МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
Фізико-математичний факультет
Кафедра інформатики та прикладної математики

«Допущено до захисту»
В.о. завідувача кафедри
_____ Моісеєнко Н.В.
«___» _____ 2023 р.

Реєстраційний № _____
«___» _____ 2023 р.

РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ ДЛЯ TELEGRAM

Кваліфікаційна робота студента групи І-19
ступінь вищої освіти «бакалавр»
спеціальності
014 Середня освіта (Інформатика)
Красільніка Андрія Сергійовича
Керівник **Моісеєнко Н.В.**
к. ф.-м. н., доцент

Оцінка:
Національна шкала _____

Шкала ECTS ____ Кількість балів _____

Члени комісії:

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

Кривий Ріг
2023

ЗАПЕВНЕННЯ

Я, Красільнік Андрій Сергійович, розумію і підтримую політику Криворізького державного педагогічного університету з академічної доброчесності. Запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Я не надавав і не одержував недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають покликання на відповідне джерело. Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Криворізького державного педагогічного університету ознайомлений. Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

ЗМІСТ

ВСТУП

Актуальність. Зі зростанням чисельності малого і середнього бізнесу стрімко посилюється і конкуренція, що неминуче призводить до необхідності ефективно і раціонально використовувати наявні ресурси. У цих умовах для успішного ведення бізнесу необхідно інвестувати в засоби і інструменти його розширення. Основними інструментам розвитку бізнесу є відкриття нових точок, розробка сайтів та розробка чат-ботів для месенджерів.

Чат-бот – це професійна автоматична система, багатофункціональна і легко модернізована. Метою автоматизації є підвищення ефективності реклами чи продажу послуг або товару, прискорення обслуговування і доступ до послуги в будь який час. Саме можливості автоматизації дозволяють оптимально поєднувати швидкість і якість.

Таким чином, в результаті автоматизації підприємство має можливість постійно підвищувати конкурентоспроможність та рентабельність свого бізнесу.

Об'єкт чат-боти для месенджерів.

Предмет розробка телеграм-бота «Магазин».

Мета розробити інтернет-магазин для Telegram у вигляді чат-боту.

Для досягнення поставленої мети треба ров'язати такі **задачі**:

– вивчити літературу з розробки чат-ботів для месенджера телеграм;

– проаналізувати інструменти для розробки чат-ботів для месенджера телеграм, що існують та обрати інструментарій для розробки телеграм-бота «Магазин»;

– проаналізувати основні сценарії для роботи телеграм-бота магазина та розробити алгоритми для їх реалізації;

– програмно реалізувати та протестувати телеграм бот «Магазин».

Практична цінність роботи полягає в можливості використовувати розроблений телеграм-бот «Магазин» як за прямим призначенням, так і в якості навчального матеріалу в курсі «Програмування».

Структура та обсяг роботи: робота складається із вступу, двох розділів, висновків та списку використаних джерел з 19 найменувань; містить 42 сторінки тексту, 15 рисунків. Загальний обсяг 44 сторінки.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ БОТІВ ДЛЯ МЕСЕНДЖЕРІВ

1.1. Телеграм-боти та їх можливості

Слово «бот» пішло від чеського слова robot, яке скоротили англійською до bot. Це деяка утиліта, яка розроблена людиною для виконання чітких дій по заданому алгоритму. Слово «chat» з англійської має переклад як «співбесіда» або «розмова».

Чат-бот – віртуальний співрозмовник, що прискорює процес обробки та надання інформації користувачам. Чат-бот може виконувати типові функції: задавати і відповідати на питання, перенаправляти користувача на різні джерела інформації або контакти.

Є чат-боти технічної підтримки, що призвані допомогти користувачу:

- відповісти на питання;
- перенаправити на помічника тех. підримки;
- зібрати інформацію корисну для власників чат-бота.

Ідея чат ботів зародилася ще в 1950 році в Алана Тюрінга. Британський вчений виразив думку, що комп’ютерна програма може думати та спілкуватися як людина. Тюрінг вважав, що комп’ютерні програми до 2020 року з легкістю розвинуться настільки, що їх буде важко відрізнити від людини.

У 1966 році професор Джозеф Вейценбаум створив програму Eliza. І хоча таке примітивне «спілкування» важко було назвати бесідою, люди були у захваті!

Вважається, що це перший чат-бот в історії комп’ютерних наук, розроблений Джозефом Вайзенбаумом з Массачусетського технологічного інституту (MIT). У 1994 році було введено термін «чаттербот». Eliza працює, розпізнаючи ключові слова або фрази з вхідних даних, щоб відтворити відповідь, використовуючи ці ключові слова з попередньо запрограмованих відповідей. Вона постійно відповідала на репліки співрозмовника зустрічними питаннями.

Наприклад, якщо людина скаже, що «моя мама добре готує». Eliza вибирала слово «мама» і відповідала, ставлячи відкрите запитання «Розкажи мені більше про свою сім'ю». Це створювало ілюзію розуміння та взаємодії з реальною людиною, хоча процес був механізованим.

У 1995 році Річард Воллес розробив бот ALICE. На відміну від Eliza, чат-бот ALICE міг використовувати обробку природної мови, що дозволяло вести більш витончену розмову. Однак це було революційно, тому що він був відкритим кодом. Розробники можуть використовувати AIML (мову розмітки штучного інтелекту) для створення власних чат-ботів на базі ALICE.

Jabberwacky — чат-бот, створений британським програмістом Ролло Карпентером. Його заявлена мета — «імітувати природне людське спілкування в цікавій, розважальній та жартівливій манері». Це рання спроба створити штучний інтелект за допомогою взаємодії людей. Заявленою метою проекту було створення штучного інтелекту, здатного пройти тест Тюрінга. Він створений для імітації людської взаємодії та спілкування з користувачами. Він не призначений для виконання будь-яких інших функцій. На відміну від більш традиційних програм штучного інтелекту, технологія навчання призначена як форма розваги, а не для використання в системах комп'ютерної підтримки чи корпоративному представництві.

Останні розробки дійсно дозволяють використовувати більш сценарний, контрольований підхід, щоб сидіти на вершині загального розмовного ШІ, з метою об'єднати найкраще з обох підходів, а також використання в сферах продажів і маркетингу. Кінцевий намір полягає в тому, щоб програма перейшла від текстової системи до повністю керованої голосом системи – навчання безпосередньо за звуком та іншими сенсорними введеннями. Його творець вважає, що його можна вставити в об'єкти вдома, такі як роботи чи домашні тварини, що розмовляють, маючи на меті бути корисним і розважальним, складаючи компанію людям.

В 2005 році Стів Уорсвік створює Mitsuku – чат-бот на основі технології AIML. Mitsuku стверджує, що вона 18-річна жінка з Лідса (Англія). Він містить

усі файли AIML ALICE з багатьма доповненнями з розмов, створених користувачем, і над ним постійно триває робота. Її інтелект включає в себе здатність міркувати. Наприклад, якщо хтось запитує «Чи можна з'їсти будинок?», Mitsuku шукає властивості «будинок». Знаходить, що значення «made_from» встановлено як «brick» і відповідає «ні», оскільки будинок не їстівний. Вона може грати в ігри та робити фокуси за бажанням користувача. У 2015 році вона розмовляла в середньому 250 тис. разів на день.

У 2006 році IBM почала розробку суперкомп'ютера Watson, який міг давати відповіді на питання уголос. А через 4 роки після цього компанія Apple представить Siri (Speech Interpretation and Recognition Interface) – голосового асистента.

Термін «чат-бот» виник у середині 90-х рр. Його вигадав розробник Майкл Маулдін. Колись, у широковідомому месенжері ICQ були популярними боти, які перекладали тексти, виконували розрахунки, розсилали жарти або інформували про прогноз погоди.

Помічники - боти, що призначені для полегшення виконання різних задач: пошуку інформації, користування комп'ютером або обробки даних. У свою чергу, співбесідники - боти для спілкування, розваг тощо.

Нині чат-боти використовуються майже всюди: в інтернет-магазинах, ресторанах та різних сервісах. Їх розміщують на сайтах або створюють на основі соціальної мережі. А в їх складанні допомагають різні мови програмування та бібліотеки для них, що полегшують написання ботів.

Типовий чат-бот - це набір алгоритмів, що реагують на дії користувача.

Чат-боти набули широкої популярності серед численних активних користувачів месенджерів, оскільки мають різні сфери використання – від розважального контенту, включаючи покрокові ігри та збір бонусних балів у ресторанах, до дотримання дієтичного плану, відстеження доставки та навіть здійснення платежів. за різні послуги.Телеграм бот - це інтерфейс взаємодії між користувачем та тим, що у нього всередині, і це може бути все, що завгодно. В офіційній документації наведені такі приклади:

– надсилати повідомлення та новини;

– приймати платежі від користувачів (оплатити підписку, послугу або товар);

– інтеграція з іншгими сервісами. Наприклад, бот може відправляти комментарі, керувати «розумним будинком», відправляти вам повідомлення при настанні якоїсь події;

– утіліти та інструменти (можуть відображати погоду, перекладати тексти або попереджати про події за замовленням);

– може бути партнером в іграх: пограти з вами в шашки або шахи, взяти участь у вікторинах та т. ін.;

– соціальні сервіси (наприклад, може знаходити вам співрозмовника, грунтуючись на ваших спільних інтересах та зацікавленостях).

Розглянемо чому чат-боти для месенджерів зручніші за додаток.

1) Мінімалістичний і простий дизайн.

Порівняно з численними додатками з різним дизайном, коли вам потрібно запам’ятати, де і що натискати, боти більш універсальні та прості; вони пропонують просте спілкування за допомогою текстів.

Таким чином, ніхто не вимагав від ботів барвистий дизайн. Однак з весни 2022 року Telegram надав розробникам ботів інструменти для створення нескінченно гнучких інтерфейсів за допомогою JavaScript. Тепер, якщо у вас є досвід створення ботів для Telegram, знання Python і бажання, ви можете створити повноцінну заміну звичайному сайту.

