**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ВСП «ЕКОНОМІКО-ПРАВНИЧИЙ ФАХОВИЙ КОЛЕДЖ**

**ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ»**

Циклова комісія математичних дисциплін та інформаційних технологій

**КВАЛІФІКАЦІЙНА РОБОТА**

нa тeмy: **«**РОЗРОБКА ТА ВДОСКОНАЛЕННЯ TELEGRAM-БОТА ДЛЯ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ ШОП-АКАУНТАМИ**»**

|  |  |
| --- | --- |
| Викoнaв: | здобувач освіти 4 кyрсy, грyпи К121-20  |
| Спeцiaльність | 121 Iнженерiя  |
|  | прoгрaмнoгo зaбeзпeчeння |
|  | (шифр i спeцiaльнiсть) |
|  | Артем ДЕГТЯРЕНКО  |
|  (ім’я та ПРIЗВИЩE) |
| Кeрiвник | Катерина РЕШЕВСЬКА  |
|  | (ім’я та ПРIЗВИЩE) |
|  |
| Рeцeнзeнт | завідувач кафедри програмної інженерії ЗНУ, доцент, к.ф.-м.н. Андрій ЛІСНЯК |
|  | (пoсaдa, вчeнe звaння, нayкoвий стyпiнь, ім’я та ПРIЗВИЩE) |
|  |

Запоріжжя

2024

|  |
| --- |
| ЗAПOРIЗЬКИЙ НAЦIOНAЛЬНИЙ УНIВEРСИТEТ |
|  |
| ВСП «Eкoнoмiкo-прaвничий фаховий кoлeдж ЗНУ» |
| Oсвiтньo–професійний ступінь | фаховий мoлoдший бакалавр |
| Спeцiaльнiсть | 121 Iнженерiя прoгрaмнoгo зaбeзпeчeння |
|  |  (шифр i нaзвa) |

|  |
| --- |
| **ЗAТВEРДЖУЮ**Гoлoва циклoвoї кoмiсiї математичних дисциплін та інформаційних технологій |
|  | Т.М. Смолянкова |
| (пiдпис) |  |
|  |
| “ | 14 | ” | червня | 2024 р. |

# ЗAВДAННЯНA КВАЛІФІКАЦІЙНУ РОБОТУ СТУДEНТУ

|  |
| --- |
| ДЕГТЯРЕНКО Артему Миколайовичу |

(прiзвищe, iм’я тa пo– бaтькoвi)

|  |  |
| --- | --- |
| 1. Тeмa роботи | «Розробка та вдосконалення Telegram-бота для автоматизованого  |
| управління шоп-акаунтами» |
|  |
| Кeрiвник роботи | к.т.н., доцент РЕШЕВСЬКА Катерина Сергіївна |
|  | (нayкoвий стyпiнь, вчeнe звaння, прiзвищe, iм’я, пo бaтькoвi) |
|  |
| Зaтвeрджeнi нaкaзoм ВСП EПФК ЗНУ вiд | « | 30 | » | листoпaдa | 2023 р. № | 2004-с |
|  |
| 2. Стрoк пoдaння стyдeнтoм роботи | 01.06.2024 |
|  |
| 3. Вихiднi дaнi дo роботи |  1. Пoстaнoвкa зaдaчi. |
|  2. Пeрeлiк лiтeрaтyри. |
|  |
|  |
| 4. Змiст рoзрaхyнкoвo– пoяснювaльнoї зaписки (пeрeлiк питaнь, якi пoтрiбнo рoзрoбити) |  |
|  1.Сучасні інформаційні системи |
|  2. Розробка проекту інформаційної системи |
|  3. Програмна реалізація інформаційної системи |
| 5. Пeрeлiк грaфiчнoгo мaтeрiaлy (з тoчним зaзнaчeнням oбoв’язкoвих крeслeнь) |  |
| прeзeнтaцiя дo зaхистy |
|  |

6. Кoнсyльтaнти рoздiлiв роботи

|  |  |  |
| --- | --- | --- |
| **Рoздiл** | **Прiзвищe, iнiцiaли тa пoсaдa кoнсyльтaнтa** | **Пiдпис, дaтa** |
| **Зaвдaння видaв** | **Зaвдaння прийняв** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| 7. Дaтa видaчi зaвдaння | 01.12.2023 |

**КAЛEНДAРНИЙ ПЛAН**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Нaзвa eтaпiв квалівфікаційної роботи** | **Стрoк викoнaння eтaпiв роботи** | **Примiткa** |
| 1. | Рoзрoбкa плaнy рoбoти: |  |  |
|  | Пoстaнoвкa зaдaчi | Грудень 2023 | викoнaнo |
| 2. | Збiр вихiдних дaних, oбрoбкa мeтoдичних  |  |  |
|  | тa тeoрeтичних джeрeл | Сiчень 2024 | викoнaнo |
| 3. | Рoзрoбкa пeршoгo рoздiлy:  |  |  |
|  | Огляд інформаційних систем та технологій для їхнього створення | Лютий 2024 | викoнaнo |
| 4. | Рoзрoбкa дрyгoгo рoздiлy: |  |  |
|  | Розробка проєкту інформаційної системи та вибір технологій | Квiтeнь 2024 | викoнaнo |
| 5. | Рoзрoбкa трeтьoгo рoздiлy: |  |  |
|  | Рoзрoбкa застосунку | Трaвeнь 2024 | викoнaнo |
| 6. | Oфoрмлeння i нoрмoкoнтрoль кваліфікаційної роботи тa пeрeвiркa нa плaгiaт | Червень 2024 | викoнaнo |
| 7. | Зaхист кваліфікаційної роботи | 21.06.2024 | викoнaнo |

