ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Розробка чат-боту для поліпшення зручності та доступності інформації про громадський транспорт»

(тема кваліфікаційної роботи українською мовою)

«Development of a chatbot to improve the convenience and availability of information about public transport»

(тема кваліфікаційної роботи англійською мовою)

Виконала: здобувачка заочної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Тран Ванесса Ванівна

(прізвище, ім'я, по-батькові здобувача)

Керівник д.т.н., професор Казакова Н.Ф. _____

(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент Корчемний П.А.

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .
№ 1 від 09 червня 2024 р.

Завідувачка кафедри

_____ КАЗАКОВА Надія .

(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № 13 .
протокол № 8 від 19 червня 2024 р.

Оцінка добре / С / 80 .

(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

_____ КОПИЧЕНКО Іван .

(підпис) (прізвище,ім'я)

Одеса 2024

ЗМІСТ

Терміни, скорочення та умовні познаки 5
Вступ 7
1 Аналіз існуючих технологій та рішень у сфері чат-ботів 9
1.1 Аналіз платформ для розробки чат-ботів в телеграм 19
1.2 Аналіз існуючих рішень в галузі інформаційного забезпечення громадського транспорту в телеграмі 22
1.3 Характеристика тенденцій та інновацій у розвитку чат-ботів для громадського транспорту в телеграмі 23
2 Визначення вимог до функціональності чат-бота 25
3 Розробка бази даних для чат-боту 33
4 Реалізація чат-боту для оптимізації доступу до інформації про громадський транспорт 43
Висновки 49
Перелік джерел посилання 51

ТЕРМІНИ, СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

База даних – це організована колекція даних, яка зберігається та управляється комп'ютерною системою таким чином, щоб дозволити легкий доступ, оновлення та управління цими даними.

Месенджер – це програмне забезпечення або платформа, яка дозволяє користувачам обмінюватися повідомленнями в режимі реального часу через Інтернет.

Операційна система – це програмне забезпечення, що управляє апаратним забезпеченням комп'ютера та забезпечує основні сервіси для інших програм.

Система управління базами даних – це програмне забезпечення для створення та управління базами даних, яке надає можливість зберігати, організовувати та отримувати доступ до даних, збережених у базі даних.

Таблиця (в базах даних) – це структура даних, що організована у вигляді рядків та стовпців, де кожен рядок представляє собою окремий запис, а кожен стовпець – певну характеристику або атрибут даних.

Фреймворк – це комплект інструментів, бібліотек і загальної структури, які надають базові засоби для розробки програмного забезпечення або побудови певного типу програм або сервісів.

Чат-бот (або просто бот) – це програмний агент, який автоматизує взаємодію з користувачем або іншою системою через текстові чати, голосові команди або інші інтерфейси.

API – це набір правил та протоколів, які визначають, які функції або сервіси можуть бути використані для взаємодії між різними програмами чи компонентами програмного забезпечення.

C# – це об'єктно-орієнтована мова програмування, розроблена Microsoft.

Foreign key – це обмеження в базі даних, яке вказує на те, що значення у стовпці (або комбінація стовпців) таблиці повинні посилатися на значення

у стовпці (або комбінації стовпців) іншої таблиці. Це створює зв'язок між двома таблицями і дозволяє виконувати операції зв'язаних даних.

Java – це об'єктно-орієнтована мова програмування, яка використовується для розробки різноманітних програм, від веб-додатків до мобільних додатків та корпоративних систем.

JavaScript – це мова програмування, яка використовується для створення інтерактивних веб-сайтів.

Natural Language Processing (Обробка природної мови) – це галузь штучного інтелекту, яка займається розумінням та обробкою людської мови з використанням комп'ютерних алгоритмів.

Primary key – це унікальний ідентифікатор для кожного рядка у таблиці бази даних. Він гарантує, що значення цього стовпця у кожному рядку унікальні, і може бути використане для однозначного ідентифікування рядка.

Python – це високорівнева мова програмування загального призначення, яка відзначається простотою синтаксису та читабельністю коду.

SQL-ін'єкції – використання введених даних для виклику SQL-запитів та отримання несанкціонованого доступу до бази даних.

БД – база даних.

ІС – інформаційна система.

ОС – операційна система.

СУБД – система управління базами даних.

API – application programming interface.

NLP – Natural Language Processing.

UML – unified modeling language.

ВСТУП

Сучасний ритм життя вимагає постійної мобільності та ефективного використання часу, особливо в умовах міського середовища, де громадський транспорт стає невід’ємною частиною щоденного життя. Однак, не зважаючи на те, що транспортні системи розвиваються, проблеми, пов'язані з доступністю та ефективністю інформації про громадський транспорт, залишаються нагальними. У багатьох містах світу пасажири стикаються з труднощами у плануванні свого маршруту, невірною інформацією про графік руху транспорту, а також нечіткістю щодо можливих затримок чи змін у роботі маршрутів. Це може призводити до втрати часу, стресу та незручностей для користувачів громадського транспорту.

Метою кваліфікаційної роботи бакалавра є комплексне вирішення проблеми оптимізації транспорту за допомогою ІТ, а саме розробка та впровадження чат-бота, спрямованого на поліпшення зручності та легкості використання системи громадського транспорту. Застосування інформаційних технологій для створення зручного та доступного інтерфейсу взаємодії з пасажирами сприятиме підвищенню рівня сервісу та задоволеності користувачів громадського транспорту.

Актуальність теми полягає в необхідності покращення якості обслуговування пасажирів та оптимізації використання громадського транспорту за допомогою сучасних технологій.

Завдання роботи включають в себе аналіз існуючих технологій та рішень у сфері чат-ботів, визначення вимог до функціональності чат-бота, розробка та тестування прототипу, оптимізація та впровадження.

Дана робота визначається не лише технічною інноваційністю, але й потенційно значущим внеском у покращення якості та зручності використання громадського транспорту, що, в свою чергу, може сприяти розвитку сталого міського транспорту та підвищенню комфорту життя громадян.

Потенційні переваги від впровадження чат-бота:

– економія часу пасажирів (швидкий пошук потрібної інформації про розклад, маршрути, затримки безпосередньо в чаті);
– зменшення навантаження на call-центри компаній (автоматизація відповідей на типові питання через чат-бота);
– автоматизація обробки типових запитів (швидке надання стандартної інформації без залучення оператора);
– збір зворотного зв'язку від пасажирів (можливість залишити відгук, повідомити про проблему через чат-бота).

Telegram є одним з найпопулярніших месенджерів в Україні. Він має зручний інтерфейс та можливості – швидкий обмін повідомленнями, наявність клавіатур тощо. Відкритий API для створення ботів – дозволяє інтегрувати чат-бота в месенджер. Клієнти Telegram доступні на різних пристроях та ОС. Однією з ключових переваг Telegram є високий рівень безпеки та захисту даних користувачів. Це особливо важливо для сервісу, який планується використовувати для надання інформації про пересування громадським транспортом.

Дана кваліфікаційна робота бакалавра складається з 52 сторінок, 12 рисунків, 1 таблиця та 16 джерел посилання.

1 АНАЛІЗ ІСНУЮЧИХ ТЕХНОЛОГІЙ ТА РІШЕНЬ У СФЕРІ ЧАТ-БОТІВ

Аналіз існуючих технологій та рішень у сфері чат-ботів включає вивчення різних підходів, платформ, фреймворків та інструментів, що використовуються для розробки та впровадження чат-ботів.

Популярні платформи для створення чат-ботів, такі як Dialogflow від Google, Microsoft Bot Framework, Amazon Lex, IBM Watson Assistant та інші.

Dialogflow від Google [1] надає широкі можливості для створення розумних чат-ботів, включаючи обробку природної мови (NLP), створення діалогових сценаріїв, інтеграцію з іншими сервісами Google, такими як Google Calendar, Google Sheets тощо. Підтримується інтеграція з різними платформами месенджерів, такими як Facebook Messenger, Telegram, Skype, а також можливість інтеграції з власними веб-сайтами через веб-хуки та API. Підтримує різні мови програмування, такі як JavaScript, Python, Java, Node.js тощо.

Microsoft Bot Framework дозволяє створювати чат-ботів для різних платформ, включаючи Skype, Slack, Facebook Messenger та інші. Вона надає інструменти для розробки, тестування та розгортання ботів. Інтегрується з Azure, що дозволяє легко розгортати та керувати ботами в хмарі. Також підтримує інтеграцію з багатьма іншими сервісами Microsoft та сторонніми API. Підтримується різні мови програмування, такі як C#, JavaScript, Python [2].

Amazon Lex – це послуга від Amazon Web Services (AWS) для розробки чат-ботів з функціями розпізнавання мови, обробки природної мови та управління діалогами [3]. Легко інтегрується з іншими сервісами AWS, такими як Lambda, S3, DynamoDB, а також підтримує інтеграцію з іншими зовнішніми сервісами через API. Підтримується різні мови програмування, такі як JavaScript, Python, Java, Node.js тощо.

IBM Watson Assistant – це інтелектуальна платформа, яка дозволяє створювати розумних чат-ботів з функціями розпізнавання мови, аналізу тональності, генерації мови тощо. Легко інтегрується з іншими сервісами IBM

та сторонніми системами через API та веб-хуки. Підтримується різні мови програмування, такі як Python, Node.js, Java тощо [4].

Також слід звернути увагу на аналіз відкритих фреймворків та бібліотек для розробки чат-ботів, таких як Rasa, Botpress, Microsoft Bot Framework SDK, Python NLTK, SpaCy та інші.

Rasa надає рішення для розпізнавання мови та обробки природної мови, що дозволяє розробляти чат-ботів зі складною логікою взаємодії. Хоча Rasa є потужним фреймворком, він може бути складним у використанні для новачків, але має високий потенціал для тих, хто оволодів його функціоналом. Rasa має активну спільноту розробників, яка регулярно вносить оновлення та підтримує фреймворк.

Botpress дозволяє побудову складних чат-ботів та розпізнавання мови, забезпечуючи широкий спектр можливостей для створення різноманітних ботів. Цей фреймворк має графічний інтерфейс, що робить його досить легким для використання, навіть для початківців. Botpress також має активну спільноту розробників, яка забезпечує оновлення та підтримку фреймворку.

Microsoft Bot Framework SDK – цей фреймворк SDK надає можливості для створення та управління ботами, а також для розпізнавання мови та інших функцій. Рекомендований для розробників на платформі .NET, Microsoft Bot Framework SDK може бути легким у використанні для тих, хто володіє знаннями цієї платформи. Цей фреймворк підтримується компанією Microsoft, що гарантує стабільну роботу та оновлення.

Python NLTK та SpaCy, ці бібліотеки для мови Python дозволяють проводити обробку природної мови та аналіз тексту в чат-ботах. NLTK та SpaCy мають простий інтерфейс, що робить їх легкими для використання, зокрема для розробників на Python. Ці бібліотеки мають активні спільноти розробників, які постійно вносять оновлення та підтримують їх.

Також роль відіграє мова програмування, тому одну з ключових оглядів потрібно приділити мовам програмування, які часто використовуються

для розробки чат-ботів, таких як Python, JavaScript (Node.js), Java, C#, і їхні переваги та недоліки для даної задачі [5].

