МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
КАФЕДРА ПРИКЛАДНОЇ ІНФОРМАТИКИ

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

_____ Гамаюн В.П.
(підпис) (ПІБ)

"_____"_____2021р.

ДИПЛОМНИЙ ПРОЕКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ "БАКАЛАВР"

Тема: Telegram-додаток

Виконавець: _____ Потапенко Володимир Сергійович
(підпис) **(ПІБ)**

Керівник: _____ Толстікова Олена Володимирівна
(підпис) **(ПІБ)**

Нормоконтролер: _____ Боровик Володимир Миколайович
(підпис) **(ПІБ)**

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра прикладної інформатки
Освітній ступень Бакалавр
Спеціальність 122 "Комп'ютерні науки"

ЗАТВЕРДЖУЮ:

Завідувач кафедри

_____ _Гамаюн В.П.
(підпис) (ПІБ)

"_____"_____2021р.

ЗАВДАННЯ
на виконання дипломного проекту

Потапенка Володимир Сергійовича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема дипломного проекту: *Telegram-додаток*

затверджена додатком до наказу ректора від "___"____2021 р., №___/ст_____

2. Термін виконання проекту: з " 10 "травня 2021 р. по " 13 " червня 2021 р.

3. Вихідні дані до проекту: *1. Проаналізувати можливості чат-ботів. 2. Розглянути можливі шляхи розробки додатку. 3. Розробити чат-бот для автоматизованого процесу продажі товарів.*

4. Зміст пояснювальної записки: *1. Огляд і порівняльний аналіз чат ботів. 2. Методології опису розробки додатку. 3. Практична реалізація Telegram-додатку.*

5. Перелік обов'язкового графічного (ілюстративного) матеріалу:

1. Демонстрація роботи бота @PkNauBot. 2. Інтелектуальна схема. 3. Схема бази даних. 4. Створення нового бота через @BotFather. 5. Перехід на веб-посилання з готовою формою. 6. Отримання купленого товару. 7. Запуск файлу для початку роботи чат-бота.

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Відмітка про виконання
1.	Ознайомлення з постановкою задачі та вивчення літератури Написання 1 розділу, представлення керівнику	10.05.21 – 17.05.21	
2.	Написання 2 розділу, представлення керівнику	18.05.21 – 24.05.21	
3.	Написання 3 розділу, представлення керівнику	25.05.21 – 01.06.21	
4.	Загальне редагування та друк пояснювальної записки, графічного матеріалу	01.06.21 – 03.06.21	
5.	Проходження нормоконтролю, перепліт пояснювальної записки.	07.06.21 – 10.06.21	
6.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	08.06.21 – 11.06.21	

Дата видачі завдання: "____"_____ 2021 р.

Керівник дипломного проекту _____ Толстікова О.В.
(підпис керівника) (ПІБ.)

Завдання прийняв до виконання _____ Потапенко В.С.
(підпис випускника) (ПІБ.)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту «Телеграм-додаток», загальним обсягом 79 сторінок, 62 рисунки, 22 джерел літератури.

СЕРВІС ПРОДАЖУ, АВТОМАТИЗАЦІЯ ПРОДАЖУ ТОВАРУ, ДОДАТОК ДЛЯ МАЛИХ ТА СЕРЕДНІХ БІЗНЕСІВ, TELEGRAM-ДОДАТОК.

Об'єкт дослідження: компанія, яка бажає автоматизувати процес продажу за допомогою чат-ботів.

Предмет дослідження: опис бізнес-процес за допомогою мов програмування для розробки чат-боту.

Мета дослідження: створення загальнодоступного *Telegram*-бота з подальшим пошуком новачків у сфері продажу, щоб допомогти зберегти кошти, які вони можуть тратити на зарплатню стаціонарному продавцю.

Завдання дослідження: огляд аналогічних програм та їх функціоналу; здобути навички у програмуванні складних проектів за допомогою вже існуючих модулів; створення власного чат-бота.

Практична цінність: результати проекту можуть бути використані як і у сферах малого бізнесу для початківців, так і в складних бізнес-системах для автоматизації роботи продавця. Розроблена система дозволить скоротити витрати зі сторони компанії та підвищить ефективність обслуговування клієнтів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

ПЗ – Програмне забезпечення

СУБД – Система управління базами даних

БД – База даних

ОС – Операційна система

API – Прикладний програмний інтерфейс (*Application Programming Interface*)

DNS – Сервіс доменних імен (*Domain Name System*)

IDE – Інтегроване середовище розробки *(Integrated Development Environment)*

CRM – Система управління взаємовідношення з клієнтами (*Customer Relationship Management)*

FAQ – Набір часто задаваних питань на первну тему та відповідей на них (*Frequently Asked Questions)*

GIT – Розподілена система керування версіями файлів та спільної роботи

HTTPS – Схема *URI*, що синтаксично ідентична http схемі, яка зазвичай використовується для доступу до ресурсів Інтернет

HTTP – протокол передачі даних, що використовується в комп'ютерних мережах (*HyperText Transfer Protocol*)

API – Опис способів (набір класів, процедур, функцій, структур або констант), якими одна комп'ютерна програма може взаємодіяти з іншою програмою

CI/CD – Технологія автоматизації тестування та доставки нових модулів розроблюваного проекту зацікавленим сторонам (*Continuous Integration, Continuous Delivery*)

PEP – Набір правил до написання «чистого» коду з подальшим розумінням іншим розробником (*Python Enhanced Proposal*)

PaaS – Платформа як послуга, наприклад, веб-сервер або база даних; клієнт управляє програми, операційну систему управляє провайдер (*Platform as a Service*)

SaaS – Програмне забезпечення як послуга, наприклад, електронна пошта або інше офісне додаток; клієнт користується додатком, базовими настройками додатку управляє провайдер (*Software as a Service*)

IaaS – Інфраструктура як послуга, наприклад, віртуальні сервери і віртуальна мережа; клієнт може встановлювати будь-яке програмне забезпечення і додатки (*Infrastructure as a Service*)

GUI – Тип інтерфейсу, який дає змогу користувачам взаємодіяти з електронними пристроями через графічні зображення та візуальні вказівки (*Graphical user interface)*

VDS – Хостинг-послуга, де користувачеві надається віртуальний сервер з максимальними привілеями (*Virtual Dedicated Server*)

CLI – Інтерфейс командного рядка (*Command-line interface*)

SQL – Мова структурованих запитів *(structured query language)*

GIST – Узагальнене пошукове дерево (*Generalized Search Tree*)

ВСТУП

У сучасному світі, де технології весь час модернізуються і оновлюються, людина не перестаючи освоює нові гаджети, спілкується за допомогою служб миттєвого обміну повідомленнями, вивчає різні програми та їх можливості. Вже немає жодної сфери діяльності, в якій не використовувалися б інформаційні технології: в банківських справах функціонують інтернет-банки, в освіті-електронні підручники.

Завдяки розвитку інтернет-технологій нові можливості щодня з'являються і в області бізнесу. Кожна людина може за лічені секунди знайти інформацію про те чи іншому підприємстві, навіть перебуваючи в затишній домашній обстановці або на роботі. Отримавши первинне ознайомлення, через мобільний додаток, він може з легкістю придбати товар або послугу, також не виходячи з дому.

Постійно мінливі ринкові умови, висока швидкість прийняття рішень, багатозадачність в управлінні активами та необхідність зниження ризиків вимагають сучасних підходів до організації підприємницької діяльності. Рішення все більш складному внутрішньому та зовнішньому середовища підприємства полягає в комплексній автоматизації бізнес-процесів. Це дозволяє вивільнити цінні ресурси для стратегічного планування та концентрації управління в ключових сферах діяльності компанії.

Необхідність автоматизації інформаційних процесів обумовлена збільшенням обсягу інформації в інформаційній системі (ІС) організації, необхідністю прискорення та використання більш складних методів їх обробки.

Автоматизація бізнесу - це частковий або повний переклад стереотипних операцій і бізнес-задач під управління спеціалізованої інформаційної системи або складного апаратного та програмного забезпечення. В результаті чого відбувається вивільнення людських і фінансових ресурсів для підвищення продуктивності праці і ефективності стратегічного управління.

Основними завданнями автоматизації інформаційних процесів є:

– усунення рутинних операцій;

– зниження трудовитрат при виконанні традиційних процесів і операцій;

– збільшення швидкості обробки інформації та процесів перетворення;

– забезпечення більшої ефективності та якості обслуговування клієнтів;

– надання широких можливостей для статистичного аналізу та підвищення точності обліку та звітності інформації;

– надання більших можливостей для організації та ефективного використання інформаційних ресурсів за рахунок використання інформаційних технологій.

Таким чином слід визнати, що автоматизація бізнес-процесів важлива для компанії та приводить до розвитку компанії, так як підвищується її ефективність. Для того, щоб процедура ознайомлення і придбання відбувалася ще більш зручно для користувача, такі додатки оснащені спеціальними програмами - ботами, що імітують живе спілкування. Тому мобільні додатки з чат-ботами — це величезний плюс для сучасних людей, які намагаються правильно розподіляти свій час.

Якщо до недавнього часу популярними були додатки або комп’ютерні програми, то на даний момент лідерство займають чат-боти, які мають великі перспективи в різних сферах нашого життя.

Здається, ще вчора ми і не знали про існування таких ботів, як зараз не можемо уявити своє життя без них. Вони стали широко відомими серед палких користувачів месенджерів.

Чат-боти — це швидкозростаючі у популярності інструменти для бізнесу, за допомогою яких можна автоматизувати рутинні завдання, запитувати та збирати актуальну інформацію про свою цільову аудиторію, автоматично консультувати клієнтів, відповідати на запитання, продавати товари та послуги, приймати оплату, висилати повідомлення та багато іншого.

РОЗДІЛ 1
ОГЛЯД І ПОРІВНЯЛЬНИЙ АНАЛІЗ ЧАТ-БОТІВ

1.1. Загальні поняття чат-бота. Функції. Класифікація

Чат-боти — це спеціальні обліковий запис, за яким не закріплено жодної людини, а повідомлення, відправлені з них або на них, обробляються зовнішньою системою. Крім того, для користувача спілкування з ботом виглядає як звичайне листування з реальною людиною.

Залежно від поставленого завдання, можна виділити велику кількість типів ботів. По-перше, це можуть бути боти, які взаємодіють з різними сервісами. Як приклад, бот, через який ведеться управління розумним будинком. По-друге, боти можуть показувати погоду, бути перекладачами текстів з різних мов. Це так звані боти-утиліти або інструменти.

Таким чином, можна зробити висновок, що в 2021 році легко підібрати бота «на будь-який смак і колір».

Телеграм-бот можуть використовувати у таких напрямках:

- підтримка клієнтів. Чат-бот допоможе замінити незручний *FAQ* на сайті, який іноді не відразу можна побачити, зможе відповісти на типові питання клієнта. Бот може працювати 24 години на добу і зменшить навантаження ваших співробітників;
- маркетинг. Чат-бот - це ще один маркетинговий інструмент, який допоможе поширювати контент, підтримувати лояльність клієнтів і збирати аналітику. За допомогою нього можна робити розсилки, інформувати клієнтів про акції, збирати коментарі про товари чи послуги, якість обслуговування;
- робота всередині компанії. Чат-боти допомагають оптимізувати в роботі такі процеси як: бронювання переговорів, інформування співробітників про

- дати відпусток, розклад корпоративного транспорту, терміни зарплати та багато іншого;
- рекрутинг. Функціональність просунутих ботів – це первинний збір інформації про кандидатів. На основі співбесіди з чат-ботом менеджер по персоналу вирішує, яких кандидатів слід запросити на живе спілкування, хто повинен пройти тестове завдання, а кому слід відмовити;
- інтернет магазин. Бот допоможе Вам контролювати автоматичні продажі. Ваш клієнт зможе пересуватись по каталогу товарів, подивитися опис товару, додати його в кошик і зробити онлайн-оплату замовлення. Також можлива інтеграція бота з вашим сайтом;
- візитка компанії. Бот з легкістю розповість клієнту все про Вашу компанію. ЇЇ історію, діяльність та послуги. Дасть відповідь на актуальні питання. Побудує маршрут до офісу або зручний спосіб зв'язку з менеджером;
- *CRM* система. Створіть бота, який буде надсилати Вам повідомлення з сайту, сповіщення про нові замовлення. Або ж отримуйте статистику і аналітику Ваших бізнес-процесів;
- онлайн консультант. Відповіді клієнтам в автоматичному порядку на базі найчастіших запитань або живе спілкування з менеджером. Все це не виходячи з телеграм. Цей бот можна використовувати як самостійно, так і інтегрувати з сайтом;
- інформаційний блог. Ведете канал або сайт? Бот допоможе Вам автоматично постити в телеграм всі нові публікації або важливі події прямо з сайту;
- інтеграція з сайтом. Будь-який функціонал з вашого веб-сайту можна перенести у месенджер. Купівля товару, автопостинг публікацій, онлайн-чат, оповіщення з сайту. При цьому обмін інформації буде моментальний.

Існує кілька варіантів класифікації чат-ботів, але проаналізувавши їх, виділимо два типи: бізнес-класифікація чат-бот додатків і класифікація чат-ботів з технічного типу.

Діаграма бізнес-класифікації показано на рис. 1.1.

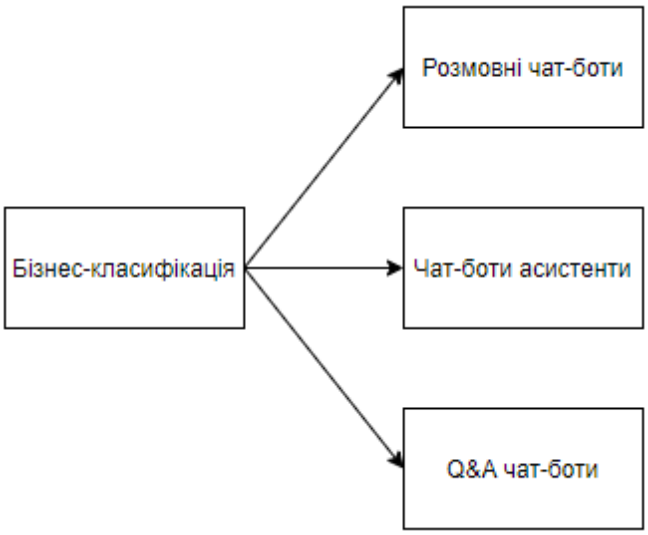


Рис. 1.1. Бізнес-класифікація чат-бот додатків

Розглянемо кожен тип більш детально:

– розмовні — створені для спілкування, дуже схожі на спілкування зі звичайною людиною, не мають конкретної мети;

– асистенти — виходячи з конкретних цілей, отримують необхідні дані від відповідей користувача;

– *Q&A* (питання-відповідь) — принцип роботи: одне питання – одна відповідь.

Діаграма класифікацій з технічного типу представлена на рис. 2.

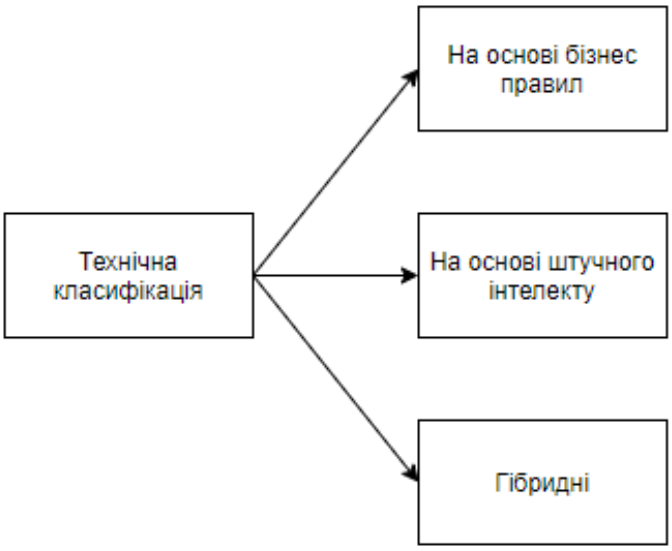


Рис. 1.2. Технічна класифікація чат-ботів

Розглянемо кожен тип більш детально:

– засновані на бізнес-правилах. У такому типі вже заздалегідь розписаний сценарій подій розробником і має дерево-подібну структуру. Завдяки великій кількості кнопок людина приходить до певного шляху. Питань з відповіддю у вільній формі в такому типі не існує;

– засновані на штучному інтелекті. повністю відрізняються від першого типу, не мають визначену структуру. Шлях розмови визначено неявним чином на основі тестованих даних, які використовувалися для здобування навичок моделі машинного навчання. Такі боти повинні мати великий обсяг даних для якісної роботи;

– гібридні. Цей тип чат-ботів використовує в собі взаємодію першого і другого типу, тобто розмова з користувачем ведеться по заздалегідь продуманому сценарію, але використовують штучний інтелект для визначення намірів користувача, і вилучення даних з їх листування.

1.2. Головні переваги чат-ботів

Доступ з будь-яких пристроїв. Немає потреби у розробці під різні операційні системи, браузери тощо.

Інтеграція з екосистемою компанії. Можливість інтеграції з іншими додатками компанії: з сайтом, *1C, CRM* і т.д.

Без додаткової реєстрації. Зі появою нових програм з'являються і нові проблеми. Велика кількість додатків знаходиться на відкритих просторах інтернету. Завантаження з неперевіреного сайту може спричинити за собою серйозні наслідки. Довга реєстрація з подальшою розсилкою реклами або додатки, які не містять в собі достатнього функціоналу, так само не отримують схвалення від користувачів. Виникає потреба створення зручних у використанні і з великим набором функцій додатків на таких платформах, які б не викликали у людини підозр. Тому для користування Телеграм-ботом досить мати особистий Телеграм кабінет з прив’язкою по номеру телефону.

Миттєві відповіді. Клієнти отримують відповіді на питання в повідомленнях від бота. Це зменшує час відповіді на заявку.

Конфіденційність повідомлень. Телеграм один з найбільш захищених месенджерів. Ніхто не отримає доступ до Вашого листування.

Автоматизація процесів. Більшість завдань менеджерів, можна покласти на плечі чат-бота. Налаштувавши бізнес-логіку бота, Ви отримаєте співробітника, котрий працює безперервно.

1.3. Платформи, які підтримують роботу чат-ботів

Бот-платформа - це додаток, на *API* якого можна створити віртуального співрозмовника. Сьогодні чат-боти підтримують більшість популярних месенджерів. Серед них *Facebook Messenger, Viber, Whatsapp, Skype, Telegram, Sender, Wechat* і навіть *Instagram.* Так як платформи у всіх месенджерів різні, то і *API* у них відрізняються. Тому не можна створити один чат-бот для всіх платформ відразу.

Вибираючи відповідну платформу для свого бота, слід виходити з її можливостей. Розберемо функціонал чат-ботів на самих популярних платформах.

Facebook Messenger

Чат-боти на платформі *Facebook Messenger* мають просунутий набір функцій. У нього входять:

– текстові повідомлення;

– кнопки з варіантами дій;

– структурні елементи (до 10 в одному повідомленні);

– рахунки на оплату.

Щоб створити чат-бот на *Facebook*, у компанії обов'язково повинна бути публічна сторінка в цій соціальній мережі.

Viber

Чат-бот в *Viber* не поступається по функціоналу іншим платформам. Взаємодія з аудиторією ведеться з публічного облікового запису в месенджері. У нього є набір стандартних можливостей і кілька додаткових. А саме:

– розсилки (відправлення повідомлень всім контактам, включаючи не підписаних на канал);

– оформлення постів у вигляді «каруселі» з товарами.

Telegram

Один із перший месенджерів, який надав можливість створювати ботів. Їх в Телеграм легко розпізнати по приставці «bot» в назві. Це одна з обов'язкових вимог платформи для власників автоматизованих чатів.

Логіка чат-бота в *Telegram* контролюється за допомогою ***HTTPS*** запитів до API платформи. Основні функції, які доступні:

– інтеграція з іншими сервісами;

– робота в інлайн-режимі (бот вбудовується в інші діалоги);

– рішення різних завдань (бот може передбачати погоду, перекладати тексти і т. д.);

– гра з користувачем, інтерактивна взаємодія;

– відправлення текстових повідомлень, коментарів, пошук інформації;

– виконувати команди.

Також на платформі є можливості для заміни інтерфейсу, кастомізації клавіатури, зовнішнього зв'язування.

Skype

На платформі *Skype bot* доступні до створення текстові чат-боти, які мало відрізняються від конкурентів. Вони точно так само відповідають на запити користувача по заздалегідь прописаним алгоритмам. У повідомленнях можна додавати і змінювати елементи. З цікавих функцій варто відзначити блок «Меню» з переліком можливостей конкретного чат-бота. Він показується користувачеві при знайомстві з роботом.

Велику увагу приділено здатності бота до саморозвитку. Так, *Skype bot* використовує *Cognitive Service Language API*. Це передбачає збір ключових слів з тексту і побудова на них відповідей, облік користувацького досвіду, самостійну генерацію відповідей з напів-структурованого набору шаблонів.

Sender

Платформа активно застосовується в бізнесі, так як має гнучкі настройками. Функціонал чат-бота на Sender забезпечують *API* інтерфейси: для побудови логіки бізнес процесів - *Corezoid*, для отримання і обробки платежів - *Liqpay*.

Можливості віртуального співрозмовника включають:

– відправку повідомлень;

– виставлення рахунків;

– отримання оплати;

– розсилки;

– маркетингові дослідження;

– проведення опитувань.

Розроблений в *Sender* чат-бот можна розмістити як в месенджері, так і на сайті.

Slack

Ця платформа підходить для створення корпоративних чат-ботів. Наприклад, для проведення опитувань серед співробітників, складання звіту про поточні справи, планування спільних нарад і графіків кожного із співробітників.

Існує два варіанти ботів, які дозволяє створити *Slack*. Перший - призначений для користувача. Він буде існувати в межах месенджера і при необхідності додавати в групи та діалоги співробітників. Другий - мобільний додаток. Такий бот розміщується за межами *Slack*, але інтегрується з месенджером.

1.4. Дослідження аналогів чат-ботів

Наведемо декілька прикладів застосування саме Телеграм-додатків та розглянемо їхнє призначення:

1) Назва: «Приймальна комісія НАУ».

Ім’я облікового запису: *@PkNauBot*.

Призначення: оптимізація процесу приймальної комісії.

Функції: бот дає можливість швидко та ефективно дати користувачеві потрібну інформацію або алгоритми дій щодо вступу до університету.

Ціль додатку: спростити навантаження працівникам приймальної комісії та абітурієнтам.

Цей бот дозволив зменшити кількість дзвінків та електронних запитів до приймальної комісію Національного авіаційного університету.

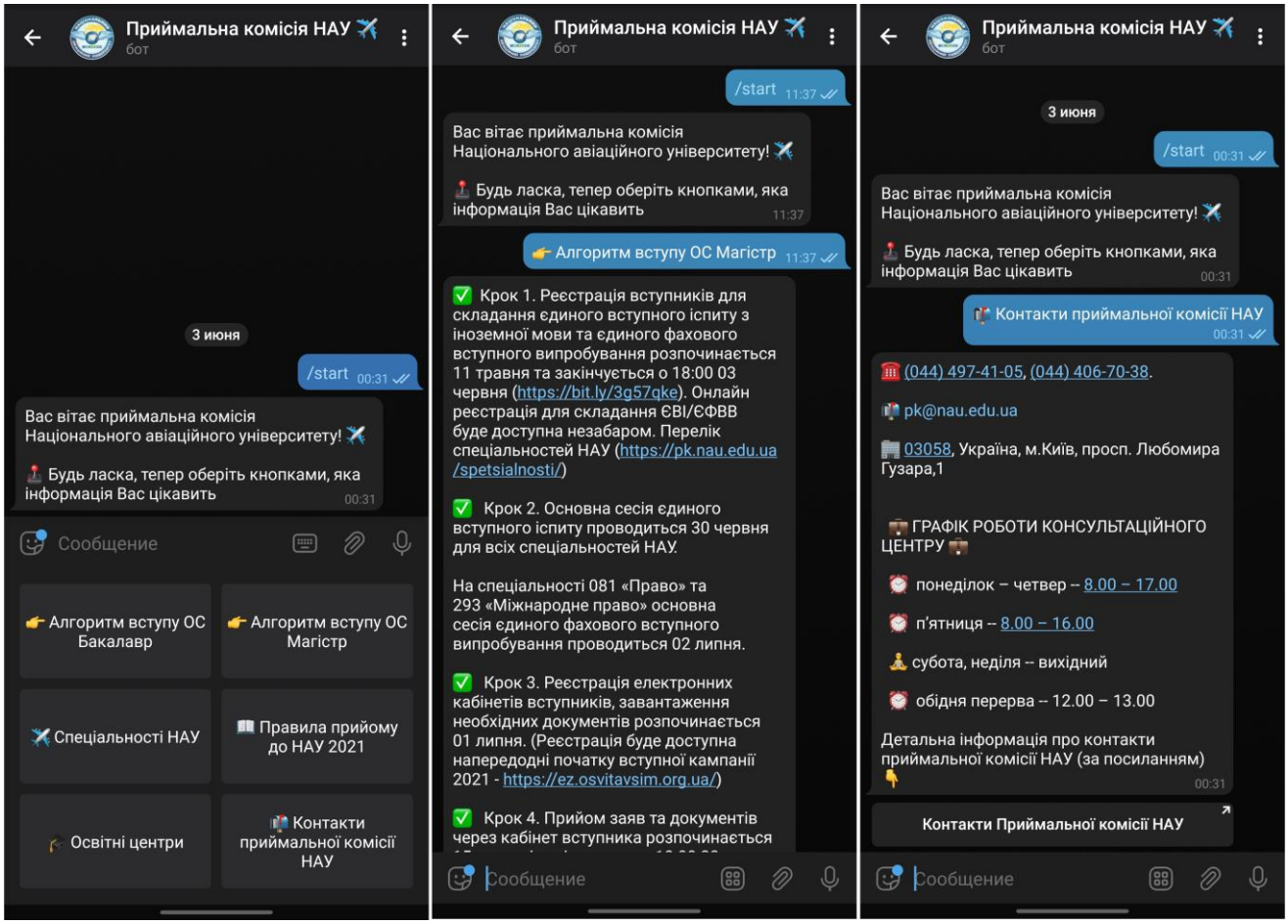


Рис. 1.3. Демонстрація роботи бота *@PkNauBot*

2) Назва: «*PostiraykaBot*».