|  |  |  |  |
| --- | --- | --- | --- |
| Здобувач освіти |  |  | Артем ДЕГТЯРЕНКО |
|  |  | (пiдпис) |  | (ім’я ПРIЗВИЩE) |
|  |  |
| Кeрiвник роботи |  |  | Катерина РЕШЕВСЬКА |
|  |  | (пiдпис) |  | (ім’я ПРIЗВИЩE) |

 **Нoрмoкoнтрoль прoйдeнo**

|  |  |  |  |
| --- | --- | --- | --- |
| Нoрмoкoнтрoлeр |  |  | Юлія БOРИСOВСЬКA |
|  |  | (пiдпис) |  | (ім’я ПРIЗВИЩE) |

# РЕФЕРАТ

Кваліфікаційна робота: 33 сторінки, 1 таблиця, 5 ілюстрації.

Об’єкт дослідження – основні методи та засоби розробки Telegram бот.

Предмет дослідження – Telegram-бот для продажу акаунтів з інтеграцією систем управління товарами.

Мета дослідження – розробка Telegram-бота на основі Python для автоматизації процесу продажу акаунтів, що дозволяє покращити користувацький досвід і спростити взаємодію з покупцями.

У кваліфікаційній роботі висвітлено теоретичні відомості про види та призначення чат-ботів. На основі цих теоретичних відомостей було розроблено Telegram-бот для продажу рекламних акаунтів, використовуючи мову програмування Python. Проєкт включає створення та розробку інтерфейсу для взаємодії з користувачами через текстові повідомлення в Telegram. Особливу увагу приділено забезпеченню безпеки даних користувачів та оптимізації процесів адміністрування акаунтів.

TELEGRAM, БОТ, API, PYTHON.

# SUMMARY

Qualification work: 33 pages, 1 table, 5 illustrations.

The object of the research is the main methods and means of Telegram bot development

The subject of the study is a Telegram bot for selling accounts with the integration of product management systems.

The purpose of the research is to develop a Python-based Telegram bot to automate the process of selling accounts, which allows to improve the user experience and simplify interaction with buyers.

The qualification project covers theoretical information about the types and purpose of chatbots. Based on this theoretical information, a Telegram bot was developed for the sale of advertising accounts using the Python programming language. The project includes the creation and development of an interface for interaction with users through text messages in Telegram. Special attention is paid to the security of user data and the optimization of account administration processes.

TELEGRAM, BOT, API, PYTHON.

**ЗМІСТ**

#

[Зaвдaння нa кваліфікаційну роботу студeнту 2](#_Toc169167324)

[Реферат 4](#_Toc169167325)

[Summary 5](#_Toc169167326)

[Вступ 7](#_Toc169167327)

[1 Сучасні інформаційні системи 9](#_Toc169167328)

[1.1 Telegram-бот сутність та функції 9](#_Toc169167329)

[1.2 Класифікація та функціональність 9](#_Toc169167330)

[1.3 Структура телеграм бота 11](#_Toc169167331)

[2 Розробка проекту інформаційної системи 12](#_Toc169167332)

[2.1 Технічне завдання 12](#_Toc169167333)

[2.2 Етапи створення інформаційної системи обліку 13](#_Toc169167334)

[2.3 Діаграма прецедентів 14](#_Toc169167335)

[2.4 Проєктування інтерфейсу 15](#_Toc169167336)

[2.5 Висновки за другим розділом 16](#_Toc169167337)

[3 Програмна реалізація інформаційної системи 17](#_Toc169167338)

[3.1 Клієнтська частина застосунку 17](#_Toc169167339)

[3.2 Адміністративна частина застосунку 22](#_Toc169167340)

[3.3 Інструкція використання Telegram-боту 30](#_Toc169167341)

[Висновки 32](#_Toc169167342)

[Перелік використаних джерел 33](#_Toc169167343)

# ВСТУП

В епоху цифрових технологій важливість віртуальної присутності для кожного бізнесу надзвичайно висока. Особливо це стосується сфери реклами, де ефективне управління акаунтами може стати ключем до успіху. Розробка телеграм боту для цієї мети вимагає ретельного аналізу та розуміння потреб клієнтів. В цьому контексті розглянуті наступні аспекти:

* аналіз цільової аудиторії;
* функціональні вимоги до боту.

Телеграм бот, крім забезпечення комфорту для клієнтів, може стати потужним інструментом для автоматизації бізнес-процесів та підвищення впізнаваності бренду.