Перевагами використання Python є простота вивчення та використання, широкий вибір бібліотек, підтримка штучного інтелекту. Python має простий синтаксис та багато вбудованих бібліотек для обробки природної мови. Є багато сторонніх бібліотек, таких як NLTK, SpaCy, Rasa, які полегшують розробку чат-ботів. Python має багато інструментів для реалізації штучного інтелекту, що корисно для розвитку інтелектуальних чат-ботів. До недоліків, слід віднести швидкодію. Python може бути повільним порівняно з іншими мовами програмування, особливо у випадках великого обсягу обробки даних.

Перевагами використання JavaScript (Node.js) є швидкодія, асинхронність. Node.js використовує JavaScript, що дозволяє створювати ефективні та швидкодіючі серверні застосунки. Можливість робити асинхронні запити до сервера дозволяє обробляти багато запитів одночасно, що важливо для чат-ботів з великою кількістю користувачів. Одним з основних недоліків є складність вивчення. JavaScript може бути складним для новачків, особливо коли мова використовується на серверній стороні.

Перевагами використання Java є надійність, широка екосистема. Java відома своєю надійністю та масштабованістю, що робить її відмінним вибором для великих проєктів чат-ботів. Java має велику кількість бібліотек та фреймворків, що полегшують розробку різних функціональних можливостей. До недоліків, слід віднести складність. Java може бути складнішою для розробки простих чат-ботів порівняно з іншими мовами програмування.

Перевагами використання C# є інтеграція зі стеком Microsoft та широка підтримка. C# ідеально підходить для розробки чат-ботів для платформи Microsoft, таких як Microsoft Bot Framework. C# має велику спільноту розробників та документацію, що полегшує розробку. До недоліків, слід віднести обмеженість платформи. C# обмежений платформою Microsoft, що може ускладнити інтеграцію з іншими платформами та сервісами.

Слід звернути увагу на інтеграцію з платформами месенджерів. Розгляд можливостей інтеграції чат-ботів з різними популярними платформами месенджерів, такими як Telegram, Facebook Messenger, WhatsApp, Slack, Viber тощо. В тому числі огляд API та можливостей цих платформ для створення та взаємодії з чат-ботами.

Інтеграція з платформами месенджерів є ключовим аспектом розробки чат-ботів, оскільки це визначає, на яких платформах користувачі зможуть взаємодіяти з ботом.

Telegram надає API Telegram Bot (рис.1) для створення та взаємодії з ботами. За допомогою цього API розробники можуть створювати ботів, налаштовувати їхню поведінку та відправляти повідомлення користувачам. Боти Telegram можуть спілкуватися з користувачами через повідомлення, кнопки, відповіді на команди та інші взаємодії [6].

Рисунок 1 – API Telegram Bot

Facebook надає Messenger Platform [7] для створення та взаємодії з чат-ботами. Розробники можуть створювати ботів, які взаємодіють з користувачами через повідомлення, картинки, кнопки та інші елементи. Боти можуть бути інтегровані зі сторінками та групами Facebook, що дозволяє використовувати їх для спілкування зі широкою аудиторією (рис. 2).

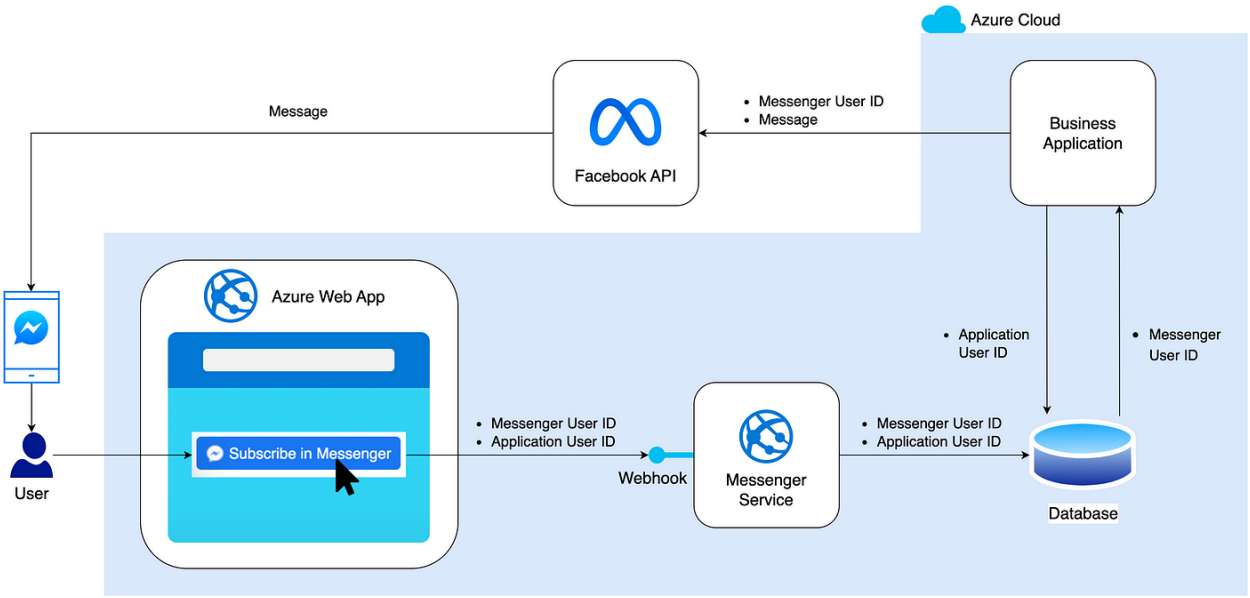

Рисунок 2 – Messenger Platform API

WhatsApp надає Business API для створення чат-ботів для підприємств. Цей API дозволяє створювати ботів, які взаємодіють з клієнтами через повідомлення та інші функції. Боти WhatsApp можуть надсилати повідомлення клієнтам, надавати їм інформацію про товари та послуги, приймати замовлення та інше (див.рис. 3) [8].

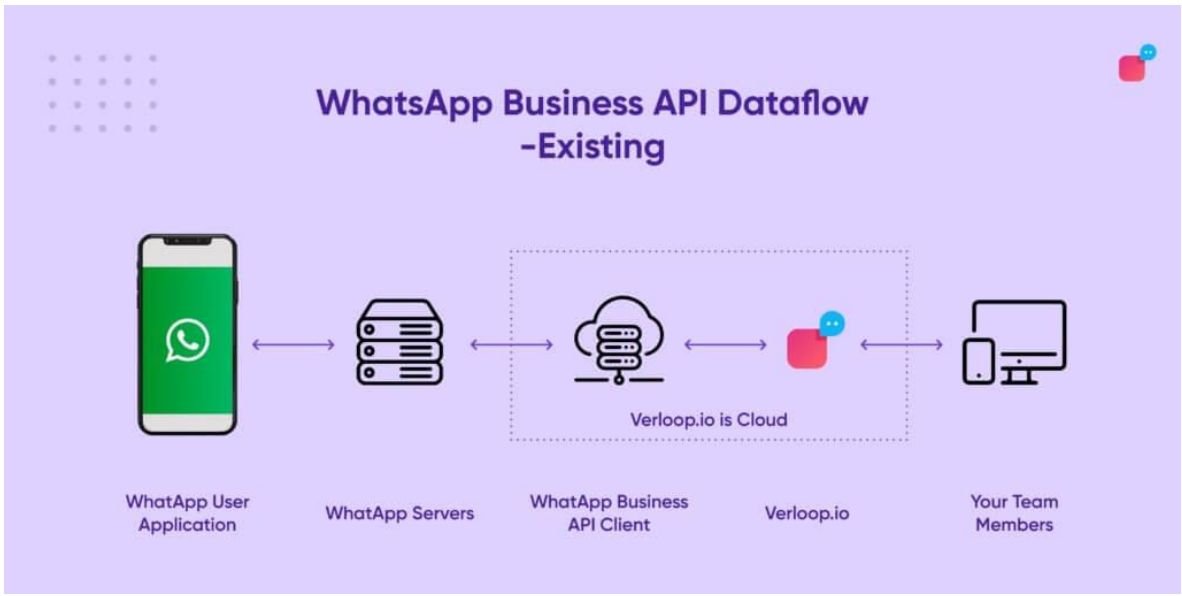

Рисунок 3 – WhatsApp Business API

Slack надає API для створення чат-ботів (рис. 4), які можуть взаємодіяти з користувачами через повідомлення, команди та інші функції [9]. Боти Slack можуть бути інтегровані в робочі простори, що дозволяє їм спілкуватися з командою та автоматизувати рутинні завдання.

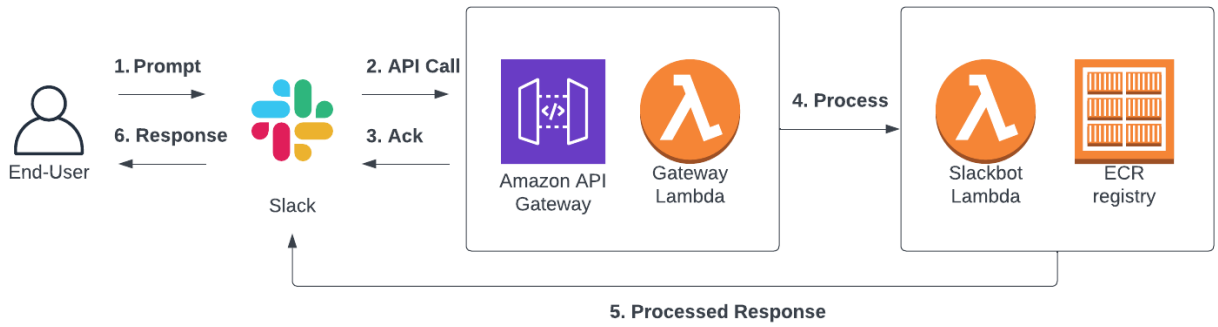

Рисунок 4 – Slack API

Viber надає REST API (див.рис. 5) для створення ботів, які можуть взаємодіяти з користувачами через повідомлення, стікери, кнопки та інші функції. Боти Viber можуть надсилати повідомлення користувачам, обробляти їхні запити та надавати інформацію [10].

Рисунок 5 – Viber REST API

Кожна з цих платформ має свої унікальні можливості та API для створення та взаємодії з чат-ботами, що дозволяє створювати ботів для різних цілей та аудиторій.

Аналіз інтелектуальних технологій у контексті розробки чат-ботів є важливою частиною процесу створення ефективних та функціональних систем. Обробка природної мови (NLP) використовується для розуміння та обробки природної мови, що дозволяє чат-ботам розпізнавати, аналізувати та генерувати текстові дані. Методи NLP можуть бути використані для розпізнавання інтентів користувачів, витягування ключової інформації з тексту, створення відповідей та багато іншого.

Машинне навчання дозволяє чат-ботам навчатися на основі великої кількості даних та розвивати більш складні та персоналізовані функції. Застосування методів машинного навчання може покращити розпізнавання інтентів, уточнення відповідей, визначення контексту та інші аспекти взаємодії з користувачем.

Технології генерації мови можуть використовуватися для автоматичного створення текстового контенту, відповідей та повідомлень чат-бота. Це може бути корисно для створення персоналізованих повідомлень, створення нового контенту або автоматизації відповідей на запитання.

Технології обробки зображень можуть допомогти чат-ботам аналізувати та розпізнавати зображення, що надходять від користувачів. Це може використовуватися для автоматичного розпізнавання об'єктів, аналізу контенту зображення та надання відповідей на запитання.

Застосування цих інтелектуальних технологій може значно покращити функціональність та ефективність чат-ботів, роблячи їх більш інтуїтивно зрозумілими та корисними для користувачів.