2) Бот має мінімум реклами та орієнтований на потреби користувачів.

Вам не потрібно встановлювати сотні програм для кожної служби, якщо ви можете отримати всю необхідну допомогу від бота. Це особливо корисно для ресторанів і магазинів. Клієнти рідко прагнуть встановлювати програми з купи місць, які вони відвідали. Через це власники бізнесу пропускають відгуки клієнтів і втрачають спілкування з ними. Якби в кожному з цих місць був свій бот, доступний у різних месенджерах, це було б зручніше та дружніше до користувачів. Ніхто не любить заповнювати пам’ять свого телефону

непотрібними програмами, які будуть використані один або два рази. Однак клієнти повинні взаємодіяти з власниками послуг, і вони оцінять це через свій улюблений месенджер.

3) Відсутність необхідності реєстрації, авторизації та постійного повторного входу

Під час налаштування бота Telegram для автоматизації авторизації можна використовувати Python та його бібліотеки. Це дозволяє користувачеві пройти авторизацію лише один раз, коли бот додається до чату. Клієнт може використовувати бота скільки завгодно, а коли в ньому більше немає потреби, користувач може просто заблокувати бота.

Не потрібно запам'ятовувати використовувані паролі або логіни. Додавання бота на сайт або в додаток збільшує аудиторію користувачів, оскільки це робить спілкування з клієнтами та надання їм допомоги набагато простішим та зручнішим.

Популярність програми також відіграє свою роль. В 2022 році Telegram оголосив про понад 700 мільйонів активних користувачів щомісяця. «Якщо тебе немає в інтернеті – тебе не існує», цей вираз справедливий в наш час тим більш справедливий для бізнесу, адже інтернет це чудове місце для реклами та продажу свого товару чи послуги.

Всі обирають різні шляхи появи в інтернеті. Бізнес може:

- створити аккаунт у соц мережі
- замовити власний сайт
- замовити розробку додатку
- замовити створення телеграм бота

Усі варіанти є повноцінними окремими інструментами для розвитку бізнесу, вони можуть використовуватись як допоміжні, для реклами чи підтримки, або ж як основні, для замовлення або оплати продукту чи послуги.

Кожен інструмент відрізняється в ціні, адже в кожного з них своя складність й основні потреби у функціоналі. Найдорожчим з варіантів буде створення додатку. Найдовшим за часом – створення аккаунту у соц мережі. До

сайту сладніше всього звернути увагу людини серед мільйонів однакових за функціоналом та зовнішнім виглядом сайтів конкурентів.

Створення телеграм бота є альтернативою серед усіх вище перерахованих варіантів, адже це відносно дешево, легко в освоєні та не потребує особливих навичок в дизайні. Також телеграм бот може виконувати будь яку роль в бізнесі, починаючи від продажу товару або послуги й закінчуючи веденням статистики та бухгалтерії.

1.2. Принципи роботи ботів

Коли користувач взаємодіє з ботом через додаток Telegram (пише повідомлення боту, натискає кнопку on-line під повідомленням або вибирає команду в меню) - додаток відправляє запит на сервер Telegram. Сервер розуміє, що це запит до бота і в залежності від того, як налаштований бот, або відправляє запит до бота, або чекає запиту від бота.

Існує дві основні технології, за допомогою яких бот може отримати те, що він призначений для отримання від сервера. Довге опитування і Webhook. Суть тривалого опитування полягає в тому, щоб постійно опитувати сервер на наявність оновлень (оновлень), призначених для бота, на кшталт: «Там, там, що для мене?». Вебхук - це коли сам сервер «стукає» до бота і каже: «У тебе щось є!», коли щось приходить для бота. Зараз ми не будемо орієнтуватися на ці технології, вони будуть деталізовані пізніше в курсі. Довгі опитування в основному використовуються при розробці і тестуванні бота, часто на власному комп'ютері розробника, і вебхук, коли бот вже знаходиться у виробництві.

Бот отримує оновлення від сервера, щось робить з ними і відправляє результат назад на сервер. Сервер розуміє, для якого користувача призначене повідомлення, і пересилає його в додаток користувача.

Зв'язок між ботом і сервером відбувається у вигляді http-запитів.

@ BotFather надає ряд варіантів управління нашими ботами (рис. 1.1).

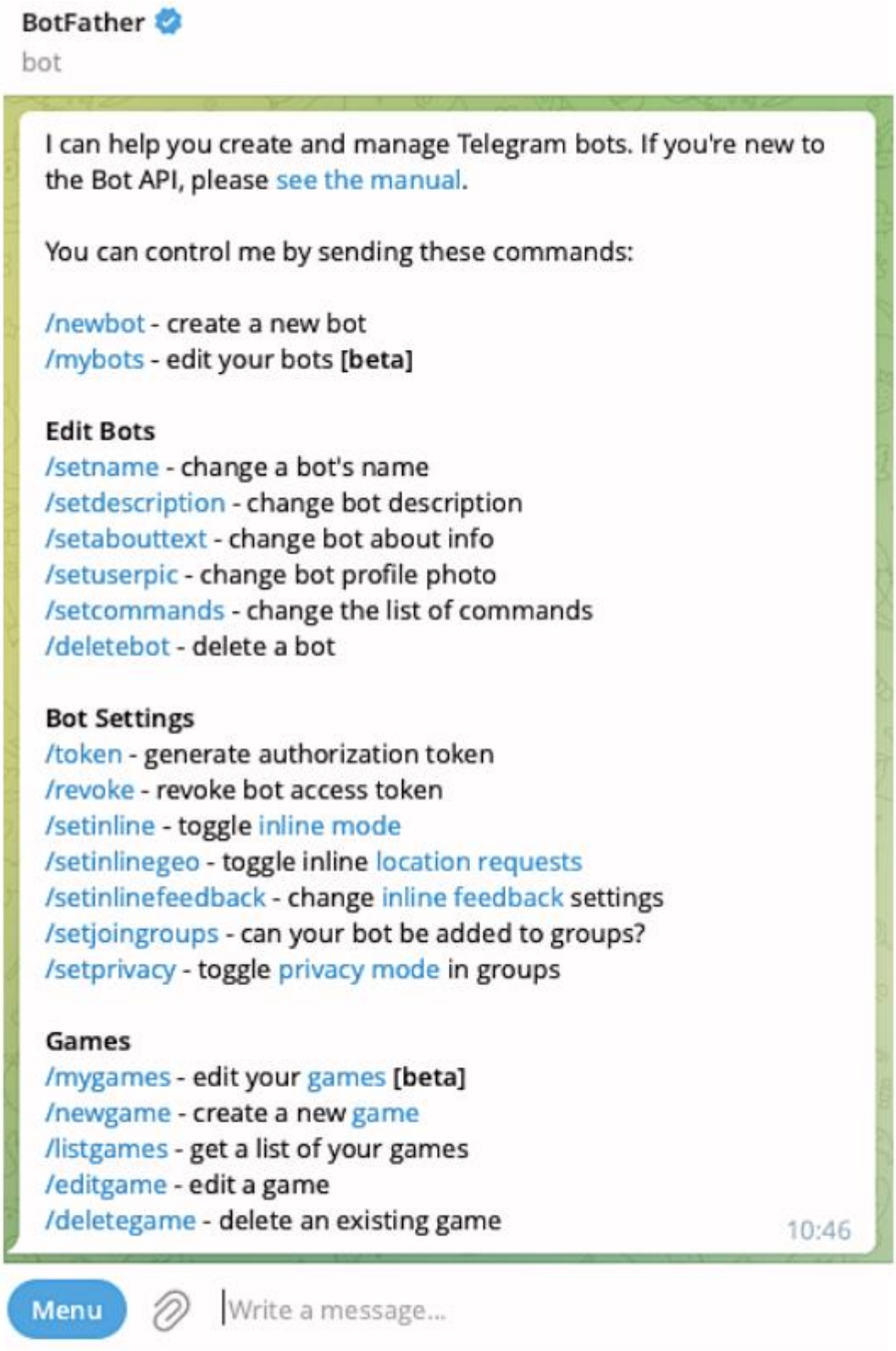


Рис. 1.1. Налаштування телеграм-бота

1.3. Класифікація ботів

Щороку надходить 265 мільярдів запитів на підтримку клієнтів, а їх обслуговування коштує компаніям 1,3 трильйона доларів. Як наслідок, фінанси компанії та її агенти з обслуговування клієнтів відчувають величезний тиск — однак до 80% запитів на підтримку можна обслуговувати без людської участі.

Кажуть, що за допомогою самообслуговування та автоматизації ви можете зменшити витрати на обслуговування клієнтів на 30% , потенційно звільнивши персонал служби підтримки для вирішення складних запитів.

За оцінками Grand View Research , до 2025 року розмір ринку чат-ботів досягне 1,25 мільярда доларів . 57% споживачів у Великій Британії знають, що таке чат-бот, і, що важливо, 35% кажуть, що хочуть, щоб їх використовувало більше компаній.

Однак шлях до успіху чат-бота далеко не простий. Розпочати свій перший проект чат-бота з метою створити ідеальний інструмент із підтримкою штучного інтелекту може бути нездійсненним завданням для будь-якого бізнесу.

Розглянемо основні типи чат-ботів.

1. Кнопкові боти

Мета кнопкового бота полягає в тому, щоб провести користувача через заздалегідь визначене дерево сценаріїв. Цей тип бота нагадує текстову систему IVR. Як і IVR, кнопковий бот пропонує вибір і запитує дані. Зазвичай там написано: «Я бот, який може допомогти вам у таких питаннях, будь ласка, натисніть відповідну кнопку: X, Y, Z…»

Створення цього типу бота не потребує жодних можливостей ШІ від платформи розробки та займає мінімум часу та ресурсів (наприклад, від одного до кількох днів для простого сценарію). Зазвичай таких ботів порівняно легко створити, вони вимагають мінімальної професійної компетенції та можуть бути розроблені в деяких інтуїтивно зрозумілих графічних конструкторах ботів, таких як Aimylogic чи інші. Цього бота можна опублікувати в будь-якому зі звичайних каналів — як веб-віджет, у додатках для обміну повідомленнями, Alexa чи Google Assistant.

