Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА TELEGRAM-БОТУ ДЛЯ ЗАМОВЛЕННЯ ПІЦИ З ІНТЕГРАЦІЄЮ ПЛАТІЖНОЇ СИСТЕМИ ТА AI З ВИКОРИСТАННЯМ PYTHON DJANGO І MYSQL

DEVELOPMENT OF A TELEGRAM BOT FOR PIZZA ORDERING WITH PAYMENT SYSTEM AND AI INTEGRATION USING PYTHON DJANGO AND MYSQL

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконала: здобувач вищої освіти
групи ІПЗ-41
Хітько Юлія Володимирівна

Керівник:
Вознюк Анастасія Вадимівна

Кваліфікаційну роботу
допущено до захисту
«10» 06 2025 р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна
_____

Луцьк – 2025 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти *бакалавр*
Галузь знань: *12 «Інформаційні технології»*
Спеціальність: *121 «Інженерія програмного забезпечення»*
Освітня програма: «*Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ

Завідувач кафедри
_____
«20» 12 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ
Хітько Юлії Володимирівні
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Розробка Telegram-боту для замовлення піци з інтеграцією платіжної системи та AI з використанням Python Django і MySQL»*
Керівник роботи: *асистент Вознюк А. В.*
затверджені наказом закладу вищої освіти від *«19» грудня 2024 р.* № *474/01-02*

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра *«10» червня 2025 р.*

3. Вихідні дані до роботи: *Python, Django, aiogram 3, Portmone, Open AI GPT, PyCharm, GitHub, методичні вказівки до виконання кваліфікаційної роботи бакалавра*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *аналіз предметної області, функціональні та нефункціональні вимоги до чат-бота, UML-діаграми, стек технологій, реалізація бота, тестування системи, проєктування структури бази даних, забезпечення захисту даних і підготовка документації*

5. Перелік графічного матеріалу: *28 рисунків, 1 додаток*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Вознюк А. В.*		
Специфікація вимог до розробленої системи	*Вознюк А. В.*		
Розробка об'єкта проектування	*Вознюк А. В.*		
Нормоконтроль	*Вознюк А. В.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	1,37 %		
Академічна доброчесність	*Вознюк А. В.*		

7. Дата видачі завдання *«20» грудня 2024 р.*

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 04.02.2025 р.	Виконано
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 01.03.2025 р.	Виконано
3	Обгрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 15.03.2025 р.	Виконано
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 29.03.2025 р.	Виконано
5	Практична реалізація об'єкта проектування та розробка бази даних	до 26.04.2025 р.	Виконано
6	Тестування та налагодження об'єкта проектування	до 03.05.2025 р.	Виконано
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 10.06.2025 р.	Виконано

Здобувач вищої освіти Юлія ХІТЬКО

Керівник кваліфікаційної роботи Анастасія ВОЗНЮК

АНОТАЦІЯ

Хітько Ю. В. Розробка Telegram-боту для замовлення піци з інтеграцією платіжної системи та AI з використанням Python Django і MySQL. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2025.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

У першому розділі здійснено аналіз предметної області, можливих способів доставки, платформ для реалізації чат-боту і сформульовано завдання. В другому розділі сформовано функціональні та нефункціональні вимоги, розроблено UML-діаграми для демонстрації ключових сценаріїв взаємодії, проаналізовано та обрано стек технологій для розробки. У третьому розділі описано практичну реалізацію, тестування та налагодження, проєктування бази даних, забезпечення безпеки інформаційної системи та розроблено документації для адмінпанелі.

У висновках було підбито підсумки кожного етапу проведеної роботи над кваліфікаційною роботою бакалавра.

Ключові слова: чат-бот, Telegram, Python, Django, Portmone, штучний інтелект, доставка.

ABSTRACT

Khitko Y. Development of a Telegram Bot for Pizza Ordering with Payment System and AI Integration Using Python Django and MySQL. Manuscript. Qualification work of the bachelor’s program "Software engineering" in the specialty "Software engineering". Lutsk National Technical University. Lutsk, 2025.

The bachelor’s qualification thesis consists of an introduction, three chapters, conclusions, a list of references, and appendices.

The first chapter presents an analysis of the subject area, possible delivery methods, and platforms for implementing the chatbot, and formulates the project’s objectives. The second chapter establishes the functional and non-functional requirements, develops UML diagrams to demonstrate key interaction scenarios, and analyzes and selects the technology stack for development. The third chapter describes the practical implementation, testing and debugging procedures, database design, security mechanisms applied to the information system, and the documentation created for the admin panel.

The conclusions summarize the results of each stage of work on this bachelor’s qualification project.

Keywords: chatbot, Telegram, Python, Django, Portmone, artificial intelligence, delivery.

ЗМІСТ

ВСТУП.....7
РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ.....9
1.1 Аналіз сучасного стану проблеми.....9
1.2 Постановка завдання на кваліфікаційну роботу бакалавра.....14
Висновки до розділу 1.....15
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО TELEGRAM-БОТУ ДЛЯ ЗАМОВЛЕННЯ ПІЦИ.....16
2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення.....16
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання...21
Висновки до розділу 2.....25
РОЗДІЛ 3 РОЗРОБКА TELEGRAM-БОТУ ДЛЯ ЗАМОВЛЕННЯ ПІЦИ З ІНТЕГРАЦІЄЮ ПЛАТІЖНОЇ СИСТЕМИ ТА AI.....26
3.1 Практична реалізація об’єкта проєктування.....26
3.2 Тестування та налагодження інформаційно-комп’ютерної системи.....41
3.3 Проєктування структури бази даних.....45
3.4 Захист інформаційно-комп’ютерної системи.....48
3.5 Розробка документації до програмного продукту.....49
Висновки до розділу 3.....54
ВИСНОВКИ.....55
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....57
ДОДАТКИ.....59

ВСТУП

Актуальність теми. Сучасний користувач очікує зручного, швидкого та персоналізованого підходу з боку сервісу без зайвих кроків та проблем. В той час зростає конкуренція серед закладів харчування, що вимагає впровадження інноваційних рішень, які не тільки роблять простішим процес оформлення замовлення, а й створюють персоналізований підхід та підтримують зацікавленість користувача.

Використання чат-ботів для автоматизації оформлення замовлення на доставку їжі у сфері харчування є хорошим рішенням, тому що для цього не потрібно завантажувати або переходити на сторонні додатки.

Метою кваліфікаційної роботи є створення Telegram-бота для замовлення піци зі штучним інтелектом та платіжною системою, який буде реалізований на таких технологіях як Python та фреймворк Django.

Об’єктом кваліфікаційної роботи є Telegram-боту для замовлення піци. Чат-бот має обробляти замовлення користувачів та створювати за допомогою ШІ унікальну піцу.

Предметом кваліфікаційної роботи бакалавра є практична реалізація Telegram-бота для замовлення піци з використанням Python та Django.

Для виконання поставленої цілі, були сформульовані наступні завдання:

– проаналізувати предметну область, способи замовлення їжі, платформи для реалізації чат-боту;

– сформулювати вимоги, розробити UML-діаграми за ключовими сценаріями взаємодії;

– проаналізувати наявні інструменти для розробки та обрати стек технологій;

– розробити чат-бот, підключити адміністраторську панель;

– здійснити оптимізацію коду, провести тестування та виправити знайденні помилки та неточності;

– спроєктувати концептуальну ER-модель бази даних та налаштувати механізми міграцій даних;

– забезпечити захист конфіденційних даних користувачів;

– розробити документацію для адміністрування чат-ботом.

Апробацію результатів кваліфікаційної роботи здійснено шляхом участі в X Міжнародній науково-практичній конференції з проблем вищої освіти і науки «Інформаційні технології в освіті, науці і виробництві (ІТОНВ-2025)», м. Луцьк, 23-24 травня 2025 р. (додаток А).

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У сучасному світі великих обертів набирає автоматизація та оптимізація процесів замовлення товарів або послуг через вебсайти у вигляді інтернет-магазинів, чат-боти в месенджерах тощо. Конкурентними та дуже актуальними є рішення для яких користувачу не потрібно докладати багато зусиль, доступ до яких може бути з мобільного телефону. Чим простішим є процес для потенційного клієнта, тим більш затребуваним є цей продукт.

Сфера доставки їжі стрімко почала розвиватися з початком пандемії COVID-19, коли обмеження на пересування та закриття закладів харчування змусили споживачів масово переходити до онлайн-замовлень. Те, що спочатку було вимушеним заходом, з часом перетворилося на зручну звичку, яка зберігає популярність і сьогодні. У постпандемічний період попит на сервіси доставки не лише не знизився, а й продовжує зростати завдяки комфорту, економії часу та постійній цифровій трансформації бізнесу (рис. 1.1).

Рисунок 1.1 – Статистика попиту на прикладі сервісу Glovo [1]

В даній сфері вже існує багато популярних рішень, про які чув практично кожен, наприклад: Glovo, MisterAm, Zakaz.ua, BoltFood та інші [2]. Кожен з цих програмних продуктів надає широкий функціонал, як для користувачів, так і для закладів харчування. Вони дають змогу переглядати, створювати, оформляти, оплачувати, відстежувати замовлення. Усі ці платформи для доставки їжі є досить складними в процесі реалізації, вони потребують значних ресурсів для розборки, тестування та супроводу.

Окрім сервісів, які спеціалізуються лише на доставках їжі, що співпрацюють з численними кафе та ресторанами, багато закладів харчування також мають власну систему доставки.

Аналіз сучасних способів замовлення їжі через різні цифрові канали показує, що користувачі мають широкий вибір платформ, кожна з яких має свої переваги й недоліки.

Мобільні застосунки є одним із найпоширеніших каналів для замовлення піци. Вони зазвичай мають зручний інтерфейс, підтримку збереження історії замовлень, профілів користувачів, бонусних програм і push-сповіщень. Серед недоліків, користувачі неохоче встановлюють нові застосунки, особливо якщо замовлення не є регулярним. Крім того, розробка та підтримка таких додатків вимагає значних ресурсів від бізнесу.

Вебсайти дозволяють користувачеві бачити повне меню з фото, описами, цінами та акціями. Це зручно для тих, хто хоче переглянути всі варіанти та налаштувати замовлення вручну. Основні недоліки – залежність від стабільного інтернет-з’єднання, потреба в адаптації сайту для мобільних пристроїв, складність навігації на екранах з низькою роздільною здатністю.

Деякі заклади дозволяють робити замовлення безпосередньо через повідомлення в соцмережах. Такий підхід не вимагає переходу на зовнішній ресурс і зручний для постійних користувачів цих платформ, що є перевагою. При цьому користувач не може робити замовлення прямо під час перегляду меню, що змушує робити дуже велику кількість кроків. Присутня висока ймовірність довгого очікування відповіді, оскільки менеджер може оформляти

інше замовлення. В даного способу немає автоматичного підтвердження замовлення, що може спричинити людську помилку, та відсутня інтеграція з платіжною системою.

Телефонні дзвінки досі є традиційним способом замовлення доставки їжі, особливо серед старшої аудиторії. Його основний плюс – живе спілкування з оператором, що дозволяє уточнити всі деталі. Проте це не завжди зручно: потрібно чекати з'єднання, повторювати замовлення вручну, оплата найчастіше відбувається лише готівкою або переказом, також цей варіант не є комфортним для людей, які через свої фізіологічні або психологічні особливості не полюбляють або не можуть здійснювати голосові дзвінки. Це можуть бути, зокрема, люди з порушенням слуху, мовлення або ті, хто відчуває тривожність під час спілкування з незнайомими людьми телефоном. Крім того, такий спосіб не дозволяє швидко порівнювати варіанти меню чи акції, що робить його менш гнучким у порівнянні з сучасними, цифровими варіантами здійснення замовлення.

Спеціалізовані сервіси доставки (наприклад, Glovo, Bolt Food та Uber Eats) дозволяють швидко оформити замовлення через додаток або вебінтерфейс, порівнювати акції, оплачувати онлайн і відстежувати кур'єра в реальному часі. Перевагами є зручність і швидкість оформлення замовлення в будь-який час, широкий вибір ресторанів, різні способи оплати, live-tracking. Недоліки: комісія платформи може підвищувати вартість замовлення, інтернет-залежність, мінімальна сума замовлення та вартість товарів іноді вища ніж в застосунку закладу харчування, висока ймовірність затримок в пікові години, відсутність в маленьких містах.

Враховуючи проведений аналіз способів доставки їжі, можна сказати, що кожен з них має свої переваги та недоліки. Оформлення замовлення через телефонні дзвінки або соціальні мережі приваблюють тим, що це живе спілкування з іншою людиною, при цьому присутня висока ймовірність людської помилки, наприклад клієнту не приїде якась позиція або вона буде замінена на іншу.

Якщо ж брати до уваги вебсайти або мобільні додатки, в яких замовлення вже здійснюється більш автоматизовано, що зменшує можливість людської помилки, то серед недоліків можна виявити шаблонність (в таких сервісах зазвичай типове меню та інтерфейс), необхідність переходити на сторонні ресурси. З боку підприємців такі рішення потребують значних ресурсів для розробки та підтримки.

У своїй розробці я прагну зберегти всі виявлені переваги та виправити знайдені недоліки, щоб створити максимально зручний, надійний і безпечний сервіс для доставки.

У сучасних реаліях стрімко популярності набирає оптимізація процесів за допомогою штучного інтелекту в різних галузях. Він дозволяє автоматизувати рутинні процеси, оптимізувати операційні витрати та підвищувати продуктивність праці. Компанії, що вчасно інтегрують штучний інтелект, здобувають конкурентні переваги – від точнішого прогнозування попиту до персоналізації клієнтського досвіду.

Нижче наведено стовпчикову діаграму, яка ілюструє відсоток в які саме процеси компанії інтегрують штучний інтелект [3]. На графіку видно, що найбільша частка (56 %) застосувала ШІ для оптимізації операцій (рис. 1.2).

Рисунок 1.2 – Використання ШІ для бізнес-функцій

Сфера доставки також успішно впроваджує штучний інтелект у свої процеси. Наприклад, ШІ аналізує погодні умови та трафік на дорогах, будує найефективніший маршрут, для економії часу та інших ресурсів. Також штучний інтелект може динамічно змінювати маршрут, тобто якщо під час руху водій наближається до затору, його буде перебудовано [4].

Під час аналізу можливих способів взаємодії для створення нової системи було розглянуто кілька популярних месенджерів – Viber, WhatsApp, Facebook Messenger та Telegram. За результатами порівняння Telegram виявився найоптимальнішою платформою для розміщення чат-бота з огляду на низку технічних, практичних та стратегічних переваг [5].

Telegram має досить високу популярність серед користувачів віком 18-35 років – тобто основної частини аудиторії доставки їжі (рис. 1.3).

Рисунок 1.3 – Топ-5 популярних месенджерів в Україні

У багатьох українських містах він став основним месенджером для спілкування, новин та підписки на сервіси. Це означає, що користувачеві не потрібно встановлювати новий застосунок – бот працює безпосередньо в знайомому середовищі. Попри те, що Telegram наразі займає третє місце в топ-5 серед найпопулярніших месенджерів в Україні, його зростання є найдинамічнішим.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

За мету цієї кваліфікаційної роботи взято розробку Telegram-бота для замовлення піци з інтеграцією платіжної системи та штучного інтелекту, цей проєкт забезпечить зручний та швидкий доступ до замовлення, для користувачів, можливість оплачувати онлайн через платіжну систему та створювати персоналізовану піцу за допомогою штучного інтелекту, які ґрунтуються на основі смаків та вподобань клієнта.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– проаналізувати предметну область, оцінити зручність способів замовлення їжі, виділити переваги та недоліки кожного з них, проаналізувати платформи для реалізації чат-боту;

– сформулювати функціональні та нефункціональні вимоги до розроблювального чат-бота, розробити UML-діаграми за ключовими сценаріями взаємодії;

– проаналізувати наявні інструменти для розробки чат-бота на сучасному ринку та обрати стек технологій;

– розробити чат-бот, а саме реалізувати базові сценарії взаємодії чат-бота з користувачем, інтегрувати платіжний модуль, інтегрувати штучний інтелект для створення унікальної піци, підключити адміністраторську панель;

– здійснити оптимізацію коду, провести функціональне, інтеграційне, навантажувальне тестування та виправити знайденні помилки та неточності;

– спроєктувати концептуальну ER-модель бази даних та налаштувати механізми міграцій даних;

– забезпечити захист конфіденційних даних користувачів;

– розробити документацію для адміністрування чат-ботом.

