МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

Медецький Сергій Юрійович

Допускається до захисту:
завідувач кафедри
інформаційних технологій,
к. т. н., доцент
_____ Т. В. Нескородєва
«____» _____ 2021р.

РОЗРОБКА TELEGRAM БОТА ДЛЯ ПІДБОРУ ПРЕПАРАТІВ В АПТЕЦІ

Спеціальність 122 «Комп’ютерні науки»

Кваліфікаційна (бакалаврська) робота

Науковий керівник:
Антонов Ю.С., доцент кафедри
інформаційних технологій,
к. фіз.-мат. н., доцент

Оцінка: _____/_____/_____
(бали за шкалою ЄКТС/за національною шкалою)
Голова ЕК: _____
(підпис)

Вінниця – 2021

Медецький С.Ю. Розробка Telegram бота для допомоги у підборі препаратів у аптеці.

У бакалаврській роботі було проаналізовано предметну область та існуючі аналоги ботів.

Розроблено монолітну архітектуру для додатку, які дозволять вирішити цю задачу.

Розроблено додаток для адміністрування бота та взаємодії з Telegram API.

Проведено аналіз існуючих аналогів, які дозволять вирішити поставлену задачу.

Ключові слова: Java, Cuba Platform, PostgreSQL, Telegram API.

Табл. 21. Рис. 15. Бібліограф : 40 найм.

Medetskyi S.Y Development of a Telegram bot to help in the selection of drugs in the pharmacy.

The subject area and existing analogues of bots were analyzed in the bachelor's thesis.

Developed a monolithic architecture for the application, which will solve this problem.

An application for bot administration and interaction with the Telegram API has been developed.

The analysis of existing analogues which will allow to solve the set task is carried out.

Keywords: Java, Cuba Platform, PostgreSQL, Telegram API.

Tabl. 21. Fig. 15. Bibliography: 40 items.

ЗМІСТ

ВСТУП

Актуальність роботи: У Telegram 200 мільйонів активних користувачів в місяць. Кожна людина використовує Telegram у своїх особистих цілях. Більшість знає, що таке Telegram-боти, і користуються ними на постійній основі. Боти використовуються в багатьох сферах такі як: страхування, комунальні послуги, медицина, фінанси і так далі. Насправді найти спосіб використання можна в любій справі.

Що ж таке Telegram-бот? «Бот Telegram – це акаунт Telegram, який управляється програмним забезпеченням, іноді має функції штучного інтелекту. Боти можуть вчити, грати, використовувати пошук, нагадувати, інтегруватись з іншими службами або навіть передавати команди іншим сервісам.»

Існує багато варіантів Telegram ботів такі як: боти для спілкування з клієнтами, боти які нагадують про якусь подію, боти на базі веб-сервісів, боти для проведення спецпроектів і конкурсів і так далі. З кожним днем користувачів Telegram стає все більше і більше, так само і людей які починають користуватись Telegram ботами в будь-якій своїй сфері.

Якщо порівнювати дві компанії, яка використовує бота і в якій всі ці дії робить людина, то там де є бот, все буде значно швидше. В такому випадку компанія, якій розробили певного бота для полегшення праці, буде значно швидше опрацьовувати всі запроси, які їм поступили.

З огляду на все сказане вище, можна стверджувати, що користування Telegram бота дуже полегшує життя, також може допомогти розв’язати певну проблему, як користуватись ботом, залежить від людини. Завдяки бота можна відпочити, попрацювати, згадати забуте і так далі.

Мета і завдання дослідження. Метою даної роботи є полегшити підбір препаратів у аптеці завдяки Telegram бота. Для досягнення мети потрібно вирішити такі завдання:

- Проаналізувати аналоги ботів.
- Проаналізувати існуючі мови програмування для створення бота.
- Розробити Telegram бота на основі результатів аналізу та моделювання.

Об’єктом дослідження є засоби миттєвого обміну повідомленнями.

Предметом дослідження: бот, як засіб взаємодії із користувачем.

Теоретичне значення одержаних результатів полягає у описанні загальних принципів побудови бота в додатку Telegram.

Практичне значення одержаних результатів полягає у розробці сервісу для платформи Telegram, який допомагає підібрати препарат.

РОЗДІЛ 1

ОГЛЯД ТА АНАЛІЗ СУЧАСНИХ ТЕНДЕНЦІЙ ТА МОЖЛИВОСТЕЙ РОЗРОБКИ TELEGRAM БОТІВ

1.1 Telegram

Особливості користування саме Telegram, які відзначили самі розробники, демонструючи це все в картинках (рис. 1.1)

Рисунок 1.1 – Особливості Telegram [3]

Конфіденційність та безпека. Відразу два важливих пункти. Захищеність особистих повідомлень від сторонніх людей стала за основу, яку хотіли всі люди, тому Telegram став одним із самих популярних месенджерів. З кожним роком витоків інформації все більше і більше, потрібно завжди

думати чим додатковим скористатись, щоб це все оминути. В Telegram всі листи зашифровані та можуть самоліквідуватись.

В Telegram дуже важко зламати дані, потрібно буде зламати мозок, навіть якщо хтось візьме в руки диск з інформацією. А повідомлення які видаляються самі по собі ніде не зберігаються. До захищення особистої інформації Дурови відносять дуже клопітливо. Кожен рік вони влаштовують конкурси на хакерські атаки, що виявили якісь недоліки в системі. Той хто переміг отримує гроші, а Telegram стає все більш надійним.

В Telegram можна отримати доступ з будь якого пристрою - комп'ютер на будь-якій ОС, так само планшет і телефон, тому що для зберігання інформації використовуються віддалені сервери.

Швидкість. В різних частинах міст, країн, континентів розташовані дата-центри, та в кожного з них своя юрисдикція. Відправлення повідомлень відбувається за лічені секунди, навіть менше. Розробники запевняються, що в них сервіс найшвидкий. Сервіс стовідсотково швидше за звичайну відправку звичайних повідомлень та пошти. А рахунок на долі секунди та відбувається таким чином у більшості інших месенджерах.

Всеосяжність. Всі сервери розкидані, можна так мовити, по всьому світу. Це зроблено для того, щоб ускладнити роботу для людей які хочуть зламати та отримати доступ до особистої інформації користувачів, а також для того щоб забезпечити стабільну та високу швидкості роботи. Загального центру не існує, є тільки п'ять дата-центрів, які відповідають за певний регіон.

Відкритість. Кожен користувач може створити свою, можливо на його погляд, поліпшену або доповнену версію, і це все завдяки відкритому коду Telegram та API MTProto. Насправді, неофіційних версій дуже багато. Після того як ви створили свою версію, і для того щоб вона стала офіційною, працівники компанії будуть дивитись на ваш продукт і приймати рішення, наскільки вона може бути повноцінною і чи дотягує вона за рівнем крутості. MTProto можна використовувати в комерційних цілях, але тільки в тому

випадку якщо гроші не розподіляються між акціонерами. Якщо коротко, то Mail.ru API використовувати заборонили.

Безкоштовність. Завдяки досвіду з "ВКонтакті", новий проект Дурова не продається і не купується. На самому початку, Дуров зробив певну пожертву, і натепер додаток ніяких вкладень не потребує. Хоча у майбутньому, якщо настане чорний день для Telegram, то можливо будуть введені необов'язкові платні функції, але це не факт. Швидше за все, проект буде жити з інвестицій. Не планують вводити навіть рекламу. А реєстрація в Telegram завжди буде безкоштовною. Всі в компанії за безпеку спілкування та просувають це в маси.

Кросплатформеність. Офіційних клієнтів тільки три, але вважаючи на це, люди постворювали багато своїх версій для будь-якої платформи, навіть створили для платформи Linux. Так само кожен з нас має змогу створити щось своє.

Потужність. Telegram не має ніяких обмежень, чи то кількість повідомлень відправлених за добу, чи то документів, музики, відео, зображень або використання програми. Ніхто не забороняє спілкуватись, скільки користувач захоче, стільки і зможе спілкуватись. Навіть двісті людей в одному чаті. Один чат для всієї компанії.

Підтримка. Останній пункт, який теж не маловажливий, і про який потрібно знати. Telegram це безпечне спілкування на просторі Інтернет, і чим більше люди один одному про це розповідають, тим більше нових користувачів, тому що кожен прагне того, щоб його дані були недоступні ні для кого. У розробників немає фейсбук-сторінки, і вони не спілкуються "ВКонтакті", але в них є офіційних Twitter, де вони розповідають про якісь новинки, про розвиток в майбутньому, також відповідають іноді на часті запитання.

1.1.1 Огляд сучасних ботів помічників.

Здається, що Telegram за останні пару років став дещо більшим, чим звичайний месенджер. Тут можна не тільки обмінюватись повідомленнями, а

ще й спілкуватись по відеозв'язку, також влаштувати голосовий чат. Не хочеться вірити в те, що Telegram може стати додатком, в якому дуже багато всього, і все воно не потрібне. Насправді кожна ця річ приносить дуже велику користь і спрощує користування додатком для багатьох людей.

На даний момент, майже кожна людина користується ботом в Telegram. Можливість створювати їх з'явилась у 2015 році, і починаючи з того часу дуже велика кількість компаній, банків та інших сфер почали створювати своїх ботів. Вони розуміють, що Telegram боти можуть пришвидшити їх працю, також можуть убезпечити комфортну взаємодію з їх послугами.

Приклад найвідоміших і найзручніших ботів:

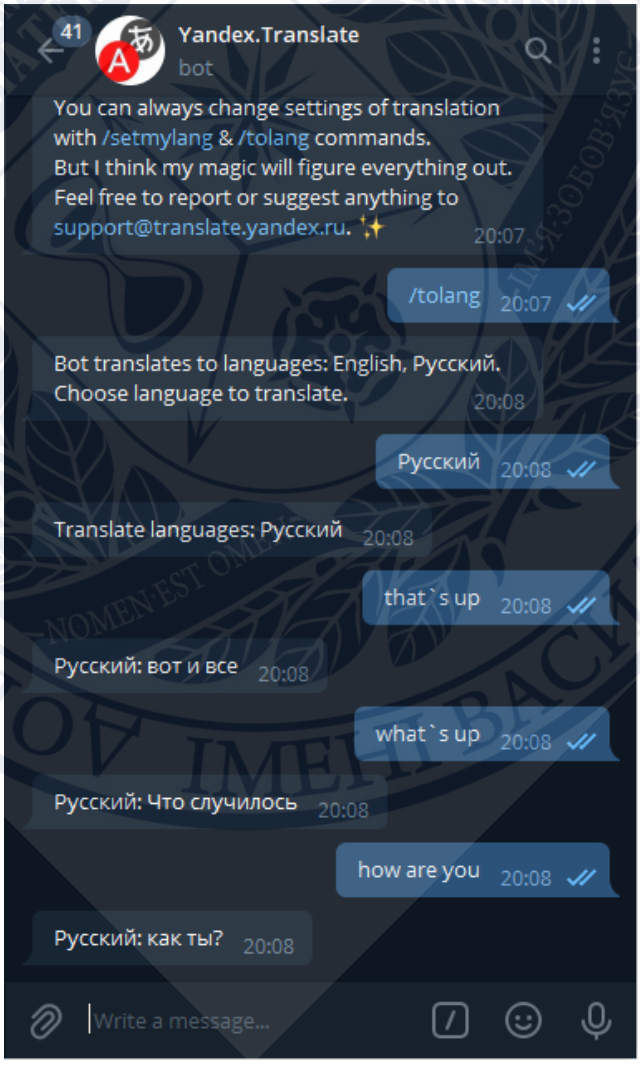

Рисунок 1.2 – бот «Yandex.Translate»

Коли спілкуєшся з людиною, набагато зручніше та швидше буде скористатись ботом-перекладачем в месенджері. На допомогу приходить Telegram бот «Yandex Translate», який інтегрований з перекладачем від Yandex.