Бот на основі кнопки ідеально підходить для адаптації, опитувань, підтримки продажів і практично для будь-якого простого завдання автоматизації процесів, де сценарії спілкування чітко визначені.

Однак після кількох тижнів або місяців роботи компанія може захотіти охопити більше сценаріїв, так що простого кнопкового бота буде недостатньо. Тоді наступним кроком у континуумі еволюції ботів є гібридний бот.

2. Гібридний бот

Гібридний бот — це кнопковий бот із можливістю поставити запитання природною мовою. Зазвичай він говорить: «Я бот, який може допомогти вам у таких питаннях; будь ласка, натисніть відповідну кнопку X, Y, Z або введіть своє запитання в полі нижче ».

Часто NLU (розуміння природної мови) у цих ботів дублює кнопки. Гібридний бот є перехідним етапом між кнопковим та ботом AI/NLU.

Гібридний бот підходить для наступних завдань:

Як і кнопковий бот, він допомагає користувачам знаходити відповіді на найпоширеніші запитання.

Він реєструє теми, не охоплені початковим сценарієм бота: які проблеми хвилюють користувачів і якими словами їх описують. Це форма дослідження клієнтів, яка допомагає зрозуміти, наприклад, які питання слід включити в FAQ або зобразити на веб-сайті, і які слова для цього слід використовувати.

Якщо компанія не має журналів історії розмов, але хоче створити бота зі штучним інтелектом у майбутньому, то гібридний бот може бути одним із прийнятних способів збору контекстних даних протягом певного часу для навчання бота. Однак краще було б створювати логи з нуля, записуючи розмову оператора з відвідувачем.

Бот може існувати на гібридній стадії досить довго, поки розробники збагачують його механізм штучного інтелекту кращим розумінням відповідного контексту.

Вартість початкового створення не набагато вища, ніж для бота на основі кнопки, однак цей тип бота зазвичай вимагає більш просунутої платформи

розробки з потужним ядром AI/NLU. Інакше обробка запитів на природній мові була б практично неможливою. Крім того, поточні витрати на проект вищі, оскільки гібридний бот потребує постійного оновлення та навчання для переходу до повнофункціонального рішення ШІ.

3. AI бот

Компанії слід розглянути можливість впровадження бота зі штучним інтелектом, якщо виконано одну або кілька умов:

Якщо компанія отримує кілька сотень або більше запитів на день.

Штат технічної підтримки або підтримки продажів компанії перевищує 20 осіб.

Компанія хоче підтримувати більше каналів зв’язку (месенджери, веб-віджети тощо) без додавання персоналу підтримки.

Більшість розмов носять цілеспрямований і структурований характер.

Бувають пікові періоди зв’язку, коли багато клієнтів звертаються до компанії одночасно й дуже незадоволені затримкою відповіді.

Коли велика частина клієнтів потребує допомоги та підтримки в неробочий час.

ШІ-бот може автоматизувати складне одночасне спілкування з декількома користувачами, охоплюючи багато слабо пов’язаних тем. Наприклад, чат-бот оператора зв’язку може проконсультуватися щодо тарифів і варіантів підвищення, а також звітувати про поточну статистику використання клієнта. Бот підтримки роздрібних продажів надає консультації щодо різних категорій товарів, умов покупки та доставки. Бот технічної підтримки проводить початкову діагностику та пропонує допомогу.

Прості та жорсткі сценарії ботів на основі кнопок не можуть автоматизувати всі ці складні завдання — справді потрібна гнучкість природної мови, щоб охопити широкий спектр намірів користувачів, збирати необхідні дані та надавати підтримку.

Проект ботів зі штучним інтелектом зазвичай вимагає значних початкових інвестицій для створення бота (зазвичай понад 7000-20 000 фунтів стерлінгів).

Крім того, компанія повинна інвестувати в підписку на платформу розробки та постійну підтримку вдосконалення бота.

Проект бота AI має керуватися та впроваджуватися командою професіоналів із сильною компетенцією розмовного AI.

Незважаючи на більш помітні інвестиції, впровадження бота ШІ приносить матеріальні вигоди, зокрема:

– скорочення загальних витрат на підтримку клієнтів

– підвищення якості, своєчасності, швидкості комунікацій з клієнтами

– покращена мотивація персоналу підтримки, якому не потрібно мати справу з повторюваними та буденними запитами.

Один із найвідоміших ботів — Джулі з Amtrack — допомагає користувачам знайти потяг і забронювати квитки, ведучи людську розмову. Такі боти також цінуються користувачами: Amtrak заявляє, що запуск Джулі призвів до збільшення кількості бронювань на 25% і заощадив 1 мільйон доларів на обслуговуванні клієнтів.

Проект ботів зі штучним інтелектом зазвичай вимагає значних початкових інвестицій для створення бота — зазвичай понад 7000-20 000 фунтів стерлінгів.

4. Багатоканальна екосистема, включаючи ботів ШІ

Навіть коли амбітна мета створення бота зі штучним інтелектом досягнута, ще занадто рано влаштовуватися. Ті, хто не може жити без виклику, починають наступний проект зі створення досвіду багатоканального спілкування для своїх клієнтів. У багатоканальній парадигмі розмова, розпочата, наприклад, у Twitter, може продовжуватися за допомогою чат-бота, за якою слідує телефонна розмова з агентом технічної підтримки та завершується електронною поштою.

Досягнення безперебійної роботи всіх систем, чіткої та повної передачі даних з каналів і між ними – це інший рівень складності, який вимагає дуже сильної компетенції та значних ресурсів. Замість того, щоб бути окремим рішенням, бот AI стає невід’ємною частиною диверсифікованої комунікаційної екосистеми.

Багато клієнтів звикли спілкуватися з компаніями за допомогою кількох каналів — соціальних мереж, телефону, електронної пошти, веб-чат-бота. І коли перемикання між каналами відбувається в рамках однієї розмови, вони очікують, що вся історія буде доступна в наступному каналі. В іншому випадку вони дратуються, представляючись і пояснюючи проблему знову і знову.

5. Голосові

До кінця 2019 року глобальна кількість власників розумних колонок перевищить 200 мільйонів зі 114 мільйонів, при цьому у Великобританії цей показник зросте на 46%. Це вибухове зростання створює новий канал зв'язку, який об'єднує компанії та клієнтів. Для використання цього каналу найбільш просунуті компанії (зазвичай B2C) створюють так звані «навички» (голосові боти для Alexa або Google Assistant).

За останні роки мовні та лінгвістичні технології прогресували до того моменту, коли стало можливим створення добре функціонуючих складних голосових рішень, наприклад, голосових помічників, інтелектуальних IVR або автоматичних викликів.

Завдяки використанню найбільш природного інтерфейсу — голосу — ці рішення можуть зацікавити користувачів і принести багато переваг компаніям. Однак розробка рішень на основі голосового зв'язку потребує спеціальних знань у галузі від архітектора, лінгвіста, UX-дизайнера, розробника та низки інших професій. Для ефективної роботи кожному члену команди потрібні спеціальні інструменти. Існує небагато платформ розробки голосових рішень, які підтримують весь цикл розробки, наприклад, DialogFlow від Google, SAP Conversational AI, JAICP від Just AI.

Багато експертів ринку вважають, що в найближчому майбутньому голосові рішення стануть звичайною та необхідною частиною всієї IT-екосистеми — так само, як мобільний зв'язок останніми роками. Тому перспективним компаніям слід починати перші експерименти з голосом вже зараз, накопичувати досвід і перевіряти гіпотези.

Чат-боти, як і будь-який правильно розроблений проект автоматизації, можуть принести величезну користь бізнесу — підвищити швидкість і якість обслуговування, заощадити кошти, збільшити дохід і зміцнити бренд.

Проведення перших експериментів із чат-ботами не потребує величезного бюджету, часу чи ресурсів — компанія може «випробувати воду» за допомогою крихітного чат-бота на основі кнопки дешево та легко. Після перевірки початкових гіпотез компанія може розвивати цей інструмент далі, поступово додаючи ресурси, підтримуючи нові варіанти використання, розвиваючи можливості.

Корпоративний чат-бот – це не якесь жорстке рішення, розроблене один раз і незмінне. Як корпоративний сайт чи мобільний додаток — це радше подорож, ніж пункт призначення. І як у справжній подорожі — успіх проекту багато в чому залежить від ваших партнерів, їх компетенції, потужності їхніх технологій та знань галузі.

Висновки до розділу 1

Телеграм-бот «магазин» – це багатофункціональна система з великими можливостями.

Вона може економити час, робочі місця та гроші свого власника за рахунок свого малого споживання часу на обслуговування. Вірно налаштована система магазину не буде потребувати довгої професійної підтримки.

При необхідності можна легко додати необхідні можливості чи видалити зайві.

Основною задачею проекту було навчитися створювати Telegram бота для просування малого та середнього бізнесу, а також опанувати мови програмування **Python** та бібліотеки **AIOGram**. Як основу була взята тема “Розробка телеграм - бота магазин”, адже ця тема включає в себе використання більшості функціоналу бібліотеки **AIOGram,** яка використовується в мові програмування **Python.**

Ця робота направлена на створення телеграм - бота «магазин», що можна буде адаптувати під свої потреби внесши невеликі зміни.

