Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп`ютпрних наук_____
(повна назва)

Кафедра _____Системотехніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

Розробка веб-системи управління замовленнями в ресторані з персоналізованими рекомендаціями з харчування
(тема)

Виконав:
здобувач ____2___ року навчання,
групи ____ІТПм-24-1 _____
_____Олексій Ахтирський_____
(власне ім'я, прізвище)

Спеціальність __122 Комп`ятерні науки_____
(код і повна назва спеціальності)
Тип програми ___освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _Інформаційні технології проєктування_____
(повна назва освітньої програми)

Керівник __проф. каф. СТ Вадим Саваневич_
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри _____ _____ ___Ігор Гребеннік____
(підпис) (власне ім'я, прізвище)

2025 р.

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

15.12.2025 *Ахтирський О.Ю.*

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведено 15 грудня 2025 р.

Керівник кваліфікаційної роботи  *Саваневич В.Є.*

Харківський національний університет радіоелектроніки

Факультет_____Комп`ютерних наук_____

Кафедра_____системотехніки_____

Рівень вищої освіти_____другий (магістерський)_____

Спеціальність_____122 Комп`ютерні науки_____
(код і повна назва)

Тип програми _____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)

Освітня програма_____Інформаційні технології проєктування_____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____ 20 ___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві_____Ахтирському Олексію Юрійовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Розробка веб-системи управління замовленнями в ресторані з персоналізованими рекомендаціями з харчування_____
затверджена наказом університету від _24_ ___листопада_____2025 р. № _1058Ст__
2. Термін подання здобувачем роботи до екзаменаційної комісії _15_ _грудня__ 2025 р.
3. Вихідні дані до роботи __*Тема дослідження: аналіз веб-технологій та архітектурних патернів, що застосовуються для автоматизації обслуговування у закладах харчування; методи побудови інтелектуальних систем персоналізації дієтичного раціону. Дані щодо структури ресторанного меню, параметрів розрахунку фізіологічних потреб користувача (BMR), алгоритмів фільтрації алергенів, інтеграції з генеративними моделями штучного інтелекту для аналізу складу страв та відображення рекомендацій у веб-інтерфейсі. Перелік використаних програмних засобів: Node.js, Express.js, React, PostgreSQL, Google Gemini API, Tailwind CSS, Vercel,Render,Neon*_____
4. Перелік питань, що потрібно опрацювати в роботі _*Вступ, обґрунтування актуальності та мети дослідження, аналіз предметної області ресторанного бізнесу (HoReCa) та проблем персоналізації харчування, класифікація сучасних систем автоматизації закладів харчування (POS, агрегатори), характеристика методів побудови рекомендаційних систем (Content-Based, Knowledge-Based), огляд підходів до інтеграції штучного інтелекту в веб-сервіси, характеристика генеративної моделі Google Gemini, архітектура програмного забезпечення (мікросервісна, клієнт-серверна), розробка серверної частини (API, контролери, моделі даних), розробка клієнтського інтерфейсу (UI/UX, компоненти React), формування нутриціологічних рекомендацій на основі даних профілю користувача, дослідження ефективності роботи гібридного алгоритму рекомендацій, аналіз впливу системи на швидкість обслуговування та якість вибору страв, висновки, перелік джерел посилання, додатки з графічним матеріалом (скріншоти інтерфейсу, діаграми)*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *1. Тема кваліфікаційної роботи. 2. Об'єкт, предмет та мета дослідження. 3. Актуальність теми та проблематика. 4. Аналіз існуючих рішень. POS-системи. 5. Аналіз існуючих рішень. Агрегатори доставки. 6. Аналіз спеціалізованих додатків. 7. Концепція гібридної рекомендаційної системи. 8. Використання великих мовних моделей (LLM). 9. Архітектурний патерн RAG (Retrieval-augmented generation). 10. Структура системного промпту та алгоритм валідації відповідей. 11. Діаграма розгортання системи (Deployment Diagram). 12. Діаграма варіантів використання (Use Case Diagram). 13. Діаграма послідовності (Sequence Diagram). 14. ER-діаграма бази даних. 15. Інтерфейси каталогу страв та детальна картка. 16. Інтерфейси налаштування профілю здоров'я та алергенів. 17. Інтерфейс діалогу з AI-асистентом та картка рекомендації. 18. Інтерфейси панелі адміністратора. 19. Графік залежності часу генерації відповіді від розміру контексту меню. 20. Висновки.*

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	*Отримання завдання на виконання кваліфікаційної роботи.*	*24.11.25*	*Виконано*
2.	*Аналіз завдання та предметної області*	*24.11 – 27.11.25*	*Виконано*
3.	*Опрацювання літератури та аналіз об'єкту дослідження*	*27.11 – 29.11.25*	*Виконано*
4.	*Формування постановки задачі*	*29.11 – 01.12.25*	*Виконано*
5.	*Проектування апаратного та програмного засобу*	*01.12 – 03.12.25*	*Виконано*
6.	*Розробка та тестування апаратного та програмного засобу*	*03.12 – 05.12.25*	*Виконано*
7.	*Аналіз результатів дослідження, отриманих за допомогою апаратного та програмного засобу*	*05.12 – 06.12.25*	*Виконано*
8.	*Оформлення пояснювальної записки та презентаційних матеріалів комп'ютерного захисту*	*06.12 – 14.12.25*	*Виконано*
9.	*Представлення роботи на рецензування*	*15.12.25*	*Виконано*

Дата видачі завдання 24 листопада 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. каф. СТ Вадим САВАНЕВИЧ
(підпис) (посада, власне ім`я, прізвище)

РЕФЕРАТ

Кваліфікаційна робота: 77 стор., 4 табл., 22 рис., 3 додатки, 29 джерел.

ВЕБ-СИСТЕМА, УПРАВЛІННЯ ЗАМОВЛЕННЯМИ, ПЕРСОНАЛІЗАЦІЯ, РЕКОМЕНДАЦІЙНІ АЛГОРИТМИ, ШТУЧНИЙ ІНТЕЛЕКТ, LLM, REST API, ЗДОРОВЕ ХАРЧУВАННЯ, PROMPT ENGINEERING.

Об’єктом дослідження є процес інформаційного супроводу та обслуговування клієнтів у закладах ресторанного господарства з урахуванням сучасних вимог до індивідуалізації сервісу.

Предметом дослідження є методи та засоби розробки веб-орієнтованої інформаційної системи, що використовує можливості генеративного штучного інтелекту для аналізу нутриціологічних даних та формування персоналізованих пропозицій.

Метою дослідження є підвищення якості обслуговування відвідувачів та автоматизація процесу вибору страв шляхом створення програмного комплексу, здатного адаптувати меню під індивідуальні фізіологічні потреби користувача в режимі реального часу.

Методи дослідження включають системний аналіз предметної області HoReCa, об’єктно-орієнтоване проєктування (ООР), моделювання реляційних баз даних, розробку клієнт-серверної архітектури (RESTful API), а також методи інженерії запитів (prompt engineering) для взаємодії з великими мовними моделями. Особливу увагу в роботі приділено проєктуванню інтелектуального модуля.

Галузь застосування результатів охоплює підприємства ресторанного бізнесу, сервіси доставки їжі, платформи для планування дієтичного харчування та стартапи у сфері FoodTech, спрямовані на персоналізацію користувацького досвіду.

ABSTRACT

Qualification work: 77 pages, 4 tables, 22 figures, 3 appendices, 29 references.

WEB SYSTEM, ORDER MANAGEMENT, PERSONALIZATION, RECOMMENDATION ALGORITHMS, ARTIFICIAL INTELLIGENCE, LLM, REST API, HEALTHY EATING, PROMPT ENGINEERING.

The object of research is the process of informational support and customer service in the restaurant industry, taking into account modern requirements for service individualization.

The subject of research is the methods and tools for developing a web-based information system that utilizes generative artificial intelligence capabilities to analyze nutritional data and generate personalized recommendations.

The aim of the research is to improve customer service quality and automate the dish selection process by creating a software complex capable of adapting the menu to the user's individual physiological needs in real time.

The research methods include system analysis of the HoReCa domain, object-oriented design (OOD), relational database modeling, client-server architecture development (RESTful API), and prompt engineering methods for interacting with large language models. Particular attention is paid to the design of the intelligent module.

The scope of application includes restaurant businesses, food delivery services, dietary planning platforms, and FoodTech startups focused on personalizing the user experience.

ЗМІСТ

ВСТУП

Актуальність теми зумовлена тим, що сучасна індустрія гостинності (HoReCa) перебуває на етапі фундаментальної трансформації, де ключовим драйвером змін стає не лише цифровізація бізнес-процесів, а й глобальний тренд на свідоме ставлення до здоров'я. Споживачі дедалі частіше висувають вимоги не лише до органолептичних якостей страв, а й до їхньої нутриціологічної цінності, відповідності індивідуальним дієтам та безпеці складу. В умовах зростання кількості людей з харчовими алергіями, непереносимістю певних продуктів та специфічними метаболічними потребами, традиційний підхід до представлення меню у вигляді статичного переліку назв і цін втрачає свою ефективність. Існуючі на ринку інформаційні системи, такі як класичні POS-термінали або агрегатори доставки, успішно вирішують задачі логістики та фінансового обліку, проте виявляються функціонально обмеженими у забезпеченні персоналізованого підходу до клієнта. Більшість з них не надає інструментів для автоматичного глибокого аналізу складу страв на сумісність з профілем споживача, що перекладає відповідальність за вибір безпечної їжі повністю на клієнта, підвищуючи ризик помилки та знижуючи лояльність до закладу.

У цьому контексті виникає гостра необхідність у розробці інтелектуальних веб-систем нового покоління, які повинні інтегрувати операційні процеси замовлення з експертними модулями, здатними аналізувати нутрієнти в реальному часі. Застосування технологій штучного інтелекту, зокрема великих мовних моделей (LLM), та алгоритмів рекомендацій дозволяє автоматизувати процес вибору, гарантуючи, що запропоновані страви відповідають як смаковим уподобанням, так і фізіологічним обмеженням користувача.

Метою дослідження є підвищення якості та безпеки обслуговування клієнтів закладів харчування шляхом розробки веб-системи управління замовленнями з інтегрованим модулем персоналізованих рекомендацій, що

базується на інтелектуальному аналізі нутриціологічних даних та профілю користувача.

Для досягнення мети вирішено комплекс завдань, що включає аналіз недоліків існуючих систем, обґрунтування архітектурного підходу та технологічного стеку, а також математичне моделювання рекомендаційного модуля з урахуванням алергенів і калорійності. Виконано проєктування та програмну реалізацію веб-застосунку з інтеграцією Google Gemini API для генерації пояснень, проведено тестування ефективності алгоритмів.

Наукова новизна отриманих результатів полягає в удосконаленні методу формування рекомендацій страв шляхом застосування гібридного підходу, що поєднує фільтрацію на основі обмежень для забезпечення безпеки споживання та контекстну фільтрацію для оптимізації нутриціологічної цінності. Також набув подальшого розвитку метод взаємодії користувача з меню за рахунок інтеграції генеративних мовних моделей, що дозволяє системі не лише ранжувати страви, а й надавати аргументовані пояснення щодо користі обраної позиції, реалізуючи концепцію пояснювального штучного інтелекту (Explainable AI).

Практичне значення роботи полягає у створенні повнофункціонального програмного продукту, який дозволяє закладам харчування автоматизувати розрахунок калорійності замовлень, мінімізувати ризики алергічних реакцій у клієнтів та надати персоналізований сервіс високого рівня без необхідності залучення додаткового кваліфікованого персоналу.

Апробація результатів дисертації. Основні положення та результати дослідження доповідались та обговорювались на Всеукраїнській науково-практичній конференції здобувачів вищої освіти і молодих учених «Комп'ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві», що відбулася 25 листопада 2025 року в Харківському національному автомобільно-дорожньому університеті [29].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ТЕХНОЛОГІЙ АВТОМАТИЗАЦІЇ РЕСТОРАННОГО БІЗНЕСУ

1.1 Сучасний стан та тенденції цифровізації сфери HoReCa

Сучасна індустрія гостинності (HoReCa - Hotel, Restaurant, Cafe/Catering) є однією з найбільш динамічних галузей світової економіки, яка в останні роки зазнала кардинальних змін під впливом глобальної цифровізації. Якщо на початку XXI століття автоматизація розглядалася переважно як інструмент фінансового контролю та складського обліку, то сьогодні інформаційні технології стають фундаментом для формування клієнтського досвіду (Customer Experience) та забезпечення конкурентоспроможності закладу.

Ключовим каталізатором цифрової трансформації стала пандемія COVID-19, яка виявила критичну вразливість традиційних бізнес-моделей, що спиралися виключно на офлайн-взаємодію. Вона сформувала стійкий попит на безконтактні технології та дистанційне замовлення. За даними аналітичних звітів, частка онлайн-замовлень у структурі доходів ресторанів продовжує демонструвати щорічне зростання на рівні 10–15% [2, 23]. Це вимагає від закладів інтеграції в єдину цифрову екосистему, яка об'єднує кухню, зал та клієнта.

Аналіз предметної області дозволяє виділити декілька домінуючих тенденцій, які визначають вектор розвитку IT-рішень у сфері HoReCa [5, 22]:

1. Омніканальність продажів. Сучасний ресторан функціонує як розподілена система. Замовлення надходять через веб-сайти, мобільні додатки, агрегатори доставки, соціальні мережі та QR-меню за столиком. Це створює потребу в системах, здатних централізувати гетерогенні потоки даних в єдиному інтерфейсі для уникнення помилок та затримок.

2. Гіперперсоналізація сервісу. Споживачі очікують, що сервіс буде враховувати їхні індивідуальні вподобання. Проте, на відміну від e-commerce, де рекомендації базуються на історії покупок ("схожі товари"), у сфері харчування персоналізація тісно пов'язана з фізіологією. Зростає популярність концепції

«Food as Medicine» (їжа як ліки), що вимагає від інформаційних систем здатності обробляти складні нутриціологічні дані.

3. Data-Driven Management. Накопичення великих обсягів даних (Big Data) про поведінку клієнтів дозволяє використовувати алгоритми машинного навчання для прогнозування попиту та оптимізації закупівель.Окрему увагу слід приділити зміні портрета споживача. Сучасний клієнт стає більш вимогливим до прозорості складу страв. Згідно з дослідженнями ринку FoodTech, понад 60% відвідувачів хочуть бачити повну інформацію про калорійність [9], алергени та походження інгредієнтів ще на етапі вибору страви [1]. Це зумовлює перехід від статичних PDF-меню до інтерактивних веб-систем, які дозволяють фільтрувати позиції за заданими критеріями.

Незважаючи на активну цифровізацію, існує суттєвий розрив між функціоналом існуючих систем та потребами сучасного споживача. Згідно з дослідженнями, понад 60% відвідувачів хочуть бачити повну інформацію про калорійність, алергени та походження інгредієнтів ще на етапі вибору страви [2]. Однак більшість наявних на ринку рішень (POS-термінали, агрегатори доставки) зосереджені на операційній ефективності - швидкості передачі замовлення на кухню та фіскалізації чека.

Можна виділити основні проблеми автоматизації, що залишаються невирішеними в контексті персоналізованого харчування:

— проблема неструктурованих даних: інформація про склад страв у базах даних ресторанів часто зберігається у вигляді простого тексту або технологічних карт, не пристосованих для автоматичного пошуку алергенів;

— відсутність семантичного зв'язку: традиційні системи не "розуміють", що "пармезан" - це молочний продукт, а отже, він небезпечний для людей з непереносимістю лактози. Це перекладає відповідальність за аналіз безпеки страви на клієнта або офіціанта, збільшуючи вплив людського фактора;

— інформаційне перевантаження: великі меню ускладнюють вибір, а статичні фільтри (наприклад, лише за ціною або категорією) не дозволяють сформувати раціон під специфічні дієтичні цілі (КБЖВ).

Таблиця 1.1 — Порівняльна характеристика підходів до автоматизації замовлень

Критерій порівняння	Традиційні системи (POS, Aggregators)	Інтелектуальні системи (Target Systems)
Основна мета	Облік транзакцій та логістика	Персоналізація та підтримка здоров'я
Робота з даними меню	Статичний текст, ручне введення	Семантичний аналіз складу, тегування
Фільтрація страв	Базова (категорія, ціна)	Глибока (алергени, макронутрієнти, цілі)
Роль користувача	Самостійний аналіз складу	Отримання готових рекомендацій
Використання AI	Відсутнє або мінімальне	Генерація пояснень та ранжування

Таким чином, актуальним науково-практичним завданням є розробка веб-орієнтованих систем нового покоління, які поєднують у собі функціонал класичних систем управління замовленнями (OMS) з можливостями штучного інтелекту. Це дозволить вирішити проблему інформаційної асиметрії між закладом та клієнтом, забезпечивши автоматизований контроль нутриціологічної безпеки. Зокрема, впровадження інтелектуальних агентів надасть можливість здійснювати динамічну адаптацію контенту меню в режимі реального часу, враховуючи приховані зв'язки між інгредієнтами та медичними протипоказаннями конкретного споживача.

1.2 Класифікація та архітектурні особливості сучасних систем управління замовленнями

Інформаційний ландшафт сучасного ресторанного бізнесу характеризується високим рівнем фрагментації та використанням гетерогенних програмних рішень. Системи управління замовленнями (Order Management Systems) є центральною ланкою цієї інфраструктури, забезпечуючи обмін даними між клієнтом, кухнею та адміністративним персоналом. З точки зору архітектурної побудови та функціонального призначення, існуючі рішення можна класифікувати за трьома основними категоріями: стаціонарні та хмарні POS-системи, платформи-агрегатори доставки та веб-сервіси безконтактного замовлення.

Історично першим та найбільш поширеним класом систем є POS-термінали (Point of Sale). Архітектурно вони еволюціонували від монолітних настільних застосунків (On-Premise), що вимагали встановлення локального сервера безпосередньо у закладі, до гнучких хмарних рішень (SaaS).

Сучасні хмарні POS-системи (наприклад, Poster, SkyService, iiko) використовують клієнт-серверну архітектуру, де «тонкий клієнт» (планшет або веб-браузер) взаємодіє з віддаленим сервером через REST API або GraphQL [28]. Хоча такі системи ефективно вирішують задачі фіскалізації, складського обліку та управління столами, вони залишаються транзакційно-орієнтованими. Їхня структура даних (схема БД) спроєктована для обліку руху товарних залишків, а не для аналізу складних семантичних зв'язків між інгредієнтами та здоров'ям клієнта. Інформація про страву в базі даних POS зазвичай обмежується назвою, ціною, категорією та технологічною картою (собівартістю), що унеможливлює глибоку персоналізацію меню.

Типовим прикладом такого інтерфейсу є система Poster POS. Як видно з рисунка 1.2, робоче місце оператора оптимізовано для швидкості: великі кнопки, чітка категоризація та функції управління чеком. Переваги: висока швидкість

обробки замовлень, інтуїтивний інтерфейс для персоналу, стабільна робота з фіскальними реєстраторами. Недоліки: відсутність детальної інформації про склад страв для кінцевого клієнта (видно лише назву), неможливість фільтрації за алергенами в режимі швидкого замовлення та відсутність персоналізованих рекомендацій на основі профілю здоров'я гостя.

На рисунку 1.1 зображено інтерфейс POS-системи Poster.

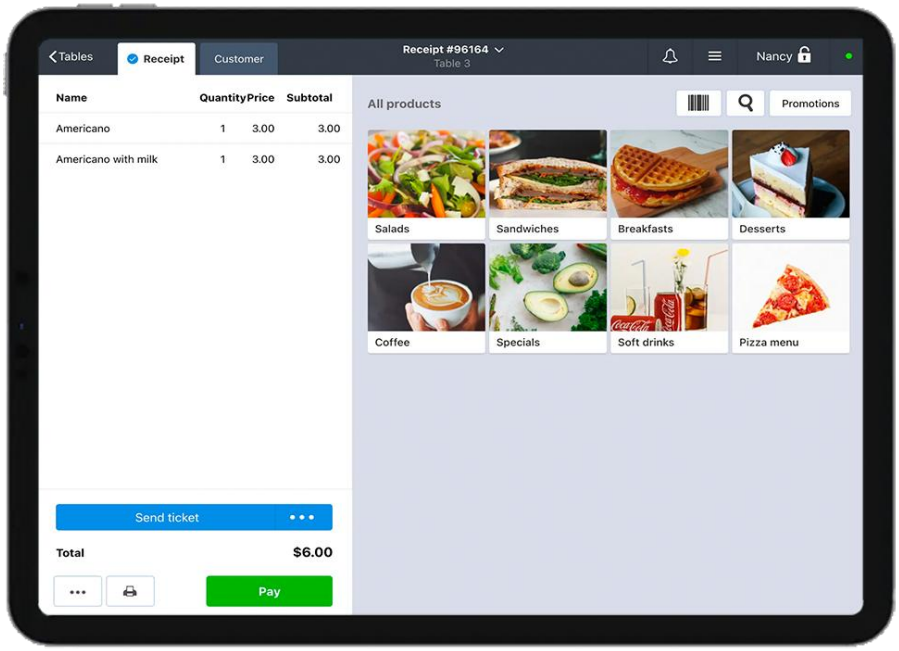

Рисунок 1.1 — Інтерфейс робочого місця касира хмарної POS-системи Poster

Другим значущим сегментом є глобальні платформи (Uber Eats, Glovo, Bolt Food). З технічної точки зору, це високонавантажені розподілені системи, побудовані на мікросервісній архітектурі.