Що стосується безпеки та конфіденційності необхідно проаналізувати методи та технології для забезпечення безпеки та конфіденційності в чат-ботах, це є критичним у забезпеченні захисту користувачів та їхніх даних.

Чат-боти повинні бути захищені від різних видів атак, таких як SQL-ін'єкції, введення команд, переповнення буфера та інші. А також слід використовувати валідації даних, параметризованих запитів, контролю доступу та інших заходів може допомогти у запобіганні атак.

Для забезпечення конфіденційності даних, які передаються між користувачами та чат-ботом, можна використовувати шифрування. Застосування шифрування TLS/SSL для захисту комунікації та шифрування даних у спокої може забезпечити конфіденційність інформації.

Чат-боти можуть використовувати різні методи аутентифікації користувачів, такі як авторизація через пароль, двофакторна аутентифікація або використання токенів доступу. Аутентифікація дозволяє переконатися, що лише авторизовані користувачі мають доступ до функцій чат-бота та їхніх даних.

Важливим аспектом забезпечення безпеки є моніторинг діяльності чат-бота та журналювання подій для виявлення потенційних загроз та відстеження дій користувачів. Моніторинг дозволяє оперативно реагувати на можливі проблеми та інциденти безпеки.

Підтримка та дотримання безпечних стандартів розробки, таких як OWASP Top 10 та інших, є важливим для забезпечення безпеки чат-ботів.

Забезпечення безпеки та конфіденційності в чат-ботах є невід'ємною частиною їхнього розроблення та експлуатації, і вимагає системного підходу та використання найкращих практик у цій сфері.

Огляд сучасних тенденцій та інновацій у сфері чат-ботів допомагає розуміти поточні тенденції та майбутні перспективи розвитку цієї технології. Нижче представлено деякі з найбільш значущих тенденцій та інновацій:

– голосові асистенти (чат-боти, які взаємодіють з користувачами за допомогою голосового інтерфейсу, стають все більш популярними. Вони дозволяють користувачам надсилати голосові команди для виконання певних завдань);
– чат-боти зі штучним інтелектом (AI) (розвиток технологій штучного інтелекту дозволяє чат-ботам розуміти та адаптуватися до контексту розмови, виконувати більш складні завдання та надавати більш персоналізовані відповіді);

- використання біг даних (великі дані (Big Data) дозволяють чат-ботам аналізувати та використовувати великі обсяги даних для покращення функціональності та якості відповідей);
- мультиканальність (сучасні чат-боти надають можливість взаємодії з користувачами через різні канали зв'язку, такі як веб-сайти, месенджери, голосові застосунки тощо);
- персоналізація та контекстуалізація (чат-боти стають все більш персоналізованими та можуть адаптуватися до потреб кожного користувача, враховуючи його історію взаємодії та контекст розмови);
- забезпечення конфіденційності (розвиток технологій шифрування та інших методів безпеки дозволяє забезпечити конфіденційність даних користувачів у чат-ботах);
- розширені функціональні можливості (чат-боти набувають розширені функціональні можливості, такі як інтеграція з платіжними системами, робота з великими обсягами даних, надання консультацій тощо).

Ці тенденції та інновації визначають сучасний розвиток технології чат-ботів і впливають на їхнє використання та роль у різних галузях та сферах діяльності.

Під час огляду сучасних тенденцій та інновацій у сфері чат-ботів, важливо врахувати роль месенджерів у розповсюдженні та використанні чат-ботів. Однією з найпопулярніших платформ для чат-ботів є Telegram, який має простий та інтуїтивно зрозумілий інтерфейс, доступний для широкого кола користувачів. Вбудована підтримка багатьох корисних функцій, таких як боти та анонімні чати, робить його популярним серед користувачів. Також, Telegram активно розвивається і регулярно випускає оновлення з новим функціоналом, що робить його привабливим для використання. Однак, слід зазначити, що через його анонімність Telegram іноді критикують за можливе використання для незаконної діяльності. Незважаючи на це, потужна система

шифрування даних та анонімності робить Telegram привабливим для тих, хто цінує конфіденційність та безпеку.

Таким чином, виходячи з огляду сучасних тенденцій та інновацій у сфері чат-ботів, можна визначити, що Telegram є оптимальним вибором для реалізації чат-бота у сфері транспорту, зокрема через його пріоритет на безпеку та конфіденційність. Інтеграція з цією платформою може бути виконана за допомогою бібліотек Python-telegram-bot, Aiogram або Telebot, які спрощують роботу з Telegram API та дозволяють створювати різноманітні команди бота для зручної взаємодії з користувачами.

Telegram має досить простий та інтуїтивно зрозумілий інтерфейс, що робить його доступним для широкого кола користувачів. Це одна з його сильних сторін. Вбудована підтримка багатьох корисних функцій, таких як налаштування каналів, боти, стікери, анонімні чати тощо. Це розширює можливості використання Telegram. Telegram активно розвивається і регулярно випускає оновлення з новим функціоналом. Це дозволяє йому йти в ногу з потребами користувачів. Потужна система шифрування даних та анонімності робить Telegram привабливим для тих, хто цінує конфіденційність та безпеку. З іншого боку, через анонімність Telegram іноді критикують за те, що він може використовуватися для незаконної діяльності. Також деякі експерти ставлять під сумнів криптографічні протоколи, що використовуються у Telegram, хоча компанія стверджує, що вони є надійними. Потрібно зазначити, що Telegram використовує протокол MTProto для шифрування переписки. Він ґрунтується на комбінації асиметричного і симетричного шифрування, що забезпечує конфіденційність та цілісність даних. Ключі шифрування генеруються для кожного користувача окремо і регулярно оновлюються. Це ускладнює дешифрування переписки сторонніми особами. Крім того, у Telegram є можливість створення «секретних чатів» з додатковим шифруванням end-to-end. Це означає, що лише учасники чату можуть читати повідомлення і сам Telegram не має до них доступу. Такі механізми захисту даних дозволяють користувачам почуватися впевненіше при використанні Telegram,

особливо якщо мова йде про конфіденційну інформацію, як от дані про поїздки чи оплату проїзду. Отже, саме через пріоритет безпеки Telegram є оптимальним вибором для реалізації чат-бота у сфері транспорту.

Також варто звернути увагу на вибір оптимального рішення для чат-бота. Використання бібліотеки Python-telegram-bot – це гарне рішення, оскільки вона дозволяє легко інтегрувати чат-бота з Telegram API. Має гнучкі можливості для створення різних команд бота. Можна розглянути використання фреймворків Aiogram або Telebot – вони спрощують роботу з Telegram API на основі asyncio та обробки подій. Чат-бот має дозволяти користувачам шукати оптимальні маршрути за пунктами початку та кінця подорожі. Варто подбати про швидкодію бота, можливо використовуючи бази даних Redis або механізми кешування запитів.

1.1 Аналіз платформ для розробки чат-ботів в телеграм

BotFather та Telegram Bot API є інструментами, спрямованими на створення та управління Telegram-ботами. BotFather, як служба, сприяє не лише створенню, але й конфігурації Telegram-ботів. Використовуючи BotFather, можна отримувати токени, налаштовувати назву, фотографію та інші параметри бота [11]. Деякі особливості BotFather включають:

– можливість генерації токенів для нових ботів;
– налаштування загальних параметрів бота;
– надання можливості конфігурувати команди та відповіді бота.

З іншого боку, Telegram Bot API надає програмний інтерфейс для взаємодії з Telegram-ботами. Цей інтерфейс дозволяє взаємодіяти з ботами, надсилати та отримувати повідомлення, обробляти команди і отримувати інформацію про користувачів. Особливості Telegram Bot API включають:

– повний доступ до можливостей Telegram для ботів;
– обробка подій, таких як отримання повідомлень та команд;

– надання API для взаємодії з різними функціональними можливостями Telegram.

BotFather (рис. 6) використовується для базової конфігурації та налаштування ботів, в той час як Telegram Bot API надає програмістам і розробникам широкі можливості для створення багатофункціональних ботів. BotFather, як інтерактивний бот, спрощує процес створення та налаштування бота для кінцевого користувача. Таким чином, обидва інструменти доповнюють один одного, надаючи повний спектр можливостей для розробки Telegram-ботів.

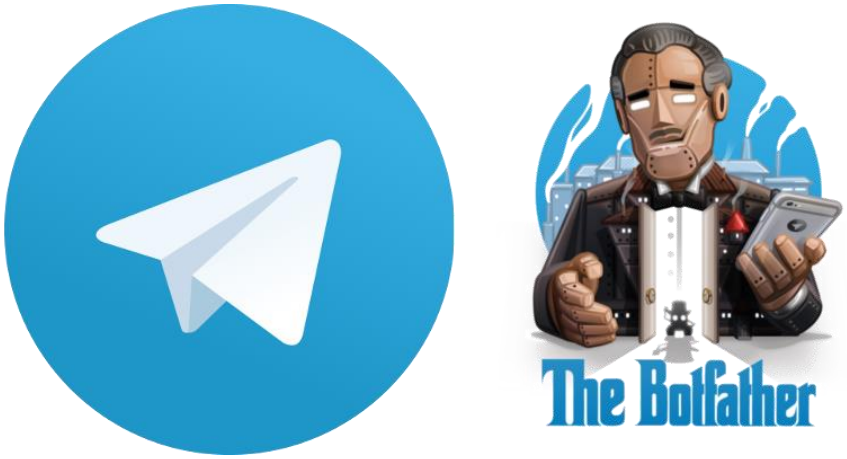

Рисунок 6 – Telegram bot BotFather

Python відкриває широкі можливості для розробки Телеграм-ботів через використання спеціальних бібліотек. Ці інструменти спрощують взаємодію з Telegram Bot API та пропонують інтуїтивно зрозумілий інтерфейс для розробників. Одна з таких бібліотек – python-telegram-bot. Вона дозволяє обробляти вхідні повідомлення та команди, а також відправляти текстові повідомлення, зображення, аудіо та інші медіафайли. Ця бібліотека також надає можливість взаємодії з клавіатурою для комфортної взаємодії з користувачем та створення інлайн-команд для використання в текстових повідомленнях. Інша популярна бібліотека – aiogram. Її особливості включають підтримку асинхронного програмування (asyncio), розширені можливості роботи з клавіатурою та інлайн-командами, і обробку різних типів повідомлень, таких як

фотографії, відео, голос тощо. Aiogram легко інтегрується з базами даних і забезпечує асинхронні запити до зовнішніх API, що сприяє оптимізації продуктивності. Обидві бібліотеки забезпечують зручні інструменти для взаємодії з іншими сервісами та системами, такими як обробка зображень, тексту чи робота з базами даних. Це робить їх популярними виборами для розробників, які прагнуть ефективно створювати та управляти функціональністю Телеграм-ботів.

Telegram, як платформа для обміну повідомленнями та взаємодії, надає розробникам широкий спектр інструментів для створення користувацьких інтерфейсів, які дозволяють забезпечити зручність та ефективність взаємодії з ботами. Inline-відповіді та клавіатури є важливими елементами для досягнення цього завдання. Telegram, як сучасний месенджер, акцентує важливість швидкості та простоти взаємодії. Користувачі віддають перевагу миттєвості та зручності отримання інформації чи виконання дій. В цьому контексті використання ефективних інструментів для взаємодії, таких як Inline-відповіді та клавіатури, є вирішальним для забезпечення задоволення користувачів та збереження їхнього інтересу.

