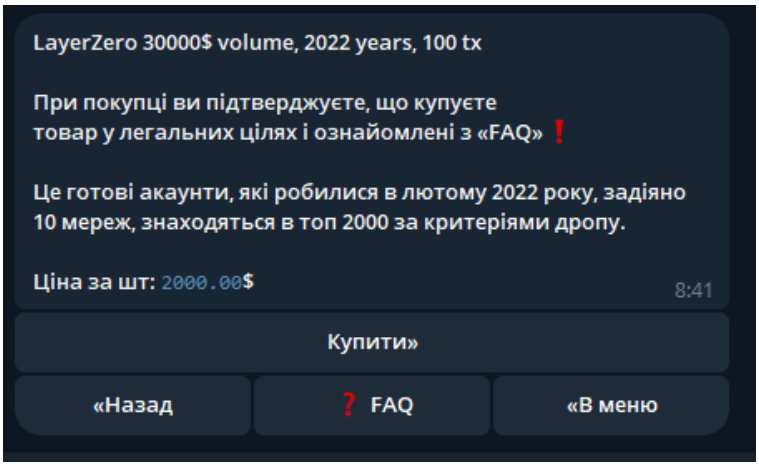

Рисунок 3.26 – Оформлення покупки

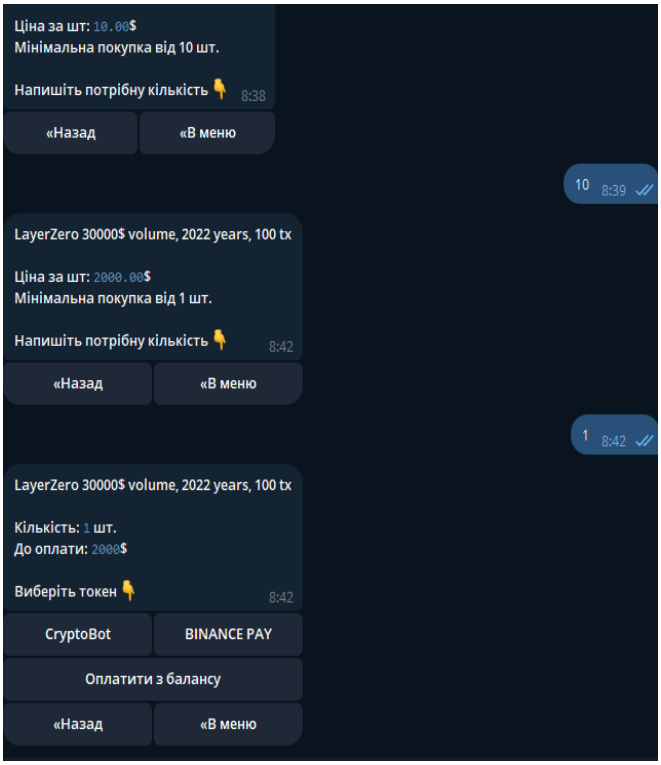

Рисунок 3.27 – Покупка LayerZero

Обираємо необхідну кількість і переходимо до сплати покупки.

У головному меню також, можна обрати пункт «Профіль» і переглянути інформацію про свій профіль:

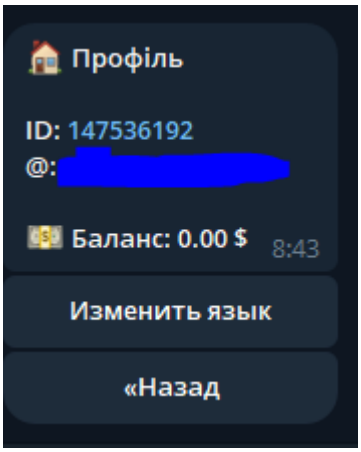

Рисунок 3.28 – Перегляд даних профілю

Тут наведені ID, ім’я користувача у телеграмі, а також баланс і можливість змінити мову.