Ім’я облікового запису: *@PostiraykaBot*.

Призначення: моніторинг пральних машин у гуртожитках.

Функції: показ вільних пральних машин у реальному часі; вирішення проблем, алгоритм яких вже описаний; дізнатись точний час завершення прання.

Ціль додатку: зберегти час студентів, бажаючих випрати свої речі; надання технічної підтримки 24/7.

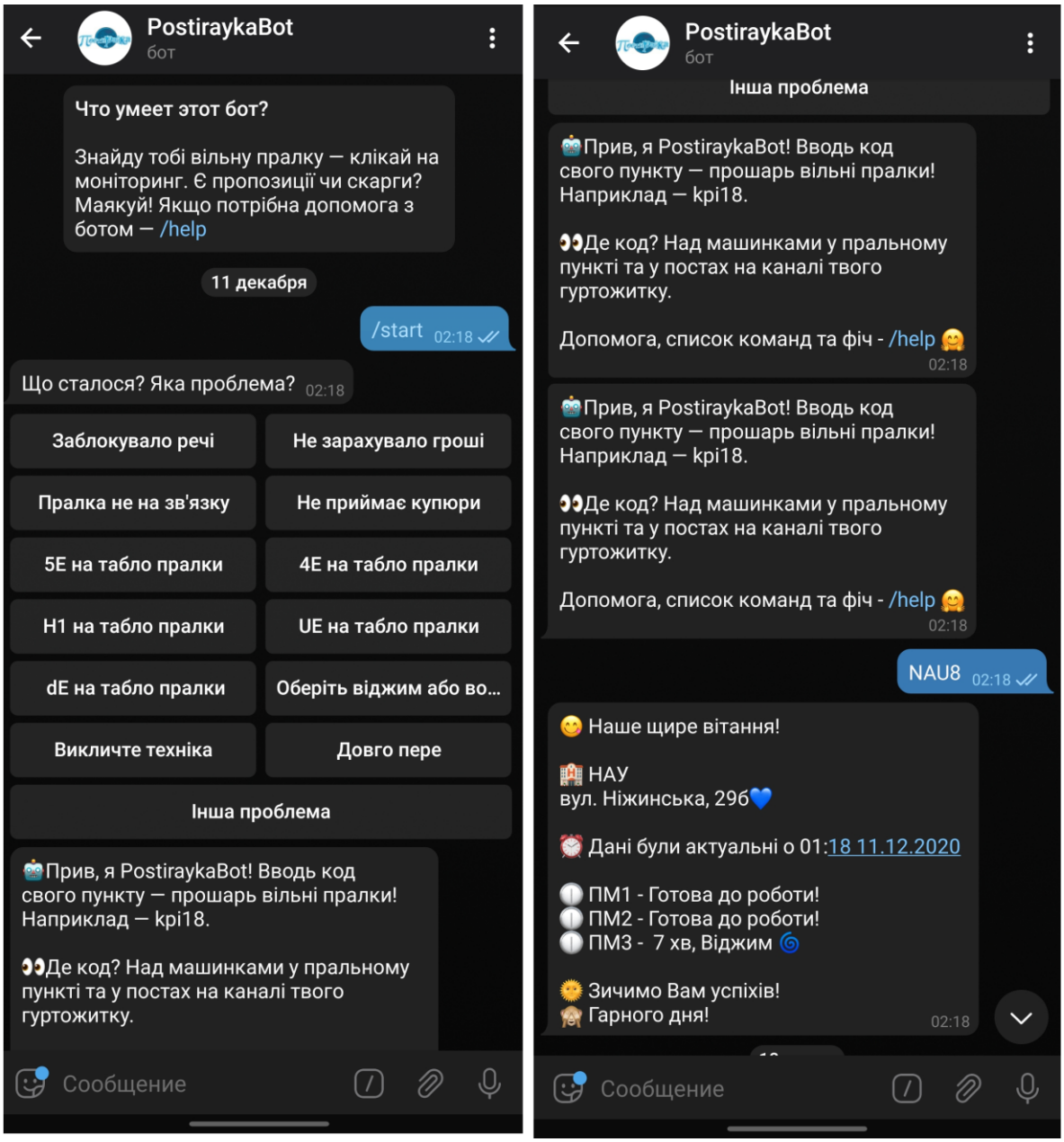


Рис. 1.4. Демонстрація роботи бота *@PostiraykaBot*

3) Назва: «ХтоНаВахті?».

Ім’я облікового запису: *@whois_doorman_bot*.

Призначення: моніторинг графіку чергування вахтерів гуртожитку.

Функції: показати дати та імена вахтерів, які чергують.

Ціль додатку: повідомляти студентів про розклад чергувань.

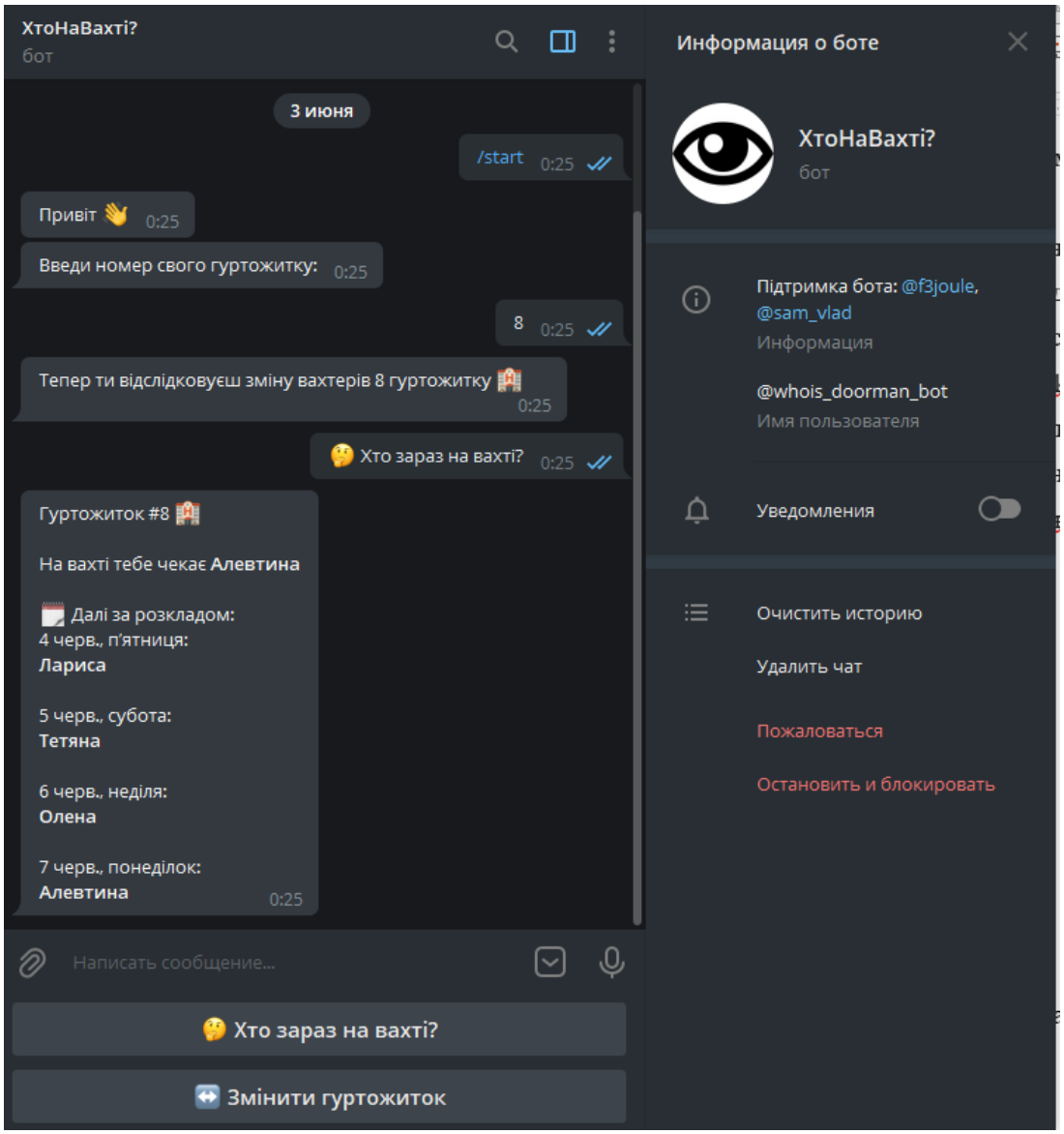


Рис. 1.5. Демонстрація роботи бота *@whois_doorman_bot*

4) Назва: *FinderMusic*.

Ім’я облікового запису: *@fmusbot*.

Призначення: надання безкоштовного доступу до музикальних файлів.

Функції: пошук аудіозаписів на просторах інтернету та їх завантаження; можливість поділитись файлом.

Ціль додатку: дозволити користувачам безкоштовно завантажувати пісні з різних джерел та прослуховувати в off-line режимі.

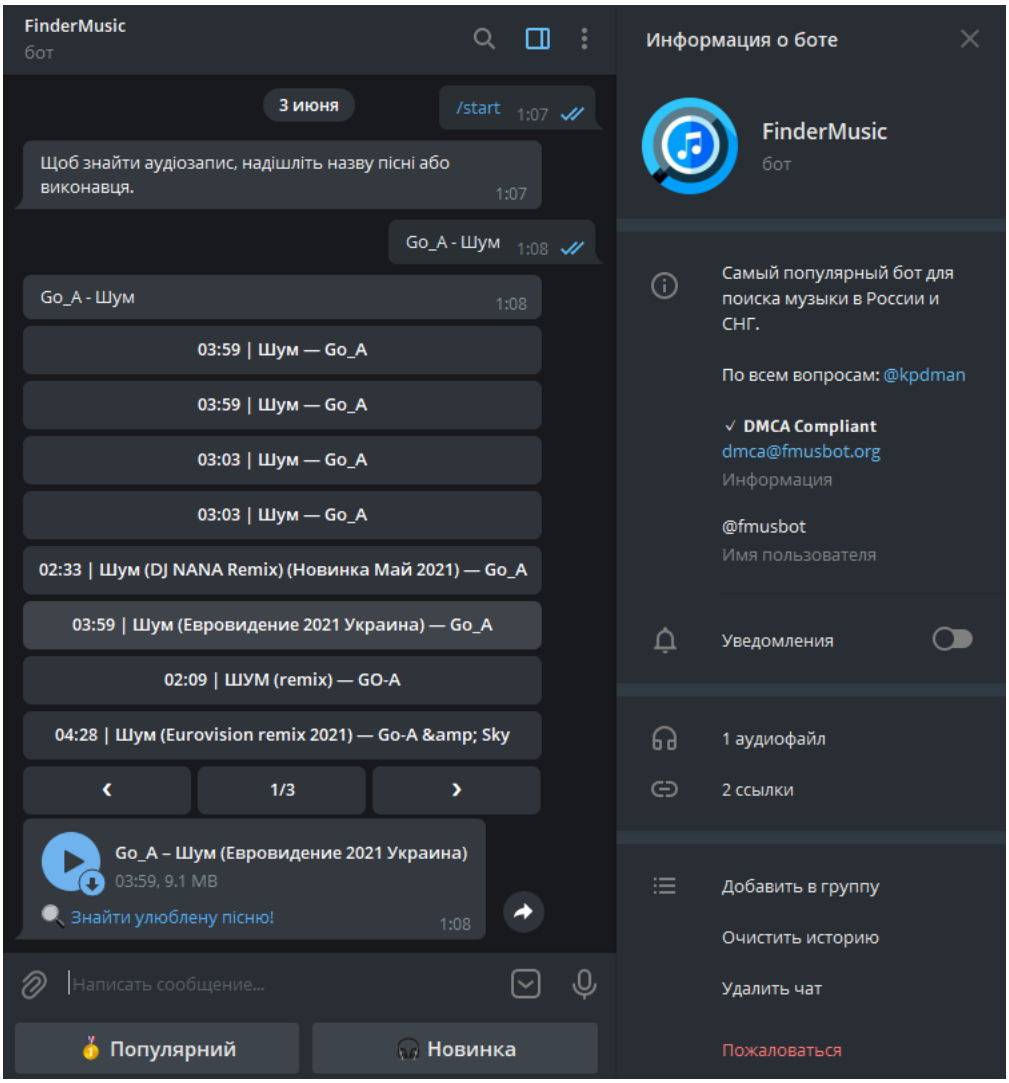


Рис. 1.6. Демонстрація роботи бота *@fmubot*

5) Назва: «Медичний центр “Мій лікар”.

Ім’я облікового запису: *@MLikarBot*.

Призначення: оптимізація роботи медичного центру.

Функції: підписання декларації для безкоштовного огляду без черг; пряме консультування з лікарем за рахунок листування.

Ціль додатку: зменшити навантаження для лікарів та спрощення запису клієнтів на огляд.

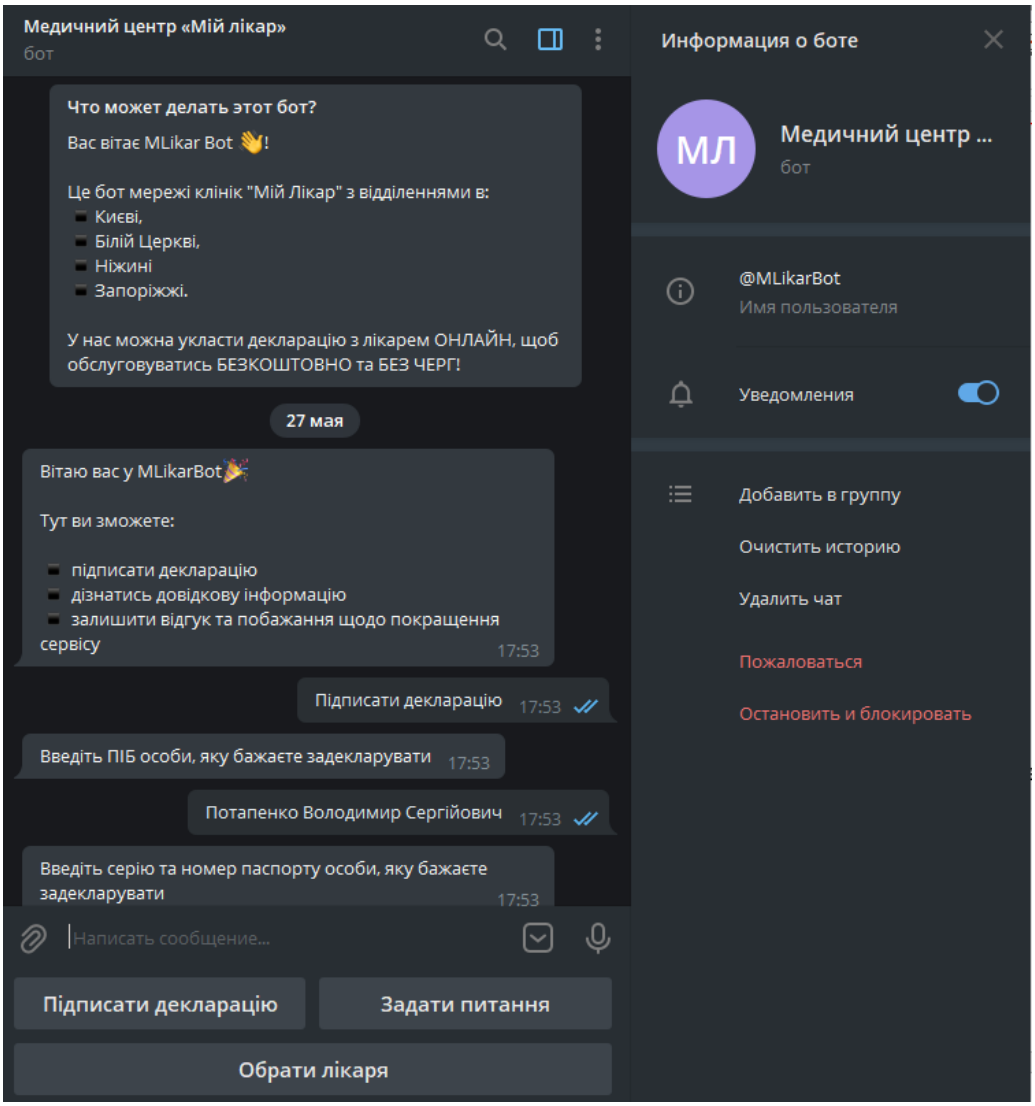


Рис. 1.7. Демонстрація роботи бота *@MLikarBot*

6) Назва: «*RailwayBot*”.

Ім’я облікового запису: *@railwaybot*.

Призначення: надання online доступу для купівлі квитків.

Функції: показ можливих місць у вагоні та їх придбання; моніторинг появи нових квитків; показ розкладу прибуття та відправлення поїздів; повернення білетів.

Ціль додатку: збільшити конверсію продажу квитків; дати можливість користувачам очікувати на повідомлення про появу нових квитків, без потреби постійного моніторингу сайту.

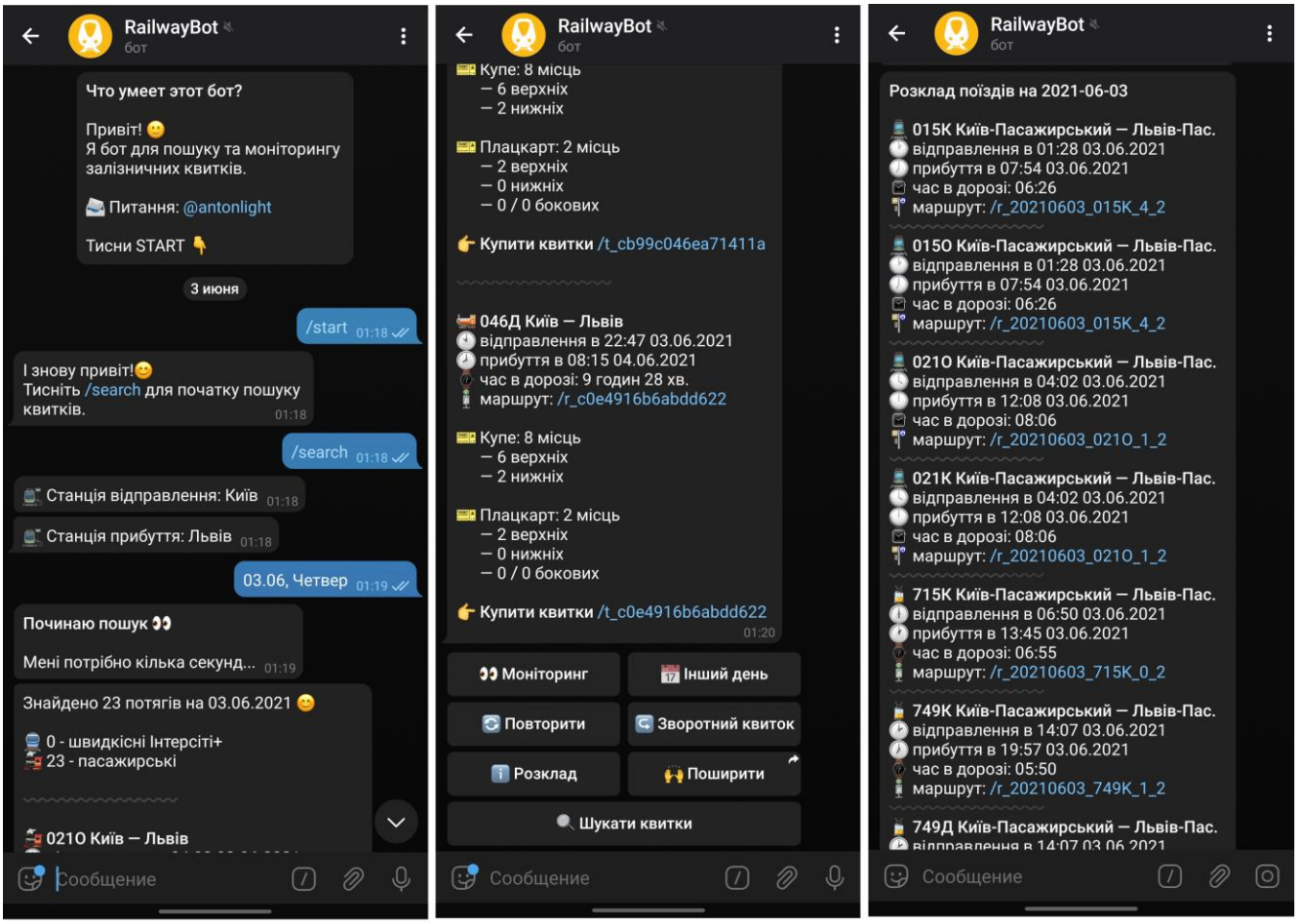


Рис.1.8. Демонстрація роботи бота *@railwaybot*

Висновки до розділу

Чат-боти Telegram збільшують кількість продаж. Завдяки цим додаткам в короткий термін можливо отримати цільову аудиторію. Як показує практика, використання чат-ботів для підтримки клієнтів може зменшити до 35% часу консультантів в онлайн-чатах. Крім того, більшість запитів від користувачів надходять у неробочий час. На відміну від живої людини, бот здатний одночасно давати відповіді декільком користувачам. Більш того, відповідь дається миттєво. Великою перевагою чат-ботів є робота на різних платформах.

Виходячи з перерахованого і розглянутого вище можна зробити висновок, що популярність чат-ботів набирає обертів і компанії, які прагнуть до збільшення ефективності свого бізнесу, все частіше замовляють розробникам нових чат-ботів.

РОЗДІЛ 2
ОПИС МЕТОДОЛОГІЇ РОЗРОБКИ ДОДАТКУ

Створення чат-ботів є важливою темою для обговорень. Хтось бажає створити бота швидко, не використовуючи знання у створенні програмного забезпечення, хтось – вручну, зате безкоштовно. Те, скільки коштуватиме додаток під замовлення, залежить від способу його створення та його функціональних здібностей.

Існує два способи створення чат-ботів: розробка з використанням мов програмування та їх технологій, або зібрати готові рішення за допомогою конструкторів.

Для розробки програми необхідно спершу визначити певні засоби та інструменти які використовуватимуться, а саме:

а) мова програмування;

б) середовище розробки;

в) засоби хостингу додатків;

г) система контролю версій;

д) *CI/CD* системи;

е) додаткові засоби та інструменти.

2.1. Основні поняття мов програмування

Мова програмування - це набір правил, які визначають, як виглядатиме написана комп'ютерна програма, та що комп'ютер може робити під її контролем. Програма – це код, написаний відповідно до правилами даної мови програмування. Код, структура якого утворює програму, називається «вихідним кодом».

Мови програмування – це, фактично, штучно створені засоби спілкування з комп’ютером. Як і естетичні мови, вони мають алфавіт, словниковий запас, граматику та синтаксис, а також семантику.

Алфавіт – дозволений набір символів, за допомогою яких утворюють слова даної мови програмування.

Синтаксис – система правил, що визначають допустимі варіанти створення слів з алфавіту.

Семантика – система правил тлумачення кожної мовної конструкції, що дозволяють описувати процес обробки даних.

Усі мови програмування діляться на два види: мови низького та високого рівня:

– мови низького рівня – це можливість написання комп'ютерних інструкцій на апаратному рівні, щоб бути в машинних кодах (у вигляді комбінації нулей та одиниць). Язики низького рівня жорстко орієнтовані на конкретний тип обладнання (система управління процесором, кожен тип процесора має свій власний машинний код);

– мови високого рівня - це мови програмування, які дозволяють записувати програми в зручній для людини формі. Ці мови орієнтовані не на систему інструкцій того чи іншого процессора, а на системі операторів (інструкцій), характерну для написання визначеного класу алгоритмів.

Мови високого рівня простіші у використанні, бо їх завдання - обслуговувати потреби розробника, а не визначати можливості комп'ютера. Програми, написані на цих мовах, повинні бути переведені на мову машинного рівня. Тому системи програмування на *Java* включає в себе або інтерпретатор мови, або компілятор.

Програми, написані мовою низького рівня, близькі до машинного коду. Це дозволяє створювати програми, які працюють швидше та забезпечують ефективне використання ресурсів комп'ютера.

2.1.1. Перелік мов програмування та їх аналіз

Серед великого переліку мов програмування слід вирішити на які ми будемо розробляти проект, так як кожна мова застосовується для різних завдань. Для вирішення на які саме мові буде розроблятись програмне забезпечення, ми повинні проаналізувати їх популярність, сфери використання та ефективність.

На рис. 2.1 продемонстровано графік популярності мов програмування в комерційних проектах станом на 2021 рік.