Цей телеграм бот буде підтримувати такі можливості:

- Обирання товару серед запропонованих
- Додавання товару до корзини покупок
- Сплата за товари, що знаходяться в коризині покупок
- Розсилка повідомлень

Для підтримки цих можливостей до проекту був доданий функционал роботи з базою данних за допомогою **SQL.**

РОЗДІЛ 2
РЕАЛІЗАЦІЯ ТЕЛЕГРАМ-БОТА «МАГАЗИН»

2.1 Інструменти розробки

Для написання телеграм-боту використовувалась мова програмування Python, дуже популярна та доволі легка у вивчені мова програмування.

Бібліотека Python – збірка модулів, об'єктів, підпрограм для вирішення близьких за тематикою задач засобами мови Python.

Для Python існує багато бібліотек для створення ботів. Розглянемо декілька найпопулярніших з них.

pyTelegramBotAPI (рис.2.1) – одна з простих бібліотек яка має зрозумілий синтаксис та проста у використанні. Підходить для нескладних телеграм ботів, має зрозумілу та обширну документацію. Має версію для асинхронного програмування.

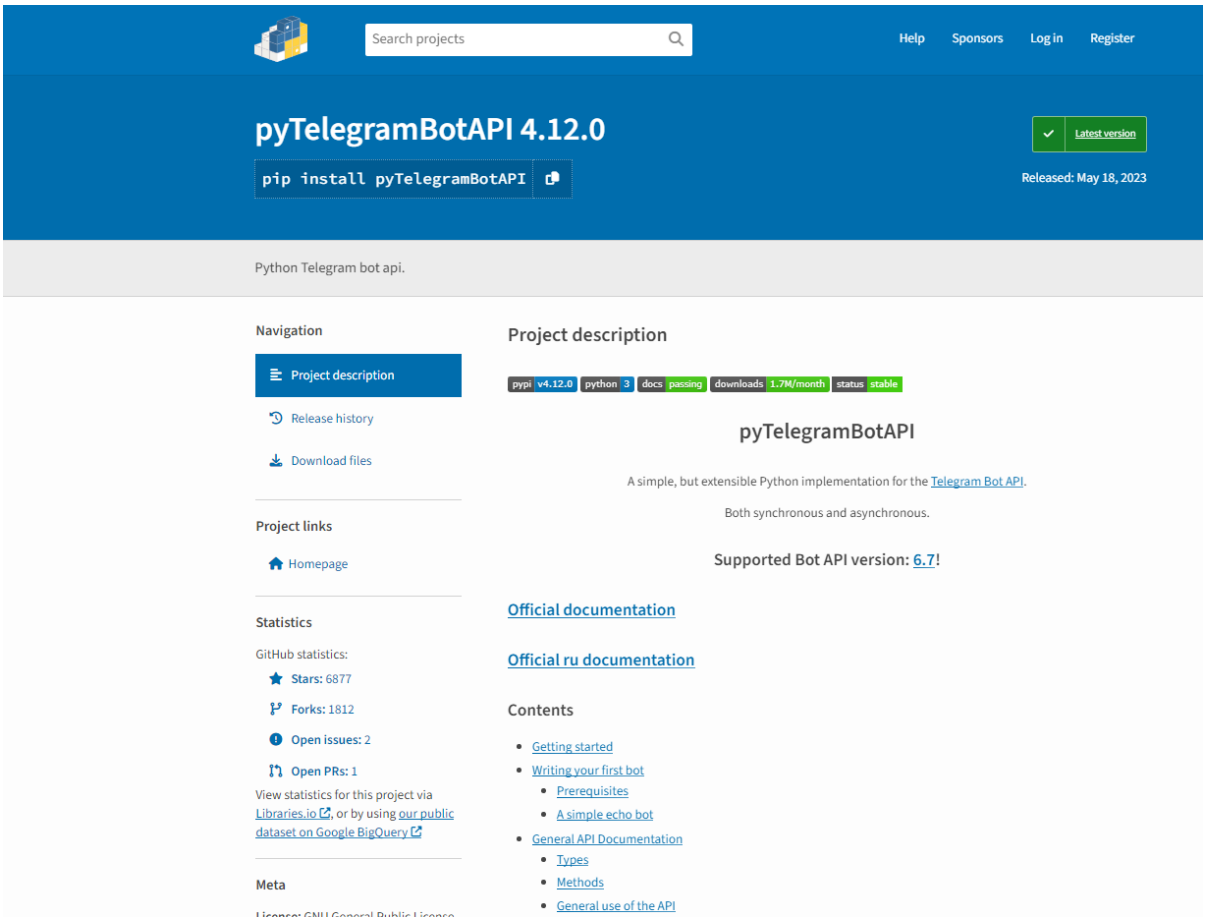


Рис. 2.1. Офіційна сторінка бібліотеки pyTelegramBotAPI

Переваги:

- простий синтаксис та код
- зручність використання
- підтримка, документація та спільнота

Недоліки:

- потребує окрему версію під асинхронне програмування.

Telethon (рис.2.2) – альтернативна бібліотека для взаємодії з API Telegram через обліковий запис бота та протокол Python 3 MTProto. Може обходити стандартні обмеження API

Telegram для клієнта(Наприклад, завантажувати файли об’ємом більше 40мб)

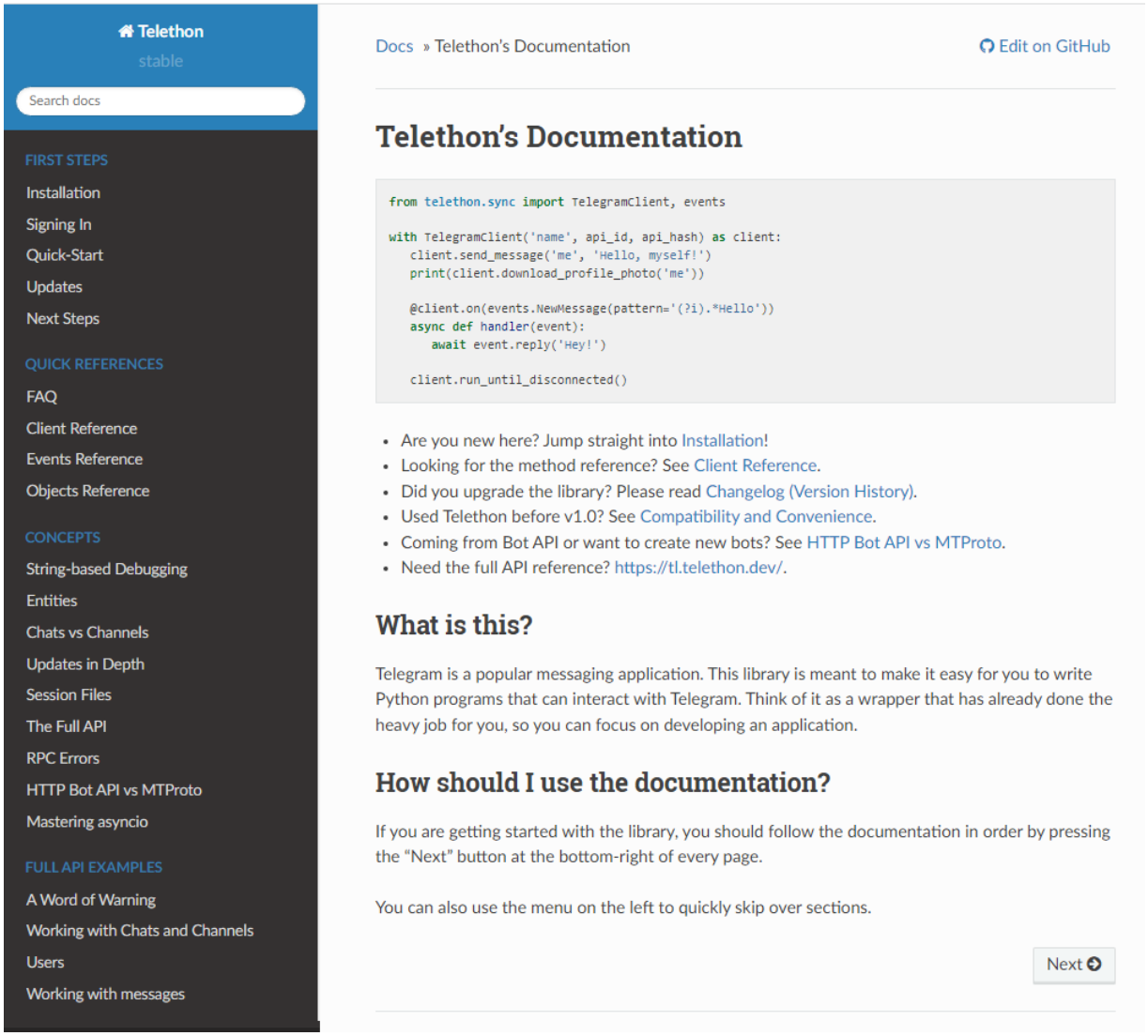


Рис. 2.2. Офіційна сторінка бібліотеки Telethon

Переваги:

- підтримка асинхронного програмування
- обхід більшості обмежень стандартного Telegram Bot API

Недоліки:

- складність у використанні
- потребує знання роботи проколу MTProto
- працює лише у режимі «клієнта»

python-telegram-bot (рис.2.3) – асинхронний інтерфейс для Telegram Bot API. Має найбільш «чисту» реалізацію функцій Telegram Bot API, працює у додатку з сабмодулем telegram.ext для більш зручної реалізації деяких функцій.

Рис. 2.3. Офіційна сторінка бібліотеки python-telegram-bot

Переваги:

- підтримка асинхронного програмування
- майже «чиста» реалізація інтерфейсу для Telegram Bot API

Недоліки:

- складність у використанні
- потребує розуміння функцій Telegram Bot API

AIOGram (рис.2.4) – підходить для важких та об'ємних проєктів, має велику спільноту підтримки користувачів, але має більш складний інтерфейс та займає більше місця для коду. Повністю підтримує асинхронне програмування.