Окрім цього користувач має змогу звернутися за підтримкою:

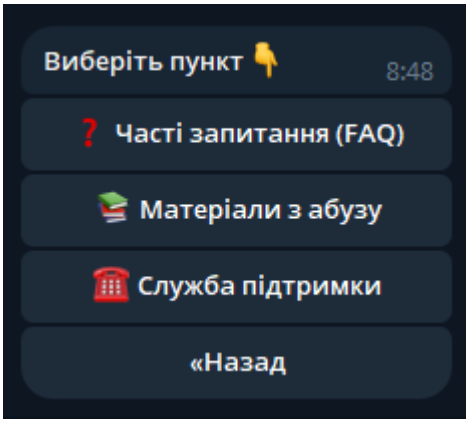

Рисунок 3.29 – Меню «Підтримка»

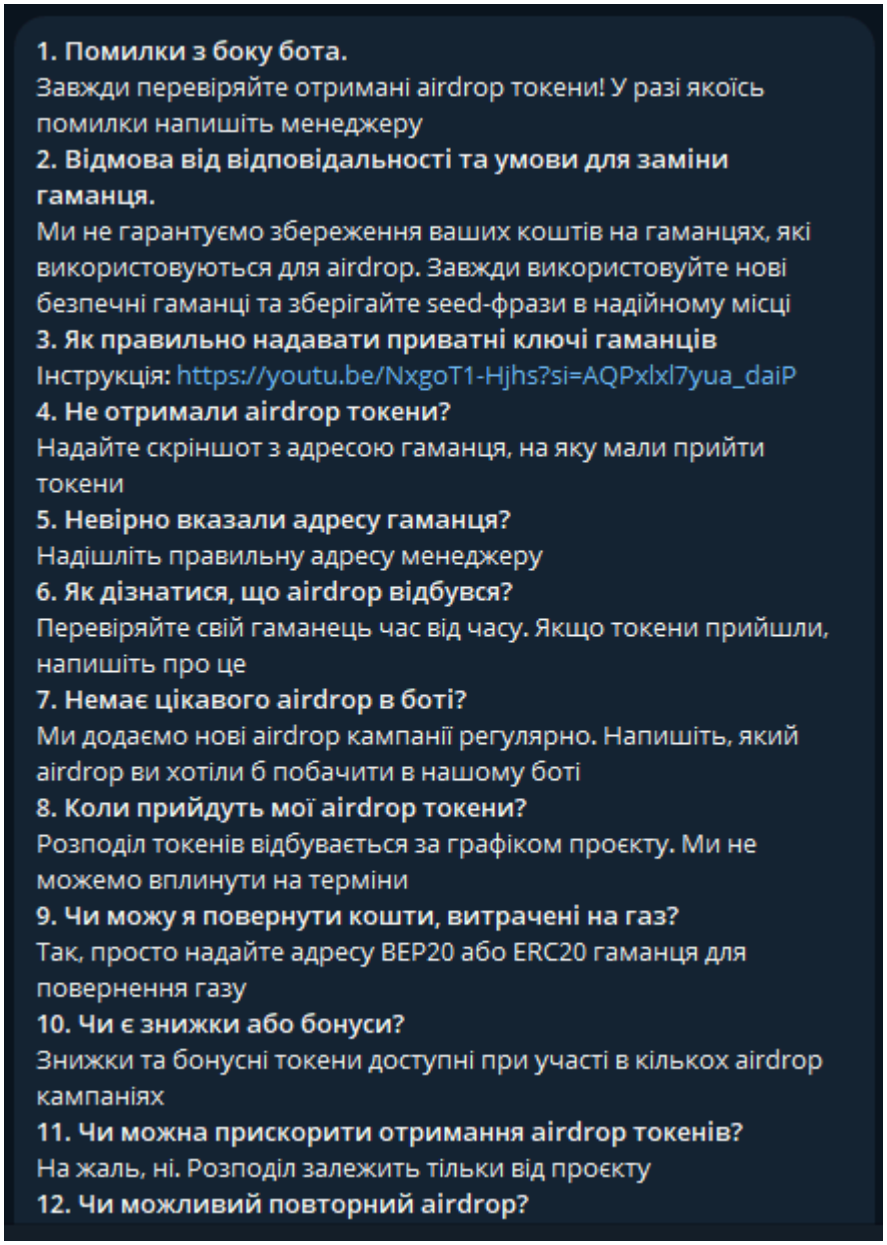

Рисунок 3.30 – Довідка

При виборі першого пункту користувачу буде надано відповіді на часті запитання.