Створення телеграм-бота для автоматизованого управління акаунтами дозволяє клієнтам отримувати інформацію про кількість товару, ціну та іншу корисну інформацію.

Створення телеграм бота у сфері реклами дозволяє не лише залучати клієнтів, але й ефективно підвищувати впізнаваність бренду, забезпечувати зручність для клієнтів та підтримувати їхню лояльність.

Метою дослідження є визначення стратегічної важливості створення телеграм бота для автоматизованого управління акаунтами в індустрії реклами. Пропонуються практичні поради щодо розробки та ефективного використання бота для залучення клієнтів і підвищення прибутковості бізнесу.

Задачі для досягнення мети:

* аналіз потреб користувачів на ринку;
* розробка функціоналом для управління товаром;
* тестування та вдосконалення бота на основі зворотнього зв’язку.

Предмет дослідження: розробка Telegram-боту з продажу рекламних акаунтів використовуючи мову програмування Python

Об’єкт дослідження: дослідження потреб цільової аудиторії, визначення функціональності бота для зручної інтеракції з користувачами, також дослідження результатів та ефективності роботи Telegram-бота, збір фідбеку від користувачів та впровадження виправлень та поліпшень на основі отриманих даних.

Отже, створення телеграм бота може значно полегшити процес купівлі акаунтів та збору відгуків від клієнтів. Завдяки цьому, цінні пропозиції та відгуки щодо покращення сервісу можна отримувати автоматично що буде дуже великим плюсом для цього бізнесу.

# 1 СУЧАСНІ ІНФОРМАЦІЙНІ СИСТЕМИ

## 1.1 Telegram-бот сутність та функції

Telegram-бот – це програма, яка функціонує в месенджері Telegram і забезпечує різноманітні можливості користувачам через автоматизовані процеси. Вони дозволяють виконувати різні завдання, від отримання інформації до здійснення покупок безпосередньо у чаті.

Телеграм боти мають різні функції та призначення, серед яких:

1. Інформаційні боти – надають користувачам інформацію на певні теми або про певні продукти, послуги чи організації.
2. Торгові боти – дозволяють користувачам здійснювати покупки безпосередньо у чаті, шляхом взаємодії з ботом.
3. Ігрові боти – надають користувачам можливість грати у різноманітні ігри прямо у Telegram, отримуючи розвагу та розвиваючи навички.
4. Сервісні боти – надають різноманітні сервіси, такі як переклад текстів, конвертація валют, отримання погодних звітів та інше.

Telegram-боти можуть бути як простими, так і складними, залежно від їхньої функціональності та потреб користувачів. Вони дозволяють автоматизувати багато процесів і полегшують взаємодію користувачів з різноманітними сервісами та інформацією прямо у месенджері [1].

## 1.2 Класифікація та функціональність

Для розуміння різноманітності та можливостей Telegram-ботів важлива їх класифікація за різними критеріями. Ось основні класифікації Telegram-ботів:

1. За призначенням.
* Інформаційні боти: надають користувачам інформацію про певні теми, продукти або послуги, включаючи новинні портали, блоги та енциклопедії.
* Торгові боти: дозволяють користувачам здійснювати покупки безпосередньо у чаті, включаючи інтернет-магазини, онлайн-аукціони та інші платформи електронної комерції.
* Розважальні боти: надають різноманітний розважальний контент, такий як відео, музика та ігри.
* Соціальні боти: сприяють взаємодії між користувачами, зокрема спілкуванню, обміну контентом та створенню спільнот.
1. За типом контенту.
* Текстові боти: надають переважно текстовий контент, такий як статті, новини та блоги.
* Мультимедійні боти: включають різні мультимедійні елементи, такі як зображення, відео та аудіо.
* Інтерактивні боти: дозволяють користувачам взаємодіяти з ботом, заповнюючи форми, голосуючи або залишаючи коментарі.
1. За функціональністю.
* Статичні боти: мають фіксований контент, який не змінюється безпосередньо з користувачем.
* Динамічні боти: можуть генерувати контент "на льоту", враховуючи запити користувачів, їхні дані та інші фактори.

Ця класифікація допомагає розуміти різноманітні можливості та функціональність телеграм ботів у месенджері Telegram [2].

## 1.3 Структура телеграм бота

Telegram-бот, подібно до веб-сайту, складається з різних компонентів, які співпрацюють для забезпечення його функціональності та ефективності. Ось як можна розглядати його структуру:

Фронтенд (інтерфейс) – це та частина Telegram-ботa, з якою користувачі безпосередньо взаємодіють. Сюди входять наступні компоненти: Інтерфейс користувача (UI): Це те, що користувач бачить та взаємодіє з ним у месенджері, включаючи кнопки, клавіатури та інші елементи дизайну [3]. Функціональність (UX): Забезпечує зручний та інтуїтивно зрозумілий досвід взаємодії користувача з ботом, включаючи логіку відповідей та обробку команд [4].

Бекенд – це та частина Telegram-ботa, яка відповідає за обробку запитів користувачів та взаємодію з базою даних. Сюди входять наступні компоненти: Серверна частина: Логіка, яка працює на сервері та обробляє запити користувачів, відправляючи відповіді через API Telegram [5].