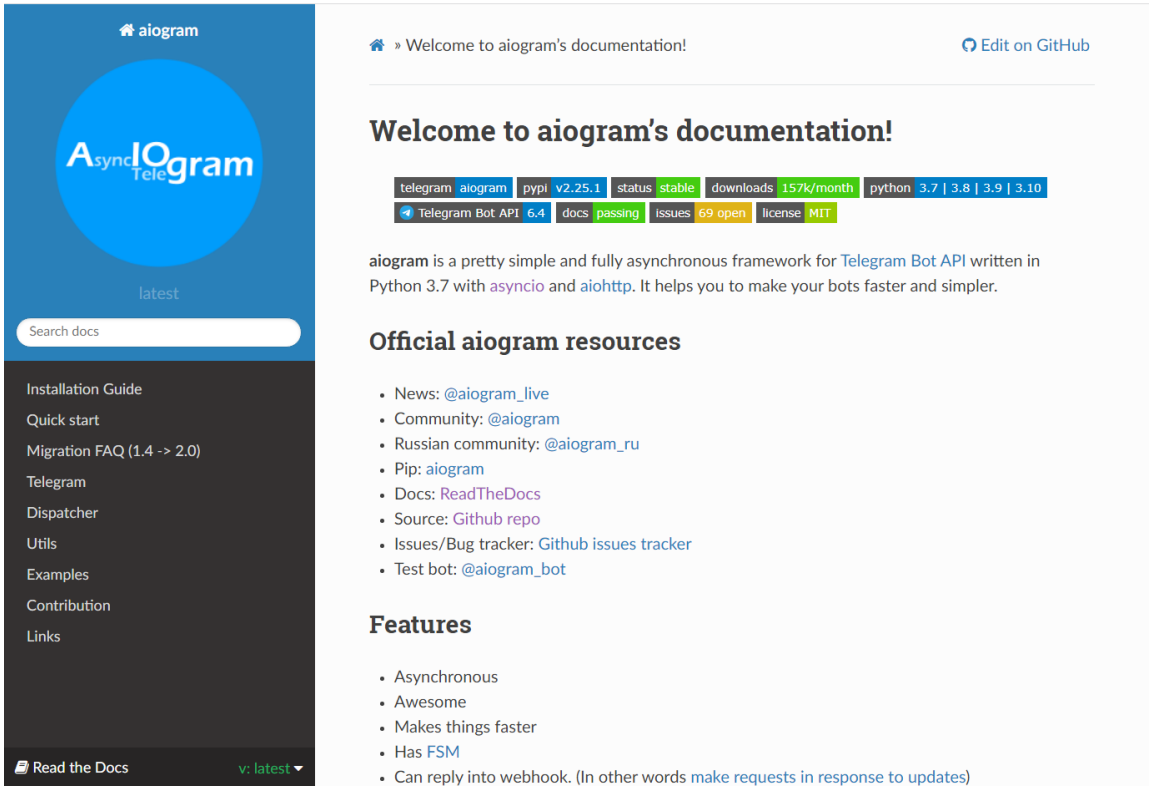


Рис. 2.4. Офіційна сторінка бібліотеки AIOGram

Переваги:

- проста робота з великим обсягом запитів
- підтримка асинхронного програмування
- підтримка, документація та спільнота

Недоліки:

- складність у використанні

Таблиця 2.1.

Порівняння біліотек для Python для створення ботів

	pyTelegramBotAPI	**Telethon**	**python-telegram-bot**	**AIOGram**
підтримка асинхронного програмування	-	+	+	+
підтримка, документація та спільнота	+	+	+	+
реалізація інтерфейсу для Telegram Bot API	+	-	+	+
не потребує додаткових знань	+	-	-	+
простота у використанні	+	-	-	-

Проаналізувавши недоліки і переваги обраних бібліотек (табл. 2.1), ми обрали бібліотеку **AIOGram**, як бібліотеку, що була розроблена спеціально для телеграм ботів, має досить широкий функціонал і задовольняє нашім потребам.

Для проекту була необхідна база данних де б зберігалися усі товари або послуги, що пропонує бот. Було обрано бібліотеку ***sqlite3,*** це база данних, що вшита у **Python**, вона має простий, але достатній для проекту функціонал, при необхідності може бути замінена на більш потужні аналоги.

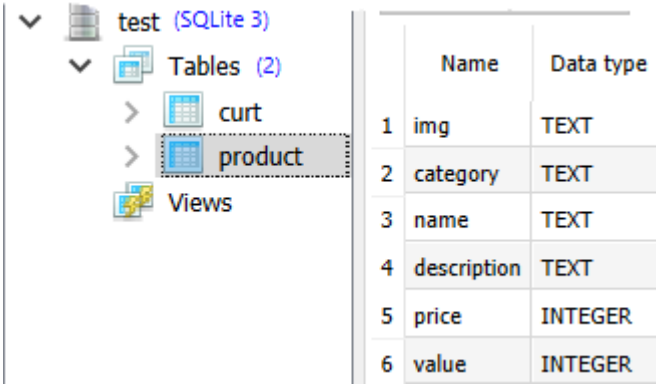
2.2 База даних проекту

Для створення та редагування бази данних використовувався **SQL.**

База данних проекту має настпну струкутуру:

Таблиця з товарами product, що має поля name, category, img, description (тип данних text) та price , vale (тип данних integer).

У перших знаходяться ім’я, категорія, фото та опис товару, а в других ціна за одиницю й кількість на складі відповідно (рис. 2.5).

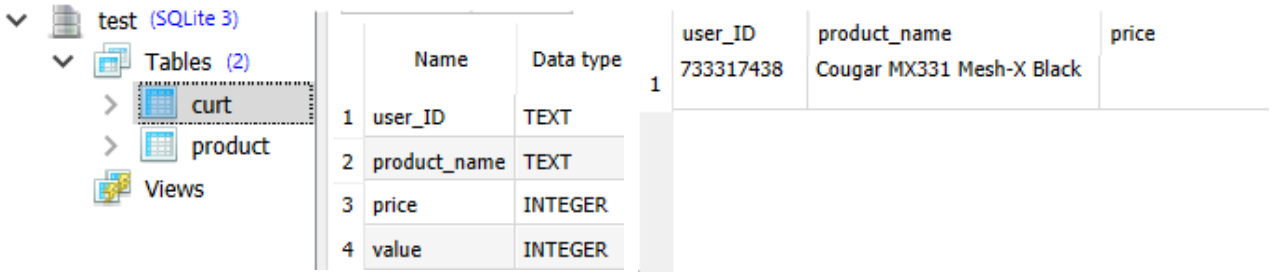


а)

б)

Рис. 2.5. Таблиця product а) структура б) дані

Таблиця з корзинами усіх клієнтів curt (рис.2.6), що має поля user_ID та product_name (тип данних text), price та value(тип данних integer).



а) б)

Рис. 2.6.Таблиця curt а) структура б) дані

2.3 Структура проєкту

Опис логіки розробки чат - боту

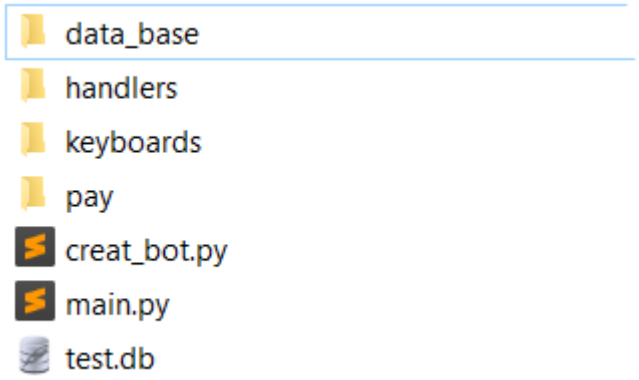
В більшості випадків першим кроком стане створення облікового запису бота. Цю операцію можна виконати по різному в кожній соціальній мережі. Для того, щоб створити обліковий запис бота в соціальній мережі Telegram потрібно звернутись до бота *@BotFather*.

Розглянемо подальшу логіку розробки чат - боту для соціальної мережі Telegram:

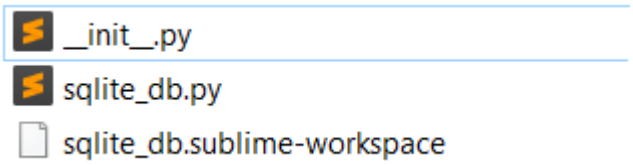
- після того, як ми створили обліковий запис бота, копіюємо його token, що надасть доступ до бота;
- обираємо мову програмування та бібліотеки потрібні для створення бота;
- описуємо алгоритм та функції бота;
- вставляємо токен та створюємо код для бота;

тестуємо бота та виправляємо помилки, баги, тощо.Створення бота я почав з створення файлу *config.py* в якому прописав змінну *TOKEN* для токена бота.

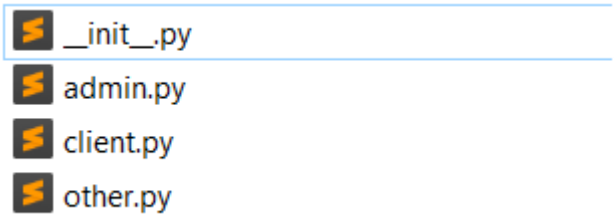
Телеграм-бот «магазин» був написаний мовою програмування ***Python*** (версія 3.11.1) з допомогою окремої бібліотеки ***Aiogram***, та вбудованою в ***Python*** базою даних ***sqlite3***.



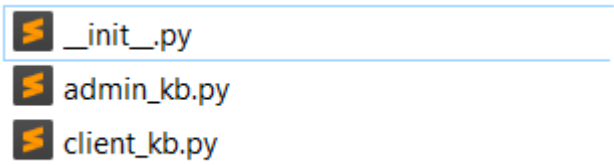
Структура проекту складається з двох основних файлів: ***creat_bot.py*** та ***main.py.*** В цих файлах створюється екземпляр бота та запускається цей екземпляр разом з базою данних відповідно. Також в корні проекту є база данних, що має в собі таблицю усіх товарів та таблицю кошика клієнта, ще в корні проєкту знаходяться пакети проекту.



