ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка Telegram-боту для аналізу складу косметичних засобів»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Анна ТИЩЕНКО
(підпис)

Виконала: здобувачка вищої освіти групи ПД-41

Анна ТИЩЕНКО

Керівник: Ігор ГАМАНЮК
ст. викладач кафедри ІПЗ

Рецензент: _____

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Тищенко Анні Володимирівні

1. Тема кваліфікаційної роботи: «Розробка Telegram-боту для аналізу складу косметичних засобів»
керівник кваліфікаційної роботи старший викладач кафедри Ігор ГАМАНЮК,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024 р. № 36.

2. Строк подання кваліфікаційної роботи «28» травня 2024 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, мова програмування Python, методи аналізу складу косметичних засобів, вимоги до розробки Телеграм чат-бота, методи обробки та зберігання даних.

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

1. Аналіз предметної області.
2. Постановка задачі та методи дослідження.
3. Вибір засобів реалізації чат-бота.
4. Функціональні вимоги.
5. Нефункціональні вимоги.
6. Діаграма послідовності системи

7. Архітектура системи.
8. Опис програмного забезпечення.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів.
2. Вимоги до Telegram-бота.
3. Програмні засоби реалізації.
4. Діаграма варіантів використання.
5. Діаграма діяльності.
6. Діаграма послідовності.
7. Діаграма розгортання.
8. Схема бази даних.
9. Екранні форми.
10. Апробація результатів дослідження.

6. Дата видачі завдання «28» лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	28.02.-06.03.2024	
2	Аналіз предметної області	07.03-13.03.2024	
3	Визначення вимог до структури та алгоритму роботи чат-бота, формування технічного завдання.	14.03.-20.03.2024	
4	Вибір засобів та інструментів розробки	22.03-25.03.2024	
4	Проєктування системи	27.03-4.04.2024	
5	Програмна реалізація чат-бота	5.04.-23.04.2024	
6	Тестування розробленого чат – бота	25.04-29.04.2024	
7	Оформлення роботи: вступ, висновки, реферат	29.04-05.05.2024	
8	Розробка демонстраційних матеріалів	06.05-12.05.2024	
9	Попередній захист роботи	13.05-31.05.2024	

Здобувачка вищої освіти _____ Анна ТИЩЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Ігор ГАМАНЮК
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: __58__ стор., __2___ табл., __41__ рис., __11__ джерел.

Мета роботи – забезпечення зручності отримання інформації та аналізу складу косметичних засобів за допомогою Telegram-бота.

Об'єкт дослідження – процес аналізу складу косметичних засобів.

Предмет дослідження – Telegram-бот для аналізу складу косметичних засобів.

Короткий зміст роботи: дипломна робота присвячена розробці телеграм-бота для аналізу складу косметичних засобів з використанням мови програмування Python. В роботі проаналізовані наявні сучасні аналогічні програмні забезпечення, визначено їх переваги та недоліки, а також складена порівняльна таблиця. Описано процес надання пропозицій по догляду за певним типом шкіри та волосся. Реалізовано базу даних SQLite3, яка містить зберігає інформацію про тип шкіри, волосся, засобу та інформацію про косметичні продукти: посилання, фото та опис. Ця база даних дозволяє швидко отримувати потрібну інформацію та надавати користувачам пропозиції косметичних продуктів для певного типу шкіри або волосся.

Telegram-бот був створений за допомогою BotFather. Було отримано токен доступу, який використовується для ідентифікації та взаємодії з Telegram API. Завдяки використанню бібліотеки Aiogram для взаємодії з Telegram API та бібліотеки Asyncio для асинхронного програмування, забезпечено швидкий та надійний обмін повідомленнями між ботом та користувачем.

Для виконання функції аналізу складу косметичних засобів була інтегрована бібліотека g4f для роботи з моделлю GPT-4, що дозволяє боту аналізувати склад косметичних засобів та надавати користувачам інформацію про безпеку та

ефективність інгредієнтів.

Користувачі мають можливість переглянути інформацію про бота, про правильне використання продуктів та відправити відгук щодо бота адміністратору.

Діаграми розгортання та послідовності описують технічні аспекти впровадження рішення, демонструючи як різні компоненти системи взаємодіють між собою. Робота також включає аналіз функціональних та нефункціональних вимог, що забезпечують надійну та ефективну роботу телеграм-бота.

Після реалізації основної функціональності, бот був протестований для виявлення та виправлення помилок. Тестування проводилось на різних етапах розробки, щоб забезпечити стабільність та надійність роботи бота.

КЛЮЧОВІ СЛОВА: TELEGRAM-БОТ, PYTHON, GPT4, SQLITE, КОСМЕТИЧНІ ЗАСОБИ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API - Application Programming Interface

NLP - Neuro-linguistic programming

CRM - Customer Relationship Management

EWG - Environmental Working Group

INS - Iodine Number of Soap

SQL - Structured Query Language

UML - Unified Modeling Language

GPT - Generative Pre-trained Transformer

IDE - Integrated Development Environment

ВСТУП

Наразі косметичні засоби відіграють важливу роль у щоденному житті людей, сприяючи їхньому зовнішньому вигляду та самопочуттю. Однак разом зі зростанням популярності косметики зростає і питання щодо безпеки її складу. Широкий асортимент продуктів та складність інгредієнтів, які вони містять, ускладнюють завдання споживачам обирати косметику, яка відповідає їхнім потребам та вимогам. З огляду на зростаючий інтерес до здорового способу життя та екологічної свідомості, дослідження складу косметичних засобів стає дуже актуальним. Користувачі все більше прагнуть знати, що саме вони наносять на свою шкіру та волосся. Також, з поширенням соціальних мереж та інтернет-ресурсів, з'являється більше можливостей обмінюватися інформацією про косметичні продукти.

Об'єкт дослідження – процес аналізу складу косметичних засобів.

Предмет дослідження – Telegram-бот для аналізу складу косметичних засобів.

Мета роботи – забезпечення зручності отримання інформації та аналізу складу косметичних засобів за допомогою Telegram-боту.

Задачі дипломної роботи:

1. Провести аналіз предметної області що охоплює концепцію та актуальність Telegram-ботів у сфері косметики.
2. Провести аналіз існуючих сервісів для аналізу складу косметичних засобів, визначити актуальність Telegram-ботів.
3. Визначити функціональні та нефункціональні вимоги програмного забезпечення.
4. Визначити послідовність етапів для проведення аналізу складу косметичних засобів та надання пропозицій по догляду за певним типом шкіри та типом волосся.
5. Змоделювати варіанти використання Telegram-бота.

6. Спроектувати діяльність телеграм-бота, взаємодію об'єктів, розгортання застосунку.
7. Обрати ресурс для отримання інформації щодо косметичних продуктів.
8. Розробити телеграм-бота відповідно до вимог та використання необхідних технологій для реалізації функціональності програми.
9. Провести тестування Telegram-бота.

Методи дослідження - аналіз літературних джерел з теми аналізу складу косметичних засобів, вивчення принципів роботи та алгоритмів чат-ботів, розробка алгоритму аналізу складу косметичних засобів на основі мови програмування Python, програмування та тестування чат-бота, оцінка ефективності чат-бота.

Практичне значення результатів: розроблений чат-бот підвищує інформованість споживачів, допомагає споживачам уникнути використання продуктів з небезпечним складом та допомагає швидко та зручно отримати інформацію про склад косметичних засобів без необхідності самостійного дослідження інгредієнтів.

Галузь використання – споживачі косметики, магазини косметики.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальна концепція чат-ботів та їх популярність

У сучасному цифровому світі чат-боти відіграють значну роль у комунікаційних технологіях, значно спрощуючи взаємодію між людьми та комп'ютерами. Чат-боти, або просто боти, – це програми, які використовують ШІ для імітації розмов з користувачами через текстові або голосові інтерфейси. Вони можуть виконувати різні завдання: від надання інформації до виконання складних операцій, таких як бронювання квитків або керування фінансами.

Telegram-боти – це автоматизовані програми, що працюють всередині месенджера Telegram і виконують різноманітні завдання на основі запитів користувачів. Вони можуть надавати інформацію, автоматизувати рутинні процеси, здійснювати інтерактивні операції, вести діалоги та багато іншого. Боти у Telegram створені за допомогою API Telegram Bot, яке дозволяє розробникам інтегрувати їхні програми з Telegram.

Телеграм-боти можуть використовуватися в різних сферах, таких як інформаційні боти для погоди, новин або курсів валют, сервісні боти для замовлення їжі, бронювання квитків або проведення опитувань, розважальні боти для ігор та вікторин, освітні боти для навчання та тестування знань, а також корпоративні боти для внутрішньокорпоративного використання та автоматизації бізнес-процесів. Вони значно розширюють функціональні можливості месенджера Telegram, надаючи користувачам інструменти для спрощення та автоматизації багатьох аспектів їхнього життя та роботи.

Чат-боти можуть бути інтегровані в різні платформи, такі як месенджери, веб-сайти, мобільні додатки та навіть голосові помічники. Вони можуть працювати в режимі 24/7, що забезпечує безперервну підтримку користувачів і значно підвищує ефективність обслуговування.

Telegram є одним з найпопулярніших месенджерів у світі, здобувши визнання завдяки своїй швидкості, безпеці та багатофункціональності. Telegram швидко набув популярності завдяки ряду унікальних особливостей, які вигідно відрізняють його від конкурентів. Основною перевагою Telegram є безпека та конфіденційність: месенджер використовує шифрування для захисту повідомлень, а також пропонує можливість створювати секретні чати з кінцевим шифруванням і налаштуванням самознищення повідомлень. Ще одна важлива перевага Telegram — його швидкість та стабільність. Месенджер відомий своєю швидкодією навіть при слабкому інтернет-з'єднанні, завдяки потужній інфраструктурі серверів, розташованих по всьому світу. Telegram також приваблює користувачів своєю відкритістю та кросплатформеністю: застосунок доступний на всіх основних платформах, включаючи iOS, Android, Windows, macOS та Linux, забезпечуючи синхронізацію повідомлень на всіх пристроях. Багатофункціональність є ще однією сильною стороною Telegram: користувачі можуть створювати групові чати з необмеженою кількістю учасників, канали для трансляції повідомлень великій аудиторії, використовувати боти для автоматизації завдань та інтеграції з іншими сервісами, а також надсилати різноманітні медіафайли і документи. Крім того, Telegram надає відкритий API, що дозволяє розробникам створювати власні боти, інструменти та інтеграції, сприяючи створенню великої екосистеми додатків і сервісів навколо месенджера. Завдяки цим унікальним особливостям, таким як висока безпека, швидкість роботи, кросплатформеність і багатофункціональність, Telegram став одним з найпопулярніших і найвикористовуваніших месенджерів у світі, привертаючи мільйони користувачів і компаній.

На рис. 1.1 зображена кількість користувачів Telegram з 2014 по 2024 рік у мільйонах.

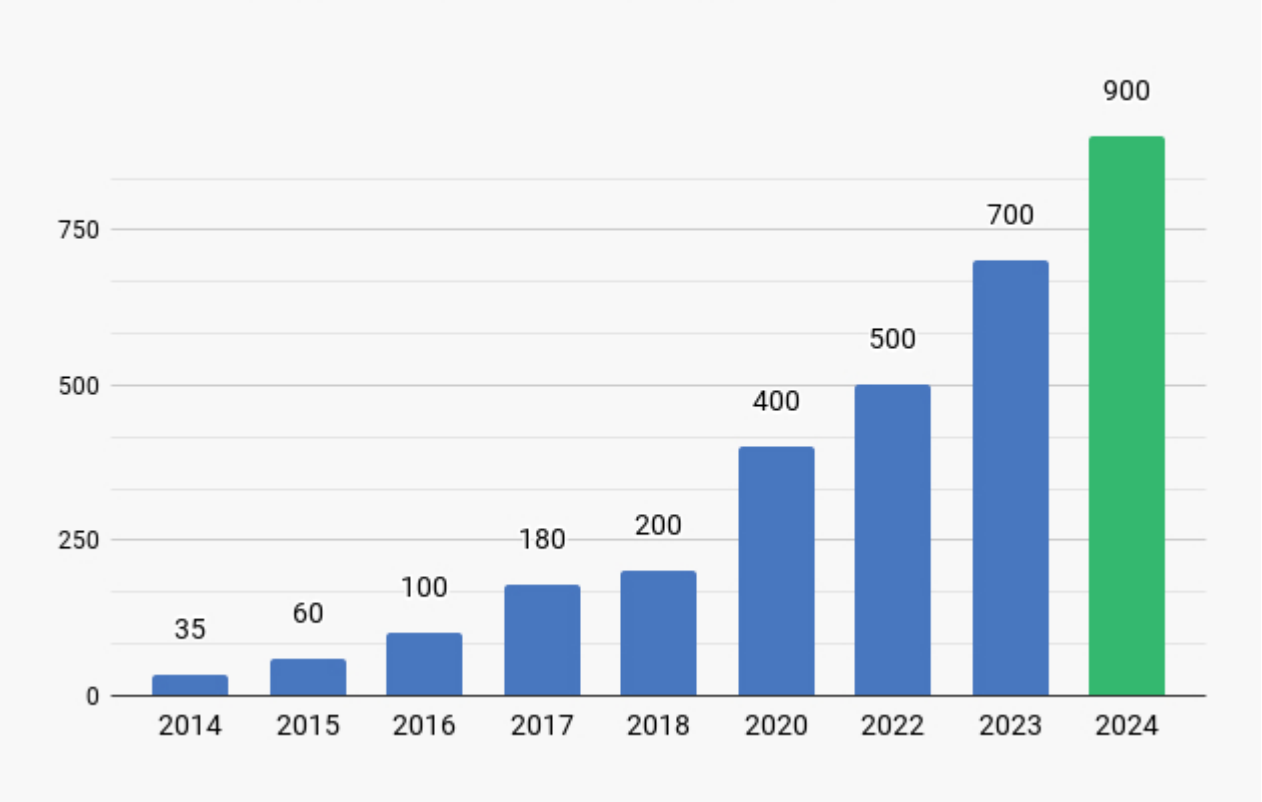


Рис. 1.1 Кількість користувачів Telegram з 2014 року по 2024 у мільйонах

Чат-боти набули значної популярності у сучасному світі завдяки своїй здатності автоматизувати взаємодію між користувачами та сервісами. Використовуючи технології штучного інтелекту, такі як обробка природної мови (NLP) та машинне навчання, чат-боти можуть надавати персоналізовані відповіді, виконувати різноманітні завдання та забезпечувати цілодобову підтримку.

Чат-боти здатні автоматизувати рутинні завдання, такі як відповіді на часті запитання, бронювання, прийом замовлень, що підвищує ефективність роботи компаній. Вони знижують навантаження на персонал, дозволяючи співробітникам зосередитися на більш складних та творчих завданнях. Завдяки аналізу даних про користувачів, чат-боти можуть надавати персоналізовані рекомендації та відповіді, що покращує користувацький досвід. Вони запам'ятовують попередні взаємодії та вподобання користувачів, забезпечуючи більш релевантні та точні відповіді.

Можливість інтегруватися з різними платформами та сервісами, такими як CRM-системи, системи управління контентом, платіжні системи робить чат-ботів універсальними інструментами для бізнесу. Вони можуть виконувати складні

багатоступінчасті завдання, залучаючи інформацію з різних джерел. Використання чат-ботів дозволяє компаніям знизити витрати на обслуговування клієнтів, зменшити необхідність у великому штаті операторів підтримки, а також зменшує ймовірність людських помилок, що може призвести до економії коштів.

1.2 Актуальність чат-ботів у сфері косметики

У сучасному світі, де ринок косметичних засобів постійно зростає і з'являються нові продукти щодня, для споживачів стає все складніше зробити усвідомлений вибір та підібрати безпечні та якісні засоби. Зростаюча популярність здорового способу життя та дбайливого ставлення до себе спонукає споживачів уважніше аналізувати склад продуктів, які вони використовують щодня.

Однією з важливих тенденцій є зростання популярності здорового способу життя. Люди все більше усвідомлюють, що їхнє фізичне та психічне здоров'я залежить від їхнього харчування, фізичної активності та здорових звичок. У зв'язку з цим, вони звертають більше уваги на те, що вони вживають усередину та наносять на свою шкіру.

Зростання обізнаності про шкідливі інгредієнти в косметичних засобах також сприяє цій тенденції. Споживачі все більше оглядають етикетки продуктів та уважно вивчають їх склад. Вони уникають продуктів, які містять шкідливі хімічні речовини та перевагають натуральні та органічні альтернативи.

У результаті цих тенденцій, споживачі стають більш свідомими щодо свого вибору косметичних продуктів. Вони шукають продукти, які відповідають їхнім потребам, цінностям та уявленням про здоров'я та красу. Таким чином, свідоме споживання стає не просто тенденцією, а важливим етапом у формуванні здорового та збалансованого способу життя для багатьох людей.

Чат-боти забезпечують безперервний доступ до інформації про косметику в режимі 24/7, що робить їх надзвичайно зручними для користувачів у будь-який час. Особливе значення має можливість надавати персоналізовані рекомендації та поради щодо підбору косметичних засобів з урахуванням особливостей шкіри та

вподобань кожної окремої людини. Крім того, чат-боти можуть надавати користувачам детальну інформацію про інгредієнти, властивості та ефекти різних косметичних продуктів. Завдяки спілкуванню та наданню цінної інформації вони стають ефективним інструментом для залучення нових клієнтів та утримання існуючих.

Загалом, чат-боти стають все більш важливим інструментом для компаній косметичної галузі, допомагаючи покращити обслуговування клієнтів, збільшити продажі та підвищити рівень задоволеності користувачів продукцією.

Цільовою аудиторією чат-бота для аналізу складу косметичних засобів є споживачі, які піклуються про своє здоров'я. Це люди, які обирають продукти, що містять безпечні та натуральні складові, і активно уникають шкідливих хімічних речовин. Вони шукають інструменти, які допоможуть їм зробити обдуманий вибір косметики.

Також можна виділити осіб з алергічними реакціями або чутливою шкірою, тому що такі люди потребують продуктів, які не викликають подразнень або алергічних реакцій. Для них важливо мати доступ до інформації про склад косметичних засобів для уникнення потенційно небезпечних інгредієнтів.

Наступні це покупці, які прагнуть до екологічних та етичних стандартів: це споживачі, які обирають продукти, вироблені з урахуванням екологічних та етичних принципів, таких як відсутність тестування на тваринах та використання відновлювальних матеріалів.

Окрім чат-ботів, існують й інші ресурси, що спеціалізуються на аналізі косметичних інгредієнтів. Серед них - веб-сайти та мобільні додатки, які надають користувачам можливість ознайомитися з детальним описом продуктів, включаючи їх склад, властивості та відгуки користувачів.

Ці ресурси зазвичай пропонують широкий асортимент продукції із різних брендів, а також фільтри та інструменти пошуку, що полегшують процес знаходження потрібного продукту.

Існує декілька аналогічних додатків, які аналізують складові косметичних продуктів такі як Safety Makeup, SkinCarisma, Inci Decoder, Cosmosweet, CosDNA.

1.3 Safety Makeup

Сайт Safety Makeup – це український онлайн-ресурс, що спеціалізується на аналізі косметичних інгредієнтів з точки зору безпеки для здоров'я. Сервіс пропонує комплексний підхід до аналізу та оцінки косметичних продуктів і їхніх інгредієнтів. Основна мета сайту - надати користувачам інформацію про потенційно шкідливі компоненти в косметиці та рекомендувати безпечні альтернативи.

На головній сторінці SafetyMakeup.ua представлений каталог інгредієнтів косметичних засобів, кожен з яких має детальний опис. Користувачі можуть знайти інформацію про походження інгредієнта, його функції у складі продукту, а також можливі побічні ефекти.

Сайт також містить розділи з корисними статтями, які охоплюють різні аспекти догляду за шкірою, волоссям та загальним здоров'ям. Ці статті можуть включати рекомендації щодо вибору косметики для різних типів шкіри, поради з догляду за проблемною шкірою, а також огляди нових косметичних продуктів і тенденцій на ринку краси.