Вони надають ресторанам доступ до широкої аудиторії, проте створюють для закладу ефект «чорної скриньки»: ресторан не отримує прямого доступу до профілю клієнта та історії його харчової поведінки. Взаємодія відбувається через стандартизовані API, які дозволяють передавати лише базову інформацію про замовлення. Інтерфейси агрегаторів уніфіковані для всіх закладів, що обмежує можливості кастомізації меню та впровадження власних алгоритмів рекомендацій.

Аналіз користувацького інтерфейсу популярних агрегаторів, таких як Bolt Food або Uber Eats , підтверджує фокус на логістиці, а не на нутриціології. Переваги: величезний асортимент, зручний пошук за категоріями кухонь, відстеження доставки в реальному часі. Недоліки: ефект «чорної скриньки» щодо складу страв — користувач бачить лише фото та короткий маркетинговий опис. Відсутній функціонал підрахунку сумарної калорійності кошика, виключення специфічних інгредієнтів (наприклад, «без цибулі») або попередження про перехресне забруднення, що є критичним для людей з важкими алергіями.

На рисунку 1.2 зображено інтерфейс додатку Bolt Food.

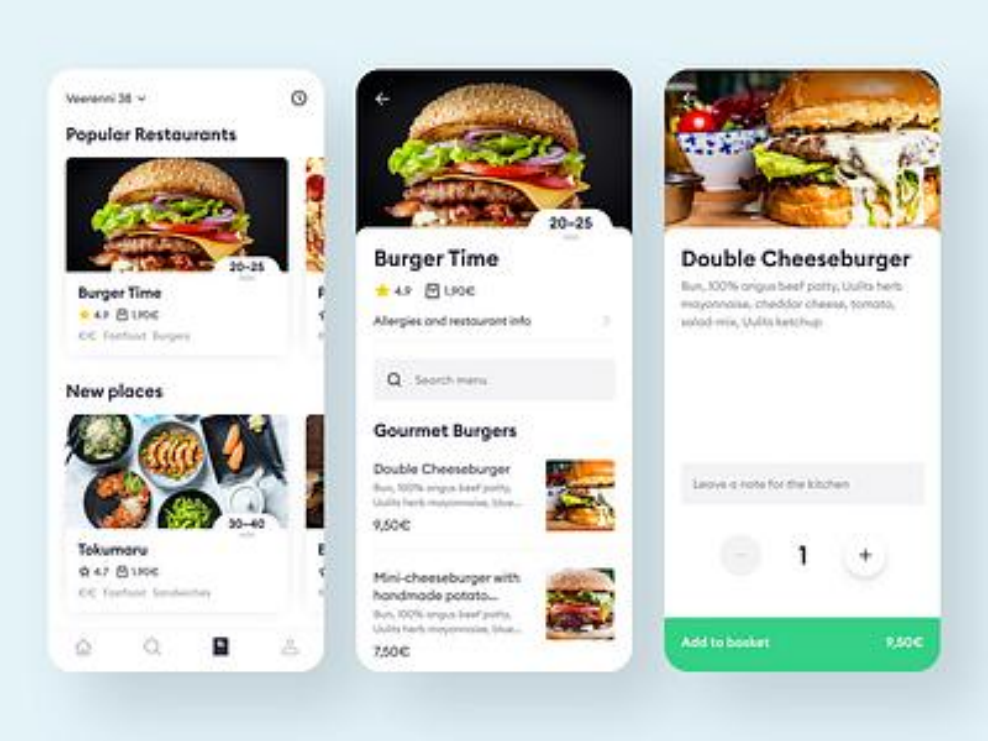

Рисунок 1.2 — Інтерфейс вибору страв у мобільному додатку агрегатора доставки Bolt Food

Найбільш близьким аналогом до розроблюваної системи є веб-додатки для замовлення за столом (QR-меню). Технологічно вони реалізуються як адаптивні веб-сайти (Responsive Web Design) або прогресивні веб-додатки (PWA), що не потребують встановлення.

Такі рішення інтегруються з POS-системою закладу для синхронізації стоп-листів та цін. Незважаючи на зручність інтерфейсу, більшість існуючих QR-меню функціонують як «цифрова копія» паперового меню. Вони відображають статичний контент без динамічної адаптації під користувача. Логіка рекомендацій у таких системах, якщо вона присутня, базується на простих правилах асоціації (Cross-selling) на кшталт «з цим товаром часто купують», ігноруючи фізіологічні потреби конкретного відвідувача [27].

Для наочного порівняння функціональних можливостей розглянутих класів систем у контексті персоналізації складено таблицю 1.2.

Таблиця 1.2 — Порівняльний аналіз архітектурних рішень для управління замовленнями

Характеристика	Хмарні POS-системи	Агрегатори доставки	QR-меню (Classic)	Розроблювана система
Архітектура	Клієнт-сервер (SaaS)	Мікросервісна	SPA / PWA	SPA + AI Module
Основний фокус	Облік та фінанси	Логістика доставки	Візуалізація меню	Персоналізація
Аналіз даних	Транзакційний (продажі)	Поведінковий (Big Data)	Статистичний	Семантичний (AI)
Врахування здоров'я	Відсутнє	Базові фільтри (теги)	Маркування іконками	Глибокий аналіз складу
Генерація порад	Ні	"Вам може сподобатися"	"З цим купують"	Пояснення (Explainable AI)

Узагальнюючи аналіз архітектури існуючих рішень, можна стверджувати, що на ринку домінують системи, орієнтовані на оптимізацію бізнес-процесів закладу, а не на підтримку здоров'я клієнта. Жоден із розглянутих класів систем

не має інтегрованого модуля семантичного аналізу складу страв на базі штучного інтелекту. Відсутність єдиного механізму, який би поєднував дані про інгредієнти (з бек-офісу ресторану) з медичним профілем користувача, створює технологічну прогалину, яку покликана заповнити розроблювана в даній роботі веб-система. Це дозволить не лише автоматизувати критично важливий процес перевірки безпечності харчування, але й трансформувати сам підхід до обслуговування, зміщуючи акцент з простого продажу страв на надання комплексного сервісу турботи про здоров'я відвідувача.

1.3 Аналіз проблем персоналізації обслуговування клієнтів з особливими дієтичними потребами

Забезпечення нутриціологічної безпеки та відповідності страв індивідуальним фізіологічним потребам споживачів стає одним із найскладніших викликів для сучасної індустрії гостинності. Зростання поширеності аліментарно-залежних захворювань, харчових алергій та метаболічних розладів трансформує процес вибору їжі з гедоністичної активності у задачу з високим рівнем ризику та когнітивного навантаження. Згідно з медичною статистикою, значна частка населення має непереносимість лактози, глютену або специфічних білків, що вимагає суворого виключення певних інгредієнтів з раціону. У традиційній моделі ресторанного обслуговування відповідальність за верифікацію складу страви покладається на клієнта, який змушений аналізувати обмежену інформацію в меню та покладатися на компетентність персоналу, що створює передумови для виникнення критичних помилок.

1.3.1 Проблема інформаційної асиметрії та ризики безпеки

Фундаментальною вадою існуючого процесу замовлення є інформаційна асиметрія між закладом та відвідувачем. Меню зазвичай містить лише маркетингові назви страв та перелік основних інгредієнтів, залишаючи «за кадром» складні соуси, маринади, спеції або сліди перехресної контамінації (Cross-contamination), які можуть містити небезпечні алергени.

Отримання детальної інформації вимагає прямої комунікації з офіціантом, що не завжди гарантує достовірність даних через вплив людського фактора. Офіціанти можуть:

— не володіти повною інформацією про технологічний процес приготування;

— забути попередити кухню про особливі вимоги клієнта;

— помилково інтерпретувати складність дієтичних обмежень.

На рисунку 1.3 зображено проблему комунікації клієнта з офіціантом та можливість появи ризиків.

Рисунок 1.3 — Схема виникнення ризиків внаслідок розриву комунікації в традиційній моделі обслуговування

У критичних випадках це призводить до анафілактичних реакцій та загрози життю споживача.

1.3.2 Проблема оптимізації нутриціологічної цінності

Окрім критичних аспектів безпеки, пов'язаних з алергенами, у сучасному суспільстві гостро постає проблема оптимізації харчування для досягнення конкретних фізіологічних та метаболічних цілей. Ця проблема охоплює широке коло споживачів: від професійних спортсменів, що контролюють кожну калорію для набору м'язової маси або «сушки», до пацієнтів з ендокринними порушеннями (цукровий діабет 1-го та 2-го типу), для яких точний підрахунок хлібних одиниць (вуглеводів) є умовою правильного дозування інсуліну.

В умовах традиційного ресторанного обслуговування клієнти стикаються з проблемою «інформаційної сліпої зони». Статичні меню, навіть якщо вони містять базову інформацію про калорійність (що часто є вимогою законодавства), надають лише усереднені дані для стандартної технологічної карти страви. Проте такі дані не враховують низку критичних факторів:

— глікемічне навантаження: для людей з інсулінорезистентністю важливо знати не лише кількість вуглеводів, а й їхній тип (швидкі чи повільні) та глікемічний індекс, який зазвичай не вказується у меню;

— динамічні модифікації: це одна з найскладніших проблем існуючих POS-систем. Коли клієнт просить «салат без соусу» або «замінити картоплю на овочі-гриль», нутриціологічна цінність страви кардинально змінюється. Традиційні системи не мають механізмів автоматичного перерахунку КБЖВ (калорії, білки, жири, вуглеводи) в реальному часі. В результаті, клієнт отримує страву зі зміненим складом, але продовжує орієнтуватися на старі, вже неактуальні цифри з меню;

— приховані нутрієнти: Складні ресторанні страви часто містять компоненти, які важко ідентифікувати візуально (олії, цукор у маринадах, загусники в соусах). Це унеможливлює точний підрахунок макронутрієнтів без використання спеціалізованих інструментів аналізу.

Інтерфейс спеціалізованих додатків для підрахунку калорій, таких як MyFitnessPal , забезпечує необхідну глибину даних, але позбавлений інтеграції з ресторанними процесами. Переваги**:** детальна візуалізація макро- та мікронутрієнтів, можливість встановлення особистих цілей, сканування штрих-кодів. Недоліки**:** повна ізоляція від меню ресторанів. Користувач змушений вручну вводити інгредієнти «на око», що призводить до значних похибок у розрахунках. Це підтверджує необхідність створення системи, яка б поєднала нутриціологічну глибину таких додатків зі зручністю замовлення в ресторані [6].

На рисунку 1.4 зображено інтерфейс додатку для контролю харчування MyFitnessPal.

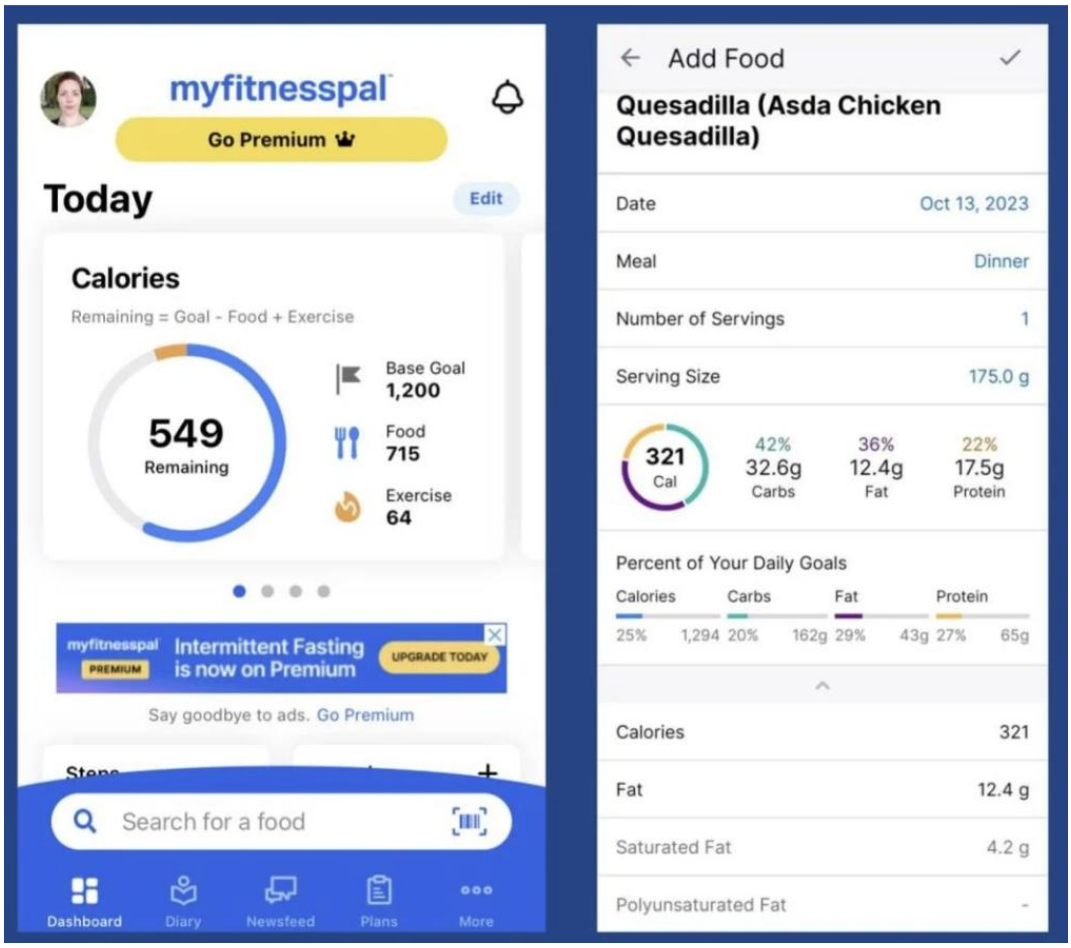

Рисунок 1.4 — Візуалізація нутриціологічної цінності у додатку для контролю харчування MyFitnessPal

Таким чином, відсутність інструментів для автоматизованого, динамічного розрахунку нутрієнтів під конкретний запит користувача є значним

технологічним бар'єром, який знижує якість життя споживачів, що свідомо ставляться до свого здоров'я.

1.3.3 Соціально-психологічні бар'єри обслуговування

Ситуація ускладнюється психологічним дискомфортом, який виникає у клієнтів з особливими потребами під час публічного з'ясування деталей замовлення. Необхідність детального опитування персоналу, прохання про заміну інгредієнтів та очікування перевірки складу шеф-кухарем знижують якість клієнтського досвіду та можуть викликати відчуття соціальної стигматизації. В умовах пікового завантаження ресторану персонал фізично не в змозі приділити достатньо часу для персоналізованого консультування кожного гостя, що робить автоматизацію цього процесу не просто бажаною опцією, а необхідною умовою для забезпечення інклюзивності та безпеки сервісу. Таким чином, існує об'єктивна потреба у впровадженні інтелектуальних систем, здатних виступити надійним посередником між складною рецептурою кухні та індивідуальним профілем клієнта, усуваючи ризики людських помилок.

1.4 Огляд методів штучного інтелекту в системах FoodTech

Інтенсивний розвиток технологій штучного інтелекту (Artificial Intelligence, AI) та машинного навчання (Machine Learning, ML) створив передумови для переходу від детермінованих алгоритмів управління до адаптивних інтелектуальних систем у сфері харчових технологій (FoodTech). Якщо на попередніх етапах автоматизації ключову роль відігравали системи управління базами даних (СУБД) із жорстко заданою логікою запитів, то сучасні підходи базуються на ймовірнісних моделях, здатних виявляти приховані закономірності у великих масивах неструктурованих даних.

У контексті персоналізації харчування найбільш релевантними є два напрями досліджень:

— рекомендаційні системи (Recommender Systems) - для ранжування та вибору страв;

— обробка природної мови (Natural Language Processing, NLP) - для інтерпретації складу страв та комунікації з користувачем.

1.4.1 Використання рекомендаційних систем у електронній комерції та сфері харчування

Рекомендаційні системи являють собою клас алгоритмів інформаційної фільтрації, метою яких є прогнозування "рейтингу" або "вподобання", яке користувач надасть об'єкту (у нашому випадку - страві). У науковій літературі виділяють три основні підходи до побудови РС, кожен з яких має специфічні обмеження при застосуванні в ресторанному бізнесі [7, 8].

1. Колаборативна фільтрація (Collaborative Filtering, CF). Цей метод базується на аналізі матриці взаємодій «користувач-об'єкт». Алгоритм припускає, що якщо користувачі A і B мали схожі смаки в минулому, то вони будуть мати схожі смаки й у майбутньому.

— математична сутність: застосування методів матричної факторизації (Matrix Factorization) або пошуку k-найближчих сусідів (k-NN);

— обмеження для FoodTech: метод страждає від проблеми «холодного старту» (Cold Start Problem) - неможливості надати рекомендацію новому користувачеві або новій страві. Крім того, CF ігнорує склад продукту. Те, що група користувачів з діабетом замовила піцу, не означає, що система повинна рекомендувати її іншому діабетику, оскільки попередні замовлення могли бути порушенням дієти.

2. Фільтрація на основі змісту (Content-Based Filtering, CBF). Метод генерує рекомендації шляхом порівняння атрибутів об'єкта з профілем інтересів

користувача. Для страв атрибутами виступають інгредієнти, калорійність, тип кухні, ціна.

— принцип роботи: страви та користувачі представляються як вектори у багатовимірному просторі ознак. Ступінь відповідності розраховується за допомогою мір подібності, наприклад, косинусної відстані (Cosine Similarity);

— переваги: вирішує проблему холодного старту для нових страв та дозволяє пояснити рекомендацію (наприклад, "рекомендовано, оскільки містить курку").

3. Системи на основі знань та обмежень (Constraint-Based / Knowledge-Based Systems). Це специфічний клас РС, де рекомендація формується не на основі статистики, а на основі явних правил та вимог користувача. У контексті медичних або дієтичних обмежень (алергії, непереносимості) цей підхід є критично важливим. Він оперує булевою логікою (дозволено/заборонено), гарантуючи "жорстку" фільтрацію небезпечних позицій, яку не можуть забезпечити ймовірнісні методи CF та CBF.

Таблиця 1.3 — Порівняльний аналіз методів рекомендацій у контексті FoodTech

Метод	Основний принцип	Переваги	Недоліки для дієтології
Collaborative Filtering	«Схожим людям подобається це»	Висока точність при великій кількості даних	Не враховує склад страв (ризик алергії), проблема «холодного старту»
Content-Based Filtering	«Вам подобається схожий склад»	Прозорість рекомендацій,	«Бульбашка фільтрів» (рекомендує одне

		робота з новими стравами	й те саме), складність виділення ознак
Constraint-Based	Це відповідає вашим правилам»	Гарантія безпеки (виключення алергенів)	Відсутність персоналізації смаків, складність створення бази правил
Гібридний підхід	Комбінація методів	Баланс між безпекою та вподобаннями	Висока складність реалізації

Аналіз показує, що для задачі персоналізованого харчування найбільш ефективним є гібридний підхід, який поєднує Constraint-Based фільтрацію (для безпеки та відсіювання алергенів) з Content-Based ранжуванням (для підбору страв за смаковими та нутриціологічними параметрами).

1.4.2 Можливості великих мовних моделей (LLM) для аналізу текстових даних меню

Традиційні рекомендаційні системи ефективно працюють зі структурованими даними. Проте значна частина інформації про страви в ресторанах представлена у неструктурованому текстовому вигляді (описи, коментарі шеф-кухаря). Поява великих мовних моделей (LLM), таких як GPT-4 або Google Gemini, побудованих на архітектурі Трансформер (Transformer), відкрила нові можливості для семантичного аналізу даних [10].

