УДК 04.092 DOI: 10.31673/2786-8362.2025.018255

Шикула О.М., д.ф.-м.н.; **Коник С.П.;**
Білоусова С.В., к.ф.-м.н.; **Прокопов С.В.,** к.т.н.**; Саміляк І.**

ВИКОРИСТАННЯ ХМАРНИХ ТЕХНОЛОГІЙ ПРИ РОЗРОБЦІ ТЕЛЕГРАМ ЧАТ-БОТУ НА МОВІ PYTHON З ВИКОРИСТАННЯМ БАЗИ ДАНИХ SQLITE

Shykula O.M., Konyk C.P., Bilousova S.V., Prokopov S.V., Samiliak I. Using cloud technologies in the development of a telegram chatbot in Python using SQLite database. Using cloud technologies in the development of a Telegram chatbot in Python using the SQLite database. This article analyzes chatbots and reveals the role that bots play in the modern world. The project was implemented using the Python programming language and the PyCharm integrated development environment. In the process of developing the Telegram chatbot, preference was given to using the SQLite database. To ensure communication with the Telegram messenger, the Aiogram, Sqlite packages and the communication interface with the Telegram Bot API application were used. The bot was officially registered in Telegram with a unique search identifier @TgSQL_bot. Testing was carried out both during the development process and after completion, including manual testing of the functioning and response speed of the program to verify the correctness of the bot's operation. This chatbot is an effective tool for controlling financial expenses through the Telegram messenger, it provides users with convenient access to the necessary information and the ability to interact with it in real time.

Keywords: Telegram chatbot, Python programming language, SQLite database

Шикула О.М., Коник С.П., Білоусова С.В., Прокопов С.В., Саміляк І. Використання хмарних технологій при розробці телеграм чат-боту на мові Python з використанням бази даних SQLite. У даній статті проведено аналіз чат-ботів, виявлено їх роль яку відіграють боти в сучасному світі. Реалізація проекту відбулася з використанням мови програмування Python та інтегрованого середовища розробки PyCharm. У процесі розробки Telegram чат-боту було віддано перевагу використанню бази даних SQLite. Для забезпечення зв'язку з месенджером Telegram використано пакети Aiogram, Sqlite та інтерфейс зв’язку з додатком Telegram Bot API. Бот був офіційно зареєстрований у Telegram з унікальним ідентифікатором пошуку @TgSQL_bot. Проведено тестування як у процесі розробки, так і після завершення, включаючи ручну перевірку функціонування, швидкості відповідей програми для перевірки коректності роботи бота. Цей чат-бот є ефективним інструментом для контроля фінансових витрат через месенджер Telegram, він забезпечує користувачам зручний доступ до необхідної інформації та можливість взаємодії з ним у режимі реального часу.

Ключові слова: телеграм чат-бот, мова програмування Python, база даних SQLite

Вступ

Чат-боти є комп'ютерними програмами, розробленими для взаємодії з користувачами шляхом обміну інформацією, що базується на попередньо визначених алгоритмах та використанні елементів штучного інтелекту.

Швидкий розвиток та поширення чат-ботів обумовлені загальною тенденцією втрати популярності звичайних додатків, оскільки чат-боти можуть виконувати аналогічні функції. Крім того, зростання популярності різноманітних месенджерів також сприяє поширенню чат-ботів. Таким чином, замість завантаження окремого додатка на пристрій користувача, він може скористатися послугою, що працює всередині звичайного месенджеру. Цей підхід забезпечує зручність та легкість використання для користувача, в той же час раціоналізуючи простір на пристрої та оптимізуючи використання ресурсів.

Чат-боти можуть оперативно та автоматично відповідати на запитання користувачів у різних месенджерах, таких як Facebook Messenger, Telegram, Viber та інші. Чат-боти виконують широкий спектр функцій, починаючи від надання інформації користувачам та закінчуючи вирішенням різних завдань. З іншого боку, для компанії використання чат-ботів в месенджерах стає важливим інструментом для взаємодії з клієнтами, покращення обслуговування та підвищення впізнаваності бренду.

Telegram є відомим месенджером, який гарантує конфіденційність і захищеність даних завдяки використанню закритого коду та сучасних алгоритмів шифрування, таких як RSA-2048 та AES. Розробка та впровадження телеграм чат-боту є актуальною задачею, оскільки

чат-боти знаходять своє застосування в різних сферах, починаючи від віртуальної підтримки клієнтів і закінчуючи автоматизацією бізнес процесів.

Постановка завдання. Основними завданнями цього дослідження є:

1. Дослідження теоретичних основ чат-боту Telegram: Провести аналіз принципів функціонування чат-ботів у Telegram, вивчити їхню архітектуру та можливості для подальшого використання у проекті.
2. Визначення всіх переваг та недоліків боту: Проаналізувати переваги та недоліки чат-ботів у Telegram з точки зору їхньої ефективності, зручності використання, можливостей і обмежень.
3. Створення технічного завдання для розробки системи: Розробити детальне технічне завдання, яке містить вимоги до функціоналу чат-боту, його інтерфейсу, методи взаємодії з користувачем, а також вимоги до безпеки та захисту даних.
4. Дослідження мови програмування для створення бота: Вивчити різні мови програмування, їхні особливості та придатність для реалізації поставленого завдання. Обрати найбільш підходящу мову програмування для розробки чат-боту, враховуючи його потреби та особливості.
5. Інтегрувати базу даних для створення унікального користувача в чат-боті.

Аналіз останніх досліджень. Актуальність чат-ботів визначається їхнім потенціалом у вирішенні різноманітних завдань та забезпеченні зручного спілкування з користувачами. З розвитком штучного інтелекту, машинного навчання та обробки природної мови чат-боти стають все більш інтелектуальними та гнучкими у взаємодії з людьми.

Однією з ключових областей використання чат-ботів є клієнтське обслуговування. Чат-боти дозволяють компаніям автоматизувати підтримку користувачів, відповідати на питання та вирішувати проблеми без прив'язки до годин роботи, що допомагає підвищити задоволеність клієнтів і знизити витрати на підтримку [1].

У сфері електронної комерції чат-боти можуть допомогти в забезпеченні персоналізованої консультації та підтримки під час покупок, що підвищує конверсію та забезпечує задоволеність клієнтів [1, 2]. Також, чат-боти можуть бути використані в освітній сфері для надання навчальних матеріалів, проведення тестувань та індивідуалізованої підтримки учнів [2, 3]. Загалом, чат-боти мають великий потенціалі в різних сферах, починаючи від підтримки користувачів та закінчуючи освітніми та розважальними застосуваннями. Ці інтелектуальні програмні агенти відкривають нові можливості для автоматизації та зручного спілкування [3, 4].