При виборі другого варіанту також отримаємо наступну довідку:

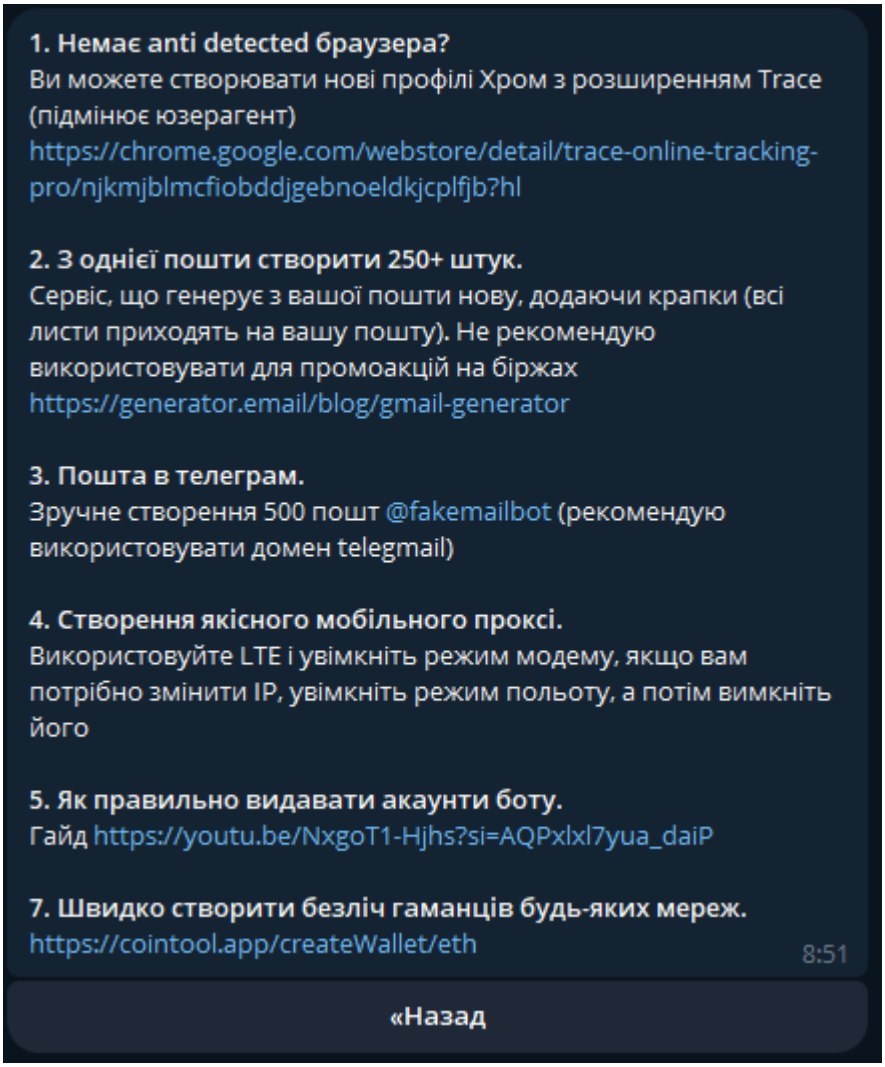

Рисунок 3.31 – «Матеріали»

І також, можна обрати пункт «Підтримка» для зв’язку з технічною підтримкою.

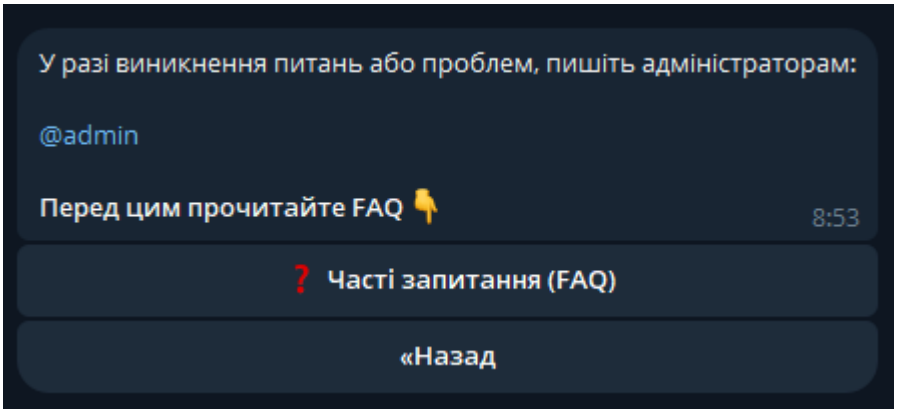

Рисунок 3.32 – «Підтримка»

