**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ**
**ім. Ю.М. Потебні**
**ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ**

**КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

# Кваліфікаційна робота

перший (бакалаврський)
(рівень вищої освіти)

на тему **Телеграм бот для моніторингу ринку криптовалют**

Виконав: студент 4 курсу, групи 6.1211-пзс
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Програмне забезпечення систем
(код і назва освітньої програми)
Д. А. Газієв
(ініціали та прізвище )
Керівник доцент, к.т.н. В.І.Заяц
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент Директор ТОВ «Дісітел»
П.О.Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2025

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ**
**ім. Ю.М. Потебні**
**ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ**

Кафедра електроніки, інформаційних систем та програмного забезпечення

Рівень вищої освіти _____перший (бакалавський)_____

Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)

Освітня програма ___Програмне забезпечення систем____
(код та назва)

**ЗАТВЕРДЖУЮ**

Завідувач кафедри_____Т. В. Критська
“_01_” __березня_____2025_року

**З А В Д А Н Н Я**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ**

Газієву Даніл-Бєку Аліхановичу
(прізвище, ім’я, по батькові)

1.Тема роботи Телеграм бот для моніторингу ринку криптовалют

керівник роботи _____к.т.н., доцент Заяц Валерій Іванович
( прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від 27.12.2022 № 2029-с_____

2. Строк подання студентом кваліфікаційної роботи__02.06.2025_____

3. Вихідні дані бакалаврської роботи

- комплект нормативних документів ;
- технічне завдання до роботи.

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

- огляд та збір літератури стосовно теми кваліфікаційної роботи;
- огляд та аналіз існуючих рішень та аналогів;
- дослідження проблеми створення мобільних застосунків;
- створення програмного продукту та його опис;
- перелік вимог для роботи програми;
- тестування програмної системи та розробка висновків та пропозицій.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

_____слайдів презентації

6. Консультанти розділів бакалаврської роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата Завдання прийняв |
|---|---|---|
| | | |

7. Дата видачі завдання 03.03.2025

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання етапів кваліфікаційної роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз предметної області | 03.03-10.03.25 | виконано |
| 2 | Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником | 11.03-12.03.25 | виконано |
| 3 | Аналіз існуючих методів рішення | 13.03-14.03.25 | виконано |
| 4 | Аналіз методів розробки телеграм ботів | 15.03-20.03.25 | виконано |
| 5 | Аналіз засобів розробки телеграм ботів | 21.03-26.03.25 | виконано |
| 6 | Узгодження подальших дій з науковим керівником | 27.03-28.03.25 | виконано |
| 7 | Проєктування телеграм бота | 29.03-05.04.25 | виконано |
| 8 | Програмна реалізація телеграм бота | 06.04-16.04.25 | виконано |
| 9 | Представлення отриманих результатів науковому керівнику та узгодження плану подальшого дослідження | 17.04-19.04.25 | виконано |
| 10 | Реалізація користувацького інтерфейсу для телеграм бота | 20.04-01.05.25 | виконано |
| 11 | Тестування застосунку | 02.05-7.05.25 | виконано |
| 12 | Оформлення звіту | 08.05-15.05.25 | виконано |
| 13 | Оформлення презентації. | 14.05-20.05.25 | виконано |

Студент _____ Д.А. Газієв
( підпис ) (прізвище та ініціали)

Керівник роботи _____ В.І. Заяц
( підпис ) (прізвище та ініціали)

**Нормоконтроль пройдено**

Нормоконтролер _____ І.А. Скрипник
( підпис ) (прізвище та ініціали)

# АНОТАЦІЯ

Сторінок – 77.

Рисунків – 12.

Джерел – 15.

Газієв Д. А. Розробка Telegram-бота для взаємодії з криптовалютною біржею Binance: кваліфікаційна робота бакалавра спеціальності 121 "Інженерія програмного забезпечення" / наук. керівник к.т.н Заяц В.І Запоріжжя: ЗНУ. 2025. 77 с.

Метою даної кваліфікаційної роботи є розробка Telegram-бота для взаємодії з криптовалютною біржею Binance (Futures Testnet), що дозволяє користувачам керувати торговими операціями, налаштовувати параметри Take Profit/Stop Loss та автоматично відстежувати сигнали з Telegram-каналів, з використанням сучасних технологій та практик програмування. Для досягнення цієї мети були визначені основні функціональні вимоги до додатку, проведений аналіз необхідних інструментів та обґрунтовано вибір технологій.

У результаті дослідження було розроблено Telegram-бота, який забезпечить користувачам можливість керування торговими операціями (розміщення ринкових та лімітних ордерів, перегляд та закриття позицій), налаштування параметрів Take Profit/Stop Loss, а також автоматизованого відстеження торгових сигналів з Telegram-каналів.

Отриманий Telegram-бот може слугувати основою для подальшого розвитку, наприклад, додавання нових торгових стратегій, інтеграції з іншими біржами або розширення аналітичних функцій, та може бути корисним для індивідуальних трейдерів, які прагнуть автоматизувати частину своїх операцій або отримати зручний мобільний інструмент для торгівлі.

**Ключові слова:** *Telegram-бот, Binance, криптовалюта, торгівля, автоматизація, Python, Aiogram, CCXT, SQLite, API*.

# ANNOTATION

Pages – 77.

Pictures – 12.

Sources – 15.

Haziiev D.A. Development of a Telegram Bot for Interaction with the Binance Cryptocurrency Exchange: Bachelor's qualification work in the specialty 121 "Software Engineering" / scientific. head Zayac V.I. Zaporizhzhya: ZNU. 2025. 77 p.

The aim of this qualification work is to develop a Telegram bot for interacting with the Binance cryptocurrency exchange (Futures Testnet), allowing users to manage trading operations, configure Take Profit/Stop Loss parameters, and automatically track signals from Telegram channels, using modern technologies and programming practices. To achieve this goal, the main functional requirements for the application were identified, an analysis of necessary tools was conducted, and the selection of technologies was justified.

As a result of the research, a Telegram bot will be developed that will provide users with the ability to manage trading operations (placing market and limit orders, viewing and closing positions), configure Take Profit/Stop Loss parameters, as well as automated tracking of trading signals from Telegram channels. The application will be implemented using the Python programming language, the Aiogram framework for Telegram bot development, the CCXT library for interacting with the Binance API, and the SQLite DBMS for local data storage.

The obtained Telegram bot can serve as a basis for further development, for example, adding new trading strategies, integrating with other exchanges, or expanding analytical functions, and can be useful for individual traders seeking to automate part of their operations or obtain a convenient mobile tool for trading. The research results can be useful for software developers working on creating financial bots or integrating with trading APIs.

**Keywords:** *Telegram bot, Binance, cryptocurrency, trading, automation, Python, Aiogram, CCXT, SQLite, API.*

# ЗМІСТ

ВСТУП 7
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ..... 12
1.1 Огляд літературних джерел ..... 12
1.2 Аналіз існуючих систем та платформ для торгівлі криптовалютами .... 14
1.3 Тенденції розвитку торгових ботів та платформ для взаємодії з криптовалютними біржами ..... 21
1.4 Особливості розробки Telegram-ботів для взаємодії з криптовалютними біржами ..... 25
1.5 Постановка завдання ..... 28
2 ДОСЛІДЖЕННЯ ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ІНСТРУМЕНТІВ ДЛЯ ВЗАЄМОДІЇ З КРИПТОВАЛЮТНИМИ БІРЖАМИ ..... 32
2.1 Класифікація програмних засобів для торгівлі криптовалютами та взаємодії з біржами ..... 32
2.2 Види програмних інструментів для взаємодії з криптовалютними біржами ..... 35
2.3 Огляд методів та інструментів розробки Telegram-бота для взаємодії з криптовалютною біржею ..... 40
2.4 Засоби розробки Telegram-бота ..... 47
2.5 Ключові фреймворки та бібліотеки, використані в проєкті ..... 50
3 РОЗРОБКА TELEGRAM-БОТА ДЛЯ ВЗАЄМОДІЇ З КРИПТОВАЛЮТНОЮ БІРЖЕЮ BINANCE ..... 54
3.1 Опис предметної області ..... 54
3.2 Основні етапи створення Telegram-бота для взаємодії з криптовалютною біржею ..... 56
3.3 Проєктування Telegram-бота для взаємодії з криптовалютною біржею 60
3.4 Програмна реалізація застосунку ..... 69
ВИСНОВКИ ..... 74
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ..... 76

# ВСТУП

## Актуальність теми

У сучасному світі торгівля криптовалютами та використання автоматизованих торгових систем (ботів) набувають все більшої популярності як серед досвідчених трейдерів, так і серед новачків. Динамічність криптовалютного ринку, його цілодобова робота та висока волатильність створюють потребу в інструментах, що дозволяють оперативно реагувати на зміни та ефективно управляти торговими операціями. Telegram-боти, завдяки своїй доступності, зручності використання на мобільних пристроях та можливостям інтеграції, стали популярною платформою для розробки таких інструментів.

Існуючі рішення для взаємодії з криптовалютними біржами часто є складними для освоєння, не надають необхідного рівня гнучкості в налаштуванні або не мають функцій, які цікавлять конкретних користувачів, наприклад, автоматичне відстеження та реагування на торгові сигнали з інформаційних каналів. Таким чином, розробка Telegram-бота, що надає зручний інтерфейс для торгівлі на біржі Binance, можливість налаштування ключових параметрів управління ризиками (Take Profit/Stop Loss) та автоматизацію торгівлі на основі сигналів, є актуальною задачею в галузі інженерії програмного забезпечення.

## Мета дослідження

Метою даного дипломного проєкту є розробка Telegram-бота для взаємодії з ф'ючерсною тестовою платформою криптовалютної біржі Binance. Проєкт передбачає створення функціонального та зручного у використанні програмного продукту, який забезпечуватиме користувачам можливість здійснювати торгові операції (розміщення ринкових та лімітних ордерів), переглядати активні позиції та ордери, налаштовувати параметри Take Profit та Stop Loss, а також автоматично відстежувати та реагувати на торгові сигнали з визначених Telegram-каналів.

В результаті реалізації проєкту користувачі отримають інструмент для ефективного управління своїми торговими операціями на криптовалютній біржі через платформу Telegram.

**Завдання дослідження**

1. Провести огляд існуючих рішень (аналогів) у сфері Telegram-ботів для торгівлі криптовалютами, зокрема для взаємодії з біржею Binance. Вивчити їх функціональні можливості, інтерфейс користувача та особливості реалізації.
2. Проаналізувати існуючі підходи та технології, що використовуються для розробки Telegram-ботів та інтеграції з API криптовалютних бірж. Дослідити переваги, недоліки популярних програмних бібліотек та фреймворків (зокрема Aiogram, CCXT).
3. Обґрунтувати вибір оптимальних технологій та інструментів для розробки Telegram-бота, враховуючи вимоги до функціональності, надійності, швидкодії та зручності розробки (Python, Aiogram, CCXT, SQLite).
4. Спроектувати та реалізувати основний функціонал Telegram-бота, що включає:
    - Автентифікацію користувача та управління API-ключами Binance.
    - Розміщення ринкових та лімітних ордерів.
    - Перегляд та управління відкритими ордерами та позиціями.
    - Налаштування та застосування параметрів Take Profit та Stop Loss.
    - Розробку модуля для відстеження повідомлень у зазначених Telegram-каналах, парсингу торгових сигналів та автоматичного відкриття позицій на їх основі.
    - Зберігання налаштувань користувача у локальній базі даних SQLite.
5. Провести тестування розробленого Telegram-бота, оцінити його працездатність, зручність використання та відповідність поставленим функціональним вимогам. Визначити переваги та потенційні напрямки для подальшого розвитку.

**Об’єкт дослідження**

Об’єктом дослідження є процес розробки програмного забезпечення для автоматизації взаємодії з API криптовалютних бірж через платформу месенджера Telegram.

**Предмет дослідження**

Предметом дослідження є технології та методи розробки Telegram-бота для торгівлі на криптовалютній біржі Binance, що включають обробку команд користувача, взаємодію з біржовим API, управління торговими операціями, автоматичне відстеження сигналів та зберігання даних. Дослідження спрямоване на аналіз особливостей розробки інтерактивних ботів та їх інтеграції з зовнішніми сервісами.

**Методи дослідження**

**Теоретичні:** аналіз науково-технічної літератури, документації API біржі Binance, бібліотек Aiogram та CCXT; порівняльний аналіз існуючих рішень; системний аналіз для визначення архітектури та компонентів бота.

**Емпіричні:** проектування, програмування, тестування та налагодження програмного коду; експериментальна перевірка функціональності бота на тестовій платформі Binance Futures Testnet.

**Практичне значення одержаних результатів**

Практичне значення одержаних результатів полягає у створенні готового до використання Telegram-бота, який надає користувачам зручний інструмент для взаємодії з криптовалютною біржею Binance. Отримані результати дослідження та розробки мають наступні практичні переваги:

1. **Зручність та мобільність управління торгівлею:** Розроблений бот дозволяє користувачам керувати своїми торговими операціями безпосередньо з мобільного пристрою через звичний інтерфейс Telegram.

2. **Автоматизація рутинних операцій:** Функція відстеження каналів та автоматичного відкриття позицій на основі сигналів дозволяє автоматизувати частину торгового процесу та оперативно реагувати на ринкові можливості.
3. **Управління ризиками:** Можливість налаштування параметрів Take Profit та Stop Loss сприяє більш контрольованій торгівлі та мінімізації потенційних збитків.
4. **Основа для подальшого розвитку:** Розроблений бот може бути розширений новими функціями, торговими стратегіями, інтеграцією з іншими біржами або аналітичними інструментами.
5. **Навчальний інструмент:** Проєкт може слугувати прикладом розробки складних інтерактивних систем з використанням сучасних технологій Python для студентів та розробників-початківців.

**Глосарій**

*Telegram-бот*. Програмне забезпечення, що працює всередині месенджера Telegram та взаємодіє з користувачами через спеціальний API, виконуючи різноманітні команди та завдання.

*Криптовалютна біржа*. Онлайн-платформа, де користувачі можуть купувати, продавати та обмінювати криптовалюти. У контексті проєкту – Binance.

*API (Application Programming Interface).* Інтерфейс програмування додатків, що надає набір функцій та процедур для взаємодії між різними програмами. У проєкті використовується API біржі Binance та Telegram Bot API.

*CCXT (CryptoCurrency eXchange Trading Library).* Бібліотека для мови Python, що надає уніфікований доступ до API багатьох криптовалютних бірж.

*Aiogram.* Асинхронний фреймворк для мови Python, призначений для розробки Telegram-ботів.

*SQLite.* Легковажна реляційна система управління базами даних, що зберігає дані у файлі на локальному диску.

*Ордер (Торговий наказ).* Заявка на купівлю або продаж певного активу за визначеною ціною або за ринковою ціною.

*Ринковий ордер (Market Order).* Ордер на негайну купівлю або продаж за найкращою доступною на ринку ціною.

*Лімітний ордер (Limit Order).* Ордер на купівлю або продаж за вказаною або кращою ціною.

*Позиція (Position).* Кількість активу, якою володіє трейдер (довга позиція – купівля з розрахунком на зростання ціни, коротка позиція – продаж з розрахунком на падіння ціни).

*Take Profit (TP).* Рівень ціни, при досягненні якого відкрита прибуткова позиція автоматично закривається для фіксації прибутку.

*Stop Loss (SL).* Рівень ціни, при досягненні якого відкрита збиткова позиція автоматично закривається для обмеження збитків.

*Кредитне плече (Leverage).* Інструмент, що дозволяє трейдерам відкривати позиції на суму, що перевищує їх власний капітал, за рахунок позикових коштів біржі.

*Торговий сигнал.* Інформація або рекомендація щодо купівлі чи продажу певного фінансового активу, часто поширюється через спеціалізовані канали.

*FSM (Finite State Machine).* Машина скінченних станів, модель обчислень, що використовується в Aiogram для управління діалогами з користувачем.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Огляд літературних джерел

У сучасному фінансовому світі торгівля криптовалютами перетворилася з нішевого заняття на глобальний феномен, що приваблює мільйони інвесторів та трейдерів. Швидкий розвиток технологій блокчейн та зростання капіталізації криптовалютного ринку стимулюють попит на інноваційні інструменти для взаємодії з ним. Особливе місце серед таких інструментів займають торгові боти – програмні комплекси, призначені для автоматизації торгових стратегій та операцій на криптовалютних біржах. Платформа Telegram, завдяки своїй широкій аудиторії, гнучкому API для розробки ботів та можливості кросплатформного доступу, стала популярним середовищем для створення таких торгових помічників. Розробка Telegram-бота для взаємодії з провідними біржами, такими як Binance, є актуальним напрямком досліджень та розробки в галузі фінансових технологій (FinTech) та інженерії програмного забезпечення.