Висновки до розділу 1

У першому розділі було виявлено та проаналізовано сучасні способи замовлення та доставки їжі, починаючи від мобільних застосунків і вебсайтів до телефонних дзвінків. Оцінено переваги зручності та швидкості з певними обмеженнями, зокрема високими витратами на розробку, залежністю від інтернету чи людським фактором кожного з них. Карантинні обмеження з приходом COVID-19 обумовили перехід споживачів до онлайн-замовлень, сформувавши у багатьох звичку, яка в свій час підвищує очікування щодо швидкості, простоти й надійності сервісів. Інтеграція штучного інтелекту у бізнес-процеси забезпечує персоналізації клієнтського досвіду та конкурентні переваги. Враховуючи стрімкий зріст популярності месенджеру Telegram, обрано цю платформу для реалізації чат-бота з інтеграцією платіжної системи та штучного інтелекту, а чітко сформульовані завдання проєкту забезпечать повноцінний підхід до розробки з прорахуванням всіх нюансів.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО TELEGRAM-БОТУ ДЛЯ ЗАМОВЛЕННЯ ПІЦИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

Метою даної розробки є створення інтерактивного продукту, який дозволяє користувачам швидко та зручно обирати щось із меню, оформлювати замовлення, здійснювати онлайн-оплату, а з боку адміністраторів – легко та просто вносити зміни в меню, відстежувати замовлення та змінювати їх статус. Дана система повинна забезпечити високий рівень користувацького досвіду, автоматизацію процесів, включаючи створення унікальної піци за допомогою штучного інтелекту, та безпеку.

Для забезпечення повноцінного функціонування Telegram-бота з можливістю замовлення піци необхідно реалізувати набір основних функцій, які охоплюють усі ключові сценарії взаємодії користувача із сервісом. До них належать такі функціональні вимоги:

– можливість ідентифікації користувача Telegram через ID або номер телефону;

– оформлення замовлення (вибір позицій меню із можливістю додавання до кошика, вказування кількості товарів, підтвердження замовлення);

– інтеграція з платіжною системою (оплата замовлення через Telegram Payments або сторонній платіжний сервіс, виведення інформації про статус платежу);

– адміністративна панель (управління позиціями меню, перегляд та зміна статусів замовлень);

– AI-модуль рекомендацій (генерація персоналізованих піц на основі вподобань користувача);

– історія замовлень (збереження та перегляд попередніх замовлень у боті).

Окрім основної функціональності, важливо забезпечити якість, стабільність та безпеку роботи системи. Нефункціональні вимоги визначають технічні та експлуатаційні характеристики розробки, такі як швидкодія, навантаженість, масштабованість, захист даних, сумісність із технологіями та зручність адміністрування. Вони є основою для довготривалої підтримки, розширення та безперебійної роботи Telegram-бота у реальних умовах. Під час проєктування даного програмного продукту було сформульовано такі нефункціональні вимоги:

– відповідь бота на запити користувача повинна надходити не пізніше ніж за 2 секунди у 95 % випадків;

– система повинна підтримувати розподілену базу даних і забезпечувати горизонтальне масштабування за допомогою Docker;

– передбачено використання SSL, хешування токенів і обмеження доступу до адміністративної панелі для забезпечення безпеки;

– необхідно реалізувати механізм повтору запитів і логування у разі помилок під час оплати або збоїв бази даних;

– інтерфейс повинен бути інтуїтивно зрозумілим, з можливістю повернення до попереднього кроку, за потреби підтримувати кілька мов;

– передбачити збір логів роботи сервісу та платіжних операцій для подальшого аналізу та моніторингу.

Для більшої зрозумілості архітектуру самої системи було обрано мову моделювання UML, яка є досить популярною та широко використовною для створення наочних діаграм, які служать для опису структури об’єктів, їх взаємодії, а також різні варіанти поведінки системи [6].

Для того, щоб візуального зобразити розроблювальну систему було розроблено діаграму класів, використання та послідовності, оскільки вони дозволяють глибоко моделювати структуру, зв’язки, функції та логіку взаємодії об’єктів Telegram-бота з функціоналом онлайн-замовлень та адміністрування.

Діаграма класів зображує взаємозв’язки між основними сутностями системи онлайн-замовлення їжі (рис. 2.1). Вона відображає логіку обробки замовлень, оплати, управління меню та адміністрування.

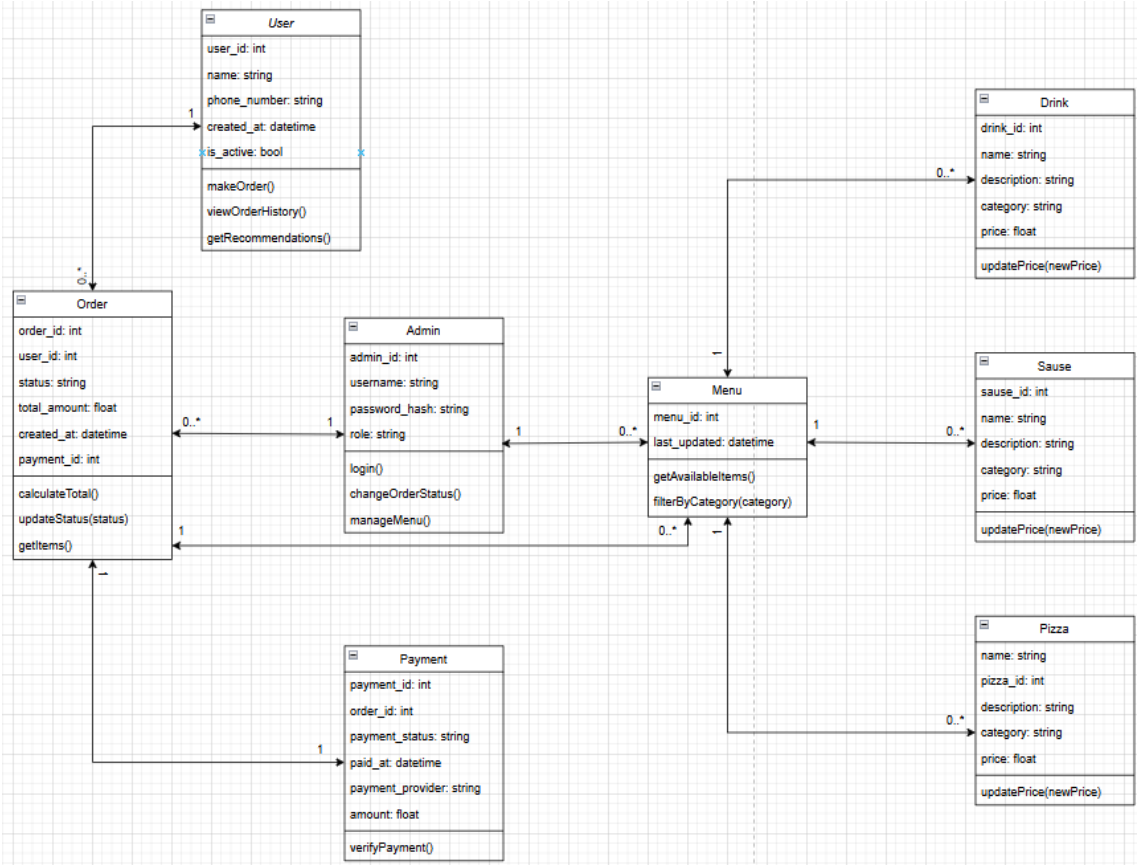

Рисунок 2.1 – Діаграма класів

Взаємодія користувача з системою реалізується через створення нового замовлень і перегляд історії попередніх замовлень, тоді як адміністратор забезпечує контроль за статусами замовлень та актуальністю меню. Меню структурується за категоріями, що дозволяє ефективно та зручно організувати вибір позицій для користувача. Уся система побудована з чіткими зв'язками сутностями, які демонструють потоки інформації та логічні залежності, необхідні для коректного виконання бізнес-процесів системи. Система також передбачає механізм сповіщень клієнтів про зміну статусу їх замовлень і може автоматично обробляти повернення платежів у разі скасування.

Діаграма використання наочно демонструє, як два актори, а саме «Користувач» і «Менеджер/Адміністратор», взаємодіють із Telegram-ботом замовлення піци (рис. 2.2).

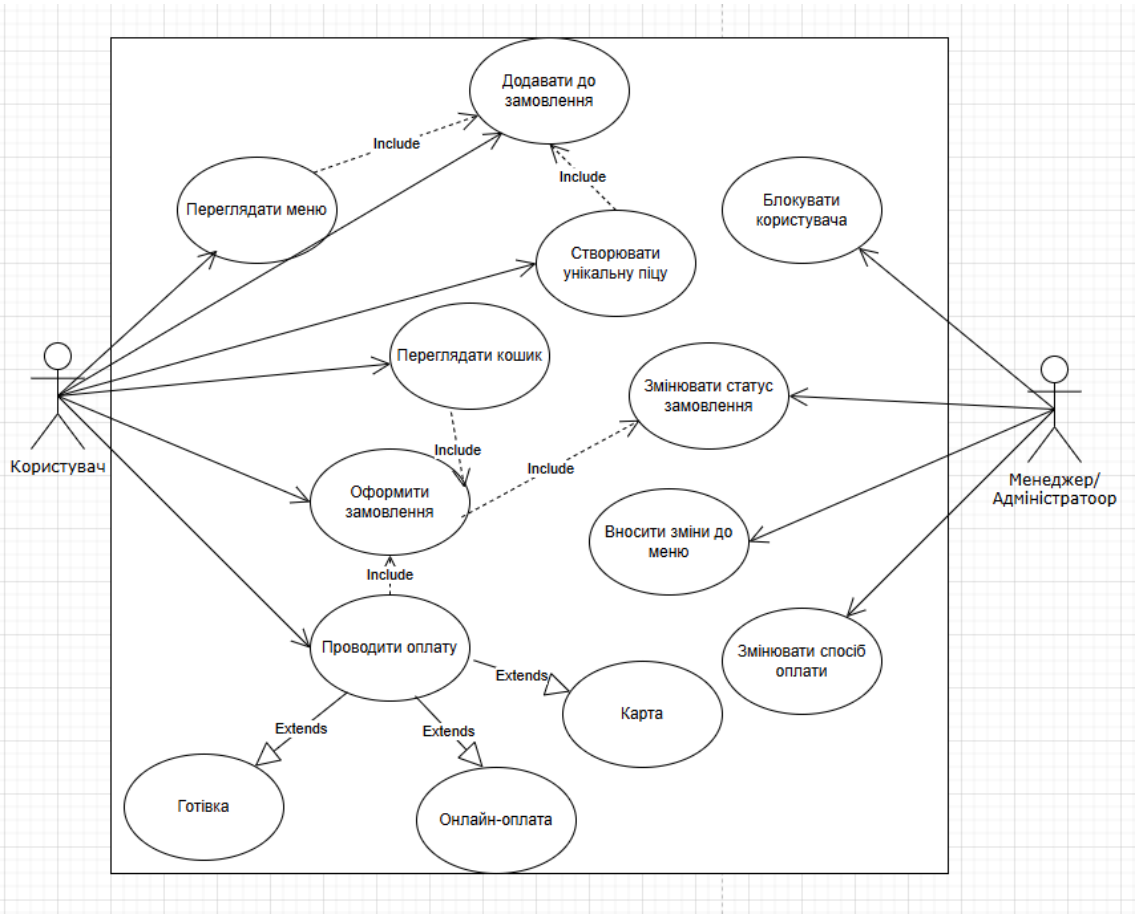

Рисунок 2.2 – Діаграма використання

«Користувач» має можливість переглядати меню, а також створювати свою індивідуальну піцу, після чого додати готові або унікальні піци до замовлення, переглядати кошик, оформляти та оплачувати його зручним для себе способам (готівка, онлайн чи карткою). Після успішної оплати клієнтом, він отримує чек в якому вказується номер замовлення, дата та сума, також користувач має змогу переглянути маршрут доставки з орієнтовним часом прибуття кур’єра.

В свою чергу «Менеджер/Адміністратор» може блокувати, додавати користувачів, змінювати статус замовлень, редагувати меню (піц, напоїв або

соусів) та способи оплати, забезпечуючи контроль і гнучкість управління сервісом.

Після визначення статичних взаємодій у діаграмі випадків використання, ця діаграма послідовності була створена, щоб деталізувати реальний хід процесу замовлення (рис. 2.3), від перегляду меню та формування унікальної піци із залученням ШІ до додавання у кошик, через генерацію інвойсу та онлайн-оплату з отриманням чеку, до зміни статусу «Оплачено» і сповіщення користувача, а також паралельних дій адміністратора з блокування чи редагування меню – таким чином забезпечується чітке розуміння кроків і обміну повідомленнями між усіма компонентами системи.

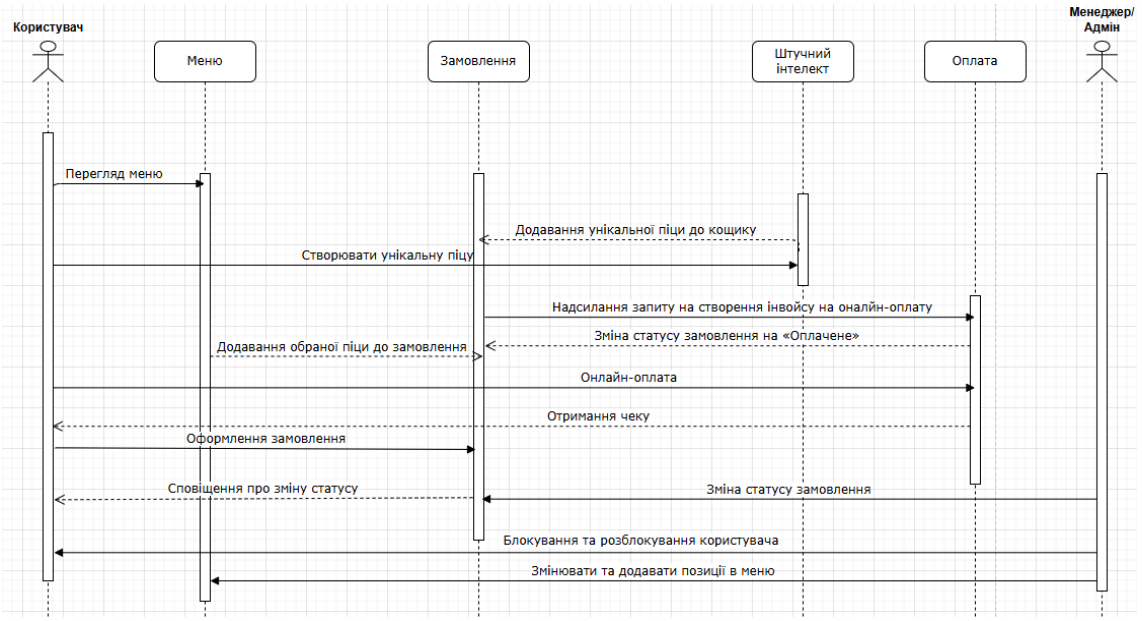

Рисунок 2.3 – Діаграма послідовності

Розробка UML-діаграм дала можливість чітко спроєктувати базові сценарії взаємодії користувача з чат-ботом та продемонструвати їх. Це моделювання дозволило на ранніх етапах виявити логічні несумісності та доповнити функціональні вимоги, мінімізувати ризик передчасного переписування коду, що забезпечило прозорість процесу розробки та спростило подальшу підтримку проєкту.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

На основі проведеного аналізу у розділі 1 було обрано месенджер Telegram, в межах якої буде реалізовуватись система. Telegram надає розробникам потужний набір інструментів через Telegram Bot API, який постійно оновлюється та підтримується. API добре задокументований, має низький поріг входу й дозволяє реалізувати інтерактивну логіку, включно з інлайн-кнопками, клавіатурами, мультимедійним контентом, платіжною інтеграцією, webhook-подіями тощо. Інші месенджери (наприклад, WhatsApp чи Viber) мають обмежений або платний доступ до подібного функціоналу, або вимагають додаткової сертифікації/використання сторонніх сервісів

Telegram доступний на всіх основних платформах: Android, iOS, macOS, Windows, Linux, а також через вебінтерфейс. Це забезпечує максимальне покриття користувачів без потреби у встановленні спеціального додатка.