Наукова новизна цього дослідження полягає в розробці та дослідженні телеграм-бота. Цей проект включає:

1. Інтеграція в месенджер: чат-бот використовується безпосередньо в Telegram, що забезпечує зручний доступ для користувачів.
2. Функціонал: бот пропонує широкий спектр можливостей для відстеження та аналізу фінансових операцій.
3. Автоматизація: використання інтелектуальних алгоритмів для автоматичної обробки даних та надання корисних рекомендацій користувачам.
4. Аналіз ефективності: оцінка ефективності роботи бота та його потенціалу для покращення.

Метою роботи є глибокий аналіз, розробка та реалізація проєкту – телеграм-боту, спрямованого на систематичне відстеження та облік фінансових операцій. Цей бот розробляється з метою забезпечення користувачів зручним та ефективним інструментом для контролю їх фінансового стану. Він буде здатен автоматично відслідковувати доходи та витрати, фіксувати їх історію, а також надавати корисні фінансові поради на основі зібраної інформації.

Виклад основного матеріалу дослідження

Функціонал Telegram чат-боту. Аналіз наявних чат-ботів показує, що доцільно, щоб Telegram чат-бот мав такий функціонал:

- Приймання повідомлень від користувачів: Бот повинен приймати повідомлення від користувачів у режимі реального часу. Це включає в себе текстові повідомлення, зображення, аудіо та відео.
- Відправлення повідомлень: Бот повинен мати можливість відправляти різні типи повідомлень користувачам, включаючи текст, зображення, аудіо та відео. Це може бути відповіддю на запит користувача або сповіщення про певні події.
- Використання клавіатур і кнопок: Для полегшення взаємодії з користувачами бот може використовувати клавіатури і кнопки. Це дозволить користувачам здійснювати швидкий доступ до певних функцій або команд, а також забезпечить зручну навігацію.
- Обробка команд користувачів: Бот повинен мати можливість розпізнавати та обробляти команди, введені користувачем. Наприклад, це може бути команда на відображення інформації про певний товар або послугу, або запит на виконання певної дії.
- Інтеграція з базою даних SQL: Для зберігання та обробки інформації бот може використовувати базу даних SQL. Це дозволить зберігати дані користувачів, історію чатів, інформацію про продукти чи послуги, а також будь-яку іншу необхідну інформацію.
- Функції штучного інтелекту: Бот може використовувати штучний інтелект для покращення якості обробки запитів користувачів. Наприклад, це може бути система розпізнавання мови, генерація відповідей або навіть система рекомендацій на основі попередніх взаємодій.
- Відправлення сповіщень: Бот може надсилати сповіщення користувачам згідно з певними подіями або регулярними оновленнями. Наприклад, це може бути сповіщення про нові товари чи послуги, акції та знижки або повідомлення про планові технічні роботи.
- Захист інформації користувачів: Надзвичайно важливою є захист інформації користувачів. Бот повинен використовувати шифрування та інші методи безпеки даних для забезпечення конфіденційності та цілісності інформації користувачів [2].

Вибір технологій розробки. Реалізація проекту відбулася з використанням мови програмування Python. Вибір мови програмування пояснюється простотою синтаксису, наявністю великої системи бібліотек та фреймворків [5-9], спеціально створених для розробки телеграм-ботів, а також наявністю потужних бібліотеки для роботи з SQL [10], що дозволяє легко інтегрувати бази даних у телеграм-боти.

Як середовище розробки було вибране інтегроване середовище розробки PyCharm. PyCharm – це потужне та універсальне середовище розробки для Python, яке допомагає розробникам створювати якісний та ефективний код [11].

Для забезпечення зв'язку з месенджером Telegram використано пакети Aiogram, Sqlite та інтерфейс зв’язку з додатком Telegram Bot API [12-14].

У процесі розробки Telegram чат-боту було віддано перевагу використанню бази даних SQLite. База даних SQLite інтегрована безпосередньо у програмний код бота, що дозволило забезпечити простоту встановлення, налаштування та використання [15]. Вбудованість бази даних дозволяє легко інтегрувати БД без необхідності запуску окремого сервера, що робить процес розгортання бота швидшим і простішим [16].

Розробка моделі бота. Розробка моделі бота з використанням сучасних технік проектування та аналізу передбачає систематичне використання передових методів розробки програмного забезпечення, включаючи машинне навчання, природну мову обробки, а також алгоритмічні стратегії для підтримки інтерактивної та ефективної взаємодії з користувачем. Цей підхід дозволяє створити бота, який не лише відповідає потребам сучасного ринку, а й максимально точно відтворює поведінку та реагує на запити користувачів, забезпечуючи високу якість обслуговування.

Для досягнення мети розробки моделі бота застосовуються передові методи аналізу даних та обробки інформації, такі як глибоке навчання, алгоритми класифікації та кластеризації, що

дозволяють моделі бота розпізнавати патерни в поведінці користувачів і надавати відповіді з високою точністю. Крім того, використання аналітики даних дозволяє постійно вдосконалювати функціональні можливості бота, адаптуючи його до змінних потреб користувачів і вимог ринку. На рисунку 1 зображено структуру контекстної діаграми процесу створення бота «ITHelper» для платформи Telegram.

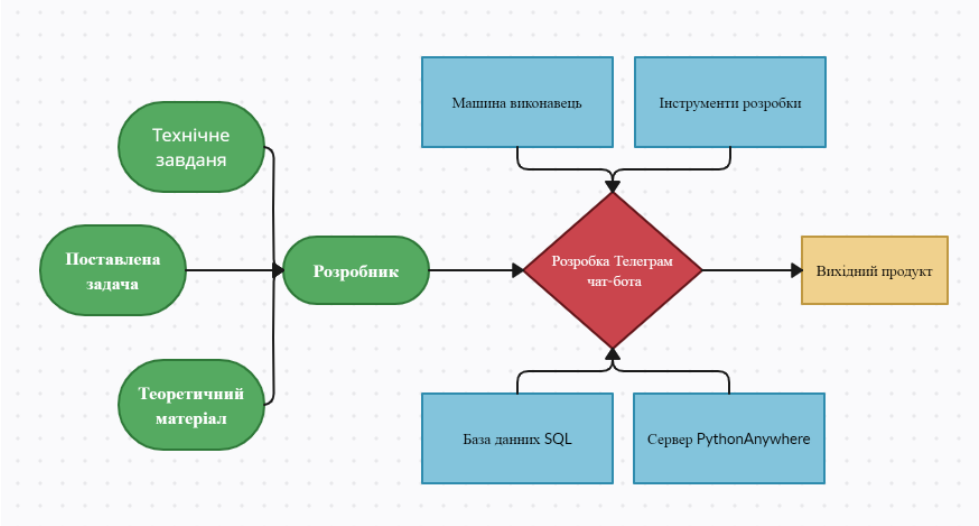

Рис. 1. Діаграма розробки чат-боту

Графічне зображення варіантів використання розкриває роль користувача та адміністратора у системі та функції, які вони можуть використовувати. Це надає можливість встановити взаємозв'язок між можливостями інтерфейсу бота та його користувачами, що є важливим етапом у процесі розробки. Аналіз цієї діаграми допомагає зрозуміти, як краще організувати навігацію та встановити необхідні обмеження. Зображення цього графіка подане на рисунку 2.