Однією з ключових особливостей SafetyMakeup.ua є його орієнтація на безпеку косметичних засобів. Кожен інгредієнт оцінюється на основі рейтингів BIODIZIONARIO та EWG (рис. 1.2). BIODIZIONARIO використовує п'ятибальну шкалу для оцінки безпеки компонентів, від "зелений" (безпечний) до "червоний" (небезпечний або не рекомендований), тоді як EWG застосовує десятибальну шкалу, де 1-2 означає низький ризик, 3-6 – середній, а 7-10 – високий ризик для здоров'я. Це дозволяє користувачам легко зрозуміти, наскільки безпечним є той чи інший продукт. з точки зору його впливу на здоров'я, що дозволяє користувачам зрозуміти, які компоненти можуть бути потенційно шкідливими, а які є безпечними і корисними. Ця інформація особливо важлива для людей з чутливою шкірою або алергіями.

Крім того, Safety Makeup активно заохочує користувачів використовувати натуральну та екологічно чисту косметику. Сайт містить багато інформації про

натуральні інгредієнти, їхні переваги та ефективність у порівнянні з синтетичними аналогами. Це робить ресурс корисним для тих, хто прагне використовувати екологічно чисті продукти та зменшити свій вплив на довкілля.

Особливо корисним є те що Safety Makeup пропонує список б’юті-спеціалістів України відображаючи їх спеціалізацію, досвід роботи та контакти. Користувачі можуть отримувати професійні поради та рекомендації щодо вибору косметики від досвідчених фахівців, що робить процес вибору продукції ще більш інформованим та надійним.

Сайт Safety Makeup пропонує не лише аналіз безпеки косметичних інгредієнтів, а й корисні статті по догляду за шкірою. Ці статті охоплюють різні аспекти догляду, включаючи рекомендації щодо вибору безпечних та ефективних продуктів, поради по догляду за різними типами шкіри, а також інформацію про сучасні тренди в косметології. Завдяки цим матеріалам користувачі можуть отримати цінні знання для підтримки здоров'я та краси своєї шкіри, використовуючи лише безпечні та перевірені засоби.

Safety Makeup пропонує різноманітні продукти, виготовлені з натуральних інгредієнтів, такі як косметика, веганські варіанти, корейські косметичні засоби та багато іншого. Спеціалізовані магазини, які продають ці товари, зазвичай пропонують асортимент екологічно чистої косметики, що відповідає високим стандартам якості та безпеки. Ці продукти часто не містять хімікатів, парабенів, сульфатів, штучних ароматизаторів та інших потенційно подразнюючих або токсичних речовин. Крім того, вони сертифіковані як веганські та екологічно чисті. Це робить їх привабливими для людей, які шукають екологічно чисті засоби для догляду за собою.

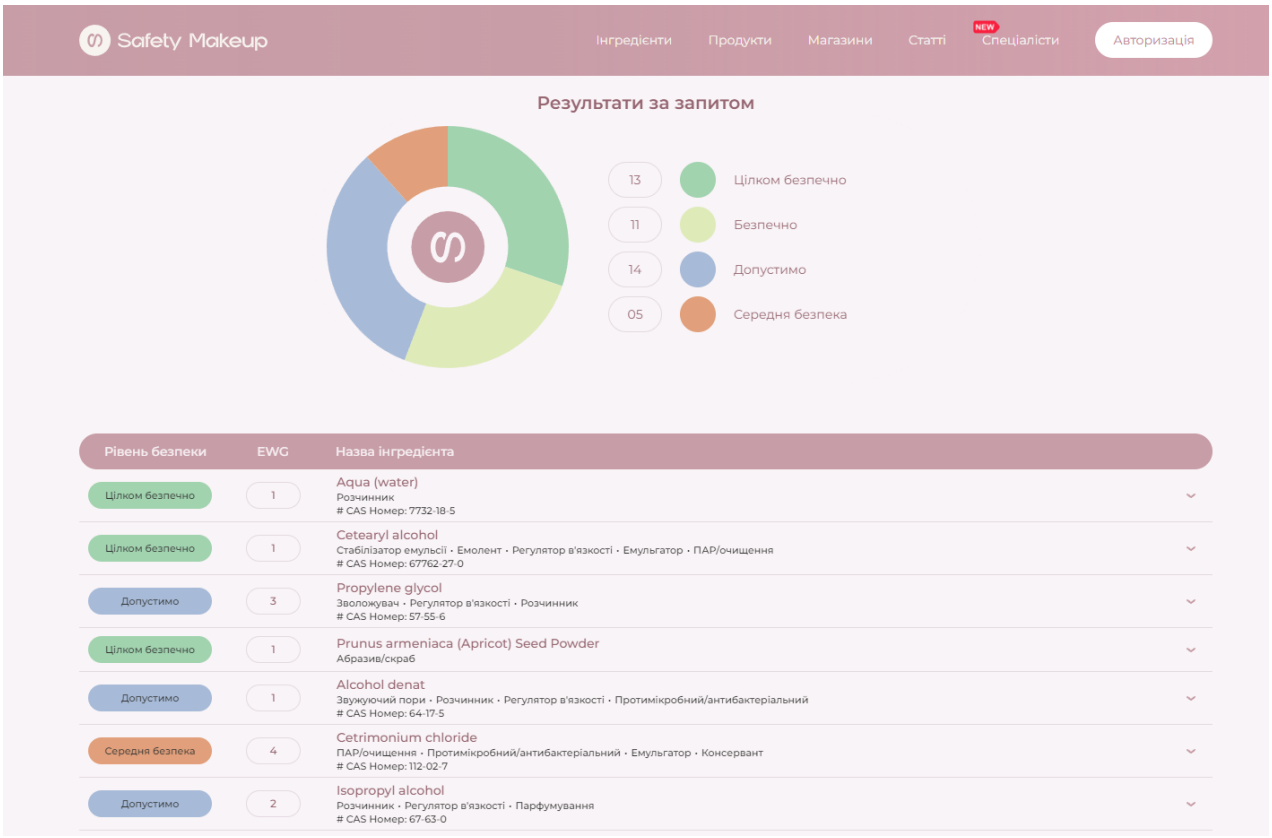


Рис. 1.2 Приклад розшифрування компонентів косметики

1.4 SkinCarisma

Skin Carisma — це онлайн-платформа, створена для допомоги користувачам у виборі безпечних і ефективних косметичних продуктів. Його головною функцією є аналізатор інгредієнтів косметичних продуктів і продуктів по догляду за шкірою, який дозволяє користувачам вводити список інгредієнтів будь-якого продукту для отримання детального аналізу. Цей інструмент визначає наявність різних інгредієнтів, у тому числі потенційно шкідливих, і надає інформацію про їх вплив на різні типи та стани шкіри.

На Skin Carisma представлений широкий каталог косметичних продуктів з детальним описом кожного інгредієнта. Користувачі можуть дізнатися про походження інгредієнта, його функціональність у складі продукту, а також можливі негативні наслідки для шкіри та здоров'я в цілому. Це особливо важливо для людей з чутливою шкірою, алергіями або специфічними дерматологічними проблемами.

Однією з ключових функцій сайту є можливість аналізу інгредієнтів косметичних продуктів. Користувачі можуть просто ввести список інгредієнтів продукту, щоб отримати детальний звіт про їх функції та безпеку. Ця інформація включає дані про те, чи є інгредієнти комедогенними, потенційними подразниками, або ж мають якісь інші шкідливі властивості (рис. 1.4).

На головній сторінці SkinCarisma представлена зручна функція пошуку косметичних продуктів за категоріями та уподобаннями користувачів (рис. 1.3). Інтерфейс складається з двох випадаючих меню і кнопки для пошуку.

Перше меню "I am looking for" дозволяє користувачам вибрати категорію продукту, наприклад, очищувачі, зволожувачі, засоби для догляду за волоссям та інші. Це допомагає швидко знайти необхідний тип косметики серед широкого асортименту.

Друге меню "My product preferences are" дає можливість вказати особисті вподобання щодо продуктів, такі як відсутність певних інгредієнтів (наприклад, парабенів чи сульфатів), або наявність специфічних властивостей, як-от гіпоалергенність або органічність. Це дозволяє звузити пошук і знайти продукти, які найбільше відповідають індивідуальним потребам користувача.

Кнопка "Show Products" запускає пошук відповідних продуктів на основі обраних категорій та вподобань. Під полями вибору розміщене посилання "Looking for something specific? Search by product or brand name", яке перенаправляє на розширений пошук за назвою продукту або бренду для більш конкретного запиту.

Ця функція робить процес пошуку косметичних засобів легким та інтуїтивно зрозумілим, забезпечуючи користувачів необхідною інформацією для прийняття обґрунтованих рішень щодо догляду за шкірою.

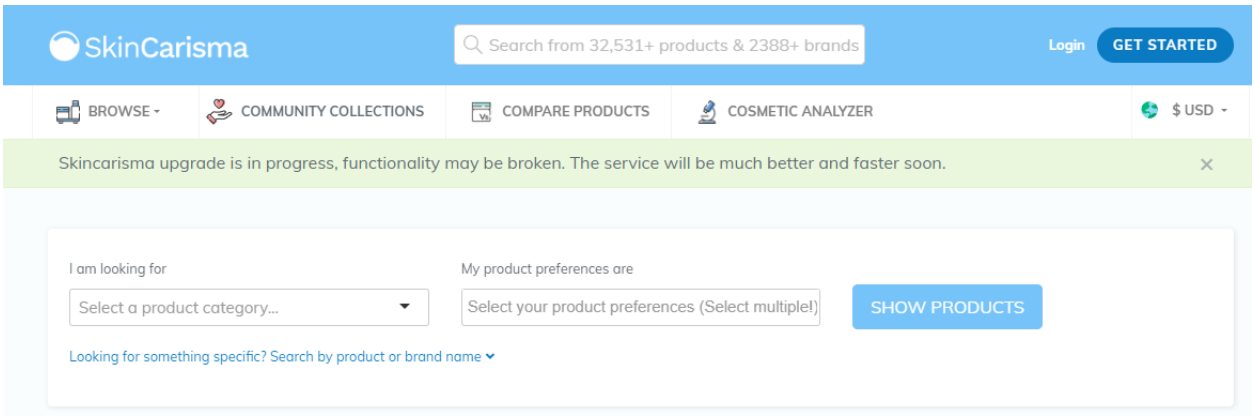


Рис. 1.3 Функція пошуку за категорією продукту та особистими вподобаннями

Функція порівняння продуктів на SkinCarisma дозволяє користувачам детально оцінити різні косметичні засоби перед покупкою. Вона забезпечує можливість порівняти кілька продуктів за складом інгредієнтів, рейтингом безпеки, користувацькими оцінками та відгуками. Користувачі можуть обрати продукти для порівняння з профілів або пошукових результатів, після чого відкривається сторінка з таблицею порівняння. У ній детально розглядаються всі інгредієнти, їхні характеристики та потенційні ризики. Рейтинг безпеки для кожного продукту базується на наукових дослідженнях. Також відображаються середні оцінки та кількість відгуків користувачів, що допомагає оцінити популярність та ефективність продуктів. Функція дозволяє швидко порівняти продукти, економлячи час, і приймати обґрунтовані рішення щодо вибору косметики, що відповідає індивідуальним потребам.

Крім того, Skin Carisma має активну спільноту користувачів, які можуть залишати відгуки про продукти, створювати колекції з улюбленими засобами (рис. 1.5). Це дозволяє створювати інтерактивне середовище, де користувачі можуть обговорювати різні аспекти догляду за шкірою та обмінюватися порадами.

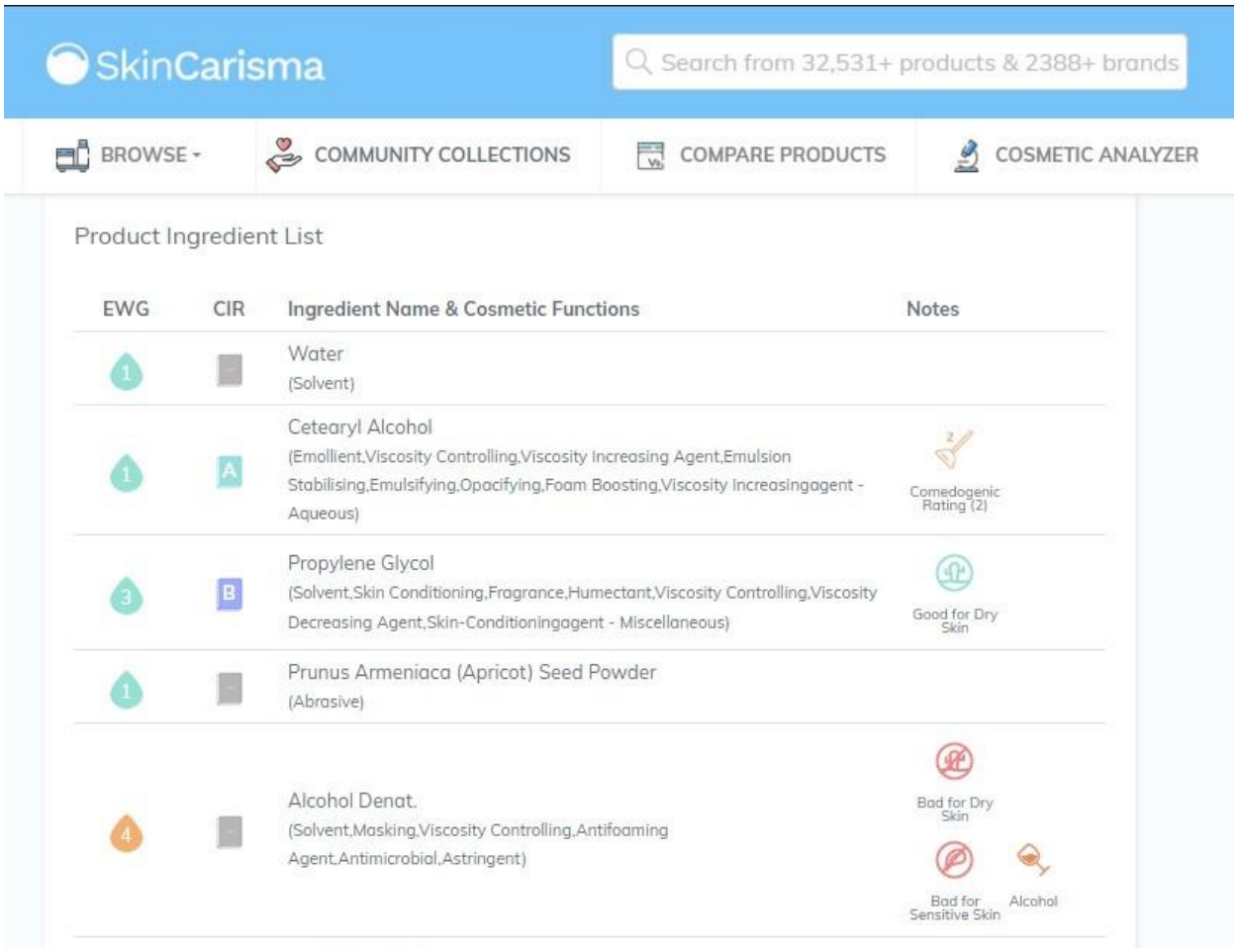


Рис. 1.4 Приклад списку проаналізованих інгредієнтів продукту

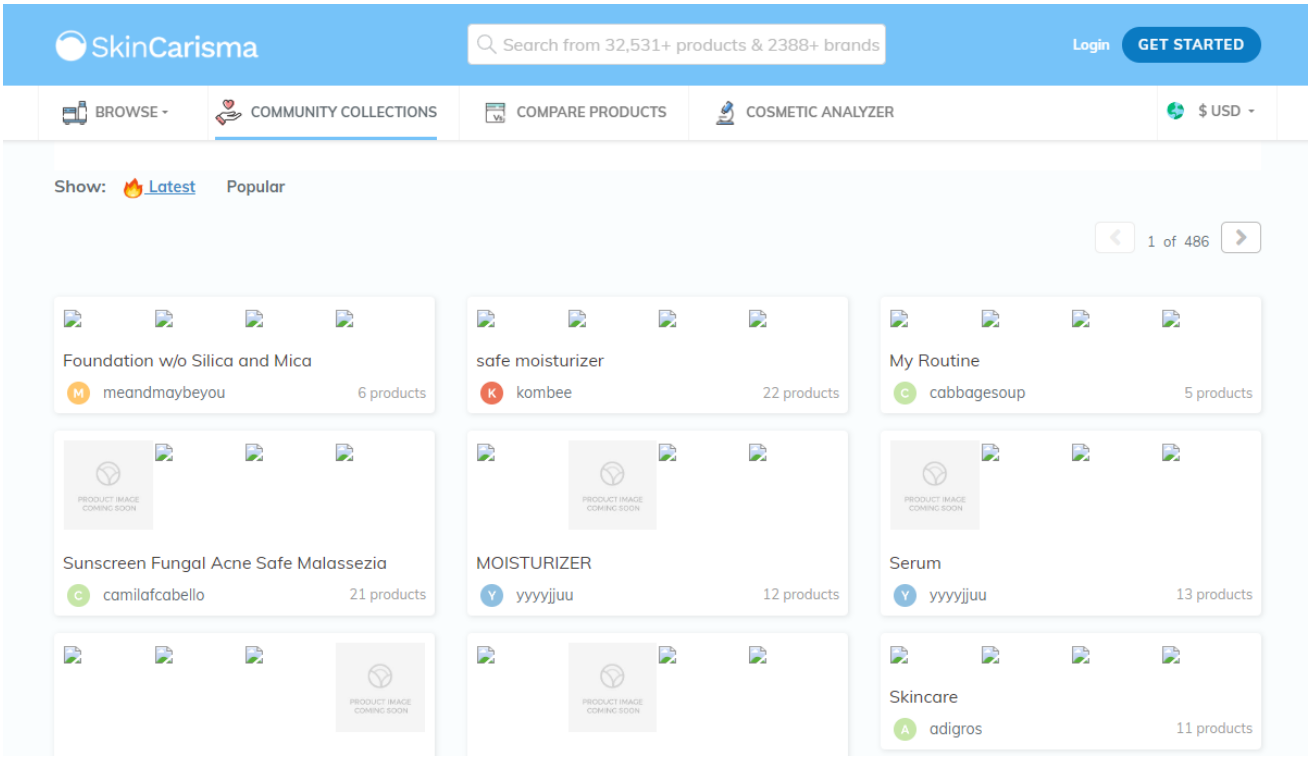


Рис. 1.5 Колекції продуктів від учасників спільноти SkinCarisma

1.5 Cosmosweet

Cosmosweet — це онлайн-платформа, що надає детальний аналіз та інформацію про склад косметичних засобів. Її основна мета — допомогти споживачам робити обґрунтовані вибори, базуючись на детальній інформації про інгредієнти косметики, яку вони використовують.

Сервіс дозволяє користувачам знаходити косметичні продукти за допомогою введення назви продукту або сканування його штрих-коду. Це забезпечує швидкий доступ до детальної інформації про склад продукту, включаючи аналіз безпеки кожного інгредієнта. Користувачі можуть використовувати цю функцію для перевірки складу продуктів, які вони планують придбати або вже використовують, що допомагає зробити обґрунтований вибір. Для цього достатньо ввести назву продукту або скористатися функцією сканування штрих-коду через мобільний застосунок чи веб-сайт. Крім того, після введення назви та вибору потрібного засобу, користувач може перевірити на скільки відсотків цей засіб йому підходить. Користувачу потрібно заповнити анкету, де потрібно вказати чи чутлива в нього шкіра та чи є алергія на певні компоненти. Після цього сайт генерує діаграму, де показані переваги, недоліки та додаткова інформація у кількості відсотків, але для того щоб побачити детальні пояснення потрібно придбати «Преміум» версію (рис. 1.6).