Telegram підтримує офіційний механізм прийому платежів у чаті, що дозволяє інтегрувати сервіси, як-от Portmone або Stripe, без залучення сторонніх інтерфейсів. Це значно спрощує процес покупки, підвищує конверсію та довіру до сервісу.

Telegram забезпечує ідеальний баланс між технічною зручністю для розробника, простотою використання для кінцевого користувача та гнучкістю для реалізації як базових, так і розширених функцій. Саме тому він був обраний як основна платформа для створення чат-бота для замовлення доставки піци.

Насамперед для початку розробки необхідно було визначитись з мовою програмування. Серед найпопулярніших мов на яких можна реалізовувати чат-боти в Telegram є – Python із його бібліотеками python-telegram-bot та aiogram; JavaScript та TypeScript на основі Node.js із пакетами node-telegram-bot-api або telegraf; Go зі вбудованою підтримкою конкурентності через goroutine; Java і Kotlin у поєднанні з Spring Boot; C# із бібліотекою Telegram.Bot на платформі .NET; PHP із Laravel або Symfony і пакетом

php-telegram-bot. Кожна з цих мов має свої переваги, але вибір впав саме на мову програмування Python, завдяки поєднанню простоти та зрозумілості синтаксису, багатої екосистеми та потужних інструментів. Також через зрозумілу і компактну структуру коду навіть початківці швидко опановують основи, а численні готові бібліотеки – зокрема python-telegram-bot і aiogram – дозволяють реалізувати будь-які сценарії: від обробки повідомлень і кнопок до інтеграції з базами даних і зовнішніми API [7]. Вбудована підтримка асинхронного програмування через asyncio гарантує ефективну роботу з великою кількістю одночасних з'єднань і мінімальні затримки, детальна документація й зручні приклади застосування. До того ж Python легко інтегрується з AI-моделями й сервісами машинного навчання, що відкриває безмежні можливості для створення інтелектуальних чат-ботів. Саме вище перераховане стало остаточним у виборі.

Після вибору мови програмування постало питання середовища розробки. На сучасному ринку існує багато різноманітних рішень, кожне з яких має свої переваги та недоліки, серед поширених можна виділити Visual Studio Code, Vim та PyCharm. Visual Studio Code є легким, популярним, розширюваним редактором з великою кількістю плагінів, проте вимагає додаткових налаштувань для повноцінної підтримки Django. Vim це класичний текстовий редактор для термінала, відомий своєю ефективністю та безліччю плагінів для розширення функціональності.

Для роботи над даною системою обрано інтегроване середовище PyCharm, яке надає зрозумілий інтерфейс, зручний функціонал для написання коду, перевірка коду на наявність синтаксичних помилок [8]. Серед плюсів використання PyCharm можна виділити підтримку віртуального середовища це зручно для встановлення бібліотек для проєкту, не вносячи зміни від глобальної системи Python. Також великою перевагою є підказки у вигляді автозаповнення коду, інтеграція з Git що допомагає легше контролювати версії, повноцінна підтримка Django та доступність повного функціоналу за студентською підпискою, яка є безкоштовною.

Python має велику кількість різноманітних бібліотек для розробки Telegram-ботів, наприклад python-telegram-bot із простим синхронним API, aiogram із повною підтримкою asyncio, Telethon для низькорівневого асинхронного доступу до всіх можливостей Telegram API. Вибір на користь aiogram 3 спричинений його переробленою системою хендлерів і маршрутизації, яка спрощує налаштування складних діалогів і керування станами. Також бібліотека aiogram забезпечує асинхронність системи, оскільки вона побудована на основі asyncio [9].

Вибір фреймворку для розробки серверної частини відбувався між трьома найпопулярнішими для Python, Flask, FastAPI та Django. Flask це мікрофреймворк з мінімальним ядром і гнучкою системою розширень. FastAPI особливий тим, що це сучасний асинхронний фреймворк з можливістю автоматичної генерації документації. Для побудови серверної частини найкраще для майбутнього програмного забезпечення підійде фреймворк Django, він досить популярний, оскільки дозволяє швидко створювати вебдодатки та масштабувати їх. Має у своєму арсеналі зручну ORM для роботи з базами даних, вбудовані механізми авторизації, автентифікації та безпеки, зручну адміністративну панель, підтримку шаблонів та багато іншого [10]. Також серед переваг особливу увагу слід приділити тому факту що, PyCharm автоматично розпізнає Django-проєкт і підключає налаштування з файлу settings.py. Середовище розробки ще дозволяє запускати міграції, Django-сервер прямо з редактора. Це значно спрощує та прискорює процес розробки сервісної логіки системи, модулів замовлення, їх оплати та адміністративної панелі в розроблюваному проєкті.

Ключову роль в Telegram-боті для доставки піци відіграє роль база даних, тому цьому вибору було приділено багато уваги. Вибір здійснювався між базами даних з якими ми стикались протягом навчання. PostgreSQL пропонує розширені типи даних, надійні транзакції з MVCC і багаті розширення, натомість потребує складнішого адміністрування й більше ресурсів на старті. MongoDB зберігає дані у вільній схемі JSON-документів, легко масштабується й

чудово підходить для швидких змін структури даних, але має менш строгі механізми валідації й раніше обмежену підтримку багатодокументних транзакцій. MySQL швидкий і простий у налаштуванні, має велику спільноту та безплатний інструментарій, але обмежені можливості аналітики й менш досконала підтримка JSON і транзакцій порівняно з конкурентами [11]. Завдяки ефективному механізму індексування, кешуванню запитів MySQL швидко обробляє велику кількість дрібних і частих запитів, які є дуже характерними для чат-ботів. Також великою перевагою гарантія цілісності даних навіть за високої навантаженості та в разі збоїв сервера. Зважаючи на всі переваги та недоліки та досвід користування було обрано MySQL.

Важливо також обрати надійну та безпечну платіжну систему, серед іноземних або вітчизняних. PayPal, давно став стандартом для міжнародних переказів та онлайн-оплати, Stripe із гнучким API для інтеграції в мобільні й вебдодатки, Google Pay та Apple Pay із підтримкою безконтактних платежів у мобільних пристроях. Серед українських найпопулярніших платіжних систем варто виділити LiqPay від ПриватБанку, яке забезпечує миттєві перекази між картками, WayForPay із гнучкими тарифами та підтримкою усіх основних карткових систем, Portmone – онлайн-сервіс для оплати комунальних та інших рахунків, а також iPay.ua, що пропонує рішення як для малого бізнесу, так і для маркетплейсів із можливістю підключення розстрочки [12]. Під час процесу вибору платіжної системи, було обрано популярну українську платіжну систему Portmone. Обрана платіжна система має високий рівень безпеки, завдяки сучасним механізмам шифрування та сертифікати PCI DSS. Ще одним важливим плюсом Portmone є інтуїтивно зрозумілий інтерфейс та покрокові налаштування, завдяки яким підключення платіжної системи займають мало часу. Саме завдяки високому рівню безпеки, гнучкому налаштуванню платіжних форм, підтримці банківських карток та саме головне прямій інтеграції з месенджером Telegram, а саме з чат-ботами було обрано платіжну систему Portmone.

Для більшої персоналізації та конкурентної спроможності майбутнього розроблювального продукту буде інтегровано штучний інтелект. На ринку існують такі готові рішення для обробки природної мови та побудови чат-ботів: Google Dialogflow (Vertex AI), Amazon Lex, Microsoft LUIS (Azure Cognitive Services), IBM Watson Assistant, open-source Rasa, платформи Cohere та Anthropic (Claude) і сервіси на базі Hugging Face Inference API; я ж обрала OpenAI GPT через високу точність розуміння контексту й намірів користувачів, зрілий та простий у використанні REST-API, відсутність необхідності розгортати власну інфраструктуру та можливість швидкого масштабування під зростаючу аудиторію [13].

Уже в процесі безпосередньої розробки один і той самий функціонал може реалізовуватись різними способами, що викликає потребу в контролі версії для цього буде використовуватись GitHub. Завдяки GitHub можна легко відстежувати всі зміни та працювати з різними гілками [14]. Інтеграції з CI/CD (GitHub Actions) дозволяє автоматизувати збірку, тестування й деплой прямо з репозиторію [15].

Висновки до розділу 2

Проаналізувавши технології, які є на сучасному ринку було прийнято що розробка буде здійснюватись за допомогою використання Python і Django, що забезпечить потужний бекенд із вбудованою ORM, авторизацією та масштабованістю, бібліотеки aiogram 3. Вбудована інтеграція з Telegram Bot API у редакторі коду PyCharm пришвидшить розробку. Для зберігання даних обрана MySQL через її перевірену продуктивність у типових вебсценаріях і легкість інтеграції з Django, платіжний сервіс Portmone гарантує безпечні гривневі операції безпосередньо в чаті, а для більшої персоналізації буде інтегровано штучний інтелект на базі OpenAI GPT, для розпізнавання людської мови та оптимізації процесу створення унікальної піци. Для контролю зміну версій обрано GitHub.

РОЗДІЛ 3
РОЗРОБКА TELEGRAM-БОТУ ДЛЯ ЗАМОВЛЕННЯ ПІЦИ З ІНТЕГРАЦІЄЮ ПЛАТІЖНОЇ СИСТЕМИ ТА AI

3.1 Практична реалізація об’єкта проєктування

Процес розробки Telegram-бота починається зі створення структури Django-проєкту та налаштування віртуального середовища з необхідними бібліотеками (рис. 3.1).

```
aiofiles             24.1.0
aiogram              3.17.0
aiohappyeyeballs     2.4.4
aiohttp              3.11.11
aiosignal            1.3.2
annotated-types      0.7.0
anyio                4.9.0
asgiref              3.8.1
asyncio              3.4.3
attrs                24.3.0
certifi              2024.12.14
charset-normalizer   3.4.1
checkout             0.2.0
colorama             0.4.6
distro               1.9.0
Django               5.1.4
django-unfold        0.44.0
djangorestframework  3.15.2
frozenlist           1.5.0
h11                  0.16.0
sqlparse             0.5.3
tqdm                 4.67.1
typing_extensions    4.12.2
tzdata               2024.2
urllib3              2.3.0
wheel                0.38.4
yarl                 1.18.3
```

Рисунок 3.1 – Список встановлених бібліотек

Ключовими тут є Django 5.1.4 – повноцінний вебфреймворк з ORM, авторизацією та адмінпанеллю; aiogram 3.17.0 – асинхронний фреймворк на базі asyncio для побудови Telegram-ботів із хендлерами й управлінням станами; aiohttp 3.11.11 – асинхронний HTTP-клієнт/сервер для webhook-підключень і

викликів зовнішніх API; та aiofiles 24.1.0 – деблокуючи модуль для роботи з файлами в асинхронному режимі.

Наступним кроком було створення репозиторію на GitHub для того, щоб була можливість відстежувати зміни. Контроль версій дав змогу, повернутись на крок назад, коли щось йшло не по плану.

Щоб середовище вже було повністю готове для безпосередньої розробки чат-боти, необхідно було налаштувати базу даних. Було встановлено драйвер mysqlclient для забезпечення зв’язку з ORM із сервером бази даних. Після успішної інсталяції драйвера у файлі settings.py було налаштовано конфігурацію (ліст. 3.1). Для створення та застосування міграцій використовувались команди python manage.py makemigrations і python manage.py migrate.

Лістинг 3.1. – Конфігурація бази даних MySQL у Django

```
DATABASES = {
  'default': {
    'ENGINE': 'django.db.backends.mysql',
    'NAME': BASE_DIR / 'db.mysql',
  }
}
```

кінець лістингу 3.1

Після того як середовище розробки було налаштоване, необхідно було отримати токен для автентифікації чат-бота перед Telegram Bot API, він слугує секретним ключем, який дозволяє серверу Telegram ідентифікувати запити з коду як запити саме від бота. Токен захищає API від неавторизованих звернень і дає змогу безпечно надсилати та отримувати повідомлення в чатах. Для того, щоб отримати персональний токен потрібно звернутись до офіційного бота @BotFather у самому Telegram. Після відправки команди /newbot і введення назви бота та унікального username, який обов’язково закінчується на «bot» (в моєму випадку це @justp1zza_bot), BotFather згенерував необхідний для початку розробки токен і надіслав його в чаті (рис. 3.2).

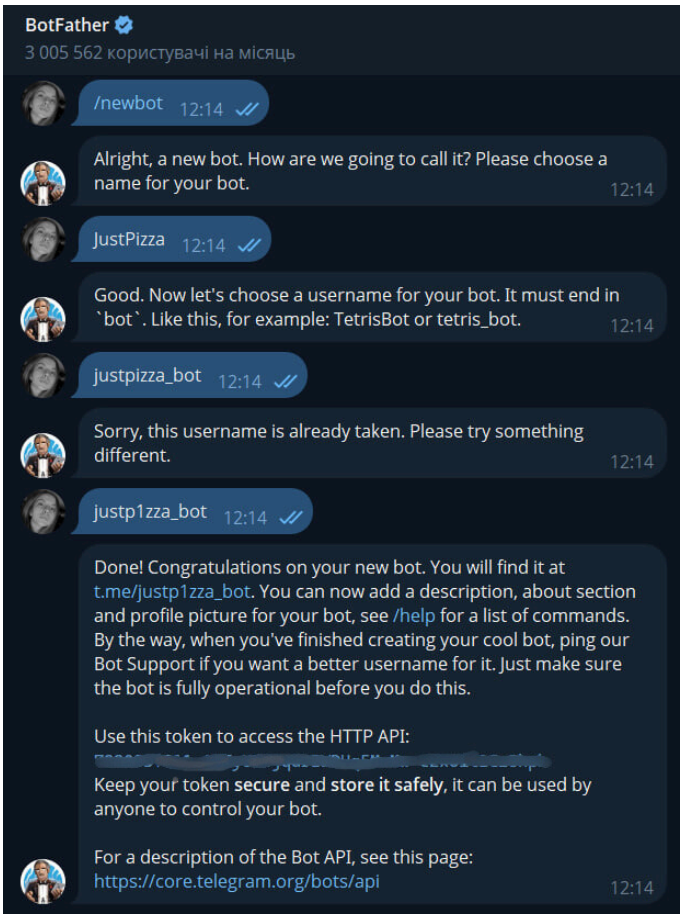

Рисунок 3.2 – Діалог з BotFather

При створенні Django-проєкту автоматично створюється файл manage.py він забезпечує взаємодію з проєктом через командний рядок. Складність викликало одночасно запускати бот та серверну частину через командний рядок. Стандартна команда python manage.py runserver запускала локальний сервер, на якому розгорталась адміністративна панель, при спробі просто додати запуск бота в цей файл виникала помилка, запускалось відразу 2 боти, один з файлу manage.py, а інший з файлу telegram_bot.py. Для того, щоб уникнути цієї проблеми, було прийнято рішення переписати файл manage.py (ліст. 3.2).