Переваги використання Inline-відповідей та клавіатур:

– швидкий доступ до інформації (inline-відповіді дозволяють користувачам отримувати необхідну інформацію прямо в поточному чаті, без необхідності переходу до іншого меню чи розділу. Це сприяє швидкій та зручній взаємодії);
– взаємодія безпосередньо в чаті (клавіатури та Inline-відповіді розміщуються безпосередньо в чаті, дозволяючи користувачам спілкуватися та виконувати дії, не виходячи з основного вікна чату. Це робить взаємодію більш ефективною та приємною);
– персоналізація та варіативність (клавіатури дозволяють розробникам створювати персоналізовані та варіативні інтерфейси для кожного бота. Inline-відповіді можуть бути адаптовані під конкретні потреби

та функції бота, надаючи користувачам більше можливостей для вибору та взаємодії).

Використання Inline-відповідей та клавіатур у Telegram є ключовим елементом для покращення взаємодії користувачів з ботами. Ці інструменти сприяють швидкості та зручності взаємодії, роблячи процес спілкування більш ефективним та задовільним. Враховуючи акцент Telegram на інновації та зручність використання, використання Inline-відповідей та клавіатур є важливим кроком для розробників ботів.

1.2 Аналіз існуючих рішень в галузі інформаційного забезпечення громадського транспорту в телеграмі

В сучасних умовах громадський транспорт стає об'єктом інтенсивного інформаційного впливу, а використання месенджерів, таких як Telegram, набуває все більшої популярності для надання зручної та оперативної інформації пасажирам.

Багато міст та компаній громадського транспорту запускають інформаційних ботів у Telegram. Ці боти можуть надавати розклади руху, інформацію про затримки, обсяг пасажиропотоку та інші корисні дані. Деякі системи інформаційного забезпечення використовують Telegram для надсилання сповіщень та оновлень пасажирам. Це може включати в себе повідомлення про зміни в розкладах, ремонтні роботи чи інші важливі події. Деякі рішення дозволяють пасажирам в режимі реального часу відслідковувати рух громадського транспорту за допомогою інтерактивних мап.

Створення спільнот для пасажирів у Telegram дозволяє обмінюватися інформацією, досвідом та порадами щодо використання громадського транспорту в конкретному регіоні.

Основні вимоги та переваги:

– доступність інформації: інформація повинна бути легко доступною та зрозумілою для користувачів у формі, зручній для сприйняття;

– реальний час: система повинна надавати актуальну інформацію в реальному часі, зокрема щодо затримок та руху транспортних засобів;
– інтерактивність: можливість взаємодії з системою, наприклад, отримання сповіщень чи взаємодія з інтерактивними картами;
– спільнотна взаємодія: сприяння обміну інформацією та взаємодії між користувачами для підвищення загального рівня інформованості.

Аналіз існуючих рішень в галузі інформаційного забезпечення громадського транспорту в Telegram виявляє різноманіття підходів та інструментів. Дослідження вимагає уважного вивчення потреб користувачів та вдосконалення існуючих рішень для досягнення оптимальної ефективності та задоволення вимог громадськості.

1.3 Характеристика тенденцій та інновацій у розвитку чат-ботів для громадського транспорту в телеграмі

Сучасні чат-боти для громадського транспорту в Telegram не лише надають базову інформацію, таку як розклад руху транспорту, але із зростанням популярності отримують розширені функціональності. Це може включати в себе відстеження в реальному часі, сповіщення про зміни в руховому графіку, обсяг пасажиропотоку, інтеграцію з іншими сервісами тощо. Однією з тенденцій є інтеграція чат-ботів для громадського транспорту в Telegram з іншими платформами та сервісами, такими як відстеження GPS, платіжні системи та сервіси для обміну інформацією про стан маршрутів та зупинок. Використання технологій штучного інтелекту стає все популярнішим трендом у розвитку чат-ботів. Можливості аналізу великих обсягів даних та автоматизованої обробки інформації дозволяють ботам швидше та ефективніше надавати користувачам актуальну інформацію.

Інновації у напрямку голосового введення дозволяють користувачам взаємодіяти з чат-ботами громадського транспорту за допомогою голосу, що робляє взаємодію більш природною та зручною. Розвиток алгоритмів для

аналізу поведінки користувачів дозволяє чат-ботам надавати персоналізовані рекомендації стосовно маршрутів, оптимальних часів подорожі та інших параметрів, що підвищує задоволення від користування ботом. Інновації у сфері розпізнавання зображень дозволяють чат-ботам аналізувати зображення з відеокамер на транспорті та надавати користувачам інформацію про заповненість транспорту, зручності та інші аспекти.

Характеристика тенденцій та інновацій у розвитку чат-ботів для громадського транспорту в Telegram вказує на постійне розширення можливостей та зростання функціональності. Використання новітніх технологій та інновацій спрямоване на забезпечення зручності та ефективності взаємодії користувачів з чат-ботами, покращення якості обслуговування та підвищення загальної ефективності громадського транспорту.

2 ВИЗНАЧЕННЯ ВИМОГ ДО ФУНКЦІОНАЛЬНОСТІ ЧАТ-БОТА

Для створення ефективного та задовільного чат-бота необхідно чітко визначити функціональні вимоги, які визначають основні можливості, що чат-бот повинен надавати своїм користувачам. Вимоги до функціональності чат-бота визначають його основні можливості і функції, які має здійснювати бот для задоволення потреб користувачів. На рис. 7 представлені основні вимоги до функціональності чат-боту [12].

Рисунок 7 – Основні вимоги до функціональності чат-бота

По-перше, чат-бот повинен мати можливість взаємодіяти з користувачами через текстові повідомлення. Це включає розпізнавання питань та команд користувачів і надання відповідей або виконання відповідних дій. По-друге, чат-бот повинен бути здатний надавати користувачам інформацію на їх запитання або проактивно надсилати корисну інформацію, яка може бути цікавою для них. Третя вимога полягає у підтримці функцій маршрутизації. Якщо чат-бот призначений для громадського транспорту, він повинен мати функції маршрутизації, що дозволяють користувачам знаходити оптимальні маршрути та отримувати інформацію про громадський транспорт. Четверта вимога стосується оновлень та сповіщень. Чат-бот може надавати користувачам оновлення та сповіщення про зміни в розкладі руху транспорту, затримки, важливі повідомлення тощо. П’ята вимога полягає в інтерактивності та

налаштуваннях. Користувачам може бути надана можливість налаштування чат-бота, вибору мови, отримання персоналізованих рекомендацій тощо. Шоста вимога стосується інтеграції з іншими сервісами. Чат-бот може інтегруватися з іншими сервісами чи додатками для надання розширених функцій, таких як оплата проїзду, бронювання квитків, отримання інформації про погоду тощо. Сьома вимога стосується захисту та конфіденційності. Важливо забезпечити захист даних користувачів та конфіденційність під час взаємодії з чат-ботом. Восьма вимога полягає в аналітиці використання та звітності про помилки. Чат-бот може збирати дані про використання та надавати аналітику користувачам або адміністраторам, а також відстежувати помилки для подальшого вдосконалення.Ці вимоги допомагають забезпечити ефективну та задовільну роботу чат-бота та забезпечити відповідність його функціональності потребам користувачів.

UML-діаграма компонентів – це діаграма, яка допомагає візуалізувати структуру системи, її компоненти та взаємозв'язки між ними. Також, вона допомагає зрозуміти, які частини системи будуть відповідати за які функції та як вони будуть взаємодіяти. іаграма включає в себе компоненти системи (наприклад, модулі, бібліотеки, інтерфейси), а також зв'язки між ними (наприклад, залежності, агрегація, композиція).

Діаграма варіантів використання (Use Case Diagram) – діаграма, що визначає всі можливі способи використання системи шляхом ідентифікації її основних взаємодій з акторами (користувачами або зовнішніми системами) [13]. Діаграма включає в себе акторів (користувачів або зовнішні системи), варіанти використання (функціональність системи, яку надає кожен актор) та зв'язки між ними.

З метою побудови UML-діаграми компонентів для системи, яка розробляється потрібно спочатку ідентифікувати основні компоненти, з яких буде складатися система.

Компонент “Користувацький інтерфейс (User Interface)” відповідає за інтерфейс, через який користувачі будуть взаємодіяти з чат-ботом. Користу-

вачі взаємодіють з системою через текстові повідомлення у месенджерах, таких як Telegram, Facebook Messenger, веб-інтерфейс або мобільний застосунок. Інтерфейс забезпечує зручність використання, маючи інтуїтивний дизайн та доступність функцій для широкого кола користувачів.

Компонент “Обробник команд (Command Processor)” приймає вхідні команди від користувачів і виконує відповідні дії. Обробляє команди, надіслані користувачами, і виконує відповідні дії, такі як розпізнавання питань та команд, а також надання відповідей або виконання операцій.

Компонент “Система управління даними (Data Management System)” відповідає за зберігання та управління даними про розклади руху транспорту, інформації про зупинки, маршрути та інші дані, необхідні для функціонування чат-бота. Забезпечує функції збереження, оновлення та видалення даних, а також доступ до них для інших компонентів системи.

Компонент “Модуль аналізу (Analysis Module)”, використовує методи аналізу даних для визначення оптимальних маршрутів та надання користувачам інформації. Використовує алгоритми аналізу даних для прийняття рішень щодо відповідей на запитання користувачів.

Компонент “Інтеграційний модуль (Integration Module)” відповідає за інтеграцію з іншими сервісами та застосунками, наприклад, сервісами місцезнаходження або сервісами оплати. Забезпечує процес інтеграції та передачі даних між чат-ботом та іншими сервісами.

Після ідентифікації основних компонентів можна побудувати UML-діаграму компонентів, в якій компоненти будуть представлені у вигляді блоків, а зв'язки між ними - у вигляді стрілок. Кожен блок буде містити назву компонента та його основні функції або відповідальності. У діаграмі будуть показані також зв'язки між компонентами, що підкреслюють їх взаємодію та залежності (див.рис. 8).

Рисунок 8 – Проста UML-діаграма компонентів для розробляємої системи

Під час проектування чат-бота важливо створити чітку архітектуру, яка відображає компоненти та їх взаємозв'язки. Нижче (рис. 9) показана архітектура чат-бота месенджера, яка представлена з використанням компонентів та їх взаємозв'язків. Чат-бот структурований за допомогою вкладених компонентів, які включали в себе інші компоненти для забезпечення необхідної функціональності. Ця архітектура включає різноманітні компоненти, такі як модулі взаємодії з зовнішніми сервісами, сервер чат-боту та інші. Кожен з цих компонентів має свої внутрішні інтерфейси для взаємодії з іншими компонентами, а також зовнішніми системами. Особлива увага приділяється створенню чітких інтерфейсів між компонентами, що дозволяє ефективно співпрацювати один з одним.