База даних: Зберігає інформацію, необхідну для роботи бота, таку як дані користувачів, налаштування та історія дій.

Структура Telegram-ботa є складною та динамічною, але вона допомагає забезпечити ефективну роботу бота та задоволення потреб користувачів у месенджері Telegram.

# 2 РОЗРОБКА ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

## 2.1 Технічне завдання

**2.1.1 Найменування і область застосування**

Програмний продукт, що ми розробляємо, називається «Розробка та вдосконалення Telegram-бота для автоматизованого управління шоп-акаунтами». Цей бот призначений для забезпечення швидкої та зручної покупки акаунтів для рекламних цілей у соціальних мережах.

Наш бот автоматизує процес взаємодії з клієнтами, надаючи їм можливість переглядати доступні акаунти, отримувати інформацію про їх параметри та ціни, а також здійснювати покупку безпосередньо через платформу Telegram.

Ми створили цей бот з метою полегшити процес придбання акаунтів для реклами, забезпечуючи наших клієнтів надійним і зручним інструментом для їх рекламних потреб.

* + 1. **Підстава для розробки**

Цей бот розробляється з причини: на даний момент дуже швидко розвивається ринок Affiliate маркетингу і кожного дня ця сфера стає більш популярною, тому цей бот допоможе більш легше орієнтуватися при купівлі акаунтів для реклами.

Використання цього боту заощадить час користувачам при пошуку та купівлі різних акаунтів для роботи.

* + 1. **Призначення розробки**

Ціль цього бота – створення зручного та функціонального інструменту для продажу акаунтів для реклами в соціальних мережах. Бот має бути легким у використанні і надавати користувачам широкий спектр можливостей для придбання акаунтів із доступом до рекламного контенту.

**2.1.4 Технічні вимоги до програмного продукту**

Програмне забезпечення – розробка Telegram-боту з продажу рекламних акаунтів використовуючи мову програмування Python

Для забезпечення зручного доступу до акаунтів та керування ними, буде використано Telegram API.

Бот буде інтегрований у чатовий інтерфейс Telegram, забезпечуючи користувачам зручний спосіб взаємодії та здійснення покупок рекламних акаунтів

## 2.2 Етапи створення інформаційної системи обліку

Аналіз вимог – на цьому етапі потрібно детально вивчити нішу в якій мі прицюємо щоб зрозуміти які вимоги потрібні користувачу. Які функції і можливості має мати бот, щоб відповідати потребам користувачів.

Проєктування архітектури – цей бот я буду створювати за допомогою Python. Для цього я розробляю схему взаємодії компонентів системи та вибираю найкраще для реалізації кожного з них.

Розробка інтерфейсу користувача –Інтерфейс боту я розробляю зручним та легким для користування знаючи вимоги користувачів у цій сфері.

## 2.3 Діаграма прецедентів

Діаграма прецедентів – це один з типів діаграм у мові моделювання UML, який використовується для опису функціональних вимог до системи. В таблиці 2.1 можна побачити що діаграма прецедентів показує взаємодію між акторами і системою через різні прецеденти [6].

Таблиця 2.1 – Діаграма прецедентів

|  |  |
| --- | --- |
| Актор | Прецедент |
| Менеджер | Авторизація(увійти) до ботуДодавання нових акаунтівОбробка платежівВидача товару |
| Користувач | Авторизація(увійти) до ботуПерегляд списку товарівКупівля товару |

****

Рисунок 2.1 – Діаграма прецедентів

На рисунку 2.1 наведена схема взаємодії між менеджером та користувачем, які використовують бота для покупок. Вона включає акторів, а також прецеденти. Менеджер має доступ до функцій адміністрування та обробки платежів, тоді як користувач здійснює покупки.

## 2.4 Проєктування інтерфейсу

На основі технічного завдання був спроектований інтерфейс застосунку.

На рисунку 2.2 наведено меню навігації в телеграм боті. Тут знаходиться 3 кнопки, такі як: Допомога, Товари, Купити. На рисунку 2.3 при натисканні кнопки Товари зображено товар який має магазин.

При натисканні на Допомога, кожний користувач може задати любе питання будь якої складності або описати проблему якщо виникла помилка нашому менеджеру.



Рисунок 2.2 – Меню магазину



Рисунок 2.3 – Відображення товару

## 2.5 Висновки за другим розділом

Зростання популярності соціальних мереж і онлайн-реклами: вивчення ринку показало, що платформи, такі як Facebook, Google та YouTube, є основними каналами для розміщення реклами. Це свідчить про значну потребу у зручному доступі до рекламних акаунтів.

Традиційні методи продажу рекламних акаунтів можуть бути неефективними та незручними для користувачів. Розробка бота, який інтегрується з Телеграмом, має на меті полегшити цей процес, забезпечуючи швидкий та зручний доступ до купівлі акаунтів.

Інтуїтивно зрозумілі команди – розробка набору команд, які легко розуміють користувачі, допомагає забезпечити зручність взаємодії з ботом. Бот має надавати чіткі інструкції та швидкі відповіді на запити користувачів.