Лістинг 3.2. – Запуск бота через manage.py

```
def run_bot():
  os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'bachelor.settings')
  if Path(PID_FILE).exists():
```

```
    with open(PID_FILE, "r") as f:
      try:
        pid = int(f.read().strip())
        if psutil.pid_exists(pid):
          print("Бот уже запущений. Завершіть інший процес перед
стартом.")
          return
      except ValueError:
pass
with open(PID_FILE, "w") as f:
    f.write(str(os.getpid()))
  try:
    from bot.telegram_bot import main as bot_main
    asyncio.run(bot_main())
  finally:
    if Path(PID_FILE).exists():
          os.remove(PID_FILE)
```

кінець лістингу 3.2

Після налагодження та успішного тестового запуску проєкту було написано перше повідомлення бота, після того, як користувач натисне команду /start (рис. 3.3).

Рисунок 3.3 – Перше повідомлення

Для зручності користування чат-ботом, після цього повідомлення в користувача з'являється ReplyKeyboardMarkup, це клавіатура з шести кнопок яка знаходиться знизу, а не відправляється з повідомленням від бота (рис. 3.4).

Рисунок 3.4 – Головне меню

При натиску на кнопку «Меню», користувачу відкриється меню з підкатегоріями: «Піца», «Напої» та «Соус». Щоб оптимізувати об'єм меню, при відкриванні будь-якої з категорій відкривається перелік позицій з назвою та ціною у вигляді кнопок, якщо користувач зацікавився тією чи іншою позицією він може дізнатись про неї детальніше натиснувши на неї після чого йому відкриється детальна інформація з фото та описом (ліст. 3.3).

Лістинг 3.3 – Детальна інформація про піцу

```
async def send_item_details(callback_query: CallbackQuery,
model_class, item_type: str):
  await callback_query.answer()
  try:
    data_parts = callback_query.data.split("_")
    if item_type not in ["pizza", "drink", "sauce"]:
      await callback_query.answer(
        "Невідома категорія. Спробуйте ще раз.", show_alert=True
      )
      return
    item_id = int(data_parts[2])
    item = await sync_to_async(model_class.objects.get)(
      id=item_id
    )
    quantity = 1
    if "|" in callback_query.data:
      cart_total = callback_query.data.split("|")[1]
    else:
      cart_total = "0"
    text = (
      f"**{item.title}**\n\n"
```

```
      f"{item.description}\n\n"
      f"💵 Ціна: {item.price} грн\n"
    )
    if item.photo and item.photo.path:
      photo = FSInputFile(item.photo.path)
      await callback_query.message.bot.send_photo(
        chat_id=callback_query.message.chat.id,
        photo=photo,
        caption=text,
        parse_mode="Markdown",
        reply_markup=keyboard,
      )
    else:
      await callback_query.message.answer(
        text, parse_mode="Markdown", reply_markup=keyboard
      )
  except Exception as e:
    await callback_query.answer("Сталася помилка. Спробуйте ще раз
пізніше.", show_alert=True
    )
```

кінець лістингу 3.3

Наступним кроком необхідно було написати функціонал для додавання товарів до корзини. Спочатку було зроблено, щоб користувач міг просто додавати, наприклад піцу, до кошика, але в ході розробки було вирішено додати логіку і для віднімання товару або скидання до початкової кількості, тобто до 1 (ліст. 3.4).

Лістинг 3.4 – Реалізація логіки редагування кількості товару

```
@dp.callback_query(lambda callback:
callback.data.startswith("increase_"))
async def handle_increase(callback_query: types.CallbackQuery,
state):
  cart = await get_or_create_cart(callback_query.from_user.id)
  await handle_callback(
    callback_query,
    state,
    cart = cart,
    model_classes ={ "pizza": PizzaList, "drink": DrinkList,
"sauce": SauceList},)
@dp.callback_query(lambda callback:
callback.data.startswith("decrease_"))
async def handle_decrease(callback_query: CallbackQuery, state):
  cart = await get_or_create_cart(callback_query.from_user.id)
  await handle_callback(
```

```
    callback_query,
    state,
    cart = cart,
    model_classes ={ "pizza": PizzaList, "drink": DrinkList,
"sauce": SauceList},)
```

кінець лістингу 3.4

Також одним з найбільш затратним по часу розробки, безпосередньо що стосується меню, було додавання можливості додавати або редагувати позицію з меню. Для цього було виведено моделі в файлі admin.py, щоб вони відображались в адміністративній панелі (рис. 3.5).

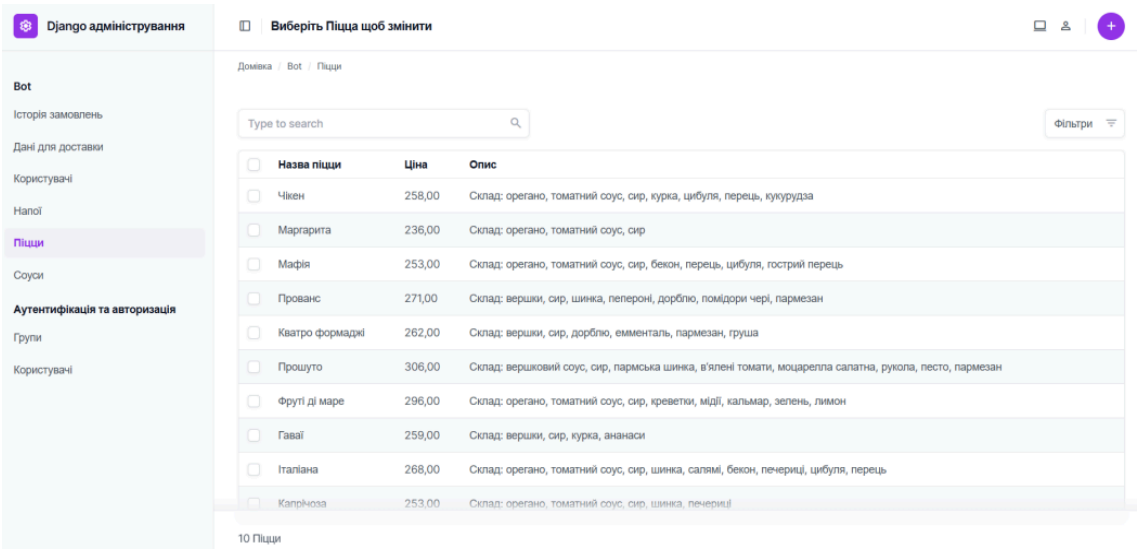

Рисунок 3.5 – Вигляд адмін панелі

Зліва зверху відображаються моделі бази даних які відображаються в адмінпанелі (рис. 3.5). Коли адміністратор захоче внести зміни вже існуючу позицію меню або додати нову, йому необхідно буде натиснути на позицію або натиснути плюсик зверху зправа, відповідно, для того, щоб заповнити необхідні поля (ліст. 3.5).

Лістинг 3.5 – Модель Pizza List для збереження інформації в базі даних

```
class PizzaList(models.Model):
  title = models.CharField(max_length = 255, verbose_name = "Назва
піцци")
```

```
  price = models.DecimalField(max_digits = 10, decimal_places = 2,
verbose_name = "Ціна")
  description = models.CharField(
    max_length = 255, verbose_name = "Опис", default = "Опис
відсутній"
  )
  photo = models.ImageField(upload_to = "pizzas/", null = True,
blank = True)
  class Meta :
    verbose_name = "Піцца"
    verbose_name_plural = "Піцци"
  def __str__(self):
    return f"{self.title} - {self.price} грн"
```

кінець лістингу 3.5

Після того як користувач обрав для себе позицію та її кількість, він натискає на кнопку Додати також на цій кнопці додатково вказується кількість товару та сума. Якщо процес успішний, то користувачу вискакує повідомлення (рис. 3.6).

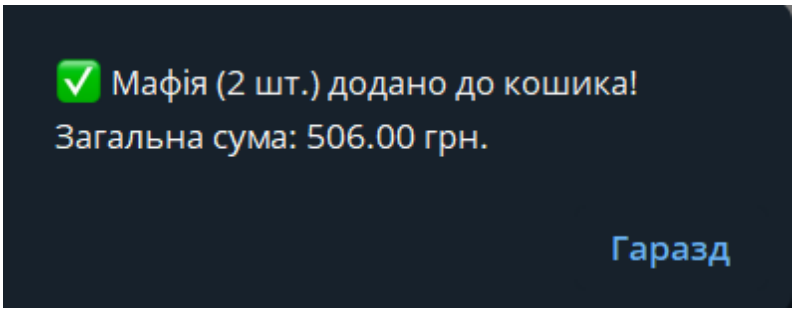

Рисунок 3.6 – Повідомлення про успішне додавання в кошик

У чат-боті для замовлення доставки піци важливим є кошик користувача, у якому тимчасово зберігаються обрані товари перед оформленням замовлення. Для реалізації цієї логіки були створені дві взаємопов’язані моделі: Cart та CartItem.

Модель CartItem описує окрему позицію в кошику. Кожен екземпляр цієї моделі відповідає одному конкретному товару (наприклад, піці, напою чи соусу), який користувач додав до кошика. Вона містить такі поля, як назва товару, ціна, кількість та категорія. Натомість модель Cart представляє сам

кошик у цілому – тобто об’єднує усі CartItem-елементи, які належать певному користувачу. Вона містить посилання на користувача (через його telegram id), загальну суму замовлення, дату створення та оновлення кошика, а також прапорець is_active, який визначає, чи є кошик поточним (активним).

Таким чином, один об’єкт Cart може містити кілька об’єктів CartItem, і разом вони формують повноцінну структуру кошика замовлень. Такий підхід дозволяє обробляти тимчасові дані про вибрані товари, наприклад редагувати безпосередньо в корзині або видалити, якщо користувач передумав замовляти той чи інший товар (рис. 3.7), до моменту остаточного підтвердження та оплати замовлення.

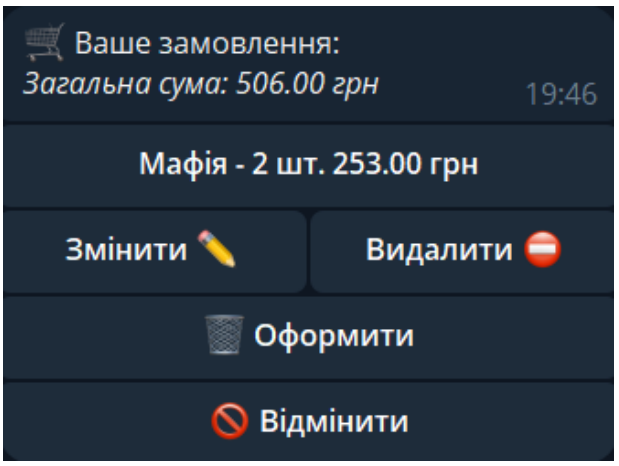

Рисунок 3.7 – Вигляд кошику

Використовуючи будь-який спосіб доставки найважливішим є етап оформлення замовлення.

Перед оформленням замовлення користувачеві пропонується ввести основні контактні дані: ім’я, адресу доставки та номер телефону. Ці дані зберігаються у моделі BotUser. Бот послідовно запитує кожне поле, забезпечуючи правильний формат введення. Після введення контактної інформації користувач обирає спосіб оплати. Доступні варіанти:

– готівка при отриманні;

– картка при доставці;

– онлайн-оплата (через інтегровану платіжну систему).

Після успішного оформлення замовлення бот надсилає підтвердження з номером замовлення, сумою та обраним способом оплати. У разі онлайн-оплати користувачу може бути надіслане посилання на сторінку оплати або використано Telegram Payments API.

Важливим етапом оформлення замовлення в Telegram-боті є здійснення онлайн-оплати безпосередньо в межах месенджера. У даній розробці для цього використано офіційний механізм Telegram Payments, а як платіжний провайдер інтегровано український сервіс Portmone.

Коли користувач обирає спосіб оплати «Онлайн», бот генерує рахунок за допомогою методу send_invoice, у якому вказуються контактні дані та дані про замовлення (ліст. 3.6).

Лістинг 3.6 – Формування та надсилання інвойсу Telegram-ботом для онлайн-оплати

```
if payment_text == "онлайн":
  price = LabeledPrice(label = "Оплата замовлення", amount =
int(cart.total_sum * 100))
  await bot.send_invoice(
    message.chat.id,
    title = "Оплата замовлення",
    description = "Оплата замовлення через Telegram",
    provider_token = "1661751239:TEST:91f0-pQ5h-f8Z4-FcOD",
currency = "uah",
    is_flexible = False,
    prices =[price],
    start_parameter = "test-start",
    payload = "order-online",
  )
```

кінець лістингу 3.6

Перед завершенням транзакції Telegram надсилає pre_checkout_query, на який обов’язково потрібно відповісти ok=True, інакше платіж не пройде (ліст. 3.7).

Лістинг 3.7 – Обробка PreCheckoutQuery та підтвердження платежу в Telegram-боті

```
@router.pre_checkout_query()
async def process_pre_checkout_query(pre_checkout_query:
PreCheckoutQuery):
    await bot.answer_pre_checkout_query(pre_checkout_query.id, ok =
True)
```

кінець лістингу 3.7

Після того як платіж успішно пройшов на місці повідомлення про необхідність провести оплату з’являється чек (рис. 3.8).

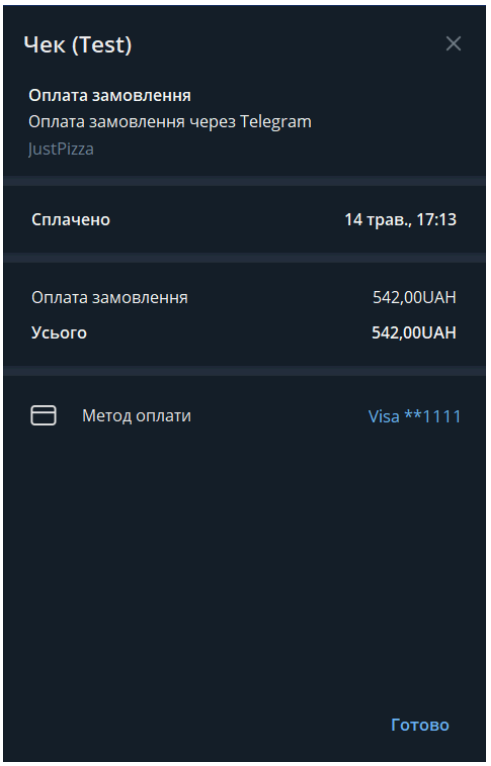

Рисунок 3.8 – Чек про оплату замовлення

На стороні бота в цей момент, зберігаються дані замовлення до бази даних, статус корзини змінюється на «неактивна», для користувача приходить повідомлення про успішну оплату та деталями замовлення (рис. 3.9), додається замовлення до історії, щоб користувач міг надалі переглянути.

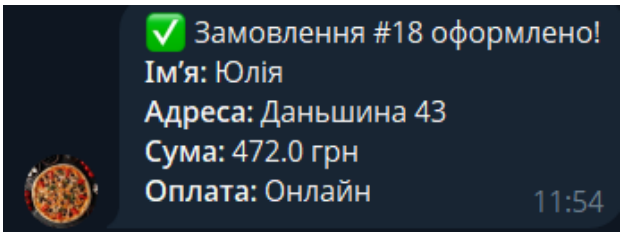

Рисунок 3.9 – Повідомлення про оформлення замовлення

Описаний вище процес є стандартним сценарієм взаємодії користувача з Telegram-ботом під час оформлення онлайн-замовлення. Якщо користувач обере для себе такий спосіб оплати як готівка чи картою при доставці, то йому прийде аналогічне повідомлення лише в полі оплата буде вказуватись актуальний спосіб.