У науковій та технічній літературі активно обговорюються різні аспекти розробки та використання торгових ботів. Значна увага приділяється архітектурним підходам до створення таких систем. Наприклад, у працях [2] та [4] розглядаються моделі взаємодії бота з біржовим API, механізми обробки ринкових даних у реальному часі та реалізація торгових алгоритмів. Підкреслюється важливість асинхронної обробки запитів для забезпечення високої швидкодії та відмовостійкості, що є критичним для динамічного криптовалютного ринку.

Важливим напрямком досліджень є інтеграція ботів з API криптовалютних бірж. Бібліотеки, такі як CCXT (CryptoCurrency eXchange Trading Library), значно спрощують цей процес, надаючи уніфікований інтерфейс для роботи з десятками бірж, включаючи Binance [3]. У літературі [16] аналізуються переваги використання таких бібліотек, зокрема зменшення часу розробки та підвищення

портативності коду між різними торговими майданчиками. Також досліджуються питання безпеки при роботі з API-ключами, методи їх зберігання та використання для запобігання несанкціонованому доступу.

Розробка інтерфейсу користувача для торгових ботів у середовищі Telegram є окремою темою досліджень. Фреймворки, такі як Aiogram для Python, надають потужні інструменти для створення інтерактивних меню, обробки команд користувача та управління станами діалогів (FSM – Finite State Machine) [2]. У роботах [4] розглядаються принципи проектування інтуїтивно зрозумілих інтерфейсів для ботів, що дозволяють користувачам легко налаштовувати параметри торгівлі, переглядати статистику та керувати своїми активами.

Особливий інтерес становить функціонал автоматичного відстеження торгових сигналів з інформаційних каналів Telegram. Ця можливість дозволяє трейдерам оперативно реагувати на ринкові новини та рекомендації. У літературі [3] досліджуються методи парсингу текстових повідомлень, виділення ключової інформації (назва активу, напрямок угоди, цілі, стоп-лоси) та інтеграції цих даних у логіку торгового бота. Обговорюються виклики, пов'язані з різноманітністю форматів сигналів та необхідністю розробки гнучких алгоритмів їх обробки.

Зберігання даних користувача, таких як API-ключі, налаштування TP/SL, списки обраних монет або параметри трекера каналів, є ще одним важливим аспектом. У дослідженнях [4] порівнюються різні підходи до зберігання даних для Telegram-ботів, від простих файлових сховищ до використання реляційних (наприклад, SQLite, PostgreSQL) або NoSQL баз даних [10]. Вибір конкретної СУБД залежить від обсягу даних, вимог до швидкодії та складності запитів. Для проєктів, подібних до розроблюваного, часто обирають легковажні СУБД типу SQLite через простоту інтеграції та мінімальні накладні витрати.

Література також містить аналіз існуючих Telegram-ботів для торгівлі криптовалютами. Досліджуються їхні функціональні можливості, переваги та недоліки. Наприклад, деякі боти можуть спеціалізуватися на певних торгових стратегіях (скальпінг, арбітраж), інші надають ширший набір інструментів для ручного

та напівавтоматичного трейдингу. Аналіз аналогів [15] дозволяє виявити незайняті ніші та потреби користувачів, які можуть бути задоволені розроблюваним програмним продуктом.

Окрім огляду технічних аспектів, варто звернути увагу на тенденції та перспективи розвитку торгових ботів. Зокрема, досліджуються можливості інтеграції з алгоритмами машинного навчання для прогнозування ринкових рухів, розширення функціональності за допомогою аналітичних інструментів (наприклад, технічний аналіз графіків безпосередньо в боті), покращення механізмів управління ризиками та забезпечення високого рівня безпеки персональних даних та коштів користувачів.

Отже, огляд літературних джерел засвідчує високу актуальність та значний дослідницький інтерес до розробки Telegram-ботів для взаємодії з криптовалютними біржами. Комплексне вивчення існуючих технологій, архітектурних підходів, аналіз функціональних можливостей аналогічних рішень та врахування сучасних тенденцій дозволить створити ефективний, надійний та зручний у використанні програмний продукт, що задовольнятиме потреби сучасних трейдерів.

## 1.2 Аналіз існуючих систем та платформ для торгівлі криптовалютами

У сучасному цифровому світі існує значна кількість інструментів та платформ, призначених для торгівлі криптовалютами. Вони варіюються від повноцінних десктопних та веб-застосунків до мобільних додатків та спеціалізованих Telegram-ботів. Проаналізувавши доступні рішення, можна виділити декілька ключових категорій та популярних представників, які надають функціонал, схожий або дотичний до того, що розробляється в рамках даного дипломного проєкту.

**Платформи для автоматизованої торгівлі (Trading Bot Platforms)**

Існує низка комерційних та відкритих платформ, що дозволяють користувачам створювати, тестувати та запускати торгових ботів без глибоких знань у програмуванні або з використанням готових шаблонів.

**3Commas:** Одна з найпопулярніших платформ, що надає широкий набір інструментів для автоматизації торгівлі на багатьох біржах, включаючи Binance (див. рис. 1).

- **Переваги:**
  - Інтуїтивно зрозумілий веб-інтерфейс.
  - Великий вибір готових стратегій (GRID боти, DCA боти, опціонні боти).
  - Можливість копіювання сигналів та налаштувань інших трейдерів (SmartTrade, Copy trading).
  - Підтримка підключення до Telegram для отримання сповіщень та виконання деяких команд.
  - Розширені інструменти для управління портфелем та аналітики.
- **Недоліки:**
  - Платна підписка для доступу до повного функціоналу.
  - Обмежені можливості для глибокої кастомізації стратегій без використання API.
  - Залежність від сторонньої платформи та її надійності.

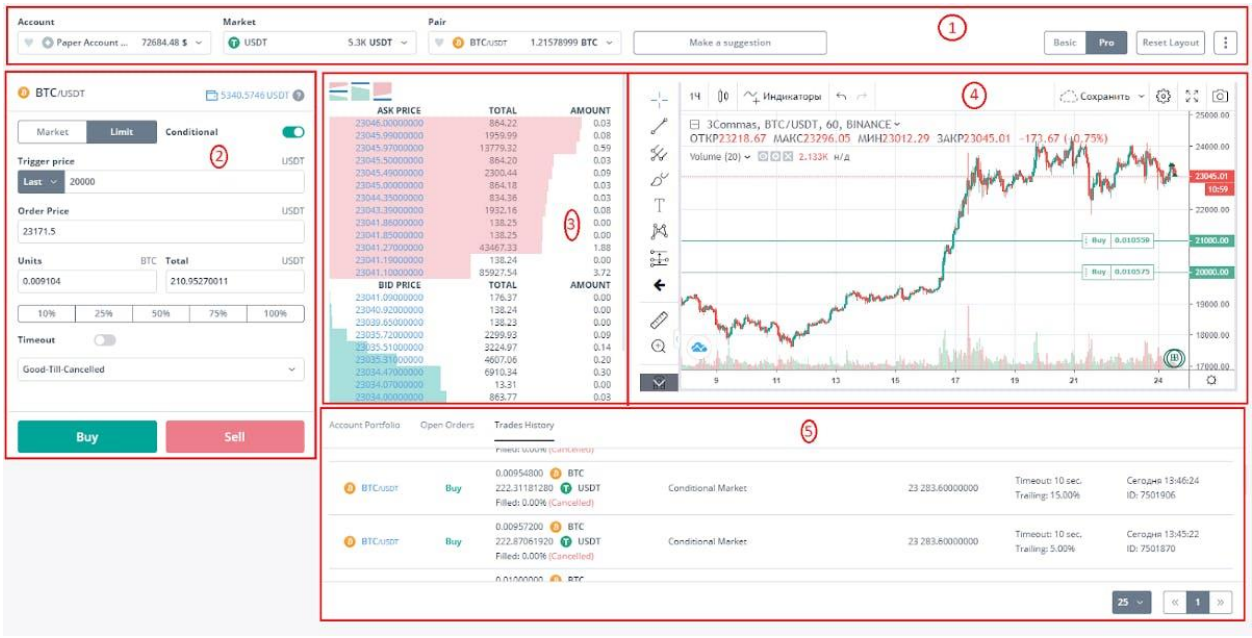



Рисунок *1— Інтерфейс платформи 3Commas*

**Cryptohopper:** Ще одна відома платформа для створення торгових ботів (див. рис 2).

- **Переваги:**
  - Хмарна платформа, не потребує постійно увімкненого комп'ютера.
  - Маркетплейс стратегій та сигналів.
  - Інструменти для бектестінгу та паперової торгівлі.
  - Візуальний конструктор стратегій.
- **Недоліки:**
  - Також працює за моделлю платної підписки.
  - Інтерфейс може здатися перевантаженим для новачків.
  - Деякі користувачі відзначають складнощі з налаштуванням деяких просунутих функцій.

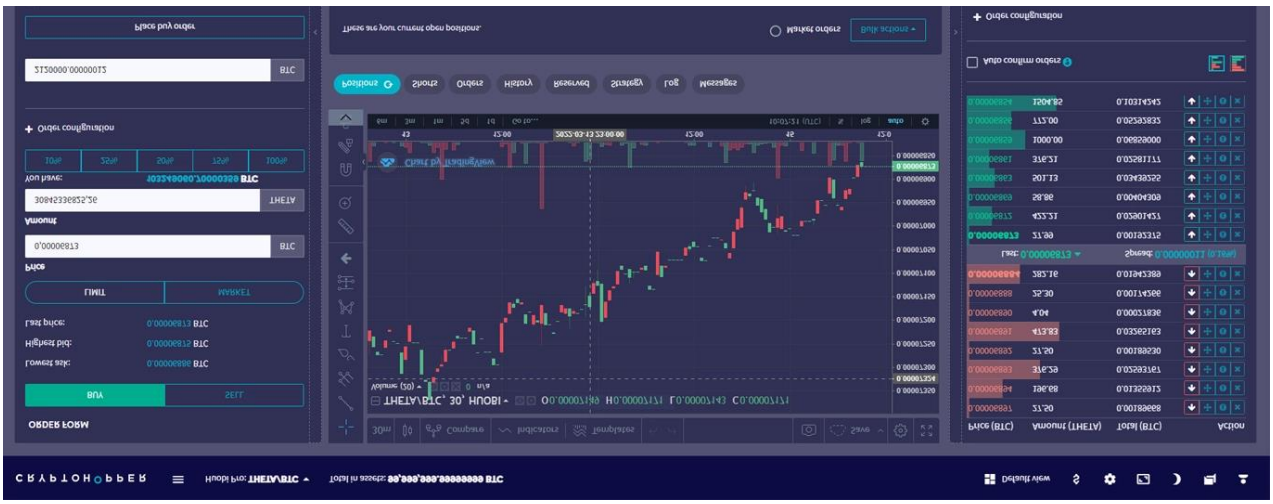


Рисунок *2 — Інтерфейс платформи Cryptohopper*

**Pionex:** Біржа з вбудованими торговими ботами.

- **Переваги:**
  - Безкоштовний доступ до 16+ типів торгових ботів (GRID, DCA, арбітражні тощо).
  - Низькі торгові комісії.
  - Простий у використанні мобільний додаток.
- **Недоліки:**
  - Обмежений вибір бірж (фактично, це сама біржа Pionex).
  - Менша гнучкість у налаштуванні ботів порівняно з 3Commas чи Cryptohopper.
  - Орієнтована переважно на автоматизовану торгівлю, менше інструментів для ручного трейдингу.

**Існуючі Telegram-боти для торгівлі**

Ринок Telegram-ботів для торгівлі криптовалютами є досить насиченим, проте багато з них мають специфічний функціонал або обмеження.

**Cornix Trading Bot:** Популярний бот для автоматичного копіювання сигналів з Telegram-каналів.

- **Переваги:**
  - Підтримка багатьох бірж, включаючи Binance.

- Автоматичне зчитування та виконання сигналів з підтримуваних каналів.
- Налаштування параметрів торгівлі (тейк-профіт, стоп-лос, трейлінг-стоп).
- Детальна статистика та управління угодами.

- **Недоліки:**
  - Платна підписка.
  - Основний фокус – саме на копіюванні сигналів, а не на наданні повноцінного інтерфейсу для ручної торгівлі.
  - Користувач залежить від якості сигналів у каналах, які він обирає.

**Кастомні та нішеві Telegram-боти:** Існує безліч менш відомих або приватних ботів, розроблених окремими трейдерами чи невеликими командами.

- **Переваги:**
  - Можуть бути високоспеціалізованими під конкретні стратегії або потреби.
  - Потенційно можуть бути безкоштовними або мати нижчу вартість.
- **Недоліки:**
  - Часто мають обмежений функціонал.
  - Відсутність публічної підтримки та гарантій надійності.
  - Ризики безпеки, пов'язані з довірою API-ключів невідомим розробникам.
  - Складність пошуку та оцінки якості таких ботів.

**Проєкт, що розробляється, в контексті аналогів:**

На відміну від великих платформ типу 3Commas чи Cryptohopper, які пропонують широкий, але часто платний та дещо стандартизований функціонал, розроблюваний Telegram-бот орієнтований на надання гнучкого та безкоштовного (у контексті використання самого бота, не враховуючи комісії біржі) інструменту для користувачів. Ключовими перевагами можуть стати:

- *Поєднання ручного трейдингу та автоматизації:* Надання інтерфейсу для розміщення ринкових/лімітних ордерів, перегляду позицій та одночасно функціоналу відстеження каналів.
- *Кастомізація відстеження каналів:* Користувач сам визначає джерело сигналів, параметри фільтрації (ETA, співвідношення ризик/прибуток), що дає більший контроль.
- *Прозорість та контроль:* Оскільки код розробляється в рамках дипломного проєкту, користувач (теоретично) може мати більше розуміння його роботи, на відміну від "чорних скриньок" комерційних рішень.
- *Відсутність плати за підписку:* Це робить бота доступнішим для широкого кола користувачів, особливо для тих, хто тільки починає знайомство з автоматизованою торгівлею.

**Тенденція популярності платформ та інструментів**

Надання точної статистики щодо розподілу користувачів між різними торговими ботами є складним завданням, оскільки багато платформ не публікують такі дані відкрито. Однак, можна відзначити загальні тенденції:

**Binance** є однією з найбільших криптовалютних бірж у світі за обсягом торгів та кількістю користувачів, що робить інтеграцію з нею пріоритетною для багатьох торгових інструментів. Згідно з даними SimilarWeb (станом на 2025 рік), Binance регулярно входить до топ-100 найвідвідуваніших веб-сайтів у категорії "Фінанси > Інвестування".

**Telegram** як платформа для ботів має понад 700 мільйонів активних користувачів щомісяця (за даними Telegram станом на 2025 рік), що створює величезну потенційну аудиторію для ботів різного призначення, включаючи торгових.

Платформи типу **3Commas** та **Cryptohopper** звітують про сотні тисяч зареєстрованих користувачів, що свідчить про значний попит на готові рішення для автоматизації торгівлі.

Таблиця *1 – Порівняльний аналіз існуючих систем та платформ*

| **Характеристика** | **3Commas** | **Cryptohopper** | **Pionex** | **Cornix Trading Bot** | **Розроблюваний бот** |
|---|---|---|---|---|---|
| **Тип платформи** | Веб-платформа, моб. додаток | Веб-платформа | Біржа з вбудованими ботами | Telegram-бот | **Telegram-бот** |
| **Основний функціонал** | Авто-трейдинг, SmartTrade, портфель | Авто-трейдинг, бектестінг, маркетплейс | Вбудовані боти (GRID, DCA) | Копіювання сигналів з Telegram-каналів | **Ручна торгівля, авто-трейдинг (сигнали), TP/SL** |
| **Підтримка Binance** | Так | Так | Обмежено (це сама біржа) | Так | **Так** |
| **Інтерфейс** | Веб, мобільний | Веб | Веб, мобільний | Telegram | **Telegram** |
| **Модель оплати** | Платна підписка | Платна підписка | Безкоштовні боти, комісії за торгівлю | Платна підписка | **Безкоштовно** |
| **Кастомізація** | Обмежена для базових планів | Висока через конструктор та API | Обмежена | Налаштування параметрів копіювання | **Висока (налаштування трекера, монет)** |
| **Відстеження каналів** | Через інтеграції або сторонні сервіси | Через інтеграції або сторонні сервіси | Ні | Основна функція | **Вбудована основна функція** |

Важливо зауважити, що ринок торгових інструментів для криптовалют є дуже динамічним. Нові проєкти з'являються регулярно, а популярність існуючих може змінюватися залежно від ринкових умов, оновлень функціоналу та маркетингових зусиль розробників.

**Порівняльна таблиця ключових аналогів**

Для наочності, основні характеристики розглянутих аналогів можна звести у таблицю (Таблиця 1).

Аналізуючи функціонал та можливості існуючих систем, можна зробити висновки щодо позиціонування власної розробки. Розроблюваний Telegram-бот має на меті запропонувати користувачам збалансоване рішення, що поєднує зручність Telegram-інтерфейсу, базові функції для ручного управління торгівлею та унікальну можливість гнучкого налаштування автоматичного відстеження сигналів з обраних каналів без необхідності платної підписки, що є його ключовою відмінністю та перевагою для певної категорії користувачів.