Отже, розробка телеграм бота для продажу акаунтів для реклами має на меті забезпечити зручний та ефективний доступ до рекламних акаунтів, автоматизуючи завдання для полегшення життя користувачів.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

## 3.1 Клієнтська частина застосунку

Клієнтська частина застосунку є інтерактивним інтерфейсом для користувачів, який надається через Telegram-бота. Користувачі можуть взаємодіяти з ботом для перегляду доступних товарів, купівлі товарів, а також отримання допомоги. Нижче детально розглянуті основні функції клієнтської частини застосунку.

**3.1.1 Привітальна частина**

Привітальна частина є першою точкою взаємодії користувача з Telegram-ботом. Вона активується командою /start і надає користувачеві необхідну інформацію для початку роботи з ботом. Крім того, ця частина відповідає за створення початкового меню, яке містить кнопки для основних дій. Це робить навігацію зручною та інтуїтивно зрозумілою.

Основна мета привітальної частини – забезпечити користувачеві приємне перше враження та надати йому необхідні інструменти для подальшої взаємодії з ботом. Вона також дозволяє боту отримати основну інформацію про користувача, таку як його ім'я.

Функціональність:

1. Привітання користувача: надсилання вітального повідомлення з ім'ям користувача.
2. Створення клавіатури: додавання кнопок для основних дій, таких як перегляд товарів, купівля товарів та отримання допомоги.
3. Відображення меню: відображення клавіатури користувачеві для зручної навігації.

def start(message):

 # Отримуємо ID користувача та його ім'я

 user\_id = message.from\_user.id

 username = message.from\_user.first\_name

 # Створюємо клавіатуру з кнопками

 markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

 products = types.KeyboardButton("🛒 Товари")

 buy = types.KeyboardButton("💲 Купити")

 help = types.KeyboardButton("💬 Допомога")

 markup.add(products, buy)

 markup.add(help)

 # Відправляємо привітальне повідомлення з клавіатурою

 bot.send\_message(

 message.chat.id,

 f"Вітаю, шановний <b>{username}</b>!\nРадий тебе бачити в нашому магазині акаунтів <b>GORILLA SHOP</b>",

 parse\_mode='HTML',

 reply\_markup=markup)

**3.1.2 Функція купівлі товару**

Функція купівлі товару в системі електронної комерції виконує декілька ключових операцій, що забезпечують коректність та надійність процесу купівлі. В цьому пункті буде розглянуто основні кроки, які виконуються під час купівлі товару, включаючи перевірку наявності товару, перевірку введеної кількості та безпосереднє виконання покупки. Нижче наведено пояснення відповідного коду.

Основні етапи функції купівлі товару:

Перевірка існування товару:

def is\_product\_exist(product\_id):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''SELECT \* FROM Product WHERE id = ?''', (product\_id,))

 product = cursor.fetchone()

 conn.close()

return product is not None

Ця функція перевіряє, чи існує товар з вказаним product\_id в базі даних. Вона підключається до бази даних G\_SHOP.db, виконує SQL-запит для отримання запису з таблиці Product з відповідним ідентифікатором та повертає результат перевірки.

Перевірка наявності необхідної кількості товару:

def check\_availability(product\_id, quantity):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''SELECT quantity FROM Product WHERE id = ?''', (product\_id,))

 available\_quantity = cursor.fetchone()[0]

 conn.close()

return available\_quantity >= quantity

Ця функція перевіряє, чи є на складі достатня кількість товару для задоволення запиту на покупку. Вона підключається до бази даних, виконує SQL-запит для отримання кількості доступного товару та порівнює її з запитуваною кількістю.

Виконання покупки:

perform\_purchase(product\_id, quantity, message)

 else:

 bot.send\_message(message.chat.id, "На складі недостатньо товару.")

 else:

 bot.send\_message(message.chat.id, "Кількість товару повинна бути більше 0.")

 except ValueError:

 bot.send\_message(message.chat.id, "Кількість товару повинна бути цілим числом.")

Основний блок коду, що виконує покупку, перевіряє умови, необхідні для успішної операції:

* Чи запитувана кількість товару більше нуля.
* Чи є достатня кількість товару на складі.

У разі невідповідності будь-якій з умов, користувачеві відправляється відповідне повідомлення:

* Якщо кількість товару менша або рівна нулю, відправляється повідомлення: "Кількість товару повинна бути більше 0."
* Якщо на складі недостатньо товару, відправляється повідомлення: "На складі недостатньо товару."
* Якщо кількість товару не є цілим числом, відправляється повідомлення: "Кількість товару повинна бути цілим числом."

Обробка винятків:

except ValueError:

bot.send\_message(message.chat.id, "Кількість товару повинна бути цілим числом.")

Цей блок коду обробляє помилки, що можуть виникнути під час введення некоректних даних, зокрема коли кількість товару не є цілим числом. У такому випадку користувачеві надсилається відповідне повідомлення.
 Таким чином, функція купівлі товару забезпечує коректність процесу покупки, перевіряючи існування товару, наявність достатньої кількості на складі та правильність введених даних.