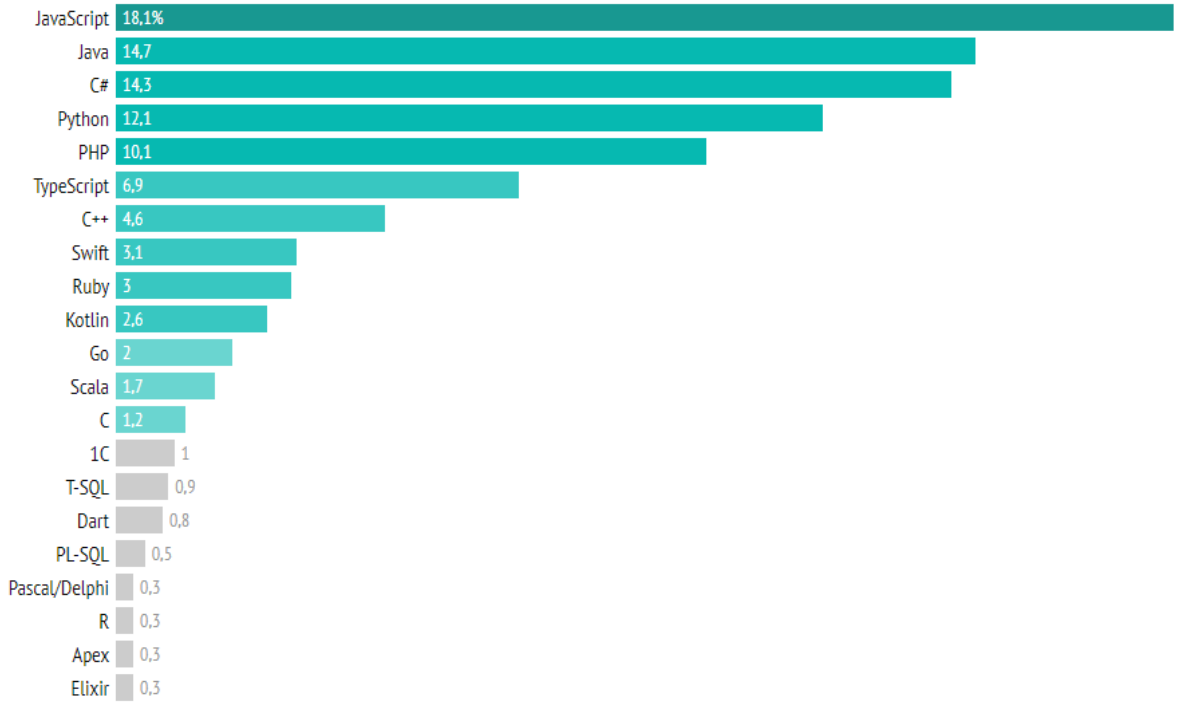


Рис. 2.1. Оцінка мов програмування у реальних проектах, %

Як бачимо, *JavaScript* займає перше місце серед мов програмування. У п’ятірку кращих також входять такі мови, як: *Java, C#, PHP*. На рис.2.2 показано діаграму змін популярності з 2012 по 2021 роки.

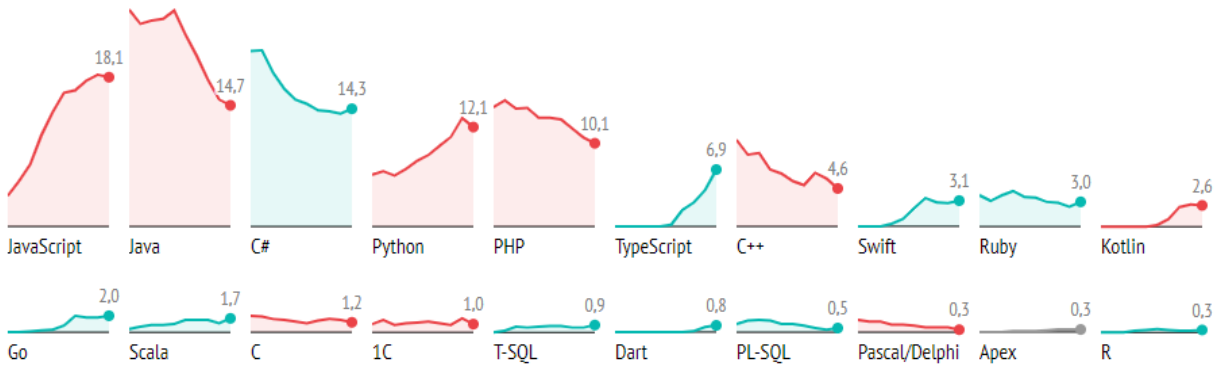


Рис. 2.2. Діаграма зміни популярності за період від 2012 по 2021 роки, %

По даній діаграмі ми можемо зробити висновки, що популярність таких мов, як *Java, C#* та *PHP*, стрімко падає. До числа швидко зростаючих у використанні мов входять: *JavaScript, Python, TypeScript*, саме для них ми проведемо аналіз.

2.1.2. Порівняння мов програмування, які набирають популярність

JavaScript – це мова для розробки програмного забезпечення, яка є адаптивною під різні сфери застосування. Також вона займає перше місце серед інших мов для розробки інтерфейсів взаємодії користувача (*Front-end*) з програмно-апаратною частиною сервісу (*Back-end*), простіше кажучи, для розробки візуальної частини додатку.

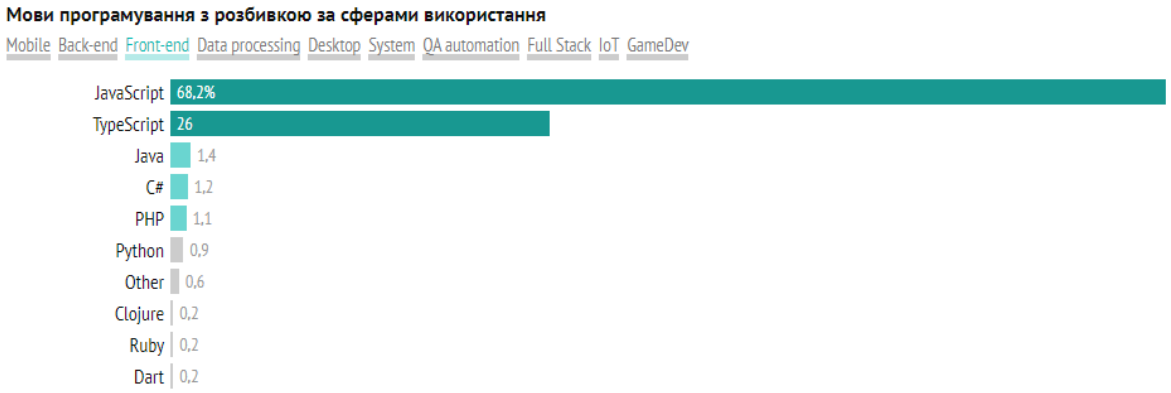


Рис. 2.3. Графік популярності мов програмування для написання *Front-End* частини, %

Переваги *JavaScript*:

– кожен сучасний браузер користується підтримкою *JavaScript*;

– простота написання скриптів;

– постійне оновлення мови – розробляється бета-версія проекту, під назвою JavaScript2;

– перспектива використання мови під час навчання;

– можливість редагувати код навіть через текстовий редактор.

Недоліки *JavaScript*:

– відкрити доступ до вихідного коду, результатом чого є низький рівень безпеки;

– при зупинці підтримки даної мови більшість проектів не будуть мати можливість подальшого розвитку, так як структура програми базується на *JavaScript*;

– не притаманна для розробників об'єктна модель. Класи та наслідування класів відрізняє від привичної реалізації ніж на *C++*, *C#, Java*;

– відсутність можливості перевірки роботи інших блоків, допоки є помилка на початку коду. Це спричинено відсутністю типізації даних.

C# – це мова програмування, структура якої складається із об'єктно-орієнтованої та контекстно-орієнтованої концепції.

Переваги *C#*:

– подібність синтаксису до *C++*, *Java, Basic*;

– об'єктно-орієнтований підхід до розробки проектів;

– широка сфера застосування;

– адаптація під різні платформи;

– безпека, в порівнянні з *C* та *C++*;

– швидкість роботи;

– зручність при написані коду.

Недоліки *C#*:

– відсутність багато потокової компіляція;

– складний синтаксис;

– відсутність готових рішень для ваших ідея, спричинених безпекою коду;

– базується під операційну систему *Windows*;

– відсутність можливості створення інтерфейсу під різні платформи.

Python – це мова програмування високого рівня, яка застосовується для вирішення різних видів задач. Хоч і не швидкий у виконанні програм, зате простий у вивченні та розумінні коду «на льоту», якщо при цьому розробник слідує правилам написання коду *PEP*.

PEP (Python Enhanced Proposal) – це набір правил, який мотивує до написання «чистого» коду з подальшим розумінням іншим розробником.

Також серед обробки даних беззаперечним лідером є *Python* (рис. 2.4).

Переваги *Python*:

– простота та швидкість у написанні коду;

– легко вивчається;

– великі можливості для застосування;

– велика кількість бібліотек для спрощення розробки проекту;

– зручний для роботи з великою кількістю даних;

– доступний у відкритому виді код;

– абсолютно все являє собою об’єктами ООП, а змінні – це лише посилання на них, при цьому використання об’єктного підходу для написання коду не обов’язкове;

– можливість розробки під різні платформи.

Недоліки *Python*:

– низька швидкодія;

– відсутність статичної типізації;

– відсутність створення приватних змінних (створення відбувається за допомогою додавання символу “_” перед змінною, наприклад: “*_privat_var*”);

– не зовсім правильна підтримка роботи багатьох потоків;

– мала кількість готових програм, написаних на *Python*, порівнюючи з *Java*;

– низька продуктивність.

Для порівняння продемонструємо написання коду для виведення надпису «*Hello, world*!» на консолі на рис. 2.4, рис.2.5, рис.2.6 на мовах програмування C, *Java*, *Python* відповідно.

```c
#include <stdio.h>
int main(int argc, char** argv)
{
    printf("Hello, World!")
}
```

Рис. 2.4. Код на мові C

```java
class HelloWorldApp {
    public static void main(String[] args) {
    System.out.println("Hello, World!"); // Вивід рядка у консоль
    }
}
```

Рис. 2.5. Код на мові Java

```python
print("Hello, World!")
```

Рис. 2.6. Код на мові Python

Важко не помітити, що для виконання однієї і тої самої задачі *Python* займає лише 1 рядок, в той час як *C* та *Java* – 5 рядків.

2.2. Середовище розробки

Для створення та редагування коду розробникові потрібно обрати комплекс програмних засобів, або іншими словами – інтегроване середовище розробки (англ. *Integrated development environment – IDE*). Такими можуть бути:

- текстовий редактор;
- транслятор (компілятор або інтерпретатор);
- засоби автоматизації збірок;
- відкладчик.

При потребі швидко відкрити та редагувати код ми можемо скористатись текстовими редакторами – це комп'ютерна програма або компонент програмного комплексу, призначений для створення та редагування текстових даних. Вони призначені для інтерактивного режиму роботи з файлами, тобто «відкрив, дописав, сберіг». Текстові редактори подібні до стандартної програми «Блокнот», але звісно ж мають в собі додаткові функції для роботи з кодом на різних мовах програмування, такі як: підкреслення або виділення тексту для простоти перегляду (відокремлює функції мов програмування від змінних), автодоповнення рядків, створення декількох вікон для роботи з кодом та інше.

Sublime Text editor – один з найкращих та найшвидших текстових редакторів. Відмінна альтернатива сильним *IDE*, вона легка для користування та ефективно виконує своє призначення.

Переваги *Sublime Text*:

- зручний інтерфейс;
- можливість встановлювання плагинів;
- приємна смуга для прокрутки з правої сторони;
- можливість створення декількох курсорів для зручного редагування;

– підсвічування та автодоповнення коду;

– перевірка граматики під більшість мов програмування;

– гнучке налаштування шрифтів та широкий вибір кольорових схем.

Недоліки *Sublime Text*:

– зависання програми при використанні багатьох плагинів.

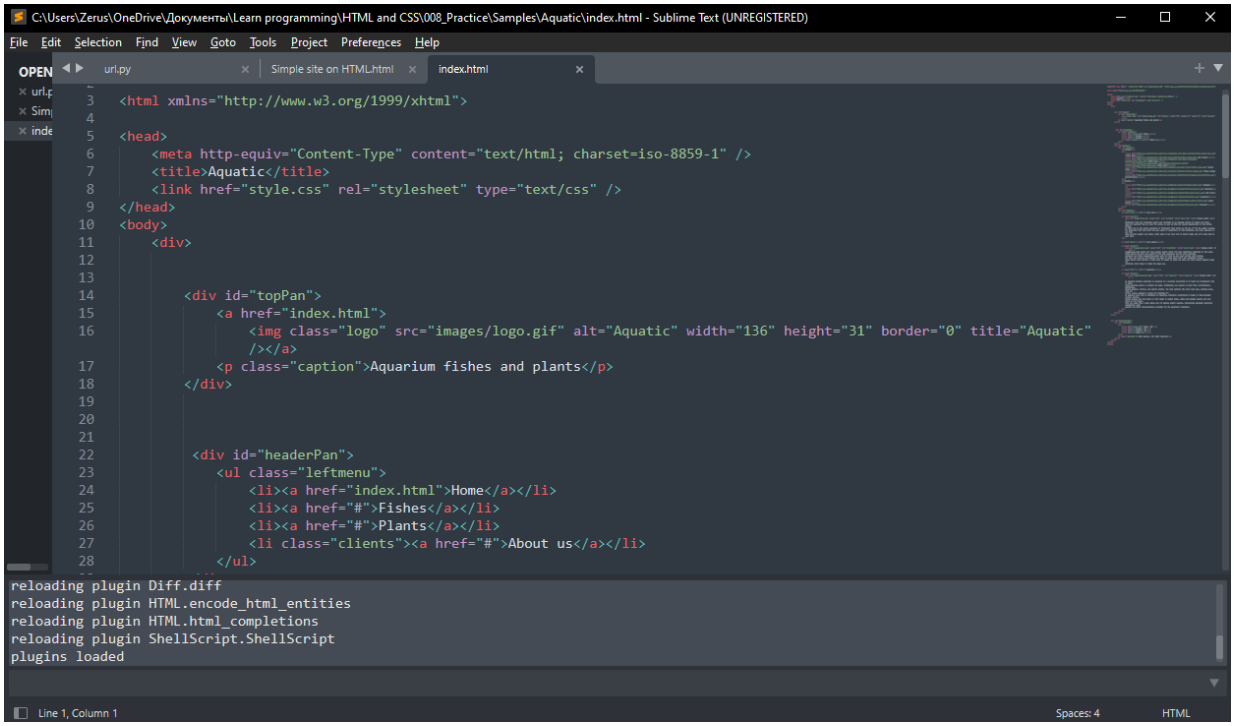


Рис. 2.7. *Sublime Text*

Транслятори можуть містити в собі все те, що має текстовий редактор, а також засоби для інтеграції з системами контролю версій та різні інструменти для легкого конструювання графічного інтерфейсу користувача (*GUI*). Більшість таких сучасних середовищ розробок містять в собі браузер класів, інспектор об’єктів та діаграму ієрархії класів – у процесі розробки програмного забезпечення на базі об’єктно-орієнтованої методології. Зазвичай вони призначені для роботи з декількома мовами програмування, такі як: *EntelliJ IDEA, NetBeans, Eclipse, Qt Creator, Geany, Embarcadero RAD Studio, Code::Blocks, Xcode або Microsoft Visual Studio,* але є і *IDE* для однієї певної мови програмування - як, наприклад, *Visual Basic, Delphi, Dev-C ++, PyCharm*.

Транслятори поділяються на два види:

– інтерпретатор – аналізує код послідовно по рядку зверху вниз та зупиняє виконання програми після знаходження першої помилки;

– компілятор – повністю аналізує код, складає звіт помилок та рекомендації для їх виправлення, після чого, в разі відсутності помилок, запускає програму.

Відкладчик – це застосунок, який автоматизує процес відкладки: пошуку помилок в інших програмах, ядрах операційних систем та інших видів коду. В залежності від вбудованих можливостей, відкладчик дає можливість відслідковувати, встановлювати та змінювати значення змінних під час виконання коду, встановлювати та видаляти контрольні точки або умови зупинки відкладки.

2.2.1. Аналіз середовищ розробки та мов програмування

В рамках дипломного проекту є ідея розробки чат-бота в месенджері Telegram, який буде надавати можливість продажу власного продукту.

Функціональність *Telegram*-бота можна реалізувати різними шляхами, кожен з яких має свої переваги. Створення програми виконується досить просто наступними способами:

1) З використанням продукту компанії *Microsoft – Visual Studio*. В якості мови програмування в цьому випадку раціонально буде вибрати *C #* або *Node.js*. При цьому, вам не потрібно буде писати з нуля великі масиви коду - набагато простіше скористатися готовим набором шаблонів, таких як *Bot Application, Bot Framework Emulator, Bot Dialog і Bot Controller*. Всю необхідну інформацію про їх встановлення та використання містить документація *Microsoft Bot Framework*, вам залишиться тільки налаштувати ці шаблони під свої вимоги. Фактично, такий варіант підходить в повній мірі тільки для користувачів *Windows*.

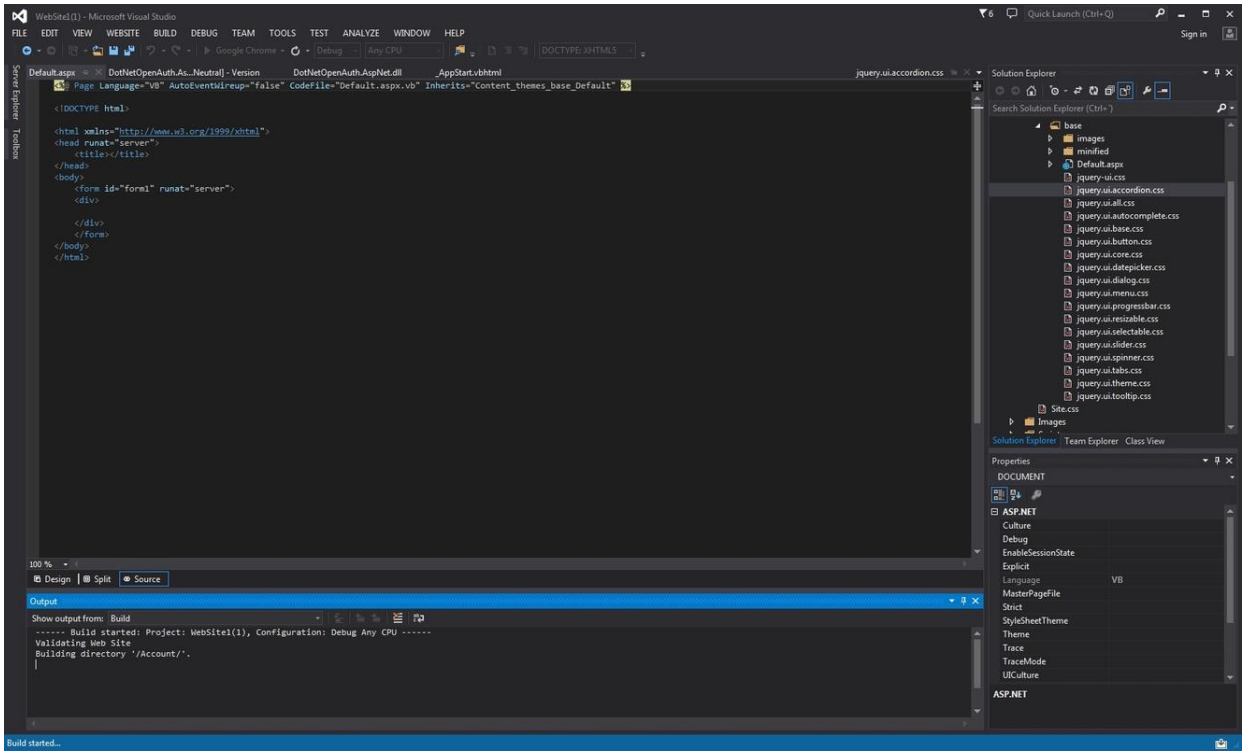


Рис. 2.8. *Visual Studio*

Переваги *Visual Studio*:

– підтримка більше ніж 30 мов програмування, а також основні мови компанії *Microsoft*: *ASP.NET, C#* і т.д.;

– можливість вибору завантаження компонентів відповідно під вашу сферу застосування.

Недоліки *Visual Studio*:

– підтримка розширень, які потребують покращення;

– конфлікту з ОС *Linux*.

2) Серед інших варіантів для створення *Telegram* бота, мова *PHP* також дає чимало можливостей. Перш за все, варто мати на увазі велику популярність цієї мови програмування - якщо щось піде не так, завжди знайдеться з ким проконсультуватися. Також підкуповує і наявність вже готових бібліотек для роботи з Телеграм, таких, наприклад, як *Telegram Bot SDK*, що дозволяє звести ваші зусилля до мінімуму.

Для зручності розробники обирають інтегроване середовище розробки *NetBeans*. Це *IDE* підтримує більшість мов програмування та підтримує роботу з базами даними, реалізована система рефакторінгу коду.

Переваги *NetBeans*:

– встановлюй та відразу користуйся. На відміну від *Visual Studio*, цей редактор не потребує до завантаження ресурсів;

– насправді відкритий код, який дає можливість переглянути та внести правки;

– можливість спільної роботи у великій компанії багатьом розробникам одночасно в рамках одного проекту;

– широка підтримка мов програмування.

Недоліки *NetBeans*:

– потребує багато ресурсів пристрою.

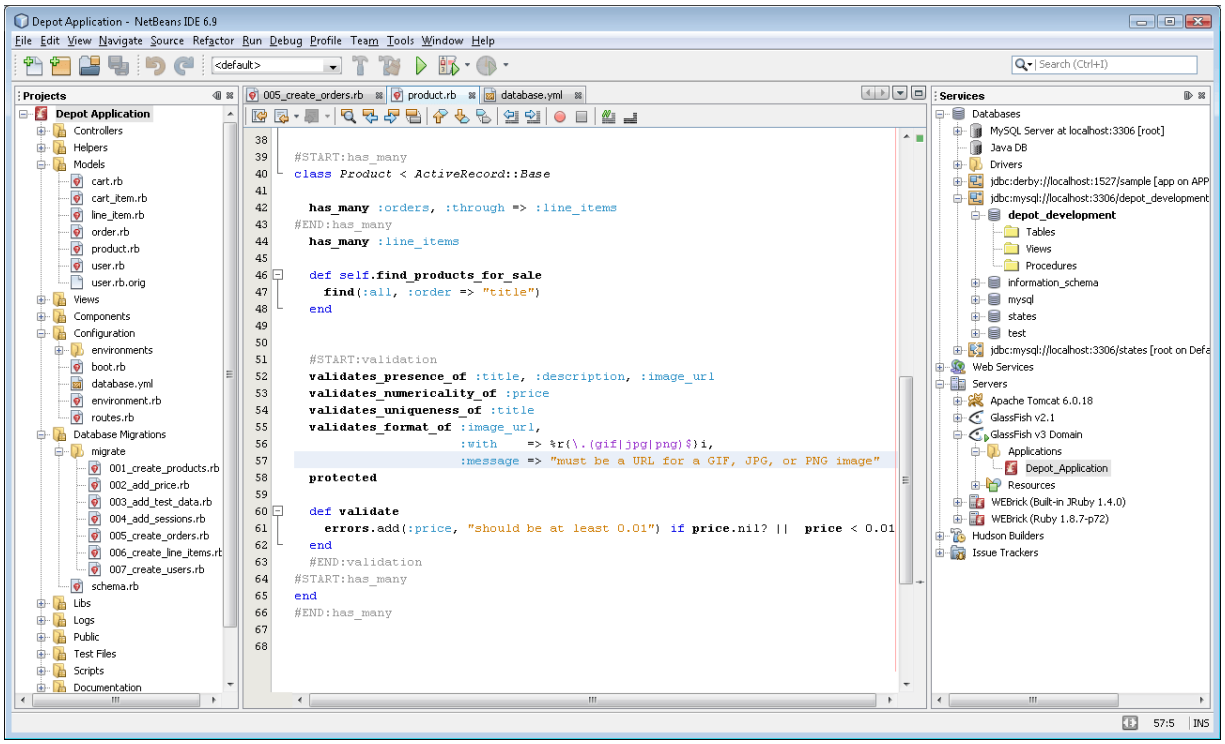


Рис. 2.9. *NetBeans*

3) Нарешті, серед мов програмування, що дозволяють швидко та без проблем виконати створення бота в телеграмі, Python є одним з найбільш популярних рішень. В основному це пов'язано з широкими можливостями, доступними як при використанні стандартних бібліотек, так і зі застосуванням вже готових варіантів, таких як *PyTelegramBotAPI*, розрахованих на роботу безпосередньо з *Telegram*. Також для цієї мови створено велика кількість бібліотек, які сильно спрощують написання шаблонного коду.

Більшість *IDE* підтримують мову *Python*, але серед них хочеться виділи *PyCharm*.

PyCharm – одне з найкращих середовищ розробки від компанії *JetBrains*, яке спеціалізується саме на створенні проектів мовою *Python* та розкриває весь потенціал цієї мови, а також підтримує *JavaScript*, *CoffeeScript*, *TypeScript*, *SQL*, *HTML*/*CSS*, мови шаблонів, *AngularJS*, *Node.js* та інші технології. Зрівнюючи з *Visual Studio*, в цьому *IDE* немає проблем роботи з операційною системою *Linux*.