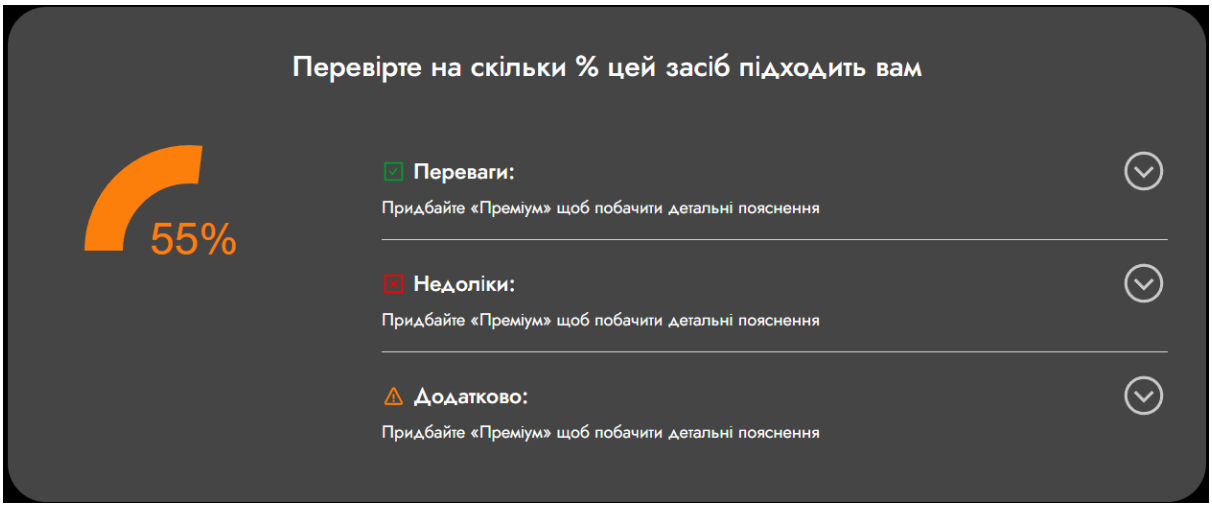


Рис. 1.6 Відсоткова діаграма відповідності засобу потребам користувача

Cosmosweet також пропонує функцію пошуку косметичних засобів за складом. Користувачі можуть ввести або скопіювати список інгредієнтів у спеціальне поле на сайті, щоб отримати опис кожного інгредієнта та його оцінку безпеки. Ця функція дозволяє користувачам знайти косметичні продукти, що містять конкретні інгредієнти, які вони хочуть уникати або, навпаки, включити у свій догляд за шкірою. Такий підхід допомагає краще зрозуміти, які продукти є найбільш підходящими для індивідуальних потреб користувачів.

Сайт також забезпечує інструмент для детального аналізу компонентів косметичних продуктів. Користувачі можуть ввести склад продукту, після чого сайт надає інформацію про кожен інгредієнт, включаючи його функції, походження та рівень безпеки. Аналіз компонентів представлений у вигляді діаграм, що робить результати візуально зрозумілими та легкими для інтерпретації (рис. 1.7). Ці діаграми включають дані про класифікацію та рівень безпеки інгредієнтів. Таке відображення дозволяє користувачам візуально оцінити потенційні ризики та користь від використання конкретного косметичного продукту.

Сервіс містить відгуки користувачів і рейтинги косметичних продуктів, що дозволяє користувачам вчитися на досвіді інших і приймати зважені рішення. Платформа допомагає визначити наявність певних компонентів, таких як силікони, алергени, сульфати, спирти, парабени, УФ-фільтри, вітаміни та зволожуючі чи протигрибкові інгредієнти. Ця розширена функціональність допомагає користувачам робити безпечніший і обґрунтованіший вибір косметичних продуктів.

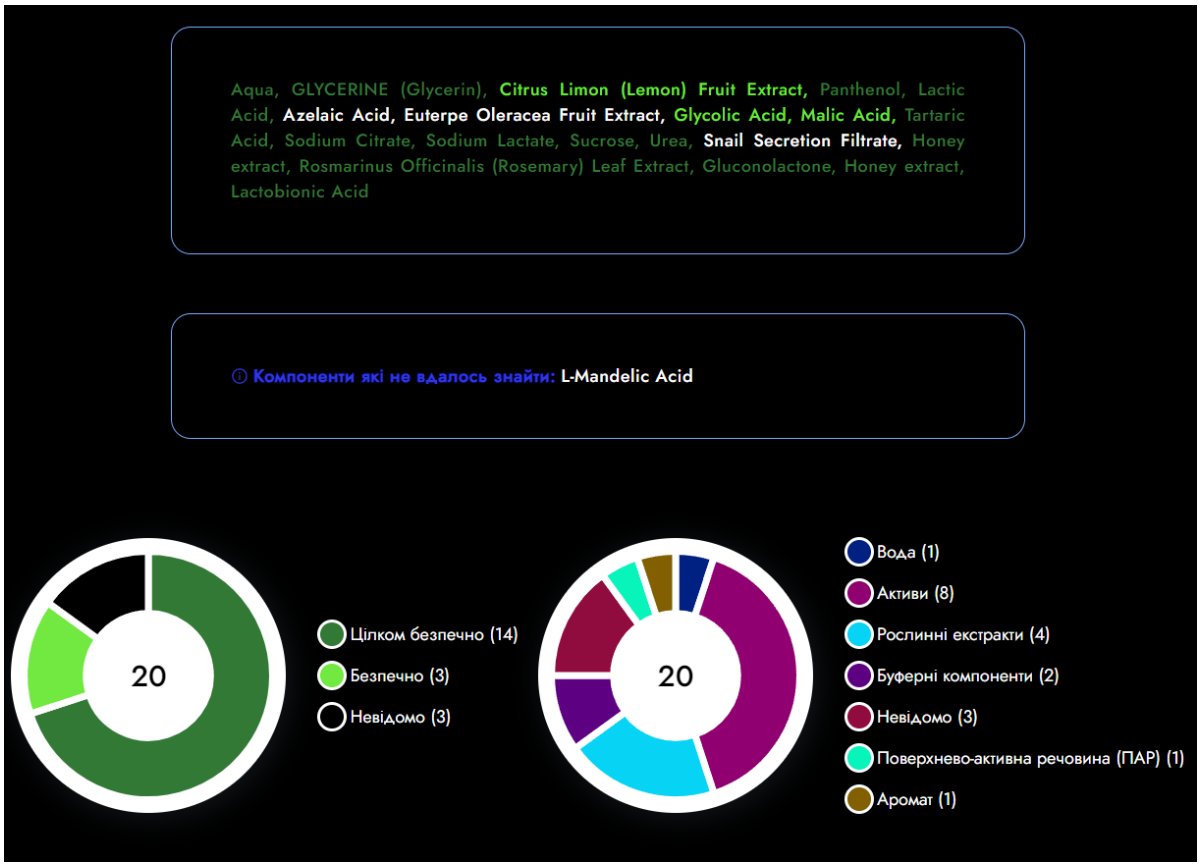


Рис. 1.7 Приклад діаграми рівня безпеки та класифікації інгредієнтів

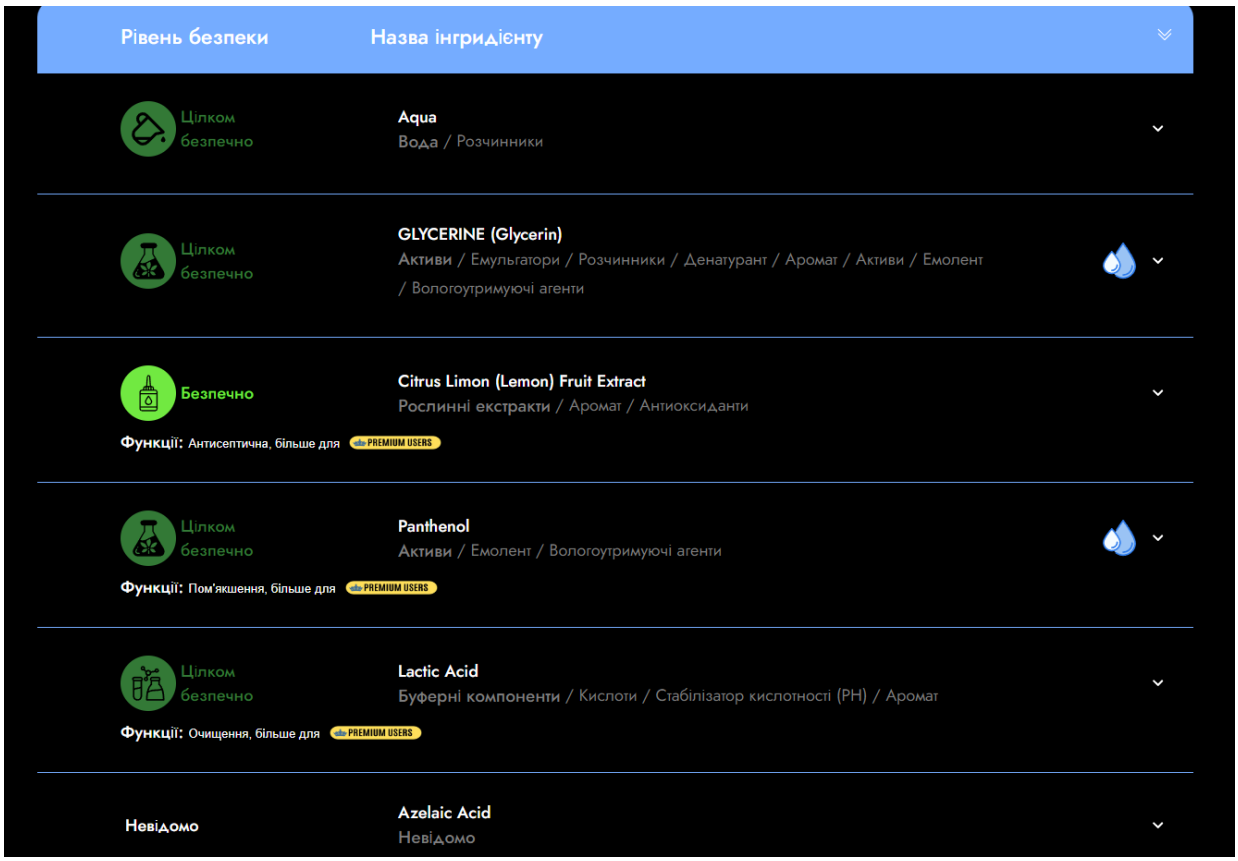


Рис. 1.8 Приклад розшифрування інгредієнтів

1.6 CosDNA

CosDNA.com — це онлайн-ресурс, який надає детальну інформацію про косметичні продукти та їхні інгредієнти. Основна функція сайту — допомогти користувачам зрозуміти склад і потенційний вплив різних косметичних і косметичних засобів.

Користувачі можуть шукати продукти за назвою або штрих-кодом, що дозволяє їм швидко знаходити списки інгредієнтів і відповідну інформацію. Сайт класифікує інгредієнти на основі їх функції, безпеки, потенціалу спричинення вугрів (комедогенність) та ймовірності спричинення подразнення шкіри. Ця класифікація відображається в зрозумілому форматі таблиці, супроводжується числовими оцінками, які вказують на серйозність кожної потенційної проблеми (рис. 1.9).

На сайті є можливість залишати відгуки для зареєстрованих користувачів про продукти у вигляді шкали від «Жахливо» до «Добре». Ця функція дозволяє користувачам швидко та легко виразити свою думку про продукт, що допомагає іншим користувачам швидко орієнтуватися у відгуках.

Також користувачі можуть оцінювати достовірність таблиці з інгредієнтами, що надається для кожного продукту. Оцінка достовірності забезпечує зворотній зв'язок і допомагає підтримувати високий рівень точності та надійності інформації на CosDNA.com.

CosDNA.com також пропонує функціональність форуму користувачів, де можна обговорювати різні косметичні продукти та їхні інгредієнти. Користувачі можуть створювати теми, задавати питання та ділитися своїм досвідом з іншими учасниками спільноти. Форум є корисним ресурсом для отримання додаткової інформації та порад від реальних користувачів, що допомагає приймати більш обґрунтовані рішення щодо вибору косметичних засобів.

Однією з примітних особливостей CosDNA є таблиця, яка допомагає користувачам розрахувати рецепт для виготовлення мила в домашніх умовах (рис. 1.10). Таблиця дозволяє вводити дані про різні види олій, їх вагу в грамах та

відсоток у загальному складі. Додатково, внизу таблиці є розрахунки для співвідношення олії та води, а також можливість додавання різних добавок до мила. Користувачі можуть вводити дані про добавки, їх масу і відсотковий вміст у загальній рецептурі.

Ingredient	Function	Acne	Irritant	Safety
Aqua	Solvent			1
Glycerine	Solvent, Viscosity Control, Skin conditioning, Moisturizer	0	0	1-2
Citrus Limon Peel Extract	Skin conditioning, Emollient			1-5
Panthenol	Antistatic, Skin conditioning, Moisturizer, Anti-inflammatory	0	0	1
L-Mandelic Acid	Anti-inflammatory			
Lactic Acid	Skin conditioning, Exfoliator, Astringent			1-4
Azelaic Acid	Fragrance, Whitening			1
Euterpe Oleracea Fruit Extract				1
Glycolic Acid	Exfoliator, Whitening			1-4

Рис. 1.9 Приклад розшифрування інгредієнтів

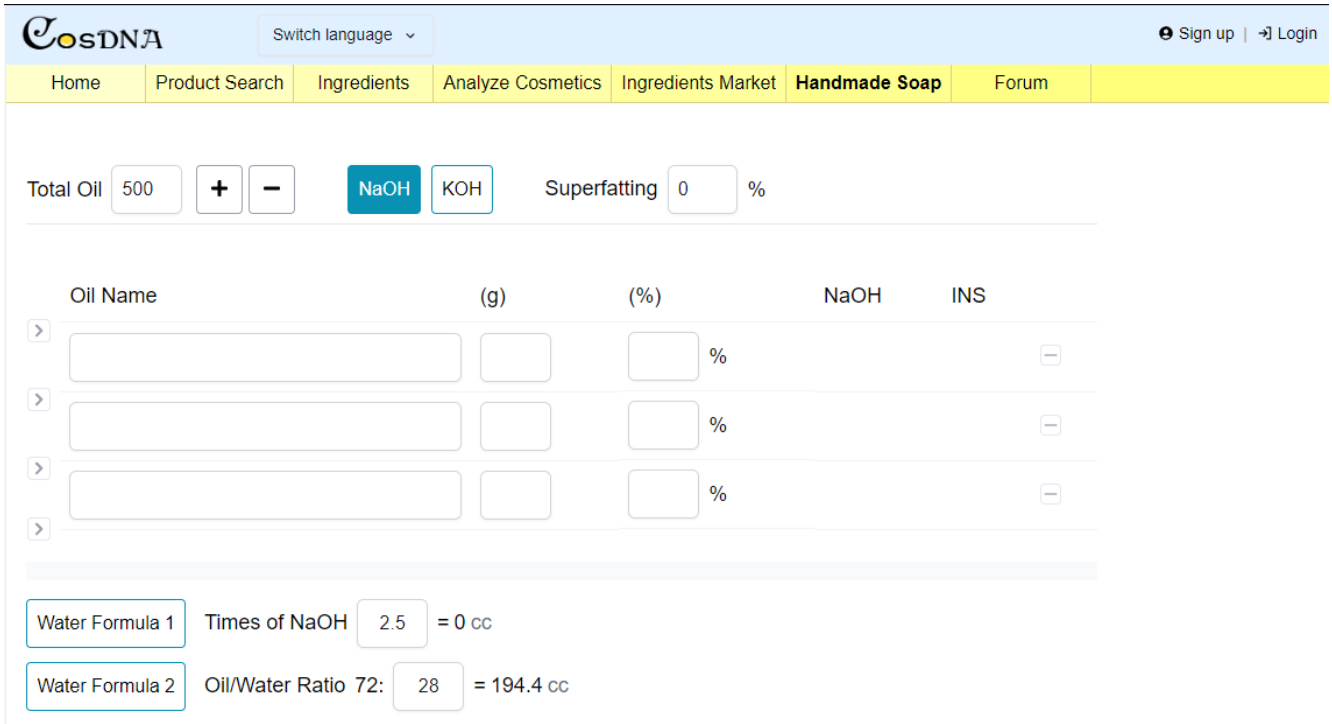


Рис. 1.10 Таблиця для розрахування рецепту для виготовлення мила

1.7 Таблиця порівнянь аналогів

Таблиця 1.1

Порівняння існуючих сервісів для аналізу складу косметичних засобів

Сервіс / Функція	Safety Makeup	Skin Carisma	Cosmo sweet	CosDNA	SkinCare Assistant
Пропозиції продуктів, згідно до особливостей шкіри та волосся	-	+	-	+	+
Інтеграція з Telegram	-	-	-	-	+
Оцінка безпеки інгредієнтів	+	+	+	+	+
Українська локалізація	+	-	+	-	+
Можливість відправлення відгуків/ форум	-	+	+	+	+

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

Провівши аналіз предметної області та дослідивши актуальність створення чат-бота для аналізу складу косметичних засобів, також сформувавши мету, об’єкт та предмет дослідження було визначено, що для створення чат-бота необхідно виконати наступні задачі проєкту:

- реалізувати команду для запуску чат-бота;
- реалізувати розділ для виведення розшифрованих складових косметичних засобів;
- реалізувати розділ для виведення пропозицій щодо косметичних засобів, які підходять до певного типу шкіри обличчя та волосся.
- реалізувати розділ для ознайомлення користувача з ботом.
- можливість відправлення відгуку або пропозицій для покращення роботи чат-бота.
- реалізувати розділ для інформування користувача про правильне використання догляду.

Для розробки вдалого чат-бота передбачено виконання функціональних та нефункціональних вимог.

Функціональні вимоги:

- забезпечення інформації щодо безпеки та функціоналу інгредієнтів косметичних засобів.
- надання пропозицій щодо засобів, які підходять до конкретного типу шкіри та волосся.
- обробка непередбачених сценаріїв при введенні користувачем некоректних запитів.
- можливість залишити відгук для покращення роботи застосунку.

Нефункціональні вимоги:

- отримання інформації протягом 1-2 хвилин.
- інтеграція з Telegram.

- спілкування чат-бота з користувачем у неформальному стилі.

2.1 Вибір засобів реалізації чат-бота

У процесі планування розробки програми необхідно визначити інструменти та технології, які будуть використовуватися. Ці інструменти включають:

- мову програмування;
- середовище розробки;
- використані інструменти;

2.1.1 Python

Для написання телеграм бота було обрано мову програмування Python з кількох важливих причин, які роблять її оптимальним вибором для цього проєкту.

Python відомий своєю простотою та читабельністю, що дозволяє розробникам швидко створювати та підтримувати код. Завдяки чистому та зрозумілому синтаксису, Python сприяє більшій продуктивності розробників, особливо на етапах написання та тестування коду. Також Python має величезну кількість готових бібліотек та фреймворків, які значно спрощують розробку телеграм ботів. Зокрема, фреймворк aiogram забезпечує зручні інструменти для взаємодії з API Telegram, що робить процес створення бота більш ефективним. Крім цього, він підтримує асинхронне програмування через бібліотеки такі як asyncio, що дозволяє створювати високопродуктивні телеграм боти. Асинхронне програмування дозволяє ефективно обробляти численні запити від користувачів одночасно, що є критично важливим для забезпечення швидкого відгуку та масштабованості бота. Мова Python легко інтегрується з багатьма іншими сервісами та технологіями, такими як бази даних, веб-фреймворки та брокери повідомлень. Це дозволяє створювати комплексні та багатофункціональні рішення для телеграм ботів.

Для написання коду було обрано середовище розробки PyCharm. Це рішення було прийнято з огляду на кілька ключових факторів, які роблять PyCharm оптимальним вибором для даного проєкту.

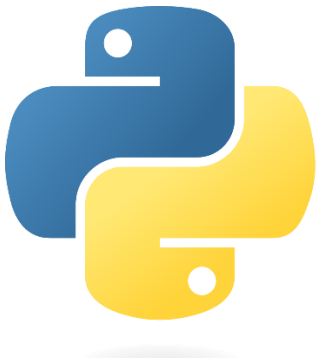


Рис. 2.1 Логотип мови програмування Python

2.1.2 PyCharm

PyCharm є потужним інтегрованим середовищем розробки (IDE) від JetBrains, спеціально створеним для розробників на Python. Він забезпечує широкий набір інструментів і функцій, що значно полегшують процес розробки. PyCharm підтримує розширене автозаповнення, що допомагає розробникам швидко писати код з меншою кількістю помилок. Завдяки цьому інструменту, IDE може передбачати наступні частини коду на основі контексту, що значно прискорює процес написання коду. Рефакторинг коду в PyCharm дозволяє легко та безпечно перейменовувати змінні, методи і класи, що допомагає підтримувати чистоту та читабельність коду. Вбудований відладчик дозволяє з легкістю знаходити та виправляти помилки в коді. Відладчик підтримує встановлення брейкпоінтів, крокування через код, і перегляд змінних у реальному часі, що значно підвищує ефективність розробки.