У головному меню можемо обрати пункт «Партнерка», та отримаємо наступне повідомлення:

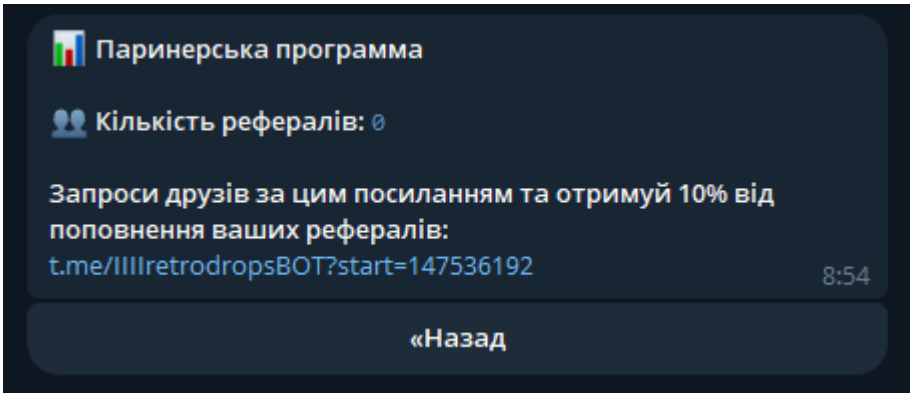

Рисунок 3.33 – «Партнерка»

Таким чином, було розроблено телеграм-бота, який надає користувачам змогу придбати певні товари, використовуючи для цього криптовалютні платежі.

ВИСНОВКИ

Тема розробки телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів є надзвичайно важливою в сучасному світі, де криптовалюти стають все більш популярними як форма інвестування та засіб оплати. Основними перевагами використання криптовалют є швидкість та низькі комісії у порівнянні з традиційними фіатними валютами. Проте, існує деяка складність у використанні криптовалют, зокрема, у впровадженні засобів для їх прийому та обробки.

Відповідно до визначених завдань, у першому розділі охарактеризовано явище блокчейн технологій, а також розглянуто Telegram як платформу для надання послуг.

У другому розділі, було проаналізовано наявні інструменти і технології розробки досліджуваних рішень, зокрема, було розглянуто мови програмування, які можна використати для розробки ботів, бази даних та платіжних систем. Для розробки програмного рішення було обрано мову програмування PHP та СУБД MySQL.

Третій розділ був присвячений розробці програмного рішення, зокрема було наведено лістинги головного модулю програмної системи, зокрема, обробка команд та повідомлень від користувачів. Також було протестовано розроблене рішення та продемонстровано основні його можливості.

У даній бакалаврській роботі було розроблено інноваційний телеграм-бот, який вирішує одну з ключових проблем існуючих платформ для надання послуг з оплатою в криптовалюті - необхідність реєстрації нового аккаунта, не зручна оплата, передача персональний даних третім лицям. Використання даних вже існуючих у Telegram профілів як автоматичний метод реєстрації дозволяє користувачам миттєво розпочати використання сервісу без додаткових кроків

верифікації чи введення даних. Це не тільки знижує бар'єри для входу, але й значно прискорює процес здійснення платежів.

Така автоматизована система обробки платежів дозволяє користувачам здійснювати операції без необхідності взаємодії з менеджерами або обслуговуючим персоналом, що гарантує швидку і зручну оплату та отримання товарів чи послуг. Окрім того, враховуючи велику популярність Telegram серед користувачів криптовалют, цей підхід робить бота особливо привабливим для цільової аудиторії, сприяючи ширшому прийняттю криптовалют в повсякденних фінансових операціях.