Ключовою перевагою LLM є здатність до розуміння контексту та виконання логічних висновків (Reasoning). Можна виділити такі напрями застосування LLM у розроблюваній системі:

— вилучення сутностей (Named Entity Recognition, NER). LLM здатна автоматично розпарсити текстовий опис страви (наприклад, «ніжне філе під вершковим соусом») та виділити конкретні інгредієнти («куряче філе», «вершки», «масло»), навіть якщо вони не вказані явно в базі даних. Це дозволяє виявляти приховані алергени (вершки -> лактоза) з високою точністю;

— генерація пояснень (Explainable AI, XAI). Однією з проблем класичних «чорних скриньок» машинного навчання є нездатність пояснити логіку прийняття рішення. LLM дозволяє реалізувати концепцію «прозорого штучного інтелекту», генеруючи аргументовані відповіді природною мовою;

— семантичний метчинг (Semantic Matching). На відміну від пошуку за ключовими словами, LLM використовує векторні представлення слів (Embeddings). Модель розуміє, що «пармезан» - це різновид «сиру» (молочний продукт), а «тофу» може бути веганським замінником м'яса.

Особливу увагу в сучасних дослідженнях приділяють методології RAG (Retrieval-Augmented Generation) [11, 12]. Технологія RAG дозволяє об’єднати генеративні можливості моделі з актуальними даними конкретного ресторану, запобігаючи виникненню «галюцинацій» (вигадуванню неіснуючих страв). Важливою архітектурною перевагою цього підходу є відсутність необхідності у ресурсомісткому донавчанні моделі при кожному оновленні асортименту, оскільки система оперує динамічним контекстом із зовнішньої бази знань. Це дозволяє забезпечити повну синхронізацію відповідей AI з реальним станом кухні, враховуючи поточні стоп-листи та зміни цін у режимі реального часу. Таким чином, інтеграція LLM у контур веб-системи управління замовленнями дозволяє трансформувати процес вибору їжі, перетворюючи систему з пасивного каталогу на інтерактивного нутриціологічного асистента.

1.5 Постановка задачі дослідження та визначення вимог до системи

На основі проведеного у попередніх підрозділах аналізу предметної області HoReCa та огляду існуючих технологічних рішень було виявлено ряд системних обмежень, притаманних сучасним засобам автоматизації обслуговування. Встановлено, що незважаючи на високий рівень розвитку транзакційних систем, сегмент інструментів для персоналізованої нутриціологічної підтримки клієнтів залишається недостатньо опрацьованим. Виявлена невідповідність між зростаючими потребами споживачів у прозорості харчування та обмеженим функціоналом наявного програмного забезпечення зумовлює необхідність переходу від загального огляду тенденцій до формалізації конкретних завдань проєктування нової системи.

Розробка ефективного програмного продукту, що інтегрує технології штучного інтелекту в операційні процеси реального часу, вимагає чіткої декомпозиції загальної проблематики на складові елементи. Процес визначення вимог виступає фундаментом для подальшого життєвого циклу розробки, дозволяючи трансформувати абстрактні потреби користувачів у чіткі технічні специфікації. Системний підхід до постановки задачі дозволяє мінімізувати архітектурні ризики та забезпечити відповідність кінцевого продукту критеріям безпеки та продуктивності.

1.5.1 Науково-прикладна проблема та мета роботи

Сучасний стан розвитку інформаційних технологій у сфері HoReCa характеризується диспропорцією між рівнем автоматизації операційних бізнес-процесів та якістю інформаційного супроводу кінцевого споживача. Аналіз архітектури існуючих програмних комплексів (POS-систем, агрегаторів доставки) свідчить, що домен нутриціологічної підтримки залишається критично недорозвиненим. Більшість рішень оперують статичними моделями даних, не пристосованими для обробки динамічних фізіологічних параметрів (алергічний профіль, метаболічні обмеження). Це створює ситуацію, коли вибір страв

базується на евристичних, часто помилкових судженнях клієнта, а не на об'єктивних даних.

Науково-прикладна проблема полягає у наявності суттєвого протиріччя. З одного боку, спостерігається експоненціальне зростання вимог споживачів до персоніфікації сервісу та гарантій харчової безпеки. З іншого боку, відсутні ефективні інструментальні засоби для автоматизованого семантичного аналізу неструктурованих описів страв та формування рекомендацій у реальному часі.

Метою роботи є наукове обґрунтування, архітектурне проєктування та програмна реалізація веб-орієнтованої інформаційної системи, спрямованої на трансформацію процесів обслуговування шляхом впровадження гібридного методу рекомендацій. Цей підхід базується на синергетичній інтеграції параметричного профілю користувача з можливостями генеративних нейронних мереж (LLM).

Для досягнення мети вирішуються такі завдання:

— математична формалізація моделі предметної області для опису взаємозв'язків сутностей «Користувач», «Інгредієнт» і «Страва»;

— розробка алгоритму гібридної фільтрації контенту, що поєднує жорстку логіку виключення алергенів (Hard Filtering) та ймовірнісну оцінку релевантності (Soft Filtering);

— створення методики інтеграції з Generative AI через розробку системних промптів для екстракції сутностей та генерації пояснень (Explainable AI);

— проєктування клієнт-серверної архітектури та програмна реалізація прототипу системи;

— проведення експериментального дослідження точності визначення алергенів та швидкодії системи.

1.5.2 Формування функціональних та нефункціональних вимог до системи

Виходячи з аналізу предметної області та поставлених завдань, сформульовано перелік вимог до розроблюваної системи.

Функціональні вимоги:

1. Управління профілем здоров'я: Система повинна дозволяти користувачеві вводити та редагувати антропометричні дані (вага, зріст, вік), обирати алергени зі списку та визначати дієтичні цілі (схуднення, набір маси).

2. Інтелектуальний пошук та рекомендації: Система має забезпечувати семантичний пошук страв (наприклад, «щось легке без глютену») та ранжувати результати відповідно до профілю користувача.

3. Автоматична фільтрація загроз: Система повинна в автоматичному режимі приховувати або маркувати страви, що містять критичні для користувача інгредієнти, використовуючи гібридний аналіз (БД + AI).

4. Пояснення рішень (Explainability): Для кожної рекомендованої страви система має генерувати текстове пояснення («Чому це мені підходить?»), обґрунтовуючи вибір на основі складу та КБЖВ.

Нефункціональні вимоги:

1. Продуктивність: Час генерації персоналізованої відповіді від AI-модуля не повинен перевищувати 3–5 секунд для забезпечення комфортного UX.

2. Масштабованість: Архітектура повинна підтримувати можливість додавання нових закладів та розширення бази інгредієнтів без зупинки системи.

3. Надійність даних: Система повинна мінімізувати вірогідність «галюцинацій» моделі шляхом використання RAG-архітектури (генерація на основі пошуку в актуальному меню).

2 МЕТОДИ ТА МОДЕЛІ ПОБУДОВИ СИСТЕМИ ПЕРСОНАЛІЗОВАНИХ РЕКОМЕНДАЦІЙ

2.1 Концептуальна модель предметної області та формалізація задачі вибору

У контексті розробки інтелектуальної системи підтримки прийняття рішень (СППР) для сфери ресторанного бізнесу, процес формування рекомендацій розглядається не як класична задача інформаційного пошуку, а як задача багатокритеріальної оптимізації (Multi-Criteria Decision Making, MCDM). Специфіка предметної області накладає на процес вибору низку суперечливих умов: необхідність задоволення гедоністичних потреб користувача (смак, вподобання) при одночасному дотриманні жорстких фізіологічних обмежень (безпека, дієта).

Замість класичного математичного моделювання, що оперує абстрактними змінними, доцільно застосувати теоретико-множинний підхід для опису об’єктів системи. Формально предметну область можна представити як взаємодію трьох ключових сутностей: множини користувачів, множини об'єктів вибору (меню) та множини контекстних умов.

1. Формалізація об'єктів вибору (Модель страв) Множина доступних для замовлення страв розглядається як набір складних об'єктів, де кожен елемент описується розширеним вектором характеристик. На відміну від простих облікових систем, у розробленій моделі кожна страва характеризується чотирма групами параметрів:

— ідентифікаційні та метадані: унікальний ідентифікатор, назва, категорія та ціна, що використовуються для базової навігації та фінансових розрахунків;

— нутриціологічний профіль: кількісний вектор макронутрієнтів, що включає значення калорійності, вмісту білків, жирів та вуглеводів. Ці параметри є основою для розрахунку відповідності дієтичним цілям;

— компонентний склад: множина інгредієнтів, що входять до складу страви. Цей набір даних є критичним для забезпечення безпеки харчування, оскільки саме він перевіряється на наявність алергенів;

— семантичний опис: текстова характеристика смакових якостей та способу приготування, яка трансформується у векторне представлення для аналізу штучним інтелектом.

2. Формалізація профілю користувача Користувач у системі моделюється не просто як обліковий запис, а як динамічний профіль, що містить набір обмежень та цілей. Структура профілю включає три компоненти:

— множина фізіологічних обмежень: це список продуктів та інгредієнтів, вживання яких є суворо неприпустимим для конкретного користувача (наприклад, алергени, непереносимість лактози). Ця множина діє як жорсткий фільтр (hard constraint) - наявність хоча б одного елемента з цього списку у страві робить її недоступною для рекомендації;

— цільовий запит: поточний намір користувача, виражений природною мовою (наприклад, «хочу легкий сніданок»). Цей запит обробляється LLM для виявлення семантичної відповідності;

— історичний контекст: дані про попередні замовлення, які використовуються для уточнення смакових вподобань (персоналізація на основі досвіду).

3. Постановка задачі оптимізації та алгоритм вибору Задачу формування персоналізованої рекомендації сформульовано як пошук такої підмножини страв із загального меню, яка максимізує функцію корисності для користувача при обов'язковому виконанні умов безпеки. Процес вирішення цієї задачі декомпозується на два послідовні етапи:

Етап А: Фільтрація обмежень (Забезпечення безпеки) Першочерговим завданням системи є гарантування нутриціологічної безпеки. На цьому етапі застосовується детермінована булева логіка. Система виконує операцію перевірки перетину множин: перевіряється, чи містить множина інгредієнтів

конкретної страви хоча б один елемент із множини алергенів користувача. Якщо перетин цих множин не є порожнім (тобто знайдено спільний елемент-алерген), страва автоматично виключається з розгляду. Таким чином формується «Множина допустимих альтернатив» - перелік страв, які є безпечними для споживання.

Етап Б: Ранжування за релевантністю (Максимізація корисності) Для страв, що пройшли етап перевірки безпеки, розраховується інтегральна оцінка релевантності. Ця оцінка формується на основі методу лінійної згортки критеріїв, враховуючи:

— семантичну подібність: ступінь відповідності текстового опису страви запиту користувача (визначається через косинусну відстань векторів);

— нутриціологічну відповідність: близькість калорійності та складу страви до поточних дієтичних цілей користувача.

Результатом роботи алгоритму є впорядкований список страв, де на перших позиціях знаходяться варіанти, що є гарантовано безпечними та максимально відповідають поточним потребам користувача. Така формалізація дозволяє перейти від емпіричного підбору до чіткої алгоритмічної реалізації гібридної системи рекомендацій.

2.2 Розробка гібридного методу рекомендацій

Традиційні підходи до побудови рекомендаційних систем, зокрема колаборативна фільтрація, яка базується на аналізі історії дій схожих користувачів, виявляють суттєві обмеження при застосуванні в сфері дієтологічного супроводу. Головними недоліками таких систем є проблема «холодного старту», що унеможливлює надання релевантних пропозицій новим користувачам, та, що є критично важливим, нездатність гарантувати суворе дотримання медичних обмежень. Імовірнісна природа колаборативних алгоритмів допускає певний відсоток похибки, що є неприпустимим у контексті

харчової безпеки, де рекомендація алергену може мати фатальні наслідки. Для вирішення цих проблем у роботі розроблено та обґрунтовано гібридну архітектуру рекомендаційної системи, яка реалізує послідовну, або каскадну, стратегію обробки даних.

Сутність запропонованого методу полягає у синергетичному поєднанні двох різнорідних алгоритмів фільтрації: детермінованого методу на основі обмежень (Constraint-Based Filtering) та імовірнісного методу на основі змісту (Content-Based Filtering). Така архітектура дозволяє розділити процес прийняття рішення на два етапи: забезпечення безпеки та оптимізацію релевантності. На першому етапі система діє як жорсткий шлюз, що категорично відсіює неприйнятні варіанти, а на другому - як інтелектуальний ранжувальник, що впорядковує безпечні альтернативи відповідно до смакових та нутриціологічних вподобань користувача.

Першим етапом роботи гібридного алгоритму є застосування методу фільтрації на основі обмежень. Враховуючи специфіку предметної області, пов'язану зі здоров'ям людини, на цьому етапі застосовується виключно детермінована булева логіка. Модуль оперує базою знань про інгредієнти та їхню приналежність до груп алергенів. Формально процес реалізується через теоретико-множинну операцію різниці. Система аналізує перетин множини інгредієнтів конкретної страви та множини алергенів, зазначених у профілі користувача. Якщо цей перетин не є порожнім, тобто знайдено хоча б один спільний елемент, страва автоматично отримує статус небезпечної та виключається з подальшого розгляду.

Така попередня фільтрація виконується на рівні бази даних або бекенд-логіки ще до залучення ресурсомістких алгоритмів штучного інтелекту. Це дозволяє досягти подвійного ефекту: по-перше, суттєво знизити обчислювальне навантаження на систему, оскільки велика мовна модель не витрачає ресурси на аналіз апріорі непридатних варіантів; по-друге, повністю усунути ризик так званих «галюцинацій» нейромережі в аспекті безпеки. Навіть якщо штучний

інтелект помилиться в оцінці смакових якостей, він фізично не зможе порекомендувати страву з алергеном, оскільки такі позиції вже були вилучені з контексту на попередньому етапі.

Другим етапом є застосування методу контентної фільтрації для аналізу складу та семантичних характеристик страв, що пройшли перевірку безпеки. У рамках розробленої системи класичний підхід, що базується на співставленні ключових слів або тегів, було модифіковано. Замість поверхневого порівняння застосовується глибокий векторний аналіз атрибутів. Профіль кожної страви та профіль інтересів користувача трансформуються у вектори в багатовимірному просторі ознак. Вектор страви формується з двох компонентів: нутриціологічного вектора, що містить нормалізовані числові значення калорійності та макронутрієнтів, та семантичного вектора (ембеддінга), отриманого в результаті обробки текстового опису страви великою мовною моделлю.

Оцінка релевантності або ступеня подібності між запитом користувача та конкретною стравою розраховується як зважена сума косинусних відстаней у відповідних підпросторах. Використання косинусної міри подібності дозволяє визначити семантичну близькість навіть за відсутності прямих лексичних збігів. Це надає системі здатність рекомендувати страви, які користувач раніше не замовляв, але які за своїм складом та нутриціологічним профілем схожі на його звичний раціон або відповідають поточним дієтичним цілям. Наприклад, система може запропонувати кіноа з овочами як альтернативу гречаній каші, базуючись на близькості їхніх векторних представлень.

Таким чином, розроблений гібридний метод забезпечує баланс між суворими вимогами безпеки та гнучкістю персоналізації. Метод обмежень виступає гарантом відсутності алергічних загроз, тоді як метод контентної фільтрації, підсилений можливостями штучного інтелекту, забезпечує високу точність та релевантність рекомендацій, адаптуючи вибір під індивідуальні потреби кожного користувача.

2.3 Методика застосування генеративного штучного інтелекту для семантичного аналізу

Інтеграція великих мовних моделей (LLM) у систему забезпечує перехід до декларативної парадигми проєктування, де AI діє як інтелектуальний посередник для інтерпретації намірів користувача. **Ключовим інструментом керування якістю відповідей виступає методика інженерії запитів (Prompt Engineering), що дозволяє структурувати вхідні дані для точного семантичного аналізу.** Для забезпечення актуальності знань та усунення ризику «галюцинацій» застосовано архітектурний патерн RAG (Retrieval-Augmented Generation), який передбачає динамічну ін'єкцію зовнішнього контексту з бази даних у запит до моделі. Це не лише гарантує повну синхронізацію згенерованих рекомендацій з реальним станом меню та наявністю страв, **а й дозволяє реалізувати функцію аргументованого пояснення вибору, перетворюючи систему на прозорого інтерактивного асистента.**

Для наочної демонстрації інформаційних потоків між базою даних, сервером та AI-моделлю розроблено відповідну схему взаємодії. На рисунку 2.1 зображено схему реалізації архітектурного патерну RAG у системі.

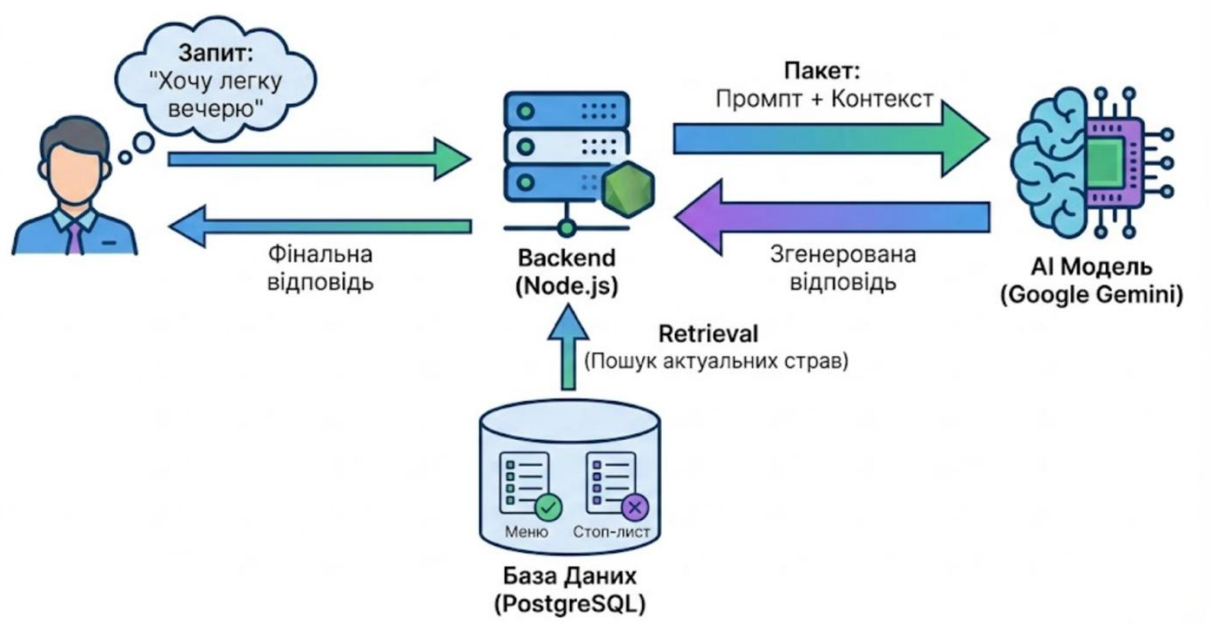

Рисунок 2.1 — Схема реалізації архітектурного патерну RAG у системі

2.3.1 Методологія інженерії запитів (Prompt Engineering)

Ключовим фактором, що визначає якість та релевантність роботи LLM, є семантична та структурна організація вхідного запиту. У роботі розроблено та застосовано методику структурованого промптингу (Structured Prompting), яка базується на логічній декомпозиції інструкції на функціональні блоки: рольову установку, контекст даних, ланцюжок міркувань та обмеження виводу.

На етапі рольового моделювання (Persona Definition) задається системна роль, що налаштовує тональність та експертність моделі. Інструкція діяти як «професійний нутриціолог та шеф-кухар» активує відповідні кластери у латентному просторі моделі, фокусуючи її увагу на біохімічному складі продуктів та їх гастрономічній сумісності, а не просто на лінгвістичній схожості слів.

Критично важливим етапом є ін'єкція контексту (Context Injection). У тіло промпту у форматі JSON передається профіль користувача, що включає дані про алергії, рівень фізичної активності та поточну мету (наприклад, схуднення або набір м'язової маси). Поруч із профілем передається вектор меню - перелік страв, які вже пройшли попередній етап фільтрації обмежень. Для кожної страви надається детальний опис складу та макронутрієнтів, що слугує фактичною базою для прийняття рішень моделлю.