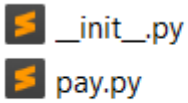
В папці ***data_base*** знаходиться файл, що відповідає за роботу з базою данних



В папці ***handlers*** знаходяться файли, що відповідають за обробку запитів адміністратора, клієнта, та за обробку запитів, що не заплановані системою.



В папці ***keybords*** знаходяться файли, що відповідають за створення натискаємих кнопок для адміністратора та клієнта.



В папці ***pay*** знаходитсья файл, що відповідає за інтернет оплату.

В кожній папці проекту є файл ***__init__.py,*** він потрібен для визначення кожної папки як *пакету* проекта.

2.4 Програмна реалізація проекту

Телеграм боту запускається з файлів **creat_bot.py**

```
# файл для створення бота


from aiogram import Bot
```

```
    from aiogram.dispatcher import Dispatcher
    from aiogram.contrib.fsm_storage.memory import MemoryStorage

    storage = MemoryStorage()

    bot                                                    =
Bot(token="5654951781:AAEX4CIoSL4yC0_tf5_8DUOjJcc24DeCRE8")
    dp = Dispatcher(bot, storage=storage)
```

main.py

```
    from aiogram.utils import executor
    from creat_bot import dp
    from data_base import sqlite_db

    async def on_startup(_): # запуск бота
        print('Бот стартанув') # сигнал що бот запустився
        sqlite_db.sql_start() # підключення бд

    from handlers import client, admin, other
    # регістрація
    client.register_handlers_client(dp)
    admin.register_handlers_admin(dp)
    #other.register_handlers_other(dp)

    executor.start_polling(dp,                skip_updates=True,
on_startup=on_startup)
```

В цих файлах йде створення бота за заданним токеном, що був отриманний у @BotFather, запуск бази данних та регистрация Хендлер для керування та використання бота

Якщо користувач **не є** адміністратором то працює файл **client.py**

```
        #файл для роботи з клиентом
```

```
  from aiogram import types, Dispatcher
  from creat_bot import dp, bot
  from keyboards import client_kb
  from aiogram.types import ReplyKeyboardRemove
  from data_base import sqlite_db
  from aiogram.dispatcher.filters import Text
  from aiogram.utils.markdown import link

  async def commands_start(message: types.Message): # команда запуску роботи з клієнтом
        await message.answer('Привіт)', reply_markup=client_kb.kb_client)

  async def where_we_are(message: types.Message):    # Друкування соц мереж
        if message.text == 'Де ми є':
                await message.answer('Тут будуть посильнн на соц мережі')

  async def product(message: types.Message): # друкування товарів що є в базі
        await sqlite_db.sql_read(message)

  async def curt(callback: types.CallbackQuery):     # додавання\видалення з кошика
        s = callback.data
        if s[4] == '+':
                sqlite_db.curt_operations_p(s[5:], callback.from_user.id)
                await callback.answer(f'{s[5:]} додано до корзину')
        elif s[4] == '-':
                sqlite_db.curt_operations_m(s[5:], callback.from_user.id)
```

```
                await callback.answer(f'Один {s[5:]} видалено з
кошика\n')

  async def shopping_cart(message: types.Message): # друкування
кошика
        mes_text=sqlite_db.curt_read(message.from_user.id)
        if len(mes_text)>0:
                await                        message.answer(f'Ваші
товари:\n\n{mes_text}',reply_markup=client_kb.curt_menu)
        else:
                await message.answer('Кошик ще пустий:(')

  async def edit_curt(message: types.Message):
        await sqlite_db.sql_read(message)

  def register_handlers_client(dp: Dispatcher): # регистрация
обробників
        dp.register_message_handler(commands_start,
commands=['start'])
        dp.register_message_handler(where_we_are, text=['Де ми
є'])
        dp.register_message_handler(product, text=['Товари'])
        dp.register_message_handler(shopping_cart,    text=['Мій
кошик'])
        dp.register_callback_query_handler(curt,
Text(startswith='curt'))
        dp.register_callback_query_handler(edit_curt,Text(starts
with='edit_curt'))
        #dp.register_callback_query_handler(payment,Text(startsw
ith='payment'))
```

Якщо користувач **Є** адміністратором то працює **admin.py**

```
   from aiogram import types, Dispatcher
```

```
    from creat_bot import dp, bot
    from data_base import sqlite_db
    from aiogram.dispatcher import FSMContext
    from aiogram.dispatcher.filters.state import State, StatesGroup
    from aiogram.dispatcher.filters import Text
    from keyboards import admin_kb

    ID = 733317438

    class FSMAdmin(StatesGroup): # создание fsm
          photo = State()
          category = State()
          name = State()
          description = State()
          price = State()
          value = State()

    async def cm_start(message:types.Message):# старт фсм
          if ID == message.from_user.id:
                  await FSMAdmin.photo.set()
                  await message.reply('Фото')

    async    def    load_photo(message:    types.Message,    state:
FSMContext):# добавление фото
          async with state.proxy() as data:
                  data['photo'] = message.photo[0].file_id
          await FSMAdmin.next()
          await message.reply("Категорія")

    async    def    load_category(message:types.Message,    state:
FSMContext):# добавление категории
          async with state.proxy() as data:
                  data['category'] = message.text
          await FSMAdmin.next()
          await message.reply("Назва товару")
```

```
    async def load_name(message: types.Message, state:
FSMContext):# добавление имя
        async with state.proxy() as data:
                data['name'] = message.text
        await FSMAdmin.next()
        await message.reply("Опис товару")

    async def load_description(message: types.Message, state:
FSMContext):# добавление описания
        async with state.proxy() as data:
                data['description'] = message.text
        await FSMAdmin.next()
        await message.reply("Ціна")

    async def load_price(message: types.Message, state:
FSMContext):# добавление цени
        async with state.proxy() as data:
                data['price'] = float(message.text)
        await FSMAdmin.next()
        await message.reply("Кількість на складі")

    async def load_value(message: types.Message, state:
FSMContext):# добавление количества на складе
        async with state.proxy() as data:
                data ['value'] = int(message.text)
        await sqlite_db.sql_add_command(state)
        await state.finish()
        await message.reply("Зроблено!)")

    async def cancel_handler(message: types.Message, state:
FSMContext):# преждевременний выход из fsm
        current_state = await state.get_state()
        if current_state is None:
                return
```

```
        await state.finish()
        await message.reply('Ок')


   def   register_handlers_admin(dp:Dispatcher):   #   регистрация
обработчиков админа
        dp.register_message_handler(cm_start,
commands=['Завантажити'], state = None)
        dp.register_message_handler(cancel_handler,    state="*",
commands='Стоп')
        dp.register_message_handler(cancel_handler,
Text(equals='Стоп',ignore_case=True), state="*")
        dp.register_message_handler(load_photo,
content_types=['photo'], state=FSMAdmin.photo)
        dp.register_message_handler(load_category,
state=FSMAdmin.category)
        dp.register_message_handler(load_name,
state=FSMAdmin.name)
        dp.register_message_handler(load_description,
state=FSMAdmin.description)
        dp.register_message_handler(load_price,
state=FSMAdmin.price)
        dp.register_message_handler(load_value,
state=FSMAdmin.value)
```

Адміністратор може додавати нові товари в бота, перевірка на адміністратора йде по IP аккаунту.

Файл **other.py** обробляє повідомлення, що не призначенні для клієнта чи адміністратора.

Файл **sqlite_db.py** виконує усі функції, що пов’язані з роботою з базою данних

```
   import sqlite3 as sq
   from creat_bot import dp, bot
   from keyboards import client_kb
```

```
    def sql_start(): # запуск базы данних
        global base, cur
        base = sq.connect('test.db')
        cur = base.cursor()
        if base:
            print('База на базі')
        base.execute('CREATE TABLE IF NOT EXISTS product (img TEXT, category TEXT, name TEXT , description TEXT, price INTEGER, value INTEGER )')
        base.execute('CREATE TABLE IF NOT EXISTS curt(user_ID TEXT, product_name TEXT, price INTEGER, value INTEGER)')
        base.commit()


    async def sql_add_command(state): # для добавление товара админимстратором через сам бот, используеться в admin.py
        async with state.proxy() as data:
            cur.execute('INSERT INTO product VALUES (?, ?, ?, ?, ?, ?)', tuple(data.values()))
            base.commit()

    async def sql_read(message): # виводит все товари что есть в базе и все данные о них, создает под каждим товаром инлайн кнопки для добавления или удаления товара из корзины
        for ret in cur.execute('SELECT * FROM product').fetchall():
            await bot.send_photo(message.from_user.id, ret[0], f'Категорія - {ret[1]}\nНазва - {ret[2]}\nОпис:\n{ret[3]}\n\nЦіна - {ret[4]} грн', reply_markup = client_kb.creat_kb_inline(ret[2]))

    def curt_read(us_id): # метод считывает данные о товаре из корзины, выводит все товары в корзине и сумму к оплате, возвращает сообщение что отправиться юзеру
        price = 0
```

```
        mes_text=''
        for ret in cur.execute('SELECT product_name, value, price
FROM curt WHERE user_ID = ?',[us_id]).fetchall():
                mes_text+=f'\n{ret[0]}\nЦіна:    {ret[2]}    *
{ret[1]} = {ret[1]*ret[2]} грн\n\n'
                price+=ret[2]*ret[1]
        if price > 0:
                mes_text+=f'До сплати: {price} грн'
        return mes_text

   def curt_operations_p(prod_id,us_id): # метод для добавления
записи в базе данних curt о том что конкретний клиент добавил
конкретний товар к себе в корзину
        ind = 0
        for ret in cur.execute('SELECT value FROM curt WHERE
user_ID = ? AND product_name = ?',[us_id, prod_id]).fetchall():
                val = ret[0]
                ind+=1
        if ind == 0:
                price = cur.execute('SELECT price FROM product
WHERE name = ?',[prod_id]).fetchall()[0][0]
                cur.execute('INSERT INTO curt VALUES (?, ?, ?,
?)', [us_id, prod_id, price, 1])
        else:
                cur.execute('UPDATE curt SET value=?+1 WHERE
(user_ID = ? AND product_name = ?)',[val, us_id, prod_id])
        base.commit()

   def  curt_operations_m(prod_id,us_id):#  метод  для  удаления
записи в базе данних curt о том что конкретний клиент добавил
конкретний товар к себе в корзину
        ind = 0
        val=-1
        for ret in cur.execute('SELECT value FROM curt WHERE
user_ID = ? AND product_name = ?',[us_id, prod_id]).fetchall():
```

```
                    val = ret[0]
                    ind+=1
        if val == 1:
                    cur.execute('DELETE FROM curt WHERE (user_ID = ?
AND product_name = ?)',[us_id, prod_id])
        if val > 0 and ind > 0:
                    cur.execute('UPDATE  curt  SET  value=?-1  WHERE
(user_ID = ? AND product_name = ?)',[val, us_id, prod_id])
        base.commit()
```