Переваги *PyCharm*:

– зрозуміла концепція системи контролю версій;

– простота в організації проектів;

– зручне автодоповнення;

– хороша інтеграція з популярними модулями;

– можливість безкоштовно отримати ліцензію для версії *PyCharm Professional* за допомогою пошти, отриманої від вашого навчального закладу;

– детальний налагоджувач, який продемонструє як працює код;

– швидкість завантаження проектів та переключення між ними;

– робота з базами даних;

– підтримка різних операційних систем.

Недоліки *PyCharm*:

– на безкоштовній версії *Pycharm Community* ви не зможете працювати з іншими мовами програмування.

2.3. Вибір бази даних

Перш за все для того, щоб вхід дані зберігались безпечно та надійно, ми повинні проаналізувати програмне забезпечення, яке дасть можливість структуровано зберігати їх.

2.3.1. *SQLite* як спосіб зберігання даних

Для зберігання даних, які необхідні для коректної роботи чат-бота була обрана база даних *SQLite*.

SQLite – це компактна інтегрована реляційна база даних, вихідний код бібліотеки якої було передано в суспільне надбання. це чисто реляційна база даних.

Підтримувані типи даних:

– *NULL*: значення *NULL*.

– *INTEGER*: ціле число зі знаком, що зберігається в 1, 2, 3, 4, 6 або 8 байтах.

– *REAL*: число з плаваючою комою, що зберігається в 8-байтовому форматі IEEE.

– TEXT: текстовий рядок, закодована *UTF-8, UTF-16BE* або *UTF-16LE*.

– *BLOB*: тип даних, що зберігаються в тій же формі, в якій вони були отримані

Переваги:

– файлова система: вся база даних зберігається в одному файлі, що полегшує переміщення;

– стандартизована: *SQLite* використовує *SQL*;

– дуже добре підходить для розробки і навіть тестування: на етапі розробки більшості потрібно масштабується. *SQLite* з його багатим набором функцій може надати більш ніж достатню функціональність, а також бути досить простим для роботи з одним файлом і пов'язаної бібліотекою;

Недоліки:

– відсутність призначеного для користувача управління: просунуті БД надають користувачам можливість управляти зв'язками в 33 таблицях відповідно до привілеями, але у *SQLite* такої функції немає;

– неможливість додаткової настройки: *SQLite* не можна зробити продуктивнішою;

– в даному підрозділі була представлена база даних, необхідна для роботи чат-бота, було розглянуто її поняття, переваги і недоліки.

2.3.2. PostgreSQL

PostgreSQL — широко розповсюджена система керування базами даних з відкритим кодом.

Функції

Функції дозволяють виконувати деякий код безпосередньо сервером бази даних. Ці функції можуть бути написані на *SQL*, який має деякі примітивні програмні оператори, такі як галуження та цикли. Але гнучкішою буде функція написана на одній із мов програмування, з якими *PostgreSQL* може працювати.

До таких мов належать:

- вбудована мова, яка зветься *PL/pgSQL*, подібна до процедурної мови *PL/SQL* компанії *Oracle*;
- мови розробки сценаріїв: *PL/Perl, PL/Python, PL/Tcl, PL/Ruby, PL/sh*;
- класичні мови програмування *C, C++, Java* (за допомогою *PL/Java*).

Функції можуть виконуватись із привілеями користувача, який її викликав, або із привілеями користувача, який її написав.

- типи даних;
- *postgresql* підтримує великий набір вбудованих типів даних:
- числові типи;
- цілі:
    - з фіксованою крапкою;
    - з нефіксованою крапкою;
- грошовий тип;
- символьні типи довільної довжини;

– двійкові типи (включаючи *blob*);
– типи «дата/час»;
– булевий тип;
– перерахування;
– геометричні примітиви;
– мережеві типи:
  - *ip* і *ipv6*-адреси;
  - *cidr*-формат;
  - *мас*-адреса;
– *uuid*-ідентифікатор;
– *xml*-дані;
– *json*-дані;
– масиви;
– *oid*-типи;
– псевдо типи;
– крім того, користувач може самостійно створювати нові необхідні йому типи та програмувати для них механізми індексування за допомогою gist.

Переваги *PostgreSQL*:

– відкрите ПЗ відповідає стандарту *SQL* - *PostgreSQL* - безкоштовне ПЗ з відкритим вихідним кодом. Ця СУБД є дуже потужною системою;

– велике співтовариство - існує досить велика спільнота в якому ви запросто знайдете відповіді на свої питання;

– велика кількість доповнень - незважаючи на величезну кількість вбудованих функцій, існує дуже багато доповнень, що дозволяють розробляти дані для цієї СУБД і управляти ними;

– розширення - існує можливість розширення функціоналу за рахунок збереження своїх процедур;

– об'єктно - *PostrgreSQL* це не тільки реляційна СУБД, але також і об'єктно-орієнтована з підтримкою успадкування і багато іншого.

Недоліки *PostgreSQL*:

– продуктивність - при простих операціях читання *PostgreSQL* може значно уповільнити сервер і бути повільніше своїх конкурентів, таких як *MySQL*;

– популярність - за своєю природою, популярністю ця СУБД похвалитися не може, хоча і є досить велика спільнота;

– хостинг - в силу названих вище чинників іноді досить складно знайти хостинг з підтримкою цієї СУБД.

2.4. Аналіз платформ для хмарних обчислень

Звичайно, коли вам потрібен повноцінно працюючий *Telegram* бот, створення його виявиться тільки половиною необхідної роботи. Другим обов'язковим етапом стане забезпечення його безперебійної роботи, для цього необхідно підібрати надійний і доступний за ціною сервер. На щастя, існує досить чимало цілком придатних безкоштовних рішень, таких як *Azure, Heroku* та *Google Cloud Platform*, однак, якщо вас цікавить необмежена функціональність, то варто віддати перевагу платним хмарним сервісам.

2.4.1. *Microsoft Azure*

Microsoft Azure – хмарна платформа від компанії Microsoft. Надає можливість розробки, виконання додатків і зберігання даних на серверах, розташованих в розподілених дата-центрах.

Microsoft Azure реалізує хмарні моделі платформи як сервісу (*PaaS*) та інфраструктури як сервісу (*IaaS*). Можливе використання як сторонніх, так і сервісів *Microsoft* в якості моделі ПО як сервісу (*SaaS*) . Працездатність платформи *Microsoft Azure* забезпечує глобальна мережа розподілених дата-центрів *Microsoft*.

Крім базових функцій операційних систем, *Microsoft Azure* має і додаткові: виділення ресурсів на вимогу для масштабування, автоматичну синхронну

реплікацію даних для підвищення надійності від відказів, обробку відмов інфраструктури для забезпечення постійної доступності та інше.

Модель надання інфраструктури (*IaaS*) реалізує можливість оренди таких ресурсів, як сервери, пристрої зберігання даних та мережеве обладнання. Управління всією інфраструктурою здійснюється постачальником, споживач управляє тільки операційною системою та встановленими додатками.

Для віртуальних машин доступні образи наступних операційних систем: *Windows Server, CoreOS, Ubuntu Server, Red Hat, Clear Linux OS, Debian, SUSE Linux Enterprise Server, Oracle Linux*

Практично всі сервіси *Microsoft Azure* мають інтерфейс взаємодії *API*, побудований на основі обмежень для розподілених систем *REST*, що дозволяє розробникам використовувати хмарні сервіси з будь-якою операційною системи, пристроїв і платформ.

Крім того, користувачі можуть створювати та керувати власними сервісами, користуючись візуальним веб-інтерфейсом порталу *Azure*. Портал дозволяє налаштовувати сервіси, редагувати права доступу, відстежувати стан ресурсів і управляти біллінгом.

2.4.2. Heroku

Heroku - це рішення *PaaS*, що належить *Salesforce*, для розробки та запуску програмних додатків на віддалених серверах. Спочатку побудований для веб-програми *Ruby on Rails. Heroku* тепер підтримує інші мови програмування, такі як *Java*, *Python* та *Node.js* так само.

Деякі ключові терміни *Heroku* включають:

– *Heroku dynos* - це віртуалізовані контейнери, що використовують операційну систему *Linux*, які є «будівельними блоками» веб-додатків *Heroku*, пропонуючи гнучкість та масштабованість. Контейнери - це абстракції, які знімають складність управління базовим обладнанням або віртуальними машинами;

– збірник - це сценарій автоматизації збірки, який визначає спосіб створення образу контейнера;

– додатки - це інструменти та служби, які розширюють функціональність програми Heroku;

– інтерфейс командного рядка (*CLI*) - це інструмент *Heroku* на основі терміналу для створення та запуску програм *Heroku*;

– *Git* - це система контролю версій програмного забезпечення для відстеження змін бази даних з часом. Користувачі *Heroku* можуть розгортатись із *Git* за допомогою інтерфейсу командного інтерфейсу *Heroku*, включаючи платформу хостингу *GitHub* для сховищ *git*.

2.4.3. Google Cloud Platform

Google Cloud Platform – це комплекс хмарних платформ від компанії *Google*, яка надає можливість підключення модульних служб для роботи з вашим проектом, таких як хмарні обчислення, зберігання даних, аналіз даних та машинне навчання.

Правда сервер на цій платформі надається тільки один, зате він набагато доступніший ніж аналогові хмарні платформи.

Крім безкоштовних, але в той час і обмежених можливостей, компанія Google надає 3-місячний пробний період з кредитними коштами у розмірі 300$, які ви можете витратити на любу продукцію цієї компанії для розкриття їх потенціалу. По закінченню цього періоду, якщо ви не перейдете на платну підписку, то всі сервіси будуть призупинені, а всі дані користувача будуть стерті. В любому випадку ніяких списань з вашої банківської карти без вашого відома не буде.

Переваги:

– швидкий вхід та вихід даних;

– широкий спектр регіонів для розміщення вашого серверу;

– чудова інтеграція з продуктами компанії *Google*;

– компанія ділиться своїми навиками в аналітиці та зберіганні даних;

Недоліки:

– більшість продуктів знаходяться на стадії бета-тестуванні;

– підходить великим проектам, а ніж початковим бізнес ідеям.

Висновки до розділу

У цьому розділі було розглянуто засоби для розробки чат-ботів, а саме:

– мови програмування та середовища розробок для роботи з ними;

– бази даних для зберігання інформації;

– платформи, за допомогою яких можливо надати чат-ботам працювати в автономному режимі.

Був проведений аналіз програмного забезпечення для комфортної роботи з проектом, по результатам якого було обрано наступні засоби:

– *Python*, як мова програмування для написання чат-ботів;

– *PyCharm* як середовище розробки;

– *SQLite3* як спосіб для зберігання даних;

– *Google Cloud Platform* як хмарна платформа.

Отже, підсумовуючи розділ, ми готові приступити до розробки ідеї дипломного проекту.

РОЗДІЛ 3
ПРАКТИЧНА РЕАЛІЗАЦІЇ TELEGRAM-ДОДАТКУ

Метою даної роботи є створення бота на широко відомій платформі Telegram, так як цільова аудиторія даного месенджера досягає 500 мільйонів чоловік по всьому світі. Бот повинен використовуватися в області продажу продуктів. У його функціонал буде входити наступний перелік можливостей:

- запропонувати клієнтові вибір серед списку асортименту;
- пошук у базі даних інформації про товар;
- оформлення купівлі через електронний гаманець;
- можливість перегляду продукту після успішної купівлі;
- панель керування адміністратором в інтерактивному режимі;
- додавання нового адміністратора;
- додавання та завантаження нового товару;
- масове повідомлення у декілька натисків;
- тех. підтримка;
- показ статистики.

3.1. Проектування чат-боту

Для зручності розробки було розроблено інтелектуальну схему, яку зображено на рис. 3.1, яка показує розгалуження подій.

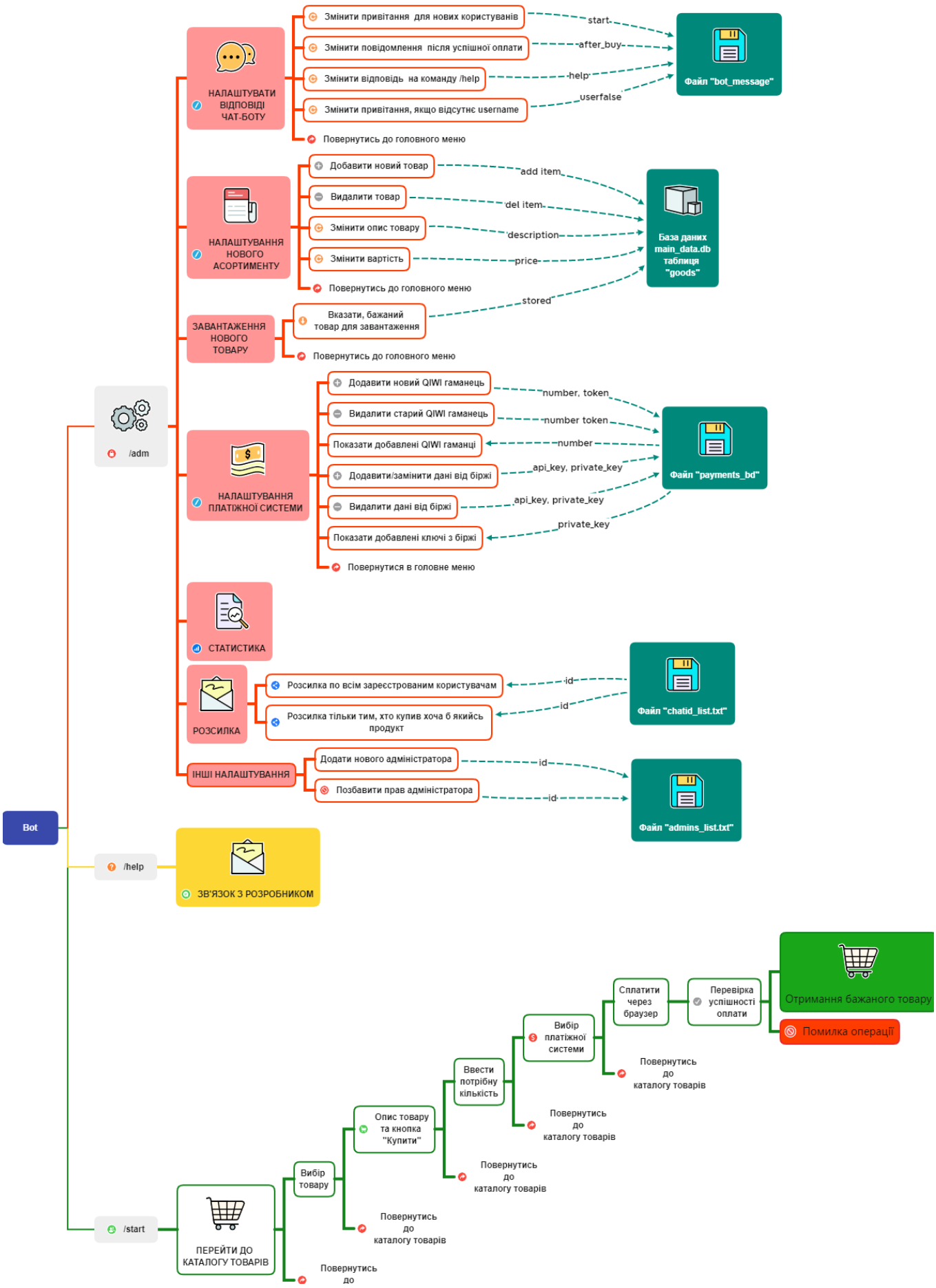


Рис. 3.1. Інтелектуальна схема

Як можемо замітити, на схемі показано три головних команди, за якими лежить подальші варіанти подій:

• */adm* – перехід до панелі керування налаштуваннями бота. Доступ тільки з правами адміністратора. Дає можливість редагування відповідей, які надсилаються користувачам, налаштування асортименту та платіжної системи, перегляд статисти, розсилання певного повідомлення всім користувачам або тільки тим, хто став покупцем (придбав хоча б 1 товар), додавання або позбавлення прав адміністратора;

• */help* – команда, яка являється помічником у користуванні, яка показує список всіх доступних команд, а також контакти для зворотного зв'язку з розробником;

• */start* – призначена для всіх користувачів, які бажають почати роботу з чат-ботом. Якщо притримуватись лінійного алгоритму, ми отримаємо змогу переглянути каталог товарів та їх опис. При бажанні отримати цей продукт, у клієнта запитують бажану кількість та платіжну систему, якою він бажає розрахуватись. На даний момент працює тільки можливість оплати через платіжну систему *QIWI*, при цьому не обов'язково мати обліковий запис чи гаманець у цій системі, варто лише скористатись картою любого банку *VISA* чи *MasterCard*.

Щодо бази даних, то вона буде виглядати так, як показано на рис. 3.2.

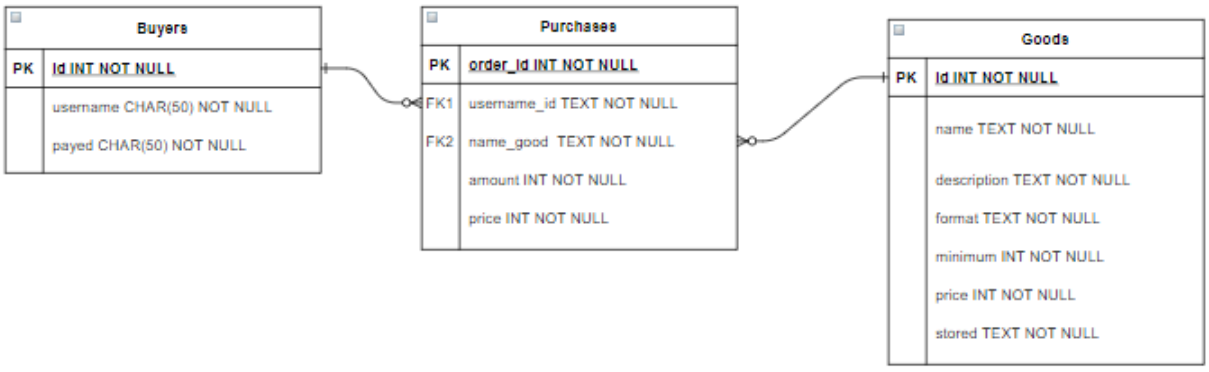


Рис. 3.2. Схема бази даних

3.2. Створення чат-бота для продажу товарів

Боти в *Telegram* не тільки мають більші можливості, а й є зручними у використанні. Для цього команда розробників *Telegram* створила бота, який є батьком для всіх ботів, через якого здійснюється створення нових чат-ботів та керування вже існуючими.

Така організація роботи дає велику перевагу, так як їх створення та управління є зрозумілим і нескладним процесом. До того ж з року в рік зростає кількість користувачів даного месенджера. За словами творця *Telegram* Павла Дурова, цей месенджер в 2021 році став дуже успішним, так як його обирало величезна кількість людей через відсутність реклами. А 2022 рік має стати ще більш продуктивним, тому що в месенджер будуть впроваджуватися нові функції.

3.2.1. Створення облікового запису для чат-бота

Для створення персонального бота ми повинні відправити команду /*newbot*, після чого *BotFather* запитає яку назву та логін ви хочете йому задати, за яким його можливо буде знайти (*username*). Після цих простих кроків, у разі унікальності логіну, бот відправить вам *API* токен, за допомогою якого ви зможете відправляти *HTTP* запити для надання інструкцій роботи бота.

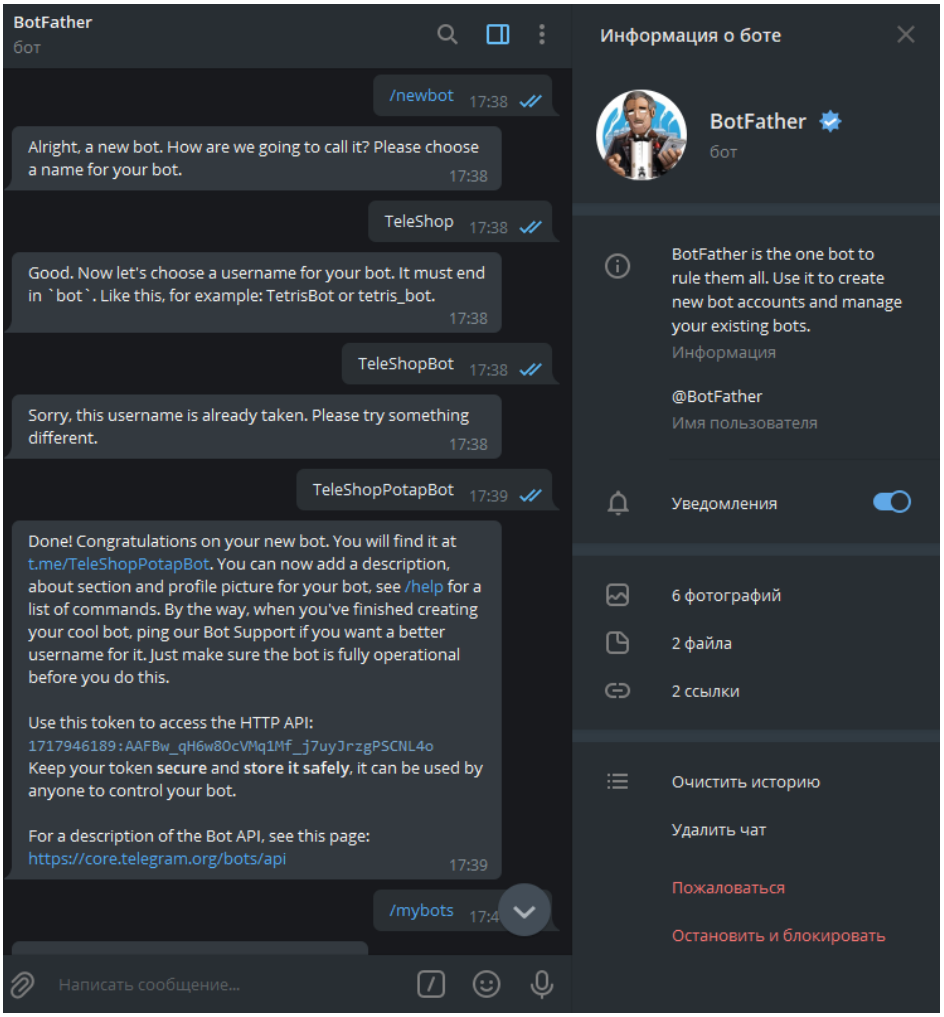


Рис. 3.3. Створення нового боту через *@BotFather*

Bot API - це інтерфейс на основі *HTTP*, створений для розробників, які прагнуть створювати ботів для *Telegram*.

API Токен – це унікальний ключ до вашого чат-боту. У разі втрати ключа вашим ботом зможуть користуватись сторонні люди. Для запобігання цього ви можете написати знову ж таки BotFather та створити новий ключ звернення.

Після створення ви маєте можливість корегувати вигляд чат-боту, а саме:

- зміна імені (рис.3.4);
- зміна домену (веб-логіну);
- завантажити картинку на заставку (рис.3.4);
- додати опис та додаткову інформацію (рис.3.4);
- редагування список команд, які висвічують користувачеві (рис. 3.5).

Додавання опису та додаткової інформації допоможе користувачам одразу зрозуміти призначення бота та правила користування.

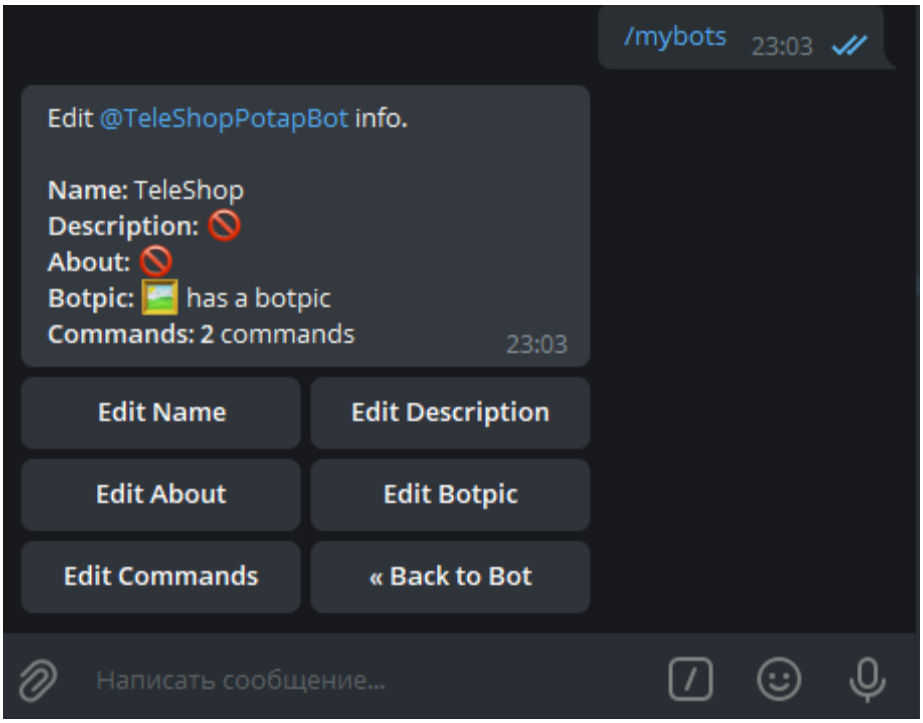


Рис. 3.4. Редагування вашого бота

Якщо ви додасте у параметр «*Edit Commands*» параметр команд та їх опис, то вони будуть показуватись при написанні знаку «/» у чаті з вашим ботом. Саме через цей знак відбувається виклик команд.