Для підвищення логічної зв'язності рекомендацій застосовується техніка «Ланцюжок міркувань» (Chain-of-Thought Reasoning). Моделі надається явна інструкція проаналізувати кожен інгредієнт на відповідність профілю користувача перед тим, як формувати фінальний висновок. Це дозволяє моделі емулювати процес людського мислення, послідовно перевіряючи страву на наявність, наприклад, продуктів з високим глікемічним індексом, якщо метою користувача є контроль цукру.

Завершальним елементом методології є суворе обмеження формату виводу (Output Schema Enforcement). Для забезпечення можливості програмної обробки

відповіді на стороні сервера, моделі забороняється генерувати вільний текст. Відповідь повинна бути валідним JSON-об'єктом із заздалегідь визначеною схемою, що містить ідентифікатори рекомендованих страв та текстове поле з поясненням вибору. Це дозволяє уникнути проблем із парсингом та інтеграцією результатів у користувацький інтерфейс.

Така компоновка блоків дозволяє перетворити абстрактний запит на чітку інструкцію для алгоритму. На рисунку 2.2 зображено структуру системного промпту для генерації рекомендацій.

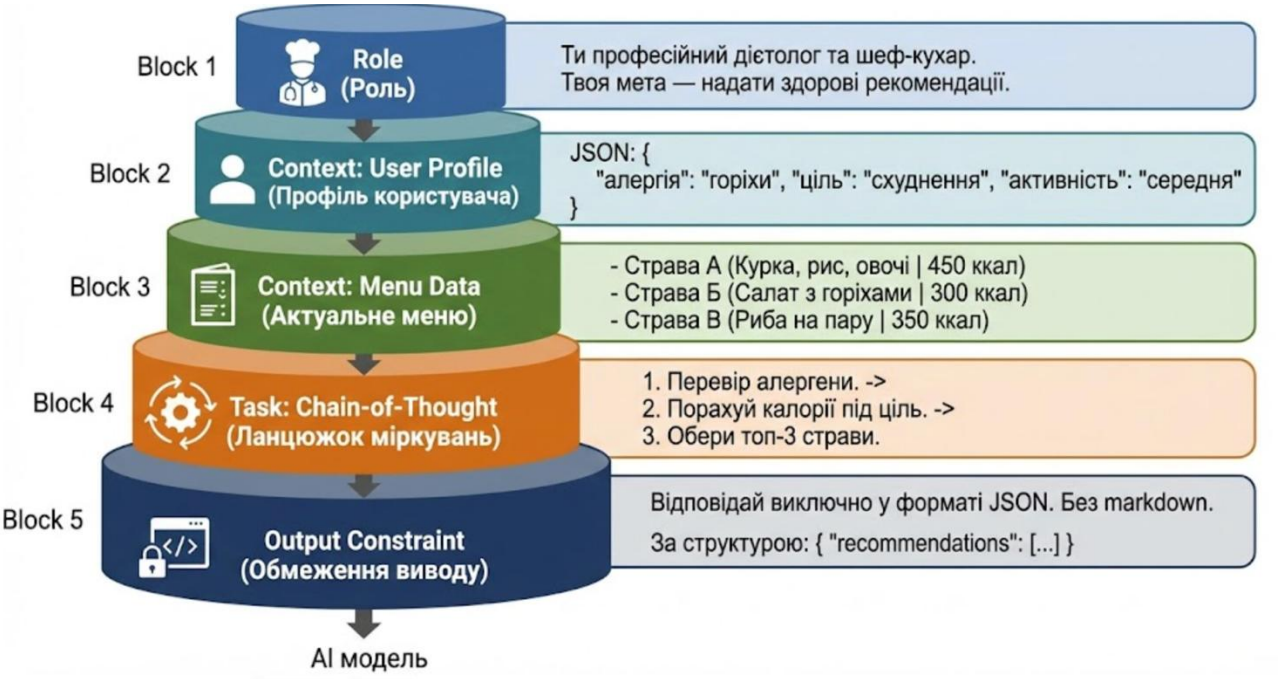

Рисунок 2.2 — Структура системного промпту для генерації рекомендацій

2.3.2 Алгоритм валідації та верифікації відповідей моделі

Враховуючи стохастичну природу генеративних моделей, існує ненульова ймовірність виникнення помилок двох типів: синтаксичних (порушення структури JSON) та семантичних (рекомендація неіснуючих страв). Для нівелювання цих ризиків розроблено багаторівневий алгоритм пост-обробки результатів.

Перший рівень - синтаксична валідація та відновлення. Отриманий від LLM текстовий рядок проходить через парсер формату JSON. У випадку виникнення помилки десеріалізації, система ініціює механізм самокорекції (Self-Correction Loop): помилковий рядок разом з повідомленням про помилку відправляється назад у модель з інструкцією виправити синтаксис. Цей цикл може повторюватися обмежену кількість разів до отримання валідної структури.

Другий рівень - семантична верифікація сутностей (Entity Linking). Після успішного парсингу виконується перевірка цілісності даних. Система екстрагує ідентифікатори рекомендованих страв та звіряє їх з базою даних. Умова валідності є суворою: рекомендація вважається допустимою тоді і тільки тоді, коли згенерований ідентифікатор присутній у списку доступних страв, переданих у контексті. Якщо модель «згалюцинувала» ідентифікатор, така позиція автоматично вилучається з видачі.

Третій рівень - фільтрація безпеки контенту (Safety Layer). Згенеровані текстові пояснення («Чому ця страва вам підходить») проходять перевірку на наявність стоп-слів та етичну відповідність. Це є критично важливим для уникнення генерації шкідливих медичних порад або гарантій лікувального ефекту їжі.

Окрім безпосередньої фільтрації, важливим елементом архітектури є механізм обробки критичних збоїв (Fallback Strategy). У сценарії, коли після проходження всіх етапів валідації результуючий список рекомендацій виявляється порожнім (наприклад, якщо всі згенеровані ідентифікатори були визнані "галюцинаціями" або містили небезпечний контент), система автоматично перемикається на детермінований алгоритм резервного пошуку. Цей алгоритм ігнорує семантичний контекст запиту і вибирає з бази даних найбільш популярні страви, які гарантовано відповідають жорстким критеріям безпеки (відсутність алергенів). Такий підхід забезпечує безперервність обслуговування (High Availability), гарантуючи, що користувач отримає

безпечну пропозицію навіть за умов тимчасової нестабільності інтелектуального модуля.

Логічна послідовність цих перевірок та умовні переходи, що забезпечують стабільність роботи системи, представлені у вигляді блок-схеми. На рисунку 2.3 зображено блок-схему алгоритму валідації та верифікації відповідей моделі.

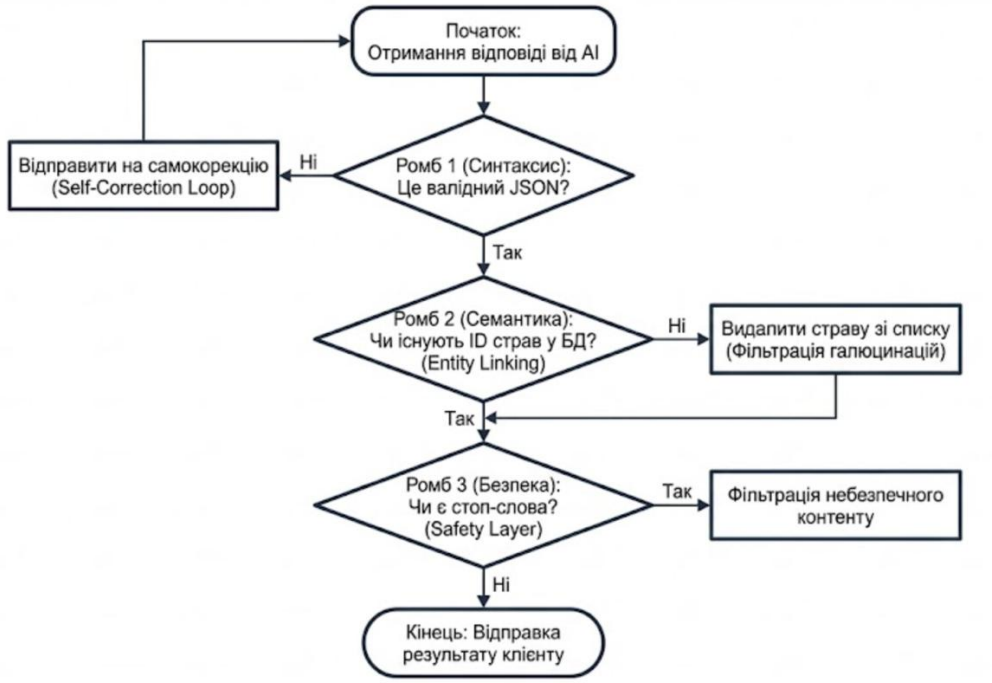

Рисунок 2.3 — Блок-схема алгоритму валідації та верифікації відповідей моделі

Запропонована методика дозволяє гармонійно поєднати гнучкість обробки природної мови та потужні аналітичні здібності великих мовних моделей з надійністю та детермінованістю класичних програмних алгоритмів. Реалізація багатоступеневого контролю перетворює стохастичну генерацію тексту на керований процес, створюючи безпечний та ефективний інструмент персоналізації харчування. Такий гібридний підхід гарантує, що система залишається чутливою до контексту та побажань користувача, але водночас безкомпромісно дотримується критичних вимог щодо безпеки інгредієнтів, що є необхідною умовою для впровадження технологій штучного інтелекту в сферу дієтології та охорони здоров'я.

3 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ТА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ ВЕБ-СИСТЕМИ

3.1 Обґрунтування архітектурного рішення та технологічного стеку

Проєктування програмного комплексу, орієнтованого на автоматизацію процесів обслуговування з елементами глибокого інтелектуального аналізу даних, вимагає застосування системного підходу до вибору архітектурного патерну. Специфіка предметної області (ресторанний бізнес та нутриціологія) накладає низку суперечливих вимог до системи: з одного боку, необхідність забезпечення миттєвого відгуку інтерфейсу для утримання уваги користувача, а з іншого - потреба у виконанні ресурсомістких обчислень, пов'язаних із семантичним аналізом меню та генерацією пояснень штучним інтелектом.

Виходячи з поставлених у другому розділі завдань, архітектуру системи побудовано за принципом розподіленої клієнт-серверної взаємодії (Client-Server Architecture). Як основний технологічний стек обрано PERN (PostgreSQL, Express, React, Node.js), що забезпечує єдиний мовний простір (JavaScript/TypeScript) на всіх рівнях розробки [20].

Реалізація серверної платформи базується на середовищі Node.js у поєднанні з фреймворком Express. На відміну від традиційних багатопотокових платформ, Node.js використовує подійно-орієнтовану модель (Event-Driven Architecture) з неблокуючим введенням-виведенням. Це архітектурне рішення є критично важливим для даної системи, оскільки взаємодія з великими мовними моделями (Google Gemini) неминуче передбачає певну затримку (latency) при генерації відповіді. Асинхронна природа Node.js дозволяє серверу продовжувати ефективну обробку тисяч інших конкурентних запитів, наприклад, завантаження каталогу страв іншими користувачами, поки один із потоків очікує завершення генерації контенту від AI, що забезпечує високу пропускну здатність системи в цілому [18].

На рисунку 3.1 зображено діагарму розгортання додатку

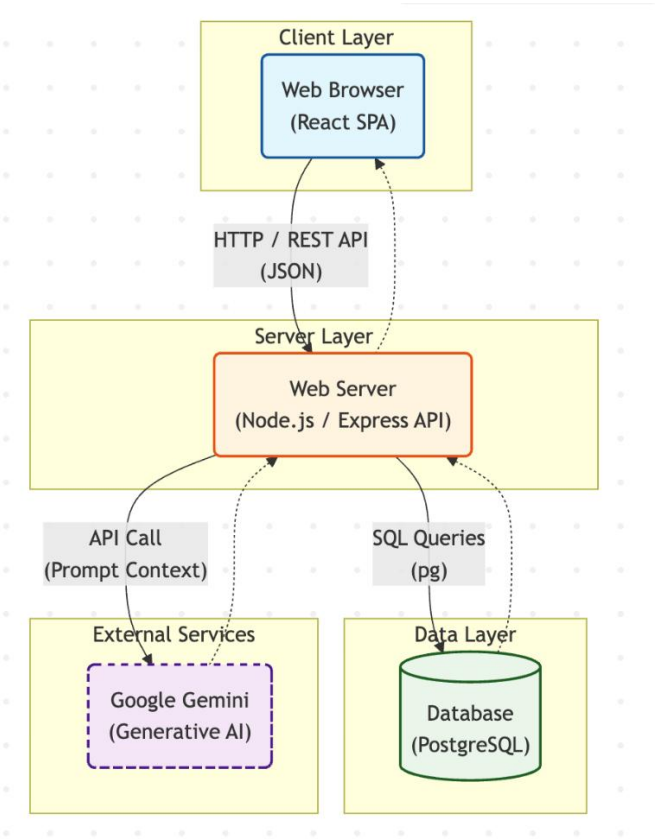

Рисунок 3.1 — Deployment Diagram

Ключовим фактором, що визначав вибір технологічного стеку, стала необхідність реалізації гібридної моделі рекомендацій. Оскільки система оперує критично важливими даними про здоров'я користувача (алергени, дієтичні обмеження), архітектура повинна гарантувати сувору типізацію та валідацію даних на всіх етапах їх обробки. Це зумовило вибір технологій, що підтримують статичну типізацію або жорсткі схеми даних, для мінімізації ризику програмних помилок, які можуть призвести до надання некоректної рекомендації.

Крім того, архітектура системи спроєктована з урахуванням принципів масштабованості (Scalability) та модульності. Функціональні блоки системи (управління меню, обробка замовлень, чат-бот дієтолог) реалізовані як слабкозв'язані компоненти. Це дозволяє в майбутньому розширювати

функціонал, наприклад, додаючи інтеграцію з фітнес-трекерами, без необхідності перебудови ядра системи.

Застосований підхід дозволяє вирішити головну проблему дослідження - подолання інформаційного розриву між складним складом страв та індивідуальними потребами клієнта, надаючи інструмент, який трансформує сухі технічні дані меню у зрозумілі персоналізовані поради.

3.1.1 Обґрунтування архітектурного стилю та реалізації клієнтської частини

Розробка клієнтського інтерфейсу (Frontend) для системи персоналізованого харчування вимагає вирішення специфічної проблеми: забезпечення безперервності користувацького досвіду (Seamless User Experience) в умовах складних асинхронних операцій. Оскільки ключовою функцією системи є інтерактивний діалог з інтелектуальним асистентом, будь-які затримки, пов'язані з перезавантаженням сторінки при відправці повідомлень або застосуванні фільтрів меню, можуть негативно вплинути на сприйняття сервісу та призвести до втрати контексту взаємодії.

Для нівелювання цих ризиків клієнтську частину спроєктовано як односторінковий веб-застосунок (Single Page Application, SPA). На відміну від традиційної моделі (Multi-Page Application), архітектура SPA завантажує оболонку застосунку одноразово, а подальша взаємодія з сервером відбувається у фоновому режимі через асинхронні запити. Вибір даного архітектурного підходу зумовлений такими факторами:

1. Бібліотека React.js (Візуалізація): Обрана завдяки механізму віртуального DOM (Virtual Document Object Model), який мінімізує кількість дорогих операцій перемальовування реального дерева елементів браузера. У контексті розроблюваної системи це критично важливо для модуля чату з AI: при

отриманні потокового повідомлення від нейромережі (stream response) інтерфейс оновлюється плавно, без мерехтіння, створюючи відчуття живої розмови [13, 17].

2. Redux Toolkit (Управління станом): Оскільки система оперує складними розподіленими даними (профіль користувача, вміст кошика, історія повідомлень чату, список алергенів), виникла потреба у централізованому сховищі стану (Global Store). Використання Redux дозволяє уникнути проблеми надлишкової передачі даних через проміжні компоненти ("prop drilling") та забезпечує передбачуваність потоків даних між незалежними модулями системи.

3. Tailwind CSS (Стилізація та адаптивність): Для реалізації інтерфейсу використано підхід Utility-First. Це дозволило ефективно реалізувати принцип Mobile First, що є обов'язковою вимогою для систем у сфері HoReCa, оскільки більшість користувачів здійснюють замовлення зі смартфонів. Адаптивна верстка забезпечує коректне відображення інтерактивних карток страв на екранах будь-якого розширення.

Для забезпечення масштабованості архітектури та уникнення проблеми високої зв'язності коду (Tight Coupling), структуру проєкту організовано згідно з методологією Feature-Sliced Design (FSD). Це сучасний архітектурний патерн, який передбачає декомпозицію функціоналу на ієрархічні шари:

— App: Шар глобальних налаштувань, стилів та провайдерів контексту.

— Pages: Шар компоновки сторінок (MenuPage, ProfilePage, ChatPage), що формуються з готових віджетів.

— Features: Шар самодостатніх бізнес-функцій, що реалізують конкретні сценарії взаємодії (authByEmail - авторизація, addToCart - додавання в кошик, sendMessageToAI - комунікація з LLM).

— Entities: Шар бізнес-сутностей (User, Dish, Order), що містять моделі даних та відповідні UI-компоненти.

— Shared: Шар перевикористовуваних інфраструктурних компонентів (кнопки, поля вводу) та утиліт, що не мають прив'язки до бізнес-логіки [21].

Така стратифікація коду дозволяє ізолювати логіку роботи з AI-рекомендаціями від логіки оформлення замовлення, що підвищує відмовостійкість системи: помилка в одному модулі не призводить до зупинки всього застосунку.

На Рисунку 3.2 зображено діаграму варіантів використання, яка демонструє функціональні можливості клієнтської частини системи:

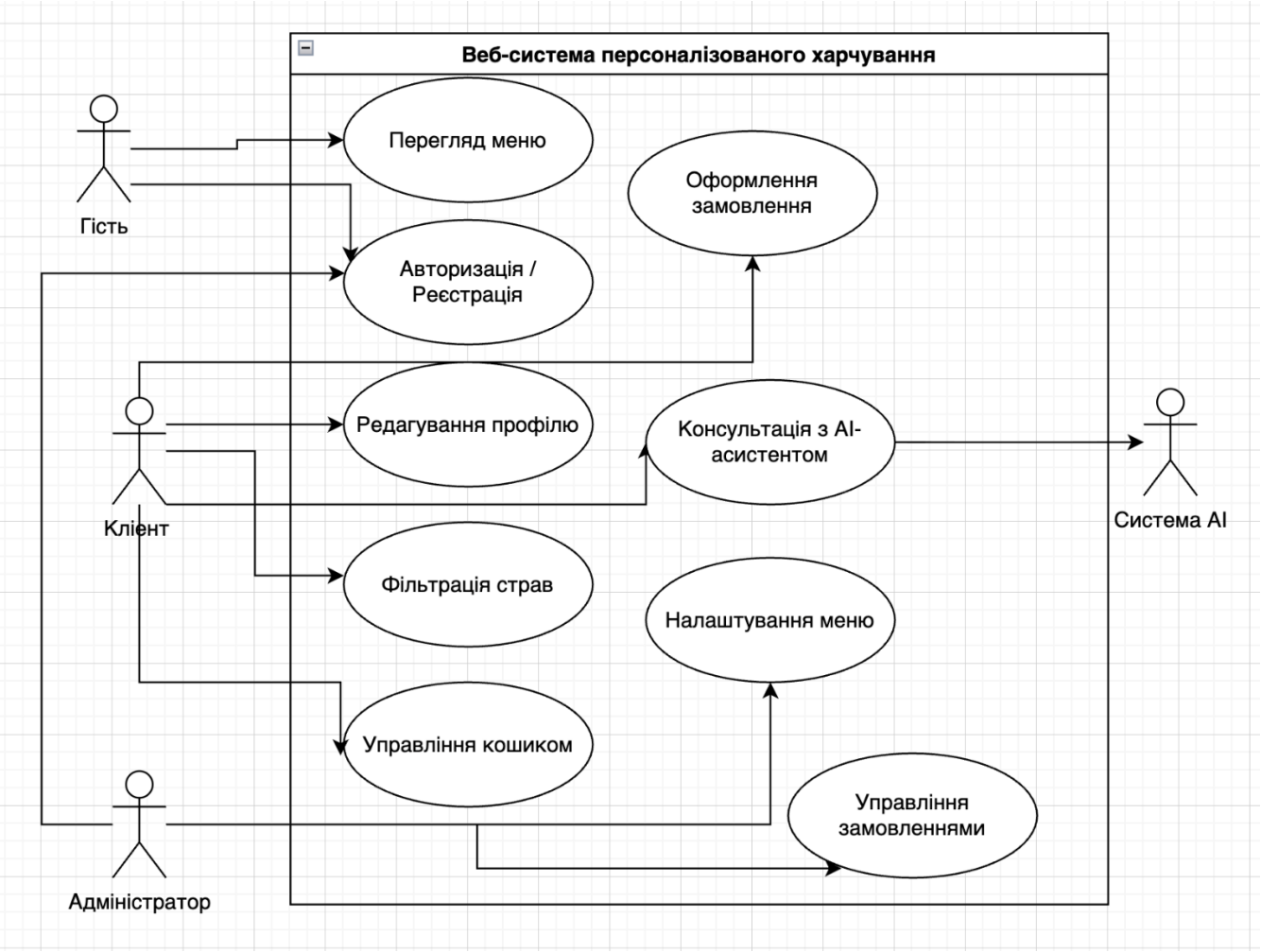

Рисунок 3.2 — Use Case Diagram

3.1.2 Обґрунтування серверної платформи та стратегії обробки даних

Проєктування серверної складової для систем, що інтегрують генеративний штучний інтелект, має суттєві відмінності від розробки класичних

транзакційних веб-сервісів. Головним технічним викликом є висока латентність (latency) операцій генерації тексту: час очікування відповіді від великої мовної моделі (LLM) може варіюватися від 2 до 10 секунд. У традиційних блокуючих архітектурах масове звернення користувачів до AI-асистента могло б призвести до вичерпання пулу потоків та відмови в обслуговуванні.