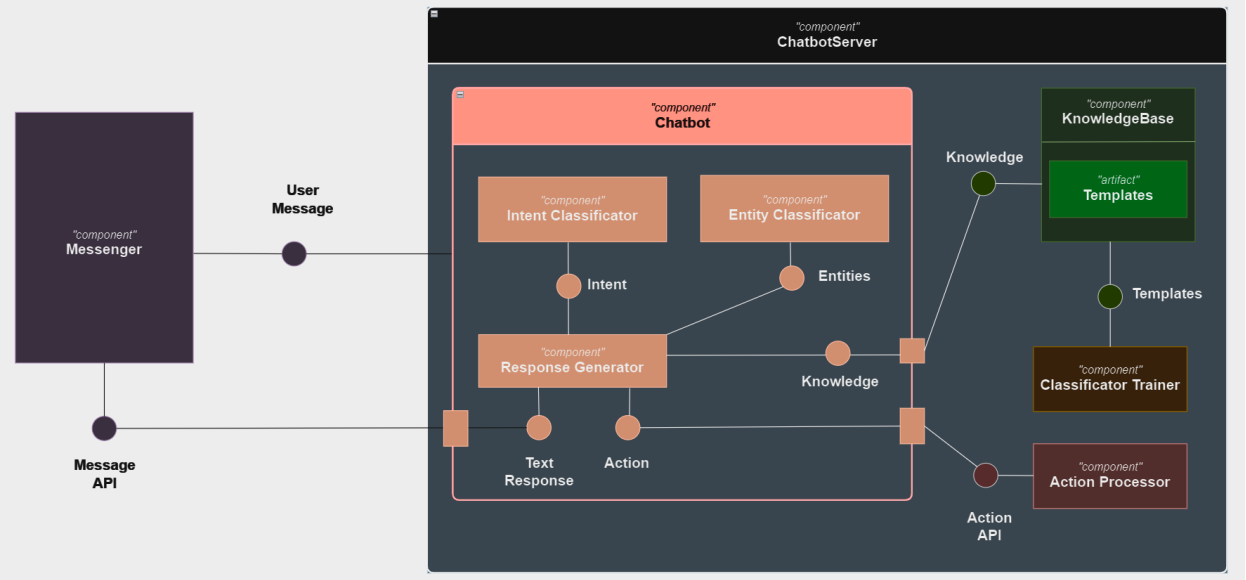

Рисунок 9 – UML-діаграма компонентів для чат-бота
(UML Component Diagram)

Далі необхідно виокремити випадки використання з різними акторами та випадками використання. Акторами можуть бути, наприклад, користувач Telegram та сам чат-бот. Крім того, важливо визначити випадки використання з різними акторами та випадками використання. Акторами можуть бути, наприклад, користувач Telegram та сам чат-бот. Для користувача Telegram функціональність може включати в себе випадки використання, такі як "Задати питання" та "Отримати відповідь", тоді як для чат-бота можуть бути "Надати інформацію про розклад руху", "Надати інформацію про реальний час руху транспорту" та інші.

Усе це можна відобразити на UML-діаграмі компонентів та діаграмі варіантів використання (Use Case Diagram), що дозволить чітко представити структуру та функціональність чат-бота. (рис. 10)

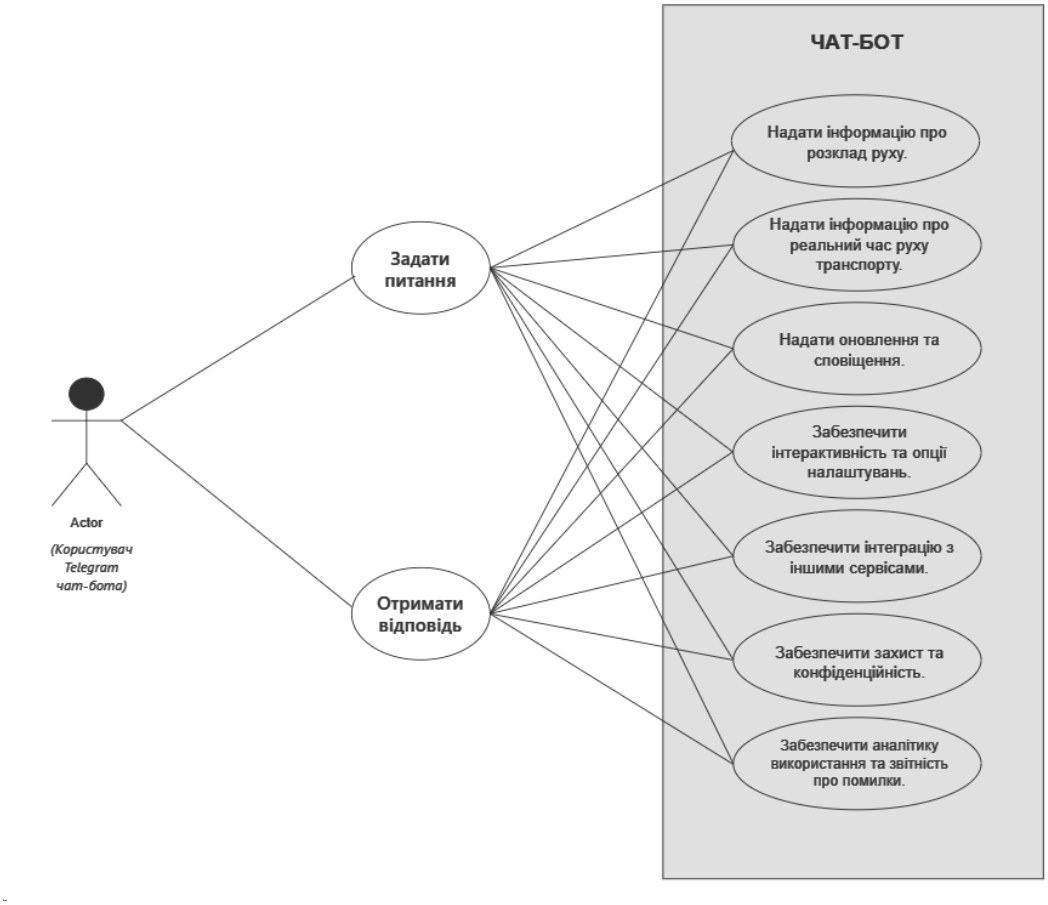

Рисунок 10 – Діаграма варіантів використання (Use Case Diagram)

Use Case Diagram є одним з ключових інструментів в аналізі вимог до програмного забезпечення. Він використовується для моделювання взаємодії між системою та її користувачами шляхом представлення різних сценаріїв використання системи. В контексті створення чат-бота для оптимізації доступу до інформації про громадський транспорт, Use Case Diagram допомагає виокремити різні функції та можливості, які пропонує система для користувачів. Відповідно до діаграми варіантів використання, можна більш детально описати та дати визначення акторам та випадкам використання:

– користувач Telegram (це основний користувач системи, який взаємодіє з чат-ботом через месенджер Telegram. Він ставить запитання, отримує відповіді та користується функціоналом, що надає чат-бот);

– чат-бот (це програмний агент, який взаємодіє з користувачем Telegram. Він приймає запити від користувачів, обробляє їх та повертає відповіді або виконує відповідні дії);
– задати питання (користувач ініціює цей випадок використання, ставляючи питання чат-боту через месенджер Telegram);
– отримати відповідь (чат-бот обробляє запитання користувача та надає відповідь на нього через месенджер Telegram);
– надати інформацію про розклад руху (чат-бот може надати інформацію про розклад руху громадського транспорту за запитом користувача);
– надати інформацію про реальний час руху транспорту (чат-бот може надати користувачеві інформацію про реальний час руху транспорту за запитом);
– інші функціональні вимоги (можливо, в системі також передбачені інші функціональні вимоги, які варто включити в аналіз випадків використання).

Залежності між випадками використання – відображають взаємозв'язки між різними випадками використання в системі. Це означає, що один випадок використання може впливати на інший випадок використання, існують певні зв'язки або взаємозалежності між ними. Залежності між випадками використання можуть бути різними за своєю природою, наприклад: включення (inclusion), розширення (extension) або включення виключень (inclusion of exceptions).

По-перше слід розглянути таку залежність як “Включення (Inclusion)”, це коли один випадок використання може включати в себе інший випадок використання. Наприклад, випадок “Придбати квиток” може включати в себе випадок “Оплатити квиток”.

Друга залежність – “Розширення (Extension)”, коли один випадок використання може розширювати інший випадок використання. Наприклад, випадок “Забронювати місце” може розширювати випадок “Придбати квиток”.

І, третя залежність “Включення виключень (Inclusion of Exceptions)” – один випадок використання може включати в себе випадок винятку для обробки помилок або відхилень. Наприклад, випадок “Придбати квиток” може включати в себе випадок винятку “Помилка оплати”.

Також варто звернути увагу на сценарії взаємодії. Сценарії взаємодії – це детальні описи послідовності подій та дій, які відбуваються між різними акторами (користувачами, системою тощо) під час виконання конкретного випадку використання (use case) в системі. У сценаріях взаємодії зазвичай описується, як актори взаємодіють між собою та з системою для досягнення певної мети або результату. Це може включати в себе обмін повідомленнями, запити та відповіді, зміни стану системи, дії користувача тощо. Сценарії взаємодії допомагають уточнити, як саме система повинна працювати у реальних ситуаціях та які кроки потрібно виконати для досягнення очікуваних результатів. Вони є важливим інструментом для розуміння та моделювання функціональності системи з точки зору користувача.

Для кожного випадку використання можна навести детальний сценарій взаємодії між акторами та системою. Це дозволяє детально описати послідовність дій, які виконуються в процесі взаємодії. Сценарій взаємодії може містити наступну інформацію:

– опис контексту: опис поточного стану системи та акторів перед початком взаємодії;
– кроки взаємодії: послідовність дій, які виконуються акторами та системою під час взаємодії;
– обмін повідомленнями: опис обміну повідомленнями, запитами та відповідями між акторами та системою під час взаємодії;
– умови завершення: опис умов, за яких взаємодія завершується успішно або невдало.

Представлення такої детальної інформації допомагає розуміти, як система взаємодіє з акторами та як вона поводиться в різних ситуаціях.

3 РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ЧАТ-БОТУ

Система управління базами даних (СУБД) – це програмне забезпечення для створення та управління базами даних, яке надає можливість зберігати, організовувати та отримувати доступ до даних, збережених у базі даних. Нижче наведено порівняльну таблицю найпопулярніших СУБД (табл.1).