Особливістю даного чат-бота є інтеграція зі штучним інтелектом. У бот було впроваджено модуль взаємодії з моделлю штучного інтелекту GPT, зокрема GPT-4o-mini від OpenAI, для генерації персоналізованої піци на основі вподобань користувача та обраних інгредієнтів. Така інтеграція дозволяє зробити процес замовлення інтерактивним, залучаючи користувача до персоналізованого досвіду створення власної піци.

Процес побудовано на основі Finite State Machine (FSM) – машини станів, де для взаємодії з користувачем було визначено окрему групу станів PizzaAI. Вона включає етапи вибору соусу, введення побажань, підбору інгредієнтів, комунікації з ШІ та формування підсумкового результату (ліст. 3.8).

Лістинг 3.8 – Визначення станів FSM для модуля PizzaAI у Telegram-боті

```
class PizzaAI(StatesGroup):
  waiting_for_preferences = State()
  choosing_sauce = State()
  collecting_additions = State()
  conversation = State()
       summarizing = State()
```

кінець лістингу 3.8

У межах реалізації Telegram-бота для замовлення піци ставиться завдання максимально спростити процес формування переліку інгредієнтів на основі довільного тексту, введеного користувачем. Традиційний підхід із лематизацією та fuzzy matching вимагає налаштування словника, високих обчислювальних витрат і не гарантує повного охоплення синонімів та помилок користувача. Тому використано GPT-4o-mini як інструмент для аналізу природної мови та витягання назв інгредієнтів у форматі JSON. Для уникнення «вигадування» зайвих позицій реалізована ще ручна перевірка, це дає змогу надати користувачеві максимально точний список інгредієнтів, адаптований як до особистих смакових вподобань, так і до фактичної наявності продуктів на складі. Користувач бачить попередній перелік, може відразу відкоригувати його, видаливши зайві чи додати відсутні пункти.

Штучний інтелект в даному проєкті відповідає за аналіз повідомлень користувача, де він описує побажання, та за формування унікальної назви та опису щойно створеної піци. Основу цієї логіки складає функція extract_ingredients(...), яка аналізує введений текст за допомогою методів обробки природної мови (NLP) і співвідносить знайдені слова з наявними компонентами з бази даних (ліст. 3.9). Вона дозволяє користувачеві писати побажання у довільній формі, наприклад: «додай бекон і гриби» або «- цибуля та перець», а бот, за допомогою штучного інтелекту, розпізнає ці інгредієнти та виконає відповідні дії. Завдяки цьому інтерфейс стає більш гнучким і наближеним до живого спілкування, що суттєво покращує користувацький досвід.

Лістинг 3.9 – Обробка введених інгредієнтів та оновлення списку додатків у Telegram-боті

```
found = extract_ingredients(user_input, component_map)
  if found:
    new_items = [i for i in found if i not in additions_list]
    additions_list += new_items
    await state.update_data(additions_list = additions_list)
    added_str = ", ".join(f"«{i}»" for i in new_items)
  await message.answer(f"Інгредієнти {added_str} додано до піци 🍕")
```

```
    else:
        await message.answer("Не вдалося знайти жодного інгредієнту у вашому повідомленні 😢")
```

кінець лістингу 3.9

Якщо такі інгредієнти знайдено є в піцерії, вони додаються до замовлення. У випадку, якщо користувач вводить інгредієнт, якого не існує, або робить орфографічну помилку, AI намагається максимально точно ідентифікувати введення та порівняти його з доступними варіантами. Така обробка даних гарантує, що до фінального замовлення потрапляють лише ті компоненти, які реально доступні у піцерії, що забезпечує точність і зручність під час формування піци (рис. 3.10).

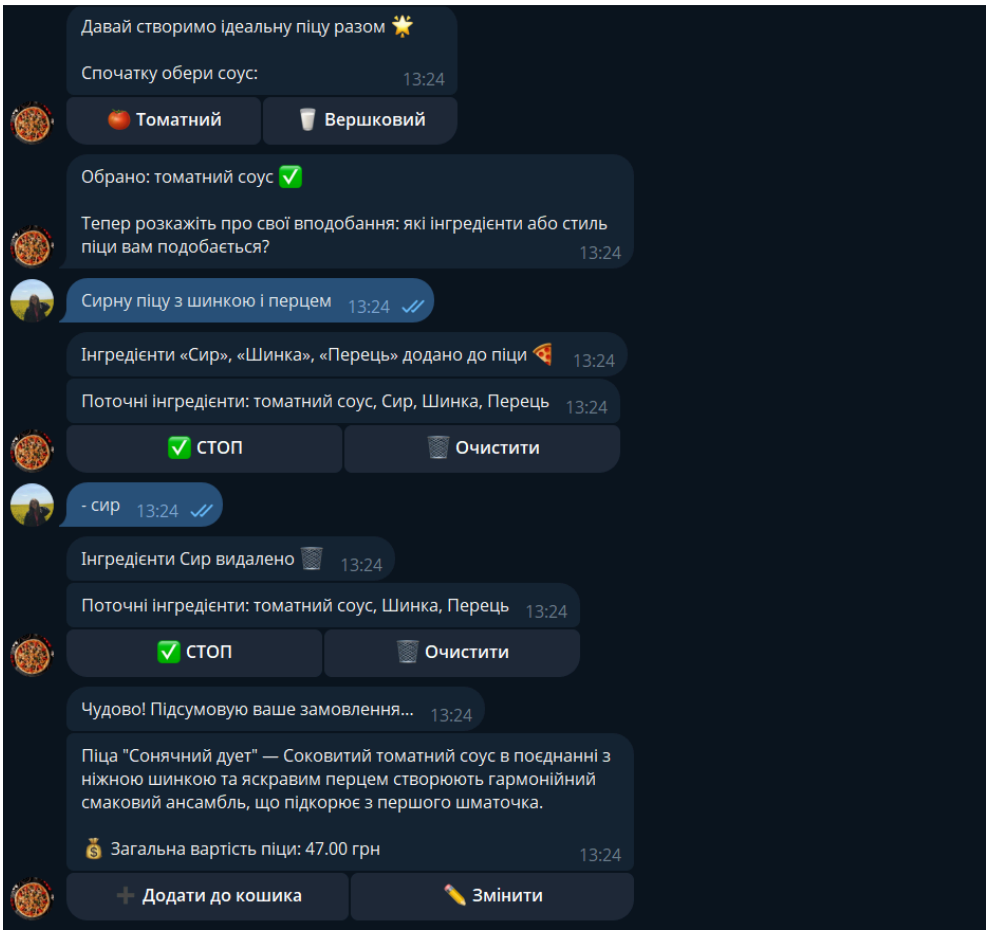

Рисунок 3.10 – Інтерфейс діалогу користувача з ШІ

Щоб зробити більший персоналізований підхід до користувача було реалізовано інтерактивну карту. Для того, щоб дізнатись час доставки піци, користувачу потрібно натиснути кнопку «Доставка» та ввести адресу, після чого йому прийде повідомлення з відстанню та орієнтовним часом прибуття кур'єра, а також картинкою на якій буде побудований маршрут від піцерії до будинку вказаної адреси (рис. 3.11). Цей маршрут визначається за допомогою функції генерування найближчого та найменш навантаженого шляху, щоб зменшити час очікування для користувача.

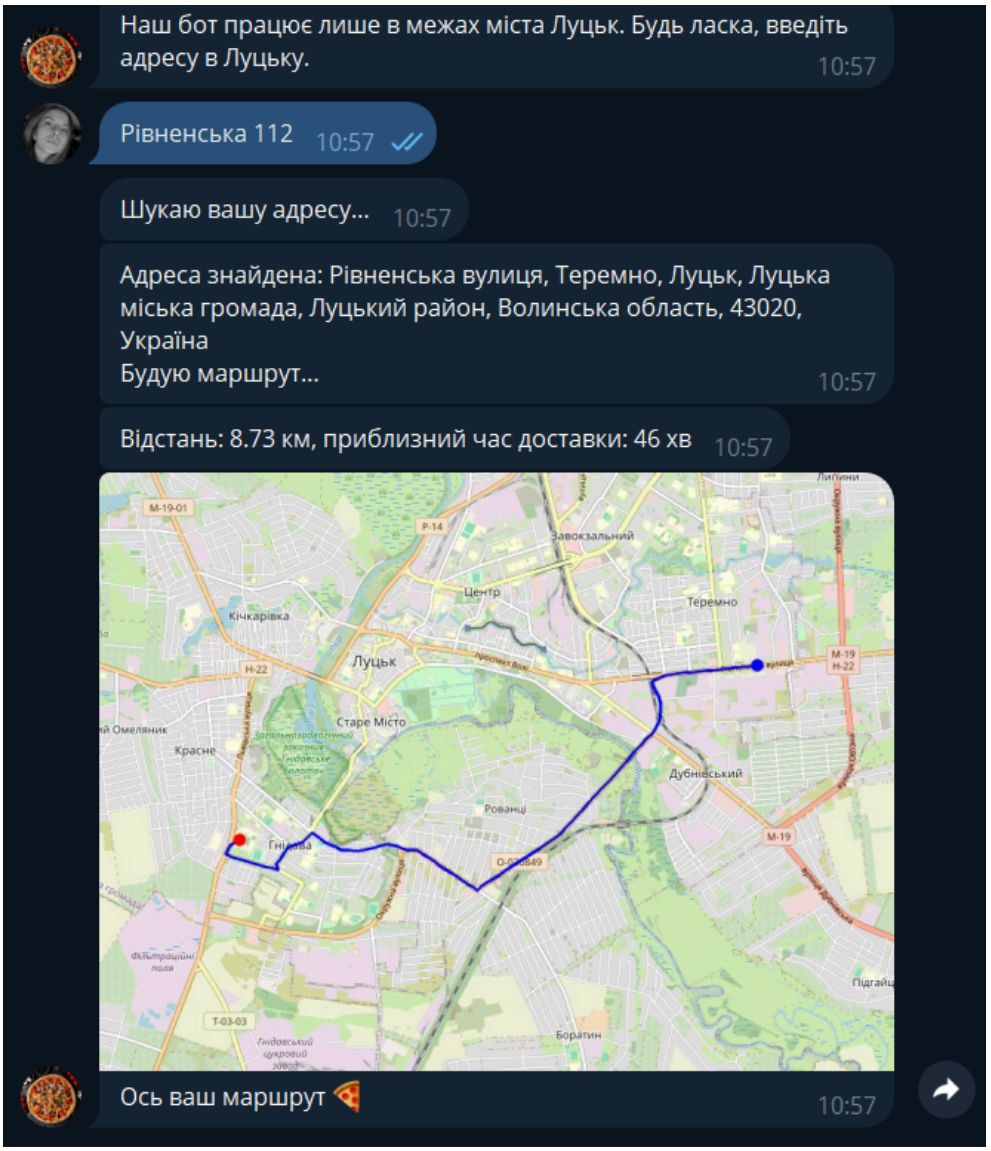

Рисунок 3.11 – Інтерфейс Telegram-бота з побудованим маршрутом доставки піци від піцерії до вказаної адреси користувача

В рамках розробки Telegram-бота було реалізовано гнучкий та інтуїтивно зрозумілий механізм взаємодії з користувачем завдяки поєднанню Python, Django, бази даних MySQL та елементів штучного інтелекту. Особливу увагу було приділено системі обробки побажань щодо інгредієнтів піци, яка здатна розпізнавати запити у природній мові, аналізувати їх, перевіряти наявність компонентів у базі та формувати персоналізоване замовлення. Це дозволяє досягти високої якості користувацького досвіду та адаптивності системи під різні сценарії взаємодії.

В майбутньому планую реалізувати акційні пропозиції, якими можна керувати з адміністраторської панелі, а також накопичувальну систему бонусів. Не завадило також би додати фільтрацію основного меню, безпосередньо в боті.

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

У процесі розробки Telegram-бота було проведено ручне (мануальне) тестування, яке є невід’ємною частиною забезпечення якості програмного продукту. На відміну від автоматизованого підходу, мануальне тестування передбачає безпосередню взаємодію тестувальника з інтерфейсом системи з метою виявлення помилок, недоліків у логіці та перевірки зручності користування.

Для того, щоб тестування було максимально продуктивним та об’єктивним щодо інтерфейсу, було залучено декілька тестувальників із різним рівнем технічної підготовки – як користувачів, знайомих із Telegram-ботами, розробників таких чат-боті, так і тих, хто вперше взаємодіє з подібними системами.

Під час тестування створення індивідуальної піци було виявлено, що у випадках, коли користувач вводив інгредієнт, якого не існує в базі піцерії, штучний інтелект все одно включав його до підсумкового опису піци (рис. 3.12). Це відбувалося через те, що механізм генерації опису не перевіряв наявність

компонентів у доступному списку, а працював лише з тим, що було передано на основі введення користувача.

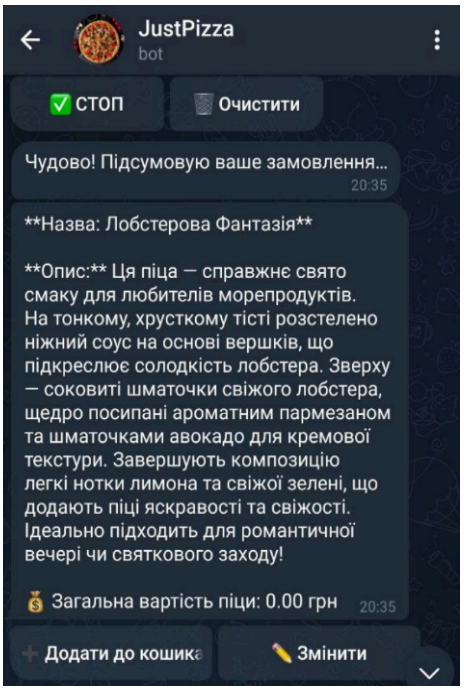

Рисунок 3.12 – Некоректний підсумок ШІ

Також більшість користувачів, які тестували бот, зауважили на, тому що великою перевагою було б отримання повідомлень щодо статусу замовлень (рис. 3.13).

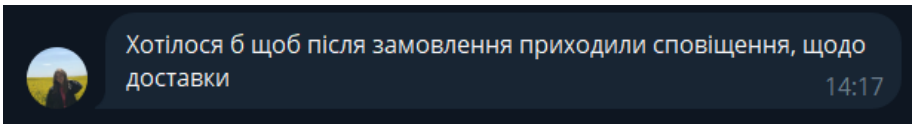

Рисунок 3.13 – Побажання потенційних користувачів

Це було реалізовано наступним чином, в клас OrderHistory було додане поле STATUS_CHOICES, яке адміністратор міг змінювати з випадаючого списку прямо з ORM Django. Тепер після зміни статусу замовлення бот надсилає користувачу повідомлення (ліст. 3.10).

Лістинг 3.10 – Визначення переліку статусів замовлення (STATUS_CHOICES)

```
STATUS_CHOICES = [
  ("paid for", "Оплачено"),
  ("new", "Новий"),
  ("pending", "Очікує підтвердження"),
  ("preparing", "Готується"),
  ("shipped", "В дорозі"),
  ("completed", "Завершено"),
  ("canceled", "Скасовано"),
  ]
```

кінець лістингу 3.10

Протягом тестування також оцінювалась коректність переходу між станами бота, починаючи від вибору соусу до формування фінального замовлення. Було важливо, щоб бот не затримувався між станами та завжди реагував на користувацькі дії згідно логіки сценарію.

Тестувалась коректність зчитування та оновлення даних про інгредієнти з MySQL через ORM Django. Також перевірялась стійкість системи до ситуацій, коли в базі відсутній зазначений інгредієнт.

Під час процесу тестування було прийняте рішення встановити чіткі години роботи, щоб бот не формував замовлення у не робочий час (ліст. 3.11).