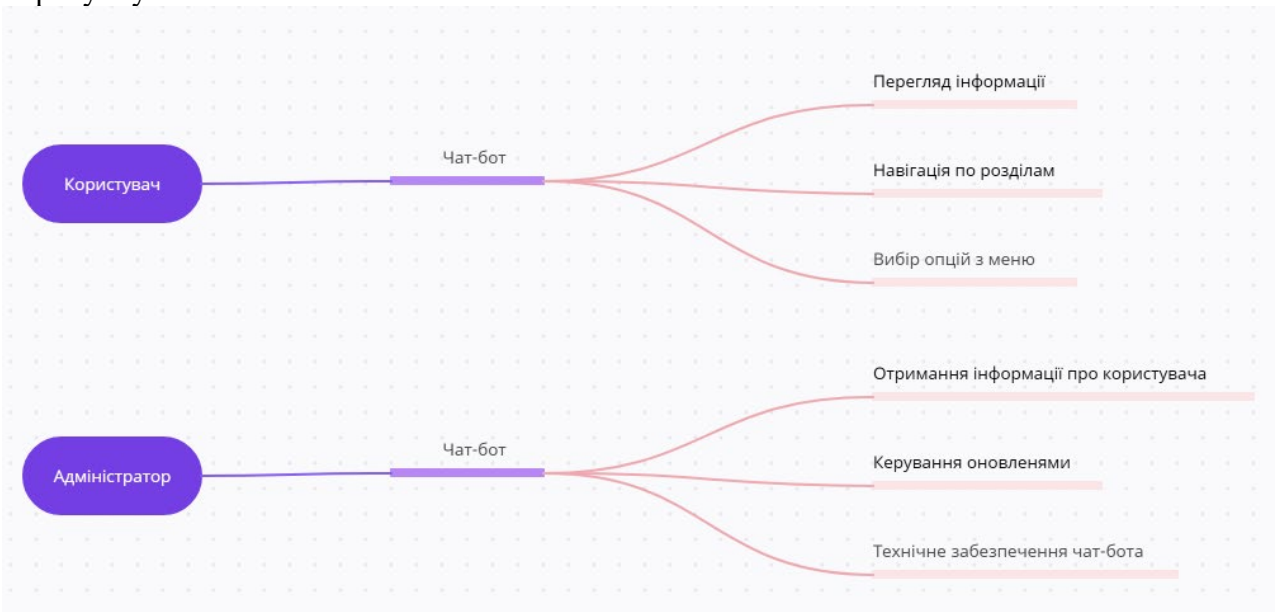

Рис. 2. Діаграма взаємодій користувача та адміністратора до чат-боту

У системі існують два види користувачів, кожен з яких має свої унікальні привілеї та доступ до функціоналу. Перший тип – це звичайний користувач, який має обмежений набір можливостей у системі. Зазвичай це доступ до основних функцій, які дозволяють взаємодіяти з системою, здійснювати запити, отримувати інформацію та виконувати базові дії.

Другий тип користувача – адміністратор. Адміністратор, крім того, що має доступ до всіх можливостей звичайного користувача, також має додаткові привілеї та функції. Наприклад,

адміністратор може виконувати адміністративні дії, такі як керування користувачами, налаштування системи, здійснення моніторингу та відстеження активності користувачів.

Однією з ключових можливостей адміністратора є отримання інформації про взаємодії користувачів в системі. Це може включати статистику використання, аналіз популярних запитів, виявлення патернів поведінки користувачів та інші дані, які допомагають у покращенні функціоналу та ефективності системи. Такий доступ до інформації дозволяє адміністраторам здійснювати інформовані рішення щодо подальшого розвитку та вдосконалення системи.

Реалізація розробки телеграм чат-боту. Після завершення відбору технологій і програмного забезпечення для реалізації проекту його потрібно зареєструвати. Засоби реєстрації ботів можуть варіюватися в залежності від обраної платформи та мети проекту. Іноді це включає в себе створення облікового запису, налаштування API ключів [13, 14], підписування угод про використання послуг, а також інші дії, необхідні для отримання доступу до інструментів створення ботів.

У пошуковій стрічці Telegram потрібно знайти @BotFather, який відповідає за керування всіма ботами у цьому месенджері [18] (рисунок 3). @BotFather є важливим інструментом для будь-якого, хто прагне створити, налаштувати чи управляти ботами у Telegram. Він надає доступ до ряду корисних функцій, включаючи створення нових ботів, налаштування їх параметрів, отримання API ключів та управління повідомленнями й командами.

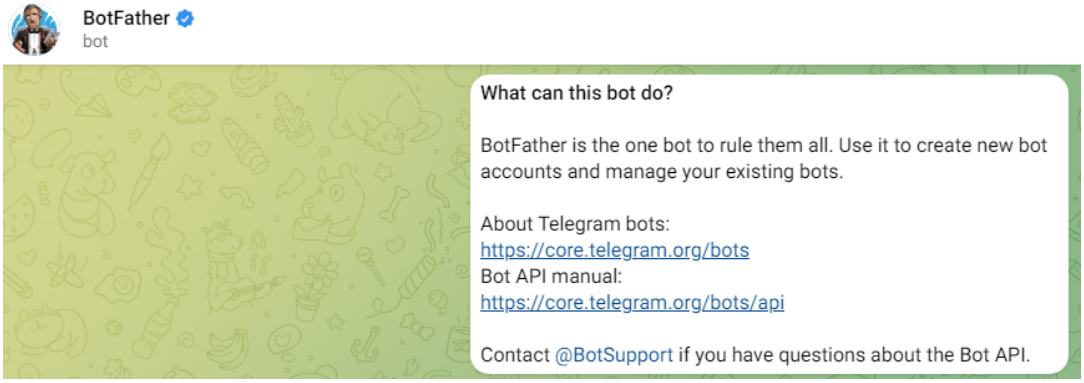

Рис. 3. Початок реєстрації бота в Telegram

Натискання кнопки "Почати" запустить діалогове вікно, в якому ми зможем взаємодіяти з @BotFather та виконати різноманітні дії, пов'язані з управлінням ботами. Наприклад створити нового бота, надати йому ім'я та налаштувати його параметри, такі як аватарка, опис тощо. Крім того, @BotFather надає доступ до командного інтерфейсу, який дозволяє налаштовувати різні аспекти бота, включаючи команди, реакції на повідомлення, доступність для користувачів тощо.