Таблиця 1 – Порівняльна таблиця найпопулярніших СУБД

Критерій	SQLite	MySQL/MariaDB	PostgreSQL	MongoDB
Мова запитань	SQL	SQL	SQL	MongoDB Query Language
Тип бази даних	Реляційна	Реляційна	Реляційна	Документ-орієнтована NoSQL
Автономність	Вбудована база даних, працює локально без необхідності сервера	Вимагає сервера для роботи	Вимагає сервера для роботи	Може працювати як на сервері, так і в хмарі
Транзакції	Підтримка транзакцій, але обмежена у порівнянні з MySQL та PostgreSQL	Повна підтримка транзакцій з використанням команд COMMIT і ROLLBACK	Повна підтримка транзакцій з використанням команд COMMIT і ROLLBACK	Підтримка транзакцій, але інший механізм у порівнянні з реляційними СУБД
Масштабованість	Добре підходить для менших проектів та вбудованих систем	Підтримує велику кількість одночасних з'єднань, добре масштабується	Висока масштабованість, підтримка розподіленої бази даних	Гнучка масштабованість завдяки горизонтальному розширенню
Зовнішні ключі	Підтримка, але обмежена у порівнянні з реляційними СУБД	Повна підтримка	Повна підтримка	Не використовує концепцію зовнішніх ключів, оскільки це NoSQL база даних
Індексація	Обмежена можливість створення індексів	Повна підтримка різних видів індексів	Повна підтримка різних видів індексів	Можливість створення індексів для полів
Спрощення схеми	Менше обмежень на рівні схеми даних	Строга структура схеми даних	Строга структура схеми даних	Гнучка схема даних, дозволяє додавати поля безпосередньо в записи
Географічна підтримка	Обмежена підтримка географічних типів даних	Так, повна підтримка географічних типів та функцій	Так, повна підтримка географічних типів та функцій	Зручна робота з геолокаційними даними через вбудовані інструменти
Ліцензія	Доменні права, обмеження використання у комерційних продуктах	GPL або комерційна ліцензія	PostgreSQL License (подібна до MIT)	Server Side Public License (SSPL), деякі обмеження для комерційного використання

Для створення бази даних було обрано MySQL Workbench [14] з кількох причин. MySQL Workbench є безкоштовним програмним забезпеченням з відкритим кодом, що робить його доступним для використання без витрат на ліцензії або підписки. MySQL Workbench надає широкий набір інструментів для моделювання баз даних, проектування схем, створення таблиць, редагування та візуалізації даних. Він підтримує велику кількість операцій з базами даних MySQL, що робить його потужним інструментом для розробників та адміністраторів баз даних. MySQL Workbench розроблено спеціально для роботи з базами даних MySQL, тому він має вбудовану підтримку для цієї системи управління базами даних. Це забезпечує ефективну роботу з MySQL, оптимізовану для використання всіх можливостей цієї системи. MySQL Workbench надає можливість візуального моделювання баз даних, що дозволяє розробникам створювати та оптимізувати схеми баз даних за допомогою графічного інтерфейсу. Це спрощує роботу з базами даних та дозволяє швидше створювати та налаштовувати їх. MySQL Workbench є популярним інструментом у спільноті розробників, тому він має широку підтримку у вигляді документації, форумів та онлайн-ресурсів. Це забезпечує доступ до допомоги та ресурсів для вирішення будь-яких питань чи проблем, які можуть виникнути під час роботи з MySQL Workbench.

Для виконання запитів та зберігання інформації про маршрути транспортних засобів (публічних), зупинки та інші деталі, можна розглянути створення декількох таблиць у базі даних. Нижче подано загальний приклад того, як виглядають таблиці для використання.

Таблиця "Маршрути" (routes):

- id (TEXT, PRIMARY KEY): унікальний ідентифікатор маршруту;
- agency_name (TEXT): назва агентства;
- short_name (TEXT): коротка назва маршруту;
- long_name (TEXT): довга назва маршруту;
- type (INTEGER): тип транспорту;
- color (TEXT): колір маршруту;

– text_color (TEXT): колір тексту маршруту.

Таблиця "Зупинки" (stops):

– id (TEXT, PRIMARY KEY): унікальний ідентифікатор зупинки;
– name (TEXT): назва зупинки;
– lat (REAL): широта зупинки;
– lon (REAL): довгота зупинки.

Таблиця "Шаблони" (patterns):

– id (TEXT, PRIMARY KEY): унікальний ідентифікатор запису;
– description (TEXT, FOREIGN KEY): опис;
– route_id (TEXT, FOREIGN KEY): зовнішній ключ, посилається на routes(id).

Таблиця "Зупинки на маршруті" (stop_patterns):

– pattern_id (TEXT, PRIMARY KEY): унікальний ідентифікатор запису, посилається на patterns(id);
– stop_id (TEXT, FOREIGN KEY): зовнішній ключ, посилається на stops(id);
– stop_sequence (INTEGER): послідовність зупинок.

Таблиця "Геометрія маршруту" (route_geometry):

– pattern_id (TEXT, PRIMARY KEY): унікальний ідентифікатор запису, посилається на patterns(id);
– points (TEXT): точки (мітки) на маршруті;
– length (INTEGER): довжина маршруту.

Ці таблиці були розроблені з урахуванням потреб системи у зберіганні та організації інформації про маршрути та зупинки транспортних засобів, а також для підтримки географічної інформації про маршрути. Розглянуті структури таблиць можуть бути використані для забезпечення ефективного та організованого зберігання даних про громадський транспорт та його маршрути.

База даних (БД) – це організована колекція даних, яка зберігається та управляється комп'ютерною системою таким чином, щоб дозволити легкий

доступ, оновлення та управління цими даними. У контексті програмного забезпечення, БД є основним механізмом для зберігання, організації та маніпулювання даними, що використовуються програмою або додатком.

База даних є важливою складовою процесу створення проєктів з різних причин:

– зберігання та організація даних;
– ефективний доступ до даних;
– забезпечення цілісності та консистентності даних;
– забезпечення безпеки даних;
– підтримка масштабованості.

БД дозволяє зберігати великий обсяг даних та організувати їх у структуровану форму, що дозволяє ефективний доступ до інформації та швидке виконання запитів. База даних надає можливість швидкого та ефективного доступу до потрібної інформації, що дозволяє програмам та додаткам оперативно виконувати свої завдання. БД забезпечує контроль за цілісністю та консистентністю даних, запобігаючи неправильному зберіганню чи втраті інформації. Бази даних мають механізми безпеки, що дозволяють захищати дані від несанкціонованого доступу, змін та втрати. Використання БД дозволяє забезпечити масштабованість проєкту, тобто легко розширювати обсяг та функціональність системи з ростом потреб користувачів.

У випадку створення чат-боту, база даних відіграє ключову роль у зберіганні користувацьких даних, історії чатів, налаштувань користувачів, а також в інших аспектах, таких як аналітика використання, звітність та інтеграція з іншими сервісами. Без ефективної бази даних чат-бот може втратити можливість зберігати та використовувати важливу інформацію для забезпечення ефективного функціонування та взаємодії з користувачами (див.рис. 11).

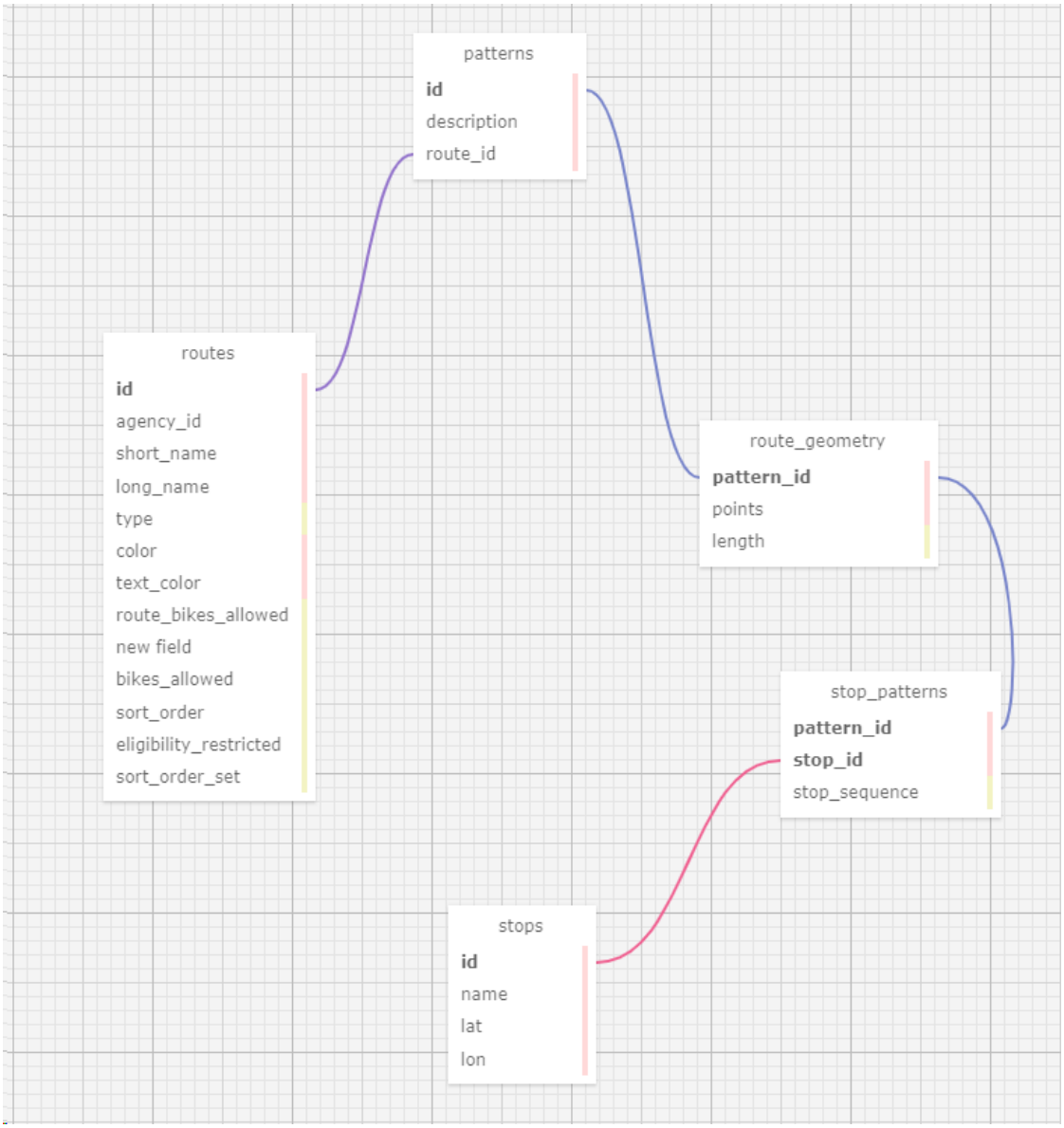

Рисунок 11 – Схема БД routes_odesa

Основні команди, які використовуються для створення БД, таблиць та внесення даних в них залежать від обраної СУБД. Враховуючи, що створюється БД для громадського транспорту, краще використовувати SQL для роботи з реляційною базою даних.

MySQL Workbench – це інтегроване середовище розробки та управління базами даних для MySQL, яка є реляційною СУБД. MySQL Workbench дозволяє розробляти моделі баз даних за допомогою графічного інтерфейсу. Користувачі можуть створювати схеми баз даних, таблиці, зв'язки між ними

та інші об'єкти за допомогою інструментів моделювання. Також, MySQL Workbench дозволяє створювати та адмініструвати бази даних MySQL безпосередньо з інтерфейсу. Користувачі можуть створювати нові бази даних, таблиці, індекси, обмеження та інші об'єкти баз даних, а також виконувати операції з резервним копіюванням, відновленням та оптимізацією баз даних. MySQL Workbench надає інтерфейс для візуального редагування даних у таблицях баз даних. Користувачі можуть додавати, змінювати та видаляти дані безпосередньо з таблиць, що полегшує роботу з даними. MySQL Workbench має інтегровану розробку SQL, яка дозволяє користувачам створювати та виконувати SQL-запити безпосередньо з інтерфейсу. Користувачі можуть використовувати інтерактивний редактор SQL для створення складних запитів та зберігати їх для майбутнього використання. MySQL Workbench дозволяє адмініструвати сервер MySQL через графічний інтерфейс. Користувачі можуть переглядати та керувати серверами, налаштовувати параметри сервера, моніторити його роботу та виконувати інші адміністративні завдання. MySQL Workbench є потужним інструментом для розробки та адміністрування баз даних MySQL, який полегшує роботу з базами даних та забезпечує розширені можливості для програмістів та адміністраторів баз даних.

Реляційні бази даних використовуються для зберігання та управління даними в табличній формі, де дані представлені у вигляді реляцій (таблиць).