Цей бот має дуже простий функціонал. Він підтримує 12 мов, серед яких: російська, англійська, німецька та інші.

Команда /setmylang дозволяє вибрати мову, з якої перекладати, а команда /tolang – мова, на яку потрібно перекласти.

Бот «Voicy» - перекладає голосові повідомлення в текстові.

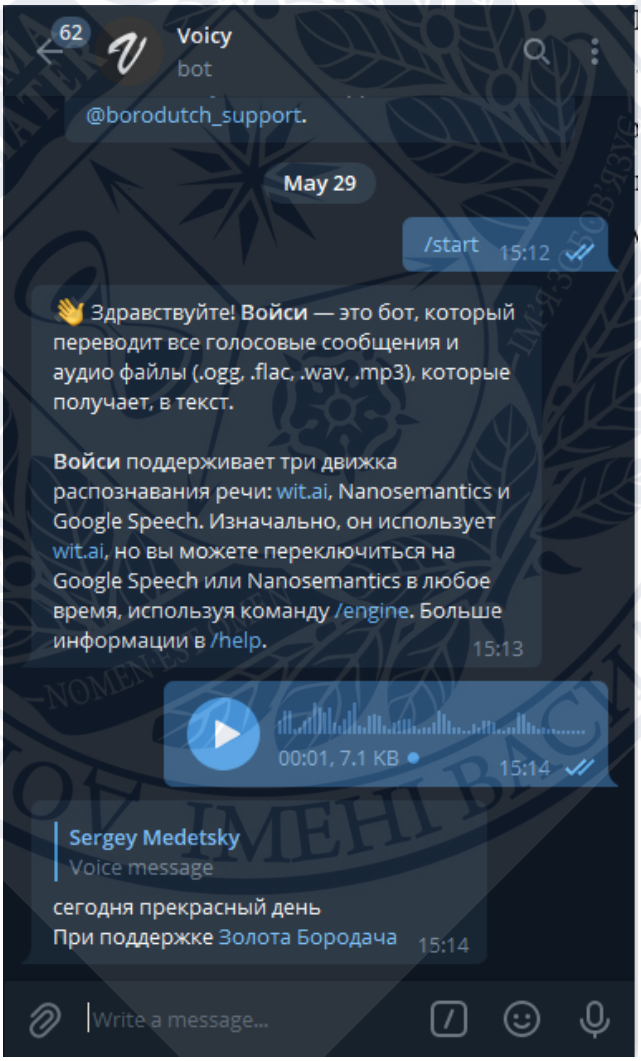

Рисунок 1.3 – бот «Voicy»

Не завжди зручно слухати голосові повідомлення, і саме для цього був створений бот Voicy. Він працює за дуже прости принципом: перекладає в текст всі голосові повідомлення, а також аудіодоріжки форматів .oog, .flac, .wav, .mp3.

Цей бот уже продемонстрував свою надійність. Він ще й може обробляти велику кількість мов. Для такої тяжкої задачі команда Voicy використовує два двигуна розпізнавання речі – Wit.ai та Google Speech. При бажанні, ними можна управляти в налаштуваннях бота.

Бот «SaveAsBot» -бот-завантажувач улюбленого контента з Instagram та TikTok.

Рисунок 1.4 – бот «SaveAsBot»

Дуже зручний бот для завантажування матеріалу з Instagram та TikTok. Іноді хочеться загрузити звідти якесь смішне відео. Цей бот саме це і робить, і дає можливість скачати все в чудовій якості.

Коли бот загружає відео з TikTok, то разом з відеороликом відправляє окремим файлом аудіодоріжку, а у випадку з Instagram – всі матеріали користувач отримує файлами для збереження високої якості зображення.

1.1.2 Мови, що підходять для створення Telegram ботів

На сьогоднішній день існує дуже велика кількість мов програмування. Розглянемо найбільш відомі, такі як: Python, C#, Java.

Мова програмування Python – інтерпретована об’єктно-орієнтована мова програмування високого рівня, в якої строга динамічна типізація. Такою чудовою та відомою її роблять в основному засіб, який поєднує компоненти, які наявні в ній, а також структури даних, які насамперед високого рівня та динамічне зв'язування. [4].

Мова програмування Python базується на підтримці модулів та пакетів модулів, завдяки цьому можна ще раз використати код. Python дозволяє писати код, де є можливість добре прочитати. Завдяки тому, що Python дуже лаконічний, програми, які будуть написані завдяки цій мові, будуть значно менше за розміром за свій аналог, який був написаний іншою мовою програмування. І ще це багатоплатформова мова. Для багатьох це дуже важливий пункт, адже завдяки цьому можна буде запускати програми, написані на мові Python, на будь-якій операційній системі, навіть без того, що змінювати код [5].

Розглянемо не менш відому і зручну мову для написання бота в Telegram – C#. C# - об’єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET [6].

Мова C# було розроблена компанією Microsoft, також цю мову можна назвати однією з самих сучасних, та ще й однією з самих відомих. Ця мова затребувана на ринку для програмістів по всьому світі. Цю мову використовують в основному для роботи з персональним комп'ютером, для

створення мобільних додатків і так далі. Спочатку ця мова була створення для особистих потреб платформи Microsoft, але з часом ця мова стала одною з самих популярних мов у світі. [6].

Синтаксис дуже близький до C++, і ще нагадає синтаксис мови Java. Цій мові власна строга статична типізація, вона має змогу підтримувати поліморфізм, вміє перевантажувати операторів і так далі. [6].

Для створення мого бота, я вибрав мову програмування Java. Java - це мова, яка являється об'єктно-орієнтованою мовою програмування, була розроблена компанією Sun Microsystems.

Компанія що займається відстеженням якості програмного забезпечення TIOBE Software представила оновлену статистику найпопулярніших мов програмування в лютому нового року. Java займає друге місце з рейтингом в 11,29% [7] (рис. 1.5).

Feb 2021	Feb 2020	Change	Programming Language	Ratings	Change
1	2	^	C	16.34%	-0.43%
2	1	v	Java	11.29%	-6.07%
3	3		Python	10.86%	+1.52%
4	4		C++	6.88%	+0.71%
5	5		C#	4.44%	-1.48%
6	6		Visual Basic	4.33%	-1.53%
7	7		JavaScript	2.27%	+0.21%
8	8		PHP	1.75%	-0.27%
9	9		SQL	1.72%	+0.20%
10	12	^	Assembly language	1.65%	+0.54%

Рисунок 1.5 Рейтинг мов програмування [7]

Програми які написані на Java майже завжди транслюються в спеціальний байт-код, саме через це у них є можливість працювати на будь-якому персональному комп’ютеру, головне щоб існувала реалізація віртуальної JVM[8].

Дуже великим плюсом являється насамперед повна незалежність байт-коду від будь-якої операційної системи та устаткування, це дозволяє запускати додатки, які написані завдяки мові програмування Java на будь-якому пристрої, для якого існує відповідна JVM. Ще один дуже значний плюс це гнучка система безпеки, завдяки чому в рамках якої виконання програми контролюється повністю JVM. Негайне преривання викликають програми, які перевищують встановленні повноваження [8].

1.2 Фреймворк Cuba Platform

Cuba об’єднує широко поширені JVM технології в високоефективну платформу, що відповідає сучасним стандартам розробки і типових вимог до корпоративних додатків.

CUBA природним чином надає більшість можливостей Spring, що дозволяє покластися на його велику екосистему і наявний у вас досвід. Якщо ви не знайомі з Spring, CUBA надає простий спосіб почати з ним працювати.

Архітектура платформи дозволяє включити будь-CUBA-додаток до складу іншого. Це дає можливість легко реалізувати модульність, розробляючи окремо компоненти програми і об'єднуючи їх в подальшому в єдину систему.

Платформа дозволяє створювати горизонтально і вертикально масштабовані рішення. Виходячи з планованого навантаження додатки і допустимого часу простою, підтримуються різні варіанти розгортання.

CUBA-додатки сумісні з популярними реляційними СУБД і працюють в будь-якому Java контейнері сервлетів. Додатки можна поширювати у вигляді WAR, Docker image, UberJar або розгортати в хмарі.

Інтерфейс для зовнішніх користувачів. Публічно доступні інтерфейси зазвичай припускають нетиповий дизайн, а також можливість комфортної роботи з будь-якого пристрою в умовах малопередбачувані навантаження. Для реалізації цих вимог платформа надає генератор коду для розробки UI на React.js або Google Polymer з бекенд на CUBA(рис. 1.6.).

CUBA пропонує модульну і масштабовану архітектуру на основі популярних фреймворків, створену для роботи в будь-якому оточенні. Модуль Generic UI в рази прискорює розробку інтерфейсів для внутрішніх користувачів.

Нефункціональні вимоги - підводна частина айсберга корпоративних додатків, яка здатна перетворити автоматизацію навіть найпростішого бізнес-процесу в головний біль. Доповнення CUBA дозволяють практично в один клік реалізувати типові вимоги, такі як управління користувачами і доступом до даних, інструменти адміністрування, звітність і підтримка BPM.

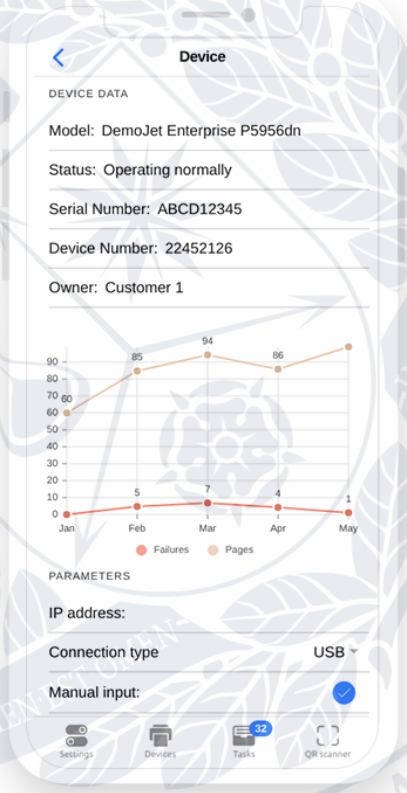

Рисунок 1.6 інтерфейс для зовнішніх користувачів

За допомогою зручних засобів розробки ви зможете швидко зануритися в екосистему CUBA. Інструменти забезпечують по-справжньому комфортний і продуктивний процес розробки завдяки генерації коду, візуальним редакторам, контекстним підказкам і інтуїтивної навігації.