**3.1.3 Функція допомоги**

Функція допомоги в нашому боті відповідає за обробку запитів від користувачів, які потребують підтримки або мають запитання. Дана функція складається з двох основних частин: початкова функція help, яка ініціює процес допомоги, та функція helpBot, яка передає запитання користувача в чат підтримки.

Функція help:

Ця функція виконується, коли користувач звертається за допомогою. Основні кроки, що виконує ця функція:

Збереження ідентифікатора користувача:

global help\_user\_id

help\_user\_id = message.from\_user.id

Використовуючи глобальну змінну help\_user\_id, ми зберігаємо ідентифікатор користувача, який звернувся за допомогою. Це може бути корисно для подальшого відстеження запитів або для надання контексту операторам підтримки.

Створення кнопки "Відміна":

markup = types.InlineKeyboardMarkup()

button1 = types.InlineKeyboardButton(text='Відміна', callback\_data='cancel')

markup.add(button1)

Використовуючи InlineKeyboardMarkup, ми створюємо клавіатуру з однією кнопкою "Відміна". Це дозволяє користувачу скасувати запит, якщо він передумає або зробив помилку.

Надсилання повідомлення користувачу:

msg = bot.send\_message(message.chat.id, 'Задайте питання боту.', reply\_markup=markup)

Бот надсилає повідомлення з проханням задати питання, додаючи клавіатуру з кнопкою "Відміна". Це повідомлення служить початковим етапом для збору інформації від користувача.

Таким чином, функція допомоги у нашому боті забезпечує простий і ефективний механізм для збору та перенаправлення запитів від користувачів до служби підтримки, дозволяючи користувачам легко зв'язуватися з операторами підтримки та отримувати необхідну допомогу.

## 3.2 Адміністративна частина застосунку

Адміністративна частина застосунку дозволяє керувати товарами, їх вартістю, видимістю для клієнтів, а також редагувати права адміністраторів. Ця частина доступна лише для користувачів з правами адміністратора, що перевіряються при вході в систему.

**3.2.1 Оновлення наявності товару**

Цей пункт присвячений функціоналу оновлення наявності товару в базі даних. Ця функція забезпечує можливість додавання нових товарів до бази даних магазину, а також перевірку поточних наявних товарів.

Розглянемо код функції, що відповідає за оновлення наявності товару, та розпишемо його деталі:

conn = sqlite3.connect('G\_SHOP.db') cursor = conn.cursor() cursor.execute('''INSERT INTO Product (name, price, quantity, type, visible) VALUES (?, ?, ?, ?, ?)''', (name, price, quantity, type, visible)) conn.commit() conn.close() bot.send\_message(message.chat.id, "Товар успішно доданий.") except ValueError: bot.send\_message(message.chat.id, "Видимість товару повинна бути True або False. Спробуйте ще раз.") bot.register\_next\_step\_handler(message, process\_visible\_step, name, price, quantity, type) def check\_prod(message): conn = sqlite3.connect('G\_SHOP.db') cursor = conn.cursor() cursor.execute('''SELECT \* FROM Product''') products = cursor.fetchall() conn.close()

1. Підключення до бази даних:

python

conn = sqlite3.connect('G\_SHOP.db')

cursor = conn.cursor()

Цей блок коду підключається до бази даних G\_SHOP.db за допомогою модуля sqlite3 та створює курсор для виконання SQL-запитів.

1. Обробка помилки ValueError:

python

except ValueError:

 bot.send\_message(message.chat.id, "Видимість товару повинна бути True або False. Спробуйте ще раз.")

 bot.register\_next\_step\_handler(message, process\_visible\_step, name, price, quantity, type)

У випадку виникнення помилки ValueError (наприклад, якщо поле visible має неправильне значення), відправляється повідомлення про помилку та реєструється наступний крок обробки повідомлення для повторного введення параметрів товару.

Функція перевірки наявних товарів реалізується наступним кодом:

python

def check\_prod(message):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''SELECT \* FROM Product''')

 products = cursor.fetchall()

 conn.close()

Функціонал оновлення наявності товару забезпечує можливість додавання нових товарів до бази даних та перевірку поточних наявних товарів. Він включає підключення до бази даних, виконання SQL-запитів для вставки нових даних або вибірки існуючих, обробку можливих помилок та інформування користувача про результати операцій.

**3.2.2 Оновлення вартості товару**

Цей пункт описує процес оновлення вартості товару в базі даних. Оновлення вартості може бути необхідним для відображення поточних цін на товари, коригування цін у випадку змін на ринку або оновлення інформації для акцій та знижок.

Функція для оновлення ціни товару

python

def update\_product\_price(product\_id, new\_price):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute("UPDATE Product SET price = ? WHERE id = ?", (new\_price, product\_id))

 conn.commit()

conn.close()

Підключення до бази даних:

* 1. Функція підключається до бази даних G\_SHOP.db за допомогою sqlite3.connect.
	2. Створюється об'єкт курсора для виконання SQL-запитів.

Виконання SQL-запиту:

* 1. Використовується метод execute для виконання SQL-запиту, який оновлює значення поля price в таблиці Product на нове значення new\_price для товару з вказаним product\_id.
	2. Значення new\_price і product\_id передаються як параметри для запобігання SQL-ін'єкціям.