Для вирішення цієї проблеми серверну архітектуру побудовано на базі платформи Node.js. Вибір цього технологічного стеку зумовлений насамперед ефективністю його подійно-орієнтованої моделі (Event-Driven Architecture). Використання механізму Event Loop та неблокуючого введення-виведення (Non-blocking I/O) дозволяє серверу обробляти тисячі конкурентних з'єднань в одному потоці. У практичному вимірі це означає, що поки система очікує відповідь від Google Gemini API для одного клієнта, вона продовжує обробляти HTTP-запити інших користувачів, забезпечуючи високу пропускну здатність (Throughput) системи.

Організація логіки обробки запитів реалізована за допомогою фреймворку Express.js, який використано для побудови конвеєра обробки запитів (Middleware Pipeline). Архітектура middleware дозволяє модульно підключати функціонал валідації вхідних даних (з використанням бібліотек Joi або Zod), логування та централізованої обробки помилок безпосередньо перед тим, як запит потрапить до дороговартісного контролера штучного інтелекту. Безпосередня взаємодія з мовною моделлю реалізована через офіційний SDK Google Generative AI. Це забезпечує нативну підтримку потокової передачі даних (streaming), що дозволяє відправляти частини згенерованого тексту на клієнт у міру їх надходження, суттєво покращуючи сприйняття швидкодії системи користувачем (Perceived Performance).

З точки зору безпеки та масштабованості, комунікація побудована згідно з принципами REST та Stateless-архітектури. Сервер не зберігає стан сесії клієнта в оперативній пам'яті, використовуючи для авторизації токени стандарту JWT (JSON Web Token). Такий підхід значно спрощує масштабування системи,

оскільки будь-який екземпляр сервера може обробити запит, маючи лише секретний ключ для валідації підпису токена. Така архітектура створює надійний фундамент для функціонування інтелектуальної системи, нівелюючи ризики, пов'язані із затримками зовнішніх AI-сервісів.

3.2 Проєктування функціональної структури системи

Етап проєктування функціональної структури є визначальним для забезпечення відповідності програмного продукту вимогам предметної області. Для формалізації вимог та візуалізації поведінкових аспектів системи було застосовано об'єктно-орієнтований підхід із використанням уніфікованої мови моделювання (UML). Це дозволило чітко розмежувати зони відповідальності між користувачами та автоматизованими компонентами, а також деталізувати алгоритми прийняття рішень у процесі персоналізації харчування.

3.2.1 Ідентифікація акторів та проєктування прецедентів використання

На першому етапі моделювання було визначено межі системи та ідентифіковано множину зовнішніх сутностей (акторів), які ініціюють виконання бізнес-логіки. Архітектура доступу до функціоналу базується на рольовій моделі, що включає три ключові категорії користувачів.

Базовим актором є «Гість» (Guest) - неавторизований користувач, взаємодія якого з системою обмежена режимом перегляду (Read-only). Гість має можливість ознайомитися з публічним каталогом страв, переглянути загальну інформацію про заклад та ініціювати процес реєстрації. Ця роль є точкою входу в систему, і її головна мета - конвертація відвідувача у зареєстрованого клієнта.

Роль «Клієнт» (Registered User) успадковує права Гостя, але отримує доступ до захищених функцій, що становлять ядро наукової новизни роботи. Ключовою відмінністю цього актора є наявність асоційованого профілю здоров'я, що дозволяє системі персоналізувати видачу контенту. Клієнт може

керувати налаштуваннями алергенів, спілкуватися з інтелектуальним асистентом, формувати замовлення та переглядати історію транзакцій для аналізу спожитих нутрієнтів.

Окремою сутністю виступає «Інтелектуальний агент» (AI Agent). На відміну від людських акторів, це внутрішній системний компонент, який функціонує як активний учасник процесу. Його задача полягає в обробці природної мови (NLP), генерації пояснень та динамічному ранжуванні страв. Взаємодія між Клієнтом та AI-агентом реалізується через прецедент «Отримання Smart-рекомендації».

Функціональну специфікацію системи формалізовано у вигляді діаграми варіантів використання, яка візуалізує сценарії взаємодії акторів з системою.

Особливу увагу при проєктуванні приділено групі прецедентів «Управління профілем та персоналізація». Цей блок функціоналу відповідає за збір та валідацію вхідних даних для алгоритму рекомендацій. Сценарій налаштування фізіологічних параметрів передбачає введення даних про харчові непереносимості (наприклад, глютен, лактоза, горіхи) та визначення поточної дієтичної мети. Ці дані зберігаються у нормалізованому вигляді в базі даних і надалі використовуються як контекст (Prompt Context) для генеративної моделі.

Група прецедентів «Замовлення та транзакції» охоплює повний цикл електронної комерції: від додавання позицій у кошик до фіскалізації оплати. Важливою особливістю розробленої структури є інтеграція прецеденту «Аналіз історії» з модулем рекомендацій: система враховує попередні замовлення користувача для уточнення його смакових вподобань у майбутніх сесіях.

3.2.2 Моделювання алгоритму інтелектуального підбору страв

Центральним та найбільш технологічно складним бізнес-процесом системи є отримання персоналізованих рекомендацій. На відміну від лінійних сценаріїв пошуку в класичних інтернет-магазинах, цей процес має адаптивний

характер і включає етапи детермінованої та ймовірнісної обробки даних. Для деталізації логіки цього процесу розроблено діаграму діяльності (Activity Diagram).

Алгоритм починається з ініціалізації запиту користувачем через інтерфейс чату. Система виконує первинну перевірку контексту, аналізуючи наявність заповненого профілю здоров'я. У цьому вузлі розгалуження (Decision Node) можливі два сценарії розвитку подій. Якщо профіль відсутній або неповний, потік керування перенаправляється на підпроцес «Анкетування» (Onboarding) для збору критично важливих метаданих. Якщо ж дані наявні, система переходить до етапу фільтрації.

Етап обробки даних реалізовано за гібридною схемою. Спочатку виконується попередня фільтрація (Hard Filtering): система звертається до бази даних і виключає з вибірки всі страви, що містять інгредієнти з «чорного списку» користувача. Цей крок є обов'язковим для забезпечення безпеки і виконується на рівні SQL-запитів, що гарантує 100% відсіювання алергенів. Тільки після цього очищений масив даних передається до AI-агента.

Наступним кроком є семантичний аналіз та генерація (RAG). Велика мовна модель отримує запит користувача, його профіль та відфільтроване меню. На основі цього контексту модель ранжує страви за релевантністю та генерує текстове пояснення вибору. Завершується процес візуалізацією результатів у вигляді інтерактивних карток, з якими користувач може взаємодіяти (відкрити деталі або додати в кошик).

3.2.3 Моделювання життєвого циклу замовлення

Процес оформлення та обробки замовлення (Checkout Flow) спроєктовано з урахуванням специфіки ресторанного бізнесу, де критично важливими є актуальність залишків та швидкість передачі даних на кухню. Життєвий цикл замовлення проходить через низку послідовних станів.

Процес ініціюється підтвердженням складу кошика клієнтом. На цьому етапі система виконує синхронну валідацію: перевіряється актуальність цін (які могли змінитися під час сесії) та наявність необхідних інгредієнтів на складі (Inventory Check). У разі успішної перевірки створюється транзакція зі статусом PENDING (Очікування).

Наступним етапом є фінансова транзакція. Інтеграція з платіжним шлюзом забезпечує безпечну обробку платежу. Після отримання підтвердження від платіжної системи (callback), статус замовлення атомарно змінюється на CONFIRMED (Підтверджено), а інформація передається на термінал кухні.

Унікальною особливістю розробленого бізнес-процесу є фінальний етап - нутриціологічна аналітика. Після успішного завершення замовлення система асинхронно оновлює статистику спожитих користувачем калорій та макронутрієнтів. Це дозволяє замкнути цикл персоналізації: наступні рекомендації AI будуть враховувати, що користувач вже спожив певну кількість калорій, і пропонуватимуть легші страви для дотримання добової норми.

Проведене моделювання дозволило формалізувати логіку роботи системи, виявити потенційні виняткові ситуації та закласти архітектурну основу для програмної реалізації компонентів.

3.3 Моделювання бізнес-процесів системи та взаємодії компонентів

Для деталізації динамічних аспектів функціонування системи та виявлення потенційних вузьких місць у логіці обробки даних було проведено моделювання ключових бізнес-процесів. На відміну від статичних структурних діаграм, моделювання поведінки дозволяє формалізувати послідовність дій, потоки керування та часові характеристики взаємодії між модулями. Для візуалізації цих процесів застосовано нотацію UML, зокрема діаграми діяльності (Activity Diagrams) для опису алгоритмів та діаграми послідовності (Sequence Diagrams) для опису технічної комунікації.

Особлива увага в процесі моделювання приділяється сценаріям, що передбачають асинхронну обробку запитів та взаємодію із зовнішніми інтелектуальними агентами. Оскільки інтеграція великих мовних моделей (LLM) вносить елемент варіативності у час відгуку та структуру відповіді, розроблені моделі дозволяють чітко визначити механізми обробки затримок (latency management), станів очікування та валідації даних ще до етапу імплементації. Такий підхід забезпечує узгодженість між клієнтською частиною (Frontend) та серверною логікою (Backend), мінімізуючи ризики виникнення архітектурних колізій та гарантуючи цілісність даних під час передачі контексту користувача (профілю алергенів та історії замовлень) між ізольованими компонентами системи.

3.3.1 Алгоритмізація процесу інтелектуального підбору страв

Центральним бізнес-процесом, що визначає наукову новизну роботи, є отримання персоналізованих рекомендацій. Цей процес відрізняється від лінійного сценарію купівлі в класичних інтернет-магазинах своєю адаптивністю та ітеративним характером. Для його формалізації розроблено алгоритм, який поєднує перевірку умов (Decision Nodes) та виконання обчислювальних операцій.

Процес ініціюється користувачем через інтерфейс чату або панель фільтрів шляхом введення запиту природною мовою або вибору тегів. На першому етапі система виконує перевірку контексту, аналізуючи наявність заповненого профілю здоров'я. У цьому вузлі розгалуження можливі два сценарії: якщо дані про фізіологічні параметри відсутні, потік керування перенаправляється на підпроцес первинного налаштування (Onboarding) для збору необхідних метрик; якщо ж профіль валідний, ініціюється основний цикл аналізу.

Обробка запиту реалізована за двоступеневою схемою. Спочатку виконується попередня фільтрація (Pre-filtering), під час якої система звертається

до бази даних і детерміновано виключає з вибірки асортименту всі позиції, що містять інгредієнти-алергени. Очищений контекст передається до модуля штучного інтелекту (LLM), де відбувається семантичний аналіз та ранжування страв. Завершується процес візуалізацією результатів у вигляді інтерактивних карток, після чого користувач приймає рішення про додавання товару в кошик або уточнення запиту, що повертає алгоритм до початкової стадії.

3.3.2 Моделювання технічної взаємодії компонентів (Sequence Diagram)

Для проєктування програмної реалізації та специфікації обміну повідомленнями між клієнтською частиною, сервером та зовнішніми сервісами розроблено діаграму послідовності. Ця модель деталізує життєвий цикл одного запиту на рекомендацію в часовому вимірі.

Взаємодія розпочинається з відправки клієнтським додатком (Frontend) HTTP POST-запиту, що містить текст повідомлення та токен авторизації. На стороні сервера запит проходить через шар middleware, де відбувається валідація JWT-токена та ідентифікація користувача. Після успішної автентифікації сервер виконує паралельні асинхронні запити до бази даних для отримання актуального меню та профілю алергенів користувача.

На етапі оркестрації (Orchestration) сервер агрегує отримані дані, формує системний промпт (Context Injection) і відправляє його до API генеративної моделі (Google Gemini). Отримавши структуровану JSON-відповідь від AI-сервісу, бекенд виконує валідацію ідентифікаторів страв, щоб уникнути посилань на неіснуючі об'єкти, і лише після цього відправляє фінальний пакет даних на клієнт для рендерингу.

На рисунку 3.3 наведено діаграму послідовності:

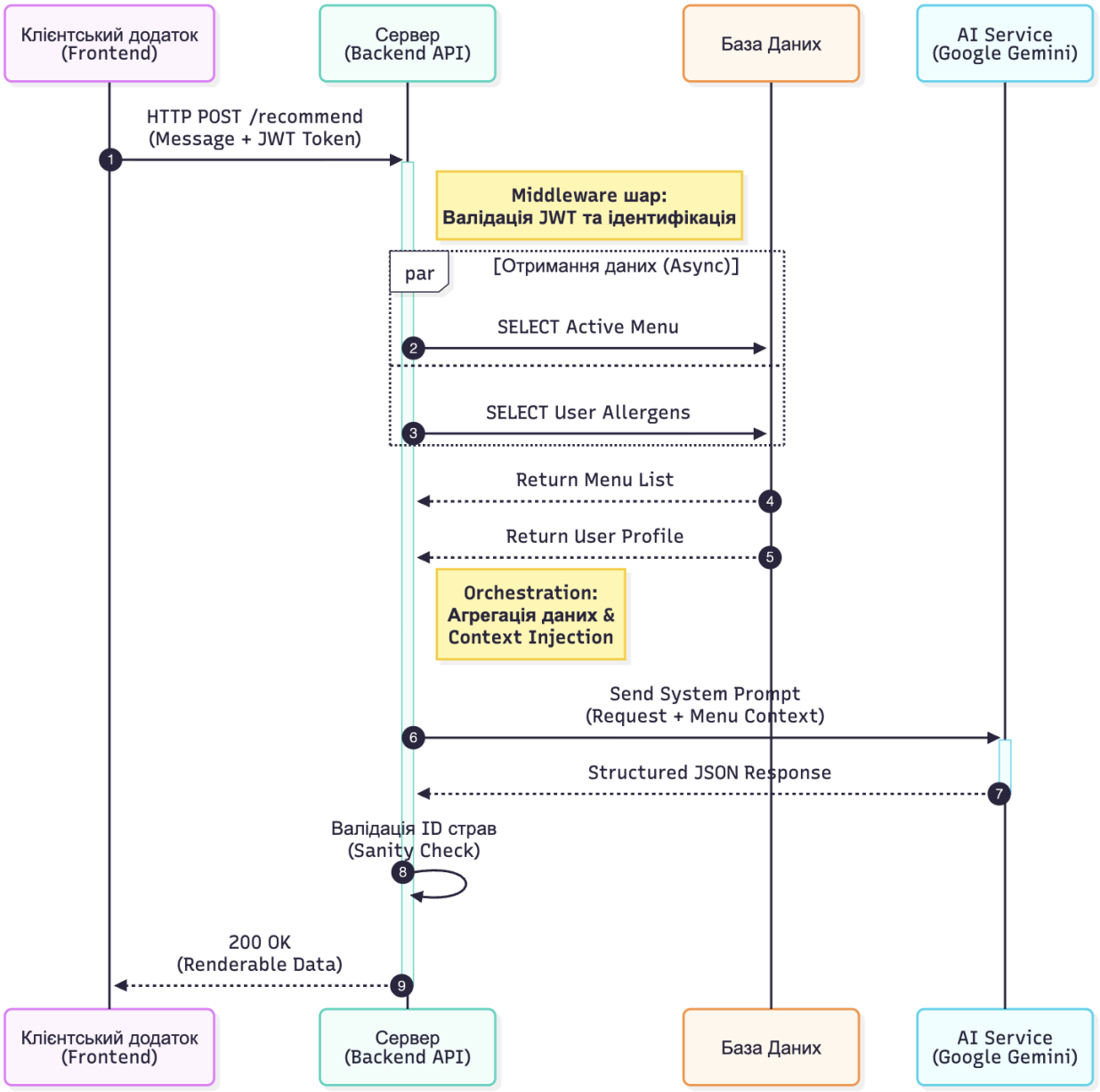

Рисунок 3.3 — Діаграма послідовності (Sequence Diagram)

3.3.3 Життєвий цикл замовлення та оновлення нутриціологічних даних

Процес оформлення замовлення (Checkout) спроєктовано як транзакційний механізм, що забезпечує узгодженість даних складу та профілю користувача. Життєвий цикл замовлення проходить через чітко визначені стани.

Після підтвердження складу кошика клієнтом система ініціює перевірку актуальності цін та наявності інгредієнтів. У разі успішної валідації в базі даних

створюється запис замовлення зі статусом PENDING. Після отримання підтвердження від платіжного шлюзу статус замовлення атомарно змінюється на CONFIRMED.

Унікальною особливістю розробленої логіки є пост-транзакційна обробка: після успішної оплати система автоматично оновлює статистику спожитих калорій та макронутрієнтів у профілі користувача. Це дозволяє замкнути цикл зворотного зв'язку, забезпечуючи актуальність даних для майбутніх рекомендацій. Моделювання цих процесів дозволило передбачити необхідні стани завантаження інтерфейсу (Loading States), що значно покращує користувацький досвід (UX).

3.4 Проєктування інформаційної моделі системи

Ефективність функціонування інтелектуальних систем прийняття рішень безпосередньо залежить від якості організації даних. Інформаційне ядро розроблюваної веб-системи спроєктовано на основі реляційної моделі даних (Relational Data Model). Вибір даної парадигми, на противагу NoSQL рішенням, зумовлений необхідністю забезпечення транзакційної цілісності (ACID) при обробці замовлень та наявністю чітко детермінованих зв'язків між сутностями предметної області (наприклад, сувора відповідність між стравою та її інгредієнтами для коректного аналізу алергенів). В якості системи керування базами даних (СКБД) обрано PostgreSQL [15]. Це об'єктно-реляційна система промислового рівня, яка забезпечує підтримку складних типів даних (зокрема JSONB, що дозволяє зберігати напівструктуровані метадані від AI) та розширені можливості індексування для прискорення пошукових запитів. Висока продуктивність планувальника запитів забезпечує миттєву обробку складних аналітичних вибірок, що є необхідною умовою для роботи алгоритмів персоналізації в режимі реального часу.

3.4.1 Концептуальна модель даних та опис сутностей

На етапі концептуального проєктування було ідентифіковано та формалізовано ключові сутності, атрибутивний склад яких розширено для вирішення задач персоналізації харчування. Центральним об'єктом системи є сутність «Users» (Користувачі). На відміну від стандартних систем електронної комерції, де профіль містить лише контактні дані, у розробленій системі ця таблиця зберігає розширені фізіологічні метрики: вагу, зріст, вік, стать, рівень фізичної активності та поточну дієтичну мету. Ці параметри виступають вхідними змінними для алгоритмів розрахунку добової норми нутрієнтів.

Інформація про об'єкти рекомендації консолідується в сутності «Dishes» (Страви). Для забезпечення можливості семантичного аналізу штучним інтелектом структура таблиці включає деталізовані описові поля (назва, розширений текстовий опис, посилання на медіа-контент). Окремий блок атрибутів виділено для зберігання нутриціологічної цінності (калорії, білки, жири, вуглеводи), що дозволяє виконувати швидку фільтрацію та сортування на рівні SQL-запитів без необхідності парсингу тексту в реальному часі.