Лістинг 3.11 – Функція формування повідомлення про графік роботи

```
def get_off_hours_message() -> str:
days_ua = {
  "Monday": "Понеділок",
    "Tuesday": "Вівторок",
    "Wednesday": "Середа",
    "Thursday": "Четвер",
    "Friday": "П'ятниця",
    "Saturday": "Субота",
    "Sunday": "Неділя",
  }
chedule_lines = []
  for eng_day, hours in working_hours.items():
    day = days_ua[eng_day]
    if hours:
      schedule_lines.append(f"{day}: {hours[0]}–{hours[1]}")
    else:
      schedule_lines.append(f"{day}: вихідний")
```

```
  schedule_text = "\n".join(schedule_lines)
      return "⚠️ Заклад зараз зачинений.\n" "🕘 Графік роботи:\n" f
"{schedule_text}\n\n"
```

кінець лістингу 3.11

Було проведено оптимізацію коду за допомогою JetBrains AI Assistant у PyCharm, а саме прибрано дублювання коду, зайві імпорти це допомогло зробити код зробити чистішим та ефективнішим. За рекомендаціями штучного інтелекту було витягнуто повторювані фрагменти в окремі функції, наприклад клавіатури, логіка обчислення кількості товару та інші.

Крім мануального тестування було проведене автоматичне тестування за допомогою unit-тестів. Спочатку було необхідно ключову бізнес-логіку (обробку команд, формування відповіді, валідацію даних) в окремі функції, щоб їх можна було викликати без запуску самого бота. Для кожної такої функції я написала набір тестів за допомогою бібліотеки pytest (ліст. 3.12). Вона перевіряла, що при різних вхідних повідомленнях вона повертає саме ту структуру даних або текст, який очікую.

Лістинг 3.12 – Приклад unit-тесту

```
import pytest
from unittest.mock import AsyncMock, MagicMock, patch
from .telegram_bot import handle_add_to_cart
@pytest.mark.asyncio
async def test_handle_add_to_cart_success(monkeypatch):
  fake_query = AsyncMock()
  fake_query.data = "add_to_cart_pizza_1"
  fake_query.from_user.id = 42
  fake_button = MagicMock()
  fake_button.text = "<< Кошик (0 грн)"
  fake_query.message.reply_markup.inline_keyboard = [fake_row]
  fake_query.message.edit_reply_markup = AsyncMock()
  fake_query.answer = AsyncMock()
  fake_state = AsyncMock()
  fake_state.get_data = AsyncMock(return_value={})
  fake_item = MagicMock(title="TestPizza", price=100)
      monkeypatch.setattr(yourmodule, "sync_to_async", lambda fn:
AsyncMock(return_value=fn()))
  class FakePizzaList:
    objects = MagicMock(get=MagicMock(return_value=fake_item))
```

```
          monkeypatch.setitem(yourmodule.model_classes,     "pizza",
FakePizzaList)
    fake_cart  =  MagicMock(id=7,  items=MagicMock(add=MagicMock()),
total_sum=500)
           monkeypatch.setattr(yourmodule,     "get_or_create_cart",
AsyncMock(return_value=fake_cart))
  fake_cart_item = MagicMock(quantity=1, asave=AsyncMock())
  monkeypatch.setattr(
    yourmodule.CartItem.objects,
    "get_or_create",
    MagicMock(return_value=(fake_cart_item, True)),
  )
  monkeypatch.setattr(yourmodule, "update_cart_total", AsyncMock())
  monkeypatch.setattr(asyncio, "sleep", AsyncMock())
  await handle_add_to_cart(fake_query, fake_state)
  fake_query.message.edit_reply_markup.assert_awaited()
  fake_query.answer.assert_awaited_with(
      "✅ TestPizza (1 шт.) додано до кошика! Загальна сума: 500
грн.",
    show_alert=True
     )
```

кінець лістингу 3.12

Також я замінила реальний об’єкт Bot на mock-екземпляр, щоб у тестах не відправляти повідомлення до справжнього Telegram API.

Проведене тестування та налагодження дозволило забезпечити стабільну й надійну роботу системи, підвищити її зручність та адаптивність. Ретельна перевірка всіх складових забезпечує готовність бота до роботи в реальних умовах з високим навантаженням і різними варіантами взаємодії користувачів.

3.3 Проєктування структури бази даних

На етапі розробки інформаційно-комп’ютерної системи важливу роль відіграє створення надійної та структурованої бази даних, яка забезпечує збереження, обробку та зв’язок усіх основних елементів системи. У рамках проєкту для Telegram-бота з оформлення замовлення піци було реалізовано базу даних за допомогою системи управління базами даних MySQL, з використанням інструментів терміналу середовища розробки PyCharm та інтеграції з ORM Django.

Реляційна структура MySQL дозволяє ефективно зберігати пов’язані між собою об’єкти, зокрема піци, компоненти, замовлення та користувачів.

Використання моделей Django для визначення структури таблиць, що спростило міграцію та забезпечило зв’язок між Python-кодом і базою даних.

Для створення та оновлення таблиць, запуску міграцій, перевірки структури бази використовувався термінал PyCharm.

Завдяки вбудованим механізмам Django ORM, реалізовано автоматичну перевірку цілісності даних через зовнішні ключі та унікальні обмеження, що запобігає некоректним зв’язкам. Крім того, через команду manage.py makemigrations та manage.py migrate було легко підтримувати версійність схеми бази даних, а Django admin надає зручний інтерфейс для ручного адміністрування записів без додаткової реалізації CRUD-функціоналу.

Для візуального представлення структури створеної бази даних було побудовано ER-діаграму (Entity-Relationship Diagram), яка демонструє основні таблиці системи, їх атрибути та зв’язки між ними (рис. 3.14).

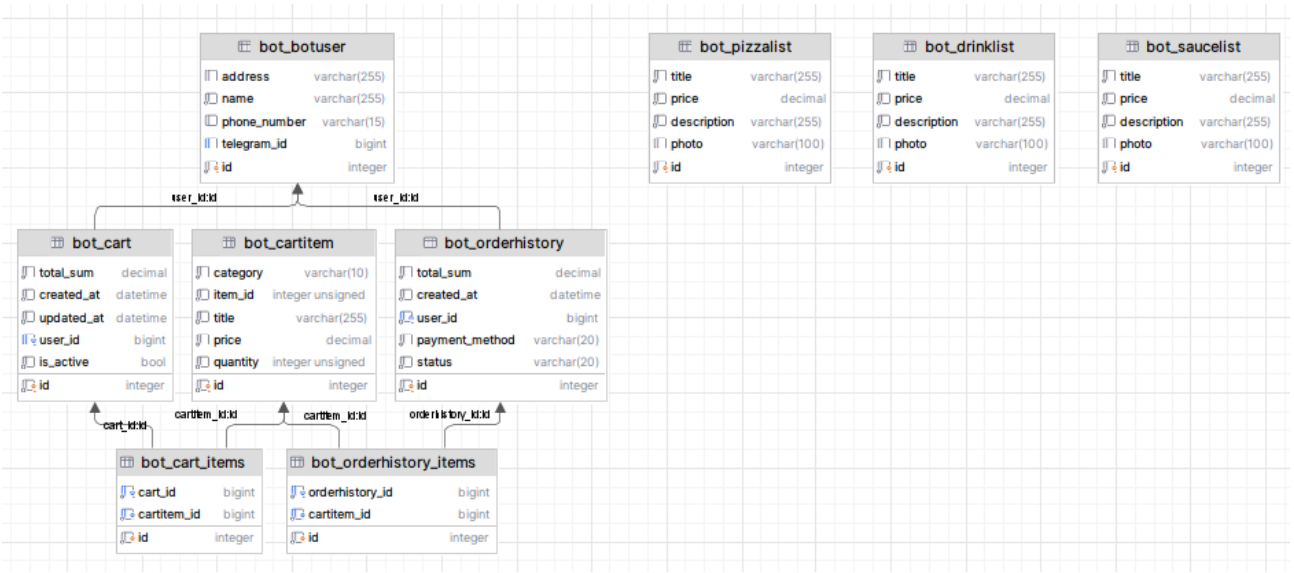

Рисунок 3.14 – ER-діаграма бази даних чат-бота для доставки піци

У розробленій базі даних для Telegram-бота було створено кілька ключових сутностей (таблиць), які забезпечують повноцінну роботу системи –

від збереження даних про піци до обробки замовлень та історії покупок. Нижче наведено основні сутності, їх призначення та взаємозв’язки:

– PizzaList – таблиця для збереження готових позицій піци, що доступні для замовлення. Містить поля для назви, ціни, опису та зображення піци;

– DrinkList – аналогічна таблиця для напоїв, що дозволяє зберігати інформацію про кожен доступний напій;

– SauceList – таблиця для соусів, які користувач може додавати до замовлення. Також містить опис, ціну та фото;

– PizzaComponent – окрема таблиця для компонентів піци (начинки), з можливістю вказати, чи є інгредієнт вегетаріанським. Саме ця таблиця використовується при створенні індивідуальних піц;

– BotUser – зберігає інформацію про користувачів Telegram: Telegram ID, ім’я, прізвище, username та дату останньої активності. Це дозволяє ідентифікувати кожного клієнта системи;

– CustomPizza – таблиця, яка дозволяє користувачам створювати унікальні піци з вибраних компонентів. Має зв’язок із PizzaComponent (багато-до-багатьох) та з BotUser;

– CartItem – елемент корзини, що містить конкретний товар (піцу, напій або соус), його категорію, кількість, ціну та, за потреби, посилання на кастомну піцу;

– Cart – таблиця для зберігання кошика користувача. Містить перелік товарів (CartItem), загальну суму, дату створення та статус активності. Має зв’язок з користувачем і з можливістю зберігати кастомну піцу;

– DeliveryData – зберігає контактні дані для доставки: адресу, ім’я та номер телефону користувача;

– OrderHistory – таблиця для фіксації історії замовлень. Містить статус, спосіб оплати, суму, список товарів, дату створення та зв’язок із користувачем.

Розробка бази даних забезпечила фундамент для коректного функціонування системи Telegram-бота. Завдяки використанню PyCharm та Django ORM процес створення структури БД був гнучким, прозорим і зручним

у налагодженні, що дозволило ефективно реалізувати логіку зберігання користувацьких дій, складу піци та історії замовлень.

3.4 Захист інформаційно-комп’ютерної системи

Важливою складовою розробки Telegram-бота для замовлення піци з інтеграцією платіжної системи та елементами штучного інтелекту є забезпечення безпека інформаційно-комп’ютерної системи. Метою є гарантування для користувачів конфіденційності та цілісності даних, а також запобігання несанкціонованому доступу, втраті або підробці інформації.

На рівні адміністративної панелі, реалізованого за допомогою фреймворку Django, захист забезпечується низкою вбудованих у фреймворк механізмів. Аутентифікація та авторизація користувачів забезпечують контроль доступу до функціональних можливостей системи та до даних користувачі. Шаблони Django автоматично обробляють HTML-контент, що дозволяє ефективно захищатися від XSS-атак. Захист від CSRF реалізовано через автоматичне додавання спеціальних токенів до форм, а використання ORM дає змогу уникнути SQL-ін’єкцій, оскільки всі запити до бази даних створюються без необхідності написання сирого SQL-коду.

База даних MySQL, яка використовується для зберігання інформації про користувачів та замовлення, захищена за допомогою системи доступу з розмежуванням прав, шифруванням з’єднання через SSL та регулярним резервним копіюванням. Також впроваджено логування активності, що дозволяє виявити та запобігти підозрілим діям у системі.

Telegram-бот, як інтерфейс взаємодії користувача з системою, також містить низку захисних заходів. Комунікація з Telegram API відбувається через зашифровані HTTPS-з’єднання.

Особливу увагу приділено інтеграції з платіжною системою. Усі транзакції здійснюються через офіційний API платіжного провайдера з використанням HTTPS. Для перевірки справжності запитів використовується

цифровий підпис, а чутливі платіжні дані не зберігаються на сервері, що відповідає вимогам стандарту PCI DSS. Кожна транзакція проходить додаткову перевірку на стороні сервера.

Серверне середовище, на якому розгорнуто систему, також захищене належним чином. Доступ до нього обмежений лише через протокол SSH з використанням ключів, паролі деактивовано. Система регулярно оновлюється для закриття вразливостей, а доступ до відкритих портів контролюється брандмауером. Використовуються додаткові інструменти для виявлення спроб несанкціонованого входу, наприклад, fail2ban.

Підбиваючи підсумок можна сказати, що реалізований комплекс заходів захисту дозволяє забезпечити надійне та безпечне функціонування інформаційно-комп'ютерної системи, що обробляє персональні дані користувачів та платіжні транзакції.

3.5 Розробка документації до програмного продукту

Адміністративна панель одна з найважливіших частин даного програмного продукту, оскільки вона дозволяє редагувати меню, контролювати замовлення, переглядати дії користувачів, а також виконувати базові адміністративні функції.

Для початку роботи з чат-ботом через адмінпанель необхідно зайти в вебзастосунок (рис. 3.15).

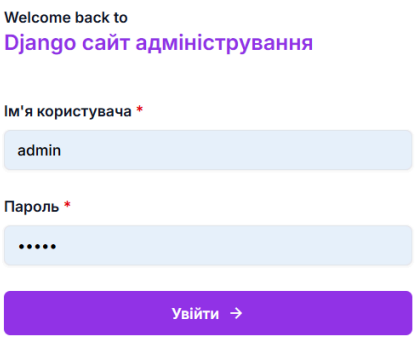

Рисунок 3.15 – Авторизація в адміністративну панель

Після успішної авторизації, відкриється головна сторінка адмін панелі (рис. 3.16).

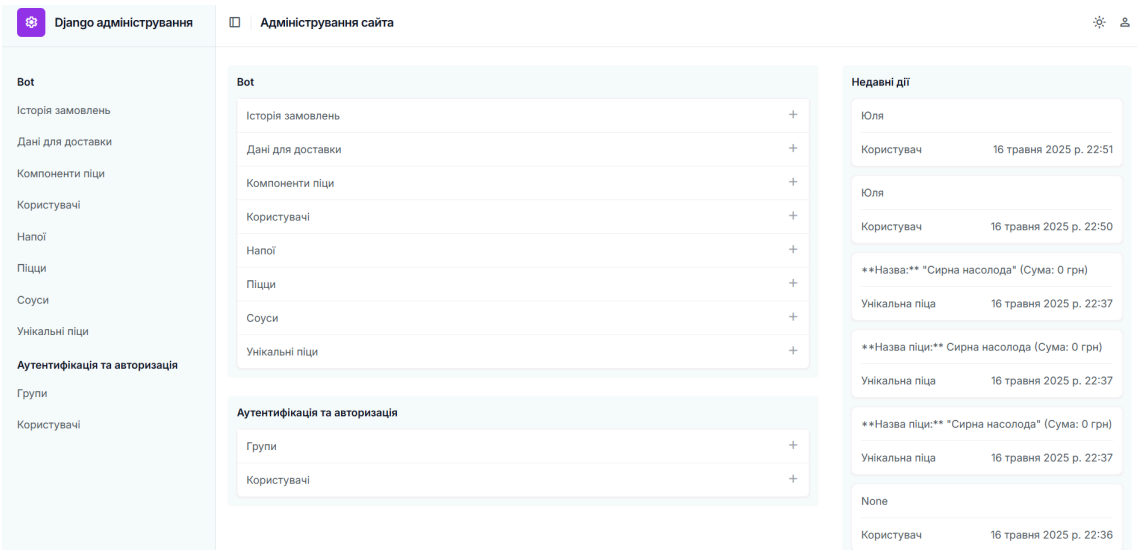

Рисунок 3.16 – Головна сторінка адмінпанелі

Зліва розташовано навігаційне меню з усіма зареєстрованими сутностями починаючи від замовлень і даних для доставки до користувачів, компонентів піци, напоїв, соусів та унікальних піц. Також розділ аутентифікації та авторизації з групами та користувачами. У центрі екрану відображаються картки з найважливішими моделями. Праворуч розміщено віджет «Недавні дії», що відображає ім’я адміністратора, виконану операцію та точний час зміни, забезпечуючи прозору історію всіх останніх змін у системі.