У файлах **clint_kb** та **admin_kb** створюються клавіатури для більш зручного користування ботом

```
    from aiogram.types import ReplyKeyboardMarkup,
KeyboardButton,  ReplyKeyboardRemove,  InlineKeyboardMarkup,
InlineKeyboardButton
    import sqlite3 as sq
    from creat_bot import dp, bot
    from array import *

    kb_client  =  ReplyKeyboardMarkup(resize_keyboard=True,
one_time_keyboard=False) #главное меню покупателя
    main_menu_1 = KeyboardButton('Товари')
    main_menu_2 = KeyboardButton('Де ми є')
    main_menu_3 = KeyboardButton('Мій кошик')
    kb_client.add(main_menu_1).row(main_menu_2, main_menu_3)

    curt_menu  =  InlineKeyboardMarkup(resize_keyboard=True)  #
инлайн меню для редактирования или оплаты корзины покупок
    curt_edit_but  =  InlineKeyboardButton(text='Редагувати',
callback_data=f'edit_curt')
    curt_payment_but  =  InlineKeyboardButton(text='Придбати  ',
callback_data=f'payment_curt')
    curt_menu.row(curt_edit_but,curt_payment_but)
```

```
    def creat_kb_inline(name):
    # метод для создания динамичной клавиатури что будет принимать имя товара
    # и отправлять обработчикам запрос на добавление или удаление єтого товара
    # используеться в файле sqlite_db.py в методе sql_read()
        curt_p = InlineKeyboardButton(text='Додати', callback_data=f'curt+{name}')
        curt_m = InlineKeyboardButton(text='Видалити',callback_data=f'curt-{name}')
        return InlineKeyboardMarkup(resize_keyboard=True).row(curt_p, curt_m)
```

Файл **pay.py** був створений для виконання оплати карткою що прив’язана до телефону.

2.5. Користування ботом

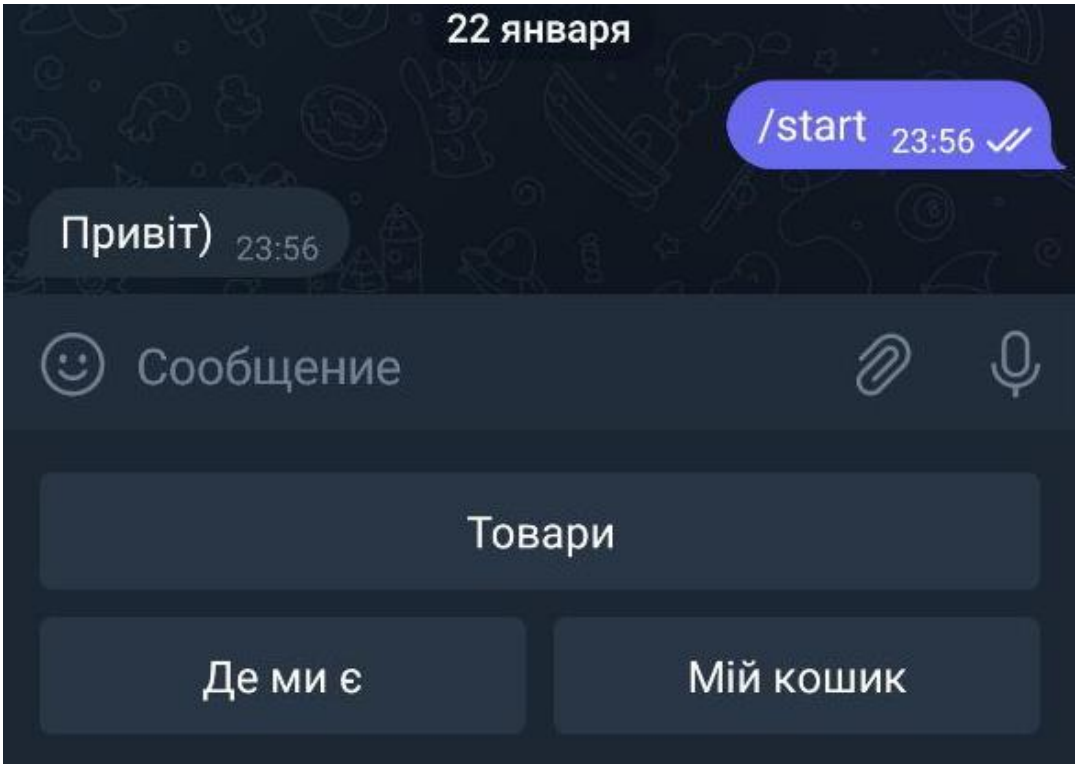


Рис. 2.7. Перший запуск бота

Перший запуск бота, коли ви на нього підписуєтеся й тицяєте старт то він з вами вітається

Рис. 2.8. Кнопка “де ми є”

При натисканні кнопки “де ми є” будуть виводитися посилання на соціальні мережі компанії замовника чи самого замовника

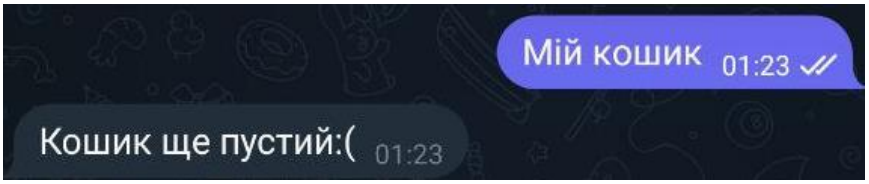


Рис. 2.9. Кнопка “Мій кошик»

Якщо кошик пустий то виводить відповідний напис

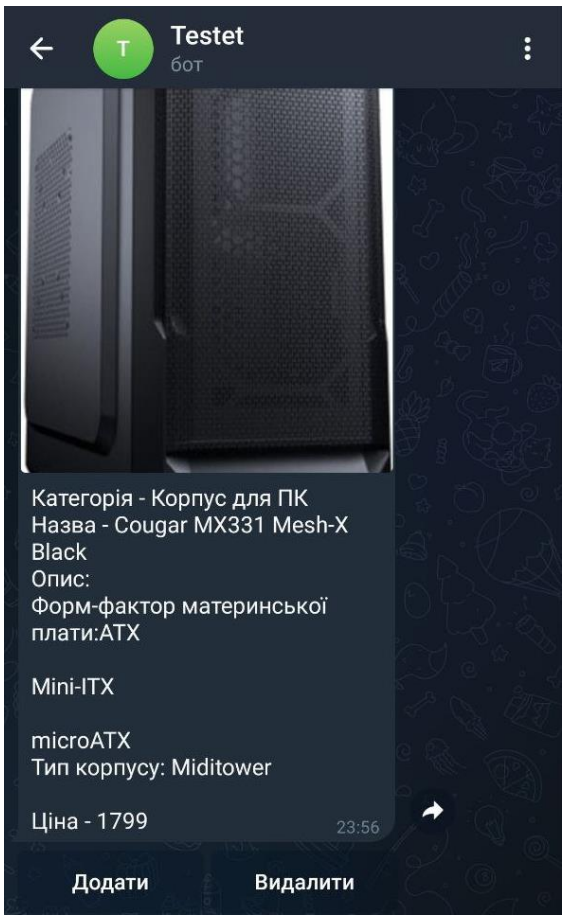


Рис. 2.10. Категорія «Товари»

Виводить список всіх товарів без поділу на категорії, та дві інлайн кнопки з допомогою яких можна додати чи видалити товар з кошика

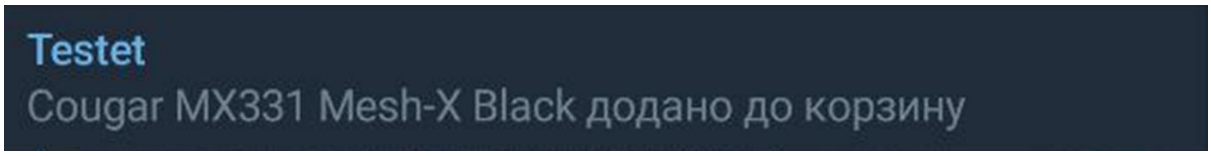


Рис. 2.11. Кнопка “Товари” при додаванні товару до кошика

Рис. 2.12. Кнопка “Товари” при видаленні товару з кошика

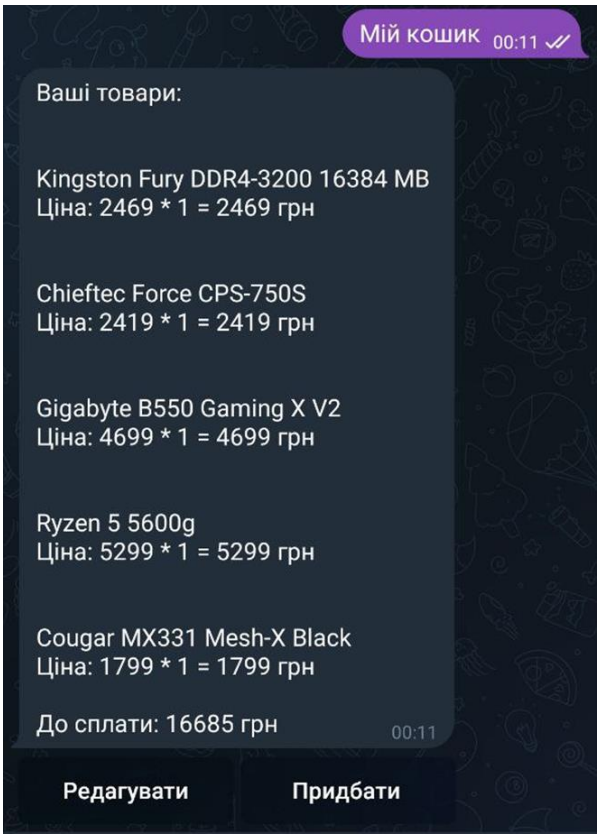


Рис. 2.13. Кнопка “Мій кошик», коли товари вже додано

Друкує заповнений кошик у форматі: назва товару – ціна за одиницю – кількість доданих в кошик товарів – ціна по цьому товару враховуючи кількість