В першу чергу, платформа CUBA націлена на розробку корпоративних інформаційних систем. Типові характеристики таких додатків: розвинена модель даних, десятки і навіть сотні щодо типових екранів, підтримка великої кількості складних бізнес-процесів, безліч звітів, вимоги щодо розмежування прав доступу, і так далі.

Ключові особливості платформи:

- високий рівень абстракції щодо технологій, що лежать в її основі - Vaadin, Spring, EclipseLink і так далі.
- готові, інтегровані компоненти, які вирішують багато типові завдання корпоративних систем
- візуальні засоби розробки і розвинена генерація шаблонного коду

У сукупності, це дозволяє мінімізувати тимчасові витрати на "системні" завдання - настройку інфраструктури проекту, інтеграцію технологій і компонентів, розробку базової функціональності - і сконцентруватися на реалізації бізнес-вимог. У той же час, CUBA не обмежує доступ до низкорівневого коду, гарантуючи можливість адаптувати його під потреби проекту.

Додатки на базі CUBA мають стандартну тришарову архітектуру. Елементом системи є метадані - інформація про моделі даних програми. Завдяки метаданим візуальні компоненти знають про те, з якими даними вони працюють. Так, наприклад, таблиця знає, що відображає атрибути сутності "водій" і автоматично заповнює імена і формат колонок. Таким же чином метадані допомагають візуальним компонентам працювати з базою даних через ORM, задаючи графи об'єктів, які потрібно завантажити або оновити. Той же принцип застосовується до підсистеми безпеки, генерації звітів і іншим частинам платформи.

CUBA надає великий набір можливостей для створення бізнес-додатків. Якщо вбудованих засобів платформи не вистачає, то можна то можна доповнити додаток зовнішніми компонентами. Платформа включає наступний набір функцій:

- управління користувачами і засоби адміністрування
- створення звітів
- управління бізнес-процесами у вбудованому візуальному дизайнера
- багатомовний інтерфейс і підтримка різних часових поясів
- повнотекстовий пошук
- універсальний REST API
- інтеграція з LDAP
- ...і багато іншого

Якщо буде потрібно розробити призначений для користувача інтерфейс з використанням JavaScript бібліотек, таких як ReactJS, Angular або Vue.js, на додаток до "стандартного" набору екранів, то CUBA надає модуль REST API і генератор TypeScript SDK. Отриманий в результаті SDK буде містити всі необхідні класи моделі даних і виклики сервісів, які можуть бути використані при розробці клієнтської програми.

1.3 База даних PostgreSQL

PostgreSQL - можна назвати вільною об'єктно-реляційною системою управлінням над базою даних.

Вона може бути реалізована в безліч платформ, такі як: AIX, BSD, IRIX, macOS, Linux, Tru64, а в особливості для Windows будь-якої розрядності, як і було реалізовано для Telegram бота. [13]

Сильні сторони PostgreSQL вважаються:

- високопродуктивність та надійність механізмів транзакцій та реплікацій;
- велика кількість підтримки різних мов програмування: в звичайній версії будуть підтримуватись PL/pgSQL, PL/Perl, PL/Python та PL/Tcl; також є можливість використовуватиі інші додаткові PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh і PL/V8, а також можливо завантажити модулі розширення на мові C [10];

- можливість спадкувати;
- можливість індексувати геометричні (в основному, географічні) об'єкти;
- наявна підтримка з слабоструктурованими даними в форматі JSON, також є можливість індексувати їх;
- можливість розширювання (можна буде створити якісь нові типи даних або індексів, різних мов програмування, наявні також розширювальні модулі).

SQL-функції виконують довільний список операторів SQL і повертають результат останнього запиту в списку. У простому випадку буде повернуто перший рядок результату останнього запиту. Якщо останній запит взагалі не поверне рядки, буде повернуто значення NULL.

Тіло SQL-функції має являти собою список SQL-операторів, розділених крапкою з комою. Крапка з комою після останнього оператора може бути відсутнім.

Функції - це фрагменти коду, які виконуються не на клієнті бази даних, а на сервери .Їх можна створювати на чистому SQL, проте створення додаткових можливостей(наприклад, перетворень, циклів та ін.) потребує певних розширень мови, оскільки така додаткова логіка виходить за рамки SQL. Функцію можна написати однією з мов [14]:

- Вбудована процедурна мова PL/pgSQL, багато в чому аналогічний мові PL/SQL, що використовується в СУБД Oracle;
- Скриптові мови - PL/Lua, PL/LOLCODE, PL/Perl, PL/PHP, PL/Python, PL/Ruby, PL/sh, PL/Tcl, PL/Scheme, PL/v8 (Javascript);
- Класичні мови - C, C ++, Java (через модуль PL/Java);
- Статистичний мову R (через модуль PL/R).

PostgreSQL може підтримувати декілька користувачів, які модифікують базу даних, це все завдяки механізму Multiversion Concorrency Control

(MVCC). Через це майже відпадає потреба в блокуванні читання і при цьому потрібно притримуватись вимоги ACID.

Користувачі PostgreSQL можуть розширити практично будь-який аспект відповідно до своїх потреб. Можна додати свій власний:

- типи даних;
- перетворення типів;
- домен (власний тип із обмеженнями з самого початку);
- функції (включаючи агрегування));
- індекс;
- Оператори (включаючи перевизначення існуючих операторів);
- мова програмування;

Висновки до розділу 1.

В цьому розділі ми розглянули три мови програмування, які найбільш підходять для створення Telegram бота, також детально розглянули про фреймворк Cuba Platform та розглянули базу даних PostgreSQL.

РОЗДІЛ 2
РОЗРОБКА МОДЕЛІ TELGRAM БОТА ДЛЯ ДОПОМОГИ У ПІДБОРІ ПРЕПАРАТІВ У АПТЕЦІ

2.1 Розробка архітектури додатку

Насамперед важливо правильно розуміти принцип роботи Telegram бота, які функції він буде виконувати. Така інформація представлена на зображенні 2.1.

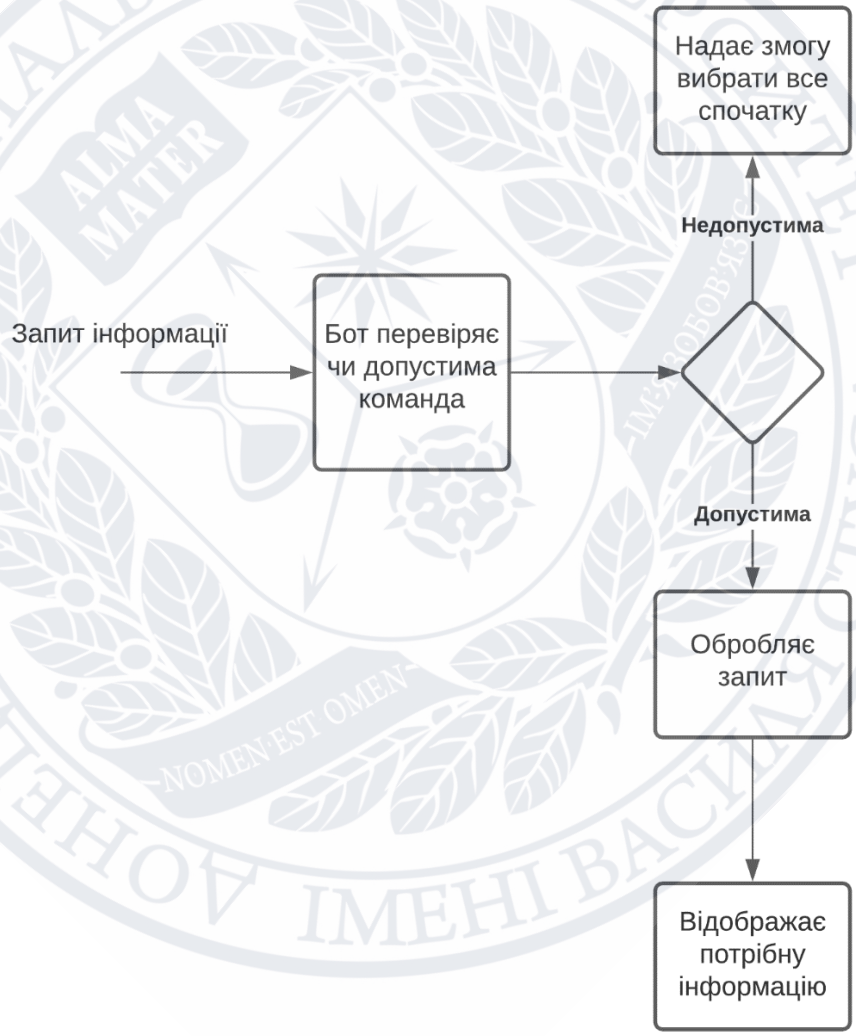

Рисунок 2.1 – Приклад роботи бота

При відправці повідомлення користувачем, воно в зашифрованому вигляді відсилається на сервера Telegram, звідки автоматично

перенаправляється в Telegram-бот. Далі в боті відбувається обробка повідомлення, якщо воно відправлено для внесення налаштувань, то відбувається запис нових даних і відправка листа у відповідь користувачеві.

Переваги правильної архітектури:

- Система має змогу масштабуватись без радикального рефакторинга. Іншими словами під час розвитку не буде потрібно міняти фундамент, є змога тільки покращувати. Внесені зміни в системі не призведуть до глобальної переробки системи.
- Якщо людина захоче залучити програмістів для покращення і розвитку системи, можна найняти різні команди, для того щоб вони не були пов'язані один з одним. Завдяки цьому можна зменшити ризик витоку даних.
- В системі є можливість налаштувати різні категорії для користувачів, і надавати їм доступ до інформації яка їм потрібна.
- Так, як зроблено для одного бота, то можна підключати безліч різних і ботів, і сервісів.

2.2 Розробка бази даних

Необхідно створити базу даних, яка буде відповідати наступним вимогам:

- зберігання вибору користувача (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, вибір користувача);
- зберігання виробника препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання форми випуску препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);

- зберігання країни виробника препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання фармакотерапевтичної групи (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання типу препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання приготування препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання терміну придатності препаратів (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, назва препарату);
- зберігання реляційного відношення (два унікальних ідентифікатора);
- зберігання всієї інформації про препарат (час створення запису, той хто створив запис, час оновлення запису, той хто редагував востаннє, час видалення запису, той хто видалив запис, торгову назву, умови відпустки, лікарський склад, адресу виробника, номер свідоцтва про реєстрацію, чи являється біологічного походження, чи являється рослинного походження, чи є препарат сирота, чи це гомеопатичний препарат, дату інструкції, термін придатності).