Для додавання нової позиції до меню, однієї з категорій необхідно натиснути плюсик, який знаходиться у правому верхньому кутку (рис. 3.17). Після цього відкриється сторінка «Додати Піцу» адміністративної панелі Django відображає просту форму для створення нового об’єкта PizzaList, де вгорі показано шлях навігації «Домівка / Bot / Піци / Додати Піцу», потім у єдиному блоці розміщені поля для введення назви, ціни та опису (за замовчуванням «Опис відсутній») з обов’язковими маркерами зірочкою, а також поле завантаження зображення.

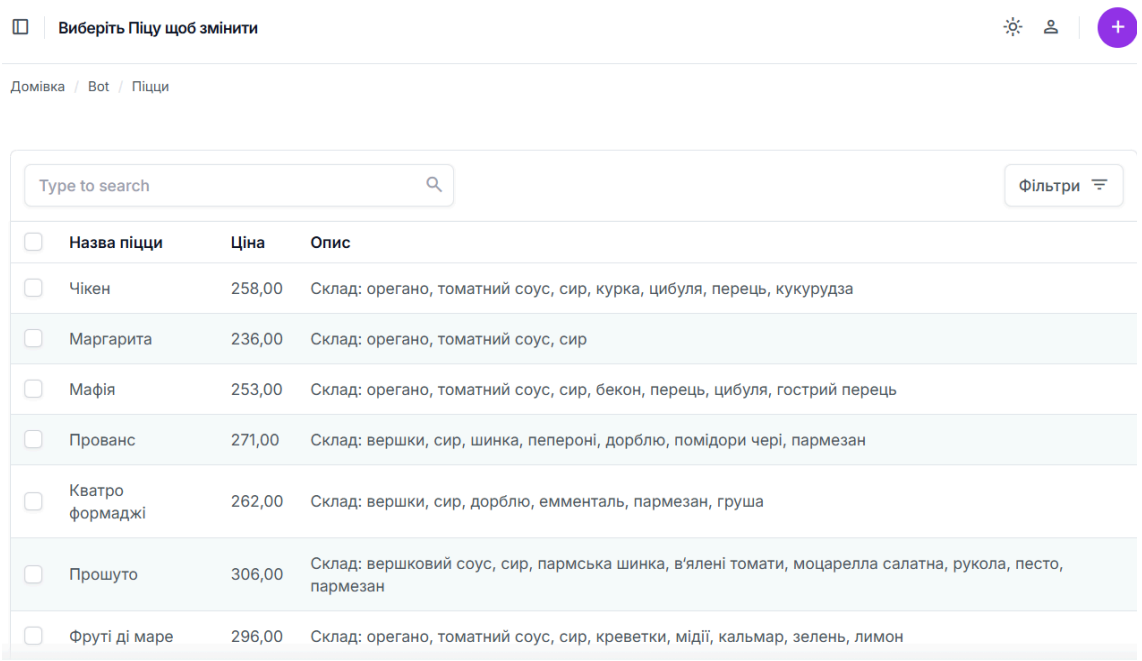

Рисунок 3.17 – Сторінка з піцами

У нижній частині сторінки доступні кнопки «Зберегти й додати інше», «Зберегти і продовжити редагування» та «Зберегти», що дозволяють зберегти запис і одразу перейти до створення нового, повернутися до редагування поточного або просто зберегти зміни (рис. 3.18).

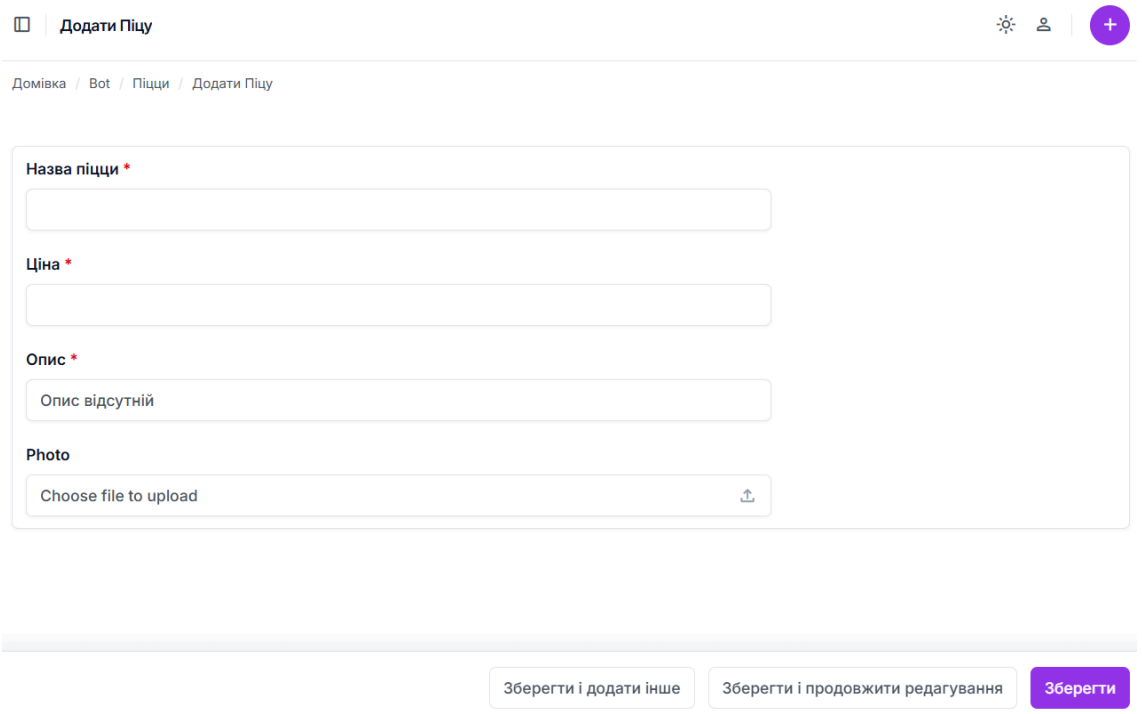

Рисунок 3.18 – Створення нової позиції меню

Для інших категорій меню додавання позицій є аналогічним. Якщо ж необхідно відредагувати товар, потрібно натиснути на нього в загальному списку, відкриється ідентична сторінка як і для створення, але вже з заповненими полями, які можна змінювати. Після усіх необхідних змін потрібно натиснути кнопку «Зберегти».

Сторінка редагування запису «Історія замовлень» відображає зверху навігаційний шлях, номер замовлення, користувача та суму. Під ним у блоці форми розміщено поля вибору користувача (з кнопками перегляду, редагування та додавання нового), загальної суми, статусу замовлення та способу оплати, а також дата створення відображається лише для читання. Нижче в окремому розділі «Товари замовлення» перелічено позиції (наприклад, «Піца – Прованс × 2»), а внизу сторінки розташовані кнопки «Видалити Історію замовлень» (червона), «Зберегти й додати інше», «Зберегти і продовжити редагування» та «Зберегти», що дозволяють видалити запис або зберегти зміни різними способами (рис. 3.19).

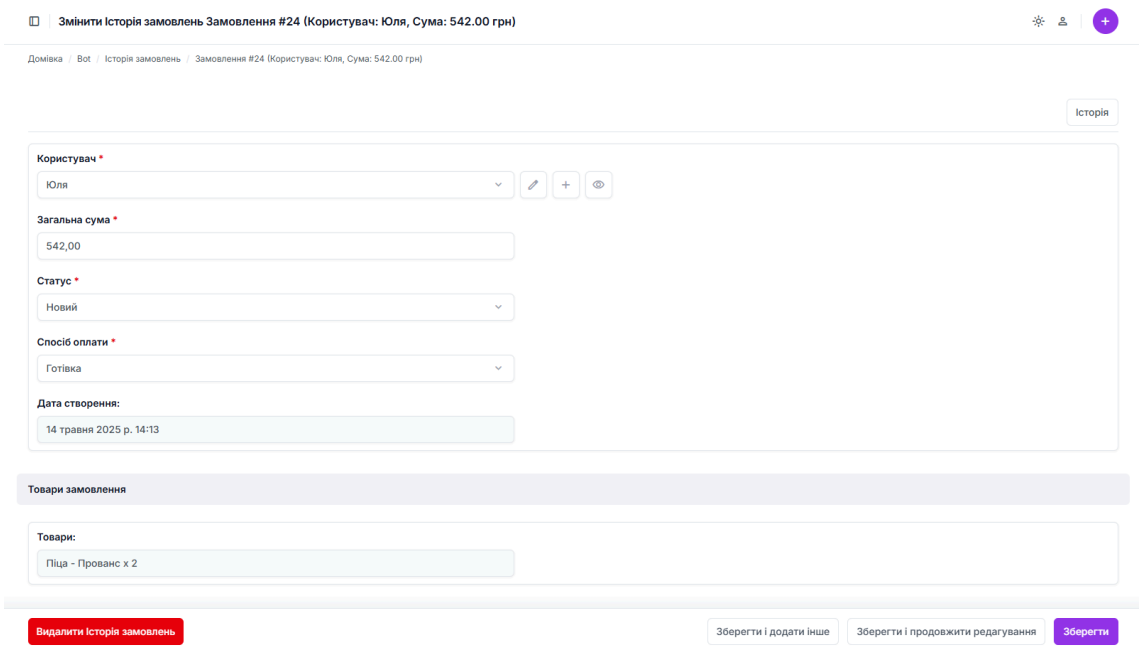

Рисунок 3.19 – Історія замовлень

Для зміни статусу замовлення необхідно натиснути на випадаючий список та обрати потрібний пункт (рис. 3.20). Після оновлення статусу користувачу прийде сповіщення в бот.

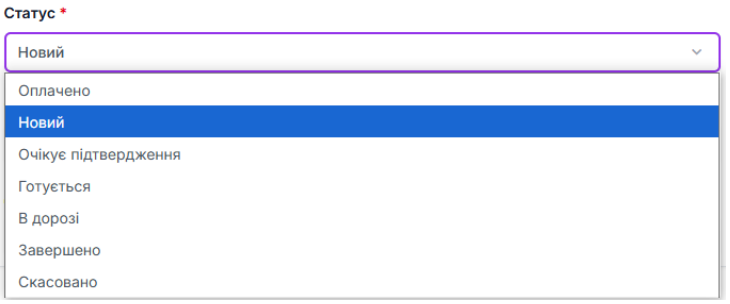

Рисунок 3.20 – Випадаючий список варіантів можливого статусу

Сторінка редагування користувача відображає навігаційний шлях, після чого в єдиному блоці розміщено поля для введення Telegram ID, ім’я, прізвище та юзернейм, а також перемикачі «Notified» і «Ban». Якщо активувати перемикач «Ban», на будь-яку дію користувача бот відправлятиме повідомлення: «Ваш профіль заблоковано!». Нижче за формою наведено вкладений розділ «Історія замовлень» із повним переліком замовлень, які коли-небудь робив користувач із можливістю швидкого редагування способу оплати та статусу або видалення запису. У нижній частині сторінки знаходяться кнопки «Видалити Користувач», «Зберегти й додати інше», «Зберегти і продовжити редагування», «Зберегти» для керування змінами.

При завершенні роботи потрібно натиснути на іконку, яка постійно відображається в правому верхньому кутку (рис. 3.21) та натиснути кнопку «Вийти».

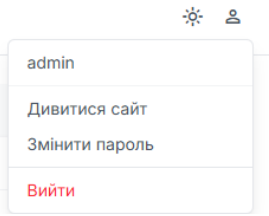

Рисунок 3.21 – Випадаюче меню користувача в адмін-панелі

Після успішного завершення роботи, на сторінці вебзастосунка буде відображатись відповідний текст (рис. 3.22).

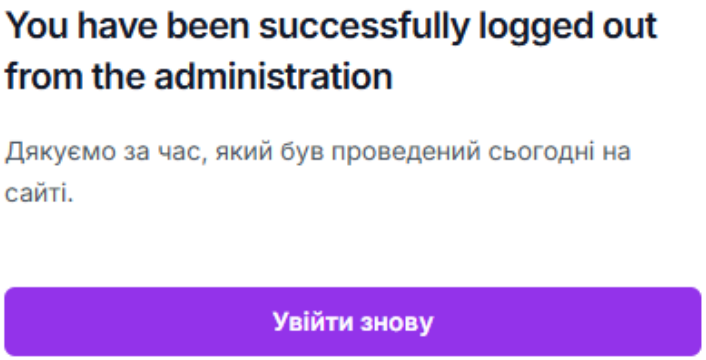

Рисунок 3.22 – Повідомлення про успішне завершення роботи

Висновки до розділу 3

Протягом роботи над кваліфікаційною робою бакалавра було успішно реалізовано та протестовано Telegram-бот для замовлення піци з інтеграцією штучного інтелекту та платіжною системою. Розроблено базу даних, що забезпечує надійне зберігання і швидкий доступ до інформації про замовлення, клієнтів і меню. Запроваджені механізми захисту інформаційної системи, зокрема автентифікація адміністратора, шифрування даних та регулярне резервне копіювання, гарантують цілісність і конфіденційність користувацьких даних. Розроблена документація для адміністративної панелі містить в собі детальні описи функціоналу та інструкції з використання.

ВИСНОВКИ

Телеграм-бот для замовлення піци з підтримкою штучного інтелекту та оплатою онлайн був успішно створений і повністю відповідає усім поставленим вимогам.

У результаті проведеного аналізу предметної області було виявлено, що наявні способи замовлення піци такі як мобільні додатки, вебсайти, соціальні мережі, телефонні дзвінки та спеціалізовані сервіси – забезпечують зручність і швидкість, але мають низку обмежень. Часто спостерігається слабка інтеграція з платіжними сервісами та недостатній рівень персоналізації. Для реалізації чат-бота було обрано месенджер Telegram, який забезпечує зручність інтеграції та широку аудиторію користувачів.

Сформульовано функціональні вимоги, які дозволили детально відтворити ключові сценарії, починаючи від створення замовлення до отримання рекомендацій від ШІ, перегляду статусу та історії попередніх покупок. Нефункціональні вимоги, що включають в себе час відгуку, доступність, пропускну здатність та показники безпеки – були визначені на рівні конкретних метрик. Розроблено UML-діаграми, а саме класів, використання та послідовності для демонстрації основних сценаріїв взаємодії користувача з чат-ботом.

Проведено аналіз сучасних технологій для розробки чат-боту. Обрані такі технології розробки: Python, Django, бібліотеку aiogram 3, Portmone, Open AI GPT, PyCharm, GitHub. Цей стек технологій гарантує високу продуктивність, масштабованість і простоту інтеграції з платіжними шлюзами та мовними моделями GPT.

Чат-бот з базовими діалоговими сценаріями та інтеграцією платіжного модуля успішно реалізовано. Було впроваджено штучний інтелект для генерації персоналізованих піц з доступних компонентів, створено адміністративну панель для моніторингу замовлень та управління меню.

Було здійснено оптимізацію коду за допомогою JetBrains AI Assistant. Проведено функціональне тестування з перевіркою коректності роботи кожного API-методу та бізнес-сценарію, а також інтеграційне тестування взаємодії між ботом, платіжною системою, базою даних і модулем штучного інтелекту. Усі виявлені дефекти успішно усунено.

Успішно спроєктовано базу даних з ключовими моделями. Процес проєктування включав побудову ER-моделі з визначенням сутностей та їх зв’язків. Було забезпечено регулярне виконання міграцій для безпечного управління змінами структури.