За допомогою команди /start розпочнемо діалог з ботом. Після обробки він відправить список доступних команд та функцій (рисунок 4).

Для реєстрації нашого бота нас цікавить команда /newbot. Після відправки нам запропонують вибрати назву бота, та @юзернейм. Важливо зазначити що юзернейм в телеграм враховує реєстр символів та має містити лише латинські літери, цифри та нижню риску. Ім’я обов’язково повинно закінчуватись на bot.

Після проведення данної процедури ми отримуємо унікальний token для доступу до Telegram API та доступ до посилання на нашого новоствореного бота (рисунок 5).

Після отримання посилання можемо подивитися на створеного бота. Зараз він немає функціоналу, команд та оброки інформації. Важливо зазначити що перший запуск будь-якого чат-боту в Telegram виконується через команду /start. В нашому випадку відповіді не буде, оскільки він ще не вміє відповідати на http запити. Він навчиться цьому пізніше після написання коду (рисунок 6).

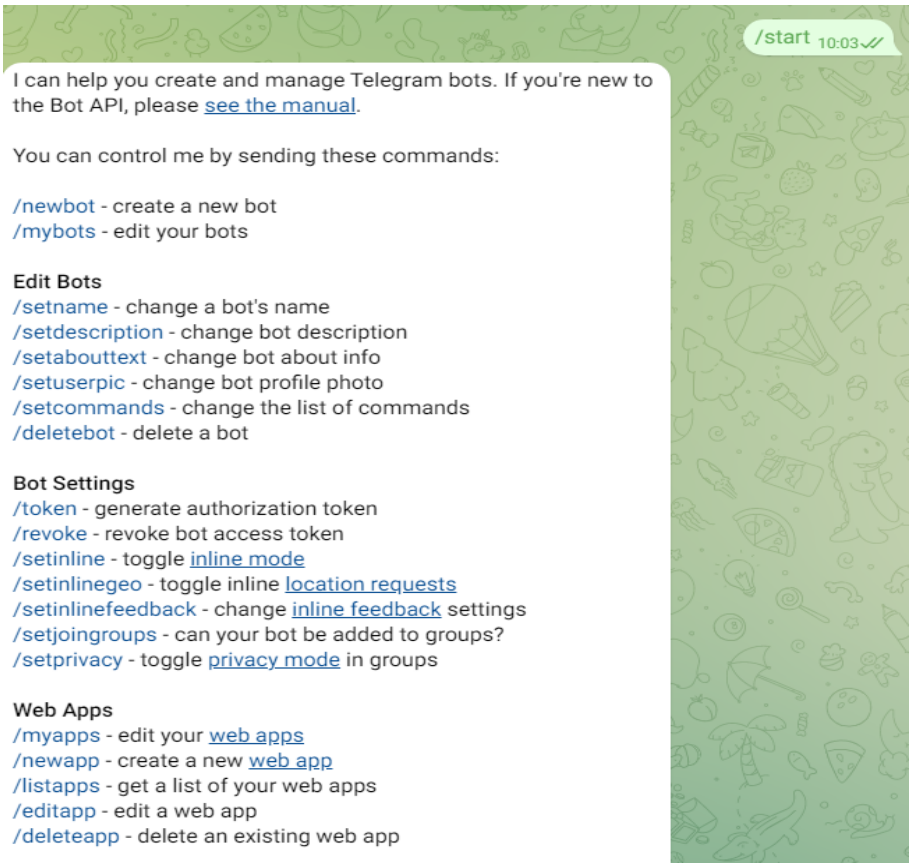

Рис. 4. Функції та команди @BotFather

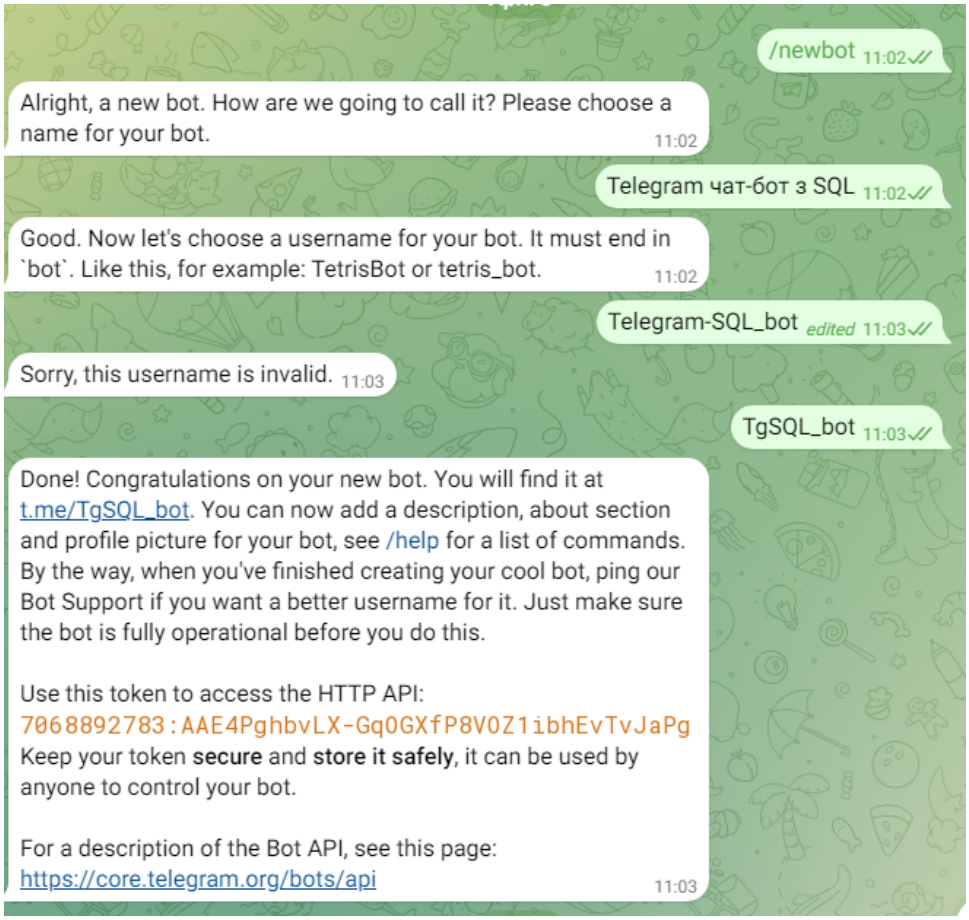

Рис. 5. Повідомлення про створення та посилання на бота

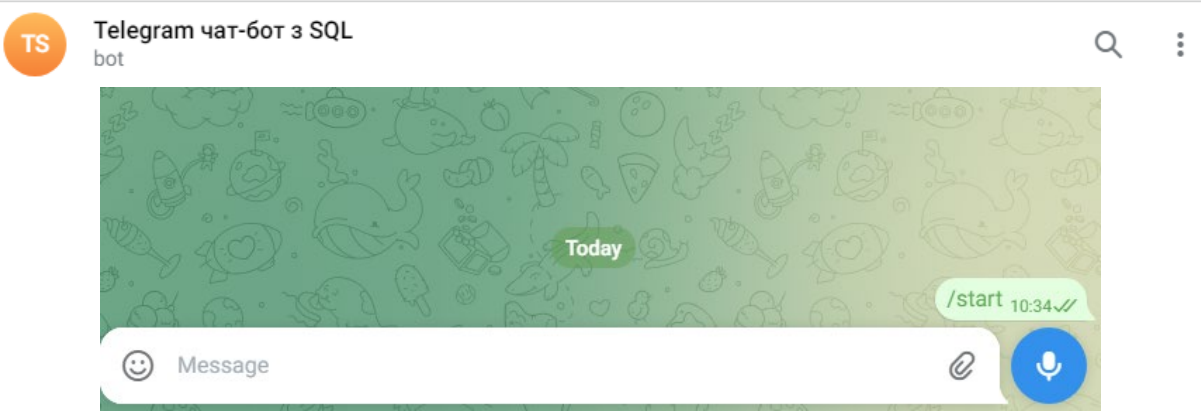

Рис. 6. Вікно чату з створеним чат-ботом

При огляді даного інформаційного блоку стає зрозумілим, що він мало відрізняється від звичайного чату з реальним користувачем. Фактично, це чатове вікно виступає як пустий аркуш для взаємодії, оскільки весь код, написаний у середовищі розробки, буде оброблюватися та виводитися у чаті бота у вигляді інформаційних вікон у текстовому форматі або у вигляді зображень та кнопок. Це є спеціальним інтерфейсом для взаємодії з програмою. Проте важливо відзначити, що всі зв'язки та шифрування не мають стосунку до протоколу шифрування Telegram. Для надсилання відповідей на запити користувача вони передаються через протокол HTTPS. Формат цих запитів має наступний вигляд: api.telegram.org/bot'token'/назва_методу [15].

Для оновлення вигляду бота варто звернутися до Bot Father і відправити команду «/setuserpic». Після цього можна надіслати в чат потрібне зображення. Після зміни графічного оформлення, повернувшись до чату з ботом, можна побачити, що зміни вже відобразилися, і у бота встановлено нове зображення профілю [18].