2.3 Обґрунтування технологій

2.3.1 Мова програмування Java

Я вибрав мову програмування, для написання Telegram бота, Java. У Java маса переваг, завдяки яким багато розробників вибирають саме його.

Простота - перша технічна гідність Java. У нього чіткі синтаксичні правила і зрозуміла семантика. Раціональність і стислість дуже корисні для

обробки коду машинами з обмеженим обсягом ресурсів. Для вбудованих пристроїв створена спеціальна платформа Java Micro Edition.

Об'єктно-орієнтований підхід. За 3 десятиліття він довів свою ефективність. Суть полягає в тому, що в центрі уваги знаходяться дані (об'єкти), інтерфейси і алгоритми вторинні. Іншими словами, ми відштовхуємося від результату при виборі інструментів, способів їх застосування.

Безпека. Найважливіший критерій, з огляду на застосування мови в мережевих / розподілених середовищах. Розробники виконали велику роботу по захисту платформи Java. І вона триває. Обійти або зламати механізми захисту вкрай складно. Приклад: використання класів, що мають цифровий підпис. Повні права надаються тільки при повній довірі автору класу.

Продуктивність. Спочатку вона викликала питання. Нові версії динамічних компіляторів Java не поступаються традиційним з інших платформ. Потужний приріст швидкості обробки дає оптимізація тих фрагментів коду, яка виповнюється частіше. При необхідності ті чи інші прийоми оптимізації включаються або скасовуються JIT-компілятором.

Надійність - одне з найважливіших переваг. Програми на Java стабільно працюють в будь-яких умовах. Компілятор здатний виявити помилки ще до виконання коду, тобто на ранніх стадіях. Контроль виконання дозволяє запобігти збоїв в пам'яті (наприклад, через неточного покажчика). Самі покажчики можна застосовувати не скрізь, а тільки там, де це необхідно (скажімо, в роботі зі зв'язковими списками).

Незалежність від апаратної частини і ОС. Важливо лише наявність виконуючого середовища і JVM. А комп'ютерна архітектура в цілому значення не має. Байт-код легко інтерпретується на будь-якій машині. Підхід довів свою спроможність багато в чому завдяки динамічної компіляції. Кроссплатформенную відрізняється також інтерфейс, реалізований в системних бібліотеках.

Динамічність і адаптованість. Ця особливість дозволяє Java не губитися в постійно мінливому середовищі. При необхідності можна додати в бібліотеки нові об'єкти, методи. При цьому чіпати програму для використання дані бібліотеки не потрібно. Відстежити інформацію про структуру об'єктів, їх поведінці, про хід виконання програми дуже легко.

Зручні та ефективні мережеві можливості. Додатки вміють знаходити потрібні об'єкти в мережі і відкривати до них доступ. Причому так само легко, як ніби ми маємо справу з локальної файлової системою. Є велика програмна бібліотека для передачі даних по найпоширенішим протоколам: FTP, HTTP, TCP / IP. Працює механізм виклику віддалених методів.

Не треба забувати, що Java - це тріо, що складається з мови програмування, потужного універсального обробника і значною бібліотеки. Всі ці напрацювання доступні програмістам. Їм не потрібно розробляти з нуля багато необхідні процедури (доступ до мережі, баз даних тощо). Це теж вагомий аргумент на користь Java.

2.3.2 СУБД PostgreSQL

PostgreSQL - це популярна вільна об'єктно-реляційна система управління базами даних. PostgreSQL базується на мові SQL і розширює можливості.

Переваги PostgreSQL[18]:

- Можливість підтримки БД необмеженого розміру;
- Наявність потужних і надійних механізмів транзакцій і реплікацій;
- Можливість розширювати системи вбудованих мов програмування і підтримка завантаження С-сумісних модулів;
- Можливість спадкування;
- Зручна та легка розширюваність.

Поточні обмеження PostgreSQL:

- Відсутнє обмеженя на максимальний об'єм БД
- Відсутнє обмеженя на кількість записів в таблиці

- Відсутнє обмеженя на кількість індексів в таблиці
- Максимальний об'єм таблиці - 32 Тбайт
- Максимальний об'єм запису - 1,6 Тбайт
- Максимальний об'єм поля - 1 Гбайт
- Максимальна кількість полів в записі від 250 до1600 (все залежить від типів полів)

Особливості PostgreSQL:

В СУБД PostgreSQL функції являються блоками коду, які виконуються тільки на сервері, і ніяким разом не на БД клієнта. SQL вимагає використання певних мовних розширень, власне тому і виходить за рамки, і завдяки цьому є можливість писати на чистому SQL, де реалізована додаткова логіка, наприклад переходи і цикли. Є можливість писати функції на різних мовах програмування. Набір записів можна використовувати таким чином, щоб виконання результату запиту було таким самим, це все допускається у використанні функцій в PostgreSQL. Функції можна використовувати у двох режимах: з правами того хто створив, і з правами самого користувача. Інколи порівнюють функції зі збереженими процедурами, але це дуже різні поняття.[24]

В СУБД PostgreSQL є таке поняття, як тригери, його ініціалізують як функції. Також вони ініціюються DML-операціями. Прикладом може бути операці INSERT, вона може запускати тригер. Ця операція перевіряє доданий запису на відповідність умовам. Для того щоб написати функцію для тригеру можна використовувати безліч різних мов програмування. Також тригери порівнюють з таблицями. Якщо тригерів багато, то їх використовують в афлавітному порядку. [24]

В СУБД PostgreSQL є так званий механізм правил. Завдяки цьому можна створювати операції виборки, а також створювати користувацькі обробки DML-операцій. Існує певна низка відмінностей від так званого механізму тригерів, але основна відмінність це та, що правила будуть спрацьовувати на

початку створення запиту, до того часу поки не буде вибрано план виконання, який можна буде назвати оптимальним і до того поки не буде виконано цей план. Також ці правила надають змогу перевизначити стан системи, коли виконуються SQL-операції.[24]

В СУБД PostgreSQL існує таке поняття, як індекси. Вони бувають таких типів: R-дерево, GIN, GiST, хеш, B-Дерево. Є можливість створювати інші типи індексів, але це вже процес не з легких.[24]

В СУБД PostgreSQL є підтримка багатоверсійності, це означає, що є можливість одночасної модифікації однієї БД кільком користувачам. Це відбувається завдяки механізму, який називається Multiversion Concurrency Control, якщо скорочено то MVCC. В наслідок цього механізму майже не потрібно буде блокувати читання, і також буде дотримуватись вимоги ACID.[24]

В СУБД PostgreSQL є можливість розширення майже для будь-якого виду діяльності. Є багато різних можливостей, такі як: додавання власних перетворень типів, додавання типів даних, доменів (призначених для користувача з самого початку з накладеним обмеженням), функцій (присутня можливість включання агрегатних), індексів, операторіві і процедурних мов.[24]

В СУБД PostgreSQL реалізовано спадкування на рівні таблиць. Всі таблиці успадковують набори полів та певні відомості від других таблиць, так званих батьківських. Водночас дані які додались в породжену таблицю, будуть в автоматичному режимі сприяти (коли не було сказано інакше) запитам які знаходять в батьківській таблиці.[24]

2.3.3 Фреймворк Cuba Platform

Платформа CUBA - результат узагальнення більш ніж десятирічного досвіду побудови корпоративних додатків, об'єднаного з найсучаснішими технологіями розробки програмного забезпечення.

Продумана архітектура і широкий набір модулів платформи CUBA забезпечують можливість швидкого і успішного вирішення будь-яких завдань по автоматизації бізнес-процесів.

Особливості платформи CUBA:

- Скорочення витрат замовника на придбання ліцензій у зв'язку із застосуванням вільно розповсюджуваних технологій.
- Крос-платформенність: можливість роботи з будь-якою операційною системою і практично з будь-якою системою управління базами даних (СКБД), в тому числі вільно поширюваними.
- Застосування ефективних засобів мережевої безпеки і розмежування прав доступу користувачів.
- Автоматичний аудит дій користувачів, включаючи історію змін даних.
- Масштабованість і відмовостійкість.
- Простота інтеграції з зовнішніми додатками.
- Простота і швидкість розробки бізнес-додатків, впровадження та оновлення; можливість поновлення без зупинки системи (для критичних систем, що працюють в цілодобовому режимі).
- Можливість роботи в системі, створеній на платформі, з будь-якої точки світу, де є Інтернет.
- Відкритий програмний код: можливість подальшого розширення додатків власними силами замовника.
- Архітектура додатків на платформі CUBA.

2.4 Розробка архітектури додатку

Монолітний додаток являється додатком, що доставляється через єдине розгортання. Завдяки цьому цей додаток буде доставлений з виглядом однієї WAR. Так само можу бути доставлений як додаток Node з однією точкою входу.

Переваги монолітної архітектури додатку:

Дуже великим плюсом монолітної архітектури являється легка його реалізація. В цій архітектурі можна буде швидко та легко реалізувати будь-яку свою ідею, не потрібно буде витрачати зайвий час на роздуми про взаємодію процесів.

Ще одним плюсом являється наскрізні тести. У монолітній архітектурі їх легше виконати.

Якщо буде мова про операції, то навряд чи можна буде сказати, що архітектура буде легко масштабуватись, або вона буде проста в розгортанні. Для того щоб завантажити модуль, та запустити додаток можна використати скрипт для розгортання. Масштабування можна досягти завдяки шляху розміщення Loadbalancer. Це і є доказом того, що моноліт досить простий в експлуатації.

Недоліки моноліту:

Зазвичай, моноліти стають так званими "великими кульками бруду", тому що вони спочатку були в чистому вигляді. Якщо коротко, то це стан, що виник, тому що зрослись певні компоненти та були порушені правила.

Кожна така зміна, збільшує час витрачений на розробку, тому що буде все важче і важче розвивати нову функцію. Завдяки тому, що кожен компонент разом, їх потрібно змінювати теж разом. Якщо потрібно створити нову функцію, то це може означати дотик до різних місць: місця, де потрібно буде написати тести, можливо що будуть якісь побічні ефекти для тих функцій, які вже існують.

В цьому виді архітектури дійсно можна легко масштабувати, але це тільки до тих пір, поки це не стане "великою кулькою бруду", як згадувалось раніше. Також буде проблематично, якщо ви захочете масштабувати якусь одну частину моноліту, тому що в цій архітектурі не можна масштабувати окремі частини.

На жаль у монолітної архітектури майже немає ізоляції. Весь додаток може зруйнувати або уповільнити одна проблема або помилка в модулі.

Дуже важливим в моноліті є вибір основи. Якщо не буде своєчасно відключено або оновлено початковий вибір, то в майбутньому це буде дуже важко зробити, тому що це потрібно робити одночасно для всіх частин системи.