Таким чином, розвиток та впровадження боту для надання послуг з обробкою криптоплатежів сприяє стимулюванню інновацій у сфері електронної комерції та фінансових технологій. Це дозволяє бізнесам розширити свої можливості оплати та привернути нових клієнтів, що активно використовують криптовалютні активи. Такий підхід сприяє розвитку цифрової економіки та розширенню сфери застосування криптовалют у повсякденному житті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Tapscott. D & Tapscott. A. Блокчейн-революція. Канада: CITIC Press, 2016: 56-65.

2. Swann, M. Blockchain: New Economic Blueprint Guide США: O'Reilly, 2016: 66–67.

3. Antonopoulos, A.M. Mastering Bitcoin: Unlocking Digital Cryptocurrencies. США: O'Reilly Media, 2014: 58-59.

4. Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. США: Princeton University Press, 2016: 115-117.

5. Чжан Цзянь. Блокчейн визначає нову схему майбутніх фінансів та економіки. У: Пекін: Machinery Industry Press, 2017: 48–51.

6. Лу, Вен. Як блокчейн змінить світ. У: Пекін: Machinery Industry Press, 2016:21–30.

7. Mougayar, W. The Business Blockchain: Promise, Practice, and Application of the Next Internet Technology. США: Wiley, 2016: 85-86.

8. Swan, M. Blockchain: Blueprint for a New Economy. США: O'Reilly Media, 2015: 45-46.

9. Дивлячись на дно криниці. Блокчейн і великі дані для побудови розумної економіки. У: Пекін: Видавничий дім People's Post, 2017: 223–225.

10. Лю Ю.К. Чжан Сіньжун. Застосування технології блокчейн у сфері фінансів. In: Бухгалтерський облік, 2019 (5): 125–126.

11. У Ю. Чжоу Цай-Лі. Дослідження зміни моделі аудиту на основі технології блокчейн. У: Китайський академічний журнал, 2019 (3): 84–90.

12. Telegram Bot Platform URL: https://telegram.org/blog/bot-revolution

13. Telegram Bots: An introduction for developers URL: https://core.telegram.org/bots

14. Python Release Python 3.9.2 URL: https://www.python.org/downloads/release/python-392/

15. PHP Objects, Patterns, and Practice, Метт Зандстра, тестування та використання нових можливостей PHP 7+.

16. Taylor, A.G. SQL For Dummies. 9th ed. Hoboken, NJ: John Wiley & Sons, 2019: 45-47.

17. Beaulieu, A. Learning SQL. 3rd ed. Sebastopol, CA: O'Reilly Media, 2020: 112-113.

18. Halfond, W.G., Viegas, J., and Orso, A. A Classification of SQL Injection Attacks and Countermeasures. Proceedings of the IEEE International Symposium on Secure Software Engineering, 2006: 13-15.

19. Delisle, M. Mastering phpMyAdmin 3.4 for Effective MySQL Management. Birmingham, UK: Packt Publishing, 2012: 28-29.

20. Bashir, I., "Mastering Blockchain: Unlocking the Power of Cryptocurrencies, Smart Contracts, and Decentralized Applications". 2nd ed. Birmingham, UK: Packt Publishing, 2018: 156-158.

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення автоматизованих систем

Бакалаврська робота

на тему:

«Розробка телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів на мові програмування PHP»

Виконав:

студент групи ІСД-41

Божок Н. Ю.

Науковий керівник:

д.т.н., професор Сторчак К.П.

Київ 2024

Об'єкт, предмет та мета досліджень

Об'єкт дослідження - процес розробки та використання телеграм-боту для автоматизації продажів та обробки криптоплатежів.

Предмет дослідження - телеграм-бот для надання криптопослуг з автоматичною обробкою криптоплатежів.

Мета роботи - розробка та інтеграція системи криптоплатежів у телеграм-боті.

У рамках дослідження розроблено телеграм-бота для надання криптопослуг, який включає автоматичну обробку криптоплатежів, використовуючи мову програмування PHP. Розробка здійснювалася через інтеграцію з блокчейн технологіями та використання API. Процес включав налаштування бази даних в PHPMyAdmin для зберігання та управління даними, розробку інтерфейсу бота, функціоналу для проведення платежів у криптовалюті.