## 1.3 Тенденції розвитку торгових ботів та платформ для взаємодії з криптовалютними біржами

Сфера розробки програмних інструментів для взаємодії з криптовалютними біржами, зокрема торгових ботів, є надзвичайно динамічною та постійно розвивається, реагуючи на технологічні інновації, зміни вподобань користувачів та еволюцію самого криптовалютного ринку. Аналіз актуальних досліджень, публікацій та ринкових трендів дозволяє виділити декілька ключових тенденцій, що визначатимуть майбутнє таких систем.

1. **Поглиблення автоматизації та використання штучного інтелекту (ШІ) і машинного навчання (МН):**
   Якщо раніше торгові боти переважно виконували заздалегідь запрограмовані алгоритми (наприклад, слідування за індикаторами технічного аналізу або виконання простих GRID/DCA стратегій), то зараз спостерігається зростаючий інтерес до інтеграції більш складних моделей.

- *Предиктивна аналітика:* Розробники прагнуть навчити ботів аналізувати великі обсяги історичних та ринкових даних для прогнозування цінових рухів. Це включає використання нейронних мереж (LSTM, RNN) для аналізу часових рядів, обробку новинного фону (Sentiment Analysis) за допомогою методів обробки природної мови (NLP) для оцінки впливу новин на ринок.
- *Адаптивні стратегії:* Боти майбутнього зможуть самостійно адаптувати свої торгові стратегії залежно від поточної ринкової ситуації (волатильність, тренд, флет), замість того, щоб сліпо виконувати один і той самий алгоритм.
- *Оптимізація параметрів:* Алгоритми МН можуть використовуватися для автоматичної оптимізації параметрів торгових стратегій (наприклад, періоди індикаторів, рівні TP/SL) на основі бектестінгу та поточної ефективності.

2. **Розширення функціоналу та інтеграція з DeFi (децентралізовані фінанси):**

   Сучасні користувачі очікують від торгових інструментів не лише базових функцій купівлі/продажу.

   - *Управління портфелем:* Боти все частіше включають інструменти для комплексного управління криптовалютним портфелем, відстеження його прибутковості, диверсифікації та ребалансування.
   - *Інтеграція з DeFi протоколами:* Зростання популярності DeFi відкриває нові можливості для ботів, такі як автоматизація участі в пулах ліквідності, фармінгу, стейкінгу або арбітражу між централізованими (CEX) та децентралізованими (DEX) біржами.
   - *Підтримка NFT та метавсесвітів:* Хоча це ще досить нова ніша, вже з'являються інструменти для автоматизації операцій з NFT або взаємодії з економіками метавсесвітів.

3. **Посилення уваги до безпеки та управління ризиками:** З огляду на високі ризики, пов'язані з торгівлею криптовалютами, та часті випадки зломів, безпека стає пріоритетом.
    - *Розширені механізми управління ризиками:* Окрім стандартних TP/SL, розробники впроваджують більш складні механізми, такі як трейлінг-стопи, динамічне управління розміром позиції (position sizing), можливість встановлення максимального допустимого денного збитку.
    - *Безпека API-ключів:* Вдосконалюються методи захисту API-ключів, наприклад, через обмеження IP-адрес, використання окремих ключів для різних операцій, шифрування ключів на стороні користувача або використання апаратних гаманців для підтвердження операцій.
    - *Прозорість та аудит:* Для комерційних платформ зростає вимога до прозорості їхніх алгоритмів та можливості незалежного аудиту безпеки.
4. **Кросплатформність та покращення користувацького досвіду (UX):** Користувачі прагнуть мати доступ до своїх торгових інструментів з будь-якого пристрою та очікують інтуїтивно зрозумілого інтерфейсу.
    - *Мобільні інтерфейси та Telegram-боти:* Telegram-боти, подібні до розроблюваного, є яскравим прикладом цієї тенденції, надаючи зручний мобільний доступ до торгових функцій.
    - *Веб-інтерфейси та десктопні додатки:* Для більш глибокого аналізу та налаштування складних стратегій все ще популярними залишаються повноцінні веб-платформи та десктопні додатки.
    - *Персоналізація та кастомізація:* Користувачі хочуть мати можливість налаштовувати інтерфейс та функціонал під свої індивідуальні потреби.
5. **Соціальний трейдинг та копіювання угод (Copy Trading):** Можливість спостерігати за успішними трейдерами та автоматично копіювати їхні угоди залишається популярною.

- *Інтеграція з соціальними платформами:* Боти можуть інтегруватися з соціальними мережами або спеціалізованими платформами для обміну торговими ідеями та сигналами.
- *Розвиток маркетплейсів сигналів та стратегій:* Платформи, де трейдери можуть ділитися своїми сигналами або готовими стратегіями для ботів, продовжують розвиватися. Це включає і відстеження сигналів з публічних Telegram-каналів, що реалізовано в даному проєкті.

6. **Регуляторні зміни та їх вплив:**

   Криптовалютний ринок поступово підпадає під все більший регуляторний тиск. Це може вплинути на розробників торгових ботів.

   - *Необхідність дотримання KYC/AML:* Для взаємодії з деякими біржами або функціями ботам може знадобитися враховувати процедури "Знай свого клієнта" (KYC) та "Протидії відмиванню грошей" (AML).
   - *Адаптація до нових правил:* Розробники ботів повинні бути готові швидко адаптувати свої продукти до змін у законодавстві різних країн.

7. **Зростання освітнього контенту та спільнот:**

Навколо торгових ботів та алгоритмічного трейдингу формуються активні спільноти, де користувачі обмінюються досвідом, стратегіями та допомагають новачкам. Зростає кількість освітніх матеріалів, курсів та вебінарів, присвячених цій темі.

Ці тенденції свідчать про те, що індустрія торгових ботів для криптовалют перебуває на стадії активного росту та інновацій. Розуміння цих трендів є важливим для розробки конкурентоспроможного та затребуваного програмного продукту, такого як Telegram-бот для взаємодії з біржею Binance. Проєкт, що розробляється, частково відповідає цим тенденціям, зокрема, надаючи мобільний доступ через Telegram, реалізуючи автоматизацію через відстеження сигналів та до-

зволяючи користувачеві контролювати ризики через налаштування TP/SL. Подальший розвиток може включати інтеграцію більш складних аналітичних інструментів або елементів ШІ.

## 1.4 Особливості розробки Telegram-ботів для взаємодії з криптовалютними біржами

Розробка Telegram-ботів, призначених для взаємодії з криптовалютними біржами, такими як Binance, має низку специфічних особливостей, які відрізняють їх від загальних месенджер-ботів або інших типів програмного забезпечення. Ці особливості зумовлені як самою платформою Telegram, так і специфікою предметної області – торгівлі криптовалютами.

1. **Взаємодія з Telegram Bot API:**

***Асинхронність:*** Telegram Bot API працює асинхронно. Це означає, що бот повинен бути здатним обробляти безліч одночасних запитів від користувачів та повідомлень від API Telegram без блокування. Використання асинхронних фреймворків, таких як Aiogram в Python, є ключовим для створення ефективних та чутливих ботів [2, 12].

***Обмеження API:*** Telegram встановлює певні ліміти на кількість запитів, які бот може надсилати (rate limits), а також на розмір повідомлень та файлів. Розробники повинні враховувати ці обмеження, щоб уникнути тимчасового блокування бота або помилок у його роботі.

***Типи оновлень (Updates):*** Боти отримують інформацію від Telegram у вигляді об'єктів Update. Це можуть бути нові повідомлення, натискання на кнопки, зміни в групах тощо. Бот повинен коректно обробляти різні типи оновлень, що стосуються його функціоналу.

***Інтерфейсні елементи:*** Telegram надає багатий набір інструментів для створення інтерактивного інтерфейсу:

- *Клавіатури (Keyboards):* Звичайні клавіатури (ReplyKeyboardMarkup) для швидких відповідей та вбудовані клавіатури (InlineKeyboardMarkup) з кнопками, що надсилають callback-дані. Це основний спосіб навігації та взаємодії у розроблюваному боті.
- *Команди:* Повідомлення, що починаються з /, дозволяють користувачам викликати певні функції бота.
- *Форматування тексту:* Підтримка Markdown або HTML для візуального оформлення повідомлень, що важливо для чіткого подання торгової інформації (ціни, обсяги, PNL).

2. **Інтеграція з API криптовалютної біржі (на прикладі Binance):**

***Автентифікація та безпека API-ключів:*** Бот повинен безпечно отримувати, зберігати (якщо це передбачено, хоча в даному проєкті ключі вводяться користувачем і зберігаються локально в БД) та використовувати API-ключі користувача для доступу до його акаунту на біржі. Це критичний аспект безпеки, оскільки компрометація ключів може призвести до фінансових втрат. Важливо використовувати ключі з обмеженими правами (наприклад, тільки для торгівлі, без права на виведення коштів).

***Обробка відповідей API біржі:*** API бірж повертають дані у структурованому форматі (зазвичай JSON). Бот повинен коректно парсити ці відповіді, обробляти можливі помилки (наприклад, недостатньо коштів, невірний символ, помилки мережі, перевищення лімітів запитів до API біржі) та надавати користувачеві зрозумілу інформацію[13].

***Синхронізація даних:*** Інформація про баланс, відкриті ордери та позиції на біржі може змінюватися незалежно від дій бота (наприклад, якщо користувач торгує через веб-інтерфейс біржі). Бот повинен періодично або за запитом оновлювати ці дані для відображення актуального стану.

***Специфіка ф'ючерсної торгівлі (для Binance Futures):*** Торгівля ф'ючерсами має свої особливості, такі як кредитне плече, типи маржі (ізольована, пере-

хресна), ліквідації, механізми фінансування (фандінг). Бот, що працює з ф'ючерсами (навіть на Testnet, як у даному проєкті), повинен враховувати ці аспекти при формуванні ордерів та відображенні інформації.

3. **Управління станами (State Management):**

Для реалізації багатоетапних діалогів (наприклад, введення параметрів ордера: символ, ціна, кількість, плече) необхідно використовувати механізми управління станами. Фреймворк Aiogram надає для цього інструмент FSM (Finite State Machine), який дозволяє боту "пам'ятати", на якому етапі діалогу з користувачем він знаходиться. Це реалізовано в обробниках команд market.py, orders.py, api_keys.py, tp_sl.py та channel_tracker.py.

4. **Локальне зберігання даних:**

Бот може потребувати зберігання налаштувань користувача (API-ключі, обрані монети, налаштування TP/SL, параметри трекера каналів). У даному проєкті для цього використовується SQLite – легковажна файлова СУБД, що зручно для невеликих та середніх проєктів, де не потрібна складна серверна інфраструктура бази даних. Важливо забезпечити коректну структуру бази даних та безпечне зберігання чутливої інформації.

5. **Обробка помилок та логування:**

- Надійна робота бота вимагає ретельної обробки можливих помилок: помилки мережі, невірні дані від користувача, помилки API Telegram або біржі.
- Детальне логування дій бота, запитів до API, отриманих відповідей та помилок є критично важливим для налагодження, моніторингу роботи та аналізу інцидентів. У проєкті реалізовано базове логування, особливо в модулі channel_tracker.py.

6. **Специфіка автоматичного відстеження каналів:**

- *Парсинг повідомлень:* Бот повинен аналізувати текстові повідомлення з каналів-джерел, використовуючи регулярні вирази або інші методи NLP для вилучення торгової інформації (назва монети, напрямок, ціна входу, цілі, ETA тощо).

- *Фільтрація та валідація сигналів:* Не всі повідомлення в каналі є торговими сигналами, і не всі сигнали є якісними. Бот повинен мати логіку фільтрації (наприклад, за ключовими словами, форматом, параметрами ETA, співвідношенням ризик/прибуток, як реалізовано в ChannelTracker).
- *Управління екземплярами трекерів:* Якщо бот підтримує декількох користувачів, кожен з яких може налаштувати власний трекер, необхідно коректно управляти екземплярами цих трекерів, їх запуском та зупинкою. У проєкті це реалізовано через словник tracker_instances та механізм відновлення активних трекерів при старті бота.

7. **Асинхронне виконання тривалих завдань:**

Деякі операції, такі як постійне відстеження каналів або очікування виконання ордерів, можуть бути тривалими. Щоб не блокувати основного бота, такі завдання повинні виконуватися асинхронно, наприклад, у фонових задачах asyncio [2].

Враховуючи ці особливості, розробка Telegram-бота для торгівлі криптовалютами є комплексним завданням, що вимагає знань у сферах програмування на Python, роботи з асинхронними фреймворками, інтеграції з API, управління базами даних та розуміння специфіки фінансових ринків. Проєкт, що розробляється, намагається врахувати ці аспекти для створення функціонального та корисного інструменту для трейдерів.

### 1.5 Постановка завдання

Метою даного дипломного проєкту є розробка Telegram-бота для взаємодії з ф'ючерсною платформою криптовалютної біржі Binance. Бот повинен надавати користувачам зручний інтерфейс для управління торговими операціями, включаючи розміщення ринкових та лімітних ордерів, перегляд та управління активними позиціями та ордерами, налаштування параметрів Take Profit та Stop Loss,

а також забезпечувати функціонал автоматичного відстеження торгових сигналів з визначених Telegram-каналів та відкриття позицій на їх основі.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. **Провести огляд літературних джерел** та існуючих досліджень, пов'язаних з розробкою Telegram-ботів, взаємодією з API криптовалютних бірж (зокрема Binance), використанням бібліотек CCXT та Aiogram, а також методів обробки та зберігання даних (SQLite).
2. **Здійснити аналіз існуючих систем та платформ** для торгівлі криптовалютами, включаючи комерційні торгові боти та інші Telegram-боти з подібним функціоналом, з метою виявлення їх переваг, недоліків та визначення конкурентних переваг розроблюваного рішення.
3. **Дослідити сучасні тенденції розвитку** торгових ботів, платформ для алгоритмічного трейдингу та технологій, що використовуються в цій галузі (наприклад, ШІ, МН, інтеграція з DeFi), для визначення потенційних напрямків удосконалення та актуальності розроблюваного функціоналу.
4. **Вивчити особливості розробки Telegram-ботів**, включаючи роботу з Telegram Bot API, асинхронне програмування, управління станами (FSM), створення інтерактивних інтерфейсів, а також специфіку інтеграції з API криптовалютних бірж, зокрема обробку запитів, відповідей та помилок, забезпечення безпеки API-ключів.
5. **Сформулювати детальні функціональні та нефункціональні вимоги** до розроблюваного Telegram-бота на основі проведеного аналізу та досліджень. Вимоги повинні охоплювати:
    - Управління API-ключами користувача.
    - Перегляд інформації про ринок (ціни, обрані монети).
    - Розміщення ринкових та лімітних ордерів (LONG/SHORT) з можливістю вказання суми, ціни, кредитного плеча.
    - Перегляд та скасування відкритих ордерів.
    - Перегляд та закриття відкритих позицій.

- Налаштування та автоматичне застосування параметрів Take Profit та Stop Loss.
- Налаштування та функціонування модуля відстеження Telegram-каналів (вибір каналу-джерела, параметри ETA, відстежувані монети, мінімальне співвідношення, кредитне плече, розмір позиції).
- Зберігання налаштувань користувача в локальній базі даних SQLite.
- Надійність, безпеку та зручність використання.

6. **Розробити архітектуру програмного забезпечення** Telegram-бота, визначити його основні модулі (наприклад, обробники команд, клієнт для взаємодії з Binance, модуль роботи з базою даних, модуль трекера каналів) та взаємозв'язки між ними.
7. **Реалізувати програмний код** Telegram-бота мовою програмування Python з використанням фреймворку Aiogram для взаємодії з Telegram, бібліотеки CCXT для роботи з API Binance та модуля aiosqlite для асинхронної роботи з базою даних SQLite, відповідно до розробленої архітектури та сформульованих вимог[1][10][13]. Це включає:
    - Створення структури бази даних.
    - Розробку обробників команд для всіх заявлених функцій.
    - Реалізацію логіки взаємодії з Binance API.
    - Розробку класу ChannelTracker та логіки його роботи.
    - Реалізацію механізму відновлення активних трекерів при перезапуску бота.
8. **Провести тестування та валідацію** розробленого Telegram-бота з метою перевірки його функціональності, коректності роботи з API Binance Futures Testnet, ефективності обробки сигналів, надійності зберігання даних та зручності користувацького інтерфейсу.
9. **Оцінити результати розробки**, проаналізувати їх відповідність поставленим цілям та вимогам, визначити можливі обмеження та напрямки для подальшого вдосконалення програмного продукту.

Виконання цих завдань дозволить реалізувати поставлену мету та створити функціональний Telegram-бот для взаємодії з криптовалютною біржею Binance, який надасть користувачам зручний інструмент для торгівлі та автоматизації деяких торгових процесів, а також матиме потенціал для подальшого розвитку та вдосконалення.