Крім того, PyCharm підтримує інтеграцію з багатьма популярними інструментами та фреймворками, дозволяє зручно працювати з базами даних, що важливо для взаємодії з SQLite. Інтеграція з менеджером пакетів pip дозволяє легко

встановлювати, оновлювати та видаляти пакети, що значно спрощує управління залежностями у проєкті.

PyCharm дозволяє зручно працювати з базами даних. Вбудовані інструменти дозволяють підключатися до різних баз даних, таких як PostgreSQL, MySQL, SQLite, та інших, виконувати SQL-запити, переглядати та редагувати дані.

Підтримується інтеграція з менеджером пакетів pip, що дозволяє легко встановлювати, оновлювати та видаляти пакети. Це значно спрощує управління залежностями у проєкті, дозволяючи тримати всі необхідні пакети актуальними та синхронізованими з проєктом.

Забезпечується підтримка різних інструментів тестування, таких як unittest, pytest, і nose. Це дозволяє створювати, запускати та аналізувати тести прямо з IDE, що сприяє підтримці високої якості коду.

Підтримується інтеграція з такими системами контролю версій як Git, SVN, і Mercurial. Це забезпечує зручне керування версіями коду, спільну роботу над проєктами та відстеження змін.

До PyCharm входить розширена підтримка плагінів, що дозволяє користувачам розширювати функціональність IDE під свої потреби. Користувачі можуть встановлювати додаткові плагіни для підтримки нових мов, інструментів, або інтеграцій.

Рис. 2.2 Логотип середи розробки PyCharm

2.1.3 SQLite

Для зберігання даних у проєкті було обрано базу даних SQLite, враховуючи її численні переваги. SQLite є легкою у використанні базою даних, яка не потребує встановлення та налаштування серверного програмного забезпечення. Це дає змогу швидко розпочати роботу з базою даних SQL та зосередитись на розробці функціональності додатка. Вбудованість SQLite означає, що вона працює безпосередньо всередині додатка, зменшуючи накладні витрати на адміністрування та дозволяючи зберігати дані у форматі файлів, які легко розповсюджувати разом із додатком. Вона забезпечує високу продуктивність для малих та середніх проєктів, що важливо для швидкого відгуку телеграм бота. Відсутність зовнішніх залежностей спрощує розгортання та підтримку додатка. SQLite підтримує більшість стандартних SQL-запитів, включаючи SELECT, INSERT, UPDATE, DELETE, та інші, що спрощує міграцію даних та інтеграцію з іншими системами. Це означає, що розробники, які вже знайомі з SQL, можуть легко адаптуватися до роботи з SQLite. Файли бази даних SQLite є повністю переносимими, що забезпечує гнучкість у розробці та розгортанні. Крім того, оскільки SQLite є вбудованою та не потребує адміністрування, витрати на її підтримку мінімальні, дозволяючи зменшити операційні витрати та зосередитись на розвитку функціональності додатка. Переваги SQLite включають простоту, високу продуктивність, легкість інтеграції та мінімальні витрати на підтримку, що робить її ідеальним вибором для зберігання даних у проєкті, де важлива швидкість розробки та ефективність роботи.

Рис. 2.3 Логотип реляційної бази даних SQLite

2.1.4 Aiogram

Aiogram — це сучасний та потужний асинхронний фреймворк для створення Telegram ботів на мові програмування Python. Він дозволяє легко і ефективно розробляти ботів, використовуючи всі можливості, які надає Telegram Bot API. Aiogram створений для асинхронного програмування на основі бібліотеки asyncio, що забезпечує високу продуктивність і дозволяє обробляти велику кількість запитів одночасно. Aiogram використовує asyncio для асинхронного виконання завдань, що дозволяє обробляти багато запитів паралельно без блокування основного потоку виконання. Це робить бота більш швидким та ефективним, особливо при великій кількості користувачів. Aiogram має простий та зрозумілий синтаксис, що полегшує процес розробки. Навіть новачки можуть швидко навчитися користуватися цим фреймворком. Aiogram надає широкі можливості для налаштування та розширення функціональності бота. Він підтримує всі методи Telegram Bot API, включаючи обробку повідомлень, команд, інтерактивних кнопок, опитувань і багато іншого. Фреймворк дозволяє організовувати код у вигляді модулів, що сприяє кращій структурованості та підтримуваності коду. Це особливо корисно для великих проєктів. Aiogram пропонує ряд вбудованих інструментів, таких як система середовища конфігурації, підтримка середовища розробки, а також інтеграція з іншими бібліотеками та сервісами.

При розробці Telegram бота на Python, Aiogram є ідеальним вибором завдяки своїм численним перевагам. Aiogram дозволяє швидко розпочати розробку бота, забезпечуючи легке налаштування та інтеграцію з Telegram Bot API. Досить лише кількох рядків коду, щоб створити основні функції бота. Фреймворк забезпечує зручну обробку подій, таких як отримання повідомлень від користувачів, натискання кнопок, виконання команд тощо. Це дозволяє розробникам зосередитися на логіці бота, а не на технічних деталях обробки запитів. Використання асинхронних функцій дозволяє ботам швидко реагувати на запити користувачів без затримок, забезпечуючи високу продуктивність і покращений користувацький досвід. Aiogram дозволяє легко інтегрувати бота з іншими веб-

сервісами та API, що розширює його можливості та функціональність. Наприклад, можна використовувати Aiogram для отримання даних з бази даних, взаємодії з зовнішніми API або виконання складних обчислень на сервері.

Рис. 2.4 Логотип фреймворку Aiogram

2.1.5 Asyncio

Асинхронність є важливим аспектом розробки сучасних додатків, зокрема чат-ботів, оскільки вона дозволяє обробляти велику кількість запитів одночасно без блокування виконання програми. У Python для цього часто використовується бібліотека asyncio, яка є стандартною бібліотекою для написання асинхронного коду з використанням синтаксису async і await. Вона забезпечує набір інструментів для роботи з асинхронними операціями, такими як мережеві запити, робота з файлами і багато іншого.

При створенні Telegram-бота на Python, asyncio використовується для забезпечення асинхронної взаємодії з Telegram API та обробки користувацьких запитів у реальному часі. Це дозволяє обробляти запити без блокування основного потоку програми, підвищуючи швидкість і ефективність роботи бота. Асинхронна обробка подій дозволяє створювати функції, які можуть одночасно обробляти кілька запитів, не чекаючи завершення обробки попередніх. Використання asyncio для конкурентності дозволяє запускати кілька завдань одночасно, що підвищує продуктивність і знижує затримки при взаємодії з базою даних або зовнішніми API. Бібліотека Aiogram, яка використовується для розробки Telegram-ботів на Python,

підтримує асинхронність і широко використовує можливості asyncio, дозволяючи обробляти повідомлення асинхронно, без блокування основного потоку програми.

2.1.6 GPT4

Бібліотека GPT4Free у Python є потужним інструментом для взаємодії з моделями штучного інтелекту, такими як GPT-4, без необхідності прямого доступу до платних API. Вона дозволяє використовувати можливості обробки природної мови (NLP) для виконання різних завдань, включаючи аналіз тексту, генерацію тексту та інші складні обчислення.

Ця Python-бібліотека забезпечує доступ до моделей GPT безпосередньо з вашого коду. Вона дозволяє отримувати результати від моделей, що працюють на основі GPT, використовуючи різні безкоштовні платформи або відкриті API.

У роботі з аналізу складу косметичних засобів головною метою було використання GPT4 для автоматизації процесу оцінки інгредієнтів косметики. Такий аналіз може включати ідентифікацію інгредієнтів, оцінку їхньої безпеки, визначення функціональних властивостей.

Відповідь від моделі може містити інформацію про кожен інгредієнт, яку далі можна обробляти, аналізувати та представляти у зручному для користувача вигляді. Використання GPT4Free для аналізу складу косметичних засобів дозволяє автоматизувати та спростити процес оцінки інгредієнтів, відкриваючи можливості для застосування передових технологій обробки природної мови у різних галузях, забезпечуючи при цьому безкоштовний доступ до потужних моделей штучного інтелекту.

Рис. 2.5 Логотип GPT4 AI

3 ПРОЄКТУВАННЯ ТЕЛЕГРАМ-БОТА

3.1 Алгоритм роботи телеграм-бота

Алгоритм роботи чат-бота включає кілька ключових компонентів, які забезпечують ефективну взаємодію між користувачем та системою. Основні етапи роботи чат-бота можна описати за наступною схемою: користувач - Telegram - Telegram конектор - ядро бота - база даних.

Користувач взаємодіє з чат-ботом через платформу Telegram. Він надсилає команду «/start» чаті з ботом. Це є початковою точкою контакту, де користувач може запитувати інформацію про склад косметичних засобів або переглянути інші функції бота. Після того, як користувач надсилає повідомлення, Telegram передає це повідомлення через свої сервери. Telegram забезпечує доставку повідомлень до зареєстрованого бота, використовуючи безпечні канали зв'язку. Бот отримує повідомлення у вигляді запиту через API Telegram.

Telegram конектор - це компонент, який приймає запити від Telegram API та передає їх до ядра бота. Він діє як посередник між Telegram та ядром бота, забезпечуючи правильну обробку запитів. Конектор приймає повідомлення від Telegram, розпізнає їхній тип (наприклад, текстові повідомлення, команди) та передає їх далі для обробки.

Ядро бота - це основний компонент, який обробляє запити користувачів. Після отримання повідомлення від конектора, ядро аналізує запит та виконує відповідні дії. Ядро обробляє повідомлення користувача, визначає необхідну дію. На основі результатів аналізу, ядро формує відповідь для користувача. Якщо запит стосується аналізу складу, відповідь міститиме інформацію про користь, шкоду та рівень безпеки інгредієнтів.

Ядро бота також взаємодіє з базою даних для зберігання та отримання інформації. База даних SQL використовується для збереження інформації про косметичні продукти.

Після формування відповіді ядро бота передає її назад через Telegram конектор. Конектор надсилає відповідь до Telegram API, який потім доставляє повідомлення користувачу. Таким чином, користувач отримує необхідну інформацію безпосередньо у чаті з ботом. Цей алгоритм роботи чат-бота забезпечує ефективну обробку запитів користувачів, надання точних відповідей та збереження історії взаємодій для подальшого аналізу та вдосконалення системи.

3.2 Моделювання варіантів використання застосунку

Діаграма варіантів використання є важливим інструментом у процесі проєктування та розробки програмного забезпечення, що допомагає візуалізувати взаємодію між користувачами системи та різними її компонентами. У контексті телеграм-бота, який призначений для аналізу складу косметичних засобів, ця діаграма допоможе краще зрозуміти, як актори взаємодіють з ботом.

Прецеденти використання в даній діаграмі включають наступні основні функції:

- аналіз складу косметики;
- надання персоналізованих пропозицій щодо засобів;
- отримання зворотного зв'язку від користувачів;
- список порад по догляду;
- надання інформації про бота;

Актори в даній діаграмі включають користувача, OpenAIGPT-4 та адміністратора. Користувач є головним актором, який взаємодіє з ботом для отримання інформації та послуг. OpenAIGPT-4 виступає в ролі обробника запитів, який аналізує склад косметичних засобів та надає відповідну інформацію. Адміністратор відповідає за налаштування та управління ботом, забезпечуючи його коректну роботу.

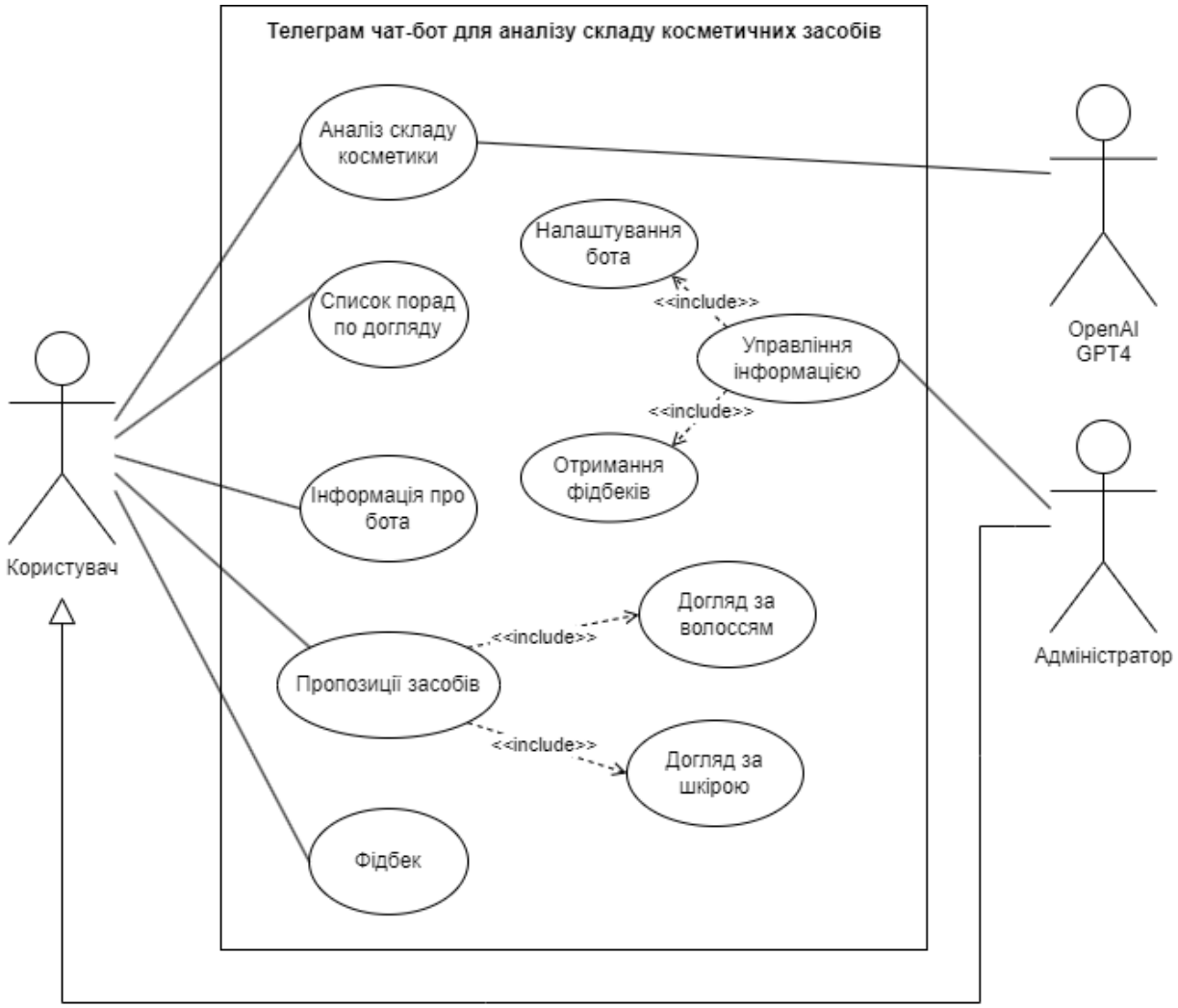


Рис. 3.1 Діаграма варіантів використання

3.3 Проектування діяльності застосунку

Діаграма діяльності відображає послідовність дій або робочий процес певної системи чи процесу.

Користувач взаємодіє з телеграм-ботом, вводячи текстовий опис складу косметичного засобу. Бот просить ввести склад, після чого користувач вводить текст, що містить перелік інгредієнтів косметичного засобу. Бот формує запит, включаючи введений користувачем текст, і передає його на обробку до GPT-4 через API.

Модель GPT-4 отримує запит від телеграм-бота. Запит приймається сервером, де працює GPT-4, і GPT-4 починає обробку запиту. GPT-4 аналізує текст

запиту для визначення змісту та контексту складу косметичного засобу, розпізнаючи інгредієнти у введеному тексті та визначаючи, які з них потребують додаткового аналізу або інформації. На основі цього аналізу, GPT-4 формує пошукові запити для збору необхідної інформації про інгредієнти. GPT-4 створює окремі запити для кожного інгредієнта або групи інгредієнтів, включаючи ключові слова, необхідні для отримання релевантної інформації.

Далі GPT-4 виконує пошук інформації, використовуючи сформовані пошукові запити, залучаючи внутрішні бази даних та зовнішні джерела для отримання інформації про інгредієнти, збираючи та узагальнюючи знайдену інформацію. На основі зібраної інформації GPT-4 формує зведену відповідь, структуруючи її у зрозумілий та логічний формат, і формує рекомендації або висновки щодо складу косметичного засобу.

Телеграм-бот отримує сформовану відповідь від GPT-4, приймає результати через API, перевіряє цілісність та коректність отриманих даних. Якщо аналіз пройшов успішно, бот надсилає користувачеві зведену інформацію про інгредієнти косметичного засобу. Якщо сталася помилка під час обробки, бот надсилає повідомлення про помилку та пропонує спробувати знову.

Алгоритм аналізу складу косметичних засобів зображений у вигляді діаграми на рисунку 3.2.

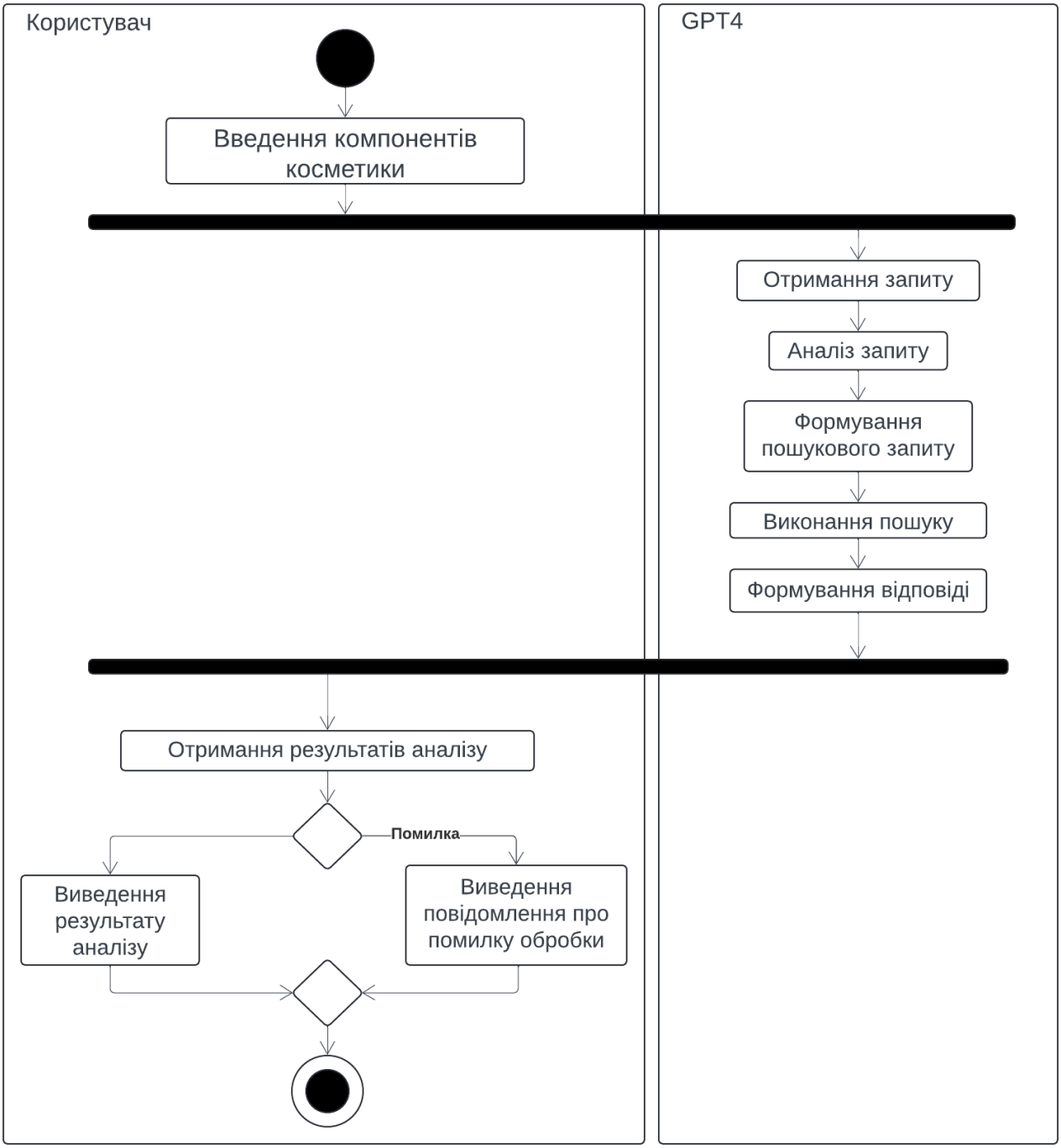


Рис. 3.2 Діаграма діяльності аналізу складу косметичних засобів

3.4 Проектування взаємодії об’єктів застосунку

Діаграма послідовності демонструє процес взаємодії користувача з телеграм-ботом для отримання персоналізованих пропозицій косметичних засобів.