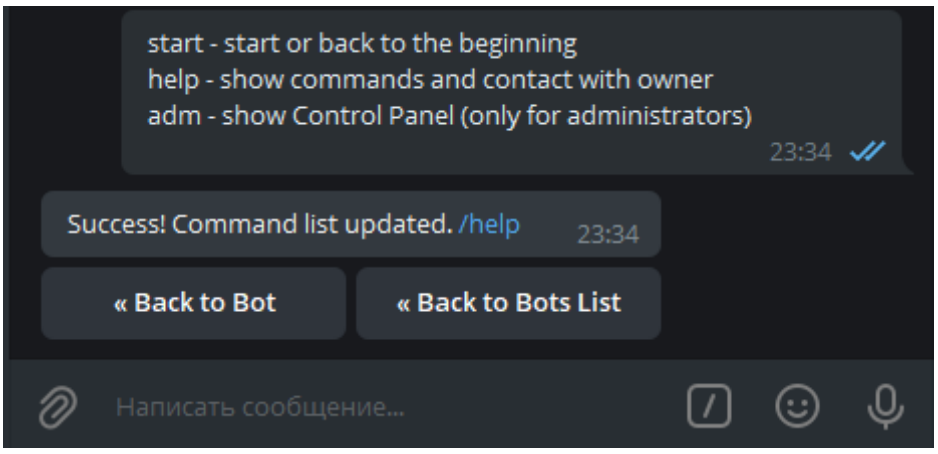


Рис. 3.5. Додавання команд для підказок

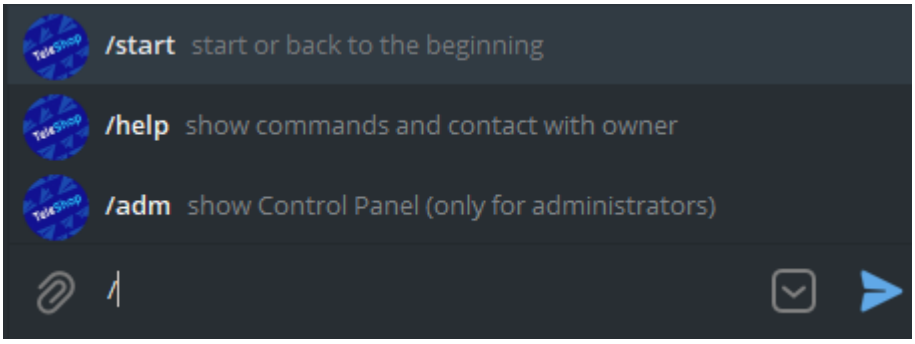


Рис. 3.6. Демонстрація підказок зі списком команд

3.2.2. Програмування чат-боту

Для написання коду на мові *Python*, який буде взаємодіяти з чат-ботом та не тільки, необхідні такі модулі:

- *SQLite3;*
- *shelve;*
- *telebot;*
- *SimpleQIWI;*
- *time;*
- *requests.*

Після створення бота у месенджері та отримання *API* токену, ми можемо описати детальний алгоритм роботи чат-бота на різні запити.

Спершу створюємо файл під назвою «*config.py*», який буде містити в собі унікальний номер користувача з правами адміністратора та *API Token*.

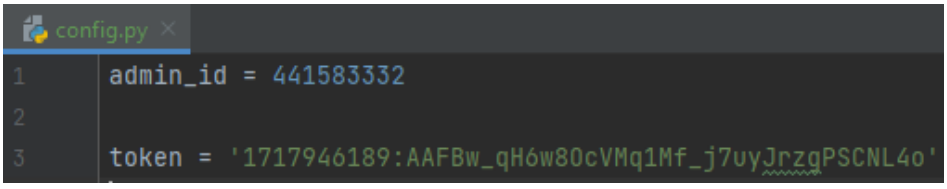


Рис. 3.7. Параметри файлу *config.py*

Завантажуємо необхідні бібліотеки та вказуємо ботові, який ключ він повинен використовувати для взаємодії з месенджером (рис. 3.8.).

```python
import telebot, sqlite3, shelve
import config, dop, files

bot = telebot.TeleBot(config.token)
```

Рис. 3.8. Імпорт бібліотек та директорій. Створюємо об’єкт класу *Telebot* та задаємо йому шлях до *API* токену

Після цього, ми вказуємо ботові, що він повинен робити при отриманні команди /*start*, використовуючи так звані функції «хендлери», які вже описані в бібліотеці *telebot* (рис. 3.9).

```python
@bot.message_handler(content_types=["text"])
def message_send(message):
    if '/start' == message.text:
        if message.chat.username:
            if dop.get_sost(message.chat.id) is True:
                with shelve.open(files.sost_bd) as bd: del bd[str(message.chat.id)]
            if message.chat.id in in_admin: in_admin.remove(message.chat.id)
            if message.chat.id == config.admin_id and dop.it_first(message.chat.id) is True:
                in_admin.append(message.chat.id)
            elif dop.it_first(message.chat.id) is True and message.chat.id not in dop.get_adminlist():
                bot.send_message(message.chat.id,
                                 'Бот ещё не готов к работе!\nЕсли вы являетесь его администратором, то войдите из под\
                                  аккаунту, id которого указали при запуске бота и подготовьте его к работе!')
            elif dop.check_message('start') is True:
                key = telebot.types.InlineKeyboardMarkup()
                key.add(telebot.types.InlineKeyboardButton(text='Перейти к каталогу товаров',
                                                           callback_data='Перейти к каталогу товаров'))
                with shelve.open(files.bot_message_bd) as bd:
                    start_message = bd['start']
                start_message = start_message.replace('username', message.chat.username)
                start_message = start_message.replace('name', message.from_user.first_name)
                bot.send_message(message.chat.id, start_message, reply_markup=key)
            elif dop.check_message('start') is False and message.chat.id in dop.get_adminlist():
                bot.send_message(message.chat.id,
                                 'Приветствие ещё не добавлено!\nЧтобы его добавить, перейдите в админку по команде \
                                 /adm и *настройте ответы бота*',
                                 parse_mode='Markdown')

            dop.user_loger(chat_id=message.chat.id)  # логгирование юзеров
        elif not message.chat.username:
            with shelve.open(files.bot_message_bd) as bd:
                start_message = bd['userfalse']
            start_message = start_message.replace('uname', message.from_user.first_name)
            bot.send_message(message.chat.id, start_message, parse_mode='Markdown')
```

Рис. 3.9. Обробка команди /*start*

Не відходячи далеко, описуємо команди /*adm* та /*help* (рис. 3.10).

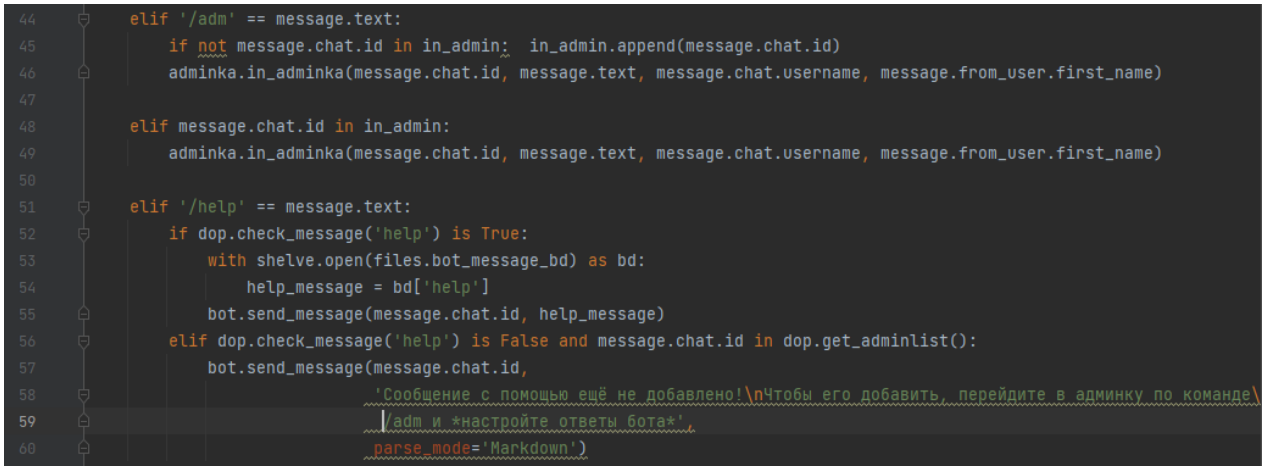


Рис. 3.10. Команди /*adm* та /*help*

Як можемо замітити, при використанні команди /*adm*, бот перевіряє користувача, чи він знаходиться у списку людей, які мають права адміністратора, після чого запускає функцію *in_adminka* у файлі *adminka*. У цьому файлі більш детально запрограмовано панель керування.

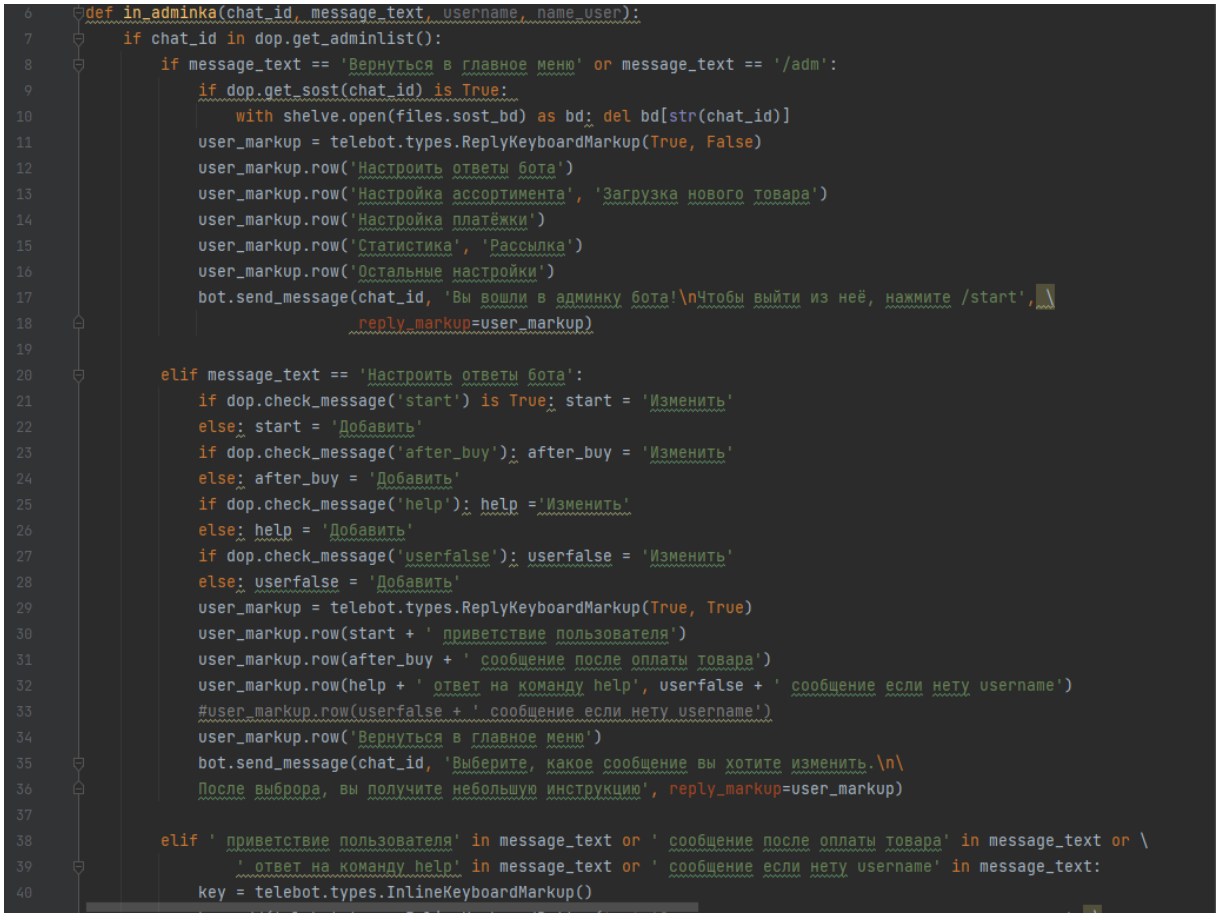


Рис. 3.11. Функція *in_adminka*

За допомогою *user_markup*, які ми присвоїли функцію для створення кнопок,

ми дублюємо її, створюючи нові кнопки.

Тепер поглянемо, що показує бот на рис. 3.12.

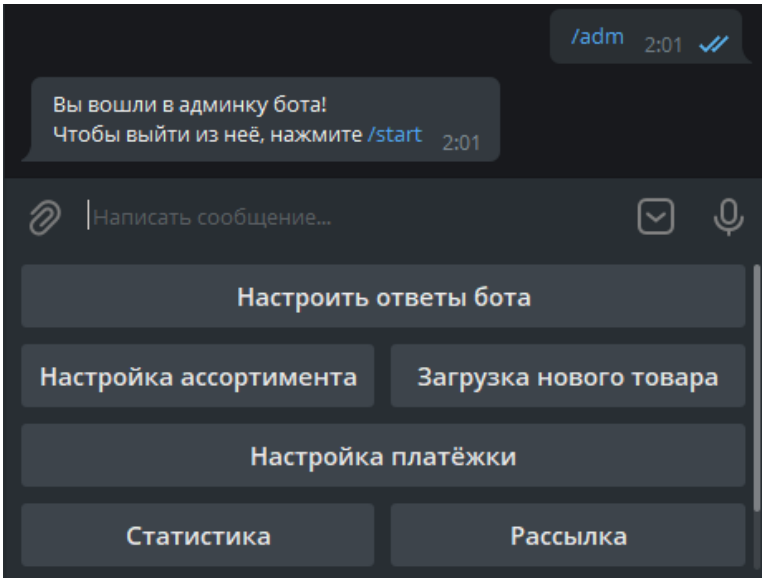


Рис. 3.12. Кнопки інтерактивного меню

Для налаштування відповідей бота користувачам ми заходимо в меню «Налаштувати відповіді бота» (рис. 3.13), після чого перед нами стоїть новий вибір:

1) Змінити повідомлення після команди /*start*.
2) Змінити повідомлення після успішної оплати.
3) Змінити відповідь на команду /*help*.
4) Змінити повідомлення, після команди /*start*, якщо *username* відсутнє.

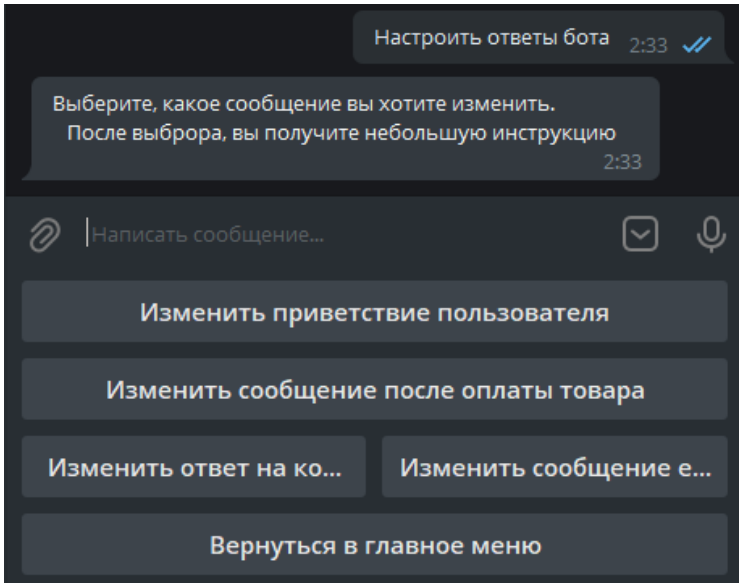


Рис. 3.13. Налаштування відповідей бота

При виборі параметру для заміни привітання користувачів, бот записує вас, щоб ви ввели нове повідомлення, при цьому можливість заміни слів *user* та *username* на імя та логін користувача.

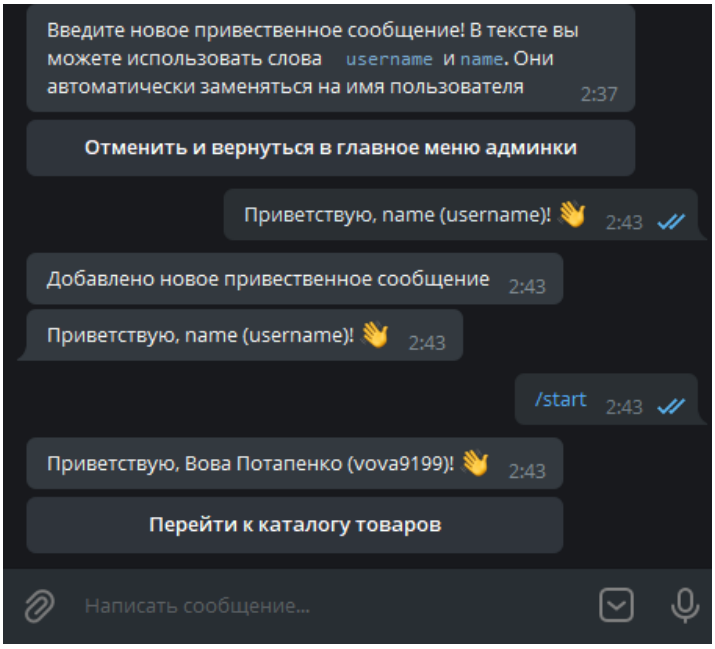


Рис. 3.14. Зміна повідомлення /*start*

Всі відредаговані відповіді будуть зберігатись у відповідних файлах, як на рис. 3.15.

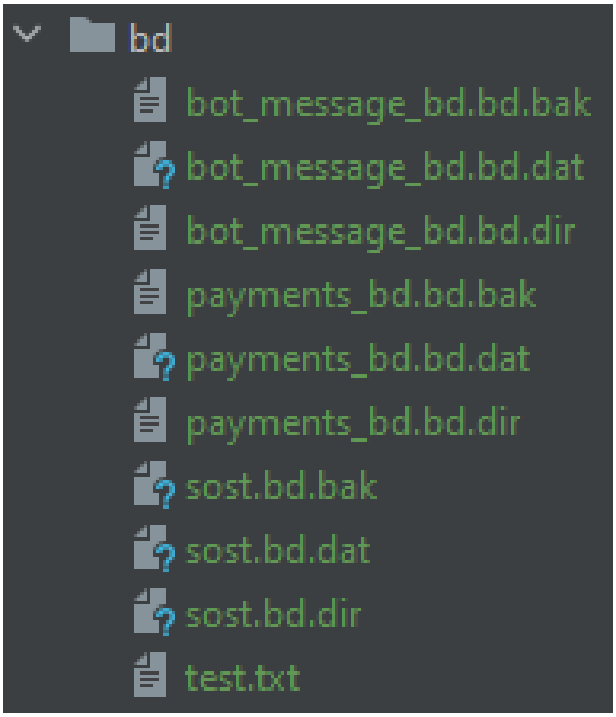


Рис. 3.15. Зберігання текстових відповідей у файлах

За допомогою бібліотеки *shelve*, яка працює з мовою програмування *Python*, ми можемо створювати об’єкти, задавати їм параметри та зберігати все це у файл. Подібний принцип до роботи з базами даними.

Завантаження нового продукту відбувається теж через панель керування, в пункті «Додати нову позицію на вітрину».

Для прикладу створимо продукт, який буде містити в собі базу даних автосалону, з форматом файлу *Access Database* (*.accdb*). Процес створення зображена на рис. 3.16.

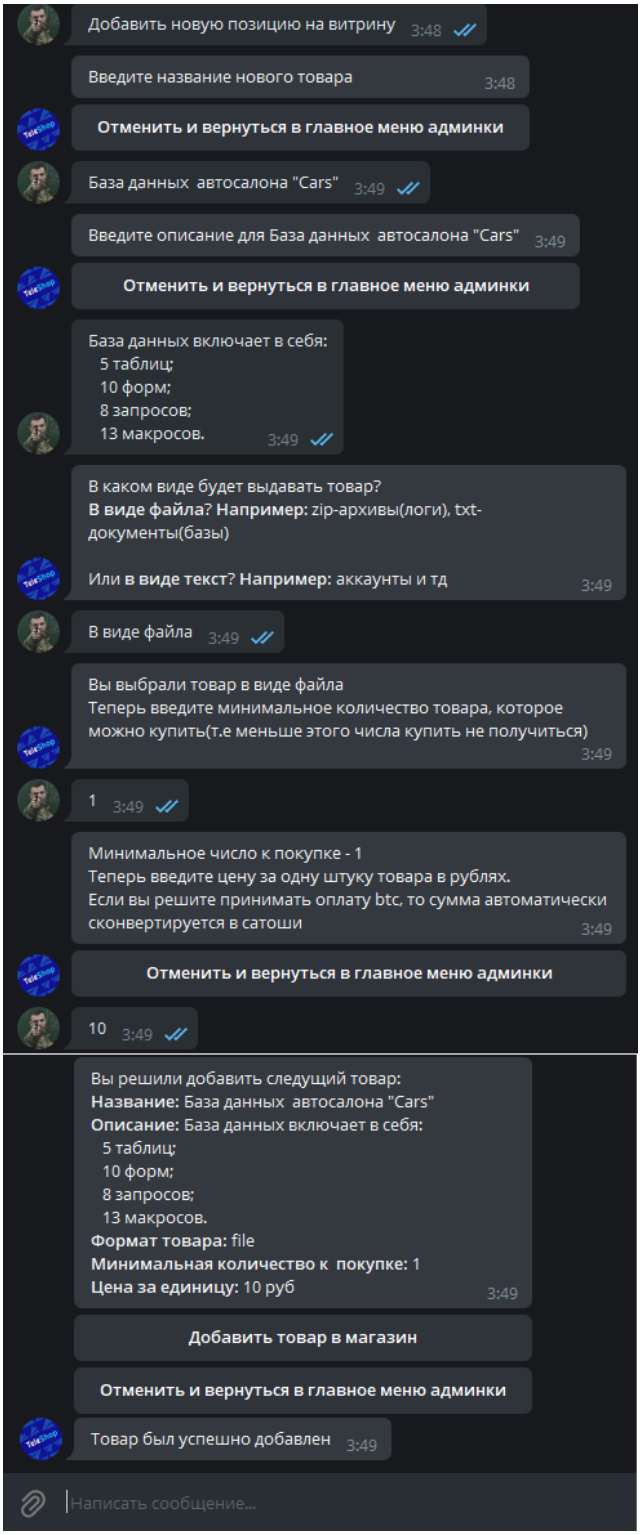


Рис. 3.16. Створення нового асортименту

Як бачимо товар був успішно добавлений на вітрину.

Тепер напишимо код для зміни ціни товару через панель панель адміністратора.

```
127     elif 'Поменять цену' == message_text:
128         con = sqlite3.connect(files.main_db)
129         cursor = con.cursor()
130         cursor.execute("SELECT name, price FROM goods;")
131         a = 0
132         user_markup = telebot.types.ReplyKeyboardMarkup(True, False)
133         for name, price in cursor.fetchall():
134             a += 1
135             user_markup.row(name)
136         if a == 0:
137             user_markup.row('Добавить новую позицию на витрину', 'Удалить позицию')
138             user_markup.row('Поменять описание позиции', 'Поменять цену')
139             user_markup.row('Вернуться в главное меню')
140             bot.send_message(chat_id, 'Никаких позиций ещё не создано!', reply_markup=user_markup)
141         else:
142             user_markup.row('Вернуться в главное меню')
143             bot.send_message(chat_id, 'У какой позиции вы хотите изменить цену?', parse_mode='Markdown', reply_markup=user_markup)
144             with shelve.open(files.sost_bd) as bd_: bd[str(chat_id)] = 9
145         con.close()
```

Рис. 3.17. Код функції для зміни ціни товару

Тепер продемонструємо зміну ціни у месенджері.