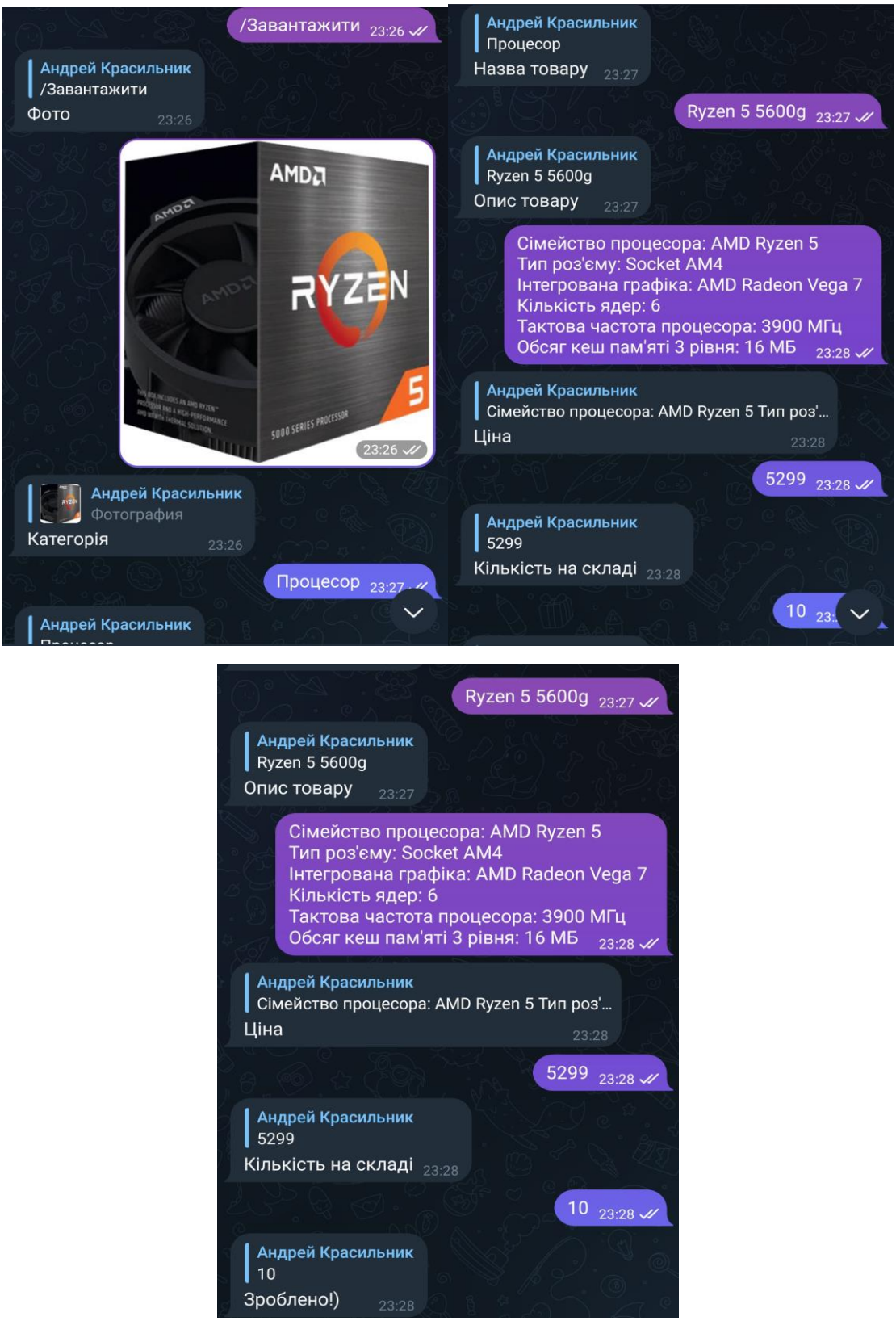


Рис. 2.14. Робота адмін панелі, виконання команди “/Завантажити”

Робота адмін панелі, а точніше команди “/Завантажити”(виконати може лише адмін бо є перевірка на IP)

Висновки до розділу 2

Проаналізувавши недоліки і переваги обраних бібліотек, ми обрали бібліотеку AIOGram, як бібліотеку, що була розроблена спеціально для телеграм ботів, має досить широкий функціонал і задовольняє нашім потребам.

Для проекту була необхідна база данних де б зберігалися усі товари або послуги, що пропонує бот. Було обрано бібліотеку sqlite3, це база данних, що вшита у Python, вона має простий, але достатній для проекту функціонал, при необхідності може бути замінена на більш потужні аналоги.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було розроблено та програмно реалізовано інтернет-магазин для Telegram у вигляді чат-боту мовою Python за допомогою бібліотеки AIOGram та отримані наступні результати та висновки.

1. Проаналізовано сучасний стан розробки чат-ботів та розглянуто актуальні підходи до створення шаблонів для телеграм бота магазина .

2. Шляхом порівняльного аналізу наявного на ринку програмного забезпечення в секторі інструментарію для створення чат-ботів для месенджера телеграм, для реалізації телеграм-бота «Магазин» і було обрано бібліотеку для Python AIOGram, та вбудовану в Python базу даних sqlite3.

3. Проаналізовані основні сценарії для роботи телеграм-бота «Магазина», розроблені алгоритми для їх реалізації.

4. Програмно реалізовано та протестовано телеграм бот «Магазин».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. A brief history of Chatbots. URL : https://chatbotslife.com/a-brief-history-of-chatbots-d5a8689cf52f (дата звернення: 12.02.2022).
2. AIOGram документація URL: https://github.com/aiogram/aiogram (дата звернення: 15.12.2022).
3. Bird S., Klein E., and Loper Ed. Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit URL: https://www.nltk.org/book/ (дата звернення: 25.12.2022).
4. pyTelegramBotAPI документація URL: https://github.com/eternnoir/pyTelegramBotAPI#asynctelebot (дата звернення: 15.12.2022).
5. Python документація URL: https://docs.python.org (дата звернення: 5.04.2023).
6. Python (programming language) URL: https://en.wikipedia.org/wiki/Python_(programming_language) (дата звернення: 1.12.2022).
7. python-telegram-bot документація URL: https://github.com/python-telegram-bot/python-telegram-bot (дата звернення: 15.12.2022).
8. Revenue models for bots and chatbots URL:https://chatbotslife.com/revenue-models-for-bots-and-chatbots-702ca78a1b07 (дата звернення: 6.12.2022).
9. Telethon документація URL:https://github.com/LonamiWebs/Telethon
10. Telegram-bot-api VS aiogram URL: https://www.libhunt.com/compare-tdlib--telegram-bot-api-vs-aiogram (дата звернення: 15.12.2022).
11. Telegram Bot API документація URL: https://core.telegram.org/bots/api (дата звернення: 14.03.2023).
12. Telegram Bot на Python3 та aiogram URL: https://surik00.gitbooks.io/aiogram-lessons/content/chapter1.html (дата звернення: 6.04.2023).

13. The five different types of chatbot - and which to choose for your business URL: https://www.mycustomer.com/service/channels/the-five-different-types-of-chatbot-and-which-to-choose-for-your-business (дата звернення: 5.12.2022).
14. Welcome to aiogram's documentation! https://docs.aiogram.dev/en/latest/#welcome-to-aiogram-s-documentation (дата звернення: 5.12.2022).
15. Даниленко Дарина. Як чат-боти стають новими медіа? Інтернет свобода. 2020. URL: https://netfreedom.org.ua/article/column-yak-chatboti-stayut-novimi-media-i-chomu-zhurnalistam-yesens-iz-nimi-rozibratisya?fbclid=IwAR0JUl4eWw_gkMs0bQ8MlGAyux76X8e1g9GGAMBPaGYvrx (дата звернення: 22.12.2022).
16. Дослідження. Цифрові чат-боти Accenture нікуди не подінуться URL: https://www.accenture.com/_acnmedia/pdf-77/accenture-research-conversationalai-platforms.pdfXrMbnCxt30S6A. (дата звернення: 8.12.2022).
17. Мельник, Ю.М., Сагер, Л.Ю., Черкас, І.Ю. Трансформація маркетингових комунікацій: нетрадиційні види. Вісник Хмельницького національного університету, 2016, 2, 164-168.
18. Пастернак Марта. Що таке чат – боти та кому вони потрібні? Creative SMM. 2017. URL: https://creativesmm.com.ua/shho-take-chatbot-ta-komu-vonu-potribni/. (дата звернення: 7.12.2022).
19. Посібник з мови програмування Python URL: https://metanit.com/python/tutorial/ (дата звернення: 1.04.2023).