Для забезпечення збереження інформації про косметичні засоби, зокрема посилань на продукти, їх опису та фото, у даному проєкті використовується база даних SQLite. SQLite є вбудованою базою даних, яка зручна для використання в мобільних та десктопних додатках завдяки своїй легкості та відсутності необхідності в налаштуванні серверного оточення.

Процес починається з того, що користувач обирає, для чого потрібен засіб (обличчя або волосся). Телеграм-бот приймає цей вибір та надає користувачу можливість вибрати тип шкіри або волосся. Після цього користувач обирає тип шкіри або волосся, а бот запитує базу даних для отримання відповідної інформації про тип шкіри або волосся, який позначений номером у базі даних. База даних повертає цю інформацію, і бот запитує користувача про вибір типу косметичного засобу для певного типу шкіри або волосся. Користувач обирає тип косметичного засобу, а бот запитує базу даних для отримання пропозицій косметичних засобів для вибраного типу шкіри або волосся. База даних повертає результати, що містять фото, опис та посилання на рекомендовані косметичні засоби. Телеграм-бот надсилає ці результати користувачу, завершуючи процес.

Діаграма послідовності механізму надання пропозицій по косметичним засобам зображена на рисунку 3.3.

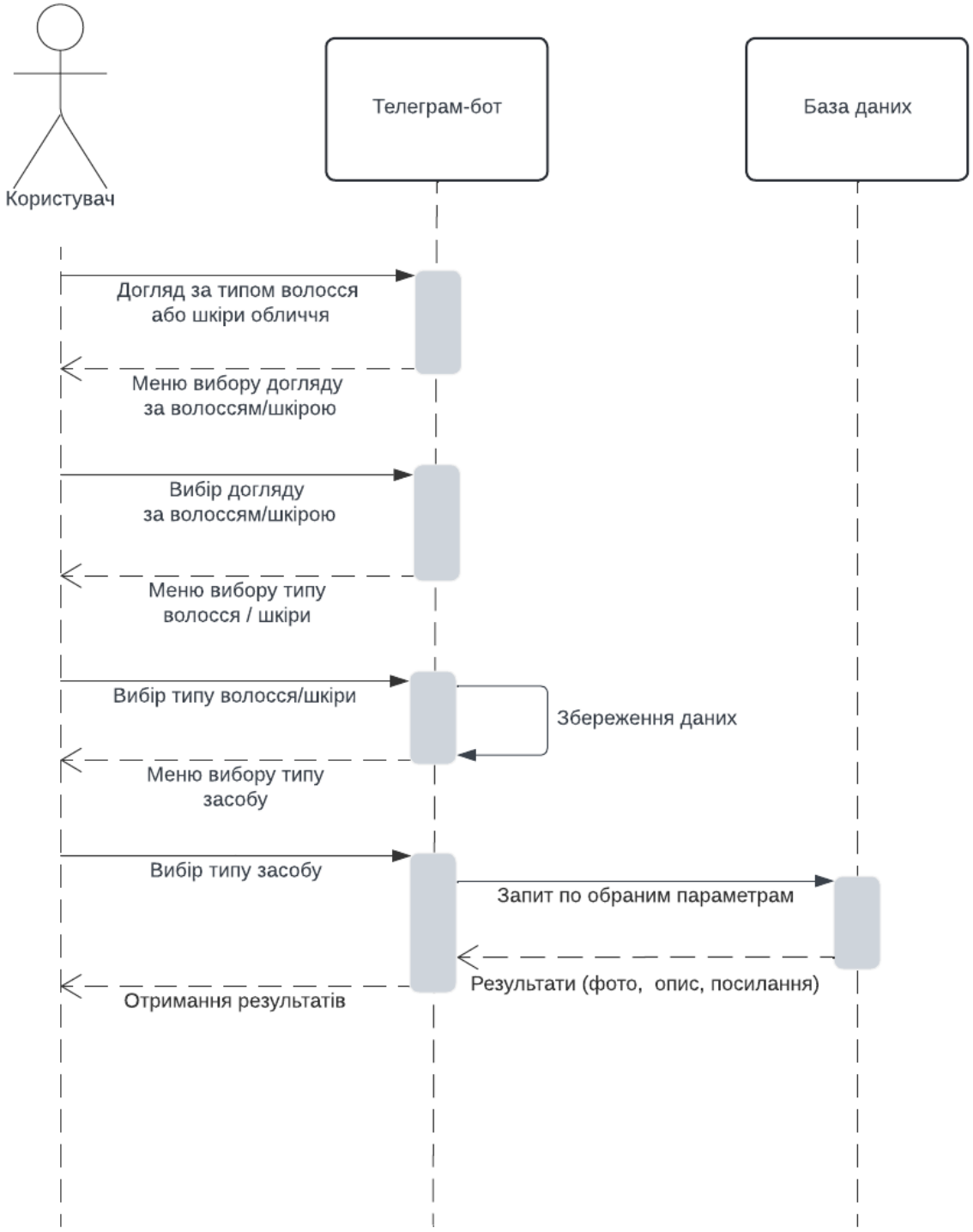


Рис. 3.3 Діаграма послідовності надання пропозицій по косметичним засобам

3.5 Проектування розгортання застосунку

Діаграма розгортання є одним із видів діаграм у моделюванні UML. Вона використовується для візуалізації фізичної архітектури системи, показуючи, як програмні компоненти та артефакти розгорнуті на фізичних пристроях. Діаграми розгортання відображають вузли, які представляють фізичне обладнання або віртуальні машини, і компоненти, які показують програмне забезпечення, розгорнуте на цих вузлах. Вони допомагають зрозуміти, як програмне забезпечення взаємодіє з фізичними ресурсами системи та як різні частини системи зв'язані між собою. Основні елементи діаграми розгортання включають вузли, компоненти, артефакти і з'єднання.

Для телеграм чат-бота для аналізу складу косметичних засобів діаграма розгортання виглядає наступним чином. Користувач взаємодіє з ботом через застосунок Telegram, надсилаючи запити до Telegram API. Telegram API передає ці запити до Telegram бота. Telegram бот обробляє запити, використовуючи Aiogram для взаємодії з Telegram API. Якщо запит потребує аналізу складу косметичних засобів, бот надсилає запит до GPT-4 API. Він обробляє запит і повертає результати аналізу ботам. Для отримання або збереження інформації про продукти бот звертається до бази даних SQLite , де зберігаються необхідні дані. Бот отримує дані з бази даних та використовує їх для формування відповіді користувачу. Бот надсилає відповідь користувачу через Telegram API, який передає її назад до додатка Telegram на пристрій користувача.

Компоненти діаграми включають зовнішню систему Telegram, GPT-4 API для аналізу складу косметичних засобів, та SQLite Database для зберігання інформації про продукти. Такі файли як main.py, inline.py, basic.py та settings.py, реалізують різні функції бота: імпорт бази даних, вибір даних з бази, реалізацію inline клавіатур, текст для кнопок, повідомлення від бота та аналіз складу косметичних засобів за допомогою GPT-4, а також налаштування бота, створення токену та Admin ID. Ця діаграма відображає, як компоненти та артефакти пов'язані

між собою та як дані передаються між ними для забезпечення функціональності Telegram бота для аналізу складу косметичних засобів.

Діаграма розгортання зображена на рисунку 3.4.

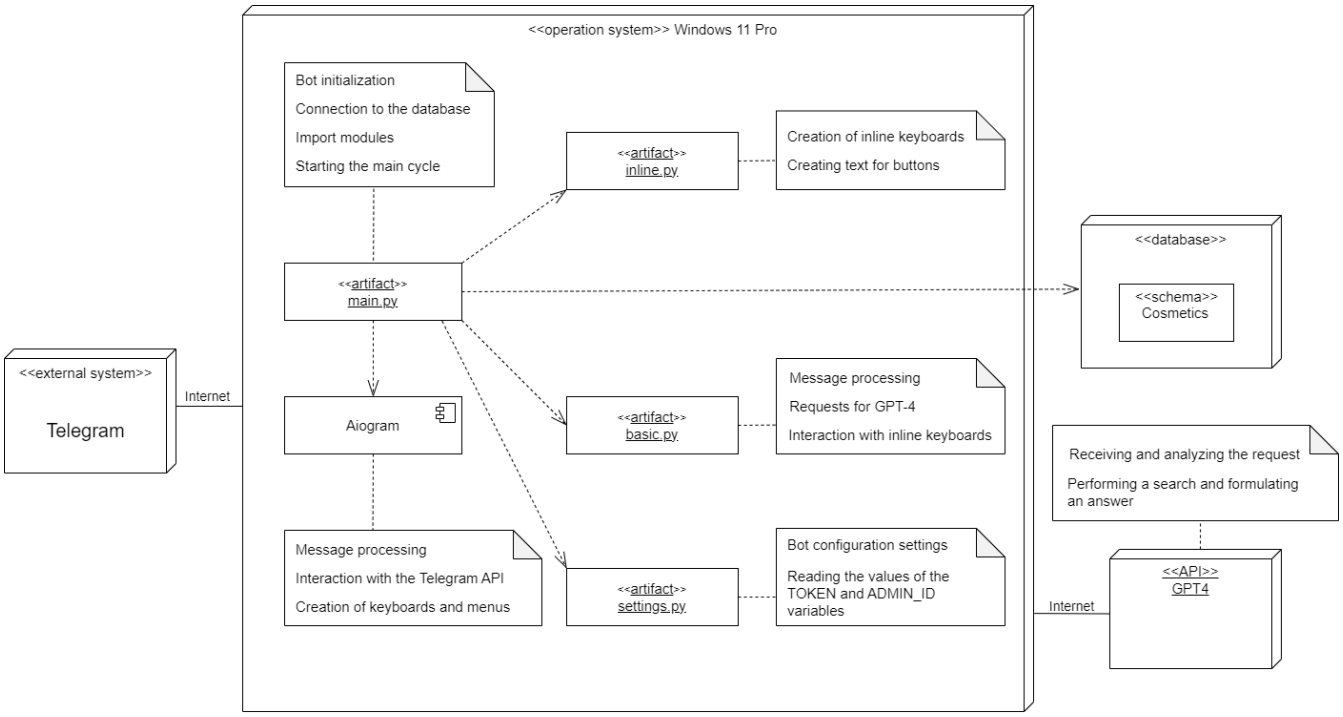


Рис. 3.4 Діаграма розгортання телеграм чат-бота

4 РОЗРОБКА TELEGRAM-БОТА

У цьому розділі описується процес створення Telegram-бота для аналізу складу косметичних засобів та підбору продуктів для певного типу волосся та шкіри. Розділ охоплює всі аспекти розробки бота: від початкового налаштування та створення бота за допомогою сервісу BotFather до реалізації бази даних для зберігання інформації та розробки функціоналу бота.

4.1 Створення бота у BotFather

BotFather - це офіційний бот Telegram, який надає користувачам можливість створення, налаштування та керування їхніми чат-ботами через спеціальний інтерфейс.

За допомогою BotFather користувач може створити нового чат-бота, надати йому унікальне ім'я та назначити його належності, такі як профільні фотографії, опис, команди та інші налаштування. Крім того, BotFather забезпечує доступ до токенів авторизації, які потрібні для взаємодії з API Telegram з боку розробника.

Бот BotFather допомагає розробникам керувати своїми чат-ботами, забезпечуючи зручний інтерфейс для створення та налаштування нових ботів, а також для отримання необхідних налаштувань та інформації про них.

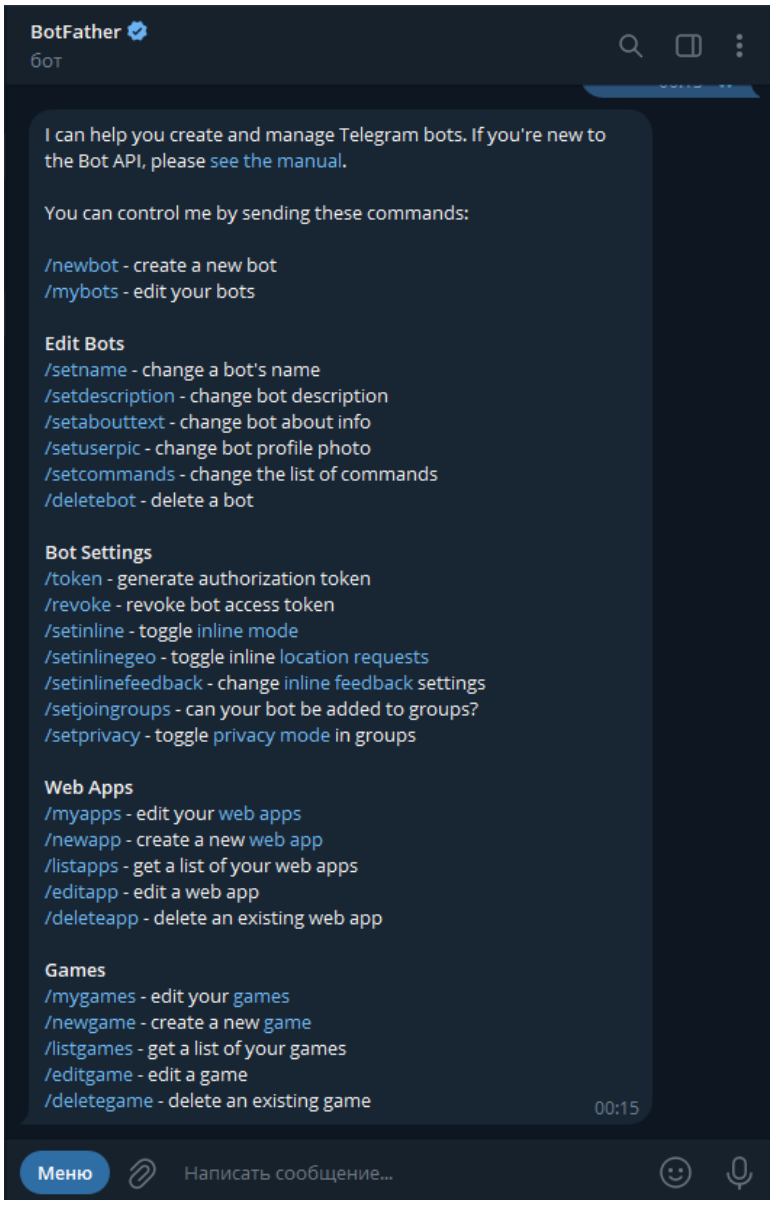


Рис. 4.1 Список команд в BotFather

Створення нового бота у BotFather є першим кроком у процесі розробки чат-бота для платформи Telegram. Для цього слід виконати наступні кроки: Створення нового бота у BotFather є першим кроком у процесі розробки чат-бота для платформи Telegram. Для цього слід виконати наступні кроки:

- Необхідно ввести "BotFather" у пошук або перейти за посиланням, щоб знайти BotFather в Telegram.
- Для того щоб почати взаємодіати з ботом необхідно ввести команду «/start» або вибрати її у меню. Після цього BotFather пропонує обрати ім'я для нового бота.

- Далі BotFather пропонує вибрати коротке ім'я для бота, яке закінчується на "bot", після обрання його імені. Для створення унікального URL доступу до бота буде використовуватися це коротке ім'я.
- Після успішного створення бота BotFather надає користувачу токен доступу у вигляді унікального ключа. Цей токен має зберігатися у безпечному місці, оскільки він є основним ідентифікатором вашого бота у Telegram API.
- Токен доступу використовується у коді вашого бота для налаштування з'єднання з Telegram API та взаємодії з серверами Telegram.

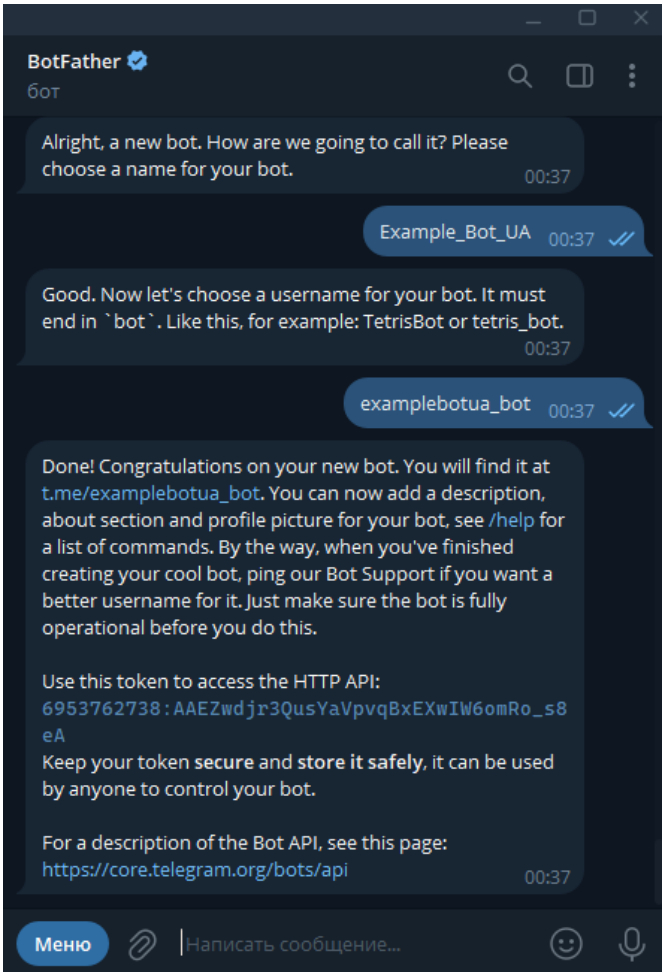


Рис. 4.2 Приклад створення бота та отримання токену

4.2 Реалізація бази даних

Для реалізації поставлених задач до дипломної роботи використовується база даних SQLite. База даних включає такі поля: hair_type, skin_type, product_type, url, photo_url, description.

Поле hair_type зберігає дані про типи волосся: пористі, кучеряві, тонкі.

Поле skin_type зберігає дані про типи шкіри: суха, жирна, комбінована.

Поле product_type зберігає дані про типи продуктів: шампунь, бальзам, маска, сироватка, пінка, тонік, міцелярна вода.

Поле url зберігає посилання на косметичні продукти на сайті makeup.ua.

Поле photo_url зберігає посилання на фото косметичного продукту.

Поле description зберігає опис косметичних продуктів.

Схема бази даних зображена на рис. 4.3.