Основні принципи реляційних баз даних включають:

– таблиці (дані зберігаються у вигляді таблиць, де кожен рядок таблиці представляє запис, а кожний стовпець відповідає певному атрибуту);
– ключі (ключі використовуються для ідентифікації унікальних записів в таблицях. Первинний ключ (PRIMARY KEY) визначає унікальний ідентифікатор кожного запису в таблиці, а зовнішній ключ (FOREIGN KEY) встановлює зв'язок між двома таблицями);

– відношення (відношення визначаються зв'язками між різними таблицями у базі даних. Це можуть бути один до одного, один до багатьох або багато до багатьох зв'язки);
– SQL (SQL (Structured Query Language) є мовою запитів, яка використовується для взаємодії з реляційними базами даних. Вона дозволяє виконувати операції вставки, оновлення, видалення та вибірки даних з таблиць);
– нормалізація (нормалізація – це процес організації даних у реляційній базі даних для уникнення аномалій та забезпечення ефективного зберігання).

Реляційні бази даних є одним з найпоширеніших типів баз даних, які використовуються в сучасних застосунках та системах. Вони надають зручний спосіб зберігання, організації та управління даними, що дозволяє ефективно працювати з великими обсягами інформації.

Окрім реляційних баз даних, існують інші типи баз даних, такі як ієрархічні, мережеві, об'єктні, NoSQL та документ-орієнтовані бази даних. Нижче описані їх характеристики та сфери застосування.

Дані в ієрархічних БД організовані у вигляді деревоподібної структури, де кожен запис має одного або більше батьківських та дочірніх записів. Ієрархічні бази даних часто використовуються у системах керування документами та веб-застосунках для організації вмісту.

Мережеві бази даних розширюють концепцію ієрархічних баз даних, дозволяючи записам мати більше одного батьківського запису. Вони використовуються для моделювання складних зв'язків між даними, таких як мережі залізниць або технологічні мережі.

Об'єктно-орієнтовані бази даних використовують об'єктно-орієнтовану модель для зберігання даних у вигляді об'єктів, що мають властивості та методи. Ці бази даних широко використовуються в програмуванні та розробці програмного забезпечення для зберігання та обробки об'єктно-орієнтованих даних.

NoSQL бази даних не використовують традиційні реляційні моделі та SQL. Вони забезпечують гнучку структуру даних та горизонтальне масштабування. NoSQL бази даних часто використовуються для обробки великих обсягів неструктурованих даних, таких як дані соціальних мереж, мережі Інтернету речей (IoT) та аналізу великих даних.

Документ-орієнтовані бази даних, в свою чергу зберігають дані у вигляді документів, таких як JSON або XML, замість традиційних таблиць та рядків. Ці БД використовуються для зберігання та обробки документів, блогів, новинних статей та інших даних, які можуть бути легко відображені у вигляді документів.

У випадку, коли потрібно зберігати дані у вигляді структурованих таблиць з визначеними зв'язками між ними, реляційна база даних може бути кращим вибором. Вона надає стандартизований підхід до організації даних та дозволяє виконувати складні операції за допомогою SQL. Такий підхід особливо ефективний для багатьох типів додатків, таких як системи управління клієнтами (CRM), системи управління інформацією про продукти (PIM), фінансові системи та інші, де важлива структурованість та зв'язки між даними. Нище наведені основні SQL-команди, які використовуються для створення бази даних (БД), таблиць та внесення даних в БД (рис. 12).

Для створення бази даних використовувалася команда:

```
CREATE DATABASE routes_odesa;
```

Створення таблиці “Маршрути” (routes), здійснювалося за допомогою команди:

```
CREATE TABLE routes (
    id TEXT PRIMARY KEY,
    agency_name TEXT,
    short_name TEXT,
    long_name TEXT,
    type INTEGER,
    color TEXT,
    text_color TEXT
);
```

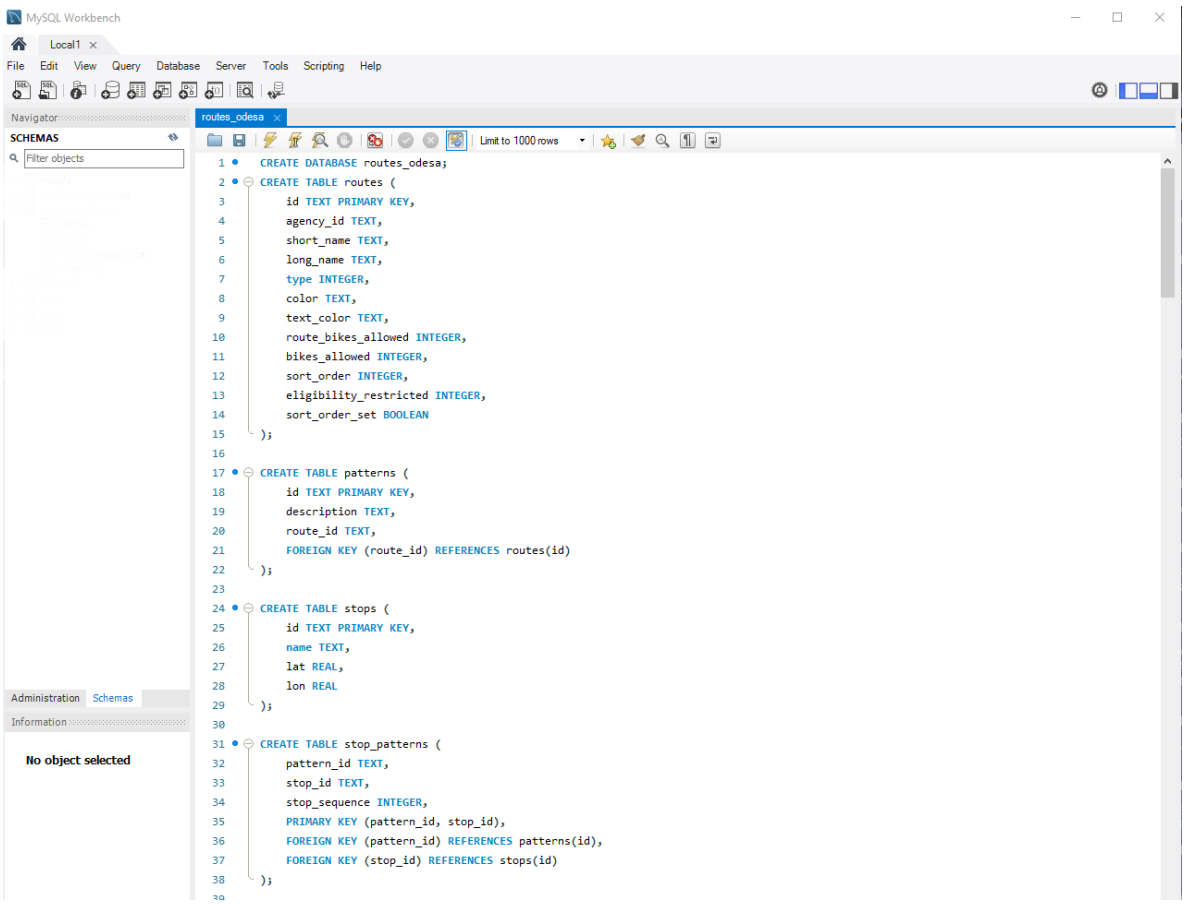

Рисунок 12 – Візуальне представлення створення БД routes_odesa в середовищі MySQL Workbench

Створення таблиці “Зупинки" (stops)”, здійснювалося за допомогою команди:

```
CREATE TABLE stops (
    id TEXT PRIMARY KEY,
    name TEXT,
    lat REAL,
    lon REAL
);
```

Для створення таблиці “Геометрія маршруту” (route_geometry) з урахуванням зовнішніх ключів, використовувалася команда:

```
CREATE TABLE route_geometry (
    pattern_id TEXT PRIMARY KEY,
    points TEXT,
    length INTEGER,
);
```

Для створення таблиці “Шаблон” (patterns) з урахуванням зовнішніх ключів, використовувалася команда:

```
CREATE TABLE patterns (
    id TEXT PRIMARY KEY,
    description TEXT,
    route_id TEXT,
    FOREIGN KEY (route_id) REFERENCES routes(id)
);
```

Для створення таблиці “Маршрути-Зупинки” (stop_patterns) з урахуванням зовнішніх ключів, використовувалася команда:

```
CREATE TABLE stop_patterns (
    pattern_id TEXT,
    stop_id TEXT,
    stop_sequence INTEGER,
    PRIMARY KEY (pattern_id, stop_id),
    FOREIGN KEY (pattern_id) REFERENCES patterns(id),
    FOREIGN KEY (stop_id) REFERENCES stops(id)
);
```

Внесення даних в таблицю “Маршрути” (routes), було виконано за допомогою команди:

```
INSERT INTO routes (id, agency_name, short_name, long_name, type, color, text_color)
VALUES ('route1', 'Agency A', 'Route A', 'Long Route A', 1, 'blue', 'white');
```

Внесення даних в таблицю “Зупинки" (stops)”, було виконано за допомогою команди:

```
INSERT INTO stops (id, name, lat, lon)
VALUES ('stop1', 'Stop A', 123.456, -78.901);
```

4 РЕАЛІЗАЦІЯ ЧАТ-БОТУ ДЛЯ ОПТИМІЗАЦІЇ ДОСТУПУ ДО ІНФОРМАЦІЇ ПРО ГРОМАДСЬКИЙ ТРАНСПОРТ

Створення чат-ботів на сьогодні є дуже актуальним напрямком розробки програмного забезпечення. Чат-боти дозволяють автоматизувати взаємодію з користувачами та надавати оперативну інформацію цілодобово.

Особливо зручно використовувати чат-боти в месенджерах, адже величезна аудиторія користувачів вже присутня саме там. Наприклад, створення бота в Телеграм дозволить миттєво охопити сотні тисяч потенційних клієнтів. Переваги чат-ботів – це швидкість, зручність, цілодобова робота без перерв. Чат-боти оптимізують робочі процеси компаній, дозволяючи автоматизувати рутинні завдання.

Одним з поширених застосувань чат-ботів є надання довідкової інформації користувачам. Наприклад, чат-бот для надання інформації про громадський транспорт. Нижче представлено етапи створення чат-боту:

– створення бота в Telegram за допомогою BotFather;
– підключення бібліотеки pyTelegramBotAPI;
– ініціалізація об'єкта бота;
– написання обробників команд користувача;
– виклик API для отримання даних;
– форматування даних;
– відправка повідомлення користувачу;
– запуск бота в режимі опитування.

Для створення нового бота потрібно знайти в Телеграм бота @BotFather та надіслати йому команду /newbot. Після цього обрати ім'я бота та його username. В результаті BotFather надасть токен доступу до HTTP API бота, який потрібно буде використовувати для подальшої взаємодії.

Підключення бібліотеки pyTelegramBotAPI надасть зручний інтерфейс для створення Telegram ботів на Python. Вона містить методи для обробки оновлень, відправки повідомлень, роботи з мультимедіа, клавіатурами тощо.

pyTelegramBotAPI регулярно оновлюється та підтримує останні можливості API Telegram. Спочатку варто встановити бібліотеку за допомогою pip. Далі, імпортувати її в проект, щоб мати доступ до класів та методів. Слід зауважити, що дана бібліотека надає зручний API для створення ботів на Python.

Після імпорту бібліотеки створюємо екземпляр класу TeleBot, передаючи в конструктор отриманий раніше токен. Це дасть змогу використовувати методи бібліотеки для взаємодії з Telegram API від імені нашого бота.