# 2 ДОСЛІДЖЕННЯ ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ІНСТРУМЕНТІВ ДЛЯ ВЗАЄМОДІЇ З КРИПТОВАЛЮТНИМИ БІРЖАМИ

## 2.1 Класифікація програмних засобів для торгівлі криптовалютами та взаємодії з біржами

Програмні засоби, призначені для торгівлі криптовалютами та взаємодії з біржами, можна класифікувати за різними критеріями, враховуючи їх архітектуру, основний функціонал, рівень автоматизації, цільову аудиторію та технологічний стек. Нижче наведено класифікацію таких програмних засобів на основі ключових ознак, що є релевантними для розуміння контексту розроблюваного Telegram-бота.

**За типом платформи та інтерфейсу користувача:**

1. ***Веб-платформи:*** Комплексні онлайн-сервіси, доступні через браузер, що надають широкий набір інструментів для аналізу ринку, ручної торгівлі та налаштування автоматизованих стратегій (наприклад, 3Commas, TradingView з функцією підключення до брокерів/бірж). Вони часто пропонують розширені графічні інструменти та аналітику.
2. ***Десктопні додатки:*** Програми, що встановлюються на комп'ютер користувача. Можуть забезпечувати вищу швидкість обробки даних та більш глибоку інтеграцію з операційною системою, але вимагають встановлення та оновлення (наприклад, термінали типу MetaTrader з кастомними скриптами для криптобірж, спеціалізовані торгові термінали від деяких бірж).
3. ***Мобільні додатки:*** Застосунки для смартфонів та планшетів, що надають доступ до торгівлі "на ходу". Зазвичай мають дещо спрощений функціонал порівняно з веб- або десктопними версіями, але забезпечують високу мобільність (наприклад, офіційні мобільні додатки бірж Binance, KuCoin; мобільні версії платформ типу 3Commas).

4. ***Telegram-боти (як розроблюваний проєкт):*** Програмні агенти, що функціонують всередині месенджера Telegram. Забезпечують швидкий доступ до основних торгових функцій через текстові команди та інтерактивні клавіатури. Їх перевагами є кросплатформність (Telegram доступний на багатьох ОС), звичний для користувачів інтерфейс та можливість отримання миттєвих сповіщень.

**За рівнем автоматизації та типом реалізованих стратегій:**

1. ***Інструменти для ручного трейдингу:*** Надають інтерфейс для розміщення ордерів, перегляду графіків та управління акаунтом, але не виконують торгові операції автоматично (наприклад, базовий функціонал веб-інтерфейсів бірж).
2. ***Напівавтоматизовані інструменти:*** Дозволяють встановлювати умовні ордери (наприклад, Take Profit, Stop Loss, трейлінг-стопи), які виконуються автоматично при досягненні певних умов, але ініціація позиції залишається за користувачем. Розроблюваний бот частково належить до цієї категорії через функціонал TP/SL.
3. ***Повністю автоматизовані торгові боти (алгоритмічний трейдинг):*** Програми, що самостійно приймають торгові рішення та виконують операції на основі заздалегідь визначених алгоритмів або моделей машинного навчання.
    - *Боти на основі технічного аналізу:* Використовують індикатори (ковзні середні, RSI, MACD тощо) для генерації сигналів на вхід/вихід.
    - *Арбітражні боти:* Шукають розбіжності в цінах на один і той самий актив на різних біржах або в різних парах.
    - *GRID-боти:* Розміщують сітку лімітних ордерів на купівлю та продаж навколо поточної ціни.
    - *DCA-боти (Dollar-Cost Averaging):* Регулярно купують актив на фіксовану суму, незалежно від його поточної ціни.

- *Боти для копіювання сигналів:* Автоматично виконують угоди на основі сигналів з зовнішніх джерел (наприклад, Telegram-каналів, як у розроблюваному проєкті).

**За способом доступу до біржі:**

1. ***Пряма інтеграція з API біржі:*** Більшість торгових ботів та платформ взаємодіють з біржами через їх офіційні API (Application Programming Interfaces). Це дозволяє програмно розміщувати ордери, отримувати ринкові дані, керувати акаунтом. Розроблюваний бот використовує цей підхід через бібліотеку CCXT[1].
2. ***Використання проміжних платформ/брокерів:*** Деякі інструменти можуть працювати через посередників, які агрегують доступ до кількох бірж.

**За моделлю розповсюдження та оплати:**

1. ***Комерційні платформи з підпискою:*** Багато просунутих платформ (3Commas, Cryptohopper) працюють за моделлю щомісячної або щорічної плати за доступ до їх функціоналу.
2. ***Продукти з одноразовою оплатою:*** Деякі десктопні програми або скрипти можуть продаватися за фіксованою ціною.
3. ***Безкоштовні інструменти з відкритим вихідним кодом (Open Source):*** Існують проєкти, код яких є загальнодоступним, що дозволяє користувачам модифікувати та використовувати їх безкоштовно (наприклад, Freqtrade, Zenbot). Розроблюваний бот, будучи дипломним проєктом, потенційно може бути віднесений до цієї категорії, якщо його код буде опубліковано.
4. ***Інструменти, що заробляють на комісіях:*** Деякі платформи (наприклад, Pionex) надають ботів безкоштовно, але заробляють на торгових комісіях, що генеруються користувачами.

**За цільовою аудиторією:**

1. ***Для початківців:*** Інструменти з простим інтерфейсом, готовими шаблонами стратегій та мінімальними налаштуваннями.

2. ***Для досвідчених трейдерів:*** Платформи з розширеними можливостями аналізу, глибокою кастомізацією стратегій, підтримкою написання власних скриптів.
3. ***Для розробників:*** Бібліотеки та фреймворки (як CCXT, Aiogram), що надають інструментарій для створення власних торгових рішень[1][2].

Розроблюваний в рамках даного дипломного проєкту Telegram-бот можна класифікувати як:

- **За типом платформи:** Telegram-бот.
- **За рівнем автоматизації:** Напівавтоматизований (TP/SL, ручне розміщення ордерів) з елементами повністю автоматизованого (відстеження сигналів з каналів).
- **За способом доступу до біржі:** Пряма інтеграція з API Binance через бібліотеку CCXT.
- **За моделлю розповсюдження:** Потенційно безкоштовний інструмент (як дипломний проєкт).
- **За цільовою аудиторією:** Користувачі Telegram, які шукають зручний мобільний інструмент для взаємодії з Binance, включаючи як ручне управління, так і можливість автоматизації на основі сигналів.

Така класифікація допомагає краще зрозуміти місце розроблюваного програмного продукту на ринку існуючих рішень та визначити його ключові характеристики та потенційні переваги.

## 2.2 Види програмних інструментів для взаємодії з криптовалютними біржами

На основі класифікації, наведеної у попередньому підрозділі, розглянемо детальніше основні види програмних інструментів, що використовуються трейдерами та інвесторами для взаємодії з криптовалютними біржами, такими як Binance. Кожен вид має свої особливості, переваги та недоліки, що визначають його придатність для різних завдань та категорій користувачів.

1. **Торгові термінали та платформи з розширеним функціоналом:** Ця категорія включає веб-платформи та десктопні додатки, що надають значно ширші можливості, ніж стандартний інтерфейс біржі.

   ***Приклади:*** TradingView (з функцією підключення до брокерів/бірж, див. рис. 3), 3Commas, Cryptohopper, ATAS, QuanTower.

   ***Ключові характеристики:***

   - *Глибокий технічний аналіз:* Надання великої кількості індикаторів, інструментів для малювання на графіках, можливість створення власних індикаторів.
   - *Управління ордерами:* Розширені типи ордерів (наприклад, OCO – One-Cancels-the-Other, айсберг-ордери), швидке розміщення та модифікація.
   - *Сканер ринку:* Інструменти для пошуку активів за певними критеріями (наприклад, зростання обсягу, пробиття рівнів).
   - *Автоматизація (часткова або повна):* Можливість налаштування торгових ботів або виконання умовних ордерів.
   - *Інтеграція з кількома біржами:* Деякі платформи дозволяють торгувати на кількох біржах з одного інтерфейсу.

   ***Цільова аудиторія:*** Професійні та досвідчені трейдери, яким потрібні потужні аналітичні інструменти та гнучкість в управлінні угодами.

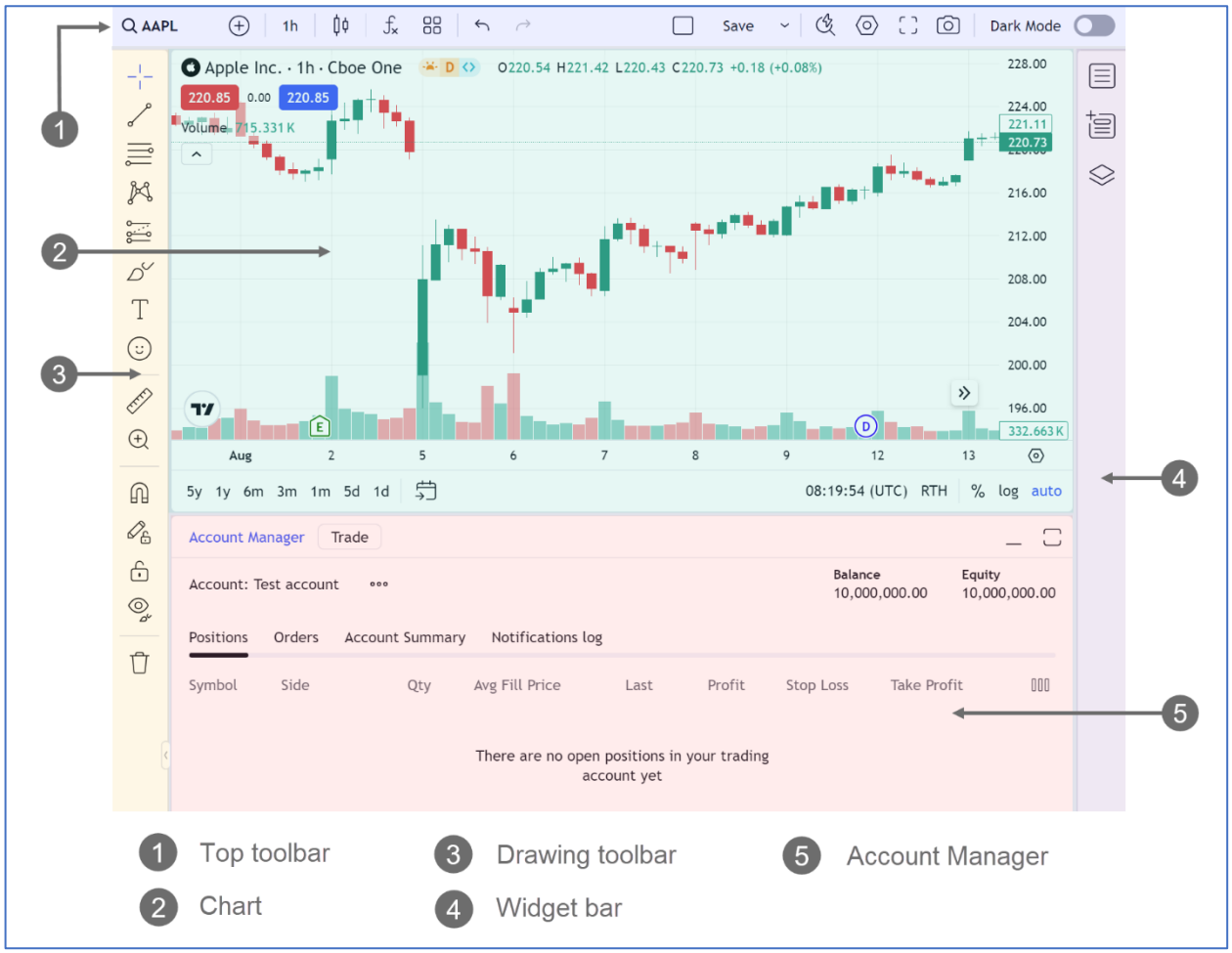



Рисунок *3 — Приклад інтерфейсу платформи TradingView з підключеним торговим акаунтом*

2. **Мобільні додатки криптовалютних бірж та спеціалізовані торгові додатки:**

Офіційні додатки бірж та сторонні мобільні платформи, оптимізовані для торгівлі зі смартфонів та планшетів.

***Приклади:*** Binance App, KuCoin App, мобільні версії 3Commas, TabTrader.

***Ключові характеристики:***

- *Мобільність:* Доступ до торгівлі та управління акаунтом з будь-якого місця.
- *Основний торговий функціонал:* Розміщення ринкових та лімітних ордерів, перегляд балансу, історії угод.
- *Сповіщення:* Push-сповіщення про виконання ордерів, зміни цін.

- *Спрощений інтерфейс:* Зазвичай більш лаконічний та адаптований під невеликі екрани.

***Цільова аудиторія:*** Трейдери, яким важлива мобільність та можливість швидко реагувати на ринкові зміни; користувачі, які віддають перевагу торгівлі зі смартфона.

3. **Бібліотеки та фреймворки для алгоритмічної торгівлі:** Програмні інструменти для розробників, що дозволяють створювати власні торгові боти та скрипти.
   - ***Приклади*:** CCXT (Python, JavaScript, PHP), python-binance (Python), Freqtrade (Python, фреймворк для створення ботів), Zenbot (JavaScript, open-source бот).
   - ***Ключові характеристики:***
     - *Прямий доступ до API бірж:* Надання функцій для надсилання запитів до API (розміщення ордерів, отримання ринкових даних, управління акаунтом).
     - *Гнучкість:* Дозволяють реалізувати будь-яку торгову логіку та стратегію.
     - *Можливість інтеграції:* Легко інтегруються з іншими бібліотеками для аналізу даних, машинного навчання, візуалізації.
     - *Вимагають навичок програмування.*
   - ***Цільова аудиторія:*** Розробники програмного забезпечення, трейдери з навичками програмування, які хочуть створити повністю кастомізовані торгові рішення. Розроблюваний в рамках дипломного проєкту Telegram-бот використовує бібліотеку CCXT.
4. **Telegram-боти для торгівлі та сповіщень (включаючи розроблюваний проєкт):**

   Інструменти, що працюють всередині месенджера Telegram та надають користувачам інтерфейс для взаємодії з біржами.

   ***Приклади:*** Cornix, Crypto Whale Bot, кастомні боти, **розроблюваний бот**.

   ***Ключові характеристики:***

- *Зручність та доступність:* Використання звичного інтерфейсу Telegram, доступність на всіх пристроях, де встановлений Telegram.
- *Сповіщення в реальному часі:* Можливість отримувати миттєві повідомлення про виконання ордерів, спрацювання TP/SL, нові сигнали.
- *Автоматизація на основі сигналів:* Деякі боти (включаючи розроблюваний) можуть відстежувати Telegram-канали та автоматично виконувати угоди на основі отриманих сигналів.
- *Основний торговий функціонал:* Розміщення ордерів, перегляд позицій, налаштування TP/SL.
- *Простота використання:* Зазвичай орієнтовані на швидке виконання операцій без необхідності заглиблюватися в складні налаштування.

**Цільова аудиторія:** Трейдери, які цінують мобільність, швидкість реакції, можливість автоматизації на основі сигналів та зручність отримання сповіщень.

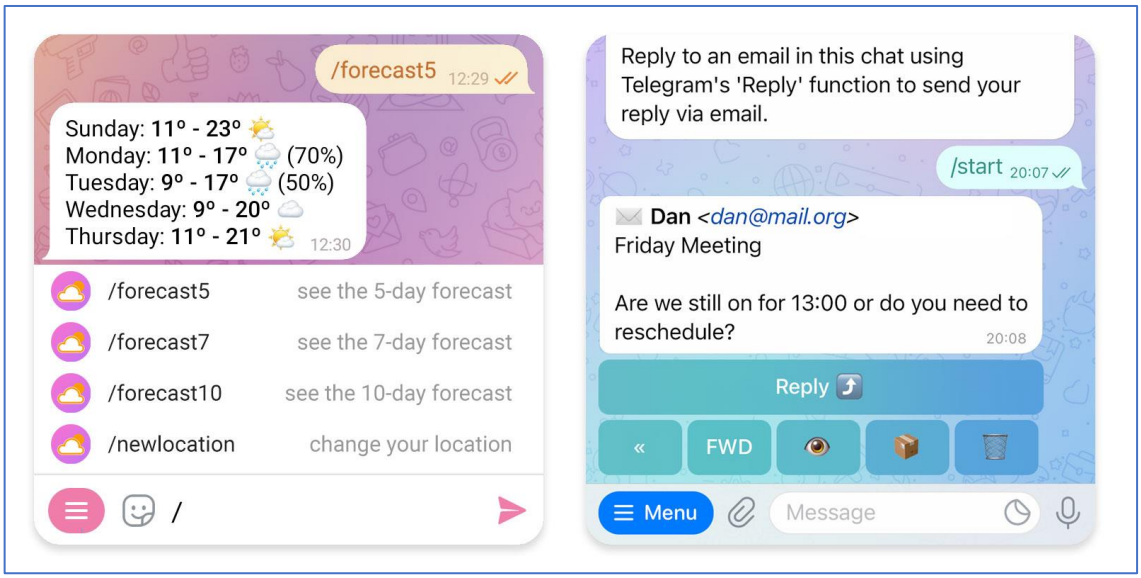



Рисунок *4 — Приклад інтерфейсу Telegram-бота*

5. **Платформи для копіювання угод (Copy Trading / Social Trading):** Сервіси, що дозволяють користувачам автоматично копіювати угоди успішних трейдерів.