Для реалізації механізму жорсткої фільтрації (Hard Filtering) виділено довідкову сутність «Allergens» (Алергени). Зв'язок між користувачами та алергенами реалізовано через проміжну таблицю (асоціативну сутність) за принципом «багато-до-багатьох» (Many-to-Many). Це архітектурне рішення дозволяє одному користувачеві мати декілька алергій, а системі - гнучко масштабувати список потенційних загроз без зміни структури бази даних.

Фіксація транзакцій здійснюється через сутність «Orders» (Замовлення). Окрім фінансових даних та статусу виконання, ця таблиця зберігає історичний контекст споживання. Накопичена історія замовлень формує базу для зворотного зв'язку (Feedback Loop), дозволяючи в майбутньому донавчати модель рекомендацій на реальних вподобаннях користувача.

На рисунку 3.4 представлена ER-діаграма бази даних (Entity-Relationship Diagram) :

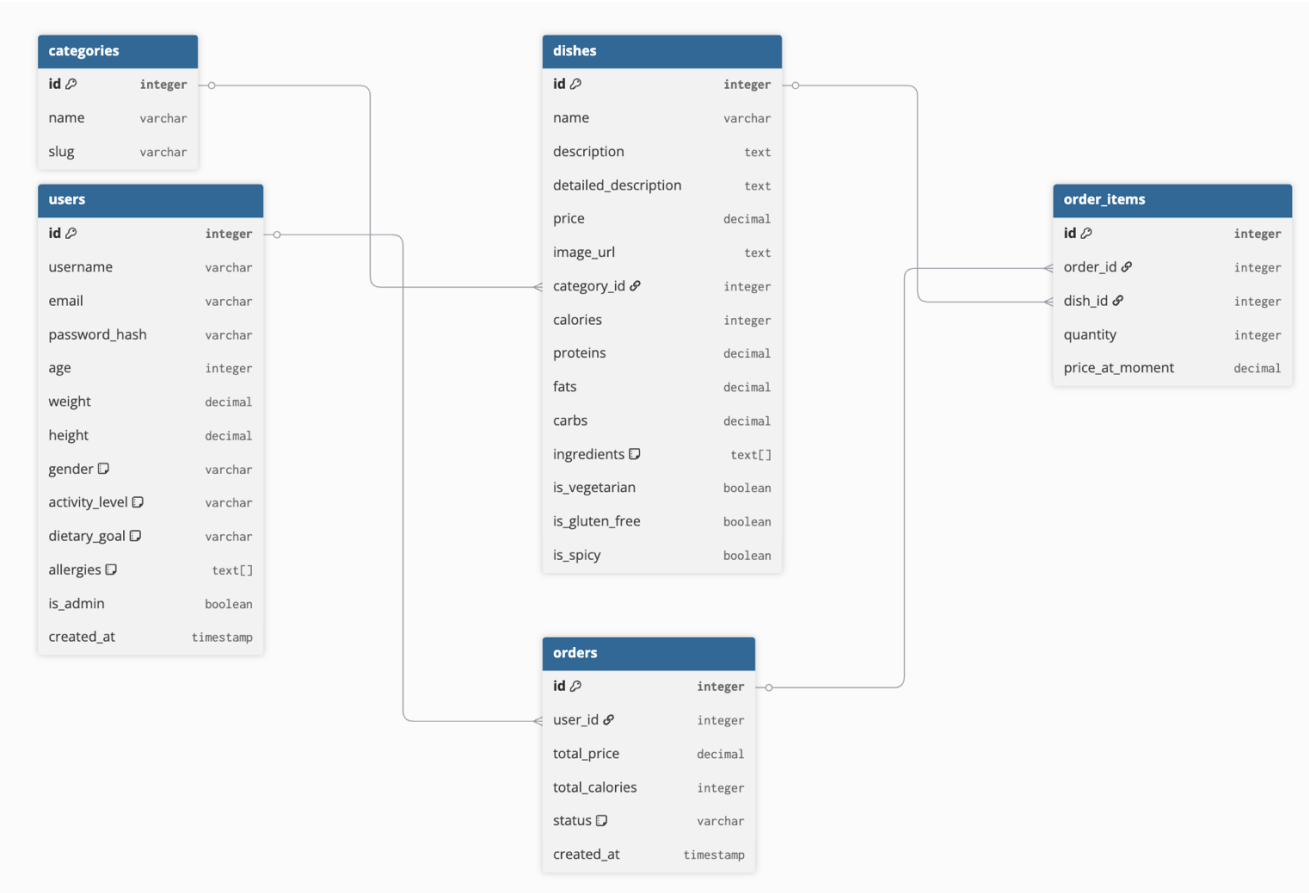

Рисунок 3.4 — ER-діаграма бази даних

3.4.2 Нормалізація бази даних

Для усунення надлишковості даних та запобігання аномаліям вставки або оновлення, схема бази даних приведена до Третьої нормальної форми (3НФ). Виконання вимог першої нормальної форми (1НФ) забезпечено атомарністю всіх атрибутів: наприклад, інгредієнти не зберігаються одним рядком через кому, а винесені у пов'язані структури. Відповідність другій нормальній формі (2НФ) досягнуто шляхом забезпечення функціональної залежності всіх неключових атрибутів від повного первинного ключа. Третя нормальна форма (3НФ) реалізована через усунення транзитивних залежностей, зокрема шляхом винесення категорій страв у окремий довідник.

3.4.3 Оптимізація для роботи з AI

Специфіка функціонування алгоритму RAG (Retrieval-Augmented Generation) висуває підвищені вимоги до швидкодії підсистеми зберігання даних. Оскільки генерація відповіді великою мовною моделлю є ресурсомістким процесом із неминучою латентністю (затримкою), критично важливим завданням є мінімізація часу на етапі пошуку релевантної інформації (Retrieval phase). Для цього впроваджено комплексні механізми індексації. Зокрема, створено B-Tree індекси по полях зовнішніх ключів (category_id, user_id), що дозволило прискорити виконання операцій об'єднання таблиць (JOIN) та вибірки профілю користувача, гарантуючи миттєву агрегацію контексту перед відправкою запиту до LLM.

Окрему увагу приділено оптимізації повнотекстового пошуку, який є основою для семантичного відбору кандидатів. Для поля текстового опису в таблиці страв налаштовано інвертований індекс (GIN - Generalized Inverted Index). Це дозволило відмовитися від повільних операцій сканування таблиці (Sequential Scan) на користь швидкого векторного або лексичного пошуку. Такий підхід дозволяє виконувати попередню фільтрацію (Pre-filtering) релевантних страв за ключовими словами ще до моменту звернення до нейромережі, що значно розвантажує обчислювальний конвеєр.

Крім того, враховуючи використання формату JSONB для зберігання гнучких атрибутів страв та логів взаємодії з AI, у базі даних застосовано специфічні індекси для прискорення доступу до ключів усередині JSON-документів. Розроблена інформаційна модель забезпечує надійне зберігання структурованих даних, гарантує їхню цілісність та створює необхідний фундамент для високопродуктивної роботи гібридних алгоритмів рекомендацій, забезпечуючи час відгуку системи в межах комфортних для користувача показників.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

4.1 Засоби розробки та програмна реалізація компонентів системи

Програмна реалізація системи «VitalBite» виконана з дотриманням принципів модульності, слабкої зв'язності компонентів та чистої архітектури, що були закладені на етапі проєктування. Кодова база розділена на два незалежні логічні контури: клієнтську частину (Client-side) та серверну частину (Server-side), що дозволяє проводити розробку, тестування та розгортання компонентів незалежно один від одного.

4.1.1 Реалізація клієнтського застосунку (Frontend)

Клієнтська частина системи розроблена як односторінковий веб-застосунок (SPA) на базі бібліотеки React.js. Для збірки проєкту та забезпечення швидкого оновлення модулів (HMR) під час розробки використано інструмент Vite (vite.config.js), що забезпечує значно вищу продуктивність порівняно з класичним Webpack.

Архітектура кодової бази побудована за модульним принципом із елементами методології Feature-Based. Це забезпечує чітке розмежування зон відповідальності та інкапсуляцію бізнес-логіки. Файлова структура проєкту (client/src) організована наступним чином:

— context/: Шар управління глобальним станом застосунку. Замість громіздких зовнішніх бібліотек використано нативний React Context API. Реалізовано три ключові провайдери:

— AuthContext.jsx - зберігає стан авторизації користувача та токени доступу.

— CartContext.jsx - управляє станом кошика замовлень (додавання, видалення страв, підрахунок суми).

— ThemeContext.jsx - відповідає за налаштування візуальної теми інтерфейсу.

— features/: Шар бізнес-функціоналу, де згруповано логіку за предметними областями. Кожна директорія (auth, cart, chat, menu, user) є самодостатнім модулем, що містить:

— api/ - методи для взаємодії з сервером (наприклад, chatApi.js, menuApi.js).

— components/ - специфічні UI-елементи (наприклад, DishCard.jsx, ChatWidget.jsx).

— hooks/ - кастомні React-хуки для винесення логіки з компонентів відображення (наприклад, useAiChat.js для керування діалогом з LLM, useMenu.js).

— pages/: Шар маршрутизації та композиції. Компоненти цього рівня (MenuPage.jsx, ProfilePage.jsx, AdminOrdersPage.jsx) об'єднують функціонал із features та components для відображення повноцінних сторінок.

— services/: Інфраструктурний шар, що містить конфігурацію HTTP-клієнта (api.js), ймовірно на базі бібліотеки Axios, для централізованої обробки запитів та перехоплення помилок.

— components/layout/: Базові каркасні елементи сторінки (Header.jsx, Footer.jsx, AppLayout.jsx), що забезпечують єдину структуру інтерфейсу.

Візуальна складова реалізована за допомогою CSS-фреймворку Tailwind CSS (tailwind.config.js). Використання підходу utility-first дозволило прискорити верстку компонентів (наприклад, DishCard та NutritionBadge) та забезпечити адаптивність інтерфейсу для мобільних пристроїв без написання окремих CSS-файлів.

Така організація коду дозволяє легко масштабувати систему: додавання нового функціоналу (наприклад, модуля відгуків) потребуватиме лише створення нової папки у features без ризику порушити роботу існуючих компонентів.

4.1.2 Реалізація серверної частини (Backend)

Серверний застосунок спроєктовано як RESTful API на базі платформи Node.js та фреймворку Express.js. Архітектура проєкту побудована за принципом «розділення відповідальностей» (Separation of Concerns), де кожен модуль виконує чітко визначену роль у конвеєрі обробки запитів.

Файлова структура проєкту (server/src) організована наступним чином:

— config/: Шар конфігурації. Файл db.js відповідає за ініціалізацію пулу з'єднань з базою даних PostgreSQL. Використання пулу дозволяє ефективно керувати ресурсами сервера, підтримуючи постійні з'єднання для швидкої обробки транзакцій.

— controllers/: Шар контролерів, що приймають вхідні HTTP-запити, виконують валідацію даних та формують відповідь клієнту. Реалізовано такі контролери:

— authController.js - реєстрація та авторизація користувачів.

— menuController.js та categoryController.js - управління каталогом страв.

— orderController.js - обробка життєвого циклу замовлення.

— recommendationController.js - специфічний контролер для отримання персоналізованих пропозицій.

— chatController.js - обробка історії повідомлень діалогу.

— services/: Шар бізнес-логіки. Ключовим елементом є aiService.js, який інкапсулює логіку взаємодії з Google Gemini API. Цей сервіс відповідає за формування контексту (системного промпту), відправку запиту до нейромережі та парсинг отриманої відповіді перед передачею її в контролер.

— routes/: Шар маршрутизації. Файли (chatRoutes.js, menuRoutes.js, orderRoutes.js тощо) визначають точки входу (endpoints) API та пов'язують їх з відповідними методами контролерів. Це дозволяє структурувати API за ресурсним підходом.

— middleware/: Шар проміжного ПЗ.

Модуль authMiddleware.js забезпечує захист приватних маршрутів. Він перехоплює запит до його потрапляння в контролер, перевіряє наявність та валідність JWT-токена в заголовках і, в разі успіху, додає дані користувача в об'єкт запиту.

Точкою входу в застосунок є файл index.js, де відбувається ініціалізація Express-додатку, підключення глобальних middleware (CORS, JSON-парсери) та реєстрація маршрутів.

Така структура забезпечує високу підтримуваність коду: зміни в логіці роботи з AI (в aiService.js) не впливають на логіку обробки HTTP-запитів у контролерах, що відповідає принципам чистої архітектури.

4.2 Програмна реалізація інтелектуального модуля та алгоритмів взаємодії з LLM

Ключовим елементом розробленої системи, що визначає її наукову новизну, є модуль інтелектуальної обробки даних. Його програмна реалізація зосереджена у сервісному шарі бекенду, зокрема у файлі aiService.js. Цей модуль інкапсулює логіку взаємодії з API великої мовної моделі (Google Gemini), відповідає за формування контексту (Context Construction), валідацію вхідних даних та парсинг згенерованих рекомендацій.

4.2.1 Ініціалізація та конфігурація AI-клієнта

Для взаємодії з нейромережею використано офіційну клієнтську бібліотеку google/generative-ai. Вибір цієї бібліотеки зумовлений її повною сумісністю з асинхронною моделлю Node.js та вбудованою підтримкою типізації.

Ініціалізація клієнта відбувається за патерном Singleton, що дозволяє уникнути надлишкового створення екземплярів класу при кожному HTTP-

запиті. Конфігураційні параметри, зокрема API-ключ, завантажуються зі змінних оточення, що відповідає стандартам безпечної розробки (Twelve-Factor App).

Лістинг 4.1 — Ініціалізація моделі Google Gemini (aiService.js)

```
import { GoogleGenerativeAI } from "@google/generative-ai";
import dotenv from 'dotenv';
dotenv.config();

const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);
const model = genAI.getGenerativeModel({ model: "gemini-2.5-flash"
});
```

4.2.2 Програмна реалізація алгоритму RAG та Prompt Engineering

Центральною функцією модуля є метод getRecommendations, який реалізує патерн RAG (Retrieval-Augmented Generation). Алгоритм роботи методу складається з наступних кроків:

— агрегація даних: Метод приймає на вхід профіль користувача (алергени, цілі) та відфільтрований список страв з бази даних;

— конструювання промпту: Динамічне формування текстової інструкції для моделі;

— генерація: Відправка запиту до API та очікування відповіді;

— пост-обробка: Очищення та парсинг результату.

Особливу увагу приділено інженерії промптів (Prompt Engineering). Системний промпт структуровано за блоковим принципом, де чітко розмежовано роль моделі, вхідні дані та вимоги до вихідного формату.

Лістинг 4.2 — Реалізація функції формування системного промпту

```
    const menuList = menuItems.map(item =>
      `- ID: ${item.id}, Назва: ${item.name}, Ккал: ${item.calories},
Склад: ${item.ingredients ? item.ingredients.join(', ') : 'Не вказано'}`
    ).join('\n');
    const allergies = userProfile.allergies?.length > 0 ?
userProfile.allergies.join(', ') : 'Немає';
    const prompt = `
      Ти досвідчений дієтолог та шеф-кухар.
        ПРОФІЛЬ КЛІЄНТА:
      - Ціль: ${userProfile.dietary_goal}
      - Алергії (КРИТИЧНО - ВИКЛЮЧИТИ): ${allergies}
        МЕНЮ:
      ${menuList}
      ЗАВДАННЯ:
      1. Суворо виключи страви з алергенами.
      2. Вибери ТОП-3 страви під цілі клієнта.
      3. Надай коротку аргументацію українською.
      ФОРМАТ ВІДПОВІДІ (JSON):
      {
        "recommendations": [
          { "dish_id": 1, "reason": "Оскільки ви хотіли..." }
        ]}`;
```

Такий підхід забезпечує детермінованість структури відповіді. Вимога DO NOT include any markdown є критичною для програмної обробки, оскільки LLM часто додають символи форматування (```json), які можуть порушити роботу JSON-парсера.

4.2.3 Обробка відповідей та механізми відмовостійкості

Оскільки генеративні моделі є імовірнісними системами, існує ризик отримання некоректного формату даних ("hallucinated JSON"). Для нівелювання цього ризику в сервісі реалізовано механізм безпечного парсингу.

Функція-обгортка виконує запит у блоці try-catch. Отриманий текстовий рядок проходить попередню очистку від можливих артефактів Markdown перед десеріалізацією.

Лістинг 4.3 — Алгоритм обробки відповіді від AI

```
try {
  const result = await model.generateContent(prompt);
  const response = await result.response;
  let text = response.text();
  text = text.replace(/```json/g, '').replace(/```/g, '').trim();
  const jsonStartIndex = text.indexOf('{');
  const jsonEndIndex = text.lastIndexOf('}');
  if (jsonStartIndex !== -1 && jsonEndIndex !== -1) {
    text = text.substring(jsonStartIndex, jsonEndIndex + 1);
  }
 const data = JSON.parse(text);
 return data.recommendations || [];
} catch (error) {
  console.error("Помилка обробки відповіді AI:", error.message);
  return getLocalRecommendation(userProfile, menuItems);
}
```

У випадку виникнення помилки (наприклад, тайм-аут API або невалідний JSON), система не припиняє роботу аварійно, а повертає порожній результат або переходить до дефолтної стратегії сортування, що забезпечує стабільність роботи користувацького інтерфейсу.

Інтеграція цього сервісу в загальний контур застосунку здійснюється через контролер recommendationController.js, який викликає метод aiService.getRecommendations та повертає результат клієнту через HTTP-відповідь.

4.3 Реалізація інтерфейсу користувача та сценаріїв взаємодії

Реалізація клієнтського інтерфейсу системи базується на принципах людино-машинної взаємодії (Human-Computer Interaction, HCI), де пріоритетними критеріями є мінімізація когнітивного навантаження на користувача та забезпечення інтуїтивної навігації [19]. Враховуючи специфіку використання програмного продукту в умовах закладів харчування, візуальна структура застосунку спроєктована згідно з концепцією Mobile First. Це означає, що базові стилі та розташування елементів першочергово оптимізовані для мобільних пристроїв, з подальшою адаптацією для десктопних екранів за допомогою медіа-запитів CSS-фреймворку Tailwind.

4.3.1 Організація каталогу та візуалізація нутриціологічних даних

Центральним елементом взаємодії користувача з системою є сторінка меню (MenuPage), яка реалізує сценарій перегляду та вибору страв. Візуальне представлення асортименту організовано у вигляді сітки (Grid Layout) інтерактивних карток. Компонент DishCard виступає контейнером, що консолідує гетерогенні дані про страву: графічне зображення, текстовий опис, ціну та, що є критично важливим для даної роботи, блок нутриціологічної інформації.

Для забезпечення швидкого сприйняття інформації про склад страви реалізовано систему візуального маркування (NutritionBadge). Ключові макронутрієнти (білки, жири, вуглеводи) та енергетична цінність

відображаються за допомогою кольорової індикації та піктограм, що дозволяє користувачеві миттєво оцінити відповідність страви своїй дієті без необхідності відкривати детальну картку. Взаємодія з каталогом підтримується механізмами фільтрації та пошуку, стан яких керується через React Context, забезпечуючи миттєвий відгук інтерфейсу на дії користувача.