Далі необхідно використати декоратор message_handler, щоб вказати функцію-обработчик для текстових повідомлень. У цій функції за допомогою операторів if/else можна аналізувати вміст повідомлення та виконувати потрібні дії – відправляти відповідь, викликати інші функції тощо. Написання обробників команд користувача дозволяє аналізувати вміст повідомлення та виконувати необхідні дії у функції-обработчику.

Для отримання даних про маршрути потрібно скористатися API громадського транспорту міста. Наприклад, для Києва це data.kyivsmartcity.com. Для отримання даних про маршрути громадського транспорту в Одесі можна скористатися наступним API: Odessa IO – відкрите API транспорту Одеси (https://odessa.io/). Це API надає інформацію про маршрути, зупинки, розклад руху, місцезнаходження транспорту в режимі реального часу. Документація доступна за посиланням https://odessa.io/ru/docs/. Щоб отримати список усіх маршрутів, можна зробити GET запит до endpoint: https://odessa.io/api/routes. У відповіді буде JSON масив з об'єктами маршрутів, що містять назву, номер маршруту, кінцеві зупинки та іншу інформацію. Для отримання розкладу конкретного маршруту використовується endpoint: https://odessa.io/api/routes/{route_id}/schedule. Де {route_id} – унікальний ідентифікатор маршруту. Отже, Odessa IO API надає всі необхідні дані для створення чат-бота з інформацією про громадський транспорт Одеси. API може повертати дані у форматі JSON, які потім обробляються в Python для створення текстового повідомлення зі списком маршрутів.

Для форматування даних необхідно перетворити JSON з API у зручний для користувача вигляд. А також, чформувати структуроване текстове повідомлення з отриманими даними.

На етапі відправки повідомлення користувачу, використовується метод send_message об'єкта бота. Що дозволяє в свою чергу передати текст повідомлення та ідентифікатор чату [15].

Сформувати структуроване текстове повідомлення з отриманими данимиМетод bot.polling() запускає основний цикл опитування Telegram API на наявність нових повідомлень. Це дозволяє боту працювати цілодобово та обробляти запити користувачів в режимі реального часу.

Для початку необхідно імпортувати необхідні бібліотеки та створення об'єкта бота. Ц е можна зробити за допомогою команди:

```
import telebot
bot = telebot.TeleBot(TOKEN)
```

Щоб створити обробник команди /start, необхідно прописати наступну команду [16]:

```
@bot.message_handler(commands=['start'])
def start(message):
  bot.send_message(message.chat.id,
     "Вітаю, я бот для надання інформації про громадський транспорт!"
     "Введи /help для перегляду можливостей.")
```

При отриманні команди /start бот відправляє привітальне повідомлення з описом функціоналу.

А для створення обробника команди /help:

```
@bot.message_handler(commands=['help'])
def help(message):
  bot.send_message(message.chat.id, HELP_TEXT)
```

Команди /help виводить довідкову інформацію про команди бота.

Команда, яка описна нище допомагає отримати список маршрутів:

```
@bot.message_handler(commands=['routes'])
def get_routes(message):
```

```
  routes = get_routes_from_api()
  bot.send_message(message.chat.id, format_routes(routes))
```

Викликає API для отримання даних про маршрути і форматує їх для відправки.

Отримання розкладу руху:

```
@bot.message_handler(commands=['schedule'])
def get_schedule(message):
  args = message.text.split()
  if len(args) < 2:
    bot.send_message(message.chat.id, "Вкажіть номер маршруту")
    return
  schedule = get_schedule_from_api(args[1])
  bot.send_message(message.chat.id, format_schedule(schedule))
```

Отримує розклад для вказаного маршруту та надсилає його користувачу.

Для запуску бота використовується наступна команда:

```
bot.polling()
```

Ця команда запускає опитування сервера Telegram на наявність нових повідомлень.

Також потрібно приділити увагу підключенню бази даних до Telegram чат-боту з використанням бібліотеки pymysql. Спочатку варто імпортувати бібліотеку pymysql для роботи з БД MySQL/MariaDB. Далі, виконується підключення до БД в коді бота. У обробнику команди слід виконати SQL-запит до потрібної таблиці. Далі, варто перейт до формування відповідей шляхом конкатенації рядків результату. Таким чином виправляються дані користувачу методом send_message. Так можна надавати актуальні дані з БД за допомогою Telegram бота.

```
import telebot
import pymysql
# створюємо бота
bot = telebot.TeleBot(TOKEN)
# підключаємося до БД
db = pymysql.connect(host='localhost', user='username', passwd='password', db='database_name')
```

```
# обробник команди /get_data
@bot.message_handler(commands=['get_data'])
def send_data(message):
    # виконуємо SQL-запит до БД
    cursor = db.cursor()
    cursor.execute("SELECT * FROM mytable")
    # формуємо відповідь
    response = ""
    for row in cursor:
        response += str(row) + "\n"
    # відправляємо дані користувачу
    bot.send_message(message.chat.id, response)
# запускаємо бота
bot.polling()
```

Нижче наведені деякі додаткові фрагменти коду, які можуть знадобитися для налаштування Telegram чат-бота. Наприклад для роботи з кнопками:

```
from telebot import types
# створюємо кнопки
btn1 = types.KeyboardButton('/start')
btn2 = types.KeyboardButton('/help')
# групуємо кнопки в меню
menu = types.ReplyKeyboardMarkup(resize_keyboard=True)
menu.add(btn1, btn2)
# відправляємо кнопки разом з повідомленням
bot.send_message(chat_id, "Текст повідомлення", reply_markup=menu)
```

Робота з кнопками дозволяє створювати зручне меню з кнопками для навігації користувача. Користувачам легше натискати на кнопки, ніж вводити команди вручну. Також можна згрупувати функціонал бота в логічне меню.

Логування помилок допомагає відстежувати помилки та проблеми в роботі бота. Логи можна зберігати в файл чи відправляти безпосередньо розробнику. Це в свою чергу полегшує налагодження та виправлення дефектів бота.

```
import logging
logger = telebot.logger
telebot.logger.setLevel(logging.ERROR)
```

Обробка помилок – перехоплює винятки, щоб бот не “вильітав” при помилках. Це дозволяє коректно відобразити помилку користувачу та запобігає перериванню роботи бота через виняткові ситуації.

```
from telebot import apihelper
@bot.message_handler(content_types=['text'])
def handle_text(message):
  try:
    # обробка логіки
  except Exception as e:
    apihelper.send_message(chat_id, 'Виникла помилка: '+str(e))
```

ВИСНОВКИ

У даній роботі було розглянуто актуальну на сьогодні задачу оптимізації доступу до інформації про громадський транспорт в містах за допомогою чат-ботів. Розробка подібних чат-ботів дозволяє суттєво спростити отримання необхідних даних для пасажирів. Натомість створення таких програмних рішень вимагає ретельного аналізу предметної області, вибору оптимальних інструментів розробки та архітектури, врахування вимог до інтерфейсу та швидкодії системи.

В ході роботи було досліджено основні варіанти реалізації чат-ботів, їх переваги та недоліки. Зокрема, розглядались месенджери Telegram, Facebook Messenger, Viber, що є найбільш поширеними платформами для чат-ботів. Було обґрунтовано вибір Telegram як оптимального варіанту для даної задачі з огляду на його гнучкі можливості, відкритість API та значну кількість користувачів в Україні.

Окрема увага приділялась вибору мови програмування та фреймворків для розробки. Розглядались такі мови, як Python, JavaScript, PHP та C++. Було показано, що для даної задачі найбільш оптимальним є використання Python завдяки зручності та простоті мови, наявності великої кількості бібліотек, в тому числі спеціалізованих для створення чат-ботів. Детально проаналізовано архітектуру чат-бота, його основні компоненти та принципи їх взаємодії. Визначено, що ключовими елементами є модуль обробки запитів користувача, модуль взаємодії з API транспортної системи міста, компонент форматування даних, а також модуль інтеграції з месенджером.

Особливу увагу приділено питанням швидкодії та надійності системи, оскільки від цих факторів значною мірою залежить зручність використання чат-бота. Було запропоновано ряд підходів для оптимізації, таких як кешування даних, використання швидких баз даних, реплікація компонентів, горизонтальне масштабування.

Для наочності в роботі наведені приклади блок-схем та діаграм, що ілюструють архітектуру чат-бота та його роботу. Завдяки використанню сучасних технологій та оптимальній архітектурі розроблений чат-бот відрізняється високою швидкодією та надійністю. Його інтеграція з месенджерами та транспортними API дозволяє максимально спростити процес отримання даних користувачами.

Таким чином, створений у рамках даної роботи чат-бот може слугувати гарною технологічною базою для розвитку та впровадження подібних рішень в інших містах. Зручний інтерфейс, швидкодія та надійність роблять його конкурентоздатним продуктом на ринку. Розроблені підходи можуть бути використані для оптимізації існуючих систем надання інформації про транспорт.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Dialogflow. URL: https://cloud.google.com/dialogflow (дата звернення 18.02.2024)
2. Microsoft Bot Framework. URL: https://dev.botframework.com/ (дата звернення 08.02.2024)
3. Amazon Lex. URL: https://aws.amazon.com/lex/ (дата звернення 11.03.2024)
4. IBM Watson Assistant. URL: https://www.ibm.com/products/watsonx-assistant (дата звернення 18.02.2024)
5. Які мови програмування треба знати для створення сайтів. URL: https://foxminded.ua/movy-prohramuvannia-dlia-stvorennia-saitiv/ (дата звернення 08.02.2024)
6. Building Your First Serverless Telegram Bot with AWS Lambda. URL: https://iamondemand.com/blog/building-your-first-serverless-telegram-bot/ (дата звернення 14.02.2024)
7. Use Meta API to Send Messages to Users via Facebook Messenger. URL: https://medium.com/@sumindaniro/send-updates-or-messages-to-users-via-facebook-messenger-69e4e46ecce8 (дата звернення 21.03.2024)
8. WhatsApp API Guide: Your Way To Master WhatsApp Business API. URL: https://verloop.io/blog/whatsapp-api-guide/ (дата звернення 18.03.2024)
9. Enhancing Slack Bot Response Time with AWS Lambda and ECR. URL: https://medium.com/@caitlin.johnson/enhancing-slack-bot-response-time-with-aws-lambda-and-ecr-bc7c6194181b (дата звернення 18.03.2024)
10. tyntec | Architecture. URL: https://www.tyntec.com/helpcenter/docs/channels/viber-business/architecture/ (дата звернення 08.02.2024)
11. From BotFather to Hello World. URL: https://core.telegram.org/bots/tutorial (дата звернення 24.03.2024)

12. Від тесту Тьюринга до ChatGPT: що таке чат-боти, для чого їх використовують та яка їхня роль у сучасному світі. URL: https://mc.today/uk/shho-take-chat-boti/ (дата звернення 28.04.2024)
13. Як будувати UML-діаграми. Розбираємо три найпопулярніші варіанти. URL: https://dou.ua/forums/topic/40575/ (дата звернення 05.04.2024)
14. MySQL Workbench. URL: https://www.mysql.com/products/workbench/ (дата звернення 18.02.2024)
15. Будуємо телеграм чат-бот на Java: від ідеї до деплою. URL: https://dou.ua/forums/topic/38358/ (дата звернення 09.04.2024)
16. Як налаштувати привітання чат-бота Telegram. URL: https://sendpulse.ua/knowledge-base/chatbot/telegram/description (дата звернення 16.04.2024)