Етапи розробки

- Отримання API Telegram
- Створення Crypto Pay застосунку
- Розробка бази даних
- Розробка боту
- Тестування

Отримання API токену для взаємодії з Telegram

1 → 2 → 3

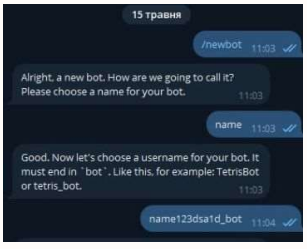

Рисунок 5.1 – Створення імені боту в BotFather

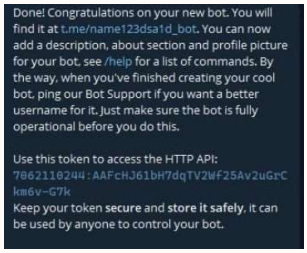

Рисунок 5.2 – Отримання API токену

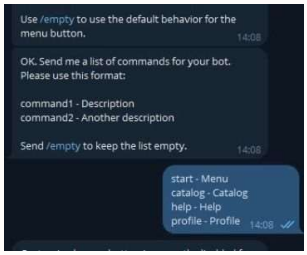

Рисунок 5.3 – Налаштування command menu

Створення Crypto Pay застосунку та отримання API токену

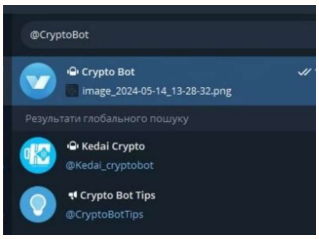

Рисунок 6.1 – Пошук та запуск CryptoBot

Рисунок 6.2 – Створення Crypto Pay застосунку

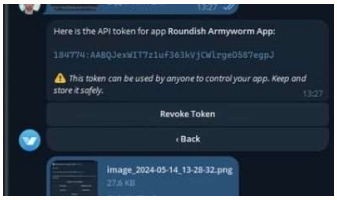

Рисунок 6.3 – Отримання API токену

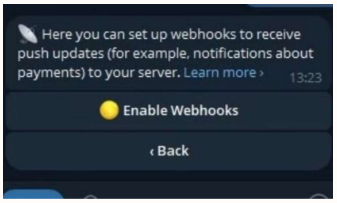

Рисунок 6.4 – Налаштування Webhook

Розробка бази даних

1 2 3

Створення таблиць

Для зберігання та управління даними, у базі даних MySQL створено кілька ключових таблиць, таких як users, items, sub_items, orders, mailing тощо. Кожна таблиця має відповідну структуру полів для збереження необхідної інформації.

Налаштування зв'язків

Між таблицями встановлено логічні зв'язки через використання зовнішніх ключів. Це забезпечує цілісність даних та можливість виконувати складні запити для отримання необхідної інформації.

Імпорт та експорт даних

Завдяки інтеграції з PHPMyAdmin, розробники можуть легко імпортувати та експортувати дані з бази, що спрощує процеси міграції, резервного копіювання та початкового наповнення даними.

Розробка бази даних

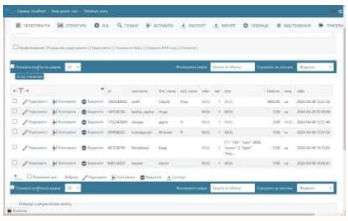

Рисунок 8.1 – **Таблиця 'users'**

Дані користувачів.

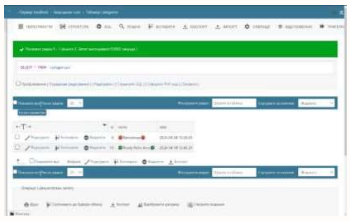

Рисунок 8.2 – **Таблиця 'categories'**

Категорії послуг.

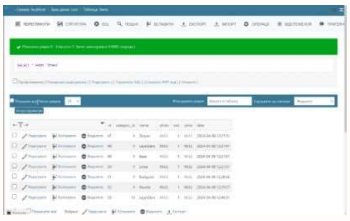

Рисунок 8.3 – **Таблиця 'items'**

Товари.