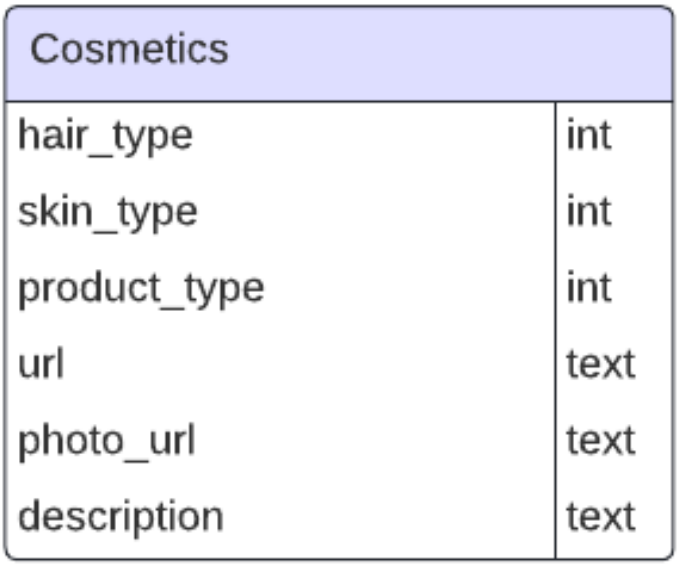
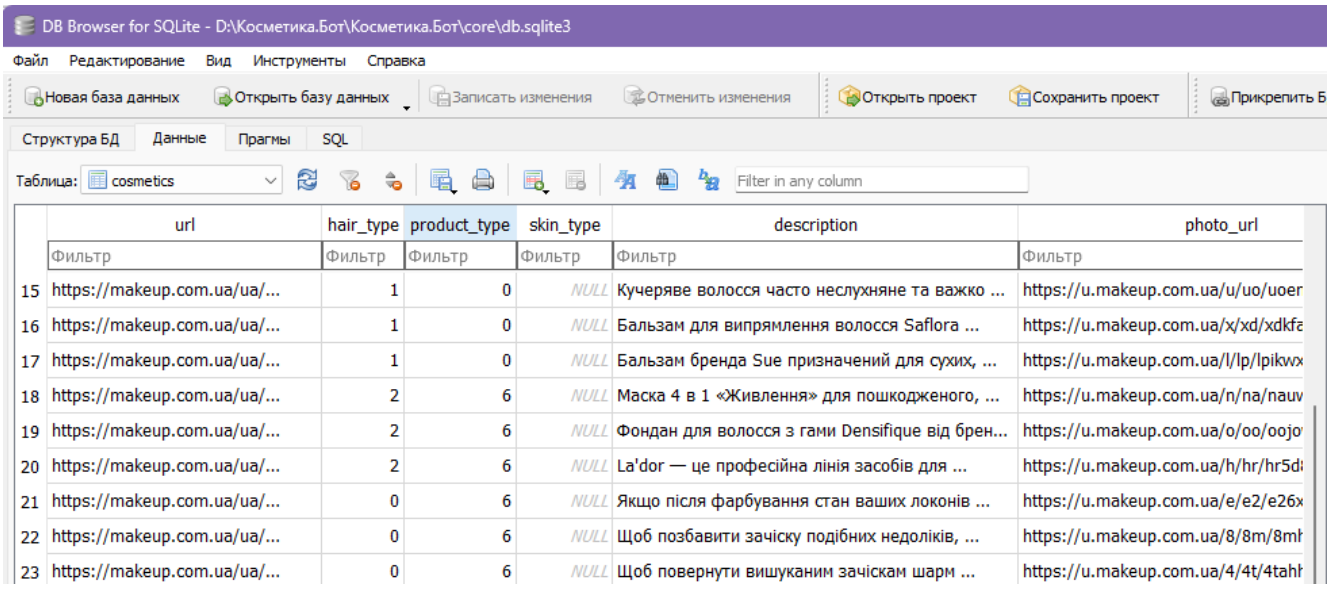


Рис. 4.3 Схема бази даних чат-бота



	url	hair_type	product_type	skin_type	description	photo_url
	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр
15	https://makeup.com.ua/ua/...	1	0	NULL	Кучеряве волосся часто неслухняне та важко ...	https://u.makeup.com.ua/u/uo/uoer
16	https://makeup.com.ua/ua/...	1	0	NULL	Бальзам для випрямлення волосся Saflora ...	https://u.makeup.com.ua/x/xd/xdkfa
17	https://makeup.com.ua/ua/...	1	0	NULL	Бальзам бренда Sue призначений для сухих, ...	https://u.makeup.com.ua/l/lp/lpikwx
18	https://makeup.com.ua/ua/...	2	6	NULL	Маска 4 в 1 «Живлення» для пошкодженого, ...	https://u.makeup.com.ua/n/na/nauv
19	https://makeup.com.ua/ua/...	2	6	NULL	Фондан для волосся з гами Densifique від брен...	https://u.makeup.com.ua/o/oo/oojo
20	https://makeup.com.ua/ua/...	2	6	NULL	La'dor — це професійна лінія засобів для ...	https://u.makeup.com.ua/h/hr/hr5d
21	https://makeup.com.ua/ua/...	0	6	NULL	Якщо після фарбування стан ваших локонів ...	https://u.makeup.com.ua/e/e2/e26x
22	https://makeup.com.ua/ua/...	0	6	NULL	Щоб позбавити зачіску подібних недоліків, ...	https://u.makeup.com.ua/8/8m/8mh
23	https://makeup.com.ua/ua/...	0	6	NULL	Щоб повернути вишуканим зачіскам шарм ...	https://u.makeup.com.ua/4/4t/4tahh

Рис. 4.4 Фрагмент з бази даних для Telegram-бота

```
import sqlite3
dp = Dispatcher()
router = Router()
connection = sqlite3.connect('db.sqlite3')
cursor = connection.cursor()
```

Рис. 4.5 Імпорт бази даних

4.3 Розробка бота у PyCharm

Зпершу необхідно створити функцію, яка зчитує конфігураційні налаштування для Telegram-бота з файлу оточення і повертає їх у структурованому вигляді. Спочатку визначаються дві класи для зберігання налаштувань: один для налаштувань бота, що містить його токен і ідентифікатор адміністратора, а інший для загальних налаштувань, який включає в себе об'єкт з налаштуваннями бота. Функція, яка читає файл оточення, витягує з нього значення токена і ідентифікатора адміністратора, створює об'єкт налаштувань і повертає його. Це дозволяє зручно зберігати і використовувати конфігураційні параметри в програмі, забезпечуючи гнучкість і зручність у керуванні налаштуваннями бота.

```python
@dataclass
class Bots:
    bot_token: str
    admin_id: int


1 usage
@dataclass
class Settings:
    bots: Bots


1 usage
def get_settings(path: str):
    env = Env()
    env.read_env(path)
    return Settings(
        bots=Bots(
            bot_token=env.str('TOKEN'),
            admin_id=env.int('ADMIN_ID')
        )
    )
```

Рис. 4.6 Створення класів для налаштування бота

Необхідно налаштувати чат-бот так, щоб він міг ефективно обробляти взаємодію з користувачами. Він реєструє різноманітні обробники для обробки команд і дій користувачів.

Починаючи з обробки команди /start, коли користувач починає спілкування з ботом, і продовжуючи з callback-запитами, коли користувачі взаємодіють з кнопками або іншими елементами інтерфейсу, бот забезпечує відповідну реакцію на всі взаємодії. Така структура дозволяє боту виконувати різноманітні дії, які можуть бути пов'язані зі стартовим повідомленням для користувача, отриманням інформації, наданням порад або переходом до попередньої сторінки в інтерфейсі.

Включення маршрутизатора допомагає організувати код та обробку повідомлень у боті, що забезпечує його ефективну роботу та легке розширення.

```
async def start():

    bot = Bot(token=settings.bots.bot_token)

    dp.message.register(start_message, Command(commands='start'))
    dp.callback_query.register(communication, F.data == 'communication')
    dp.callback_query.register(bot_info, F.data == 'bot_info')
    dp.callback_query.register(advice, F.data == 'advice')
    dp.callback_query.register(start_message, F.data == 'back')
    dp.callback_query.register(hair_or_face_skin, F.data == 'get_hair_face')
    dp.callback_query.register(get_info, F.data == 'get_info')
    dp.callback_query.register(get_pick_up, F.data == 'get_pick_up')
    dp.include_router(router)
    dp.message.register(get_info_response)
```

Рис. 4.7 Реалізація команди ‘/start’ та callback-запитів

Щоб користувач міг використати меню для взаємодії з ботом необхідно створити клавіатуру з кнопками для чат-бота. Ці кнопки допоможуть користувачам взаємодіяти з ботом та обирати різні опції.

Кожна кнопка має свою функцію:

Перша кнопка, позначена як " Догляд за твоїм типом волосся обличчя", дозволяє користувачам отримати поради щодо догляду за їх волоссям або шкірою обличчя.

Друга кнопка, позначена як " Аналіз складу", спрямована на аналіз складу косметичних засобів, що дозволить користувачам дізнатися більше про безпеку та функції комконентів косметики.

Третя кнопка, позначена як " Зворотній зв'язок", надає можливість користувачам зв'язатися з адміністратором бота, наприклад, для задання питань або надання відгуків.

Четверта кнопка, позначена як " Корисні поради за доглядом", надає користувачам корисні поради щодо догляду за собою.

П'ята кнопка, позначена як " Про бота", містить інформацію про самого бота.

Після побудови клавіатури функція повертає її у вигляді розмітки, яка може бути відправлена користувачеві у чаті. Таким чином, користувачі матимуть

можливість легко і зручно вибирати опції взаємодії з ботом, що сприятиме покращенню їх досвіду користувача.

```
def start_mes():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Догляд за твоїм типом волосся або шкіри обличчя', callback_data='get_hair_face')
    keyboard_builder.button(text='Аналіз складу', callback_data='get_info')
    keyboard_builder.button(text="Зворотній зв'язок", callback_data='communication')
    keyboard_builder.button(text='Корисні поради за доглядом', callback_data='advice')
    keyboard_builder.button(text='Про бота', callback_data="bot_info")
    keyboard_builder.adjust( *sizes: 1, 1, 1, 1)
    return keyboard_builder.as_markup()
```

Рис. 4.8 Реалізація головного меню

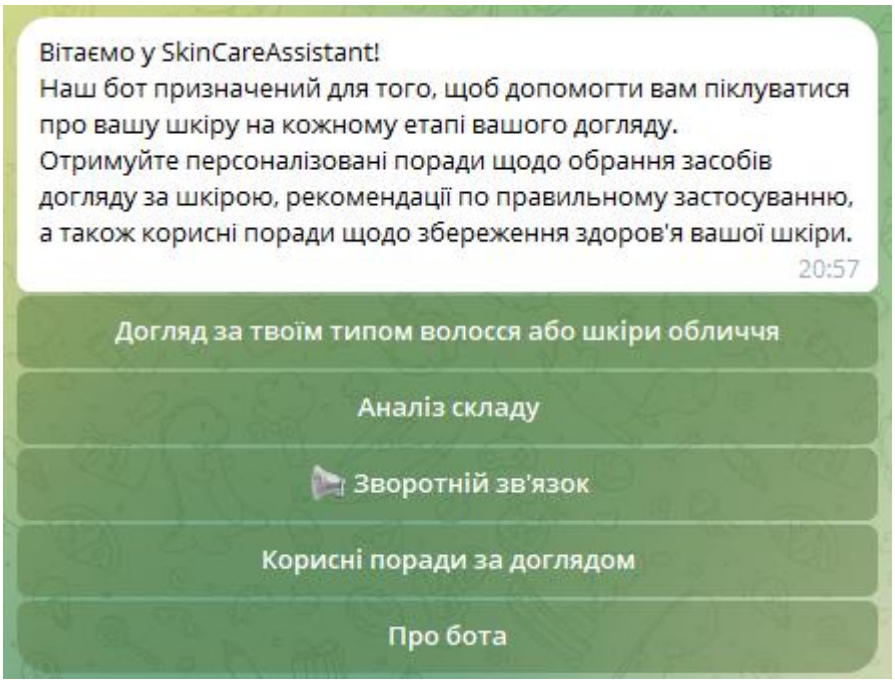


Рис. 4.9 Відображення головного меню у telegram

Створено три функції для створення клавіатур для взаємодії з користувачами чат-бота. Перша функція створює клавіатуру для вибору між доглядом за волоссям і шкірою обличчя, друга - для вибору характеристик волосся, а третя - для вибору конкретних продуктів для догляду за волоссям залежно від обраної характеристики. Кожна клавіатура містить кнопки з відповідними текстовими назвами та значеннями callback_data для ідентифікації вибору користувача. Після

створення клавіатури вона повертається у вигляді розмітки і готова для використання в чат-боті для спрощення взаємодії користувачів з ботом.

```
def hair_or_face():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Волосся', callback_data='hair')
    keyboard_builder.button(text='Шкіра обличчя', callback_data='get_pick_up')
    keyboard_builder.adjust( *sizes: 1, 1)
    return keyboard_builder.as_markup()
```

Рис. 4.10 Реалізація клавіатур для вибору між доглядом за волоссям і шкірою обличчя

```
def create_inline_keyboard():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Пористі", callback_data="dat.0"),
    keyboard_builder.button(text="Кучеряві", callback_data="dat.1"),
    keyboard_builder.button(text="Тонкі", callback_data="dat.2")
    keyboard_builder.adjust( *sizes: 1, 1, 1)
    return keyboard_builder.as_markup()

4 usages
def create_second_inline_keyboard(first_choice):
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Бальзам", callback_data=f"second.{first_choice}.0"),
    keyboard_builder.button(text="Шампунь", callback_data=f"second.{first_choice}.1"),
    keyboard_builder.button(text="Сироватка", callback_data=f"second.{first_choice}.2")
    keyboard_builder.button(text="Маска", callback_data=f"secondskin.{first_choice}.6")
    keyboard_builder.adjust( *sizes: 1, 1, 1)
    return keyboard_builder.as_markup()
```

Рис. 4.11 Реалізація клавіатур для вибору типу волосся та засобу

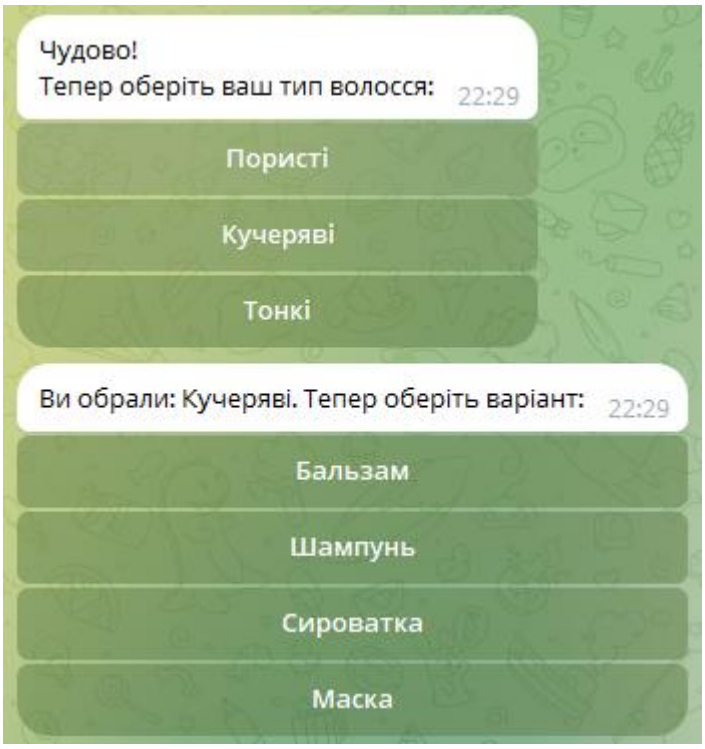


Рис. 4.12 Відображення клавіатур для вибору типу волосся та засобу у telegram

На цьому етапі описані дві функції для створення клавіатур для взаємодії з користувачами чат-бота, спеціально для вибору характеристик шкіри. Перша функція створює клавіатуру для вибору типу шкіри, такого як "Суха", "Жирна" або "Комбінована". Друга функція створює додаткову клавіатуру, яка залежить від вибору користувача з першої клавіатури і містить кнопки для обрання конкретних продуктів для догляду за обраною характеристикою шкіри, таких як "Пінка", "Тонік" або "Міцелярна вода". Кожна клавіатура створюється з використанням об'єкта InlineKeyboardBuilder та містить кнопки з відповідними текстовими назвами та значеннями callback_data. Після створення клавіатури вона готова для використання у чат-боті для відображення користувачеві та полегшення вибору продуктів догляду за шкірою.

```python
def create_inline_keyboard_skin():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Суха", callback_data="skin.0"),
    keyboard_builder.button(text="Жирна", callback_data="skin.1"),
    keyboard_builder.button(text="Комбінована", callback_data="skin.2")
    keyboard_builder.adjust( *sizes: 1, 1, 1)
    return keyboard_builder.as_markup()

# Функция для создания второй inline клавиатуры
2 usages
def create_second_inline_keyboard_skin(first_choice):
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Пінка", callback_data=f"secondskin.{first_choice}.3"),
    keyboard_builder.button(text="Тонік", callback_data=f"secondskin.{first_choice}.4"),
    keyboard_builder.button(text="Міцелярна вода", callback_data=f"secondskin.{first_choice}.5")
    keyboard_builder.adjust( *sizes: 1, 1, 1, 1)
    return keyboard_builder.as_markup()
```

Рис. 4.13 Реалізація клавіатур для вибору типу шкіри та засобу

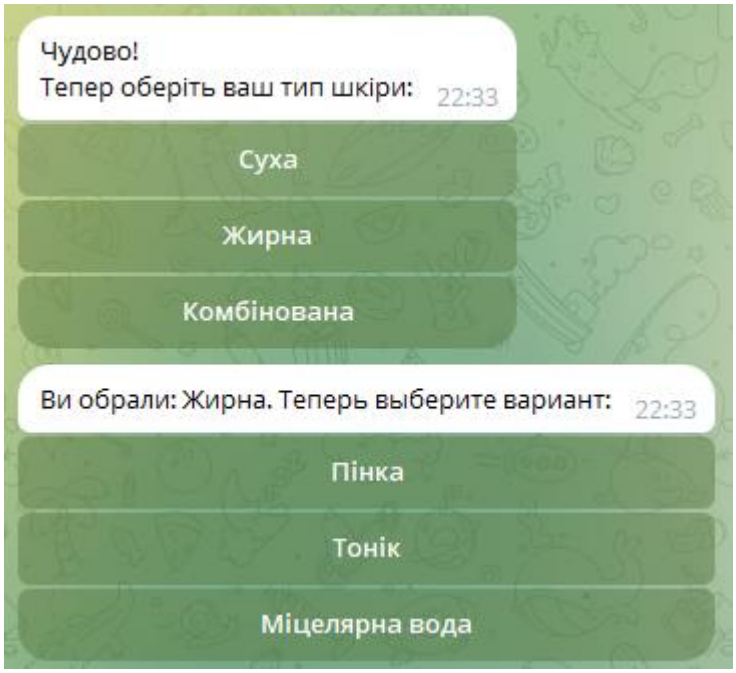


Рис. 4.14 Відображення клавіатур для вибору типу шкіри та засобу у Telegram

Для відповіді на callback-запити користувача створено дві асинхронні функції.

Перша функція обробляє callback-запити, які починаються з префіксу 'dat.'. Під час обробки запиту вона отримує перше значення після крапки, як вибір

користувача, та надсилає повідомлення користувачеві з обраним варіантом і створює другу клавіатуру для вибору конкретних продуктів.

Друга функція обробляє callback-запити, які починаються з префіксу 'second.'. Під час обробки запиту вона розбиває дані з callback-запиту на окремі частини, використовуючи крапку як роздільник. Потім вона виконує запит до бази даних, щоб отримати інформацію про конкретний продукт для догляду за волоссям, використовуючи обрані параметри користувачем. Після цього вона надсилає фотографію продукту та його опис з бази даних користувачеві.

```python
@router.callback_query(F.data.startswith('dat.'))
async def process_callback(callback_query: types, bot: Bot):
    first_choice = callback_query.data.split('.')[1]
    await bot.answer_callback_query(callback_query.id)
    await bot.send_message(callback_query.from_user.id, text: f'Ви обрали: {types_choose_hair[int(first_choice)]}. Тепер оберіть варіант:', reply_markup=create_second_inline_keyboard(first_choice))
```

Рис. 4.15 Реалізація функції для обробки запитів по типу волосся

```python
@dp.callback_query(F.data.startswith('second.'))
async def process_second_callback(callback_query: types.CallbackQuery, bot: Bot):
    _, first_choice, second_choice = callback_query.data.split('.')
    cursor.execute( __sql: 'SELECT url, hair_type, product_type,description, photo_url from cosmetics where hair_type = ? AND product_type = ? ORDER BY RANDOM() LIMIT 1;', __parameters: (first_choice, second_choice))
    url, _, _,description, photo_url = cursor.fetchone()
    await bot.answer_callback_query(callback_query.id)
    await bot.send_photo(callback_query.from_user.id, caption=description, photo=photo_url, reply_markup=get_care(url))
```

Рис. 4.16 Реалізація функції для обробки та виконання запитів до бази даних

Рис. 4.17 Приклад отримання інформації про сироватку для тонкого волосся

На цьому етапі описана асинхронна функція get_info_response, яка відповідає на повідомлення користувача, пов'язане з отриманням інформації про компоненти косметики.

При отриманні повідомлення функція надсилає користувачеві повідомлення про те, що триває обробка даних і це може зайняти деякий час. Потім вона використовує бібліотеку GPT-4 для створення відповіді на запит користувача. Функція передає текстове повідомлення користувача як вхідні дані для моделі GPT-4 та очікує згенеровану відповідь.