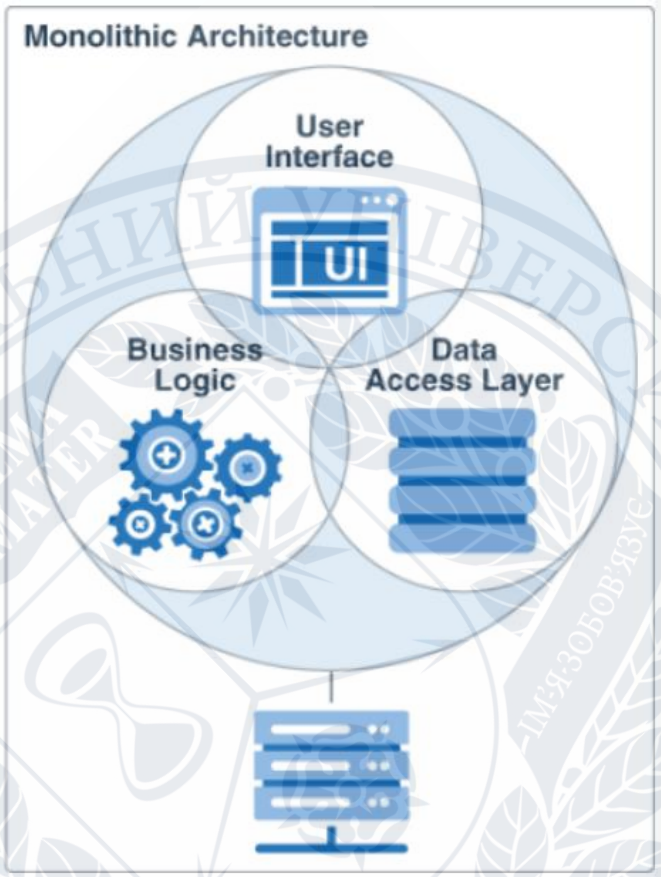

Рисунок 3.1– приклад моноліту.

Висновки до розділу 2.

В другому розділі було розглянуто переваги правильної архітектури. Було більш детально описано про СУБД PostgreSQL, також було описано про те, що використовувалось під час написання бота. Було розглянуто та вибрано монолітну архітектуру для створення бота.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ TELEGRAM БОТА ДЛЯ ДОПОМОГИ У ПІДБОРІ ПРЕПАРАТІВ У АПТЕЦІ

3.1 Огляд Telegram API

Telegram API – це простір для того, щоб розробляти пошукових ботів. Це середовище містить в собі вбудовані інструменти і сервер, де ці боти зберігаються. Кожен бот має власну адресу, там він знаходиться і звідти взаємодіє з людьми. Telegram бот – це програма, яка може виконувати певні дії за людину. Насправді способів використання ботів дуже багато, але в основному це пошук якоїсь інформації або наприклад імітація живого спілкування.

Telegram Bot API дозволяє створювати ботів. Боти - спеціальні акаунти, для реєстрації яких не потрібен номер телефону. Код бота повинен бути запущений на боці людини (ПК або сервері) - Telegram служить тільки інтерфейсом для прийому і відправки повідомлень.

Для використання API не потрібно мати ніяких специфічних знань про низькорівневі протоколи Telegram - всі запити виконуються по HTTP. Його можна відправити навіть вручну, за допомогою браузера.

Bot API це HTTP-інтерфейс для взаємодії з ботами в додатку Telegram. Кожен бот – це створений людиною акаунт, який автоматично обробляє та відправляє повідомлення.

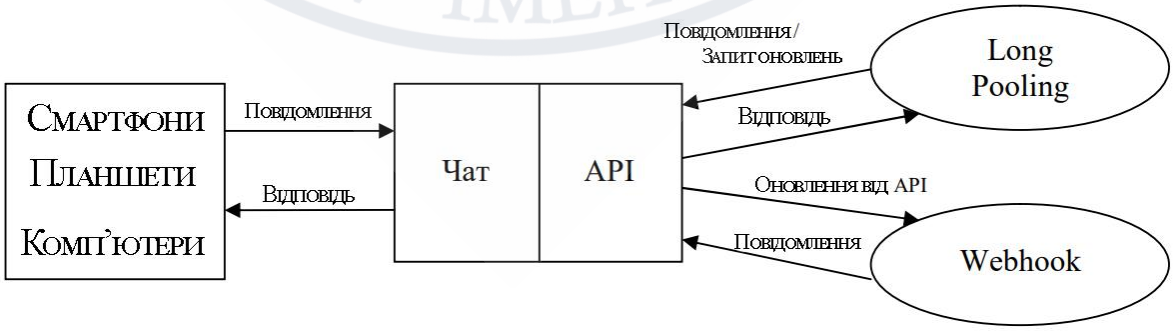

Рисунок 2.2 – Принцип роботи чат-бота на платформі Telegram

Коли створюється бот, йому присвоюється унікальний токен виду 234567: ABD-EFG1434ghIkl-zyx57S2v1u123ew11. Замість бота при веденні документації для полегшення сприйняття буде використовуватись <token>.

Такі запити GET та POST допускаються. Існують 4 доступних способи для передачі параметрів в Bot API [40]:

- Запит в URL
- application / x-www-form-urlencoded
- application / json (не підходить для завантаження файлів)
- multipart / form-data (для завантаження файлів)

Відповідь прийме вигляд JSON-об'єкта, в якому на постійній основі буде boolean поле "ок" та опціональне строкове поле descrition, в якому міститься опис результату, який може прочитати людина. Запит пройшов успішно та результат можна буде побачити в полі result, тільки в тому випадку, коли поле "ок" правдиве. Якщо допускається помилка, поле "ок" буде дорівнювати false, а в полі descriptrion буде описана помилка. На додаток, у відповіді буде цілочисельне поле error_code, але в майбутньому коди помилок будуть змінені.

Якщо бот працює через веб-хуки, то можна здійснити запит до Bot API одночасно з відправкою відповіді на веб-хук. Для того, щоб передати параметри, потрібно використати один з типів:

- application / json
- application / x-www-form-urlencoded
- multipart / form-data

Метод, який використовується для виклику має бути зазначений в запиті в полі method. Однак, неможливо дізнатися статус і результат такого запиту.

Існує два способи отримання оновлення від бота, які є діаметрально протилежні за логікою: getUpdates і веб-хуки. Максимальний час зберігання на сервері вхідних оновлень буде відбуватись до моменту їх обробки, але не довше 24 годин.

В любому випадку у відповіді ми маємо отримати об'єкт Update, який серіалізовний в JSON і все це незалежно від способу отримання оновлень.

Update - це об'єкт, який представляє якесь вхідне оновлення. Оновлення вжите в цьому випадку, в тому сенсі, що над ботом відбуваються будь-які дії - наприклад, коли приходить повідомлення від користувача.

Може бути тільки один необов'язковий параметр в кожному оновленні.

Для отримання оновлень з допомогою long polling слід використати метод GetUpdates. Відповідь буде повертатись у вигляді масиву Update.

SetWebhook - це метод, який потрібний для URL веб-хука. Бот повинен буде відправити повідомлення на нього. HTTPS з серіалізований в JSON об'єктом update, будуть кожен раз відправленні отримані оновлення. Якщо не вийде під'єднатися до сервера, це буде повторятись до того часу, поки не вийде [24].

Таблиця 3.1 – Параметри setWebhook [24]

Параметри	Тип
url	String
Sertificate	InputFile

Url - HTTPS url для відправки запитів; Sertificate – відбувається скачування публічного ключа для того, що перевірити кореневий сертифікат.

getWebhookInfo – це метод, який містить інформацію про поточний стан веб-хука. Параметр цього метода наведені в таблиці 3.2 [24].

Таблиця 3.2 – Параметри які повертає getWebhookInfo [24]

Параметри	Тип
url	String
has_custom_certificate	Boolean
pending_update_count	Integer

Продовження таблиці 3.2

Параметри	Тип
last_error_date	Integer
last_error_mess age	String

url - url веб-хука, це поле може бути порожнє; has_custom_certificate – якщо поле правдиве, і при умові що веб-хук використовує свій сертифікат, який самозавірений; pending_update _count – це поле демонструє кількість оновлень, які очікують своєї черги; last_error_date – це поле не є обов'язковим. Дата відправлення останньої помилки доставленого поновлення на даний веб-хук; last_error_mess age - це поле не є обов'язковим. В цьому полі відбувається описання помилки доставлення оновленнь на данний веб-хук, також це має бути остання помилка.

sendMessage – це метод, який потрібно використовувати для відправки сповіщень. Параметри методу наведені в таблиці 3.3 [24].

Таблиця 3.3 – Параметри sendMessage [24]

Параметри	Тип	Обов'язковий
chat_id	Integer або String	Так
Text	String	Так
parse_mode	String	Ні
disable_web_pag e_ preview	Boolean	Ні

Продовження таблиці 3.3

Параметри	Тип	Обов'язковий
disable_notificati on	Boolean	Ні

replay_to message_id	nteger	Hi
replay_markup	InlineKeyboar або ReplyKeyboard Hide	Hi

chat_id – це унікальний ідентифікатор основного чату або імені основного каналу; Text - текст повідомлення, яке потрібно відправити; parse_mode – для того щоб в додатку Telegram відображався різний за стильом текст, такий як: напівжирний, або жирний, або курсивний в повідомленні бота, потрібно буде надіслати markdown or HTML; disable_web_page_ preview – це поле відключає попередній перегляд посилань в повідомленні; disable_notificati on – це поле надає можливість тихо відправляти повідомлення. Ті люди, які користуються екосистемою Apple, то вони не отримують повідомлення, а ті хто користуються екосистемою Android отримують, але без звуку; replay_to message_id – це поле містить інформацію про ідентифікатор вихідного повідомлення; replay_markup – це поле дозволяє розширити можливості для пошуку інтерфейсу. Також це JSON-серіалізований об'єкт для клавіатури, яка вбудована, і також, яка повинна бути призначено для користувача.

sendPhoto – це метод, який використано для того, щоб відправляти фото. Параметри методу наведені в таблиці 3.4 [24].

Таблиця 3.4 – Параметри sendPhoto [24]

Параметри	Тип	Обов'язковий
chat_id	Integer або String	Так
Photo	InputFile або String	Так
Caption	String	Hi

Продовження таблиці 3.4

Параметри	Тип	Обов'язковий
replay_markup	InlineKeyboard Markup or ForcerReply	Ні
replay_to message_id	Integer	Ні

chat_id – це поле містить інформацію про унікальний ідентифікатор основного чату або назви основного каналу; Photo – це поле містить інформацію про фото, яке потрібно відправити. Є два варіанта: перший – це можливість завантаження нової фотографії на сервер Telegram, або друга -це можливість передати, як рядок, file_id, для того щоб відправити фото яке вже є на сервері; Caption – це поле містить інформацію про тему фотографії, можна написати від 0 до 200 символів; Replay_markup – це поле містить інформацію про розширення можливостей для пошуку інтерфейсу. Це об’єкт, який являє собою JSON, тобто це вбудована клавіатура, є певна низка інструкцій для того щоб приховати клавіатуру, або потрібно примусово дати відповідь користувачеві; replay_to message_id – це поле містить інформацію про ідентифікатор вихідного повідомлення.

editMessageText – це метод, який потрібно використовувати для того щоб редагувати текстові повідомлення, які відправив бот, або які були відправленні завдяки йому. Параметри методу наведені в таблиці 3.5. [24].