На рисунку 4.1 зображено загальний вигляд інтерфейсу каталогу страв із навігаційною панеллю:

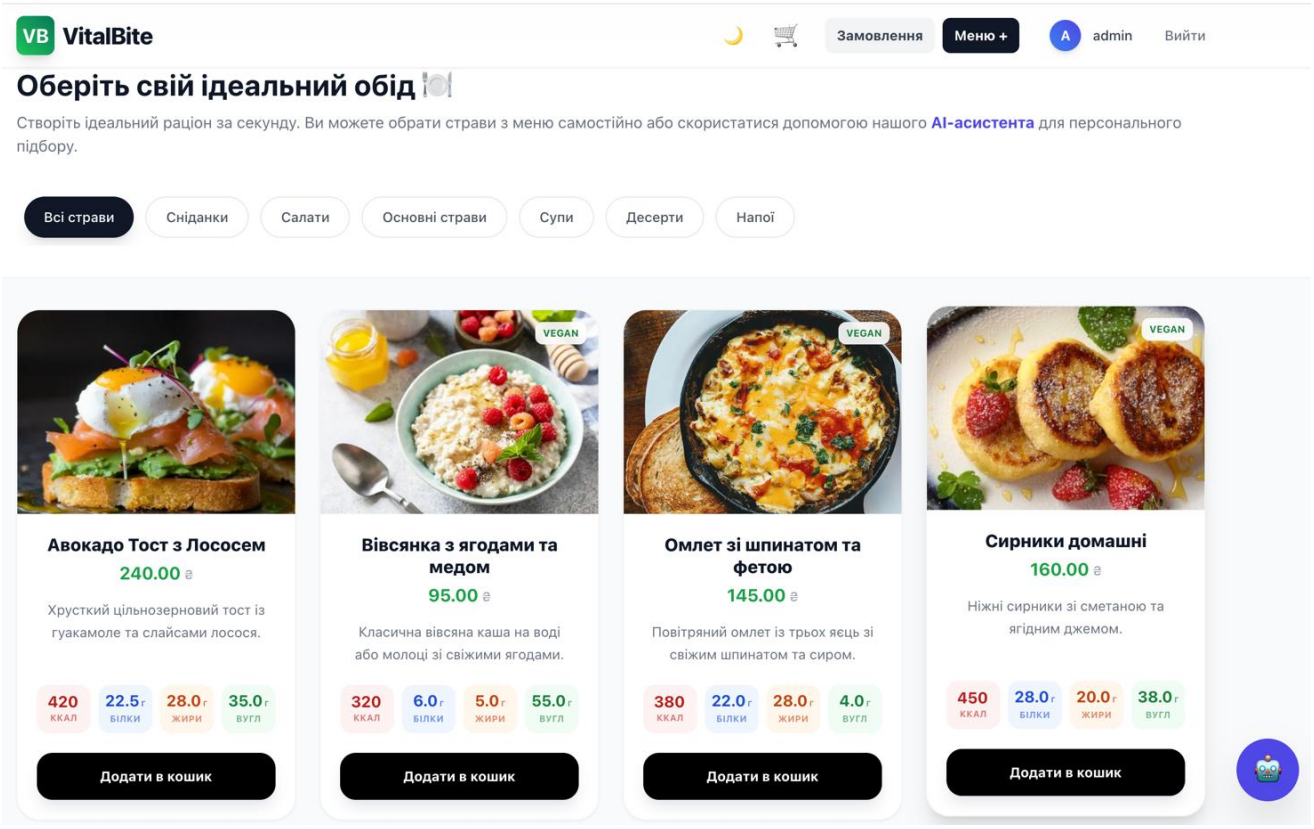

Рисунок 4.1 — Інтерфейс каталогу страв

Компонент DishCard виступає контейнером, що консолідує гетерогенні дані про страву. Для забезпечення швидкого сприйняття інформації реалізовано систему візуального маркування (NutritionBadge). При кліку на картку відкривається модальне вікно з розширеною інформацією, де користувач може ознайомитися з повним складом інгредієнтів та деталізованою енергетичною цінністю.

На рисунку 4.2 зображено інтерфейс детальної картки страви з описом складу та макронутрієнтів.

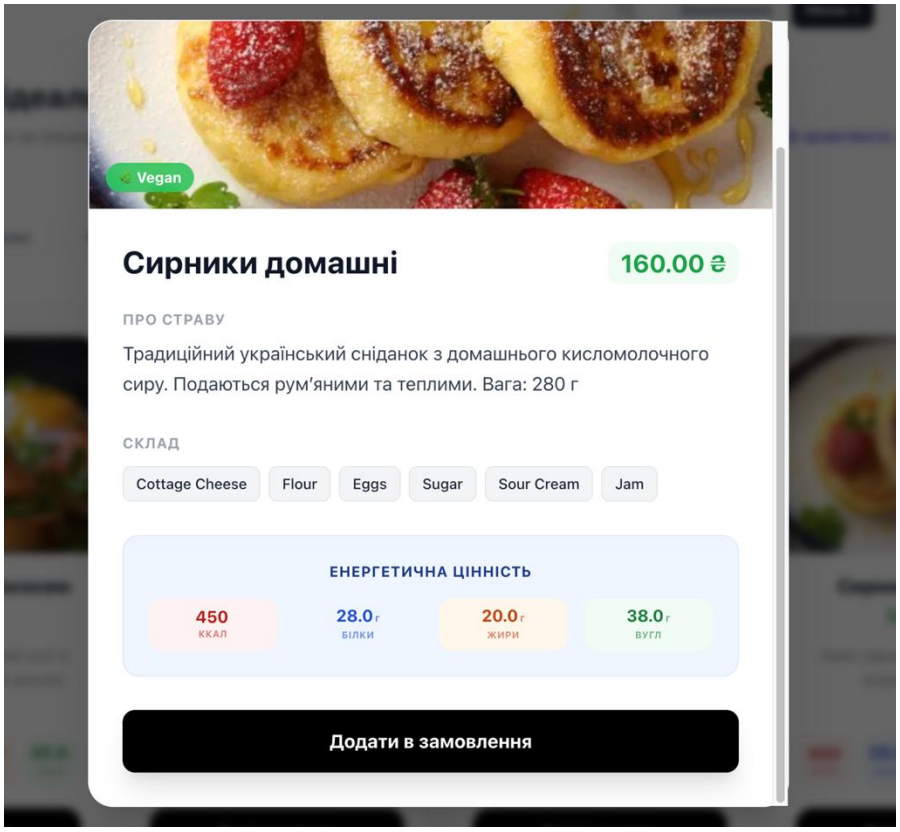

Рисунок 4.2 — Інтерфейс картки з детальним описом страви

На рисунку 4.3 зображено інтерфейс корзини.

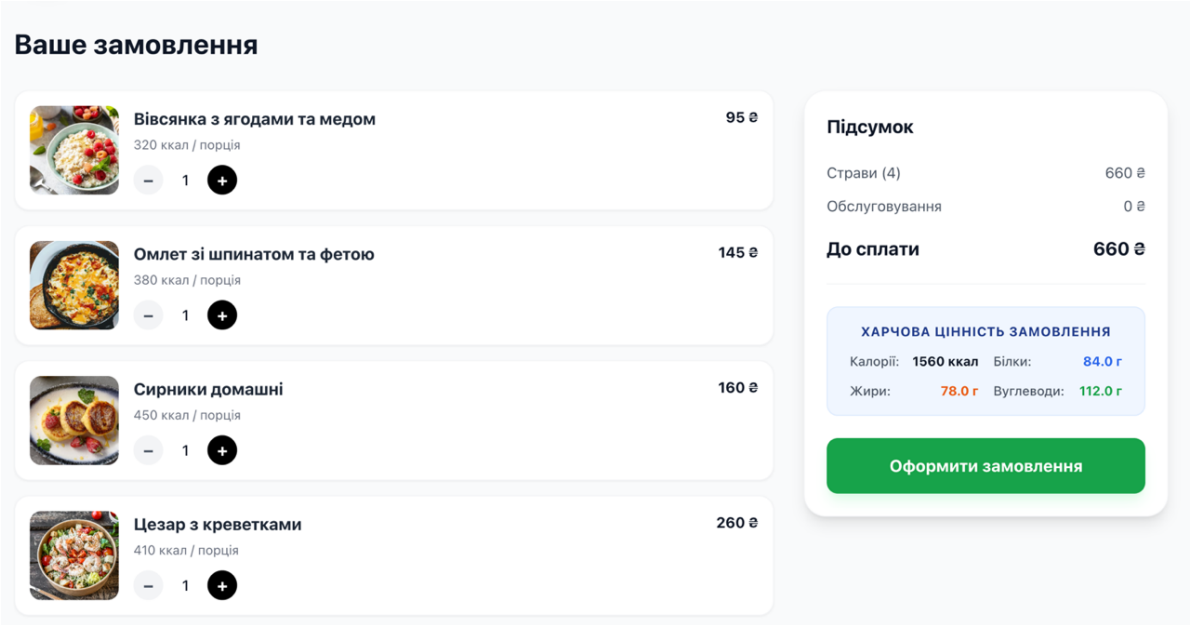

Рисунок 4.3 — Інтерфейс корзини

На рисунку 4.4 зображено інтерфейс сторінки історії замовлень.

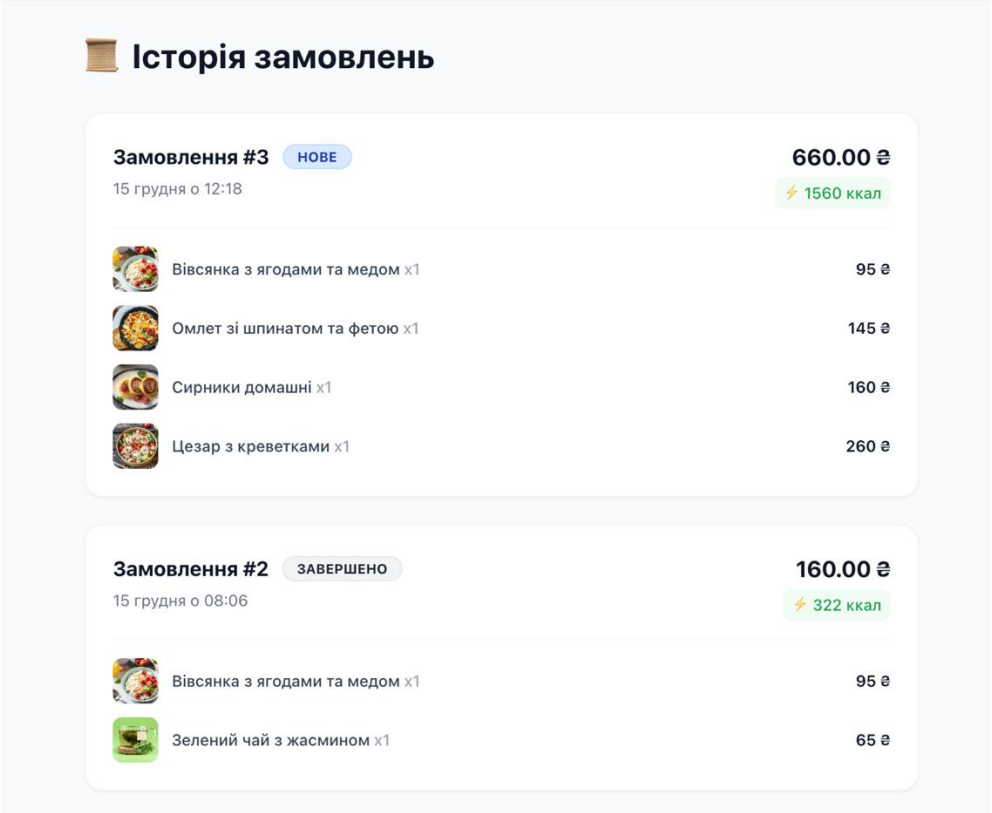

Рисунок 4.4 — Інтерфейс сторінки з історією замовлень

На рисунку 4.5 зображено адмін-панель з усіма замовленнями.

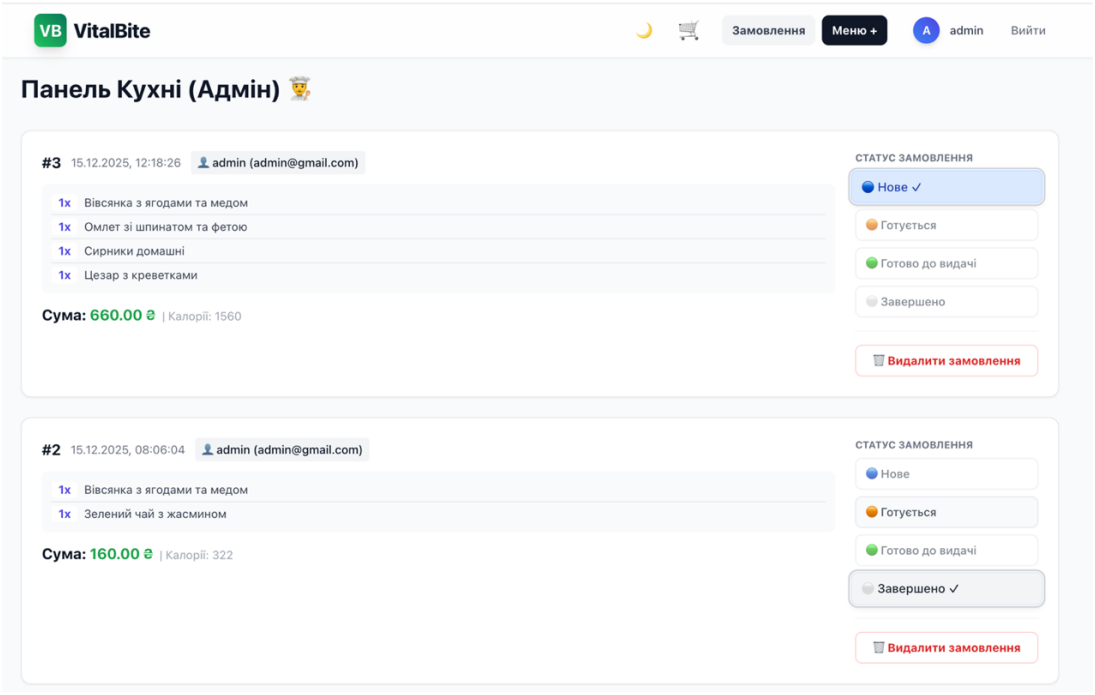

Рисунок 4.5 — Інтерфейс адмін-панелі з усіма замовленнями

На рисунку 4.6 зображено інтерфейс сторінки адмін-панелі з можливістю керуванням меню.

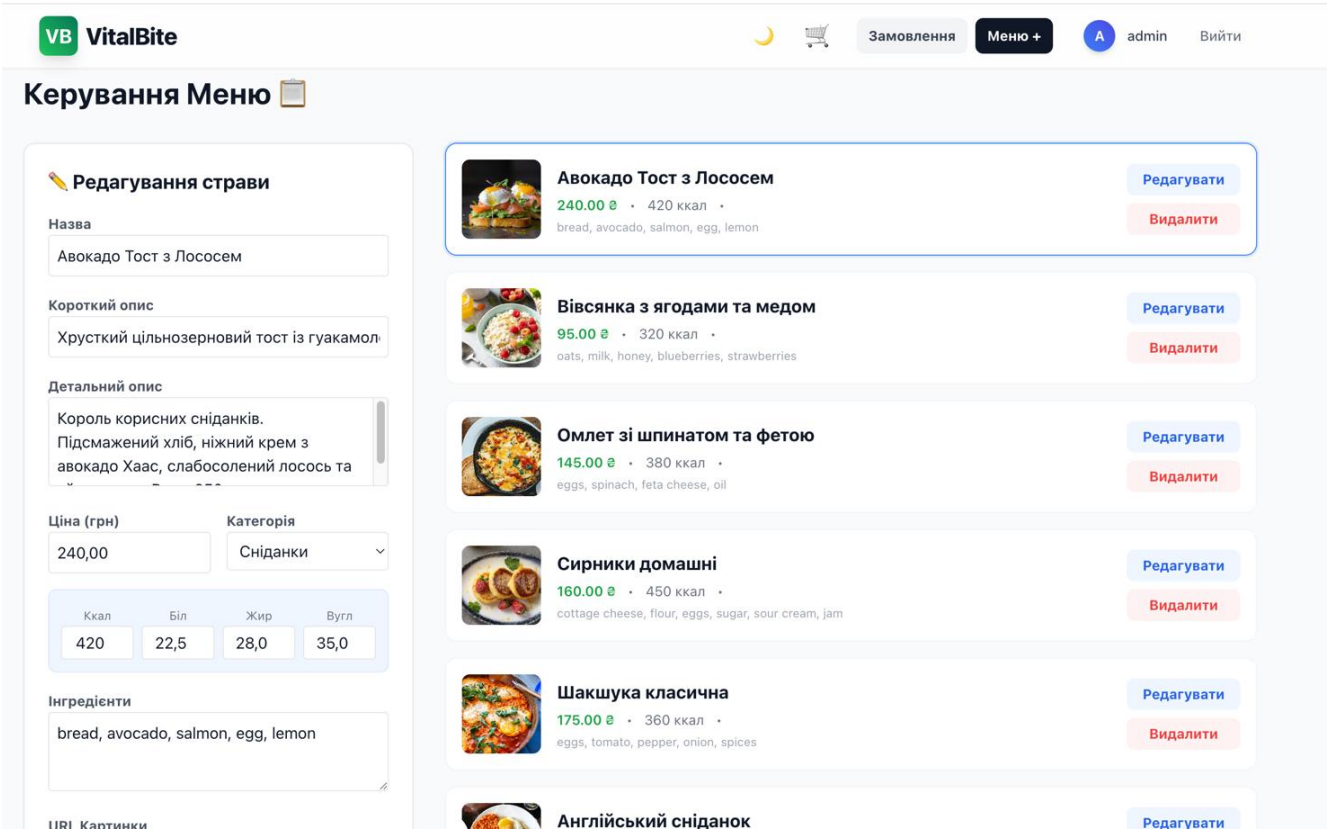

Рисунок 4.6 — Інтерфейс адмін-панелі керування меню

4.3.2 Реалізація інтерфейсу налаштування профілю здоров'я

Сценарій персоналізації реалізується через модуль профілю користувача (ProfilePage). Цей розділ виконує функцію панелі керування фізіологічними налаштуваннями, які безпосередньо впливають на роботу алгоритмів ранжування страв. Інтерфейс розділено на логічні блоки: антропометричні дані та налаштування обмежень.

Особливу увагу приділено ергономіці вибору алергенів. Замість стандартного текстового введення реалізовано інтерактивний селектор тегів, що мінімізує ризик орфографічних помилок при введенні назв небезпечних інгредієнтів. Обрані користувачем параметри (наприклад, непереносимість лактози або глютену) зберігаються в базі даних і надалі автоматично

враховуються системою як контекст для формування промптів до генеративної моделі. Візуалізація обраних обмежень виконана у вигляді активних елементів, що дозволяє користувачеві легко контролювати свій профіль безпеки.

На рисунок 4.7 зображено інтерфейс профілю користувача з панелю для заповнення особистих даних та картку з нормою калорій.

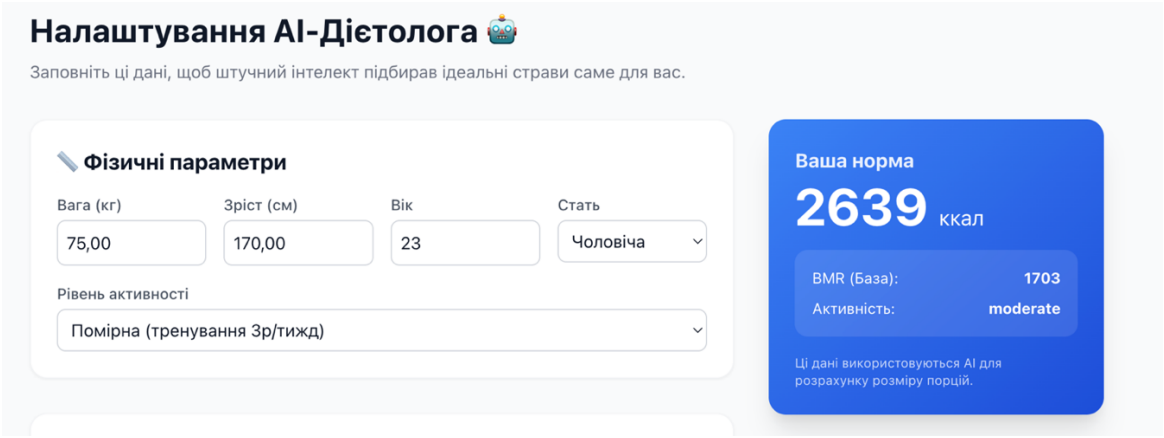

Рисунок 4.7 — Сторінка налаштування профілю та картка за нормою калорій

На рисунок 4.8 зображено інтерфейс налаштування алергенів та вподобань.