Фіксація змін:

* 1. Викликається метод commit для збереження змін у базі даних.

Закриття підключення:

* 1. Закривається підключення до бази даних, щоб звільнити ресурси.

Функція для обробки введеної видимості

python

def process\_visibility\_step(message, product\_id):

 if message.text.lower() in ['так', 'ні']:

 new\_visibility = message.text.lower() == 'так'

 update\_product\_visibility(product\_id, new\_visibility)

 visibility\_text = 'показано' if new\_visibility else 'приховано'

 bot.send\_message(message.chat.id, f"Товар з ID {product\_id} тепер {visibility\_text}.")

 else:

 bot.send\_message(message.chat.id, "Ви ввели неправильне значення. Введіть 'Так' або 'Ні'.")

Пояснення:

Перевірка введеної видимості:

* 1. Якщо користувач ввів "Так" або "Ні", визначається нове значення видимості товару (new\_visibility).
	2. "Так" означає, що товар буде показаний (True), а "Ні" - прихований (False).

Оновлення видимості товару:

* 1. Викликається функція update\_product\_visibility, яка змінює значення видимості товару в базі даних.

Відправка повідомлення користувачу:

* 1. Користувачу відправляється підтвердження про оновлення видимості товару.

Обробка неправильного введення:

* 1. Якщо користувач ввів інше значення, відмінне від "Так" або "Ні", йому відправляється повідомлення про помилку з пропозицією ввести правильне значення.

**3.2.3 Налаштування видимості товару для клієнта**

Функціонал, пов'язаний з налаштуванням видимості товару для клієнта, полягає у двох основних аспектах: оновленні статусу видимості товару та отриманні списку видимих товарів. Нижче наведений код містить дві функції, які відповідають за ці аспекти: update\_product\_visibility та products.

Функція update\_product\_visibility

python

def update\_product\_visibility(product\_id, new\_visibility):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 print(new\_visibility)

 cursor.execute("UPDATE Product SET visible = ? WHERE id = ?", (new\_visibility, product\_id))

 conn.commit()

 conn.close()

Ця функція виконує наступні дії:

1. Встановлює з'єднання з базою даних G\_SHOP.db за допомогою sqlite3.connect.
2. Створює курсор для виконання SQL-запитів.
3. Приймає два параметри: product\_id (ідентифікатор товару) та new\_visibility (новий статус видимості товару).
4. Виводить новий статус видимості на екран для налагодження (debugging).
5. Виконує SQL-запит на оновлення поля visible у таблиці Product, встановлюючи його значення відповідно до new\_visibility для товару з ідентифікатором product\_id.
6. Зберігає зміни у базі даних за допомогою conn.commit().
7. Закриває з'єднання з базою даних.

Функція products

python

def products(message):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''SELECT \* FROM Product WHERE visible = 1 AND type = 'autoreg' AND quantity > 0 ''')

 autoreg = cursor.fetchall()

 cursor.execute('''SELECT \* FROM Product WHERE visible = 1 AND type = 'farm' AND quantity > 0 ''')

 farm = cursor.fetchall()

 conn.close()

Ця функція виконує наступні дії:

1. Встановлює з'єднання з базою даних G\_SHOP.db за допомогою sqlite3.connect.
2. Створює курсор для виконання SQL-запитів.
3. Виконує SQL-запит для вибору всіх товарів, які відповідають наступним умовам:
	1. Поле visible має значення 1 (товар видимий).
	2. Поле type має значення 'autoreg'.
	3. Поле quantity більше 0 (товар у наявності).
4. Зберігає результати запиту у змінній autoreg.
5. Виконує аналогічний SQL-запит для товарів з типом 'farm', зберігаючи результати у змінній farm.
6. Закриває з'єднання з базою даних.

**3.2.4 Редагування адміністраторів застосунку**

Редагування адміністраторів в додатку включає в себе додавання та видалення адміністраторів. У цьому розділі ми розглянемо функціональність, яка дозволяє виконувати ці операції.

Додавання адміністратора

Для додавання нового адміністратора використовується функція add\_admin. Вона приймає два параметри: user\_id та user\_firstname. Ця функція виконує наступні кроки:

1. Підключення до бази даних G\_SHOP.db.
2. Створення курсора для виконання SQL-запитів.
3. Перевірка наявності таблиці Admins та її створення у разі відсутності. Таблиця містить два поля: user\_id (цілочисловий ідентифікатор адміністратора, який є первинним ключем) та user\_firstname (текстове поле для імені адміністратора).
4. Вставка нового запису в таблицю Admins з переданими user\_id та user\_firstname.
5. Збереження змін у базі даних.
6. Закриття з'єднання з базою даних.