Отримавши відповідь від моделі, функція надсилає її користувачеві у вигляді повідомлення. Таким чином, користувач отримує відповідь на свій запит щодо користі та шкоди компонентів косметики, а також рівень безпеки кожного з них.

```
async def get_info_response(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id,  text: 'Триває обробка данних, це може зайняти декілька секунд...')
    response = g4f.ChatCompletion.create(model=g4f.models.gpt_4, messages=[
        {"role": "user", "content": f"Яка користь та шкода таких компонентів косметики, вказати рівень безпеки кожного: {str(message.text)}"}])
    await bot.send_message(message.from_user.id, response, reply_markup=back())
```

Рис. 4.18 Реалізація функції для створення відповіді на запит користувача

Рис. 4.19 Приклад розшифрування інгредієнтів у Telegram

На рисунку 4.20 зображено вітальне повідомлення, яке Telegram-бот надсилає користувачу, коли той вирішує залишити відгук. У повідомленні подяка за бажання залишити відгук, прохання написати адміністратору і описати свої враження, зауваження або пропозиції. Також підкреслюється важливість відгуків, оскільки вони допомагають покращувати сервіс і робити його зручнішим для користувачів. Нижче розташована кнопка "Написати", при натисканні на яку користувача перенаправляють на чат з адміністратором чат-бота.

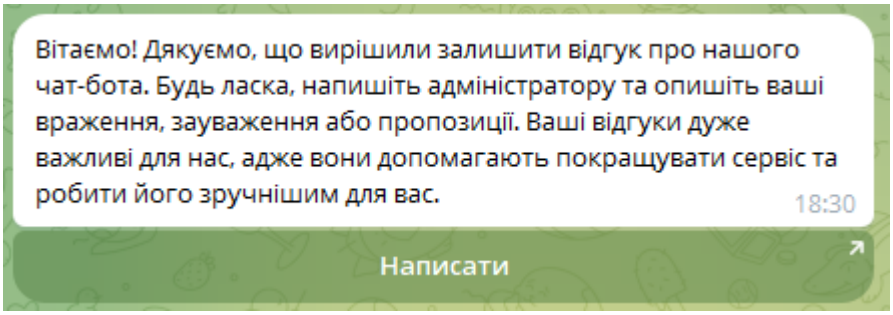


Рис. 4.20 Повідомлення з пропозицією залишити відгук та кнопкою переходу до чату з адміністратором

На рисунку 4.21 представлено повідомлення Telegram-бота з переліком корисних порад щодо догляду за шкірою, яке містить рекомендації з тестування продуктів, їх застосування в невеликих кількостях, дотримання правильного порядку використання, методів нанесення, уникнення надмірного використання та регулярного застосування для досягнення найкращих результатів. Внизу повідомлення розміщена кнопка "Назад" для повернення до головного меню.

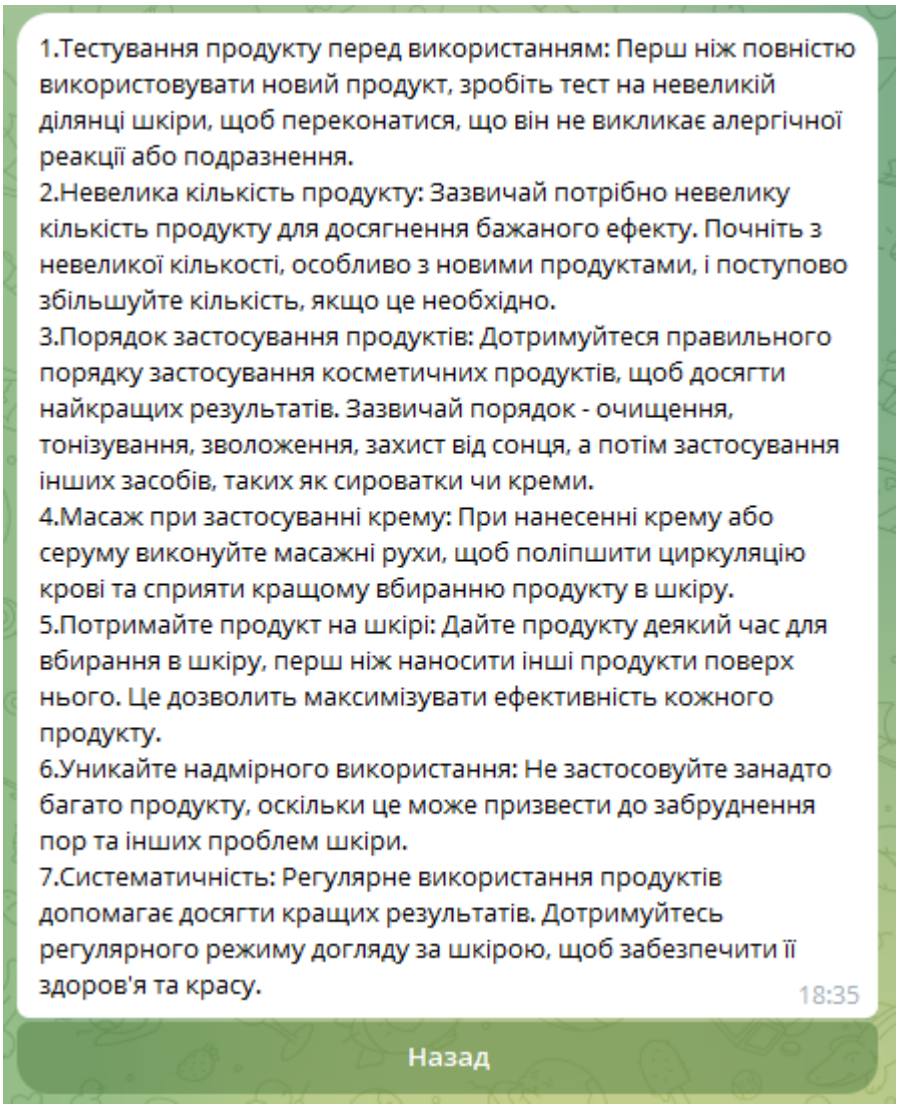


Рис. 4.21 Перелік порад щодо догляду за шкірою або волоссям

Вітальне повідомлення Telegram-бота, яке представляє бота як надійного помічника у виборі косметики представлено на рисунку 4.22. У даному повідомленні бот пропонує допомогти користувачу знайти найкращі продукти для конкретного типу шкіри та уподобань користувача, надаючи поради щодо найефективніших засобів для догляду за обличчям, волоссям та тілом. Також бот запрошує користувача задавати питання про рекомендації, поради з використання та інші питання, пов'язані з косметикою. Внизу повідомлення розміщена кнопка "Назад" для повернення до головного меню.

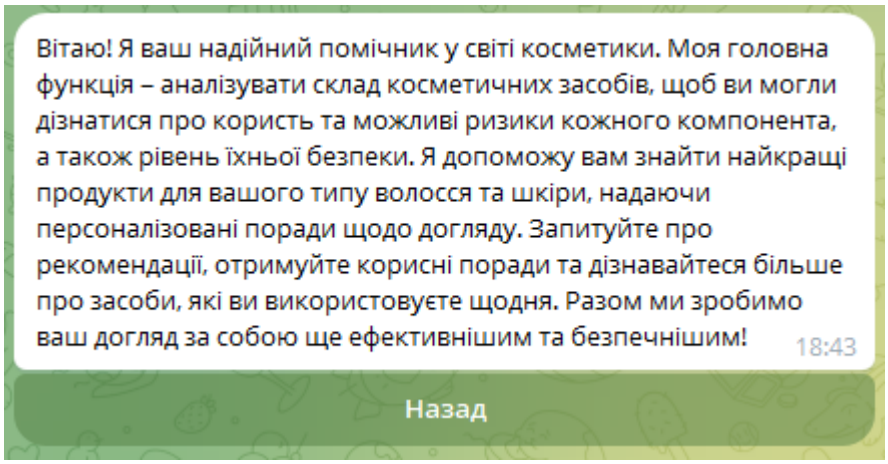


Рис. 4.22 Повідомлення з описом бота

4.4 Тестування

Мануальне тестування є важливою складовою процесу розробки програмного забезпечення. Воно дозволяє вручну перевірити функціональність програми та виявити потенційні проблеми, які можуть залишитися невиявленими при автоматизованому тестуванні. Мануальне тестування Telegram-бота включає наступні кроки:

Протестовано введення команди /start у чат з ботом і виявлено, що бот відповідає відповідним повідомленням і відображає меню з кнопками. Перевірено, чи кнопки меню відповідають очікуваному функціоналу та чи відбувається перехід на відповідні функції при їх виборі.

Протестовано введення некоректних даних після вибору опції "Аналіз складу". В результаті бот відправляє відповідне повідомлення про помилку та інформує користувача про правильний формат введення даних.

Протестовано натискання на різні кнопки меню. Бот правильно реагує на вибір кожної кнопки, відображаючи відповідні повідомлення або виконуючи відповідні дії. Перевірено, чи коректно працює перехід між різними функціями бота та чи не виникає помилок під час їх виконання.

Таблиця 4.1

Результати мануального тестування

Пункт меню, який тестувався	Протестована функція	Очікуваний результат	Результат тесту
Запуск бота	Ініціація чату	Після введення команди /start у чат з ботом, бот відправляє меню з інтерактивними кнопками	Успіх
Пункт "Догляд за твоїм типом волосся або шкіри обличчя"	Відображення меню вибору «Волосся/Шкіра»	Після натискання на кнопку відображається меню з опціями "Волосся" та "Шкіра обличчя".	Успіх
	Відображення меню вибору типу волосся	Після натискання на кнопку відображається меню з опціями "Пористі", "Кучеряві", "Тонкі"	Успіх
	Відображення меню вибору типу шкіри	Після натискання на кнопку відображається меню з опціями "Суха", "Жирна", "Комбінована"	Успіх
	Підтвердження вибору типу волосся/шкіри	Після натискання на кнопку бот виводить повідомлення: "Ви обрали: [тип волосся/шкіри]. Тепер оберіть варіант:".	Успіх
	Відображення меню вибору типу засобу для волосся	Після натискання на кнопку відображається меню з опціями "Бальзам", "Шампунь", "Сироватка", "Маска"	Успіх
	Відображення меню вибору типу засобу для обличчя	Після натискання на кнопку відображається меню з опціями "Пінка", "Тонік", "Міцелярна вода"	Успіх

Продовження таблиці 4.1

Результати мануального тестування

Пункт меню, який тестувався	Протестована функція	Очікуваний результат	Результат тесту
Пункт "Догляд за твоїм типом волосся або шкіри обличчя"	Відображення фото, опису та посилання на косметичний продукт	Після натискання на кнопку відображається фото, опис та посилання на косметичний продукт	Успіх
Пункт "Аналіз складу косметичних засобів"	Запуск аналізу складу	Після натискання на кнопку бот виводить повідомлення: "Чудово! Тепер введіть склад". Після введення складу користувачем бот відправляє повідомлення про обробку даних і виводить опис кожного компонента.	Успіх
	Обробка некоректних даних	Після введення даних, які не стосуються компонентів косметичних засобів, бот виводить повідомлення "Будь-ласка, введіть назви компонентів косметичних засобів, щоб я зміг провести аналіз."	Успіх
Пункт "Зворотній зв’язок"	Ініціація зворотного зв'язку	Після натискання на кнопку бот виводить повідомлення і кнопку, що перенаправляє користувача на чат з адміністратором.	Успіх

Продовження таблиці 4.1

Результати мануального тестування

Пункт меню, який тестувався	Протестована функція	Очікуваний результат	Результат тесту
Пункт "Корисні поради по догляду"	Відображення повідомлення з рекомендаціями	Після натискання на кнопку бот виводить повідомлення з рекомендаціями по догляду за волоссям або обличчям.	Успіх
Пункт "Про бота"	Відображення повідомлення описом бота	Після натискання на кнопку бот виводить повідомлення з описом можливостей бота.	Успіх

ВИСНОВКИ

У дипломній роботі було розроблено телеграм-бот для аналізу складу косметичних засобів. Актуальність цієї теми зумовлена зростаючою увагою суспільства до питань здоров’я, безпеки та ефективності косметичних продуктів. Сучасні споживачі все частіше прагнуть дізнатися більше про компоненти, що входять до складу їхніх улюблених косметичних засобів, аби зробити усвідомлений вибір і уникнути потенційно шкідливих речовин.

Розроблений бот відповідає на цю потребу, надаючи користувачам можливість швидко та зручно отримати інформацію про користь і шкоду кожного компонента косметики, а також оцінку їхньої безпеки. Це особливо важливо в умовах ринку, перенасиченого різноманітними продуктами з неоднозначними інгредієнтами.

Крім того, телеграм-бот є доступним інструментом, який можна використовувати в будь-який час і з будь-якого місця, що робить його зручним для широкої аудиторії. Використання технології GPT-4 для аналізу складу косметичних засобів забезпечує високу точність і наукову обґрунтованість наданої інформації, що ще більше підвищує корисність та надійність цього інструменту.

1. Проведено аналіз предметної області та існуючих сервісів для аналізу складу косметичних засобів, а саме: Safety Makeup, SkinCarisma, Inci Decoder, Cosmosweet, CosDNA. На основі аналізу було визначено головні переваги та недоліки сервісів.
2. Визначено функціональні та нефункціональні вимоги програмного забезпечення.
3. Визначено послідовність етапів для проведення аналізу складу косметичних засобів. Для проведення даної операції було використано GPT4 API.
4. Визначено послідовність етапів для надання пропозицій по догляду за певним типом шкіри та типом волосся. Для проведення даної операції було створено базу даних, яка зберігає інформацію про продукти, створено запити до неї.

5. Ресурсом для отримання інформації щодо косметичних продуктів було обрано веб-сервіс makeup.com.ua. По даним сайту similarweb.com він займає друге місце у рейтингу категорії «Краса та косметика» в Україні, загальна кількість візитів – 7.7 мільйонів.
6. Розроблено чат-бот, який забезпечує інформацію щодо безпеки та функціоналу інгредієнтів косметичних засобів, надає пропозиції щодо засобів, які підходять до конкретного типу шкіри або волосся, дає можливість залишити відгук для покращення роботи застосунку. Для реалізації проєкту було використана мова програмування Python і такі інструменти як PyCharm, GPT4, Aiogram, Assyncio, SQLite.
7. Проведено мануальне тестування Telegram-бота. Протестовано введення команди «/start», введення некоректних даних, натискання на різні кнопки меню. Усі виявлені випадки некоректної поведінки бота були усунуті. В результаті, бот реагує на всі команди відповідно до очікуваних результатів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Telegram bot API [Електронний ресурс] – Режим доступу до ресурсу : https://core.telegram.org/bots/api
2. The CIR Program [Електронний ресурс] – Режим доступу до ресурсу : https://www.researchgate.net/publication/320380535
3. Epravda [Електронний ресурс] – Режим доступу до ресурсу : https://www.epravda.com.ua/publications/2023/09/26/704732/
4. Mariefreshcosmetics [Електронний ресурс] – Режим доступу до ресурсу : https://mariefreshcosmetics.com/posts/nebezpechni-komponenti-v-kosmetici
5. Makeup Ua [Електронний ресурс] – Режим доступу до ресурсу : https://makeup.com.ua/ua/
6. Python documentation [Електронний ресурс] – Режим доступу до ресурсу : https://docs.python.org/3/
7. GPT 4 free [Електронний ресурс] – Режим доступу до ресурсу : https://pypi.org/project/g4f/
8. Asyncio [Електронний ресурс] – Режим доступу до ресурсу : https://pypi.org/project/asyncio/
9. PyCharm documentation [Електронний ресурс] – Режим доступу до ресурсу : https://www.jetbrains.com/help/pycharm/getting-started.html
10. SQLite Documentation [Електронний ресурс] – Режим доступу до ресурсу : https://www.sqlite.org/docs.html
11. Aiogram [Електронний ресурс] – Режим доступу до ресурсу : https://docs.aiogram.dev/en/latest/

ДОДАТОК А
ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА TELEGRAM-БОТУ ДЛЯ АНАЛІЗУ СКЛАДУ КОСМЕТИЧНИХ ЗАСОБІВ

Виконала студентка 4 курсу

групи ПД-41
Тищенко Анна Володимирівна
Керівник роботи

Ст. викладач кафедри ІПЗ Гаманюк Ігор Михайлович

Київ – 2024

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи -** забезпечення зручності отримання інформації та аналізу складу косметичних засобів за допомогою Telegram-боту.
- **Об'єкт дослідження -** процес аналізу складу косметичних засобів.
- **Предмет дослідження –** Telegram бот для аналізу складу косметичних засобів.

2

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Провести аналіз предметної області.
2. Провести аналіз існуючих сервісів для аналізу складу косметичних засобів, визначити актуальність Telegram-ботів.
3. Визначити функціональні та нефункціональні вимоги програмного забезпечення.
4. Визначити послідовність етапів для проведення аналізу складу косметичних засобів та надання пропозицій по догляду за певним типом шкіри та типом волосся.
5. Спроектувати діаграми варіантів використання, діяльності, послідовності, розгортання.
6. Обрати ресурс для отримання інформації щодо косметичних продуктів.
7. Розробити телеграм-бота відповідно до вимог та використання необхідних технологій для реалізації функціональності програми.
8. Провести мануальне тестування Telegram-бота.

3

АНАЛІЗ АНАЛОГІВ

Сервіс / Функція	Safety Makeup	SkinCarisma	Cosmosweet	CosDNA	SkinCare Assistant
Пропозиції продуктів, згідно до особливостей шкіри та волосся	-	+	-	+	+
Інтеграція з Telegram	-	-	-	-	+
Оцінка безпеки інгредієнтів	+	+	+	+	+
Українська локалізація	+	-	+	-	+
Можливість відправлення відгуків/ форум	-	+	-	+	+

4

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги:

1. Забезпечення інформації щодо безпеки використання та функціоналу інгредієнтів косметичних засобів.
2. Надання пропозицій щодо засобів, які підходять до конкретного типу шкіри та волосся.
3. Обробка непередбачених сценаріїв при введенні користувачем некоректних запитів.
4. Можливість залишити відгук для покращення роботи застосунку.

Нефункціональні вимоги:

1. Отримання інформації протягом 1-2 хвилин.
2. Інтеграція з Telegram.
3. Спілкування чат-бота з користувачем у неформальному стилі.