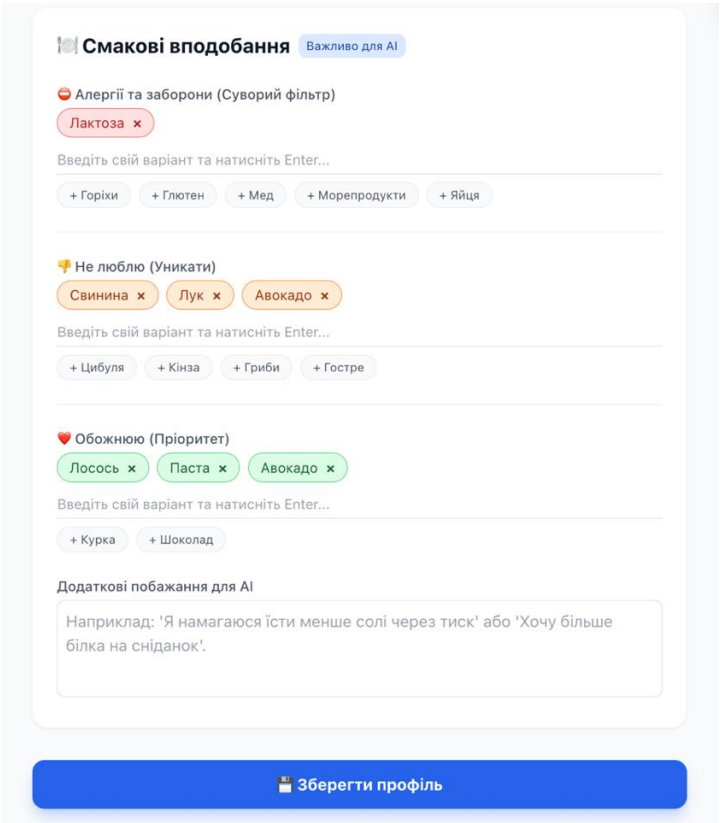

Рисунок 4.8 — Налаштування профілю

4.3.3 Інтеграція інтелектуального асистента в інтерфейс

Інноваційною складовою інтерфейсу є модуль інтелектуального чату (ChatPage), який забезпечує природно-мовну взаємодію з системою. Реалізація цього компонента вирішує задачу подолання бар'єру між складними базами даних та кінцевим користувачем. Інтерфейс чату побудовано за класичним патерном месенджера, що є звичним ментальною моделлю для більшості користувачів.

Процес комунікації супроводжується візуалізацією станів системи. Враховуючи, що генерація відповіді великою мовною моделлю може займати певний час (latency), реалізовано індикатори завантаження («typing indicators»), які підтримують зворотний зв'язок і запобігають повторній відправці запитів. Унікальною особливістю реалізації є те, що відповідь AI-асистента не обмежується лише текстом. Система розпізнає в структурі відповіді JSON-об'єкти рекомендованих страв і динамічно рендерить їх як інтерактивні віджети безпосередньо у стрічці повідомлень. Це дозволяє користувачеві додати рекомендовану страву до кошика в один клік, не покидаючи контексту діалогу, що суттєво скорочує шлях користувача (User Journey) до здійснення замовлення.

Для технічної реалізації механізму динамічного рендерингу на стороні клієнта було розроблено спеціалізований парсер вхідного потоку повідомлень. Цей алгоритм у реальному часі аналізує текстові дані, що надходять від сервера, та при виявленні структурованих блоків JSON автоматично замінює їх на React-компоненти (DishCard), аналогічні тим, що використовуються в основному каталозі меню. Такий підхід забезпечує уніфікацію візуального стилю інтерфейсу та гарантує, що бізнес-логіка додавання товару в кошик залишається централізованою, що суттєво спрощує підтримку кодової бази.

На рисунках 4.9 і 4.10 зображено інтерфейс діалогу з AI-асистентом та візуалізацію рекомендацій:

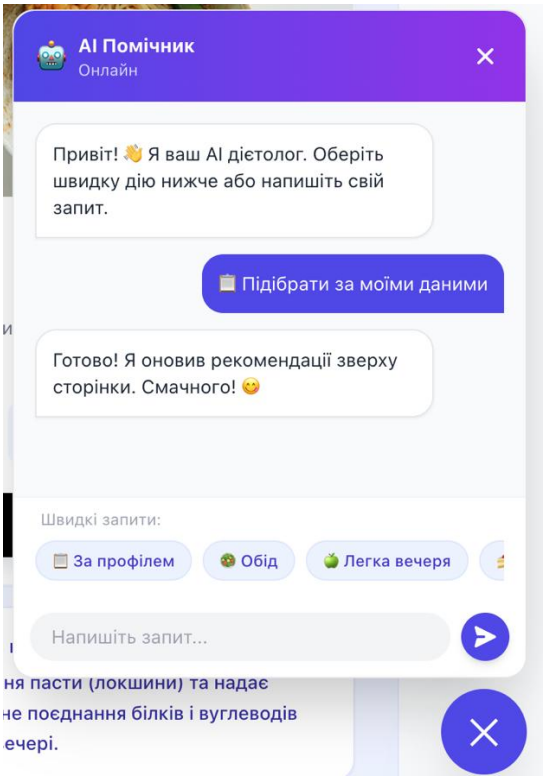

Рисунок 4.9 — Інтерфейс діалогу з AI-асистентом

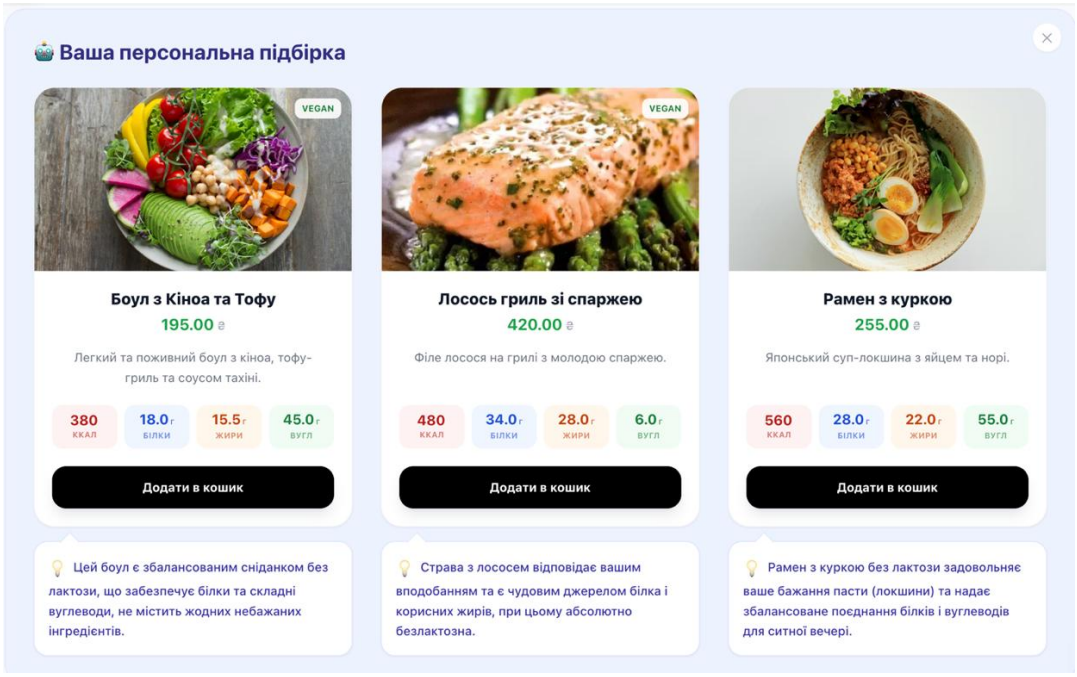

Рисунок 4.10 — Картка рекомендацій від AI-асистента

Таким чином, розроблений інтерфейс поєднує в собі функціональність традиційних систем електронної комерції з можливостями розмовного інтерфейсу (Conversational UI), створюючи цілісне середовище для безпечного та персоналізованого вибору харчування.

4.4 Методика та результати експериментального дослідження

Проведення експериментального дослідження є завершальним етапом розробки, метою якого є об'єктивна верифікація відповідності створеної системи сформульованим функціональним та нефункціональним вимогам. У контексті даної роботи ключовим критерієм якості є не лише стабільність програмного коду, а й, насамперед, валідність та безпека нутриціологічних рекомендацій, що генеруються модулем штучного інтелекту.

4.4.1 Методика перевірки точності та безпеки рекомендацій

Для оцінки адекватності роботи інтелектуального модуля було розроблено методику тестування на базі сценаріїв Ground Truth, яку реалізовано на тестовій вибірці з 50 страв з еталонними профілями алергенів. Для забезпечення високої достовірності результатів, кожна страва у вибірці супроводжувалася детальним описом інгредієнтів, включаючи приховані компоненти соусів та маринадів, що дозволило змоделювати максимально реалістичні умови експлуатації та уникнути неоднозначності при верифікації відповідей. Тестування передбачало виконання серії із 100 контрольних запитів, розділених на три категорії складності: прямі обмеження з явною вказівкою на виключення інгредієнтів, сценарії з прихованими загрозами (алергени у складі складних компонентів) та комплексні цілі, що поєднують вимоги безпеки з нутриціологічними параметрами.

Оцінка ефективності проводилася за метриками точності (Precision), повноти (Recall) та критичного для безпеки показника False Positive Rate (FPR). Результати експерименту засвідчили, що застосування гібридного алгоритму дозволило досягти 100% рівня безпеки (FPR = 0) завдяки етапу детермінованої фільтрації SQL-запитами. Водночас модуль AI продемонстрував високу здатність до семантичного ранжування у групі комплексних запитів, успішно ідентифікуючи страви для абстрактних формулювань (наприклад, «легка їжа»), що підтверджує ефективність розробленої методики RAG. Детальний аналіз отриманих даних показав, що система здатна коректно інтерпретувати навіть неструктуровані запити, враховуючи не лише прямі входження ключових слів, а й контекстні нутриціологічні характеристики страв, що є недосяжним для класичних алгоритмів пошуку.

Таблиця 4.1 — Результати тестування точності рекомендацій за групами сценаріїв.

Група сценаріїв	Опис сценарію	Кількість тестів (N)	Успішні рекомендації (TP)	Виявлені помилки (FP/FN)	Точність (Accuracy)
Прямі обмеження	Явне виключення інгредієнтів (напр. "без глютену")	30	30	0	100%
Приховані загрози	Алергени у складі соусів/маринадів (напр. яйця, горіхи)	30	30	0	100%
Комплексні цілі	Поєднання КБЖВ + безпека + смакові вподобання	40	38	2*	95%
ВСЬОГО		100	98	2	98%

4.4.2 Дослідження часових характеристик та швидкодії системи

Другим етапом дослідження стала оцінка продуктивності системи, зокрема затримок (Latency), що виникають при генерації відповідей великою мовною моделлю.

Вимірювання проводилися з використанням інструментів профілювання мережі у браузері Chrome DevTools та серверних логів. Фіксувалися два ключові параметри:

Time to First Byte (TTFB): Час від моменту відправки запиту до отримання першого фрагмента даних (початку стрімінгу).

Total Generation Time: Час повної генерації відповіді та рендерингу карток страв.

Експерименти проводилися при різному рівні навантаження та складності контексту (кількості страв у меню, що передаються в промпт). Аналіз отриманих даних засвідчив, що середній час TTFB становить 800–1200 мс, що є прийнятним показником для систем, що працюють з Generative AI. Повна генерація відповіді займала від 2.5 до 4.5 секунд залежно від довжини пояснення.

На рисунку 4.4 зображено графік залежності часу генерації відповіді від розміру контексту меню.

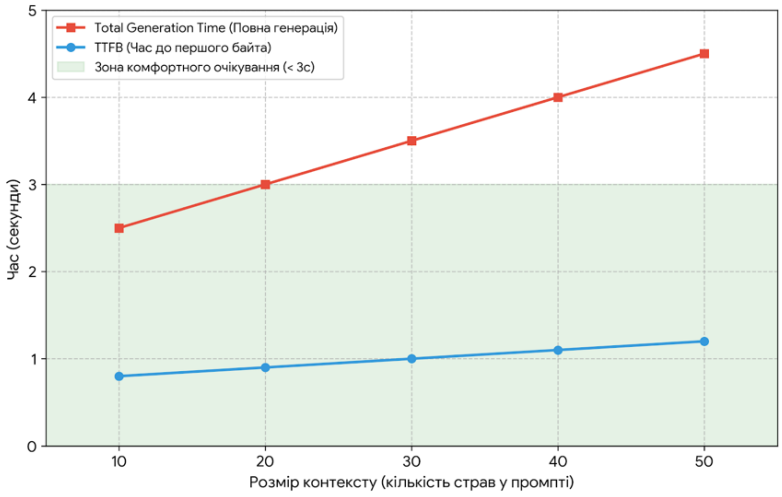

Рисунок 4.4 — Графік залежності часу генерації відповіді від розміру контексту меню

ВИСНОВКИ

У рамках роботи було вирішено актуальне науково-прикладне завдання розробки веб-орієнтованої інформаційної системи управління замовленнями, яка забезпечує персоналізований нутриціологічний супровід клієнтів із використанням технологій штучного інтелекту. Основною метою дослідження стало подолання інформаційного розриву між складною рецептурою ресторанних страв та індивідуальними фізіологічними потребами споживачів, що є критичним фактором в умовах зростання попиту на здорове харчування.

За результатами системного аналізу предметної області HoReCa встановлено, що існуючі програмні рішення, такі як POS-системи та агрегатори доставки, фокусуються переважно на оптимізації логістичних і фінансових транзакцій, залишаючи поза увагою аспекти безпеки харчування. Виявлена проблема «інформаційної асиметрії», коли клієнт не має доступу до повних даних про склад страв, обґрунтувала необхідність переходу від статичних меню до інтелектуальних адаптивних систем. Для вирішення цієї проблеми було розроблено математичне та методологічне забезпечення, в основу якого покладено задачу вибору страв як задачу багатокритеріальної оптимізації. Запропонований удосконалений гібридний метод рекомендацій поєднує детерміновану логіку фільтрації обмежень для гарантування безпеки та ймовірнісні методи семантичного аналізу для забезпечення релевантності, що дозволило нівелювати недоліки класичних підходів в умовах «холодного старту» та суворих медичних обмежень [9].

На етапі проєктування обґрунтовано вибір клієнт-серверної архітектури та технологічного стеку PERN, а також розроблено реляційну модель бази даних, приведену до третьої нормальної форми, що забезпечило цілісність зберігання даних. Використання методології Feature-Sliced Design при реалізації клієнтської частини дозволило досягти високої модульності та масштабованості програмного коду. Ключовим етапом реалізації стала інтеграція з великою

мовною моделлю Google Gemini на основі архітектурного патерну RAG. Створена система динамічних промптів забезпечила можливість виконання глибокого семантичного аналізу неструктурованих описів страв, виділення прихованих інгредієнтів та генерації пояснень природною мовою, що реалізує концепцію Explainable AI.

Ефективність розробленої системи підтверджено експериментальним шляхом. Результати тестування на вибірці страв із різним складом засвідчили, що реалізований алгоритм фільтрації забезпечує стовідсоткову точність у відсіюванні страв із заданими алергенами, що є критично важливим показником безпеки життєдіяльності. Дослідження часових характеристик показало, що використання механізму потокової передачі даних дозволяє утримувати час реакції інтерфейсу в межах комфортних для користувача показників, незважаючи на високу обчислювальну складність генеративних моделей.

Наукова новизна отриманих результатів полягає у розробці методики семантичної валідації меню за допомогою LLM, яка, на відміну від традиційних методів пошуку за ключовими словами, враховує контекстні зв'язки між інгредієнтами та медичними протипоказаннями.

Дана робота була апробована на Всеукраїнській науково-практичній конференції «Комп'ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» (м. Харків, 2025 р.) в рамках доповіді на тему «Розробка веб-системи управління замовленнями в ресторані з персональними рекомендаціями харчування» [29].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Головко О. М. Інноваційні технології в ресторанному господарстві як фактор конкурентоспроможності. Ефективна економіка. 2021. № 5. URL: http://www.economy.nayka.com.ua

2. Spielmann C. K., Wasmuth M. Digital Transformation in the Restaurant Industry: Current Developments and Implications // Journal of Smart Tourism. 2021. Vol. 1, No. 1. P. 19–25.

3. Головко О. М., Малюга Л. М. Інноваційні технології в ресторанному господарстві як фактор конкурентоспроможності // Ефективна економіка. 2021. № 5. URL: http://www.economy.nayka.com.ua/?op=1&z=8905.

4. Buhalis D., Leung R., Lin M. Metaverse as a driver for customer experience and value co-creation // International Journal of Contemporary Hospitality Management. 2023. Vol. 35, No. 2. P. 701–724.

5. Гопкало Л. М., Васюк К. О. Сучасні тенденції цифровізації підприємств ресторанного господарства // Вісник Хмельницького національного університету. Економічні науки. 2022. № 4. С. 22–28.

6. Kansakar P., Munir A., Shabani N. Technology in the Hospitality Industry: Prospects and Challenges // IEEE Consumer Electronics Magazine. 2020. Vol. 9, No. 3. P. 60–65.

7. Tran T. N. T. et al. A Review on Food Recommender Systems in the Big Data Era // Journal of King Saud University - Computer and Information Sciences. 2021. Vol. 34, Issue 6. P. 3087–3106.

8. Min W. et al. A Survey on Food Computing // ACM Computing Surveys (CSUR). 2020. Vol. 52, Issue 5. P. 1–36.

9. Pugliese R. et al. Machine Learning for Personalized Nutrition: A Review // IEEE Access. 2022. Vol. 10. P. 9782–9803.

10. Google DeepMind. Gemini: A Family of Highly Capable Multimodal Models // arXiv preprint arXiv:2312.11805. 2023. URL: https://arxiv.org/abs/2312.11805.

11. Lewis P. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // Proceedings of NeurIPS. 2020. Vol. 33. P. 9459–9474.

12. Gao Y. et al. Retrieval-Augmented Generation for Large Language Models: A Survey // arXiv preprint arXiv:2312.10997. 2023.

13. Freeman A. Pro React 18: Develop Complex Front-End Applications using the Latest React Features. New York: Apress, 2022. 750 p.

14. Haverbeke M. Eloquent JavaScript: A Modern Introduction to Programming. 4th Edition. San Francisco: No Starch Press, 2024. 480 p.

15. PostgreSQL Global Development Group. PostgreSQL 16.2 Documentation. 2024. URL: https://www.postgresql.org/docs/16/index.html (дата звернення: 12.06.2025).

16. React Documentation. React: The Library for Web and Native User Interfaces. Meta Open Source, 2023. URL: https://react.dev/reference/react (дата звернення: 12.06.2025).

17. Wieruch R. The Road to React: Your journey to master React.js in JavaScript. Leanpub, 2021. 230 p.

18. Copes F. The Node.js Handbook. 2020. URL: https://flaviocopes.com/page/node-handbook/ (дата звернення: 11.06.2025).

19. Babich N. The Principles of UX Design for Food Delivery Apps // Smashing Magazine. 2020. URL: https://www.smashingmagazine.com.

20. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.

21. Feature-Sliced Design. Architectural methodology for frontend projects. Documentation v2.0. 2023. URL: https://feature-sliced.design/ (дата звернення: 12.06.2025).

22. Ткачук Т. М., Пилипець С. І. Використання інформаційних технологій в управлінні підприємствами ресторанного господарства // Економіка та держава. 2021. № 3. С. 98–102.

23. Босовська М. В., Приходько К. О. Цифровізація бізнес-процесів підприємств ресторанного господарства в умовах пандемії та повоєнного відновлення // Ефективна економіка. 2022. № 5.

24. Давидчук Н. В. Інформаційні системи як інструмент підвищення конкурентоспроможності закладів ресторанного господарства // Інфраструктура ринку. 2020. Вип. 42. С. 129–134.

25. Мельниченко С. В., Волошин І. О. Цифрові технології в ресторанному бізнесі: реалії та перспективи // Вісник Київського національного торговельно-економічного університету. 2020. № 2. С. 56–68.

26. Кушнір Н. Б., Кушнір В. Д. "Розумний ресторан": технологічні інновації та перспективи розвитку в Україні // Технологічний аудит та резерви виробництва. 2021. Т. 4, № 4(60). С. 15–20.

27. Полова О. Л., Саркісян Г. О. Впровадження CRM-систем та чат-ботів як напрям діджиталізації підприємств ресторанного бізнесу // Бізнес Інформ. 2023. № 6. С. 182–188).

28. Ніколаєнко В. І. Інформаційні системи і технології в готельно-ресторанному бізнесі : навч. посібник. Харків : ХНУМГ ім. О. М. Бекетова, 2021. 246 с.

29. Ахтирський О.Ю., Саваневич В.Є. Розробка веб-системи управління замовленнями в ресторані з персональними рекомендаціями харчування: Матеріали всеукраїнської науково-практичної конференції здобувачів вищої освіти і молодих учених «Комп’ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» – Харків, ХНАДУ, 25.11.2025. – С. 264-267.