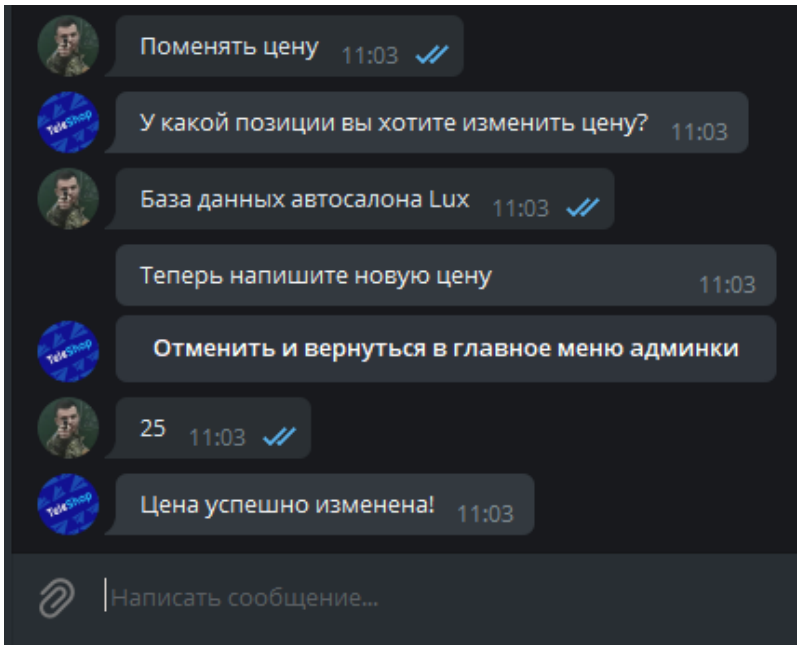


Рис. 3.18. Зміна ціни товару через панель керування

Після створення товару потрібно загрузити файл, який буде відправлятись покупцеві, після успішною оплати товару.

```
147         elif 'Загрузка нового товара' == message_text:
148             key = telebot.types.InlineKeyboardMarkup()
149             key.add(telebot.types.InlineKeyboardButton(text='Отменить и вернуться в главное меню админки',
150                                                        callback_data='Вернуться в главное меню админки'))
151             con = sqlite3.connect(files.main_db)
152             cursor = con.cursor()
153             cursor.execute("SELECT name, price FROM goods;")
154             a = 0
155             user_markup = telebot.types.ReplyKeyboardMarkup(True, True)
156             for name, price in cursor.fetchall():
157                 a += 1
158                 user_markup.row(name)
159             if a == 0: bot.send_message(chat_id, 'Никаких позиций ещё не создано!')
160             else:
161                 user_markup.row('Вернуться в главное меню')
162                 bot.send_message(chat_id, 'Товары какой позиции вы хотите загрузить?',
163                                  parse_mode='Markdown', reply_markup=user_markup)
164                 with shelve.open(files.sost_bd) as bd: bd[str(chat_id)] = 11
165             con.close()
```

Рис. 3.19. Функція обробки завантаження файлу

Функції завантаження товару дає можливість обрати на вибір який тип продукції отримає клієнт:

- файлом: *zip*-архів, *rar*-архів, *.txt*, *.db*, *.html*, *.sql*, та інші;
- текстом: код, текст, символи, емодзі та інші.

Завантаження файлу товару у базу даних продемонстровано на рис. 3.20.

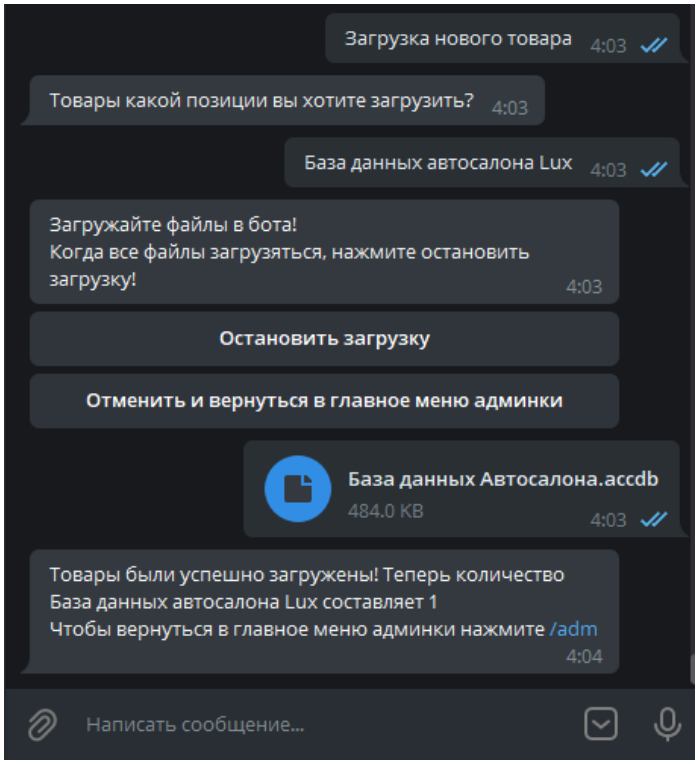
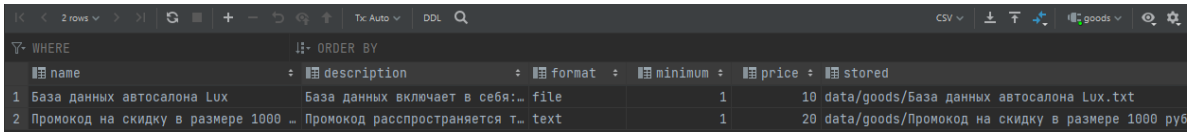


Рис. 3.20. Завантаження файлу

Як отриманий файл зберігається у базі даних ми можемо переглянути на рис. 3.21.



	name	description	format	minimum	price	stored
1	База данных автосалона Lux	База данных включает в себя:…	file	1	10	data/goods/База данных автосалона Lux.txt
2	Промокод на скидку в размере 1000 …	Промокод расспространяется т…	text	1	20	data/goods/Промокод на скидку в размере 1000 ру6

Рис. 3.21. Файл в базі даних *SQLite3*

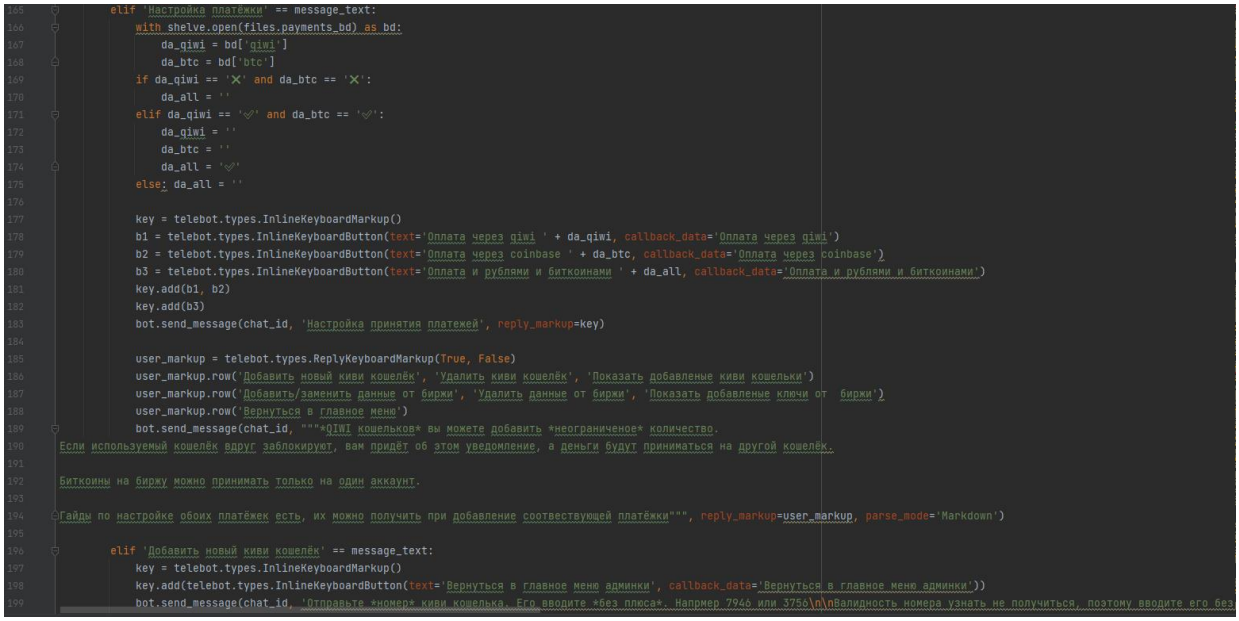
Поле name відповідає за назву товару, *description* – за опис товару, *format* – формат файлу, в якому буде подаватись куплений товар клієнтові, мінімальна кількість при замовленні, ціна файл з описом розташування.

Наступним кроком потрібно описати показ каталогу товарів клієнтові, який тільки почав роботу з чат-ботом. Для цього нам потрібно відкрити базу даних з уже існуючими товарами та за допомого запитів на мові *SQL* отримати потрібні дані. За допомого модуля *SQLite3* все це робиться просто, як зображено на рис. 3.22.

```
@bot.callback_query_handler(func=lambda c: True)
def inline(callback):
    the_goods = dop.get_goods()
    if callback.message.chat.id in in_admin:
        adminka.ad_inline(callback.data, callback.message.chat.id, callback.message.message_id)

    elif callback.data == 'Перейти к каталогу товаров':
        con = sqlite3.connect(files.main_db)
        cursor = con.cursor()
        cursor.execute("SELECT name, price FROM goods;")
        key = telebot.types.InlineKeyboardMarkup()
        for name, price in cursor.fetchall():
            key.add(telebot.types.InlineKeyboardButton(text=name, callback_data=name))
        key.add(telebot.types.InlineKeyboardButton(text='Вернуться в начало', callback_data='Вернуться в начало'))
        con.close()

        if dop.get_productcatalog() == None:
            bot.answer_callback_query(callback_query_id=callback.id, show_alert=True,
                                      text='На данный момент в боте не было создано ни одной позиции')
        else:
            try:
                bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id,
                                      text=dop.get_productcatalog(), reply_markup=key, parse_mode='Markdown')
            except:
                pass

    elif callback.data in the_goods:
        with open('data/Temp/' + str(callback.message.chat.id) + 'good_name.txt', 'w', encoding='utf-8') as f:
            f.write(callback.data)
        key = telebot.types.InlineKeyboardMarkup()
        key.add(telebot.types.InlineKeyboardButton(text='Купить', callback_data='Купить'))
        key.add(telebot.types.InlineKeyboardButton(text='Вернуться в начало', callback_data='Вернуться в начало'))
        try:
            bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id,
                                  text=dop.get_description(callback.data), reply_markup=key)
        except:
            pass
```

Рис. 3.22. Отримання каталогу товарів з бази даних

Наступним головним кроком буде додавання платіжної системи. В даному випадку було реалізовано оплату через платіжну систему *QIWI*, яка використовує грошову одиницю в рублях. Це не мішає нам здійснювати оплату з любої карти банки, головне щоб була *Visa* або *MasterCard*. Головним параметром є *API Token* електронного гаманця. Через цей токен будуть здійснюватися перевірки успішність платежу.



```
        elif 'Настройка платёжки' == message_text:
            with shelve.open(files.payments_bd) as bd:
                da_qiwi = bd['qiwi']
                da_btc = bd['btc']
            if da_qiwi == '✕' and da_btc == '✕':
                da_all = ''
            elif da_qiwi == '✓' and da_btc == '✓':
                da_qiwi = ''
                da_btc = ''
                da_all = '✓'
            else: da_all = ''

            key = telebot.types.InlineKeyboardMarkup()
            b1 = telebot.types.InlineKeyboardButton(text='Оплата через qiwi ' + da_qiwi, callback_data='Оплата через qiwi')
            b2 = telebot.types.InlineKeyboardButton(text='Оплата через coinbase ' + da_btc, callback_data='Оплата через coinbase')
            b3 = telebot.types.InlineKeyboardButton(text='Оплата и рублями и биткоинами ' + da_all, callback_data='Оплата и рублями и биткоинами')
            key.add(b1, b2)
            key.add(b3)
            bot.send_message(chat_id, 'Настройка принятия платежей', reply_markup=key)

            user_markup = telebot.types.ReplyKeyboardMarkup(True, False)
            user_markup.row('Добавить новый киви кошелёк', 'Удалить киви кошелёк', 'Показать добавленые киви кошельки')
            user_markup.row('Добавить/заменить данные от биржи', 'Удалить данные от биржи', 'Показать добавленые ключи от  биржи')
            user_markup.row('Вернуться в главное меню')
            bot.send_message(chat_id, """*QIWI кошельков* вы можете добавить *неограниченое* количество.
Если используемый кошелёк вдруг заблокируют, вам придёт об этом уведомление, а деньги будут приниматься на другой кошелёк.

Биткоины на биржу можно принимать только на один аккаунт.

Гайды по настройке обоих платёжек есть, их можно получить при добавление соотвествующей платёжки""", reply_markup=user_markup, parse_mode='Markdown')

        elif 'Добавить новый киви кошелёк' == message_text:
            key = telebot.types.InlineKeyboardMarkup()
            key.add(telebot.types.InlineKeyboardButton(text='Вернуться в главное меню админки', callback_data='Вернуться в главное меню админки'))
            bot.send_message(chat_id, 'Отправьте *номер* киви кошелька. Его вводите *без плюса*. Напрмер 7946 или 375б\n\nВалидность номера узнать не получиться, поэтому вводите его без
```

Рис. 3.23. Додавання платіжної системи на рівні програмування

У месенджері зі сторони адміністратора це буде виглядати так, як показано на рис. 3.23.

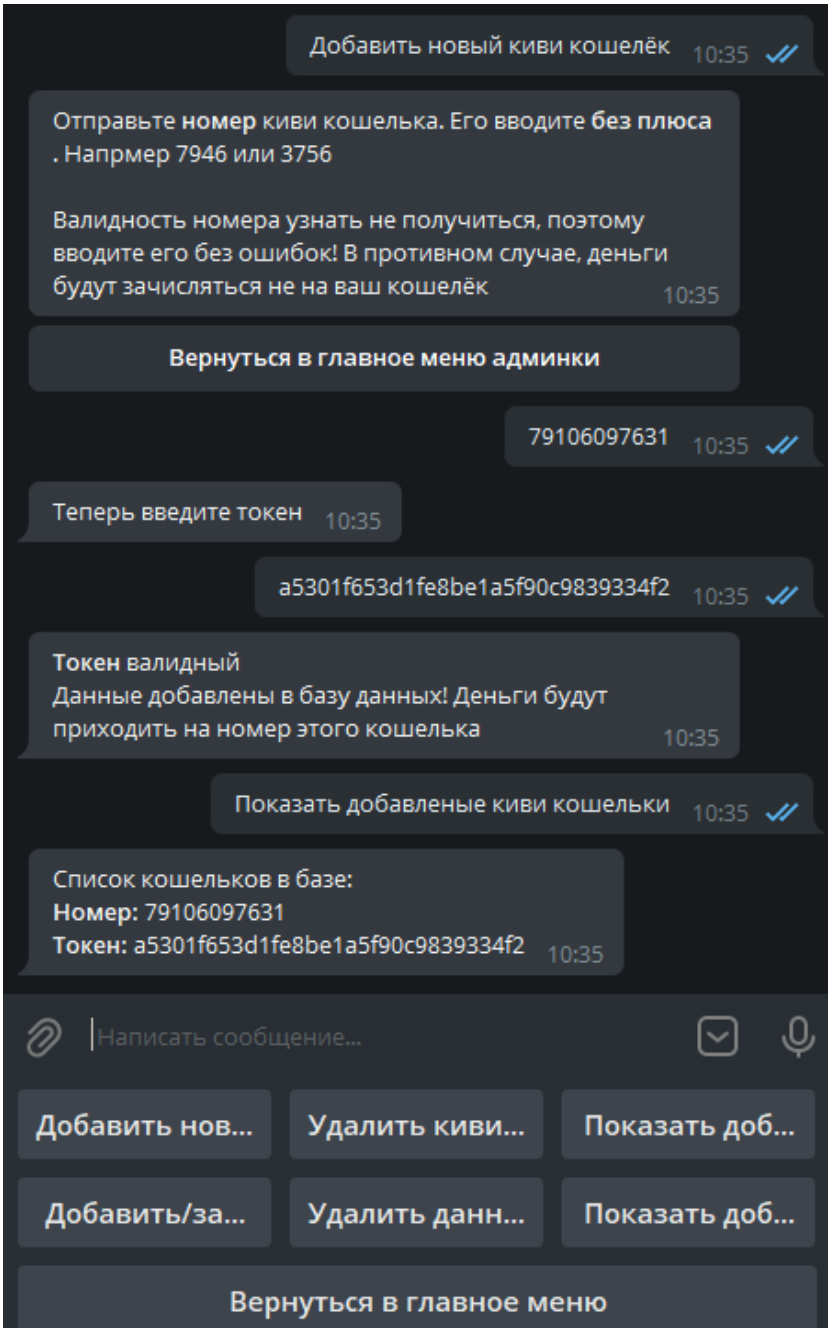


Рис. 3.23. Додавання нового *QIWI* гаманцю для прийняття оплати

Реалізація надання прав адміністратора іншому користувачеві на програмному рівні описано як на рис. 3.24.

```
273     elif 'Остальные настройки' == message_text:
274         user_markup = telebot.types.ReplyKeyboardMarkup(True, False)
275         user_markup.row('Добавить нового админа', 'Удалить админа')
276         user_markup.row('Вернуться в главное меню')
277         bot.send_message(chat_id, 'Выберите, что желаете сделать', reply_markup=user_markup)
278
279     elif 'Добавить нового админа' == message_text:
280         key = telebot.types.InlineKeyboardMarkup()
281         key.add(telebot.types.InlineKeyboardButton(text = 'Отменить и вернуться в главное меню админки',
282                                                    callback_data = 'Вернуться в главное меню админки'))
283         bot.send_message(chat_id, 'Введите id нового админа')
284         with shelve.open(files.sost_bd) as bd: bd[str(chat_id)] = 21
285
286     elif 'Удалить админа' == message_text:
287         user_markup = telebot.types.ReplyKeyboardMarkup(True, False)
288         a = 0
289         for admin_id in dop.get_adminlist():
290             a += 1
291             if int(admin_id) != config.admin_id: user_markup.row(str(admin_id))
292         if a == 1: bot.send_message(chat_id, 'Вы ещё не добавляли админов!')
293         else:
294             user_markup.row('Вернуться в главное меню')
295             bot.send_message(chat_id, 'Выбери id админа, которого нужно удалить', reply_markup=user_markup)
296             with shelve.open(files.sost_bd) as bd: bd[str(chat_id)] = 22
```

Рис. 3.24. Код обробки функції додаваня адміністратора

Через інтерфейс месенджеру додавання людини з правами адміністратора виглядає так, як на рис. 3.25.

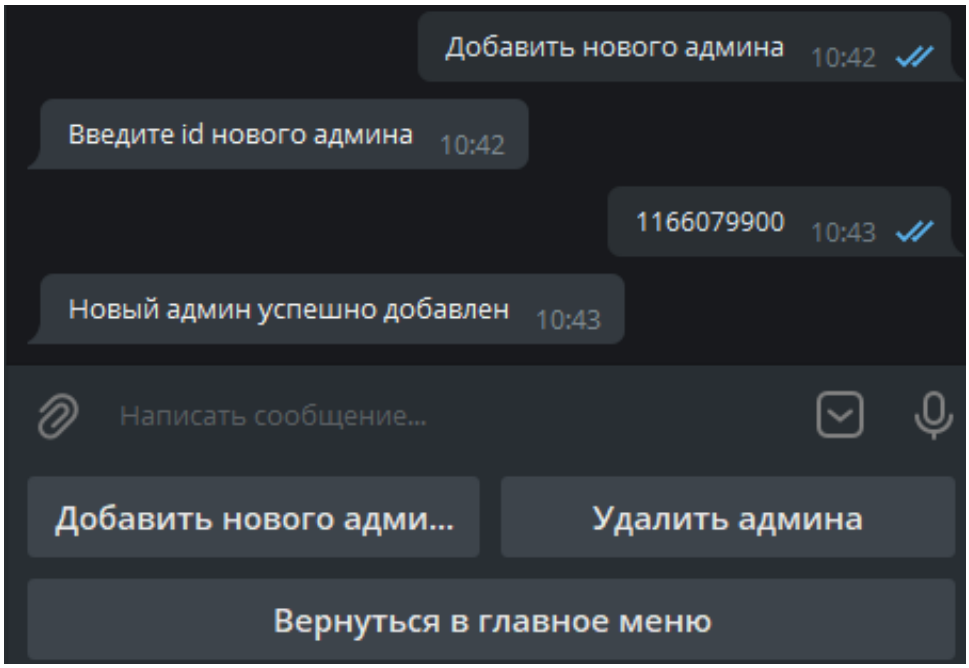


Рис. 3.25. Надання прав адміністратора

3.3. Результат роботи *Telegram*-додатку зі сторони клієнта

Всі користувачі, які бажають придбати якийсь товар через нашого *Telegram*-додатку, повинні найти бота за іменем *@TeleShopPotapBot*, після чого запустити команду /*start* (рис. 3.26).

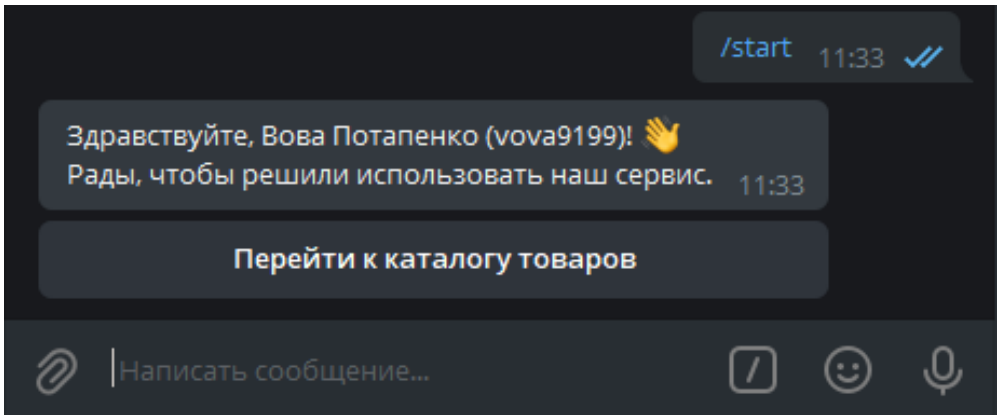


Рис. 3.26. Початок роботи з ботом

Після вибору кнопки для переходу до каталогу ми бачимо список доступних продуктів (рис. 3.27).

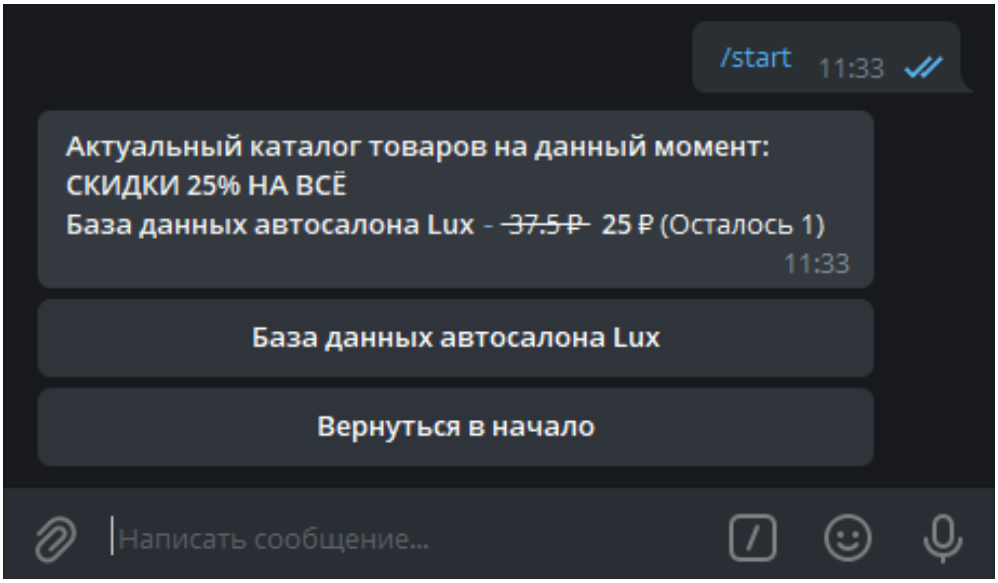


Рис. 3.27. Каталог товарів

Обираємо товар під назвою «База даних автосалону *Lux*», після чого повідомлення змінюється на опис продукту.

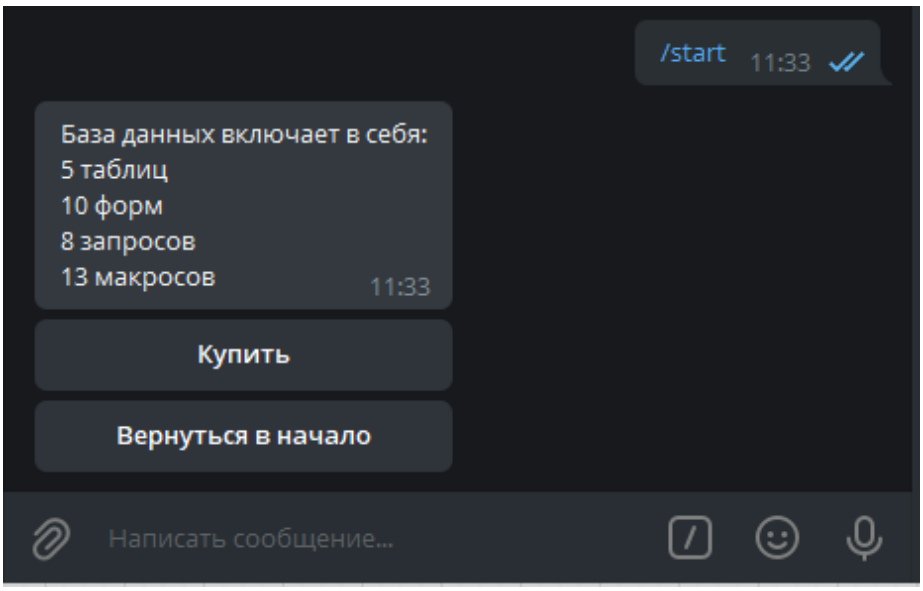


Рис. 3.28. Опис продукту

Наступним кроком буде вибір бажаної кількості товарів (мінімальна к-сть 1), після чого нам пропонують вибір платіжної системи, а саме *QIWI* (рис. 3.29).