***Приклади:*** Вбудовані функції на деяких біржах (наприклад, Binance Copy Trading, eToro), спеціалізовані платформи.

***Ключові характеристики:***

- *Пасивний дохід:* Можливість заробляти, копіюючи досвідчених трейдерів.
- *Вибір трейдерів:* Надання статистики та рейтингів трейдерів для вибору.
- *Управління ризиками:* Можливість встановлювати ліміти на копіювання.

***Цільова аудиторія:*** Початківці, які хочуть вчитися у досвідчених трейдерів; інвестори, які шукають пасивні способи заробітку на криптовалютах.

Кожен з цих видів програмних інструментів має свою нішу та задовольняє певні потреби користувачів. Розроблюваний Telegram-бот поєднує в собі елементи інструменту для ручного мобільного трейдингу (через інтерфейс Telegram) та автоматизованого інструменту (через відстеження сигналів), що робить його потенційно привабливим для широкого кола користувачів, які шукають баланс між контролем та автоматизацією.

## 2.3 Огляд методів та інструментів розробки Telegram-бота для взаємодії з криптовалютною біржею

При розробці Telegram-бота, призначеного для взаємодії з криптовалютними біржами, як-от Binance, розробники використовують різноманітні методи, мови програмування, фреймворки та бібліотеки. Вибір конкретних інструментів залежить від вимог до функціоналу, продуктивності, масштабованості та досвіду команди розробників. Розглянемо ключові аспекти та технології, релевантні для створення такого програмного продукту.

**Вибір мови програмування**

***Python (використовується у даному проєкті):*** Python є однією з найпопулярніших мов програмування для розробки Telegram-

ботів та роботи з API криптовалютних бірж. Його популярність зумовлена низкою переваг:

***Простота синтаксису та швидкість розробки:*** Python відомий своїм чистим та легко читаним синтаксисом, що дозволяє швидше писати код та зменшує ймовірність помилок. Це особливо важливо для швидкого прототипування та ітеративної розробки (див. рис. 5).

***Велика кількість бібліотек:*** Існує величезна екосистема бібліотек для Python, що значно спрощує розробку. Для даного проєкту ключовими є:

*aiogram:* Асинхронний фреймворк для створення Telegram-ботів, що забезпечує зручну роботу з Telegram Bot API, управління станами (FSM), створення клавіатур тощо.

*ccxt:* Універсальна бібліотека для взаємодії з API понад 100 криптовалютних бірж, включаючи Binance. Вона абстрагує відмінності між API різних бірж, надаючи єдиний інтерфейс.

*aiosqlite:* Асинхронна бібліотека для роботи з SQLite, що дозволяє ефективно взаємодіяти з базою даних в асинхронному середовищі asyncio.

***Підтримка асинхронного програмування:*** Завдяки asyncio та сумісним бібліотекам (aiogram, aiohttp, aiosqlite), Python дозволяє створювати високопродуктивні боти, здатні обробляти велику кількість одночасних запитів без блокування, що критично для взаємодії з Telegram та API бірж у реальному часі.

***Велика спільнота та доступність ресурсів:*** Велика спільнота розробників Python забезпечує наявність численних навчальних матеріалів, форумів та готових рішень.

***Придатність для аналізу даних та машинного навчання:*** Хоча це не є основним фокусом даного проєкту, Python є лідером у галузі аналізу даних та ШІ, що відкриває перспективи для подальшого розвитку бота, наприклад, інтеграції предиктивних моделей.




Рисунок *5 — Логотип мови програмування Python*

**Інші мови програмування:**

Хоча Python є дуже популярним вибором, для розробки торгових ботів також можуть використовуватися інші мови:

***JavaScript (Node.js):*** Також популярний завдяки своїй асинхронній природі (особливо з Node.js) та наявності бібліотек для роботи з Telegram API (наприклад, Telegraf.js) та API бірж (наприклад, ccxt також доступний для JavaScript).

***Go:*** Відомий своєю високою продуктивністю та ефективною роботою з паралелізмом, що може бути перевагою для високочастотних торгових ботів.

***Java/Kotlin:*** Традиційно сильні у корпоративній розробці, можуть використовуватися для створення надійних та масштабованих бекенд-систем для торгових ботів, хоча для швидкої розробки Telegram-інтерфейсів можуть бути менш гнучкими, ніж Python або JavaScript.

**Фреймворки для розробки Telegram-ботів**

***Aiogram (використовується у даному проєкті):***

Сучасний, потужний та повністю асинхронний фреймворк для створення Telegram-ботів на Python 3.7+ [2] (див. рис. 6).

**Ключові переваги:**

- *Асинхронність "з коробки":* Побудований на asyncio, що забезпечує високу продуктивність.

- *Простий та гнучкий API:* Зручний для роботи з усіма можливостями Telegram Bot API.
- *Finite State Machine (FSM):* Вбудована система управління станами для створення складних діалогів з користувачем, що активно використовується в даному проєкті для покрокового введення даних (наприклад, при створенні ордера або налаштуванні трекера).
- *Фільтри (Filters):* Потужна система фільтрації вхідних повідомлень та колбеків.
- *Мідлвари (Middlewares):* Дозволяють додавати власну логіку обробки запитів на різних етапах.




Рисунок *6 — Логотип фреймворку Aiogram*

**Інші фреймворки для Python:**

*python-telegram-bot:* Один з найстаріших та найпопулярніших фреймворків, також підтримує асинхронність, має велику спільноту.

*Telethon:* Більше орієнтований на роботу з Telegram API як клієнт (тобто, автоматизація дій від імені користувача), але може використовуватися і для створення ботів.

**Бібліотеки для взаємодії з API криптовалютних бірж:**

***CCXT (CryptoCurrency eXchange Trading Library):***

Бібліотека, що надає єдиний інтерфейс для доступу до API багатьох криптовалютних бірж [2] (див. рис. 7).

**Ключові переваги:**

- *Уніфікація:* Дозволяє писати код, який може працювати з різними біржами з мінімальними змінами.
- *Підтримка великої кількості бірж:* Включаючи Binance, Kraken, Coinbase Pro, KuCoin та багато інших.
- *Реалізація публічних та приватних методів API:* Дозволяє отримувати ринкові дані (тікери, книги ордерів, свічки) та виконувати торгові операції (розміщення/скасування ордерів, перевірка балансу).
- *Обробка помилок та лімітів запитів:* Має вбудовані механізми для обробки специфічних помилок бірж та дотримання лімітів API.
- *Активна розробка та підтримка спільнотою.*



Рисунок *7 — Логотип бібліотеки CCXT*

**Спеціалізовані бібліотеки для конкретних бірж:**

Наприклад, *python-binance* для роботи виключно з API Binance. Можуть надавати доступ до деяких дуже специфічних функцій біржі, які ще не реалізовані в універсальних бібліотеках, але обмежують портативність коду.

**Робота з базами даних**

***SQLite*:**

Легковажна, файлова, реляційна СУБД (див. рис. 8).

**Ключові переваги для даного проєкту:**

- **Простота налаштування та використання:** Не потребує окремого сервера бази даних, дані зберігаються в одному файлі.

- **Достатня продуктивність для потреб бота:** Для зберігання налаштувань користувачів, API-ключів, параметрів трекера каналів тощо, продуктивності SQLite цілком достатньо.
- **Асинхронна підтримка через aiosqlite:** Дозволяє не блокувати основний потік виконання бота при операціях з базою даних.




Рис*унок 8 — Логотип SQLite*

**Інші СУБД:**

*PostgreSQL, MySQL:* Більш потужні реляційні СУБД, які можуть знадобитися для проєктів з великим навантаженням, складними запитами або потребою в розширених функціях (реплікація, кластеризація).

*MongoDB, Redis:* NoSQL бази даних, які можуть бути корисними для зберігання неструктурованих даних або для кешування.

**Архітектурні підходи**

Хоча для Telegram-ботів не завжди застосовуються строгі архітектурні патерни, як MVVM або MVP, що характерні для мобільних чи веб-додатків з GUI, деякі принципи залишаються важливими:

*Розділення відповідальностей (Separation of Concerns):* Логіка обробки команд, взаємодії з API біржі, роботи з базою даних та бізнес-логіка трекера каналів винесені в окремі модулі(файли handlers/*.py, binance_client.py, db.py, клас ChannelTracker в channel_tracker.py), що покращує читабельність, супровід та тестування коду.

*Обробники (Handlers) та маршрутизація (Routing):* Фреймворки типу Aiogram використовують систему обробників, які реагують на певні типи

повідомлень, команд або колбеків. Це дозволяє структурувати логіку взаємодії з користувачем.

*Асинхронна архітектура:* Весь проєкт побудований на принципах асинхронного програмування для забезпечення ефективної обробки I/O операцій (мережа, база даних).

**Тестування та налагодження**

*Модульне тестування:* Перевірка окремих функцій та методів (наприклад, функцій для розрахунку TP/SL, парсингу сигналів).

*Інтеграційне тестування:* Перевірка взаємодії між компонентами бота (наприклад, обробник команди -> клієнт Binance -> база даних).

*Тестування на Testnet:* Використання тестової мережі Binance Futures Testnet (як у даному проєкті) є критично важливим для безпечного тестування торгової логіки без ризику втрати реальних коштів.

*Логування:* Детальне логування всіх важливих подій, помилок, запитів та відповідей API допомагає в налагодженні та моніторингу роботи бота.

**Оптимізація продуктивності**

*Асинхронні операції:* Максимальне використання async/await для всіх операцій, що можуть блокувати (робота з мережею, файловою системою, базою даних).

*Ефективні запити до API:* Мінімізація кількості запитів до API Telegram та біржі, використання пакетних запитів, де це можливо.

*Оптимізація роботи з базою даних:* Використання індексів, оптимізація запитів.

Вибір та правильне застосування цих методів та інструментів дозволяють створювати надійні, функціональні та ефективні Telegram-боти для взаємодії з криптовалютними біржами, задовольняючи потреби сучасних трейдерів. Даний дипломний проєкт базується на використанні Python, Aiogram, CCXT та aiosqlite, що є сучасним та ефективним стеком для вирішення поставлених завдань.

## 2.4 Засоби розробки Telegram-бота

Під час розробки Telegram-бота для взаємодії з криптовалютною біржею Binance, як і будь-якого іншого програмного продукту на Python, розробники використовують набір специфічних інструментів, бібліотек та середовищ, які допомагають ефективно створювати, тестувати та розгортати додаток. Розглянемо ключові засоби, що були використані або є релевантними для даного дипломного проєкту.

1. **Інтегроване середовище розробки (IDE) та редактори коду:**

   *Visual Studio Code (VS Code):* Популярний, безкоштовний та багатофункціональний редактор коду з відмінною підтримкою Python. Надає такі можливості, як підсвічування синтаксису, автодоповнення коду (IntelliSense), інтегрований термінал, налагоджувач, система контролю версій (Git). Велика кількість розширень дозволяє налаштувати середовище під конкретні потреби проєкту.

   *PyCharm:* Спеціалізоване IDE для Python від JetBrains. Пропонує глибокий аналіз коду, потужні інструменти для рефакторингу, інтеграцію з фреймворками та інструментами тестування. Існує як безкоштовна (Community), так і платна (Professional) версія.

   Інші редактори, такі як Sublime Text, Atom, також можуть використовуватися залежно від вподобань розробника.

2. **Мова програмування Python та її стандартна бібліотека:**

   Як зазначалося раніше, *Python* є основною мовою розробки даного проєкту.

   *Стандартна бібліотека Python:* Надає широкий набір модулів для виконання різноманітних завдань, включаючи роботу з рядками, файловою системою, мережею, датою/часом, а також модуль asyncio для асинхронного програмування, який є фундаментом для aiogram. Модуль sqlite3 (хоча в проєкті використовується його асинхронний аналог aiosqlite) також є частиною стандартної бібліотеки для роботи з SQLite.

3. **Ключові сторонні бібліотеки та фреймворки (використані у проєкті):**

   *aiogram (версія 3.4.1):* Асинхронний фреймворк для створення Telegram-ботів. Забезпечує всю необхідну інфраструктуру для обробки повідомлень, команд, колбеків, управління станами (FSM), створення клавіатур та взаємодії з Telegram Bot API.

   *ccxt (версія 4.1.22):* Бібліотека для взаємодії з API криптовалютних бірж. Використовується для підключення до Binance Futures Testnet, отримання ринкових даних, розміщення та управління ордерами й позиціями.

   *aiosqlite (версія 0.19.0):* Асинхронний драйвер для роботи з базою даних SQLite. Дозволяє виконувати операції з базою даних (створення таблиць, вставка, оновлення, вибірка даних) в асинхронному режимі, не блокуючи основний потік виконання бота.

4. **Система управління віртуальними середовищами та залежностями:**

   *venv (стандартний модуль Python):* Використовується для створення ізольованих віртуальних середовищ Python. Це дозволяє мати окремий набір бібліотек та їх версій для кожного проєкту, уникаючи конфліктів між ними.

   *pip (менеджер пакетів Python):* Використовується для встановлення та управління залежностями проєкту (бібліотеками), які перераховані у файлі requirements.txt. Файл requirements.txt у даному проєкті містить aiogram==3.4.1, ccxt==4.1.22, aiosqlite==0.19.0.

5. **Система контролю версій:**

   *Git:* Найпопулярніша розподілена система контролю версій. Дозволяє відстежувати зміни у коді, повертатися до попередніх версій, працювати над проєктом у команді (хоча даний проєкт є індивідуальним, використання Git є хорошою практикою).

   *Платформи для хостингу Git-репозиторіїв (наприклад, GitHub (див. рис. 9), GitLab, Bitbucket):* Використовуються для зберігання коду проєкту в хмарі, спільної роботи та резервного копіювання.




Рисунок *9 — Логотип GitHub*

6. **Інструменти для тестування та налагодження:**

*Вбудований налагоджувач IDE (VS Code, PyCharm):* Дозволяє покроково виконувати код, перевіряти значення змінних, встановлювати точки зупину.

*Модуль logging (стандартна бібліотека Python):* Використовується для запису інформації про роботу бота, помилок та важливих подій. У проєкті налаштоване базове логування, особливо в channel_tracker.py та main.py.

*Binance Futures Testnet:* Спеціальне середовище від Binance, що дозволяє тестувати торгові стратегії та роботу з API без використання реальних коштів. Це є ключовим інструментом для безпечного тестування функціоналу даного бота.

7. **Операційна система та середовище виконання:**

Проєкт може бути розроблений та запущений на різних операційних системах (Windows, macOS, Linux), де встановлений Python. У файлі main.py є специфічна перевірка sys.platform.startswith("win") для встановлення політики циклу подій asyncio для Windows (asyncio.WindowsSelectorEventLoopPolicy()), що демонструє врахування особливостей ОС.

Використання цього набору засобів та технологій дозволяє ефективно розробляти, тестувати та підтримувати Telegram-бота для взаємодії з криптовалютною біржею. Обраний стек (Python, aiogram, ccxt, aiosqlite) є сучасним, добре документованим та має активні спільноти, що сприяє успішній реалізації проєкту.

## 2.5 Ключові фреймворки та бібліотеки, використані в проєкті

У цьому розділі детально розглядаються основні фреймворки та бібліотеки, які були обрані та використані під час розробки Telegram-бота для взаємодії з криптовалютною біржею Binance. Ці інструменти надають необхідний функціонал для реалізації асинхронної взаємодії з Telegram Bot API, підключення до API біржі, роботи з базою даних та організації структури проєкту.

**aiogram (версія 3.4.1) – Асинхронний фреймворк для Telegram-ботів:** aiogram є ключовим компонентом даного проєкту, оскільки він забезпечує всю необхідну інфраструктуру для створення Telegram-бота на мові Python.

***Призначення та роль у проєкті:***

*Взаємодія з Telegram Bot API:* aiogram обробляє HTTP-запити до Telegram API для надсилання повідомлень, клавіатур, обробки команд та колбеків від користувача.

*Асинхронна модель:* Побудований на основі бібліотеки asyncio, aiogram дозволяє боту ефективно обробляти велику кількість одночасних оновлень від Telegram та виконувати асинхронні операції (наприклад, запити до API біржі або бази даних) без блокування основного потоку. Це критично важливо для забезпечення швидкої реакції бота.

*Обробники (Handlers) та маршрутизація (Dispatching):* Надає гнучку систему для реєстрації обробників для різних типів подій (повідомлення, команди, натискання на inline-кнопки, зміни стану FSM). У проєкті це реалізовано через роутери (Router) в кожному файлі з обробниками

(main_menu.py, market.py, orders.py тощо), які потім включаються до головного диспетчера (Dispatcher).