Таблиця 3.5 – Параметри методу editMessageText [24]

Параметри	Тип	Обов'язковий
chat_id	Integer або String	Ні
inline_message_id	String	Ні
Text	String	Так
parse_mode	String	Ні

Продовження таблиці 3.5

Параметри	Тип	Обов'язковий
disable_web_page_ preview	Boolean	Hi
replay_markup	InlineKeyboard або ReplyKeyboard Hide	Hi

chat_id – цей параметр являється необхідним, тільки в тому випадку, якщо inline_message_id не вказано. Це поле містить інформацію про унікальний ідентифікатор основного чату або назви каналу; inline_message_id – це поле обов'язкове, тільки в тому випадку, якщо chat_id та message_id не зазначені. Це поле містить інформацію про ідентифкатор вбудованого повідомлення;
Text – це поле містить інформацію про новий текст повідомлення;
parse_mode - для того щоб в додатку Telegram відображався різний за стильом текст, такий як: напівжирний, або жирний, або курсивний в повідомленні бота, потрібно буде надіслати markdown or HTML;
disable_web_page_preview - відключає попередній перегляд посилань для посилань в цьому повідомленні;
replay_markup – це поле містить інформацію про розширення можливостей для пошуку інтерфейсу;

Користувач надає інформацію про себе в Telegram. Приклад наведено в таблиці 3.6 [24].

Таблиця 3.6 – Поля об’єкта User [24]

Поле	Тип
Id	Integer
first_name	String
last_name	String
Username	String

Id – це поле містить інформацію про унікальний ідентифікатор бота або користувача; first_name – це поле містить інформацію про ім'я бота або користувача; last_name – поле не є обов’язковим. Містить інформацію про прізвище бота або користувача; Username - поле не є обов’язковим. Містить інформацію про логін бота або користувача;

В об’єкті Chat міститься інформація про чат. Приклад наведено в таблиці 3.7 [24].

Таблиця 3.7 – Поля об’єкта Chat [24]

Поле	Тип
Id	Integer
Type	Enum
Title	String
Поле	Тип
Username	String
first_name	String
last_name	String
all_members_are_adminis trators	Boolean

Id - це поле містить інформацію про унікальний ідентифікатор чату. Значення не має перевищувати за 1e13; Type - це поле містить інформацію про тип чату такі як: приватний, група, супер група та канал; Title - поле не є обов’язковим. Містить інформацію про назву для каналів або груп; Username - поле не є обов’язковим. Містить інформацію про логін, для деяких чатів або каналів; first_name - поле не є обов’язковим. Містить інформацію про Username, для чатів і деяких каналів; last_name - поле не є обов’язковим. Містить інформацію про Прізвище співрозмовника в чаті;

all_members_are_adminis trators - поле не є обов’язковим. Якщо поле правдиве, то всі хто знаходяться в чаті, являються адміністраторами.

В об’єкті Message має міститись інформація про повідомлення. Приклад наведено в таблиці 3.8 [24].

Таблиця 3.8

Поле	Тип
message_id	Integer
From	User
Поле	Тип
Date	Integer
Chat	Chat
forward_from	User
forward_date	Integer
reply_to_message	Message
Text	String
Entities	Масив з MessageEntity
Document	Document
Photo	масив з PhotoSize
Sticker	Sticker
Video	Video
Voice	Voice
Caption	String
Contact	Contact
Location	Location
Venue	Venue

Продовження таблиці 3.8

Поле	Тип
new_chat_member	User
left_chat_member	User
new_chat_title	String
new_chat_photo	масив з PhotoSize
group_chat_created	True
supergroup_chat_created	True
channel_chat_created	True
migrate_to_chat_id	Integer
migrate_from_chat_id	Integer
Поле	Тип
pinned_message	Message

message_id – це поле містить інформацію про унікальний ідентифікатор повідомлення; From - поле не є обов'язковим. Містить інформацію про відправника. Може бути порожнім в каналах; Date - це поле містить інформацію про дату відправлення повідомлення (Unix time); Chat - це поле містить інформацію про діалог, в якому було відправлено повідомлення; forward_from - поле не є обов'язковим. Для повідомлень, які переслані, показує відправника оригінального повідомлення; forward_date - поле не є обов'язковим. Для повідомлень, які переслані, показує дату відправлення повідомлення; reply_to_message - поле не являється обов'язковим. Повідомлення з відповіддю містить оригінал повідомлення. Зверніть увагу на те, що об'єкт Message в даному випадку не буде, навіть, містити додаткові поля reply_to_message; Text - поле не являється обов'язковим. Інформація в текстовому повідомленні має вміститись в діапазон від 0 до 4096 символів; Entities - поле не є обов'язковим. Для текстових повідомлень: особливі суті в тексті повідомлення; Document - поле не є обов'язковим. Містить

інформацію про фото; Photo - поле не є обов’язковим. Містить інформацію про доступні розміри фото; Sticker - поле не є обов’язковим. Містить інформацію про стікери; Video - поле не є обов’язковим. Містить інформацію про відеозапис; Voice - поле не є обов’язковим. Містить інформацію про голосове повідомлення; Caption - поле не є обов’язковим. Містить інформацію про підпис до файлу, фото або відео, не має перевищувати 200 символів; Contact - поле не є обов’язковим. Містить інформацію про відправлений контакт; Location - поле не є обов’язковим. Містить інформацію про місцезнаходження; Venue - поле не є обов’язковим. Містить інформацію про місце на карті; new_chat_member - поле не є обов’язковим. Містить інформацію про користувача, доданого до групи; left_chat_member - поле не є обов’язковим. Містить інформацію про користувача, якого видалили; new_chat_title - поле не є обов’язковим. Назва групи було змінено на це поле; new_chat_photo - поле не є обов’язковим. Фото групи було змінено на це поле; group_chat_created - поле не є обов’язковим. Сервісне повідомлення: група створена; supergroup_chat_created - поле не є обов’язковим. Сервісне повідомлення: супергрупа створена; channel_chat_created - поле не є обов’язковим. Сервісне повідомлення: канал створений; migrate_to_chat_id - поле не є обов’язковим. Це поле надає інформацію про то що звичайна група була перероблена в супергрупу. Ідентифікатор не повинен перевищувати 1e13; migrate_from_chat_id - поле не являється обов’язковим. Це поле надає інформацію про те, що була створена супергрупа із звичаної групи. Ідентифікатор не повинен перевищувати 1e13; pinned_message - поле не є обов’язковим. Зазначене повідомлення було прикріплене. Зверніть увагу на те, що об’єкт Message в даному випадку не буде, навіть, містити додаткові поля reply_to_message;

В об'єкті KeyboardButton міститься ще одна кнопка для відповіді. Для текстових кнопок, даний об'єкт скріш за все буде змінено на рядок, який буде містити текст на кнопці. Приклад наведено в таблиці 3.9 [24].

Таблиця 3.9 – Поля об’єкта KeyboardButton [24]

Поле	Тип
Text	String
request_contact	Boolean
request_location	Boolean

Text – містить інформацію про текст, який знаходиться на кнопці. Якщо ніяке опціональне поле не використалось, то коли користувач нажме на кнопку, то боту воно буде відправлене звичайне повідомлення. request_contact – поле являється не обов’язковим. Якщо по умові значення правдиве, то коли користувач натискає на кнопку, боту повинен мав відправитись контакт того, хто натискав разом з його номером. Це фукнція можлива тільки в діалозі з ботом; request_location - поле є не обов’язковим. Якщо значення правдиве», то коли користувач натискає на кнопку, боту повинне надіслатись поточне розташування того, хто натискав. Дана функція можлива тільки в діалогах з ботом;

В об’єкті InlineKeyboardMarkup міститься інформація про вбудовану клавіатуру, яка буде з’являтись під певним повідомленням. Приклад наведено в таблиці 2.10 [24].

Таблиця 3.10 – Поля об’єкта InlineKeyboardMarkup [24]

Поле	Тип
inline_keyboard	Масив масивів з InlineKeyboardButton

inline_keyboard - Масив рядків, кожна з яких є масивом об'єктів InlineKeyboardButton.

Об’єкт типу InlineKeyboardMarkup це ще одна кнопка вбудованої клавіатури. Ми обов’язково повинні задіяти рівно одне опціональне поле. Поля типу наведені в таблиці 2.11 [24].

Таблиця 3.11 – Поля об’єкта InlineKeyboardButton [24]

Поле	Тип
Text	String
url	String
callback_data	String
switch_inline_query	String
switch_inline_query_current_chat	String
callback_game	CallbackGame

Text – містить інформацію про текст на кнопці; url – це поле є не обов’язковим. Містить інформацію про URL, який з’являється тільки коли натиснули на кнопку callback_data - це поле є не обов’язковим. Містить інформацію про дані, які будуть відправлятись в callback_query, коли корисутвач буде натискати на кнопку. switch_inline_query - це поле є не обов’язковим. Цей параметр означає можливість натискання на кнопку, і при цьому бота можна буде потім добавити в будь-який чат, спочатку бот запропонує відкрити потрібний користувачеві чат, а потім потрібно буде вписати його назву в поле вводу. switch_inline_query_current_chat - це поле є не обов’язковим. Ця функція означає, що якщо вона працює, то при натисканні на кнопку, введеться ім’я бота в поле вводу того чату в якому знаходиться зараз користувач. callback_game - це поле є не обов’язковим. Містить інформацію про описання гри, яка запуститься при натисканні на кнопку.

3.2 Розробка бази даних

Проектування бази даних було реалізовано на основі реляційної моделі. В результаті було отримано наступну базу даних (див. додаток А)

В таблиці 3.12 було реалізовано забезпечення збереження інформації про вибір користувача.

Таблиця 3.12- tabletbot_choise_user

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(2048)	Той хто видалив запис
choise	character varying(50)	Вибір користувача

В таблиці 3.13 було реалізовано забезпечення збереження інформації про виробника препаратів.

Таблиця 3.13– tabletbot_manufacture

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис

Продовження таблиці 3.13

Назва	Тип	Опис
name	character varying(2048)	Назва

В таблиці 3.14 було реалізовано забезпечення збереження інформації про форму випуску препаратів.

Таблиця 3.14– tabletbot_release_form

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
name	character varying(1024)	Назва

В таблиці 3.15 було реалізовано забезпечення збереження інформації про країну виробника препаратів.

Таблиця 3.15– tabletbot_country

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє

Продовження таблиці 3.15

Назва	Тип	Опис
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
name	character varying(2048)	Назва

В таблиці 3.16 було реалізовано забезпечення збереження інформації про фаракотерапевтичну групу.

