Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

*До захисту допущено*
*Кафедрою комп'ютерних систем та робототехніки*
*протокол №__ від ____ червня 2026 р.*

*завідувач кафедри _____ Максим ХРУСЛОВ*
*(підпис)*
*«___»_____ 2026 р.*

**Кваліфікаційна робота**

**здобувача першого (бакалаврського) рівня вищої освіти**

# «ІНФОРМАЦІЙНА СИСТЕМА ОПТИМІЗАЦІЇ РАЦІОНУ ХАРЧУВАННЯ СТУДЕНТА»

Спеціальність 123 – ***Комп'ютерна інженерія***

Освітня програма ***Комп'ютерна інженерія***

Виконавець _____ **Андрій ЛИТВИН**
(підпис)

Науковий керівник _____ **Ольга ЧУБ**
( підпис)

Харків – 2026

# АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи містить вступ, 4 розділи основної частини, висновки, список використаних джерел та 3 додатки. Загальний обсяг роботи складає 63 сторінки, з них основного тексту – 39 сторінок, 9 рисунків, 6 таблиць та 30 найменувань використаних джерел.

Проблема, яку вирішує кваліфікаційна робота: низький рівень автоматизації процесів підбору індивідуального раціону та відсутність спеціалізованих програмних рішень, здатних ефективно розв'язувати задачу оптимізації харчування із врахуванням індивідуального бюджету та фізіологічних норм користувача.

**Об'єктом дослідження** є процеси визначення оптимального складу раціону харчування з урахуванням харчових потреб та фінансових обмежень студента.

**Предметом дослідження** є архітектурні рішення, математичні моделі, алгоритми лінійного програмування та програмні засоби реалізації веб-орієнтованої системи оптимізації раціону харчування.

**Метою кваліфікаційної роботи** є оптимізація індивідуального раціону харчування шляхом розробки та впровадження веб-орієнтованої інформаційної системи «DietOpt» на основі методів лінійного програмування.

Область застосування: розробка інформаційних вебсистем, сфера HealthTech, розробка застосунків для контролю раціону, економії фінансів та персоналізованої дієтології.

***Ключові слова:*** *інформаційні технології, оптимізація раціону, лінійне програмування, PULP, FastAPI, веб застосунок, штучний інтелект.*

# ABSTRACT

The explanatory note to the qualification work contains an introduction, 4 sections of the main body, conclusions, a list of references, and 3 appendices. The total volume of the work is 63 pages, including 39 pages of the main text, 9 figures, 6 tables and 30 referenced sources.

The problem addressed by the qualification work: the low level of automation in individualized diet selection processes and the lack of specialized software solutions capable of effectively solving the nutrition optimization task considering the user's individual budget and physiological norms.

**The object of the study** is the process of determining the optimal dietary composition, taking into account the student's nutritional needs and financial constraints.

**The subject of the study** is the architectural solutions, mathematical models, linear programming algorithms, and software tools for implementing a web-based dietary optimisation system.

**The aim of this thesis** is to optimise individual dietary plans through the development and implementation of the web-based information system 'DietOpt', based on linear programming methods.

Field of application: development of web information systems, the HealthTech sector, and the development of applications for diet control, financial savings, and personalized dietetics.

***Keywords:*** *information technologies, diet optimisation, linear programming, PULP, FastAPI, web application, artificial intelligence.*