Звернення уваги до візуального оформлення та професійний підхід до цього аспекту можуть відігравати ключову роль у привертанні уваги користувачів. Візуально привабливий бот здатний не лише зацікавити користувачів, але й спонукати їх до початку діалогу. Таким чином, важливо приділяти увагу не лише функціональності, але й зовнішньому вигляду свого програмного помічника (рисунок 7).

Наступним етапом в розробці чат-боту є налаштування середовища розробки та підключення ресурсів для зручного програмування. Першим кроком буде встановлення останньої версії Python на комп'ютер. Після вибору актуальної версії та стабільного релізу потрібно звернути увагу на процес встановлення мови програмування. Важливо зазначити що обов'язково потрібно вибрати пункт "add python.exe to PATH". Це дозволяє запускати команди Python з будь-якої теки у командному рядку або терміналі без необхідності вказувати повний шлях до виконуваного файлу. Встановлення python.exe в PATH спрощує роботу з Python та виклик його програм з будь-якого місця в системі.

Наступним важливим аспектом роботи є налаштування середовища розробки. Оскільки ми використовуємо PyCharm важливо оптимізувати процесс роботи, для цього в середовищі створені хоткеї (HotKeys – це комбінації клавіш на клавіатурі комп'ютера, які використовуються для виклику різноманітних функцій, операцій або команд у програмах або операційних системах.).

Також корисно використовувати плагіни для простоти та легкості роботи. Плагіни які ми будемо використовувати в роботі: Tabnine – часто рятує під час введення довгих рядків, які він дописує самостійно. Він навіть може дописувати прості функції або логічні конструкції. Загалом, надзвичайно корисна річ. Звісно, є й мінус – більша частина функціоналу платна за підпискою. Але навіть безкоштовна версія буде корисною.

Rainbow Brackets – змінює колір дужок залежно від їх вкладеності. Дуже спрощує роботу та дозволяє швидше орієнтуватися в коді. Requirements – плагін для більш зручної роботи з файлом requirements.txt.