*Finite State Machine (FSM):* Вбудований механізм управління станами дозволяє створювати складні, багатоетапні діалоги з користувачем. Це активно використовується для покрокового введення даних при створенні ордерів, налаштуванні API-ключів, параметрів TP/SL та налаштувань трекера каналів (клас TrackerSettings та стани waiting_for_... у channel_tracker.py).

*Клавіатури та кнопки:* aiogram спрощує створення різних типів клавіатур (ReplyKeyboardMarkup, InlineKeyboardMarkup) та кнопок (KeyboardButton, InlineKeyboardButton), які є основним засобом навігації та взаємодії в боті.

*Фільтри:* Дозволяють точно визначати, які обробники повинні реагувати на певні повідомлення або події, наприклад, фільтрація за текстом повідомлення (lambda m: m.text == "..."), станом FSM (StateFilter) або типом чату (F.chat.type.in_({"channel", "supergroup"})).

***Переваги використання у проєкті:*** Висока продуктивність завдяки асинхронності, зручний API, потужні інструменти для управління діалогами та маршрутизації, активна спільнота та хороша документація.

**ccxt (CryptoCurrency eXchange Trading Library, версія 4.1.22) – Бібліотека для взаємодії з API криптовалютних бірж:**

ccxt є фундаментальною бібліотекою для реалізації функціоналу взаємодії з біржею Binance у даному проєкті.

***Призначення та роль у проєкті:***

*Уніфікований доступ до API Binance:* Надає стандартизований набір методів для виконання операцій на біржі Binance Futures Testnet, таких як отримання інформації про символи (get_symbol_info), поточних цін (get_price), розміщення ринкових (create_market_buy, create_market_sell_order) та лімітних ордерів (create_limit_order), встановлення кредитного плеча (set_leverage), управління маржею (set_margin_mode), отримання відкритих ордерів (get_open_orders) та позицій (get_position), скасування ордерів (cancel_order), а також встановлення TP/SL для позицій (set_tp_sl_for_position).

*Обробка автентифікації:* Спрощує процес автентифікації на біржі за допомогою API-ключів користувача.

*Нормалізація даних:* ccxt намагається нормалізувати формати даних, що повертаються різними біржами, до єдиного вигляду, що полегшує їх обробку.

*Підтримка асинхронних операцій (опосередковано):* Хоча сама бібліотека ccxt є синхронною, її методи у даному проєкті викликаються всередині асинхронних функцій asyncio.to_thread (як видно у channel_tracker.py при взаємодії з біржею в методі open_position або у positions.py при закритті позиції), щоб не блокувати асинхронний цикл подій aiogram. У файлі binance_client.py методи також визначені як async, що передбачає їх використання в асинхронному контексті, а синхронні виклики ccxt всередині них будуть виконуватися в тому ж потоці, якщо не обгорнуті в asyncio.to_thread. *Для повної асинхронності з ccxt зазвичай використовують ccxt.pro або обгортки.*

***Переваги використання у проєкті:*** Широка підтримка функціоналу Binance, відносна простота використання, активна підтримка, можливість потенційного розширення на інші біржі в майбутньому з мінімальними змінами коду завдяки уніфікованому API.

**aiosqlite (версія 0.19.0) – Асинхронна бібліотека для роботи з SQLite:**

aiosqlite дозволяє виконувати операції з базою даних SQLite в асинхронному режимі, що є важливим для інтеграції з асинхронною архітектурою бота на aiogram.

***Призначення та роль у проєкті:***

*Зберігання даних користувачів:* У файлі db.py визначено функції для створення таблиць та управління даними користувачів (users), їх API-ключів, обраних монет (favorites), збережених ордерів (хоча ця таблиця менш активно використовується для актуальних ордерів, які беруться з біржі), налаштувань TP/SL (tp_sl_settings), налаштувань трекера каналів (channel_tracker_settings) та відстежуваних монет для трекера (tracked_coins).

*Асинхронні операції з БД:* Всі функції в db.py є асинхронними (async def) і використовують await для виконання SQL-запитів, що запобігає блокуванню бота під час очікування відповіді від бази даних.

*Простота інтеграції:* SQLite не потребує окремого сервера бази даних, що спрощує розгортання та налаштування бота.

***Переваги використання у проєкті:*** Асинхронність, легкість використання для зберігання структурованих даних, відсутність необхідності у зовнішньому сервері БД, що ідеально підходить для такого типу проєкту.

Використання саме цих фреймворків та бібліотек (aiogram, ccxt, aiosqlite) формує надійний та ефективний технологічний стек для розробки Telegram-бота. aiogram забезпечує потужну та гнучку взаємодію з користувачем через Telegram, ccxt надає стандартизований доступ до функціоналу криптовалютної біржі, а aiosqlite дозволяє асинхронно та зручно зберігати необхідні дані. Такий вибір сприяє швидкості розробки, підтримці асинхронної моделі роботи та забезпечує реалізацію всього запланованого функціоналу.

# 3 РОЗРОБКА TELEGRAM-БОТА ДЛЯ ВЗАЄМОДІЇ З КРИПТОВАЛЮТНОЮ БІРЖЕЮ BINANCE

## 3.1 Опис предметної області

Предметною областю даної дипломної роботи є розробка Telegram-бота, призначеного для полегшення та часткової автоматизації процесу торгівлі криптовалютними ф'ючерсами на тестовій платформі біржі Binance (Binance Futures Testnet). У сучасному фінансовому світі криптовалюти стали значущим класом активів, а торгівля ними приваблює все більшу кількість як індивідуальних, так і інституційних інвесторів. Висока волатильність та цілодобовий режим роботи криптовалютних ринків створюють попит на інструменти, що дозволяють оперативно реагувати на ринкові зміни, ефективно управляти ордерами та позиціями, а також автоматизувати рутинні торгові операції.

Telegram, будучи однією з найпопулярніших платформ для обміну повідомленнями, надає потужний API для розробки ботів (Telegram Bot API). Це дозволяє створювати інтерактивні додатки, доступні користувачам безпосередньо через звичний інтерфейс месенджера на різних пристроях. Telegram-боти для торгівлі криптовалютами пропонують зручний спосіб моніторингу ринку, управління торговим рахунком та виконання операцій без необхідності постійного доступу до веб-інтерфейсу біржі або спеціалізованих десктопних програм.

У даному дипломному проєкті фокус робиться на взаємодії з криптовалютною біржею **Binance**, яка є однією з найбільших та найпопулярніших у світі за обсягами торгів та кількістю користувачів. Конкретно, проєкт орієнтований на роботу з **Binance Futures Testnet**, що дозволяє безпечно тестувати торгові стратегії та функціонал бота без ризику втрати реальних коштів. Розробка такого бота вимагає інтеграції з API Binance для отримання ринкових даних, розміщення ордерів, управління позиціями та балансом, а також реалізації логіки взаємодії з користувачем через Telegram.

Ключовими аспектами предметної області, які враховуються в проєкті, є:

- **Торгові операції:** Можливість розміщення ринкових (market) та лімітних (limit) ордерів на купівлю (LONG) та продаж (SHORT) ф'ючерсних контрактів.
- **Управління ризиками:** Реалізація механізмів встановлення рівнів Take Profit (TP) та Stop Loss (SL) для автоматичного закриття позицій при досягненні певних цінових рівнів.
- **Кредитне плече (Leverage):** Можливість для користувача вказувати бажане кредитне плече для своїх угод, що є невід'ємною частиною ф'ючерсної торгівлі.
- **Моніторинг та управління:** Надання користувачеві інформації про його відкриті ордери, поточні позиції, їх прибутковість (PNL), а також можливість їх модифікації або закриття.
- **Автоматизація на основі сигналів:** Унікальною особливістю предметної області даного проєкту є розробка модуля для відстеження повідомлень у зазначених користувачем Telegram-каналах. Бот повинен аналізувати ці повідомлення на наявність торгових сигналів (назва монети, напрямок, параметри угоди) та, за умови відповідності певним критеріям (наприклад, ETA, співвідношення ризику до прибутку), автоматично відкривати позиції на біржі.
- **Зберігання користувацьких даних:** Безпечне зберігання налаштувань користувача, таких як API-ключі від біржі (хоча в даному проєкті вони вводяться і зберігаються локально у користувача бота), списки обраних монет, параметри TP/SL за замовчуванням, налаштування трекера каналів.

Метою даної дипломної роботи є розробка та реалізація Telegram-бота, який би надавав користувачам вищезазначений функціонал, використовуючи сучасні технології та інструменти, зокрема мову програмування Python, фреймворк aiogram для розробки ботів, бібліотеку ccxt для взаємодії з API Binance та СУБД SQLite для локального зберігання даних. Для досягнення цієї мети необ-

хідно провести аналіз предметної області, визначити детальні вимоги до функціоналу, спроєктувати архітектуру системи, реалізувати програмні модулі та провести їх ретельне тестування.

Розробка такого Telegram-бота має значний практичний потенціал, оскільки може слугувати зручним та функціональним інструментом для трейдерів різного рівня, які прагнуть оптимізувати свою взаємодію з криптовалютною біржею та автоматизувати частину своїх торгових стратегій.

## 3.2 Основні етапи створення Telegram-бота для взаємодії з криптовалютною біржею

Розробка Telegram-бота для взаємодії з криптовалютною біржею Binance є комплексним процесом, що вимагає послідовного виконання низки етапів, від формування концепції до тестування та підготовки до потенційного розгортання. Кожен етап є важливим для досягнення поставленої мети та створення функціонального й надійного програмного продукту.

**Формування концепції та визначення ключових принципів Telegram-бота:**

Концепція розроблюваного Telegram-бота полягає у створенні програмного інструменту, що надає користувачам зручний та частково автоматизований спосіб взаємодії з ф'ючерсною тестовою платформою біржі Binance через інтерфейс месенджера Telegram. Основні принципи, що лежать в основі цієї концепції, включають:

*Мобільність та доступність:* Надання можливості керувати торговими операціями та відстежувати ринок з будь-якого пристрою, де встановлений Telegram, без необхідності використання спеціалізованих десктопних програм або постійного доступу до веб-інтерфейсу біржі.

*Поєднання ручного контролю та автоматизації:* Бот повинен дозволяти користувачеві як самостійно розміщувати ринкові та лімітні ордери, так і автоматизувати частину торгових процесів, зокрема через відстеження сигналів з Telegram-каналів.

*Інтуїтивно зрозумілий інтерфейс:* Взаємодія з ботом має відбуватися через прості команди та інтерактивні клавіатури, зрозумілі для користувачів з різним рівнем технічної підготовки та досвіду в трейдингу.

*Основний торговий функціонал:* Забезпечення ключових функцій для торгівлі:

- Управління API-ключами (додавання, видалення, тестування).
- Моніторинг цін на обрані криптовалютні пари.
- Розміщення ордерів (market, limit) на купівлю (LONG) та продаж (SHORT).
- Управління відкритими ордерами та позиціями.
- Налаштування та застосування параметрів Take Profit та Stop Loss.

*Автоматизація на основі сигналів:* Надання унікальної можливості налаштовувати відстеження обраних Telegram-каналів, аналізувати повідомлення на наявність торгових сигналів та автоматично відкривати позиції відповідно до заданих користувачем критеріїв (монета, ETA, співвідношення ризик/прибуток, кредитне плече, розмір позиції).

*Безпека (в межах проєкту):* Хоча бот працює з Testnet, принципи безпечного поводження з API-ключами (зберігання в локальній БД користувача бота, рекомендації щодо обмеження прав ключів) є важливими.

*Локальне зберігання налаштувань:* Збереження індивідуальних налаштувань кожного користувача (API-ключі, обрані монети, параметри TP/SL, налаштування трекера) у локальній базі даних SQLite для забезпечення персоналізованого досвіду.

**Основні етапи розробки Telegram-бота:**

Для успішної реалізації проєкту було пройдено декілька ключових етапів:

***Аналіз вимог та дослідження предметної області:***

*Визначення функціональних та нефункціональних вимог:* Детальне визначення того, що саме повинен робити бот (функціонал) та якими характеристиками він має володіти (надійність, швидкість реакції, зручність використання).

*Аналіз цільової аудиторії:* Розуміння потреб потенційних користувачів бота (трейдери-початківці, досвідчені трейдери, які шукають автоматизацію).

*Дослідження API Binance Futures Testnet:* Вивчення документації API для розуміння доступних ендпоінтів, форматів запитів та відповідей, лімітів та особливостей роботи з ф'ючерсами.

*Аналіз Telegram Bot API:* Вивчення можливостей, що надаються Telegram для створення інтерактивних ботів (типи повідомлень, клавіатури, команди, FSM).

*Огляд існуючих аналогів та технологій:* Аналіз конкурентних рішень та вибір оптимального стеку технологій (Python, aiogram, ccxt, aiosqlite).

***Проєктування архітектури та бази даних:***

*Розробка загальної архітектури бота:* Визначення основних модулів (обробники команд, клієнт для взаємодії з Binance, модуль роботи з базою даних, модуль трекера каналів) та їх взаємодії. У даному проєкті це відображено у структурі каталогів та файлів (handlers, database, mybinance, main.py).

*Проєктування схеми бази даних SQLite:* Визначення таблиць (users, favorites, orders, tp_sl_settings, channel_tracker_settings, tracked_coins), їх полів та зв'язків для зберігання необхідної інформації.

*Проєктування логіки взаємодії з користувачем:* Розробка сценаріїв діалогів для кожної функції бота, визначення станів FSM.

*Визначення форматів даних для обміну з API Binance та обробки сигналів.*

***Реалізація функціональних модулів:***

*Налаштування середовища розробки:* Встановлення Python, необхідних бібліотек, налаштування віртуального середовища.

*Розробка модуля бази даних (db.py):* Написання функцій для ініціалізації БД, створення таблиць, додавання, оновлення, видалення та вибірки даних.

*Розробка клієнта для Binance (binance_client.py):* Інкапсуляція логіки взаємодії з API Binance за допомогою бібліотеки ccxt (розміщення ордерів, отримання даних тощо).

*Розробка обробників команд та інтерфейсу користувача (handlers/*.py):* Реалізація логіки для кожної кнопки меню та команди бота, включаючи обробку введення користувача, валідацію даних, взаємодію з binance_client та db.

*Реалізація модуля трекера каналів (channel_tracker.py):* Розробка класу ChannelTracker, логіки підписки на повідомлення, парсингу сигналів, фільтрації, автоматичного відкриття позицій та сповіщення користувача.

*Налаштування головного диспетчера та запуск бота (main.py):* Ініціалізація бота, диспетчера, підключення роутерів, налаштування логування, реалізація функції відновлення активних трекерів.

***Тестування та налагодження:***

*Модульне тестування:* Перевірка коректності роботи окремих функцій (наприклад, розрахунок розміру позиції, парсинг ETA з повідомлення).

*Інтеграційне тестування:* Перевірка взаємодії між модулями (наприклад, чи коректно обробник команди викликає функції binance_client та зберігає результат у db).

*Тестування на Binance Futures Testnet:* Активне використання тестової платформи для перевірки всіх торгових операцій (розміщення ордерів, TP/SL, закриття позицій) без ризику реальними коштами.

*Тестування функціоналу трекера каналів:* Перевірка коректності зчитування повідомлень з тестових каналів, спрацювання фільтрів та автоматичного відкриття позицій.

*Виявлення та усунення помилок (дебаггінг):* Використання логів та налагоджувача для пошуку та виправлення проблем у коді.

*Тестування користувацького інтерфейсу:* Перевірка зручності навігації, зрозумілості повідомлень та загального досвіду користувача.

Послідовне виконання цих етапів дозволило систематично підійти до розробки Telegram-бота та створити програмний продукт, що відповідає поставленій меті.

## 3.3 Проєктування Telegram-бота для взаємодії з криптовалютною біржею

Проєктування програмного забезпечення є фундаментальним етапом життєвого циклу розробки, який визначає архітектуру, компоненти, інтерфейси, структуру даних та інші ключові характеристики майбутньої системи. Результатом проєктування є проєкт – цілісна сукупність моделей та специфікацій, що слугують основою для подальшої реалізації. Для проєктування Telegram-бота для взаємодії з криптовалютною біржею Binance були застосовані підходи, що дозволяють структурувати систему, визначити її основні функціональні блоки та взаємозв'язки між ними. Хоча для Telegram-ботів не завжди створюється повний набір формальних UML-діаграм, як для великих корпоративних систем, використання концептуальних моделей та діаграм варіантів використання є корисним для візуалізації та кращого розуміння функціоналу та взаємодії користувача з системою.

**Визначення архітектури системи**

Архітектура розроблюваного Telegram-бота базується на модульному принципі та асинхронній моделі взаємодії. Ключовими компонентами архітектури є:

1. Модуль взаємодії з Telegram (Telegram Interface Module):
   - *Відповідальність:* Прийом та обробка оновлень (повідомлень, команд, колбеків) від Telegram Bot API, надсилання відповідей користувачеві, управління клавіатурами та станами діалогів.
   - *Реалізація:* Здійснюється за допомогою фреймворку aiogram. Включає диспетчер (Dispatcher), роутери (Router) та обробники (handler) для

різних типів взаємодії. Файли в каталозі handlers (main_menu.py, api_keys.py, favorites.py, market.py, orders.py, positions.py, tp_sl.py, channel_tracker.py) реалізують цю частину.