Код функції:

python

def add\_admin(user\_id, user\_firstname):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''CREATE TABLE IF NOT EXISTS Admins (user\_id INTEGER PRIMARY KEY, user\_firstname TEXT)''')

 cursor.execute("INSERT INTO Admins (user\_id, user\_firstname) VALUES (?, ?)", (user\_id, user\_firstname))

 conn.commit()

 conn.close()

Видалення адміністратора:

Для видалення адміністратора використовується функція remove\_admin. Вона приймає один параметр: user\_id. Ця функція виконує наступні кроки:

1. Підключення до бази даних G\_SHOP.db.
2. Створення курсора для виконання SQL-запитів.
3. Перевірка наявності таблиці Admins та її створення у разі відсутності (це може бути зайвим, якщо таблиця завжди існує до виклику функції).
4. Видалення запису з таблиці Admins, де user\_id відповідає переданому значенню.
5. Збереження змін у базі даних.
6. Закриття з'єднання з базою даних.

Код функції:

python

def remove\_admin(user\_id):

 conn = sqlite3.connect('G\_SHOP.db')

 cursor = conn.cursor()

 cursor.execute('''CREATE TABLE IF NOT EXISTS Admins (user\_id INTEGER PRIMARY KEY, user\_firstname TEXT)''')

 cursor.execute("DELETE FROM Admins WHERE user\_id = ?", (user\_id,))

 conn.commit()

 conn.close()

Ці дві функції забезпечують базову функціональність для управління адміністраторами у додатку, дозволяючи додавати нових адміністраторів та видаляти існуючих. Важливо відзначити, що для коректної роботи цих функцій база даних повинна бути доступна, а таблиця Admins повинна мати правильну структуру.

## 3.3 Інструкція використання Telegram-боту

Крок 1: Реєстрування в боті

1. Перейти у додаток Telegram.
2. Перейти за посиланням : <https://t.me/IgorillaIBot>.
3. Прописати /start для активації боту.

Крок 2 : Придбання товару.

1. Натискаємо кнопку Товари (див. рис. 3.1).



Рисунок 3.1 – Відображення товару

1. Після вибору товару, натискаємо на кнопку Купити.
2. Потрібно ввести Id товару(воно завжди зображене біля назви самого товару).
3. Далі ввордимо кількість товару (див. рис. 3.2).



Рисунок 3.2 – Придбання товару

1. Оплачуємо товар та за інструкцією у повідомленні скидаємо квитанцію менеджеру.

# ВИСНОВКИ

У кваліфікаційній роботі було реалізовано Telegram бот для продажі рекламних акаунтів.

Для ефективного створення телеграм бота було ретельно проаналізувано цільову аудиторію і визначено функціональні вимоги до бота.

При розробці цього бота була вибрана мова програмування Python. За допомогою якої було розглянуто усі важливі аспекти при розробці, які можуть знадобитися в майбутньому.

Подальший розвиток проєкту:

1. Добавити систему знижок або промокодів
2. Додавання бонусних балів за покупку товару
3. Створення магазину з бонусами

Розробка телеграм бота для автоматизованого управління акаунтами в індустрії реклами є стратегічно важливим кроком, який може значно полегшити процес купівлі акаунтів та збору відгуків від клієнтів, що в свою чергу допомагає отримувати цінні пропозиції та покращувати сервіс автоматично, надаючи великі переваги для бізнесу.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вебсайт: Telegram-бот сутність та функції. URL: https://helpcrunch.com/blog/uk/shcho-take-chat-bot/ (дата звернення: 10.03.2024).
2. Вебсайт: Класифікація та функціональність. URL: https://heroi.zapisi.cx.ua/ukraincyam/vidi-telegram-botiv-osnovni-tipi-ta-ikh-funkcional.html#google\_vignette (дата звернення:15.03.2024).
3. Вебсайт: Що таке бекенд в телеграм боті. URL: https://medium.com/wearewaes/how-to-build-a-reliable-scalable-and-cost-effective-telegram-bot-58ae2d6684b1 (дата звернення: 13.04.2024).
4. Вебсайт: Що таке UI дизайн? URL: https://te.itstep.org/blog/ui-and-ux-design (дата звернення: 19.04.2024).
5. Вебсайт: Що таке UX дизайн? URL: https://te.itstep.org/blog/ui-and-ux-design (дата звернення: 19.04.2024).
6. Вебсайт: Що таке діаграма прецедентів. URL: https://uk.wikipedia.org/wiki/%D0%94%D1%96%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%B0\_%D0%BF%D1%80%D0%B5%D1%86%D0%B5%D0%B4%D0%B5%D0%BD%D1%82%D1%96%D0%B2 (дата звернення: 12.04.2024).

**Декларація**

**академічної доброчесності**

**здобувача освіти ВСП «Економіко-правничого фахового коледжу ЗНУ»**

Я, Дегтяренко Артем Миколайович, здобувач(-ка) освіти 4 курсу, спеціальності/освітньо-професійної програми Інженерія програмного забезпечення/Розробка програмного забезпечення, групи К121-20 , адреса електронної пошти guminezy143@gmail.com

- підтверджую, що написана мною дипломна робота на тему «Розробка та вдосконалення Telegram-бота для автоматизованого управління шоп-акаунтами» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений/ознайомлена;

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

- згоден/згодна 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.

17.06.2024 Артем ДЕГТЯРЕНКО

17.06.2024 Катерина РЕШЕВСЬКА