5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

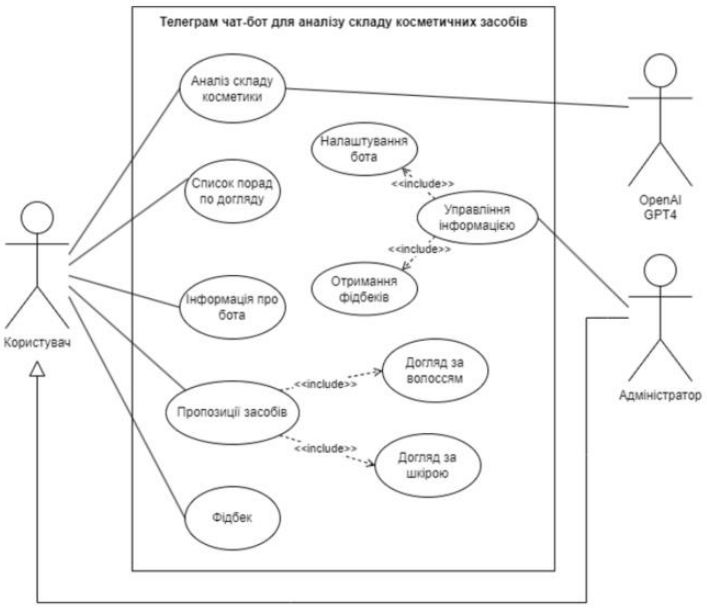
GPT4

Python

PyCharm

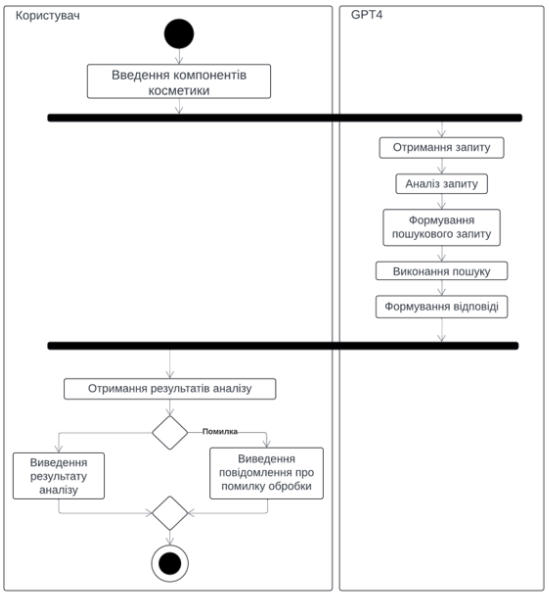
6

Діаграма варіантів використання



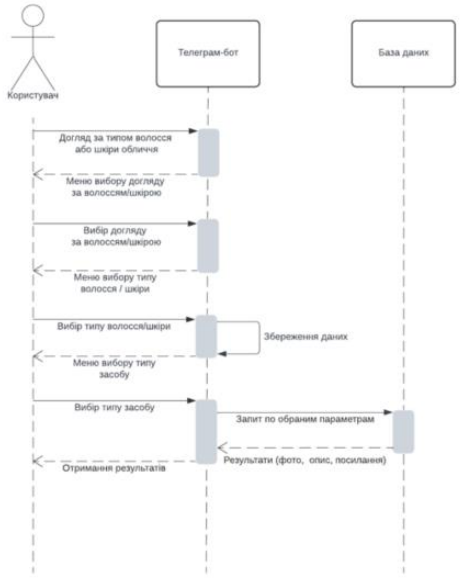
7

Діаграма діяльності процесу аналізу складу косметичного продукту



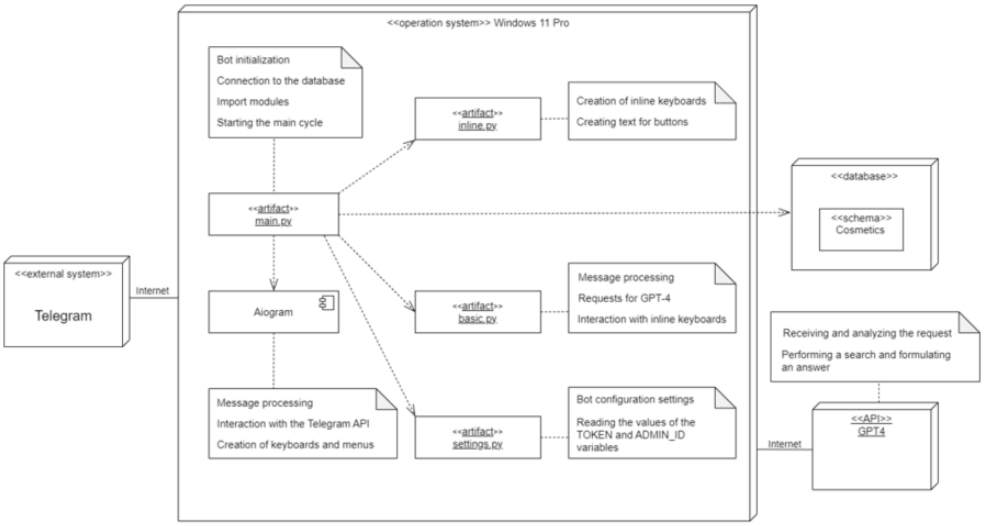
8

Діаграма послідовності процесу надання пропозицій засобів по типу шкіри або волосся



9

Діаграма розгортання



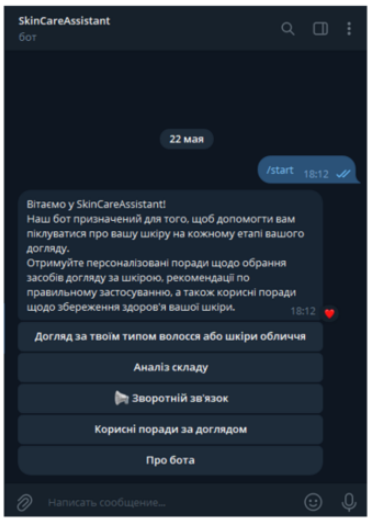
10

Схема бази даних

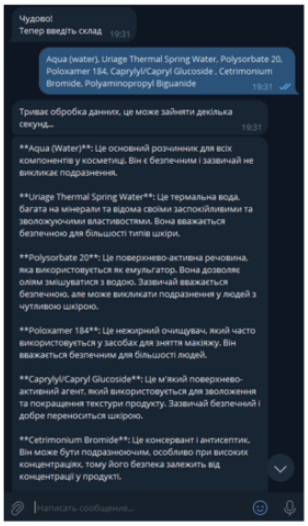
Cosmetics	
hair_type	int
skin_type	int
product_type	int
url	text
photo_url	text
description	text

11

ЕКРАННІ ФОРМИ



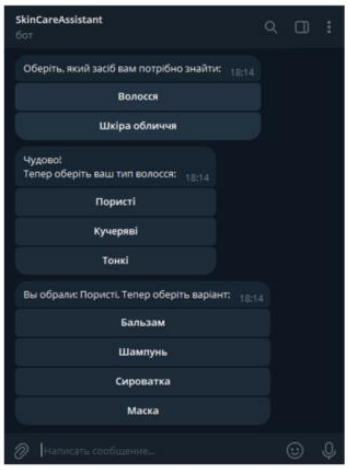
Головне меню чат-бота



Аналіз складових

12

ЕКРАННІ ФОРМИ

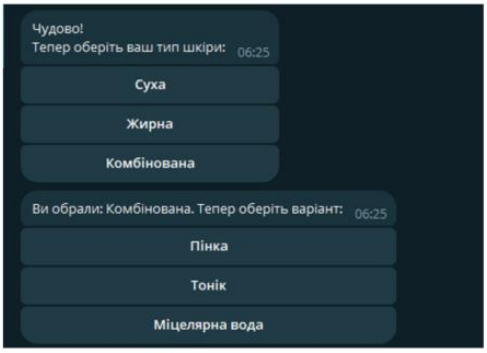


Вибір типу волосся

Приклад запропонованого засобу

13

ЕКРАННІ ФОРМИ



Вибір типу шкіри

Приклад запропонованого засобу

14

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Тищенко А.В., Гаманюк І.М. Визначення вимог до телеграм-бота для аналізу складу косметичних засобів: Матеріали Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в ІКТ». Збірник тез. 24.04.2024, ДУІКТ, м. Київ. К.: ДУІКТ, 204 с.
2. Тищенко А.В., Гаманюк І.М. Аналіз ресурсів, що надають інформацію про склад та безпеку косметичних засобів для створення телеграм-бота: Матеріали Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в ІКТ». Збірник тез. 24.04.2024, ДУІКТ, м. Київ. К.: ДУІКТ, 225 с.

15

ВИСНОВКИ

1. Проведено аналіз предметної області та існуючих сервісів для аналізу складу косметичних засобів, а саме: Safety Makeup, SkinCarisma, Inci Decoder, Cosmosweet, CosDNA. На основі аналізу було визначено головні переваги та недоліки сервісів.
2. Визначено функціональні та нефункціональні вимоги програмного забезпечення.
3. Визначено послідовність етапів для проведення аналізу складу косметичних засобів. Для проведення даної операції було використано GPT4 API.
4. Визначено послідовність етапів для надання пропозицій по догляду за певним типом шкіри та типом волосся. Для проведення даної операції було створено базу даних, яка зберігає інформацію про продукти, створено запити до неї.
5. Ресурсом для отримання інформації щодо косметичних продуктів було обрано веб-сервіс makeup.com.ua. По даним сайту similarweb.com він займає друге місце у рейтингу категорії «Краса та косметика» в Україні, загальна кількість візитів – 7.7 мільйонів.
6. За допомогою діаграм спроектувано варіанти використання, діяльність, взаємодії об’єктів, розгортання застосунку.
7. Розроблено чат-бот, який забезпечує інформацію щодо безпеки та функціоналу інгредієнтів косметичних засобів, надає пропозиції щодо засобів, які підходять до конкретного типу шкіри або волосся, дає можливість залишити відгук для покращення роботи застосунку. Для реалізації проєкту було використана мова програмування Python і такі інструменти як PyCharm, GPT4, Aiogram, Assyncio, SQLite.
8. Проведено мануальне тестування Telegram-боту. Протестовано введення команди «/start», введення некоректних даних, натискання на різні кнопки меню. Усі виявлені випадки некоректної поведінки бота були усунуті. В результаті, бот реагує на всі команди відповідно до очікуваних результатів.

16

ДОДАТОК Б

ОСНОВНІ БЛОКИ КОДУ

Main.py

```
from aiogram import Bot, Dispatcher, F, Router, types
from core.handlers.basic import start_message, get_info, get_info_response, get_pick_up, hair_or_face_skin, bot_info, advice, communication, hair_skin
from core.middlewares.settings import settings
from aiogram.filters import Command
from core.keyboards.inline import create_second_inline_keyboard, create_second_inline_keyboard_skin, get_care
import asyncio
import sqlite3
dp = Dispatcher()
router = Router()
connection = sqlite3.connect('db.sqlite3')
cursor = connection.cursor()
types_choose_skin ={0: "Суха", 1: "Жирна", 2: "Комбінована"}
types_choose_hair ={0: "Пористі", 1: "Кучеряві", 2: "Тонкі"}

async def start():

    bot = Bot(token=settings.bots.bot_token)

    dp.message.register(start_message, Command(commands='start'))
    dp.callback_query.register(hair_skin, F.data == 'hair')
    dp.callback_query.register(communication, F.data == 'communication')
    dp.callback_query.register(bot_info, F.data == 'bot_info')
    dp.callback_query.register(advice, F.data == 'advice')
    dp.callback_query.register(start_message, F.data == 'back')
    dp.callback_query.register(hair_or_face_skin, F.data == 'get_hair_face')
    dp.callback_query.register(get_info, F.data == 'get_info')
    dp.callback_query.register(get_pick_up, F.data == 'get_pick_up')
    dp.include_router(router)
    dp.message.register(get_info_response)

    try:
        await dp.start_polling(bot)
    finally:
        await bot.session.close()


@router.callback_query(F.data.startswith('dat.'))
async def process_callback(callback_query: types, bot: Bot):
    first_choice = callback_query.data.split('.')[1]
    await bot.answer_callback_query(callback_query.id)
    await bot.send_message(callback_query.from_user.id, f'Ви обрали: {types_choose_hair[int(first_choice)]}. Тепер оберіть варіант:', reply_markup=create_second_inline_keyboard(first_choice))

@dp.callback_query(F.data.startswith('second.'))
async def process_second_callback(callback_query: types.CallbackQuery, bot: Bot):
    _, first_choice, second_choice = callback_query.data.split('.')
    cursor.execute('SELECT url, hair_type, product_type,description, photo_url from cosmetics where hair_type = ? AND product_type = ? ORDER BY RANDOM() LIMIT 1;', (first_choice, second_choice))
    url, _, _,description, photo_url = cursor.fetchone()
```

```
    await bot.answer_callback_query(callback_query.id)
    await bot.send_photo(callback_query.from_user.id, caption=description, photo=photo_url,
reply_markup=get_care(url))
    #await bot.send_message(callback_query.from_user.id, f'Ваш окончательный выбор:
{types_choose[int(first_choice)]}.{second_choice}')

@router.callback_query(F.data.startswith('skin.'))
async def process_callback_skin(callback_query: types, bot: Bot):
    first_choice = callback_query.data.split('.')[1]
    await bot.answer_callback_query(callback_query.id)
    await bot.send_message(callback_query.from_user.id, f'Ви обрали: {types_choose_skin[int(first_choice)]}. Теперь
оберіть варіант:', reply_markup=create_second_inline_keyboard_skin(first_choice))

# Обработчик для callback_query второй клавиатуры
@dp.callback_query(F.data.startswith('secondskin.'))
async def process_second_callback_skin(callback_query: types.CallbackQuery, bot: Bot):
    _, first_choice, second_choice = callback_query.data.split('.')

    cursor.execute('SELECT url, skin_type, product_type,description, photo_url from cosmetics where skin_type = ? AND
product_type = ? ORDER BY RANDOM() LIMIT 1;', (first_choice, second_choice))
    url, _, _, description, photo_url = cursor.fetchone()
    await bot.answer_callback_query(callback_query.id)
    await bot.send_photo(callback_query.from_user.id, caption=description, photo=photo_url,
reply_markup=get_care(url))
    #await bot.send_message(callback_query.from_user.id, f'Ваш окончательный выбор:
{types_choose[int(first_choice)]}.{second_choice}')


if __name__ == '__main__':
    asyncio.run(start())
```

Bot send message

```
from aiogram import Bot
from aiogram.types import Message
from core.keyboards.inline import start_mes, back, get_pick, hair_or_face, get_admin_url, create_inline_keyboard,
create_second_inline_keyboard, get_keyboard, create_inline_keyboard_skin

import g4f


async def communication(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, 'Вітаємо! Дякуємо, що вирішили залишити відгук про нашого чат-
бота. Будь ласка, напишіть адміністратору та опишіть ваші враження, зауваження або пропозиції. Ваші відгуки дуже
важливі для нас, адже вони допомагають покращувати сервіс та робити його зручнішим для вас.',
reply_markup=get_admin_url())


async def bot_info(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Вітаю, я ваш надійний помічник у виборі косметики. Я допоможу
вам знайти найкращі продукти для вашого типу шкіри та уподобань, пораджу найефективніші засоби для догляду
за обличчям, волоссям та тілом. Запитуйте про рекомендації, поради з використання і будь-які інші питання,
пов'язані з косметикою!", reply_markup=get_keyboard())

async def advice(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "1.Тестування продукту перед використанням: Перш ніж повністю
```

```
використовувати новий продукт, зробіть тест на невеликій ділянці шкіри, щоб переконатися, що він не викликає алергічної реакції або подразнення.\n2.Невелика кількість продукту: Зазвичай потрібно невелику кількість продукту для досягнення бажаного ефекту. Почніть з невеликої кількості, особливо з новими продуктами, і поступово збільшуйте кількість, якщо це необхідно.\n3.Порядок застосування продуктів: Дотримуйтеся правильного порядку застосування косметичних продуктів, щоб досягти найкращих результатів. Зазвичай порядок - очищення, тонізування, зволоження, захист від сонця, а потім застосування інших засобів, таких як сироватки чи креми.\n4.Масаж при застосуванні крему: При нанесенні крему або серуму виконуйте масажні рухи, щоб поліпшити циркуляцію крові та сприяти кращому вбиранню продукту в шкіру.\n5.Потримайте продукт на шкірі: Дайте продукту деякий час для вбирання в шкіру, перш ніж наносити інші продукти поверх нього. Це дозволить максимізувати ефективність кожного продукту.\n6.Уникайте надмірного використання: Не застосовуйте занадто багато продукту, оскільки це може призвести до забруднення пор та інших проблем шкіри.\n7.Систематичність: Регулярне використання продуктів допомагає досягти кращих результатів. Дотримуйтесь регулярного режиму догляду за шкірою, щоб забезпечити її здоров'я та красу.", reply_markup=get_keyboard())
async def start_message(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Вітаємо у SkinCareAssistant!\nНаш бот призначений для того, щоб допомогти вам піклуватися про вашу шкіру на кожному етапі вашого догляду. \nОтримуйте персоналізовані поради щодо обрання засобів догляду за шкірою, рекомендації по правильному застосуванню, а також корисні поради щодо збереження здоров'я вашої шкіри.", reply_markup=start_mes())


async def get_pick_up(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Чудово! \nТепер оберіть ваш тип шкіри:",
reply_markup=create_inline_keyboard_skin())


async def hair_or_face_skin(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Оберіть, який засіб вам потрібно знайти:",
reply_markup=hair_or_face())

async def hair_skin(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Чудово! \nТепер оберіть ваш тип волосся:",
reply_markup=create_inline_keyboard())

async def hair_type_skin(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Оберіть, який засіб вам потрібно знайти:",
reply_markup=create_second_inline_keyboard())

async def get_info(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, "Чудово!\nТепер введіть склад")


async def get_info_response(message: Message, bot: Bot):
    await bot.send_message(message.from_user.id, 'Триває обробка данних, це може зайняти декілька секунд...')
    response = g4f.ChatCompletion.create(model=g4f.models.gpt_4, messages=[
        {"role": "user", "content": f"Яка користь та шкода таких компонентів косметики, вказати рівень безпеки кожного: {str(message.text)}"}])
    await bot.send_message(message.from_user.id, response, reply_markup=back())
```

Клавіатури

```
from aiogram.utils.keyboard import InlineKeyboardBuilder

def start_mes():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Догляд за твоїм типом волосся або шкіри обличчя', callback_data='get_hair_face')
    keyboard_builder.button(text='Аналіз складу', callback_data='get_info')
    keyboard_builder.button(text="Зворотній зв'язок", callback_data='communication')
    keyboard_builder.button(text='Корисні поради за доглядом', callback_data='advice')
    keyboard_builder.button(text='Про бота', callback_data="bot_info")
```

```
    keyboard_builder.adjust(1, 1, 1, 1)
    return keyboard_builder.as_markup()


def hair_or_face():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Волосся', callback_data='hair')
    keyboard_builder.button(text='Шкіра обличчя', callback_data='get_pick_up')
    keyboard_builder.adjust(1, 1)
    return keyboard_builder.as_markup()


def create_inline_keyboard():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Пористі", callback_data="dat.0"),
    keyboard_builder.button(text="Кучеряві", callback_data="dat.1"),
    keyboard_builder.button(text="Тонкі", callback_data="dat.2")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()

def create_second_inline_keyboard(first_choice):
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Бальзам", callback_data=f"second.{first_choice}.0"),
    keyboard_builder.button(text="Шампунь", callback_data=f"second.{first_choice}.1"),
    keyboard_builder.button(text="Сироватка", callback_data=f"second.{first_choice}.2")
    keyboard_builder.button(text="Маска", callback_data=f"secondskin.{first_choice}.6")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()

def create_inline_keyboard_skin():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Суха", callback_data="skin.0"),
    keyboard_builder.button(text="Жирна", callback_data="skin.1"),
    keyboard_builder.button(text="Комбінована", callback_data="skin.2")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()

# Функция для создания второй inline клавиатуры
def create_second_inline_keyboard_skin(first_choice):
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text="Пінка", callback_data=f"secondskin.{first_choice}.3"),
    keyboard_builder.button(text="Тонік", callback_data=f"secondskin.{first_choice}.4"),
    keyboard_builder.button(text="Міцелярна вода", callback_data=f"secondskin.{first_choice}.5")
    keyboard_builder.adjust(1, 1, 1, 1)
    return keyboard_builder.as_markup()
def back():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Назад', callback_data='back')
    return keyboard_builder.as_markup()


def get_pick():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Суха', callback_data='dry')
    keyboard_builder.button(text='Жирна', callback_data='fat')
    keyboard_builder.button(text='Комбінована', callback_data='combo')
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()
```

```
def get_keyboard():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Назад', callback_data='back')
    #keyboard_builder.button(text='Поради по використанню продуктів', callback_data='advice')
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()


def get_hair_types():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Пористе', callback_data="hair_pic.0")
    keyboard_builder.button(text='Тонке', callback_data="hair_pic.2")
    keyboard_builder.button(text='Кучеряве', callback_data="hair_pic.1")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()


def get_type_of_care():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Шампунь', callback_data="shampoo")
    keyboard_builder.button(text='Бальзам', callback_data="balm")
    keyboard_builder.button(text='Маска', callback_data="mask")
    keyboard_builder.button(text='Сироватка', callback_data="serum")
    keyboard_builder.adjust(1, 1, 1, 1)
    return keyboard_builder.as_markup()


def get_skin_types():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Суха', callback_data="dry")
    keyboard_builder.button(text='Жирна', callback_data="fat")
    keyboard_builder.button(text='Комбінована', callback_data="combo")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()


def get_skin_type_of_care():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Пінка', callback_data="penka")
    keyboard_builder.button(text='Тонік', callback_data="tonic")
    keyboard_builder.button(text='Міцелярна вода', callback_data="mic")
    keyboard_builder.adjust(1, 1, 1)
    return keyboard_builder.as_markup()


def get_care (url):
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Переглянути на сайті', url=url)
    keyboard_builder.button(text='Назад', callback_data='back')
    return keyboard_builder.as_markup()


def get_admin_url():
    keyboard_builder = InlineKeyboardBuilder()
    keyboard_builder.button(text='Написати', url='https://t.me/skincare_support')
    return keyboard_builder.as_markup()
```