2. Модуль взаємодії з API Binance (Binance Client Module):
   - *Відповідальність:* Інкапсуляція логіки роботи з API біржі Binance Futures Testnet. Це включає автентифікацію, розміщення та скасування ордерів, отримання ринкових даних (ціни, інформація про символи), запит балансу, відкритих ордерів та позицій, встановлення кредитного плеча та параметрів TP/SL.
   - *Реалізація:* Здійснюється за допомогою бібліотеки ccxt та власного класу-обгортки BinanceClient у файлі mybinance/binance_client.py.
3. Модуль управління даними (Data Management Module):
   - *Відповідальність:* Зберігання та управління даними користувачів, їх налаштуваннями, API-ключами, списками обраних монет, параметрами TP/SL, налаштуваннями трекера каналів та відстежуваними монетами.
   - *Реалізація:* Здійснюється через взаємодію з базою даних SQLite за допомогою бібліотеки aiosqlite. Функції для роботи з БД інкапсульовані у файлі database/db.py.
4. Модуль трекера каналів (Channel Tracker Module):
   - *Відповідальність:* Відстеження повідомлень у зазначених користувачем Telegram-каналах, аналіз повідомлень на наявність торгових сигналів, фільтрація сигналів за заданими критеріями та автоматичне відкриття позицій на біржі через Binance Client Module.
   - *Реалізація:* Представлений класом ChannelTracker у файлі handlers/channel_tracker.py, а також механізмом відновлення активних трекерів у main.py.
5. Головний модуль управління (Main Control Module):
   - *Відповідальність:* Ініціалізація всіх компонентів системи, налаштування логування, запуск бота та управління його життєвим циклом.
   - *Реалізація:* Файл main.py.

Така модульна архітектура забезпечує розділення відповідальностей, що спрощує розробку, тестування та подальшу модифікацію системи. Асинхронна природа взаємодії між модулями (особливо з Telegram API та потенційно тривалими операціями з API біржі) є ключовою для забезпечення високої швидкодії та чутливості бота.

**Діаграма варіантів використання (Use Case Diagram)**

Діаграма варіантів використання є ефективним інструментом для візуалізації функціональних можливостей системи та взаємодії користувачів (акторів) з нею. Для розроблюваного Telegram-бота основним актором є Користувач (Трейдер).

Для побудови діаграми варіантів використання були виконані наступні кроки:

- Визначено актора:
    - Користувач (Трейдер): Особа, яка взаємодіє з Telegram-ботом для отримання інформації про ринок, управління своїм торговим рахунком на Binance Futures Testnet та налаштування автоматичного відстеження сигналів.
- Ідентифіковано основні варіанти використання (use cases): Це процеси або функції, які Користувач може виконувати за допомогою бота.

Основні варіанти використання для Telegram-бота:

1. Керування API-ключами:
    - Додати API-ключі Binance.
    - Перевірити API-ключі.
    - Видалити API-ключі.
2. Керування обраними монетами:
    - Додати монету до списку обраних.
    - Видалити монету зі списку обраних.
    - Переглянути поточні ціни обраних монет.
3. Торгові операції:
    - Розмістити ринковий ордер (Market Order) (LONG/SHORT).

- Розмістити лімітний ордер (Limit Order) (LONG/SHORT).

4. Керування ордерами:
    - Переглянути відкриті ордери.
    - Скасувати відкритий ордер.
5. Керування позиціями:
    - Переглянути відкриті позиції (включаючи PNL).
    - Закрити відкриту позицію.
6. Налаштування TP/SL:
    - Встановити/змінити значення Take Profit за замовчуванням.
    - Встановити/змінити значення Stop Loss за замовчуванням.
    - Скинути налаштування TP/SL.
7. Керування трекером каналів:
    - Налаштувати канал-джерело сигналів.
    - Налаштувати мінімальне та максимальне ETA для сигналів.
    - Налаштувати кредитне плече для угод з трекера.
    - Налаштувати розмір позиції (% від балансу) для угод з трекера.
    - Додати монету для відстеження трекером (з мінімальним співвідношенням).
    - Видалити монету з відстеження трекером.
    - Переглянути список відстежуваних монет трекером.
    - Запустити трекер каналів.
    - Зупинити трекер каналів.
8. Отримання сповіщень:
    - (Пасивний варіант використання) Отримувати сповіщення про відкриття позиції трекером.

Візуалізація діаграми варіантів використання (див. рис. 10):

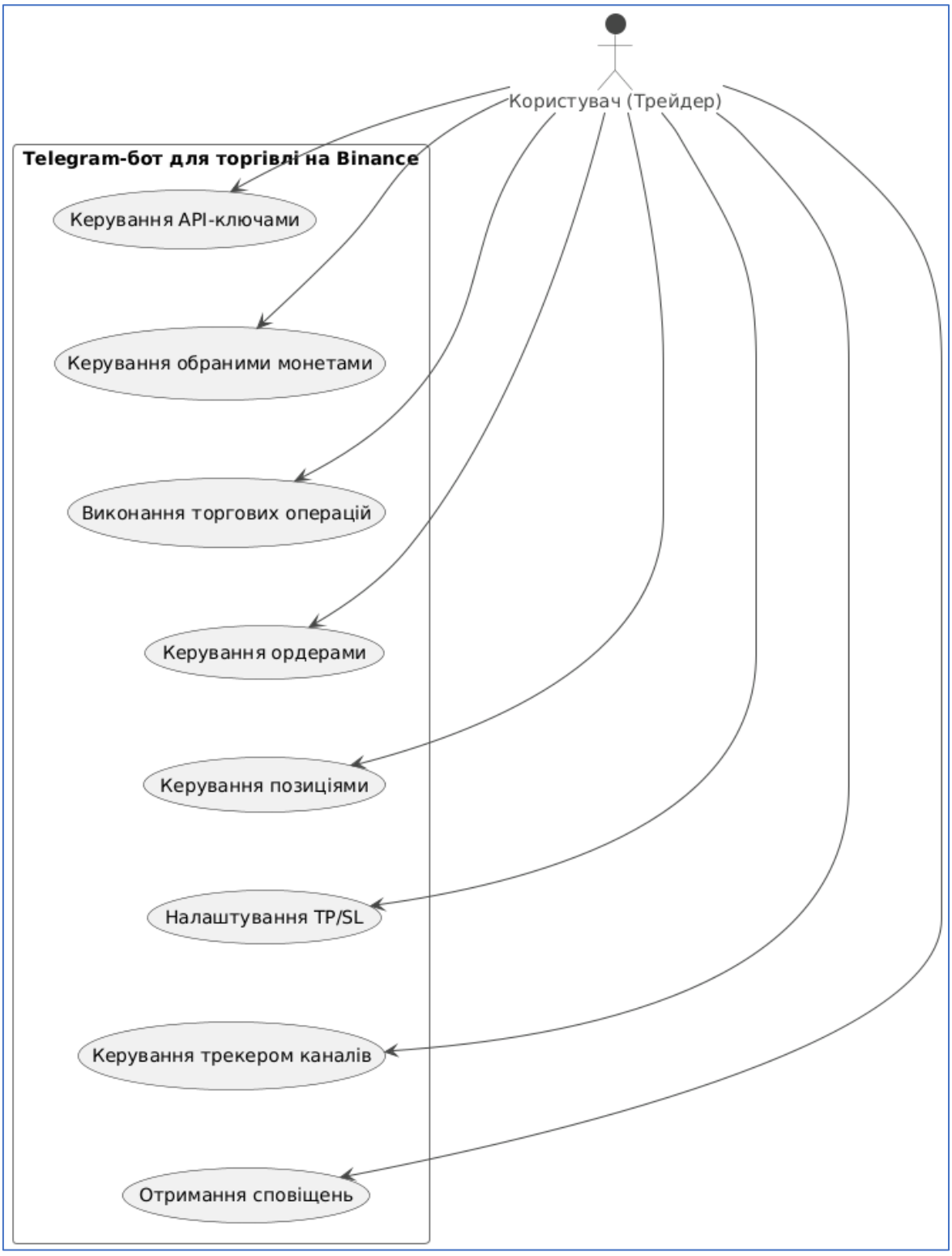



Рисунок *10 — Стисла діаграма варіантів використання для Telegram-бота*

**Актор:** "Користувач" (зображується у вигляді фігурки людини).

**Система:** Прямокутник з назвою "Telegram-бот для торгівлі на Binance".

**Варіанти використання:** Овали всередині прямокутника системи, кожен з яких відповідає одному з перерахованих вище пунктів (наприклад, "Додати API-ключі", "Розмістити ринковий ордер", "Налаштувати трекер каналів" тощо).

**Зв'язки (Associations):** Лінії, що з'єднують Актора з кожним варіантом використання, з яким він може взаємодіяти.

Ця UML-діаграма варіантів використання допомагає чітко визначити обсяг функціоналу системи з точки зору користувача та слугує основою для подальшого детального проєктування та розробки кожного функціонального блоку. Вона візуалізує основні можливості, які надає Telegram-бот, та способи взаємодії користувача з ним для досягнення своїх торгових цілей.

**Проєктування взаємодії компонентів (Діаграма послідовності)**

Для деталізації взаємодії між основними компонентами системи під час виконання ключових операцій використовуються діаграми послідовності (Sequence Diagram). Вони візуалізують обмін повідомленнями між об'єктами в часі. Розглянемо проєктування взаємодії на прикладі сценарію **"Автоматичне відкриття позиції трекером каналів при отриманні сигналу (див рис. 11)"**.

Цей сценарій ілюструє взаємодію наступних учасників (ліній життя):

- **Telegram API**: Зовнішня система, що надсилає оновлення (нові повідомлення в каналах).
- **Aiogram Dispatcher**: Компонент вашого бота, що приймає та маршрутизує оновлення.
- **ChannelTracker Handler**: Обробник в channel_tracker.py, відповідальний за логіку трекера.
- **ChannelTracker Instance**: Конкретний екземпляр класу ChannelTracker для користувача.
- **Database (db.py)**: Модуль для взаємодії з базою даних SQLite.
- **BinanceClient (binance_client.py)**: Клієнт для взаємодії з API Binance.
- **Binance API**: Зовнішня система (API біржі Binance).

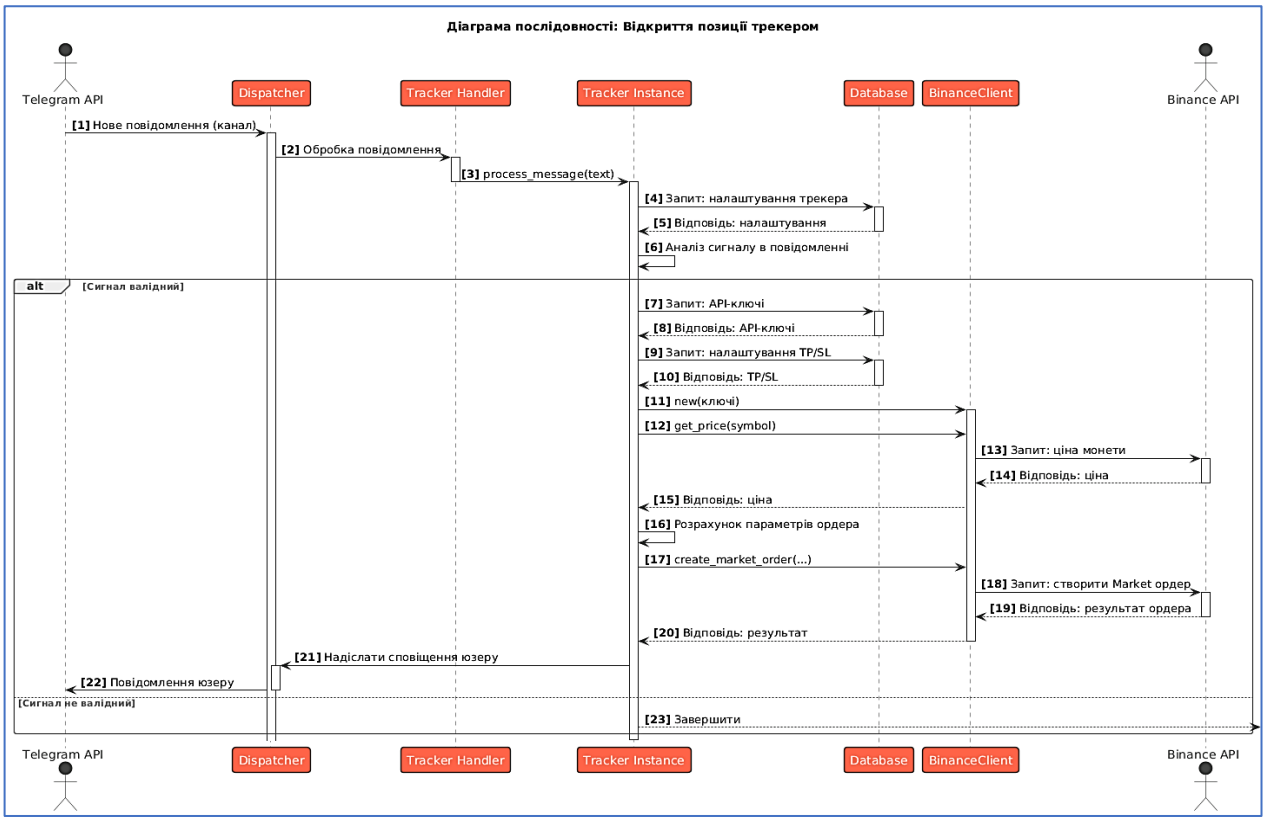



Рисунок *11 — Діаграма послідовності "Автоматичне відкриття позиції трекером"*

1. **Отримання повідомлення:** Telegram API надсилає нове повідомлення з відстежуваного каналу до Aiogram Dispatcher.
2. **Маршрутизація:** Aiogram Dispatcher визначає, що це повідомлення має бути оброблене відповідним ChannelTracker Handler (на основі фільтрів чату та типу повідомлення).
3. **Передача повідомлення до екземпляру трекера:** ChannelTracker Handler передає текст повідомлення до активного екземпляру ChannelTracker Instance відповідного користувача.
4. **Завантаження налаштувань трекера:** ChannelTracker Instance звертається до Database для отримання актуальних налаштувань трекера для даного користувача (список відстежуваних монет, мінімальне ETA, співвідношення, кредитне плече, розмір позиції).

5. **Аналіз повідомлення:** ChannelTracker Instance аналізує (парсить) текст повідомлення на наявність торгового сигналу та відповідність заданим критеріям (монета, ETA, потенціал росту, співвідношення).
6. **Прийняття рішення:** Якщо сигнал відповідає всім критеріям:
    - **Отримання API-ключів:** ChannelTracker Instance запитує API-ключі користувача з Database.
    - **Отримання налаштувань TP/SL:** ChannelTracker Instance запитує налаштування Take Profit та Stop Loss користувача з Database.
    - **Ініціалізація BinanceClient:** Створюється екземпляр BinanceClient з отриманими API-ключами.
    - **Отримання ринкових даних:** BinanceClient надсилає запит до Binance API для отримання актуальної ціни монети.
    - **Розрахунок параметрів ордера:** ChannelTracker Instance розраховує розмір позиції, ціни TP/SL на основі поточних даних та налаштувань.
    - **Створення ордера на біржі:** BinanceClient надсилає запит на створення ринкового ордера (та подальших ордерів TP/SL) до Binance API.
    - **Обробка відповіді від біржі:** Binance API повертає результат виконання ордера до BinanceClient.
7. **Сповіщення користувача:** ChannelTracker Instance (через Aiogram Dispatcher) надсилає повідомлення користувачеві в Telegram про успішне (або неуспішне) відкриття позиції.

Ця діаграма послідовності демонструє логічний потік операцій та обмін даними між ключовими компонентами системи при реалізації однієї з найскладніших функцій бота. Аналогічні діаграми можуть бути побудовані для інших важливих сценаріїв, таких як розміщення ордера вручну або налаштування API-ключів, для більш глибокого розуміння внутрішньої взаємодії.