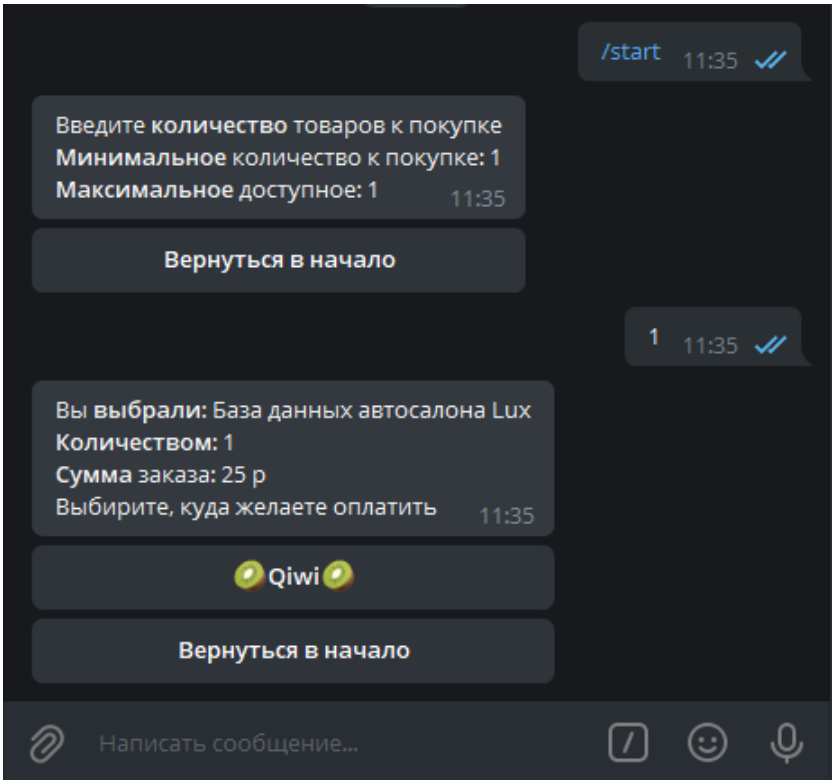


Рис. 3.29. Вибір кількості товарів та платіжної системи

Після вибору платіжної системи бот пропонує перейти на веб-посилання на якому вже заповнена форма для оплати товару (рис. 3.31).

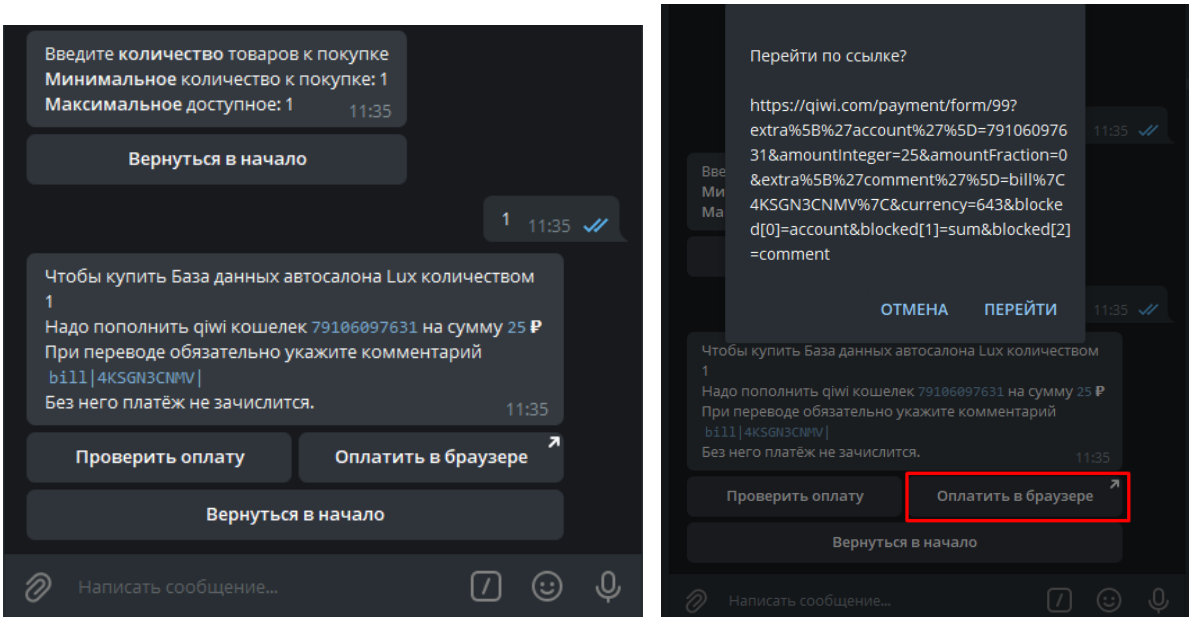


Рис. 3.30. Повідомлення з параметрами для оплати та перехід на веб-посилання

Номер на який будуть нараховані кошти – «79106097631», на суму 25 рублів, в еквіваленті на гривні - 9.82 *UAH*. Головним параметром для отплати є коментар «*bill|4KSGN3CNMV*», за допомогою якого через систему *API* буде здійснюватись перевірка платежу та її успішність.

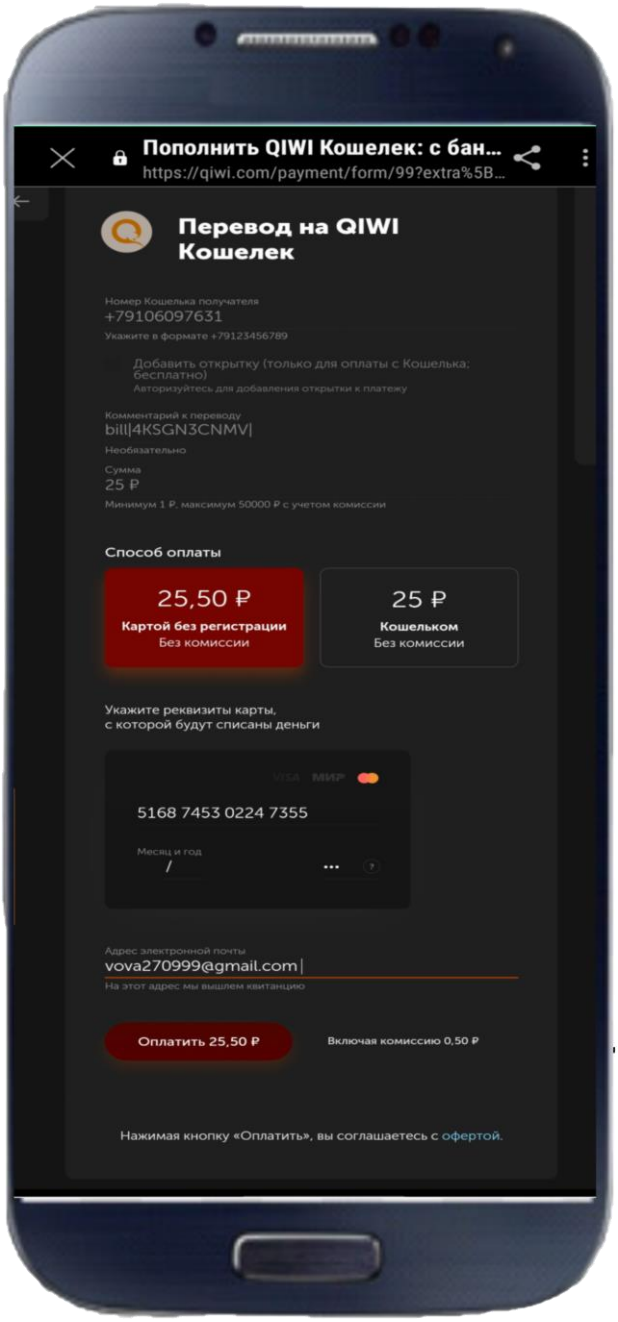


Рис. 3.31. Перехід на веб-посилання з готовою формою

Після заповнення даних про банківську карту та пошти, на яку буде відправлятись квитанція про оплату, натискаємо кнопку «Сплатити 25.50 *RUB*» (Рис. 3.31).

Далі система посилає нас на етап варифікації платежу вашої банківської системи, у моєму випадку це ПриватБанк (рис. 3.32).

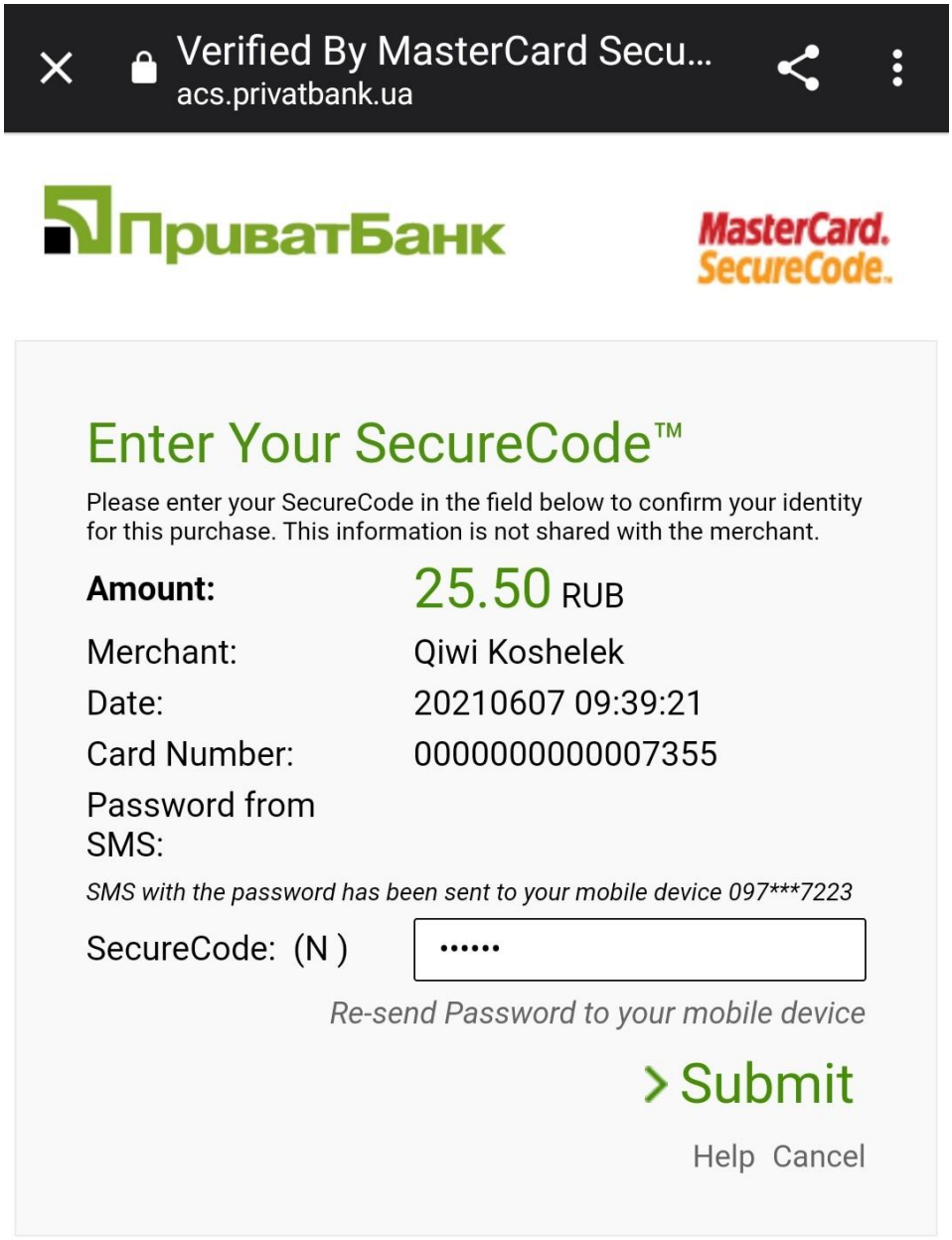


Рис. 3.32. Варифікація платежу

Після підтвердження особи паролем, *QIWI* обробляє платіж (рис. 3.33) та підтверджує його успішність (3.34).

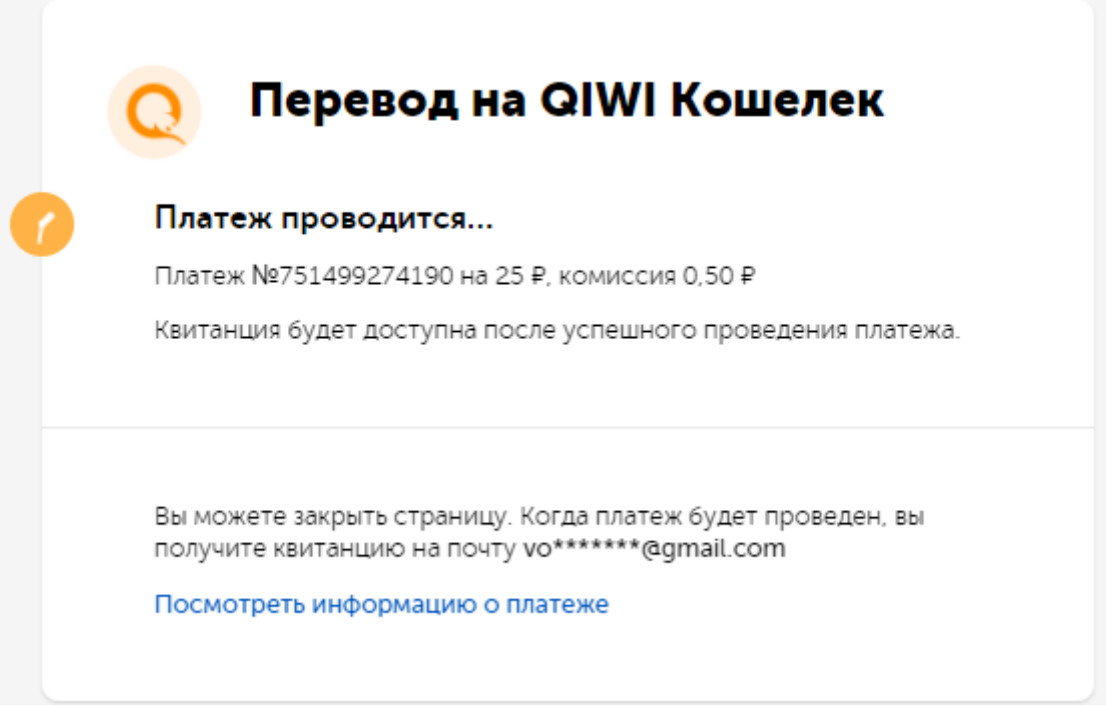


Рис. 3.33. Обробка платежу

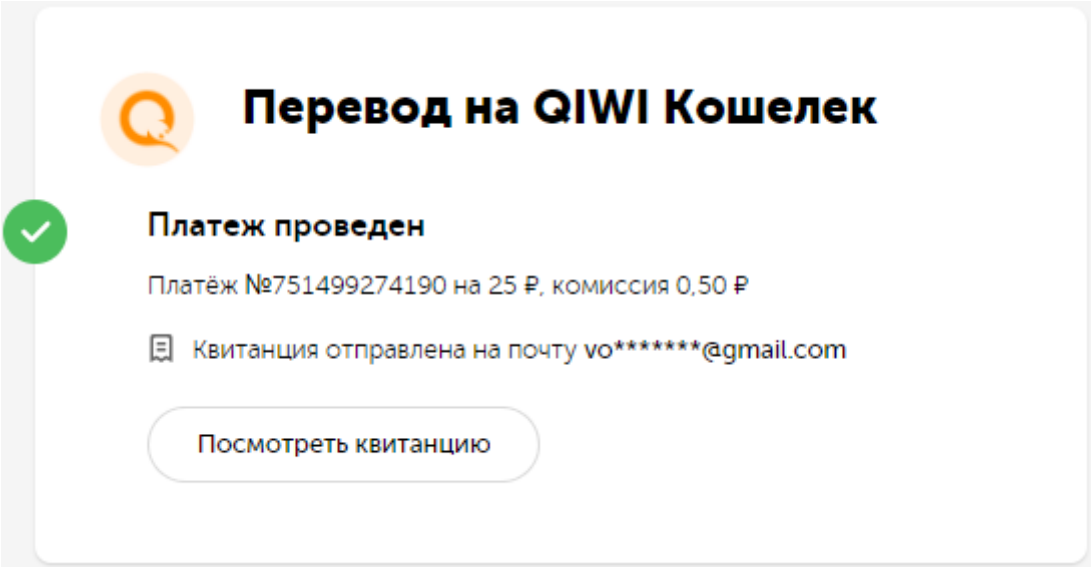


Рис. 3.34. Підтвердження платежу

За підтримки *API* токену бот робить запит на серверну платформу платіжної системи *QIWI*, для отримання історії платежів та перевіряє чи коментар співпадає з виданим певному користувачу (рис. 3.35).

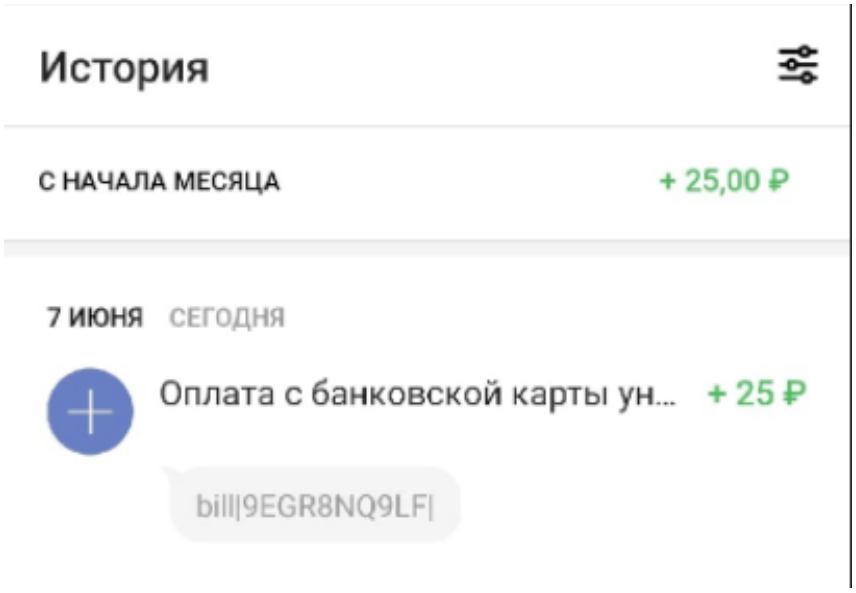


Рис. 3.35. Історія платежів

Останнім кроком являється натискання кнопки для перевірки платежу та отримання певного продукту, за умов успішної перевірки кому саме було відправлено унікальний коментар для оплати (рис. 3.36).

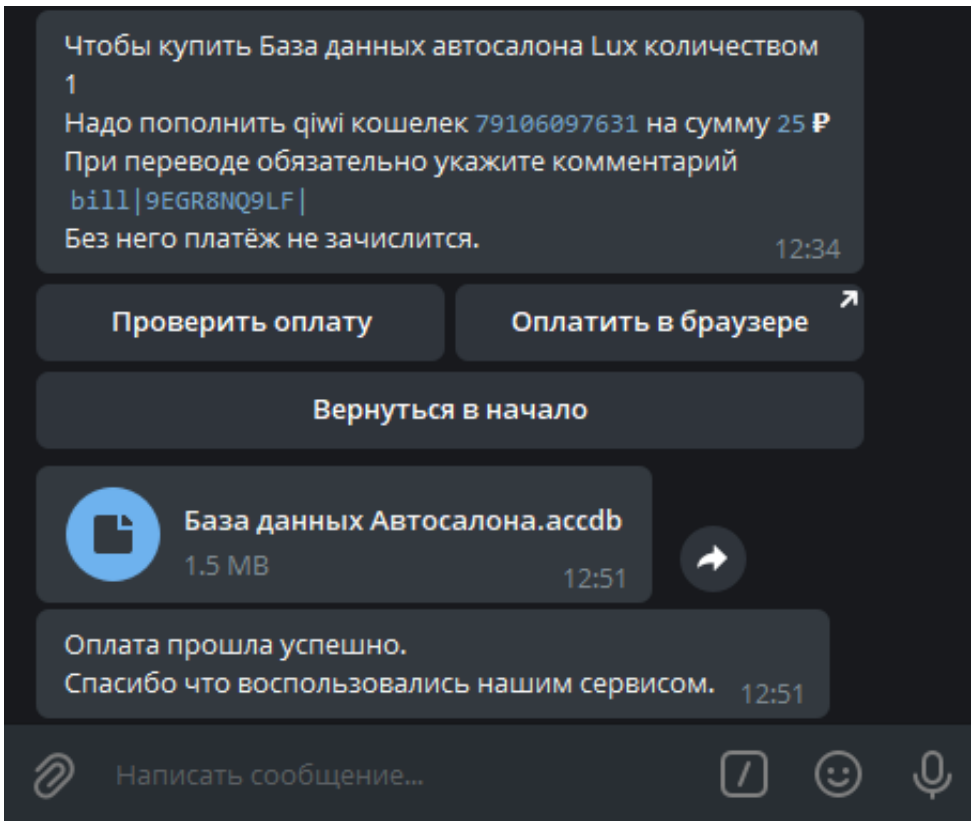


Рис. 3.36. Отримання купленого товару

В той час усім адміністраторам приходить сповіщення про успішне проведення транзакції: який товар був придбаний, ким та на яку суму (рис. 3.37).

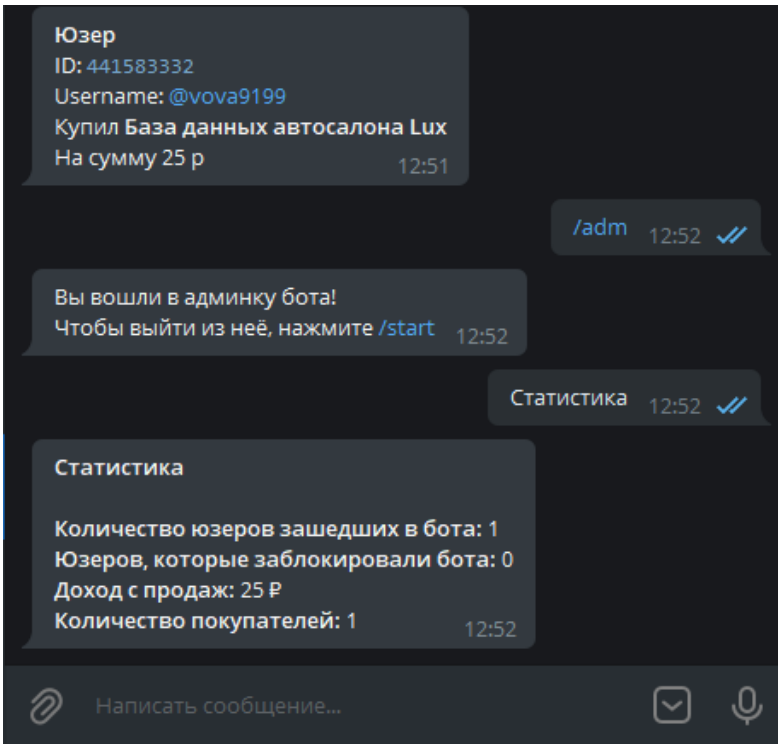


Рис. 3.37. Отримання повідомлення про успішність транзакції та вивід статистики

3.4. Розміщення застосунку у хмарному сервісі

У другому розділі було проаналізовано сервіси, які надають можливість розміщення додатку для хмарних обчислень, цим самим наш застосунок може працювати в автономному режимі. Це дозволить не займати ресурси розробника, а запустити програму на віддаленому сервері.

Для використання скрипту бота на виділеному сервері я застосував хмарну платформу *Google Cloud VDS*.

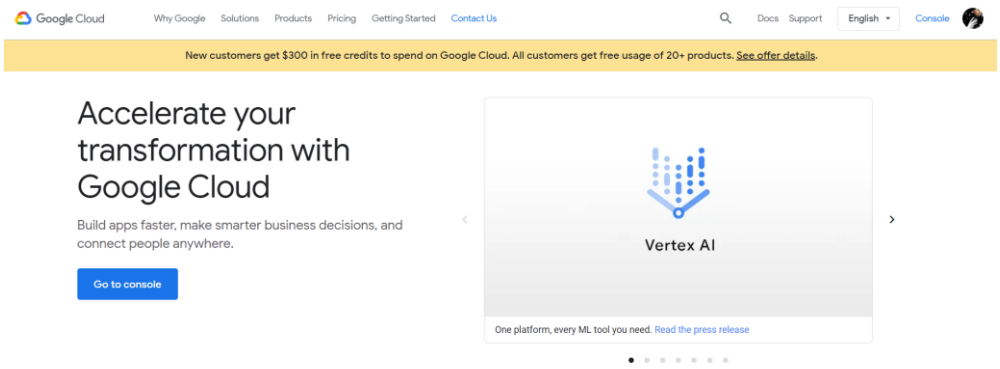


Рис. 3.38. *Google Cloud VDS*

Спершу потрібно створити на сайті віртуальну машину на базі операційної системи *Ubuntu Linux* (рис. 3.39).