Таблиця 3.16– tabletbot_pharmacotherapeutic_group

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
name	character varying(5096)	Назва

В таблиці 3.17 було реалізовано забезпечення збереження інформації про тип препаратів.

Таблиця 3.17– tabletbot_tablet

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис

Продовження таблиці 3.17

Назва	Тип	Опис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис

В таблиці 3.18 було реалізовано забезпечення збереження інформації про виготовлення препаратів.

Таблиця 3.18– tabletbot_drug_type

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
name	character varying(2048)	Назва

В таблиці 3.19 було реалізовано забезпечення збереження інформації про країну виробника препаратів.

Таблиця 3.19– tabletbot_expiration_date_type

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія

Продовження таблиці 3.19

Назва	Тип	Опис
create_ts	timestamp	Початок створення запису
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
name	character varying(2048)	Назва

В таблиці 3.20 було реалізовано забезпечення збереження реляційного відношення.

Таблиця 3.20– tabletbot_preparation_prepareation_link

Назва	Тип	Опис
preparation_1_id	uuid	Унікальний ідентифікатор
preparation_2_id	uuid	Унікальний ідентифікатор

В таблиці 3.21 було реалізовано забезпечення збереження інформації повного описання препарату.

Таблиця 3.21– tabletbot_preparation

Назва	Тип	Опис
id	uuid	Унікальний ідентифікатор
version	integer	Версія
create_ts	timestamp	Початок створення запису

Продовження таблиці 3.21

Назва	Тип	Опис
created_by	character varying(50)	Той хто створив запис
update_ts	timestamp	Початок оновлення запису
updated_by	character varying(50)	Той хто редагував востаннє
delete_ts	timestamp	Початок видалення запису
deleted_by	character varying(50)	Той хто видалив запис
id_driz	character varying(64)	Унікальний ідентифікатор у державному реєстрі
trade_name	character varying(2048)	Торгова назва
vacation_conditions	character varying(2048)	Умови відпустки
drug_composition	character varying(5096)	Лікарський склад
manufacture_id	uuid	Унікальний ідентифікатор
country_id	uuid	Унікальний ідентифікатор
address_manufacture	character varying(2048)	Адреса виробництва
registration_certificate_number	character varying(1024)	Номер свідоцтва про реєстрацію

Продовження таблиці 3.21

Назва	Тип	Опис
drug_type_id	uuid	Унікальний ідентифікатор
biological_origin	boolean	Чи являється біологічного походження
plant_origin	boolean	Чи являється рослинного походження
orphan	boolean	Чи це препарат сирота
homeopathic	boolean	Чи це гомеопатичний препарат
instruction_date	character varying(2048)	Дата інструкції
expiration_date	integer	Термін придатності
expiration_date_type_id	uuid	Унікальний ідентифікатор
realese_form_id	uuid	Унікальний ідентифікатор
pharmacotherapeutic_group_id	uuid	Унікальний ідентифікатор

3.3 Результат роботи бота

В цьому розділі відображено результат роботи бота. На скріншотах буде показано більш детально:

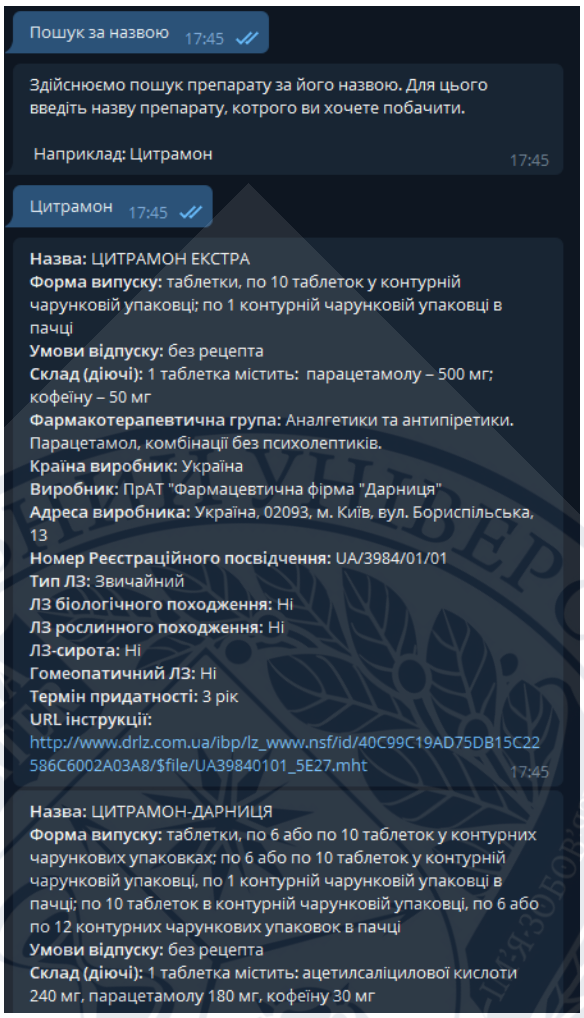

Рисунок 3.2 – Принцип пошуку за назвою препарату

На даному скріншоті (рис. 3.2) ми бачимо як реалізовано принцип пошуку препарату за його назвою. На рисунку ми можемо побачити назву препарату, форму випуску, умови відпуску. склад, фармакотерапевтичну групу, країну виробника, виробника, адресу виробника, номер реєстраційного посвідчення, тип лікарського засобу, чи являється лікарський засіб біологічного походження, чи являється рослинного походження, чи являється сиротою, чи являється гомеопатичним, термін придатності, а також інструкцію.

Бот демонструє 5 прикладів препаратів, які мають однакову назву, але відрізняються складом, якщо кількість таких препаратів більше за 5, то є можливість натиснуту кнопку «Наступні 5 препаратів»(рис.3.3)

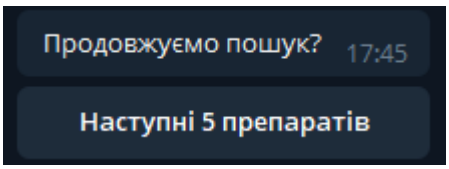

Рисунок 3.3 – кнопка «Наступні 5 препаратів»

В Telegram бота є можливість підбору лікарського засобу за діючою речовиною. Даний спосіб пошуку реалізований на рисунку 3.4.

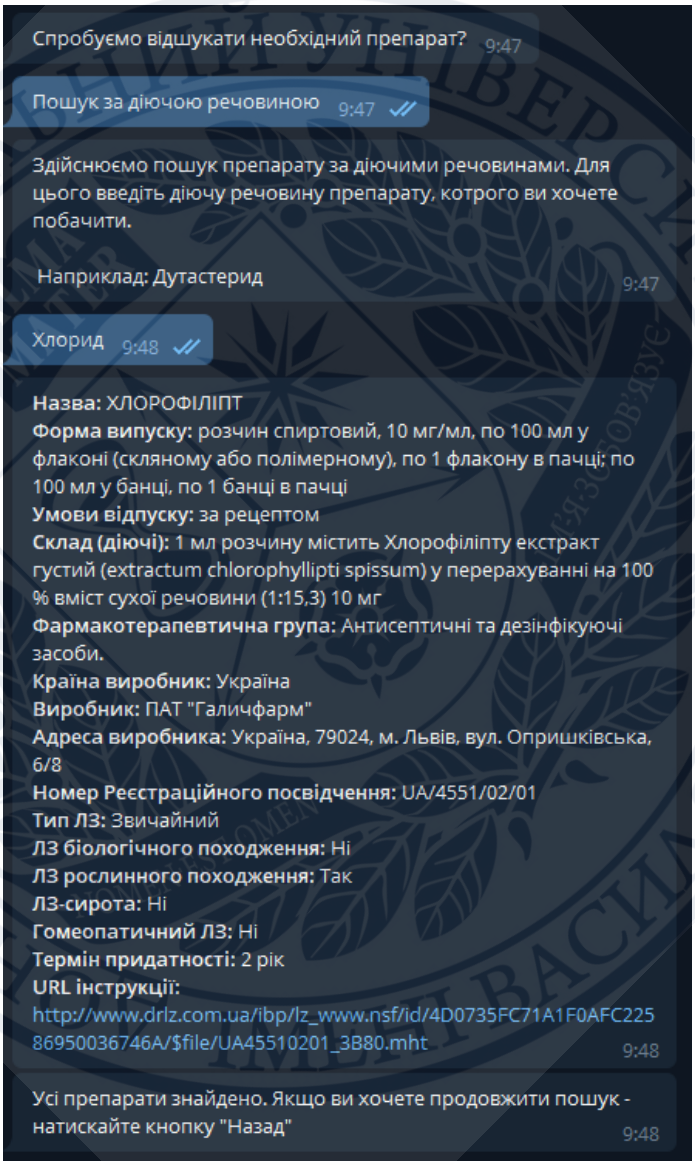

Рисунок 3.4 – Пошук препарату за діючими речовинами

В боті реалізовано спосіб пошуку аналогів препаратів. Якщо користувач захоче купити певні ліки, є можливість подивитись які є схожі препарати за складом, але відрізняють назвою. Приклад на рисунку 3.6.

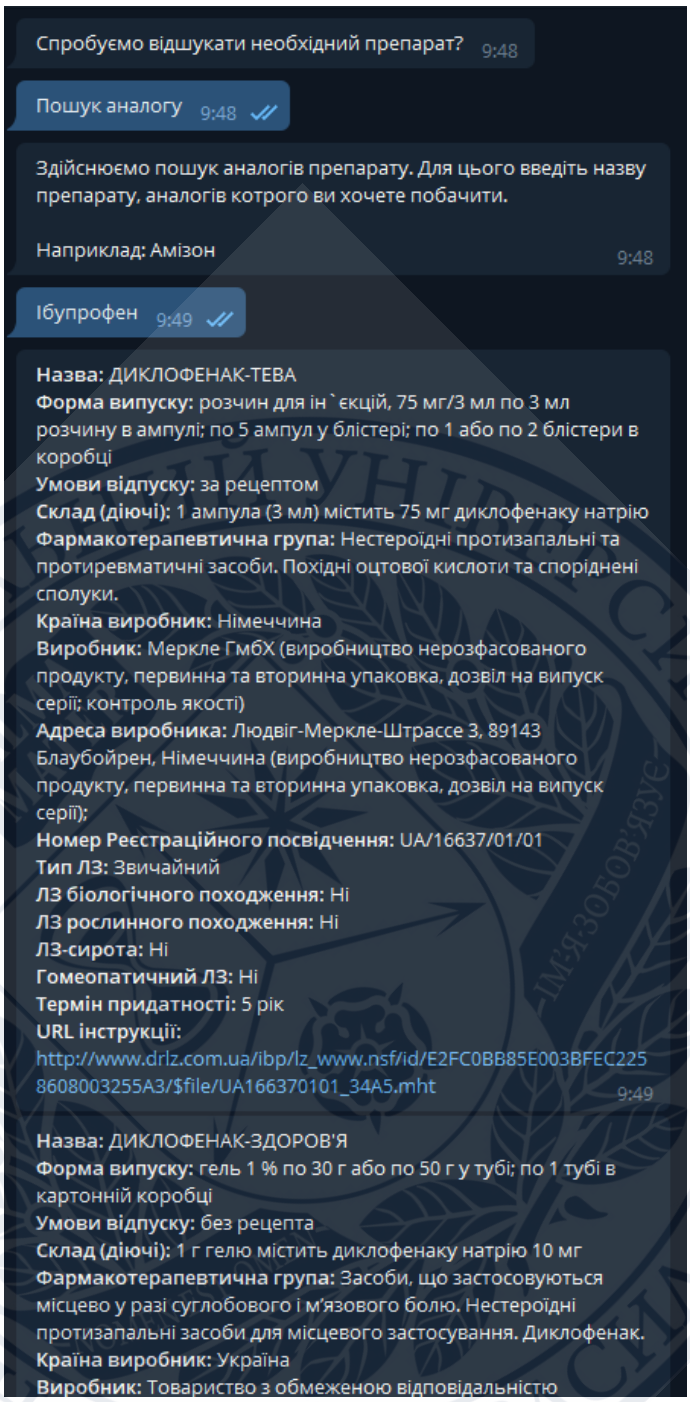

Рисунок 3.6 – Пошук аналога препарата

Якщо користувачу буде незрозуміло як користуватись ботом, то потрібно ввести команду /help, і буде продемонстровано як зробити пошук препарату за його назвою.