**Проєктування структури бази даних**

Як було зазначено, для зберігання даних використовується SQLite. Схема бази даних була спроєктована для зберігання наступної інформації (детальний опис таблиць та полів наведено у файлі database/db.py та може бути представлений у вигляді ER-діаграми):

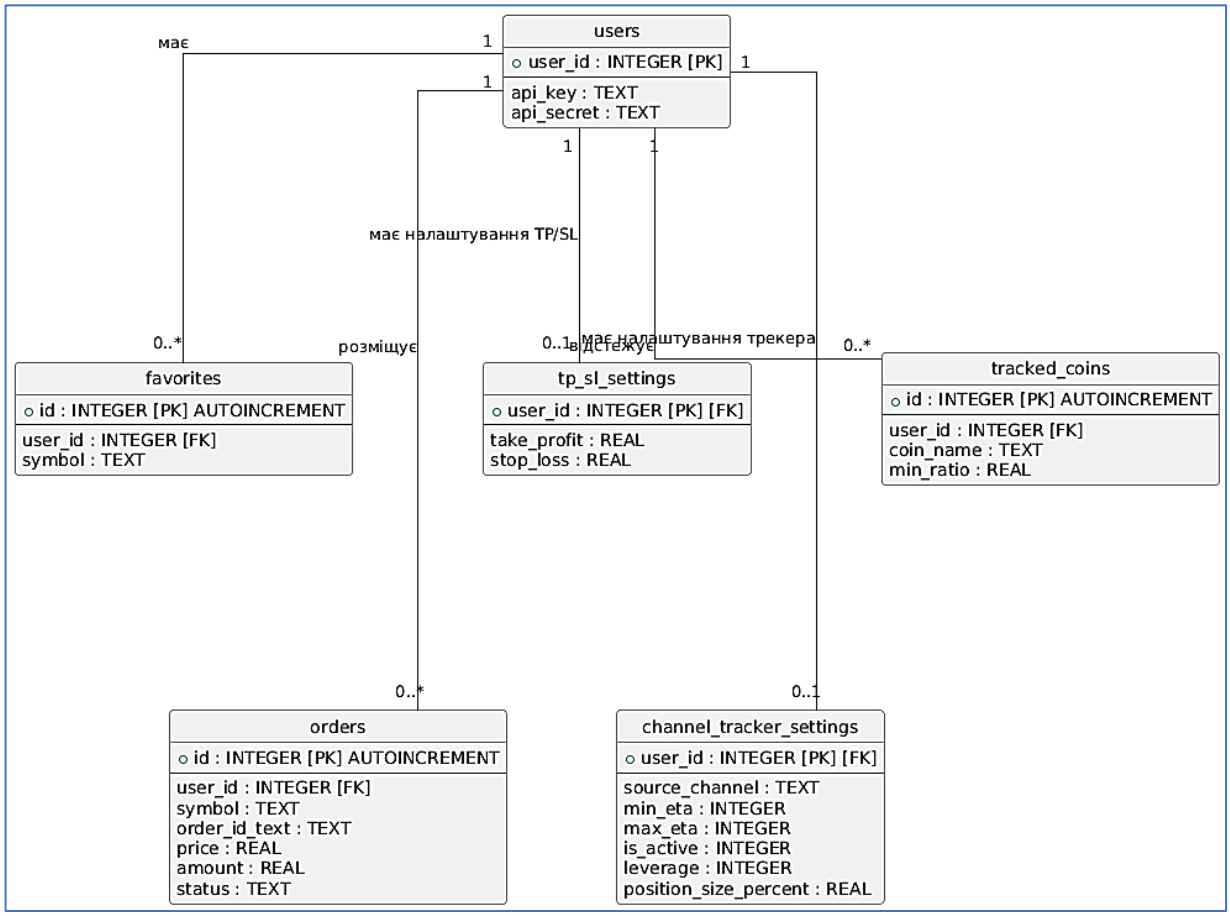



Рисунок *12 — Схема бази даних*

- **users**: Інформація про користувачів бота, їх API-ключі.
- **favorites**: Списки обраних користувачами криптовалютних пар.
- **orders**: (Потенційно для історії або кешування, оскільки актуальні ордери беруться з біржі) Інформація про створені ордери.
- **tp_sl_settings**: Індивідуальні налаштування Take Profit та Stop Loss для кожного користувача.
- **channel_tracker_settings**: Налаштування модуля відстеження каналів для кожного користувача (канал-джерело, ETA, активність, плече, розмір позиції).

- **tracked_coins**: Список монет, які користувач відстежує через трекер, з їх мінімальними співвідношеннями.

Проєктування бази даних враховувало необхідність забезпечення цілісності даних (через зовнішні ключі, де це доречно) та ефективного доступу до інформації.

**Проєктування інтерфейсу користувача (Telegram UI)**

Інтерфейс користувача реалізується через команди Telegram та інтерактивні клавіатури (ReplyKeyboardMarkup та InlineKeyboardMarkup). Проєктування UI включало:

- **Визначення основних команд:** /start, /help (потенційно).
- **Структурування меню:** Розробка логічної ієрархії меню для доступу до різних функцій (головне меню, меню API-ключів, меню обраних монет, меню ордерів, меню позицій, меню налаштувань TP/SL, меню трекера каналів).
- **Дизайн діалогів:** Продумування послідовності запитань та відповідей для кожної операції (наприклад, при додаванні API-ключа бот послідовно запитує ключ та секрет; при створенні ордера – символ, ціну, суму, плече, сторону).
- **Формати повідомлень:** Визначення того, як бот буде надавати інформацію користувачеві (наприклад, формат відображення цін, відкритих позицій, повідомлень про помилки).

Проєктування інтерфейсу було спрямоване на забезпечення інтуїтивності, простоти використання та надання користувачеві всієї необхідної інформації для прийняття рішень.

## 3.4 Програмна реалізація застосунку

Файл main.py слугує головною точкою входу та координаційним центром для запуску Telegram-бота. Його основне завдання – ініціалізувати всі ключові

компоненти системи та запустити процес обробки повідомлень. На початку роботи скрипт імпортує необхідні бібліотеки, зокрема asyncio для асинхронної роботи, компоненти aiogram (Bot, Dispatcher) для створення та управління ботом, а також модулі з конфігурацією, базою даних та обробниками команд. Також налаштовується базова система логування для моніторингу роботи додатка.

Лістинг 1. *Файл запуску торгового бота, main.py*

```
API_TOKEN = config.BOT_TOKEN

logging.basicConfig(level=logging.INFO)

async def main():
    bot = Bot(token=API_TOKEN,
default=DefaultBotProperties(parse_mode=ParseMode.HTML))

    from handlers import main_menu, market, api_keys,
favorites, orders, positions, tp_sl, channel_tracker
    from database.db import init_db,
get_channel_tracker_settings

    await init_db()
    await bot.delete_webhook(drop_pending_updates=True)
    await restore_active_trackers(bot)
    await dp.start_polling(bot, allowed_updates=["message",
"edited_message", "channel_post", "edited_channel_post"])

async def restore_active_trackers(bot):
    from database.db import get_channel_tracker_settings
    from handlers.channel_tracker import ChannelTracker,
tracker_instances
    import logging

    async with aiosqlite.connect("users.db") as db:
        db.row_factory = aiosqlite.Row
        async with db.execute("SELECT user_id FROM
channel_tracker_settings WHERE is_active=1") as cursor:
            active_trackers = await cursor.fetchall()

    for tracker_data in active_trackers:
        user_id = tracker_data['user_id']
        source_channel, min_eta, max_eta, is_active,
leverage, position_size_percent = await
get_channel_tracker_settings(user_id)
```

```
if __name__ == "__main__":
    asyncio.run(main())
```

Файл db.py відповідає за всю взаємодію Telegram-бота з локальною базою даних SQLite, що має назву users.db. Він інкапсулює логіку створення структури бази даних та надання асинхронних функцій для маніпулювання даними. При першому запуску або за необхідності викликається функція init_db(), яка створює шість основних таблиць, якщо вони ще не існують: users (для зберігання ID користувачів та їхніх API-ключів Binance), favorites (для обраних криптовалютних пар), orders (для запису інформації про ордери), tp_sl_settings (для індивідуальних налаштувань Take Profit та Stop Loss), channel_tracker_settings (для параметрів відстеження Telegram-каналів, включаючи канал-джерело, ETA, кредитне плече та розмір позиції) та tracked_coins (для монет, які користувач відстежує через трекер). Також в init_db() передбачена логіка для міграції схеми, зокрема додавання нових колонок (leverage, position_size_percent) до таблиці channel_tracker_settings без втрати існуючих даних.

Лістинг 2 *Створення та ініціалізація бази даних.*

```
async def init_db():
    async with aiosqlite.connect(DB_PATH) as db:
        await db.execute("""
        CREATE TABLE IF NOT EXISTS users (
            user_id INTEGER PRIMARY KEY,
            api_key TEXT,
            api_secret TEXT
        )
        """)
        await db.execute("""
        CREATE TABLE IF NOT EXISTS favorites (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            user_id INTEGER,
            symbol TEXT,
            FOREIGN KEY(user_id) REFERENCES users(user_id)
        )
        """)
        await db.execute("""
        CREATE TABLE IF NOT EXISTS orders (
```

```
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            user_id INTEGER,
            symbol TEXT,
            order_id TEXT,
            price REAL,
            amount REAL,
            status TEXT,
            FOREIGN KEY(user_id) REFERENCES users(user_id)
        )
        """)
        await db.execute("""
        CREATE TABLE IF NOT EXISTS tp_sl_settings (
            user_id INTEGER PRIMARY KEY,
            take_profit REAL,
            stop_loss REAL,
            FOREIGN KEY(user_id) REFERENCES users(user_id)
        )
        """)
```

Файл binance_client.py інкапсулює всю логіку взаємодії Telegram-бота з API криптовалютної біржі Binance Futures Testnet. Він визначає клас BinanceClient, який використовує бібліотеку ccxt для уніфікованого доступу до функціоналу біржі. При ініціалізації екземпляра BinanceClient приймає API-ключ та секрет користувача, налаштовуючи з'єднання з тестовою мережею ф'ючерсів Binance та активуючи обмеження швидкості запитів для дотримання правил біржі. Клас надає набір асинхронних методів, які абстрагують низькорівневі виклики ccxt та надають зручний інтерфейс для інших модулів бота.

Лістинг 3 Модуль взаємодії з біржею Binance

```
class BinanceClient:
    def __init__(self, api_key: str, api_secret: str):
        self.exchange = ccxt.binanceusdm({
            'apiKey': api_key,
            'secret': api_secret,
            'enableRateLimit': True,
            'options': {
                'defaultType': 'future',
                'testnet': True,
                'adjustForTimeDifference': True,
                'warnOnFetchOpenOrdersWithoutSymbol': False
            }
```

```
        })

    async def test_api_keys(self):
        try:
            account_info = self.exchange.fetch_balance()
            if account_info and 'total' in account_info:
                return True
            return False
        except Exception as e:
            return False

    async def get_symbol_info(self, symbol: str):
        markets = self.exchange.load_markets()
        if symbol in markets:
            return markets[symbol]

        if symbol.endswith('USDT'):
            base = symbol[:-4]
            futures_symbol = f"{base}/USDT:USDT"
            if futures_symbol in markets:
               return markets[futures_symbol]
```

# ВИСНОВКИ

В ході виконання роботи було досягнуто наступних результатів:

1. Проведено комплексний аналіз предметної області, що включав огляд літературних джерел, дослідження існуючих програмних засобів для торгівлі криптовалютами (включаючи торгові платформи та Telegram-боти-аналоги), а також вивчення сучасних тенденцій розвитку таких систем. Аналіз показав значний попит на інструменти, що поєднують зручність мобільного доступу, можливості ручного управління та елементи автоматизації.
2. Досліджено та обґрунтовано вибір програмних засобів та технологій для реалізації проєкту. Основною мовою программування обрано Python завдяки її простоті, великій кількості бібліотек та підтримці асинхронного програмування.
3. Здійснено проєктування Telegram-бота, що включало визначення архітектури системи, розробку діаграми варіантів використання для візуалізації взаємодії користувача з системою, проєктування схеми бази даних SQLite для зберігання користувацьких даних та налаштувань, а також проєктування логіки користувацького інтерфейсу на базі команд та інтерактивних клавіатур Telegram. Модульна архітектура забезпечила чітке розділення відповідальностей між компонентами системи, що спростило розробку та тестування.
4. Проведено тестування розробленого Telegram-бота на тестовій платформі Binance Futures Testnet. Тестування підтвердило коректність роботи основного торгового функціоналу, правильність обробки сигналів модулем трекера каналів, надійність зберігання та отримання даних з бази SQLite, а також зручність користувацького інтерфейсу.

5. Розроблений Telegram-бот надає користувачам зручний та функціональний інструмент для взаємодії з криптовалютною біржею Binance, поєднуючи можливості ручного трейдингу з елементами автоматизації на основі аналізу сигналів з Telegram-каналів.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Офіційна документація бібліотеки CCXT. веб-сайт. URL: https://docs.ccxt.com/ (дата звернення: 07.05.2025).

2. Офіційна документація фреймворку Aiogram. веб-сайт. URL: https://docs.aiogram.dev/ (дата звернення: 07.05.2025).

3. Баздирев А. (2021). *Алгоритмічна торгівля з використанням текстових сигналів та навчання з підкріпленням* (Магістерська дисертація). Національний університет «Києво-Могилянська академія». Київ. веб-сайт. URL: https://ela.kpi.ua/bitstream/123456789/46542/1/Bazdyrev_magistr.pdf (дата звернення: 07.05.2025).

4. Ролік О. О., Корнієнко Б. Я. (2023). *Автоматизована система арбітражної торгівлі криптовалют* (Дипломний проєкт). КПІ ім. Ігоря Сікорського. Київ. веб-сайт. URL: https://ela.kpi.ua/items/3cb6bd1e-1dbc-4a36-9588-b0667570d44b (дата звернення: 07.05.2025).

5. Основи AI Rig Complex (ARC): Що потрібно знати кожному трейдеру. (2025, 24 травня). *CoinUnited.io*. веб-сайт. URL: https://coinunited.io/uk/learn/ai-rig-complex-arc-basics (дата звернення: 07.05.2025).

6. Що таке бот для трейдингу криптовалют | Як працює торговий криптобот. (2024, 17 жовтня). *WhiteBIT Blog*. веб-сайт. URL: https://whitebit.com/uk/blog/what-is-a-cryptocurrency-trading-bot (дата звернення: 07.05.2025).

7. Алгоритмічна торгівля: автоматизація ринку для трейдерів. (2025, 15 травня). *Incrypted*. веб-сайт. URL: https://incrypted.com/algorithmic-trading-automation-market-for-traders/ (дата звернення: 07.05.2025).

8. Як створити свій бот у Telegram. *SendPulse UA*. веб-сайт. URL: https://sendpulse.ua/knowledge-base/chatbot/telegram/create-telegram-chatbot (дата звернення: 07.05.2025).

9. Парсинг Telegram каналів: 7 ботів для ефективного парсингу. (2024, 28 лютого). *Lead Panda Media*. веб-сайт.
URL: https://leadpanda.media/uk/blog/parsing-telegram-channels/ (дата звернення: 07.05.2025).
10. Документація SQLite. веб-сайт.
URL: https://www.sqlite.org/docs.html (дата звернення: 07.05.2025).
11. Як створити телеграм-бота на Python. *SpaceLab*. веб-сайт. URL: https://spacelab.com.ua/blog/yak-stvoryty-telehram-bota-na-python/ (дата звернення: 07.05.2025).
12. Офіційна документація Binance API. веб-сайт. URL: https://binance-docs.github.io/apidocs/futures/en/ (дата звернення: 07.05.2025).
13. Як захистити свої ключі API на Binance: 5 порад для безпеки вашої криптовалюти. *Binance Support*. веб-сайт. URL: https://www.binance.com/uk-UA/support/faq/how-to-secure-your-api-keys-on-binance-5-tips-for-keeping-your-crypto-safe-360043930052 (дата звернення: 07.05.2025).
14. 10 найкращих ботів для криптовалютної торгівлі AI (червень 2025 р.). (2025, 31 травня). *Unite.AI*. веб-сайт. URL: https://www.unite.ai/uk/10-best-ai-crypto-trading-bots/ (дата звернення: 07.05.2025).
15. A CCXT Alternative & Better Universal Crypto Exchange API. веб-сайт. URL: https://medium.com/vezgo/a-ccxt-alternative-better-universal-crypto-exchange-api-vezgo-ec7cb8431d22 (дата звернення: 07.05.2025).

**Декларація**
**академічної доброчесності**
**здобувача ступеня вищої освіти ЗНУ**

Я, Газієв Даніл-Бєк Аліханович, студент 4 курсу, форми навчання денної, Інженерного навчально-наукового інституту ім. Ю.М. Потебні ЗНУ, спеціальність 121 Інженерія програмного забезпечення освітньої програми "Програмне забезпечення систем", адреса електронної пошти se121-01@stu.zsea.edu.ua, — підтверджую, що написана мною кваліфікаційна робота на тему «**Комп'ютерна системи розпізнавання автомобільних номерів**» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст.42 Закону України «Про освіту», зі змістом яких ознайомлений.

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

- згоден на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою інтернет-системи, а також на архівування моєї роботи в базі даних цієї системи.

Дата 02.06.2025 _____ Газієв Даніл-Бєк Аліханович
(підпис ) (прізвище та ініціали) (студент)
Дата 02.06.2025 _____ Заяц Валерій Іванович
(підпис ) (прізвище та ініціали) (керівник)