Розробка боту

Конфігурація

Файл `config.php` відіграє ключову роль, зберігаючи налаштування підключення до бази даних, токени API та інші важливі параметри, необхідні для функціонування бота.

Інтерфейс і обробка запитів

Файл `callback.php` відповідає за інтерфейс та обробляє зворотні виклики дозволяючи реагувати на дії користувача, такі як натискання кнопок.

Інтеграція з базою даних

Файл `db.php` забезпечує надійне підключення та взаємодію з базою даних, що дозволяє зберігати і обробляти всю необхідну інформацію для бота.

Навігація

`users_tab_selector.php` навігація між різними вкладками користувацького інтерфейсу в боті. Він забезпечує динамічну взаємодію, дозволяючи користувачам переходити між різними товарами в меню.

Обробка повідомлень

Скрипт `bot.php` є центральною точкою, яка відповідає за отримання та обробку вхідних повідомлень від користувачів, координуючи дії бота.

Локалізація

Файл `language.php` забезпечує багатомовну підтримку, дозволяючи користувачам взаємодіяти з ботом рідною мовою.

Тестування та демонстрація можливостей

Рисунок 10.1 – Користувацький інтерфейс

Рисунок 10.2 – Процес оплати

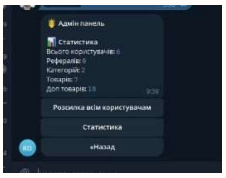

Рисунок 10.3 – Адміністративні можливості

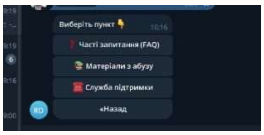

Рисунок 10.4 – Меню підтримки та матеріалів

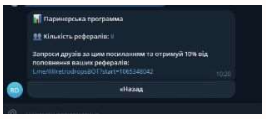

Рисунок 10.5 – Реферальна програма

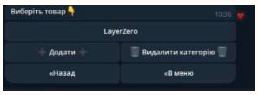

Рисунок 10.6 – Додавання та видалення категорій та товарів

Переваги використання Telegram-бота

Швидкість

Бот забезпечує миттєву обробку замовлень та платежів, що значно прискорює взаємодію з клієнтами.

Зручність

Інтеграція з Telegram дозволяє користувачам здійснювати покупки та платежі безпосередньо у знайомому месенджері.

Безпека

Завдяки використанню криптовалют та Telegram авторизації, бот забезпечує високий рівень безпеки та конфіденційності транзакцій.

Автоматизація

Автоматизація процесів продажу та обробки платежів значно підвищує ефективність та масштабованість бізнесу.

Апробація результатів дослідження

Божок Н.Ю. Розробка телеграм-боту для надання криптопослуг з автоматичною обробкою криптоплатежів на мові програмування PHP. Матеріали IV Всеукраїнська науково-практичної конференція «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез, ДУІКТ, м. Київ - К.: ДУІКТ, 2024. - С. 49.

Висновки

1. Тема розробки телеграм-бота для надання криптопослуг з автоматичною обробкою криптоплатежів є надзвичайно актуальною та важливою в сучасному світі, де криптовалюти стають все більш популярними як форма інвестування та засіб оплати. Основними перевагами використання криптовалют є швидкість та низькі комісії у порівнянні з традиційними фіатними валютами.
2. У ході виконання роботи було розроблено інноваційний телеграм-бот, який вирішує ключову проблему існуючих платформ з оплатою в криптовалюті - необхідність реєстрації нового акаунта та передачі персональних даних третім особам. Використання вже існуючих Telegram профілів як автоматичний метод реєстрації дозволяє користувачам миттєво розпочати використання сервісу без додаткових кроків верифікації.
3. Розроблена автоматизована система обробки платежів дозволяє користувачам здійснювати операції без необхідності взаємодії з персоналом, що гарантує швидку і зручну оплату та отримання товарів чи послуг. Враховуючи велику популярність Telegram серед користувачів криптовалют, такий підхід робить бота особливо привабливим для цільової аудиторії, сприяючи ширшому прийняттю криптовалют в повсякденних операціях.