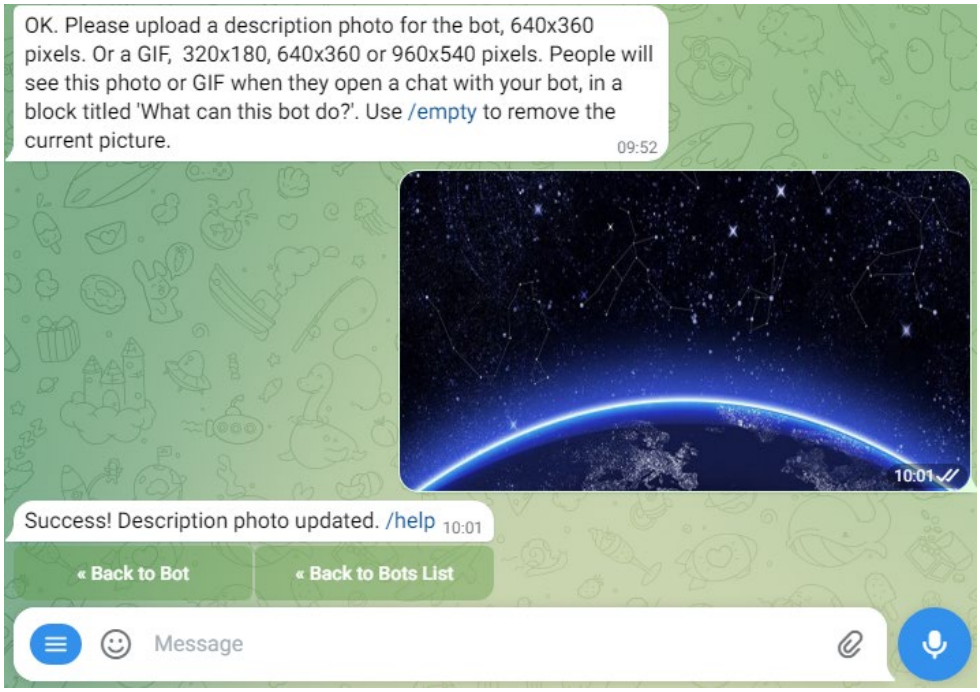

Рис. 7. Оновлення фотографії чат-боту

Наступним важливим аспектом є вибір на налаштування бази даних. Оскільки SQLite є безкоштовним дистриб'ютором то скачуємо та встановлюємо з офіційного сайту. SQLite чудово підходить для написання телеграм-ботів - це локальна база даних, яка не вимагає окремого сервера. Це означає, що не потрібно налаштовувати складну інфраструктуру для роботи з ним, що особливо зручно для невеликих проєктів, таких як телеграм-боти. SQLite працює швидко, оскільки база даних зберігається локально на диску. Це означає, що телеграм-бот може швидко звертатися до бази даних для отримання необхідної інформації під час обробки запитів користувачів. База даних не вимагає значних обсягів ресурсів для роботи. Це означає, що ваш телеграм-бот може працювати ефективно навіть на обмежених серверах або на пристроях з низькими технічними характеристиками [7].

Для початку розробки телеграм бота потрібно встановити бібліотеку Aiogram через команду PIP – це інструментарій для керування пакунками у Python [12]. Встановлювати бібліотеку SQLite немає потреби, адже вона вже інтегрована в середовище програмування PyCharm. Після повного налаштування робочого простору та додавання необхідних бібліотек, був створений головний файл у форматі ".py". У цьому файлі були імпортовані основні бібліотеки та модулі, щоб забезпечити коректну роботу їх компонентів та команд під час написання коду.

Для того, щоб бот розпочав взаємодію з користувачем у чаті та надавав йому доступ до своїх послуг, необхідно створити команду "/start". Після активації цієї команди бот автоматично відправить користувачеві меню з різними варіантами інформації або послуг, які він може обрати.

Наступним кроком було сформування загального концепту функціонування бота. Аналізуючи різноманітні варіанти оформлення інтерфейсу чату, приходимо до висновку, що додавання кнопок для зручної оброки інформації є важливим кроком. Це забезпечить якісну навігацію для користувачів бота. Однак перед початком реалізації необхідно з'ясувати різницю між різними типами кнопок у Telegram, щоб визначити найбільш підходящий варіант для використання у проекті.

ReplyKeyboardMarkup – це шаблони повідомлень. Наприклад, ваш бот ставить користувачеві запитання і пропонує варіанти відповіді. Користувач може самостійно надрукувати відповідь або натиснути на готову кнопку. Така клавіатура відображається замість основної і не прив'язана до жодного повідомлення. Такі кнопки призначені лише для

відображення тексту, що написаний на ній, і використовується переважно в звичайних чат-ботах, які здійснюють базові функції, наприклад, взаємодію з користувачем.

InlineKeyboardMarkup – це вже справжня кастомна клавіатура. З її допомогою ми можемо виконувати складніші дії. Вона прив'язується до повідомлення, з яким було відправлено. На кнопки можна закласти будь-який текст розміром від 1 до 64 байт. Інлайн кнопки дозволяють приховати в собі внутрішню телеграму посилання, посилання на зовнішній ресурс, а також шорткат для інлайн запиту.

І ту й іншу клавіатуру можна редагувати, але у різний спосіб. Перша оновлюється під час надсилання повідомлення з новою клавіатурою типу ReplyKeyboardMarkup, другу можна редагувати разом із повідомленням, до якого вона прикріплена (або лише саму розмітку) [12].

Наступним кроком після створення кнопок є інтеграція бази даних в наш чат-бот, створення логічної частини та створення дій бота на запити від користувача.

Наступним кроком є створення декоратора. Декоратор – це функція в Python, яка приймає іншу функцію або клас як аргумент, додає якісь додаткові можливості до цієї функції або класу і повертає змінену або оновлену версію. У Python декоратори зазвичай використовуються для зміни поведінки функцій або методів, не змінюючи їхнього вихідного коду. Вони дозволяють додавати функціональність до існуючих функцій або методів, роблячи їх більш гнучкими і розширюваними. Декоратори використовуються для означення спеціальних маркерів, які позначають функції або методи, які потребують додаткової обробки. Коли викликається функція або метод, обгортка декоратора виконує додаткові дії перед, під час або після виконання основної функціональності.

Так, якщо користувач введе дані без опціонального знаку, програма не допустить запис даних до бази даних та виведе помилку: "Неправильний формат. Введіть суму ще раз." Це допоможе забезпечити точність та коректність введених даних у базу даних, уникнути непорозумінь і сприяти зручності користувачів щоб вони вносили лише коректні дані в базу даних. Такий підхід зменшить ймовірність помилок та спростить процес обробки інформації (рисунок 8).

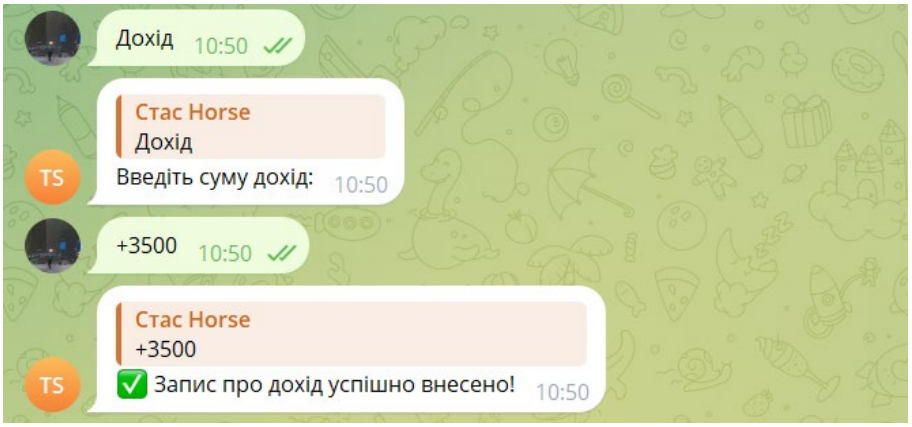

Рис. 8. Вірне введення, інформація вноситься в БД

Після успішного створення та налагодження нашого чат-боту для платформи Telegram настає час забезпечити його доступність для користувачів. Це включає в себе процес розгортання на веб-сервері, щоб чат-бот був доступний через Інтернет. Один зі способів здійснити це скористатися хмарною платформою PythonAnywhere [19, 20]. PythonAnywhere – це сервіс, який дозволяє запускати та хостити веб-додатки, написані на Python, в хмарному середовищі. Після завершення розробки чат-боту ми можемо внести його на PythonAnywhere для забезпечення його постійної доступності.