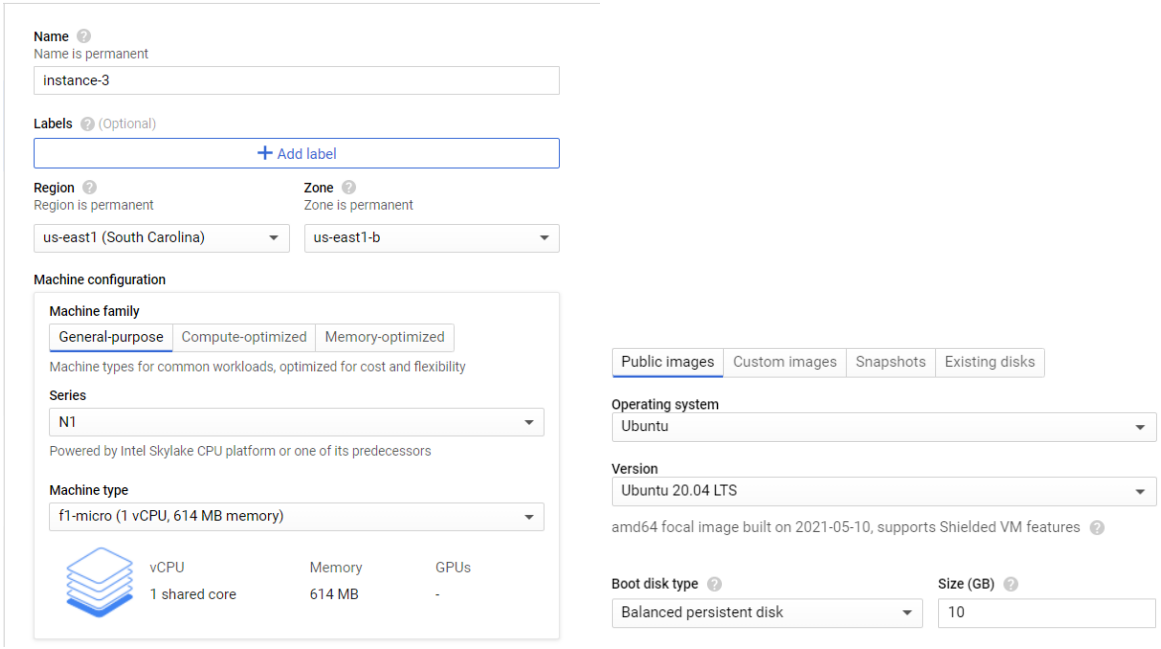


Рис. 3.39. Створення *VM instance Ubuntu Linux*

Для того щоб підключитись до віддаленого сервера необхідно сгнерувати *ssh* ключ використовую для цього програму *PuTTY Key Generator*.

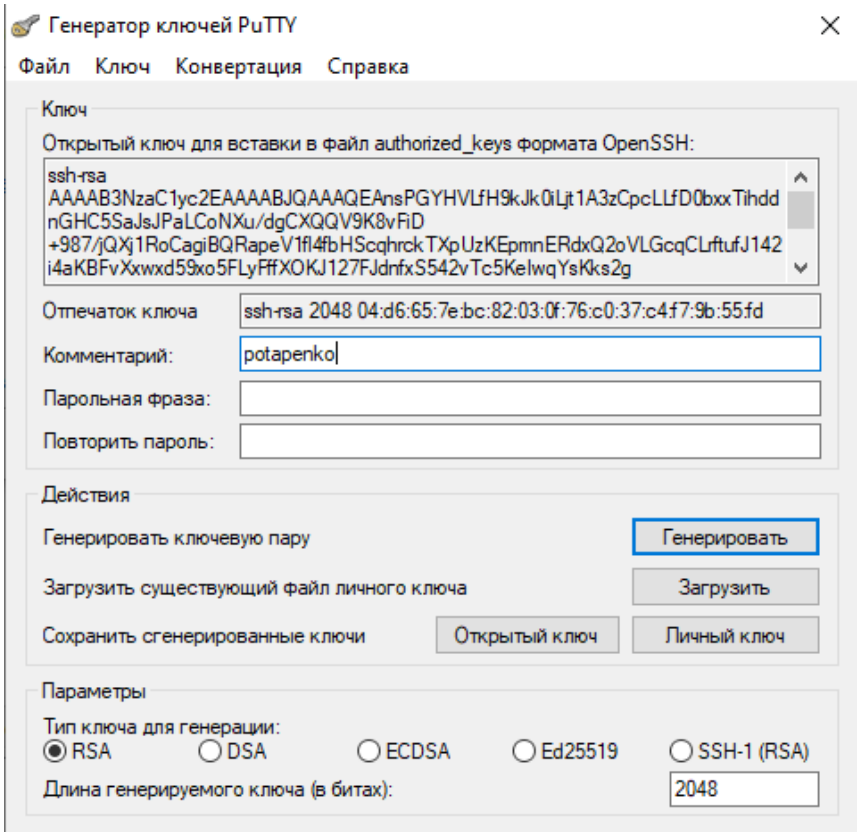


Рис. 3.40. Створення *SSH* ключа

Зберігаю *private key* на свій персональний комп'ютер та відкриваю програму *Pagent*, після чого додаю *private key*.

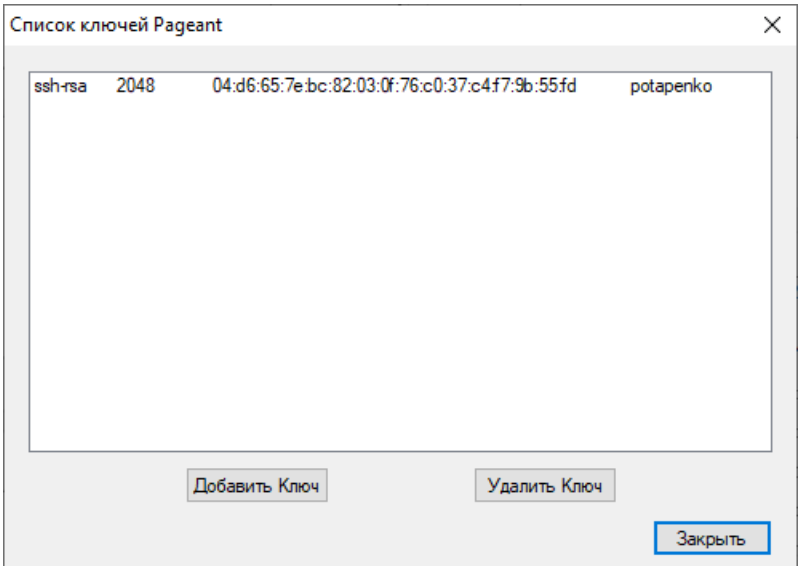


Рис. 3.41. Додавання *private key*

Копіюю *public key* з рис. 3.41 та додаю його до свого віддаленого серверу.

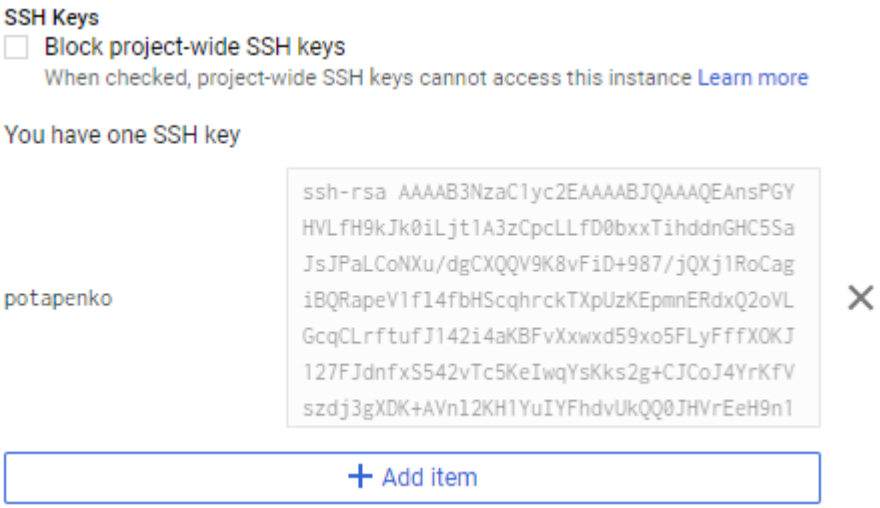


Рис. 3.42. Додавання *SSH key* до виділеного сервера

Підключаюсь до віддаленого сервера для передачі своїх файлів за допомогою програми *FileZilla*.

В полі Хост вписую тип підключення *sftp*://34.75.155.73.

В полі Ім’я користувача вписую свого користувача *potapenko*.

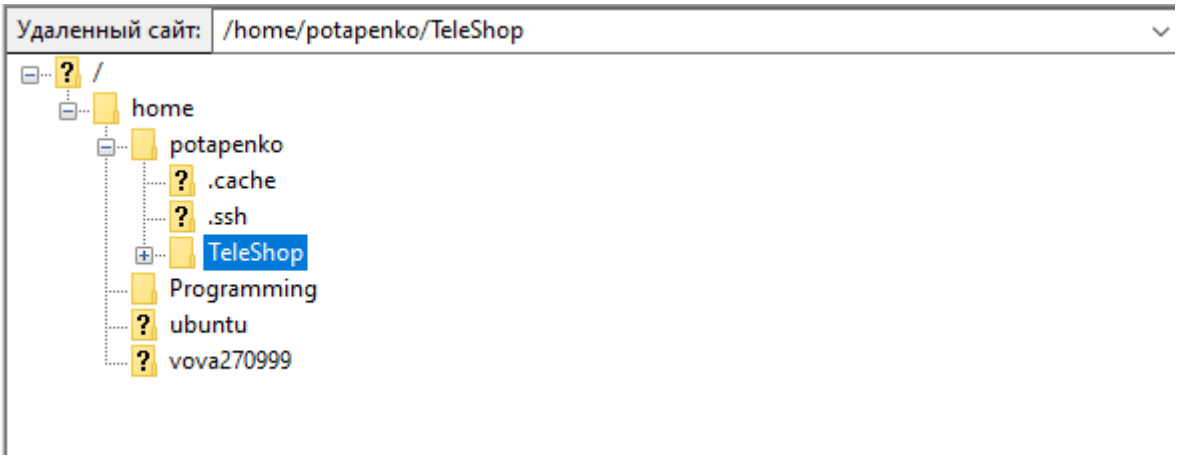


Рис. 3.43. Підключення до віддаленого серверу та передача файлів

Після завантаження пакетних модулів, необхідних для коректної роботи програми, запускаємо головний файл для роботи з ботом (*main.py*) на своєму сервері.

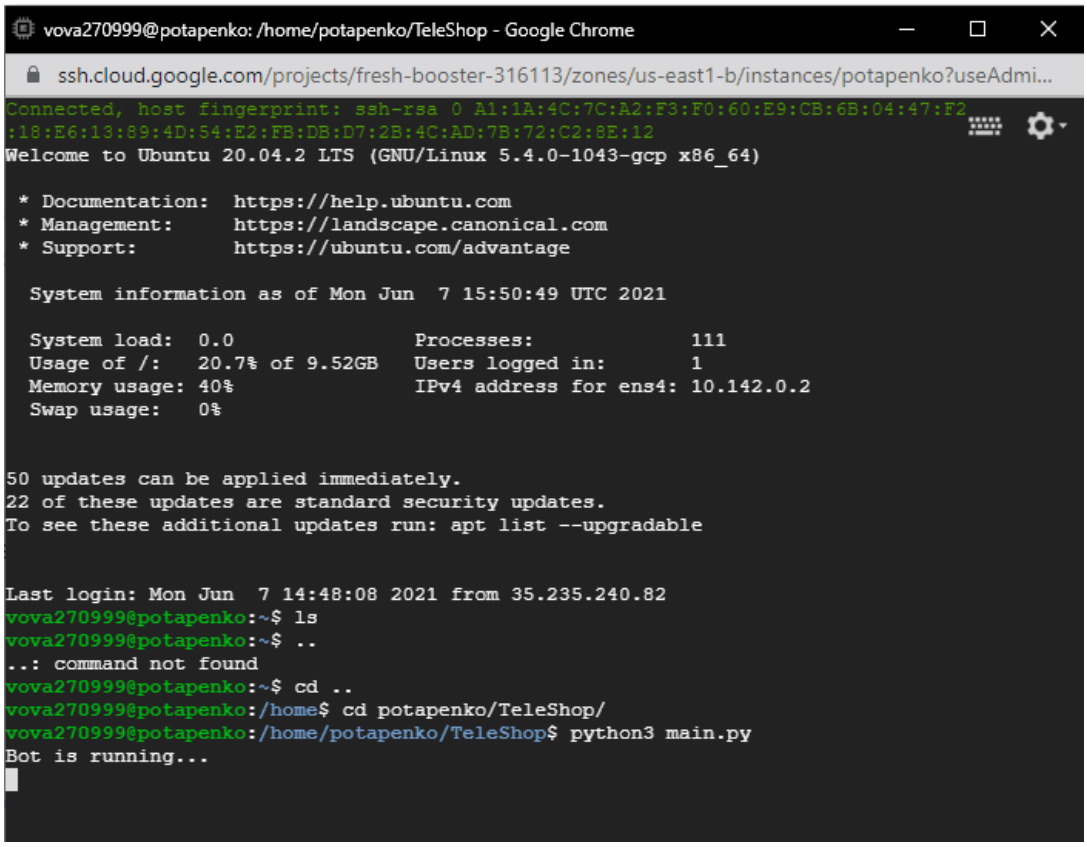


Рис. 3.44. Запуск файлу для початку роботи чат-бота

Перевіряємо роботу бота командою /*start* (рис. 3.45).

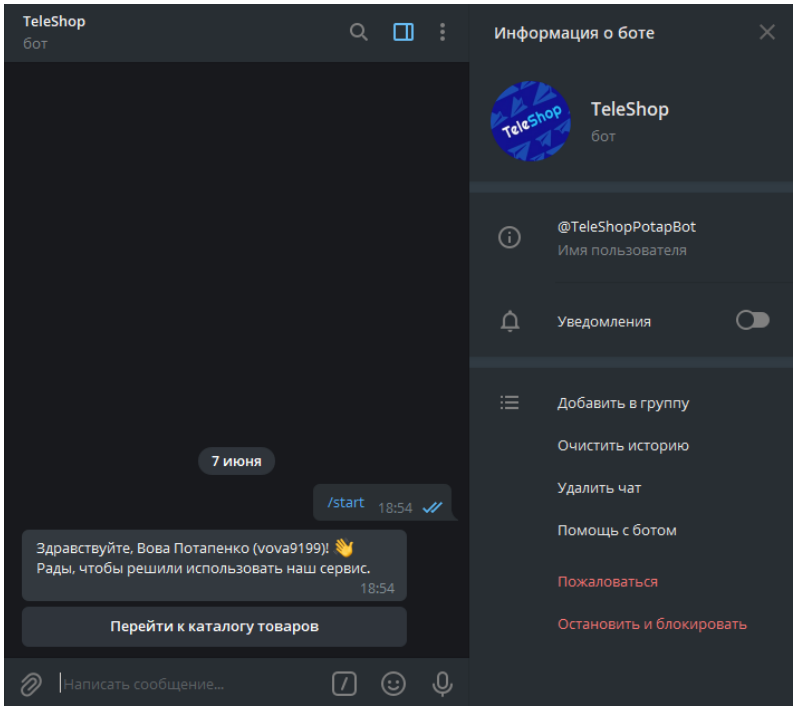


Рис. 3.45. Перевірка роботи бот, який запущений на хмарному сервісі

Висновки до розділу

У даному розділі було продемонстровано реалізацію *Telegram*-додатку, а саме – чат-боту. Було розроблено та запрограмовано на мові *Python*, з використанням різних бібліотек, графічний інтерфейс користувача (*GUI*), для взаємодії з програмним забезпеченням, що дозволить повноцінно керувати налаштуваннями додатку без знань мов програмування та інших технологій. Завдяки *GUI* ми заповнили базу даних товарам, додали їх опис та ціну. Також здійснили можливість оплати через банківську карту. Всі головні налаштування керуються через панель керування адміністратора, що, як було підкреслено вище, спрощує в декілька раз керування додатком.

Було продемонстровано з клієнтської сторони процес купівлі продукту, головною частиною якого являється оплата через банківську систему та отримання бажаного результату. Всі успішні транзакції клієнтів сповіщають адміністраторів про їх виконання, надаючи інформацію особистого характеру: хто купив, який товар, за яку суму. Також про кількість покупок та прибуток додатку показується в панелі керування в пункті «Статистика».

Головною частиною роботи, яка забезпечує роботу Telegram-додатку незалежно від розробника, це розміщення програми на сервісі для хмарних обчислень. Даному випадку було використано та розміщено на платформі *Google Cloud Platform*. Також було продемонстровано запуск скрипту та перевірка роботи чат-боту командою /*start*.

ВИСНОВКИ

Великі можливості *Telegram* ботів дають всі переваги для розробників, так як не вимагають додаткового пошуку аудиторії і є зрозумілими для обох сторін.

У 2021 на просторах інтернету активно розвиваються чат-боти та програми, які допомагають людям визначатися з вибором. Для підтримки конкурентоспроможності необхідно створити таку програму, яка буде містити в собі кілька підпрограм, тобто з легкістю здійснить пошук в базі даних, ґрунтуючись на перевагах клієнта та дасть інформацію про бажаний вибір товару.

Таким чином, бачимо, що створення *Telegram*-бота є актуальним завданням в сфері продажу, оскільки дозволяє спростити процеси представлення та продажу товарів, відповідно, скорочуючи витрату часу на непотрібні дії.

Об'єкт дослідження: компанія, яка бажає автоматизувати процес продажу за допомогою чат-ботів.

Предмет дослідження: опис бізнес-процес за допомогою мов програмування для розробки чат-боту.

Мета дослідження: створення загальнодоступного *Telegram*-бота з подальшим пошуком новачків у сфері продажу, щоб допомогти зберегти кошти, які вони можуть тратити на зарплатню стаціонарному продавцю.

Завдання дослідження: огляд аналогічних програм та їх функціоналу; здобути навички у програмуванні складних проектів за допомогою вже існуючих модулів; створення власного чат-бота.

Практична цінність: результати проекту можуть бути використані як і у сферах малого бізнесу для початківців, так і в складних бізнес-системах.

Сукупність запропонованих методів можна розглядати як спробу формування теоретичних основ нового наукового напряму досліджень, що тільки формується, необхідність розвитку якого диктується практикою і неминучими суперечностями, завжди супутніми науково-технічному прогресу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Оношко, В. Бизнес в *Telegram* из любой точки мира/ В. Оношко. - М.: Издательское решение, 2018.-100 с.

2. Андриянова С.С. Использование мессенджера *Telegram* для продвижения бренда/ С.С.Андриянова, А.А.Веретено// *Economics*. – 2018. – с. 54-56.

3. Козлов А.А. Телеграм – бот как простой и удобный способ получения информации/ А.А.Козлов, А.В.Батищев// Територія науки. – 2018. – с.55-64.

4. Матвеева, Н.Ю. Технологии создания и применения чат-ботов/ Н.Ю. Матвеева, А.В. Золотарюк// Наукові записки молодих дослідників. - 2018. - №1. - С.28-30.

5. Стефанова Н.А. Мессенджеры как цифровой бизнес – инструмент/ Н.А.Стефанова, К.О.Шматок// Карельський науковий журнал. – 2018. – Т.7. №2(23). – С. 127-129.

6. Акбердина Л. Что такое мессенджер? Популярные мобильные мессенджеры. – від 9 березня, 2014. [Електронний ресурс] *URL*: *http://fb.ru/article/139644/chto-takoe-messendjer-populyarnyie-mobilnyie-messendjeryi* (Дата звернення: 28.04.2021)

7. Офіційний сайт *Telegram*. [Електронний ресурс]. – Режим доступу: *https://telegram.org/* (Дата звернення: 01.05.2021).

8. Официальный сайт *Telegram store*. [Електронний ресурс]. – Режим доступу: https://telegram-store.com/ (Дата звернення: 04.05.2021).

9. Создаем Telegram бота на *API.AI* [Електронний ресурс] – 2017. – Режим доступу до ресурсу: *https://habr.com/ru/post/336668/*.

10. Боты: информация для разработчиков [Електронний ресурс] – 2016. – Режим доступу до ресурсу: *https://tlgrm.ru/docs/bots*

11. *Telegram APIs* [Електронний ресурс] – Режим доступу до ресурсу: *https://core.telegram.org/api#bot-api*

12. Инструкция по созданию *Telegram* ботов. Часть 5. Пишем *Telegram*

бота на *php* для работы через *longpolling* [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: *https://radiohlam.ru/telegram_bot_5/*

13. *Python Telegram bot api*. [Електронний ресурс] – Режим доступу до ресурсу*: https://github.com/eternnoir/pyTelegramBotAPI/#types*

14. Примеры ботов [Електронний ресурс]. – 2017. – Режим доступу до ресурсу*: https://tlgrm.ru/docs/bots/samples*

15. Примеры использования чат-ботов в бизнесе [Електронний ресурс]. – 2017. – Режим доступу до ресурсу*: https://vc.ru/flood/25197-business-bot*

16. Создание бота в *Telegram*. Основы [Електронний ресурс] – Режим доступу до ресурсу: *https://wibe.team/sozdanie-bota-v-telegram/*

17. *P.Mell, T.Grance. The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology*, 2011, sp. 800-145.

18. Кухаренко А.А. Облачные вычисления. Платформа *Windows Azure*: реферат / Кухаренко А.А. – Гомель, 2012. — 59 с.

19. Батура Т.В., Мурзин Ф.А., Семич Д.Ф. Облачные технологии: основные понятия, задачи и тенденции развития // ЭЛЕКТРОННЫЙ НАУЧНЫЙ ЖУРНАЛ:ПРОГРАММНЫЕ ПРОДУКТЫ, СИСТЕМЫ И АЛГОРИТМЫ – 2014. doi: 10.15827/2311-6749.10.141

20. Воробьёв Андрей Игоревич Модели и методы повышения эффективности предоставления информационных услуг в центрах обработки данных: дис. на соиск. учен. степ. канд. техн. наук: 24.13.01: защищена 24.01.12 / Воробьёв Андрей Игоревич. — Санкт-Петербург, 2012. — 144 с.

21. Яремко І.М. Імовірнісні характеристики центрів обробки даних і резервування / І.М. Яремко, В.В. Турупалов, І.О. Молоковский // Наукові праці інституту проблем модулювання в енергетиці ім. Г.Є. Пухова "Моделювання та інформаційні технології". – Київ, 2011 р. - Випуск 60. – С.141-146.

22. *SQLite vs MySQL vs PostgreSQL*: сравнение систем управления базами данных [Електронний ресурс] Режим доступу: *https://devacademy.ru/article/sqlite-vs-mysql-vs-postgresql*

23. *Heroku* — обзор, плюсы и минусы программы, отзывы. [Електронний ресурс] Режим доступу: *https://8d9.ru/program/heroku*

24. Как разместить телеграм бота на *Heroku* [Електронний ресурс] Режим доступу: *https://qna.habr.com/q/615463*

25. Преимущества *Microsoft Azure* [Електронний ресурс] Режим доступу: *https://premier.region35.ru/preimushchestva-microsoft-azure.dhtm*.

26. *Google Cloud Vs AWS* [Електронний ресурс] Режим доступу: *https://coderlessons.com/tutorials/veb-razrabotka/osnovy-amazon-web-services-aws/9-google-cloud-vs-aws*

27. Чат-боты и мессенджеры: чем они могут быть полезны бизнесу [Електронний ресурс] Режим доступу: *https://vc.ru/marketing/134130-chat-boty-i-messendzhery-chem-oni-mogut-byt-polezny-biznesu*

28. Топ-10 мов програмування в Україні, 2010–2021. Динамічна інфографіка [Електронний ресурс] Режим доступу: *https://dou.ua/lenta/articles/top-10-lang-in-ukraine/*

29. Основи алгоритмізації та програмування [Електронний ресурс] Режим доступу:

https://sites.google.com/site/liseyoap/osnovni-ponatta-movi-programuvanna

30. Чем полезен корпоративный чат-бот [Електронний ресурс] Режим доступу: *https://www.vedomosti.ru/management/blogs/2018/05/15/769431-korporativnii-chat-bot*

31. Как создать *Telegram*-бота с помощью библиотеки *python-telegram-bot* [Електронний ресурс] Режим доступу: *https://highload.today/kak-sozdat-telegram-bot-na-python-poshagovoe-rukovodstvo/*

32. *Telegram*-бот на *Python*: от первой строчки до запуска на *Heroku* [Електронний ресурс] Режим доступу: *https://tproger.ru/translations/telegram-bot-create-and-deploy/*

33. Что такое Чат-Бот: Определение и Руководство [Електронний ресурс] Режим доступу: *https://sendpulse.ua/ru/support/glossary/chatbot*

34. Все о чат-ботах: типы и примеры, какому бизнесу подойдет, список конструкторов для создания [Електронний ресурс] Режим доступу: *https://web-promo.ua/blog/vse-o-chat-botah-tipy-i-primery-kakomu-biznesu-podojdet-spisok-konstruktorov-dlya-sozdaniya/*

35. Топ-5 платформ для создания чат-бот [Електронний ресурс] Режим доступу: *https://aiconference.ru/ru/article/top-5-platform-dlya-sozdaniya-chat-bot-95969 © AI Conference*

36. Девять платформ для создания чат-бот магазина в *Telegram* [Електронний ресурс] Режим доступу: *https://vc.ru/services/182007-devyat-platform-dlya-sozdaniya-chat-bot-magazina-v-telegram*

37. Популярные среды разработки и их недостатки [Електронний ресурс] Режим доступу*: https://gb.ru/posts/ide_negative*

38. Среды разработки в программировании [Електронний ресурс] Режим доступу: *https://it-black.ru/sredy-razrabotki-v-programmirovanii/*