Було проведено аналіз ризиків (SQL-ін’єкції, XSS, CSRF, DDoS), а потім реалізовано захист через TLS-шифрування, багаторівневу авторизацію, ведення аудиту та оперативне реагування на інциденти для забезпечення цілісності та конфіденційності даних користувачів.

Завершальною стадією стала підготовка документації для адміністратора чат-бота доставки піци: у ній описано всі модулі інтерфейсу (керування користувачами, перегляд і обробка замовлень).

Реалізований Telegram-бот для замовлення піци з ШІ-рекомендаціями та платіжною системою повністю відповідає поставленим вимогам. Він забезпечує персоналізований підхід для користувача, безпечні платежі та відстеження статусу замовлення, працює стабільно й готовий до масштабування та подальшого розвитку.

В майбутньому хочу впровадити акційні пропозиції, якими можна буде керувати з адміністраторської панелі, накопичувальну бонусну систему, додати фільтрацію основного меню безпосередньо в боті та створити систему відгуків, для того, щоб користувачі могли залишати коментарі для покращення якості обслуговування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Попит на доставку їжі та напоїв в Україні. Kyivstar Business Hub – корпоративний блог для бізнесу. URL: https://hub.kyivstar.ua/articles/stan-rinku-fud-dostavki-v-ukrayini-yak-zrostaye-popit-na-dostavku-yizhi-ta-napoyiv (дата звернення: 06.02.2025).

2. Професії майбутнього: ТОП-100 в Україні та світі. URL: https://bizmag.com.ua/profesiyi-majbutnogo/ (дата звернення: 10.02.2025)

3. AI Growth Statistics: The Transformative Impact of Artificial Intelligence. MarketingProfs. URL: https://www.marketingprofs.com/articles/2024/51231/ai-growth-statistics-transformative-impact-artificial-intelligence (date of access: 15.02.2025).

4. How AI-Powered Delivery Management is Shaping the Future of Logistics. FarEye. URL: https://fareye.com/resources/blogs/ai-delivery-management (date of access: 23.02.2025).

5. Топ-10 популярних месенджерів світу та України у 2024 році. SiteCat IT News. URL: https://sitecat.net/review/top-10-popular-messengers/ (дата звернення: 28.02.2025).

6. Unified Modeling Language (UML) description. URL: https://www.uml-diagrams.org/ (date of access: 01.03.2025).

7. Python 3.13 documentation. URL: https://docs.python.org/3/ (date of access: 17.03.2025).

8. Getting started – PyCharm. URL: https://www.jetbrains.com/help/pycharm/getting-started.html (date of access: 19.03.2025).

9. Aiogram 3.20.0.post0 documentation. URL: https://docs.aiogram.dev/en/v3.20.0.post0/ (date of access: 21.03.2025).

10. Django documentation. URL: https://docs.django project.com/en/5.2/ (date of access: 23.03.2025).

11. MySQL Documentation. URL: https://dev.mysql.com/doc/ (date of access: 25.03.2025).

12. Платіжний шлюз Portmone.com. API для партнерів з сертифікатом PCI DSS – Документація. URL: https://docs.portmone.com.ua/docs/uk/PortmoneHostToHostUkr/ (дата звернення: 05.03.2025).

13. Getting Started. OpenAPI Documentation. URL: https://learn.openapis.org/ (date of access: 12.03.2025).

14. GitHub.com Help Documentation. GitHub Docs. URL: https://docs.github.com/en (date of access: 17.04.2025).

15. What is CI/CD?. GitHub. URL: https://github.com/resources/articles/devops/ci-cd (date of access: 18.04.2025).

ДОДАТКИ

Додаток А

Підтвердження апробації результатів кваліфікаційної роботи

Міністерство освіти і науки України
Волинська обласна рада
Луцька міська рада
Луцький національний технічний університет
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Національний університет «Львівська політехніка»
Військовий інститут телекомунікацій та інформатизації імені Героїв Крут (м. Київ)
Тернопільський національний педагогічний університет імені В. Гнатюка
Рівненський державний гуманітарний університет
Навчально-науковий інститут «Українська інженерно-педагогічна академія» Харківського національного університету ім. В.Н. Каразіна
Люблінська політехніка (Польща)
Фрайберзька гірнича академія (Німеччина)
Гронінгенський університет (Нідерланди)
Кавказький університет (Грузія)
Політехнічний університет Браганси (Португалія)

ТЕЗИ ДОПОВІДЕЙ
X Міжнародної науково-практичної конференції з проблем вищої освіти і науки «Інформаційні технології в освіті, науці і виробництві (ІТОНВ-2025)

23-24 травня 2025 р.

Луцьк 2025

Shumyliak Liliia **Cibák Luboš** **Tarnovetska Olha**	Implementation of web technologies in the management of medical department workflows	271

СЕКЦІЯ 5. ІНФОРМАЦІЙНІ ТА ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ І ТЕХНОЛОГІЇ

Антонюк А.О. **Повстяна Ю.С.**	Голосовий помічник на основі штучного інтелекту	275
Вовк П.Б.	Підготовка середовища та налаштування локального кластера Kubernetes: від теорії до практики	282
Вознюк А.В. **Данилко К.С.**	Розробка 2D ігор на unity з генерацією світу	286
Вознюк М.В. **Неділько О.В.**	Аналіз впливу темної теми на зручність користування веб-додатками	289
Вознюк А.В. **Хітько Ю.В.**	Автоматизація сервісів доставки їжі за допомогою Telegram-бота з AI та платіжним модулем	293
Вознюк А.В. **Чоботар Ю.С.** **Примич М.О.**	Технічні рішення та освітні переваги при створенні і впровадженні вебплатформи для онлайн-навчання	298
Газдюк К.П. **Дмитращук К.М.** **Кривинчук В.О.**	Адаптивне планування логістичних маршрутів в умовах воєнного стану	304
Головня С.А.	Застосування сучасних бібліотек Python для задач з аналізу та прогнозування	308
Демідов П.Г. **Андрієнко В.А.** **Маргаза Д.Ю.** **Муляр В.П.**	Порівняльна характеристика нейро-нечітких та нейронних технологій на прикладі розв'язання задачі прогнозування вартості валюти на фінансовому ринку	312

УДК 004.85

АВТОМАТИЗАЦІЯ СЕРВІСІВ ДОСТАВКИ ЇЖІ ЗА ДОПОМОГОЮ TELEGRAM-БОТА З AI ТА ПЛАТІЖНИМ МОДУЛЕМ

Вознюк Анастасія Вадимівна
Луцький національний технічний університет, асистент кафедри інженерія програмного забезпечення, a.vozniuk@lutsk-ntu.com.ua
Хітько Юлія Володимирівна
Луцький національний технічний університет, студентка 4 курсу кафедри інженерія програмного забезпечення, khitko.y1012@lntu.edu.ua

AUTOMATION OF FOOD DELIVERY SERVICES USING A TELEGRAM BOT WITH AI AND A PAYMENT MODULE

Vozniuk Anastasiia Vadymivna
Lutsk National Technical University, Assistant at the Department of Software Engineering, a.vozniuk@lutsk-ntu.com.ua
Khitko Yulia Volodymyrivna
Lutsk National Technical University, Student at the Department of Software Engineering, khitko.y1012@lntu.edu.ua

The development of a chatbot for food ordering in Telegram is presented. It has been established that chatbots can serve as convenient and cost-effective tools for automating commercial services. By integrating artificial intelligence and online payment systems, such bots significantly simplify the ordering process and enhance user experience. This technology has strong potential in small and medium-sized businesses, particularly in the food industry, enabling fast deployment of digital sales channels.

***Keywords**: chatbot, food ordering, artificial intelligence, automation, online payment.*

Стрімкий розвиток цифрових технологій зумовлює активне впровадження автоматизованих рішень у сферу сервісного обслуговування. Під час карантинних обмежень пов'язаних з Covid-19 великою популярністю почали користуватись послуги доставок. Ця тенденція збереглась і на теперішній час. Це, у свою чергу, викликало потребу в автоматизації процесівю для зниження затратності ресурсів та підвищення сервісу для користувачів.

Зростання попиту на онлайн-сервіси обумовлює потребу у швидкому та зручному способі оформлення замовлень, що робить автоматизацію доставки їжі надзвичайно актуальною. Автоматизовані рішення знижують навантаження на персонал, мінімізують ризики людського фактора та скорочують час обробки замовлення.

Автоматизація дозволяє малим бізнесам конкурувати з великими мережами, надаючи сучасні цифрові інструменти без значних витрат. Вона також дозволяє вести точний облік замовлень, зменшувати кількість помилок при обробці та формувати детальну статистику для прийняття управлінських рішень.

Впровадження автоматизації сприяє підвищенню рівня довіри до сервісу, оскільки користувачі отримують прозору інформацію про замовлення, час доставки та підтвердження оплати, також упровадження автоматизації відповідає сучасним трендам цифрової трансформації бізнесу та є важливою конкурентною перевагою на ринку послуг.

Системи онлайн-доставки з автоматизованим управлінням замовленнями можуть масштабуватися та адаптуватися до різних моделей бізнесу — від локальних кафе до мережевих служб доставки.

Чат-боти є сучасною формою цифрової взаємодії, що поєднує зручність звичного спілкування з можливостями автоматизації. Вони забезпечують інтуїтивний доступ до сервісів, дозволяючи користувачам у діалоговому форматі швидко здійснювати дії — від запитів до оформлення замовлень — без потреби встановлення додаткових застосунків, при цьому гарантуючи персоналізований підхід та цілодобову доступність.

Компанія Klarna, запустила чатбот, який обробляє клієнтські звернення повністю автоматизовано.За оцінками компанії, цей бот виконує завдання, що раніше потребували участі близько 700 співробітників служби підтримки. Це свідчить про суттєвий вплив ШІ на структуру зайнятості в сервісних компаніях [3]. Замість очікування відповіді від

оператора, клієнти отримують миттєві відповіді завдяки ШІ, що сприяє вищому рівню задоволеності та підвищенню лояльності до бренду.

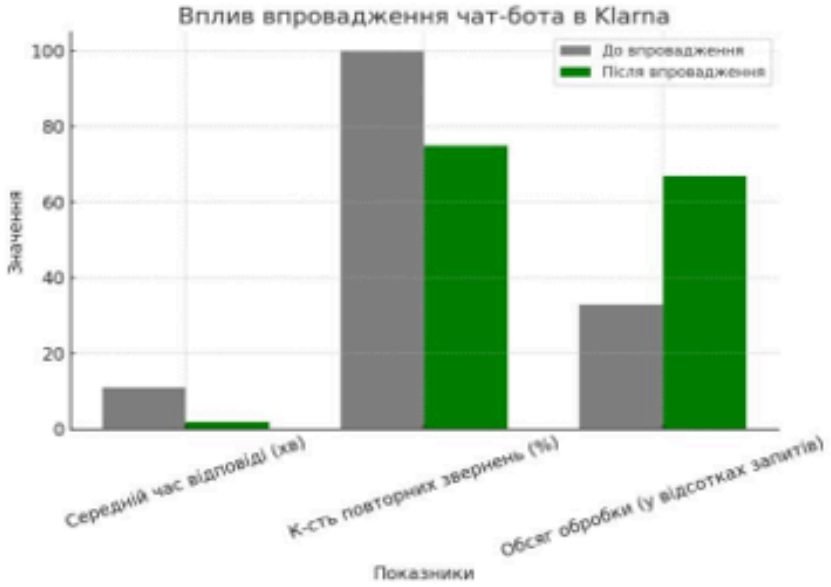

Рисунок 1 – Графік впливу чат-боту підтримки[3]

Telegram, як один із найпопулярніших месенджерів серед молоді, є ідеальною платформою для створення чат-ботів для автоматизації завдяки відкритому API, зручному інтерфейсу, кросплатформенності та широким можливостям інтеграції з сервісами автоматизації.

У порівнянні з класичними вебсайтами та застосунками, Telegram-боти мають нижчий поріг входу, не потребують встановлення, доступні через мобільні пристрої і підтримують інтерактивну взаємодію [4].

Telegram-бот підтримує мультимовний інтерфейс, що дозволяє адаптувати сервіс під різні мовні аудиторії, зокрема українську, англійську та польську мови, що є важливим у контексті інтернаціоналізації продукту.

Бот підтримує контекстну взаємодію з користувачем, зберігаючи стан розмови, що забезпечує зручну багатокрокову

логіку оформлення замовлення (вибір категорії, позиції, кількості, способу оплати тощо).

Telegram-бот легко масштабується, має спрощене тестування компонентів і можливість інтеграції з додатковими сервісами, такими як CRM чи сервіси аналітики.

Telegram-бот взаємодіє з зовнішніми API, що дозволяє у майбутньому забезпечити повний цикл обробки замовлення – від прийому до доставки клієнту з відстеженням у реальному часі.

Підтримка функціональність push-сповіщень — бот інформує користувача про стан замовлення чи зміну статусу доставки, що сприяє підвищенню залученості.

Інноваційність полягає у поєднанні чат-бота, рекомендаційного механізму на основі машинного навчання та інтеграції з платіжною платформою Portmone. Завдяки цьому користувач отримує пропозиції страв на основі вподобань, що підвищує лояльність і пришвидшує процес замовлення[1]. Інтеграція з платіжними системами забезпечує безпечну та оперативну оплату, що підвищує зручність для користувача та прискорює завершення транзакції[2].

Поєднання ШІ та Telegram-бота дозволяє створити інтелектуальну систему обслуговування, яка не просто приймає замовлення, а й формує персональні пропозиції[5].

Використання Django в Telegram-боті забезпечує зручне адміністрування завдяки вбудованій адміністративній панелі, яка дозволяє ефективно керувати меню, замовленнями, аналітикою продажів та користувачами, а кастомізовані інтерфейси додатково спрощують роботу працівників і управління даними.Обробка помилок і винятків з детальним логуванням, що дозволяє швидко діагностувати і виправляти технічні збої у роботі бота.

Отже, автоматизація сервісів доставки їжі за допомогою Telegram-бота із вбудованим штучним інтелектом та платіжним модулем є доцільною та перспективною. Такий підхід дозволяє скоротити витрати на персонал, прискорити обробку замовлень,

забезпечити цілодобове обслуговування та підвищити зручність для користувачів. Поєднання ШІ для обробки запитів і модулів оплати створює ефективний та масштабований інструмент сучасної взаємодії з клієнтами.

Список використаних джерел

1. Оптимізація процесу оформлення замовлення: як збільшити конверсію - Фулфілмент для інтернет-магазинів LP-Sklad. Фулфілмент для інтернет-магазинів LP-Sklad. URL: https://lp-sklad.biz/blog/optymizacziya-proczesu-oformlennya-zamovlennya-yak-zbilshyty-konversiyu/ (дата звернення: 29.04.2025).
2. Як штучний інтелект може покращити роботу з підтримки клієнтів та підвищить якість сервісу? - Блог RX-NAME UA. *Блог RX-NAME UA*. URL: https://rx-name.ua/blog/yak-shtuchnyj-intelekt-mozhe-pokrashhyty-robotu-z-pidtrymky-kliyentiv-ta-pidvyshhyt-yakist-servisu (дата звернення: 03.05.2025).
3. Alt C. Klarna's AI chatbot does the work of 700 full-time staff. *The Times & The Sunday Times*. URL: https://www.thetimes.com/business-money/technology/article/klarnas-ai-chatbot-does-the-work-of-700-full-time-staff-nbmd2qwnp?utm_source=chatgpt.com&region=global (date of access: 01.05.2025).
4. Bots: An introduction for developers. *Telegram APIs*. URL: https://core.telegram.org/bots (date of access: 28.04.2025).
5. s.yelbaiev. Чат-боти на базі AI: розбираємо переваги для бізнесу. *Custom Web & Mobile Development Company - New Line Technologies*. URL: https://newline.tech/ai-chatbots-exploring-the-advantages-for-businesses-ua/ (дата звернення: 03.05.2025).