Першим кроком буде створення віртуального середовища Python на PythonAnywhere, щоб мати можливість виконати наш код. Потім ми завантажуємо файли нашого чат-боту, включаючи сам код, файли налаштувань та всі необхідні залежності.

Один з переваг PythonAnywhere полягає в тому, що він надає можливість моніторингу та керування веб-додатками, що запущені на їх серверах. Це дозволяє вчасно реагувати на будь-

які проблеми з функціонуванням чат-боту та забезпечує його надійну роботу для користувачів. Крім того, сервіс пропонує різні плани та можливості масштабування, що дозволяє збільшувати або зменшувати ресурси, необхідні для нашого чат-боту в залежності від обсягу роботи та потреб користувачів. Завдяки PythonAnywhere ми можемо ефективно розгорнути та управляти нашим Telegram чат-ботом, забезпечуючи його доступність та надійність для користувачів, що забезпечує позитивний досвід взаємодії з нашим ботом [19].

Коли ми запускаємо веб-сервіс для нашого телеграм чат-боту на PythonAnywhere, спочатку ми переходимо до пункту "Web" в панелі керування PythonAnywhere. Потім ми обираємо директорію, яка буде ініціалізована на веб-сервері. Ця директорія міститиме наш код, включаючи файли програми бота та всі необхідні ресурси. Ініціалізація цієї директорії на веб-сервісі PythonAnywhere дозволить нам запустити наш телеграм чат-бот, який буде доступний для користувачів в будь-який час. Після ініціалізації директорії ми можемо налаштувати веб-сервер для запуску нашого коду, щоб обробляти запити від користувачів і відправляти їм відповіді через телеграм API. Після цього наш чат-бот буде готовий функціонувати в будь-який час, і користувачі зможуть спілкуватися з ним через телеграм, незалежно від часу та місцезнаходження.

Бот був офіційно зареєстрований у Telegram з унікальним ідентифікатором пошуку @TgSQL_bot.

Тестування та огляд роботи програмного забезпечення. Тестування програмного додатку відбувалося в ході розробки з метою виявлення та виправлення помилок, які систематично виникали. Основний акцент був зроблений на огляді інтерфейсу, зв’язку з базою даних, перевірці працездатності чат-боту, а також визначенні швидкості реакції на запити користувачів та правильності функціонування алгоритму

Процес тестування включав:

1. Перевірка зручності та доступності інтерфейсу для користувача, впевненість у правильному відображенні елементів та їх взаємодії.
2. Виконання різних команд та взаємодія з чат-ботом для перевірки його функціональності та відповідності очікуваним результатам.
3. Вимірювання часу відправки та отримання відповіді від бота, щоб забезпечити швидку та безперебійну роботу.
4. Коректне введення інформації в базу даних SQLite.

Результат підтвердив успішність програми: головне меню з'явилося на екрані, як очікувалося. Хоча відповідь затрималася на декілька секунд, ймовірно, через обсяг коду, але це не вплинуло на коректність відображення. Все це свідчить про те, що бот працює так, як заплановано, і готовий до подальшої взаємодії (рисунок 9).

Головне меню було реалізоване у вигляді текстового блоку інформації, а також кілька кнопок для подальшої роботи з додатком. Це забезпечило зручний інтерфейс для користувачів і дозволило їм легко працювати з базою даних.

Наступний етап полягає в оцінці правильності функціонування кнопок та їх взаємодії з базою даних. Це важливий крок для забезпечення стабільності та надійності програми. Під час цього етапу було проведено перевірку наступних аспектів:

1. Перевіряється, чи виконують кнопки ті дії, які передбачені в програмі. Наприклад, якщо кнопка "Додати" призначена для додавання даних у базу, вона дійсно має виконувати цю дію.
2. Впевнюємося, що дані, які вводяться через кнопки, правильно зберігаються у базі даних. Також перевіряємо, чи можна отримати дані з бази за допомогою відповідних кнопок.
3. Перевіряється, як програма реагує на непередбачувані ситуації, наприклад, якщо відбувається конфлікт при внесенні даних.
4. Важливо переконатися, що взаємодія з базою даних через кнопки відбувається ефективно, без зайвого затримання.

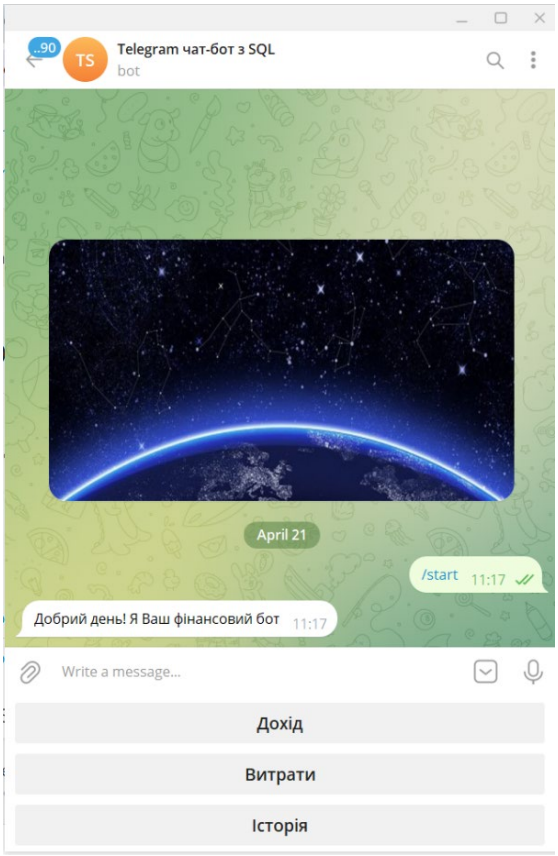

Рис. 9. Головне меню чат-боту

Зберігання даних, що вводяться через інтерфейс користувача, у базі даних є ключовою складовою для забезпечення цілісності та надійності інформації. Перевірка успішності цього процесу може бути визначена через кілька критеріїв. Перш за все, необхідно переконатися, що дані правильно відображаються в базі даних і відповідають введеним користувачем. Далі, слід переконатися, що дані зберігаються вірно та без втрати інформації. Крім того, важливо перевірити, чи виконані всі необхідні правила та обмеження бази даних, такі як унікальність ключів або відповідність типам даних. Якщо всі ці критерії виконані, можна стверджувати, що мета забезпечення цілісності та надійності даних виконана. Таким чином, можна зробити висновок, що перевірка пройдена успішно, оскільки дані зберігаються належним чином та відповідають вимогам системи.