Всі інші функції основані за таким самим принципом, натискаєм кнопку «пошук» і вводимо те що потрібно. Продемонстровано на рисунку 3.7.

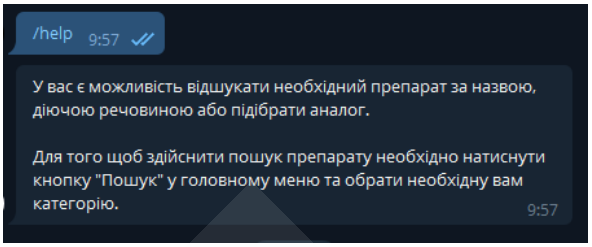

Рисунок 3.7– Команда допомоги

Висновки до розділу 3.

В третьому розділі було розглянуто та детально описано про монолітну архітектуру. Було більш детально описано про базу даних. Було продемонстровано результат та принцип роботи бота. Також було детально розписано про принцип роботи з Telegram API, та використання його для створення бота.

ВИСНОВОК

В ході виконання бакалаврської роботи було досліджено особливості пошуку медичних препаратів та їх аналогів. Проведено аналізування переваг та недолік аналогів телеграм боту. Здійснено огляд інструментів які використовуються при розробці телеграм ботів.

Використовуючи отримані результати здійснено побудову комп'ютерно математичної моделі. Для вирішення задачі пошуку препаратів було сформовано локальну базу даних, яка базується на відкритих джерелах.

Вивчено інфраструктура Cuba Platform, бібліотека EclipseLink, база даних PostgreSQL, REST архітектура.

Реалізовано веб-додаток на основі REST архітектури, що здійснює взаємодію із Telegram API. Також сам Telegram бот має зручний користувацький інтерфейс.

Завдання було виконано в повному обсязі.

СПИСОК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Важливість Telegram бота [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/WnqAu5B - Дата звернення 20.05.2021
2. Сфери використання Telegram бота [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/TnqASfc - Дата звернення 20.05.2021
3. Особливості Telegram [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/wnt2kt3 - Дата звернення 25.05.2021
4. Python [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/dnqY2c0 - Дата звернення 24.05.2021
5. Про мову Python [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/8nqUO21 - Дата звернення 24.05.2021
6. Короткі відомості про мову C# [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/6nqPI0z - Дата звернення 24.05.2021
7. Рейтинг мов програмування [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/xnqDBPt - Дата звернення 24.05.2021
8. Мова програмування Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/bnqDQc1 - Дата звернення 24.05.2021
9. Відомості про Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/xnqGnpk - Дата звернення 24.05.2021
10. Відомості про фреймворк Cuba Platform [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/6nqKJpx - Дата звернення 24.05.2021
11. Екосистема Cuba Platform [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/enqZRY3 - Дата звернення 24.05.2021
12. Засоби розробки Cuba Platform [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/8nqZz5M - Дата звернення 24.05.2021
13. Набір функцій Cuba Platform [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/lnqCxra - Дата звернення 24.05.2021

14. База даних PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/snqMw1A - Дата звернення 24.05.2021
15. Функції PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/fnqM2OW - Дата звернення 24.05.2021
16. Детально про Telegram API [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/nnwjYCS - Дата звернення 24.05.2021
17. Плюси Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/XnwFhNb - Дата звернення 25.05.2021
18. Плюси PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/IneE9AJ - Дата звернення 25.05.2021
19. Монолітна архітектура [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/ZnabWja - Дата звернення 27.05.2021
20. Платформа CUBA [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/LnaIoZz - Дата звернення 27.05.2021
21. Роберт Мартин. Чистый код. Создание, анализ и рефакторинг. Перевод на русский язык ООО Издательство «Питер», 2015. 464 с.
22. Саймон Ригс, Ханну Кросинг. Администрирование PostgreSQL 9. Книга рецептов. Оформление, перевод на русский язык ДМК Пресс, 2013. 366 с.
23. Ханс-Юрген Шониг. Mastering PostgreSQL 12: Advanced Techniques to Build and Administer Scalable and Reliable PostgreSQL Database Applications, 3rd Edition. Published by Packt Publishing Ltd. 2018. 443 с.
24. Telegram API [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/tnwfkGh - Дата звернення 24.05.2021
25. Nicolas Modrzyk. Building Telegram Bots. Издатель Apress, 2018. 277 с.
26. Офіційний сайт Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/9ndylKf - Дата звернення 27.05.2021
27. Типи даних Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/2ndyQDF - Дата звернення 27.05.2021

28. Плюси Cuba Platform (Jmix) [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/fndyVDo - Дата звернення 27.05.2021
29. Telegram API [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/undy18R - Дата звернення 27.05.2021
30. Telegram APIs [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/ondy5JE - Дата звернення 27.05.2021
31. Telegram API Methods [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/gndueIg - Дата звернення 27.05.2021
32. Telegram API Bots [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/HnduHJE - Дата звернення 27.05.2021
33. Інструкція по Bot API [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/9ndylKf - Дата звернення 27.05.2021
34. Telegram бот на Java [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/8nduX8j - Дата звернення 27.05.2021
35. Створення Telegram бота [Електронний ресурс] – Режим доступу до ресурсу: https://cutt.ly/HnduBYU - Дата звернення 27.05.2021
36. Герберт Шилдт. Java: Полное руководство. Издатель Вильямс, 2012. 1101 с.
37. Джошуа Блох. Effective Java. Издатель Pearson Education, 2016. 265 с.
38. Васильев Алексей Николаевич. Java для всех. Издательский дом «Питер», 2019. 512 с.
39. Валерий Романчик, Игорь Блинов. Java. Методы программирования. Издатель Litres, 2017. 895 с.
40. Хабибулин Ильдар Шаукатович. Java. Издатель БХВ-Петербург, 2012. 768 с.

Додаток А

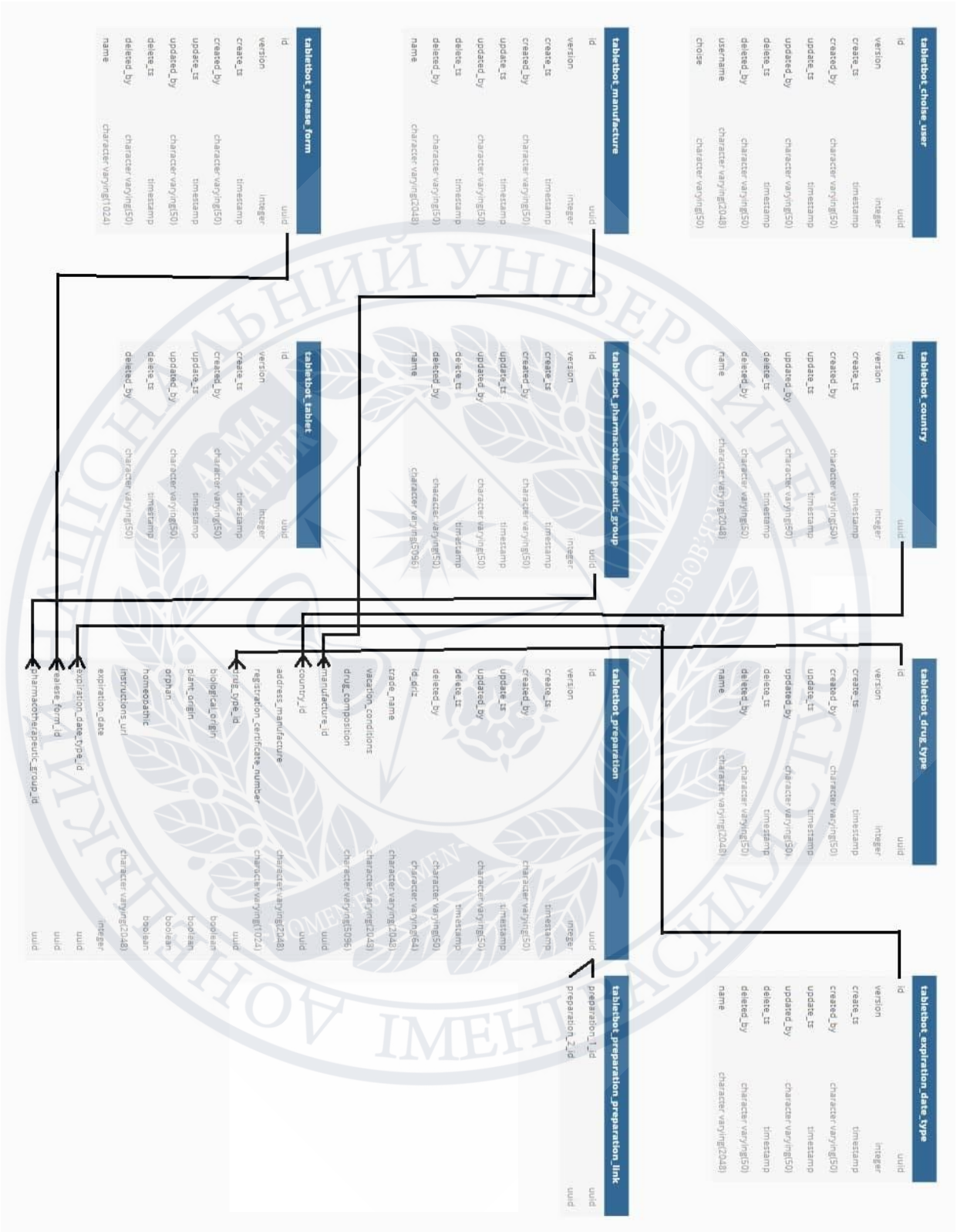

Рисунок – реляційна модель БД