Висновки

В статті було проведено аналіз чат-ботів, виявлено роль яку відіграють боти в сучасному світі. Досліджено соціальні мережі та мобільні додатки з інструментами для інтеграції ботів для вибору оптимального варіанту для розробки конкретного чат-боту. Сформульовано вимоги до функціоналу чат-боту.

Реалізація проекту відбулася з використанням мови програмування Python та інтегрованого середовища розробки PyCharm. У процесі розробки Telegram чат-боту було віддано перевагу використанню бази даних SQLite. Для забезпечення зв'язку з месенджером Telegram використано пакети Aiogram, Sqlite та інтерфейс зв’язку з додатком Telegram Bot API. Бот був офіційно зареєстрований у Telegram з унікальним ідентифікатором пошуку @TgSQL_bot.

Проведено тестування як у процесі розробки, так і після завершення, включаючи ручну перевірку функціонування, швидкості відповідей програми для перевірки коректності роботи бота.

В результаті вдалої реалізації проекту були досягнуті головні цілі та очікування. Чат-бот став ефективним інструментом для контроля фінансових витрат через месенджер Telegram,

забезпечивши користувачам зручний доступ до необхідної інформації та можливість взаємодії з ним у режимі реального часу.

Перспективи вдосконалення чат-боту:

- можливість розширення функціоналу бота для включення додаткових корисних опцій та можливостей;
- подальше удосконалення і оптимізація алгоритмів для підвищення швидкодії та продуктивності бота;
- планування інтеграції з іншими сервісами та програмами для розширення функціоналу та зручності використання.

Список використаної літератури:

1. Що таке чат-бот: секрети використання та основні переваги для бізнесу. HelpCrunch. URL: https://helpcrunch.com/blog/uk/shcho-take-chat-bot/.

2. Чат-бот. Переваги, засоби використання та як створити бота. Gerabot. URL: https://gerabot.com/article/detalno_pro_chatboti.

3. Месенджери довіри. Reputation Construction. URL: https://reputation.construction/mediatrust2023

4. Для 50,6% читачів основним месенджером є Telegram. Результати опитування AIN.UA. URL: https://ain.ua/2023/03/09/telegram-osnovnyj-mesendzher-opytuvannya/.

5. Lutz M. Learning Python: Powerful Object-Oriented Programming. – O'Reilly, 2025. – 1501 p.

6. Hillard D. Practices of the Python Pro. Manning, 2019. 248 p.

7. Scavetta R. J., Angelov A. Python and R for the Modern Data Scientist: The Best of Both Worlds. O'Reilly, 2021. 198 p.

8. Ernesti J., Kaiser P. Python 3: The Comprehensive Guide to Hands-On Python Programming. Rheinwerk Computing, 2022. 1078 p.

9. Gorelick M., Ozsvald I. High Performance Python: Practical Performant Programming for Humans. O'Reilly, 2020. 469 p.

10. Beaulieu A. Learning SQL: Generate, Manipulate, and Retrieve Data. O'Reilly, 2020. 380 p.

11. Кращі IDE для Python в 2023 році. Блог Mate academy. URL: https://mate.academy/blog/python/ide-for-python-2023/.

12. Smetana M. Y. How Python brings efficiency to chatbots: enhancing user experience with magic filters in Aiogram. Connectivity. 2024. Vol. 168, no. 2. URL: https://doi.org/10.31673/2412-9070.2024.025559.

13. Що таке API: навіщо використовується програмістами та базові основи роботи з ним. Академія ITSTEP. URL: https://cloud.itstep.org/blog_3/what-is-an-api-why-is-it-used-by-programmers-and-the-basics-of-working-with-it.

14. Лавренчук С., Чабан А. Дослідження зміни погодних умов за допомогою Telegram Bot API. COMPUTER-INTEGRATED TECHNOLOGIES: EDUCATION, SCIENCE, PRODUCTION. 2020. № 41. С. 46–50. URL: https://doi.org/10.36910/6775-2524-0560-2020-41-08.

15. SQLite / K. P. Gaffney et al. Proceedings of the VLDB Endowment. 2022. Vol. 15, no. 12. P. 3535–3547. URL: https://doi.org/10.14778/3554821.3554842.

16. de Quattro A. Guide to SQLite: Practical Guide. Independently Published, 2024. 118 p.

17. BotFather. Можливості, команди та функціонал. Gerabot. URL: https://gerabot.com/article/botfather_mozhlivosti_ta_funkcional.

18. Chatbot Analysis / Mr. Bhor Shubham et al. *International Journal of Advanced Research in Science, Communication and Technology*. 2022. P. 405–408. URL: https://doi.org/10.48175/ijarsct-3547.

19. Using Python on PythonAnywhere. Python for Everybody. URL: https://www.py4e.com/software-pyaw.php.

20. Python in the Cloud: Let's Explore PythonAnywhere and Other Alternatives. Codemotion. URL: https://www.codemotion.com/magazine/languages/python-in-the-cloud-lets-pythonanywhere-and-other-alternatives/.

Автори статті

Шикула Олена – доктор фізико-математичних наук, професор, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0000-0002-7385-2816

Коник Станіслав – студент, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0009-0005-9924-6864

Білоусова Світлана – кандидат фізико–математичних наук, доцент, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0000-0002-2557-8146

Прокопов Сергій - кандидат технічних наук, доцент, доцент, Державний університет інформаційно-комунікаційних технологій, Україна.

ORCID: 0009-0005-2251-9473

Саміляк Іван – аспірант, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0009-0008-2157-1263

Authors of the article

Shykula Olena – Doctor of Sciences (physics and mathematics), Professor, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0000-0002-7385-2816

Konyk Stanislav – student, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0009-0005-9924-6864

Bilousova Svitlana – Candidate of Sciences (physics and mathematics), Associate Professor, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0000-0002-2557-8146

Prokopov Serhii - Candidate of Sciences (technical), Associate Professor, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0009-0005-2251-9473

Samiliak Ivan – postgraduate, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0009-0008-2157-1263