# ЗМІСТ

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ** ..... 6
**ВСТУП** ..... 7
**РОЗДІЛ 1. СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МЕТОДІВ ОПТИМІЗАЦІЇ** ..... 10
**1.1 Аналіз проблеми харчування студентів як соціального явища** ..... 10
**1.2 Огляд і порівняльний аналіз існуючих програмних рішень** ..... 12
**1.2.1 Застосунок для контролю харчування MyFitnessPal (США)** ..... 12
**1.2.2 Сервіс детального обліку мікронутрієнтів Cronometer (США)** ..... 12
**1.2.3 Платформа персоналізованих дієт Lifesum (Швеція)** ..... 12
**1.2.4 Платформа обліку калорій з базовим аналізом КБЖУ FatSecret (США)** ..... 13
**1.3 Обґрунтування вибору методу лінійного програмування** ..... 13
**1.4 Обґрунтування вибору технологічного стеку** ..... 15
**1.4.1 Мова програмування Python** ..... 15
**1.4.2 FastAPI як серверний фреймворк** ..... 15
**1.4.3 SQLite як сховище даних** ..... 15
**1.4.4 Google Gemini як генеративна AI-модель** ..... 16
**Висновки до розділу 1** ..... 16
**РОЗДІЛ 2. МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ** ..... 17
**2.1 Формалізація задачі оптимізації раціону харчування** ..... 17
**2.2 Побудова математичної моделі лінійного програмування** ..... 17
**2.3 Математична модель розрахунку норм КБЖУ** ..... 20
**2.4 Проєктування бази даних системи** ..... 22
**2.4.1 Концептуальна модель бази даних системи** ..... 22
**2.4.2 Логічна реляційна модель бази даних** ..... 23
**2.4.3 Фізична реалізація засобами СКБД SQLite** ..... 24
**Висновки до розділу 2** ..... 24
**РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ «DIETOPT»** ..... 25
**3.1 Загальна архітектура системи** ..... 25
**3.2 Розробка серверної частини на FastAPI** ..... 28
**3.2.1 Структура проєкту та ініціалізація** ..... 28

**3.2.2 Pydantic-моделі даних** ..... 29
**3.2.3 Реалізація ендпоінту розрахунку норм КБЖУ** ..... 30
**3.3 Реалізація модуля оптимізації PuLP/CBC** ..... 30
**3.3.1 Завантаження та фільтрація продуктів** ..... 30
**3.3.2 Вибір продуктів за шаблоном** ..... 30
**3.3.3 Розв'язання задачі лінійного програмування** ..... 31
**3.3.4 Стратегія перебору шаблонів** ..... 31
**3.4 Інтеграція хмарного API Google Gemini** ..... 33
**3.5 Розробка клієнтської частини** ..... 33
**3.5.1 Архітектура SPA** ..... 33
**3.5.2 Організація покрокового інтерфейсу** ..... 34
**3.5.3 Стилізація та адаптивність** ..... 37
**Висновки до розділу 3** ..... 37
**РОЗДІЛ 4. ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ СИСТЕМИ** ..... 38
**4.1 Функціональне тестування вебзастосунку** ..... 38
**4.1.1 Методика тестування** ..... 38
**4.1.2 Тестові сценарії** ..... 38
**4.2 Тестування продуктивності алгоритму оптимізації** ..... 39
**4.2.1 Методика вимірювання** ..... 39
**4.2.2 Результати вимірювань** ..... 40
**4.2.3 Масштабованість** ..... 41
**4.3 Аналіз фізіологічної ефективності підібраного раціону** ..... 41
**4.3.1 Методика аналізу** ..... 41
**4.3.2 Відповідність калорійності** ..... 42
**4.3.3 Відповідність нормам макронутрієнтів** ..... 42
**4.3.4 Порівняння оптимізованого та типового раціонів** ..... 42
**Висновки до розділу 4** ..... 44
**ВИСНОВКИ** ..... 45
**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** ..... 47
**ДОДАТКИ** ..... 50

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

**AI** - Artificial Intelligence, штучний інтелект

**API** - Application Programming Interface, прикладний програмний інтерфейс

**BMI** - Body Mass Index, індекс маси тіла

**BMR** - Basal Metabolic Rate, базовий обмін речовин

**CBC** - Coin-or branch and cut, алгоритм-розв'язувач задач цілочисельного та лінійного програмування

**CORS** - Cross-Origin Resource Sharing, механізм спільного використання ресурсів між різними джерелами

**HTTP** - Hypertext Transfer Protocol, протокол передачі гіпертексту

**ІС** - інформаційна система

**КБЖУ** - калорії, білки, жири, вуглеводи

**ЛП** - лінійне програмування

**LLM** - Large Language Model, велика мовна модель

**PAL** - Physical Activity Level, коефіцієнт фізичної активності

**PuLP** - бібліотека Python для задач лінійного та цілочисельного програмування

**REST** - Representational State Transfer, архітектурний стиль побудови розподілених систем

**SQLite** - компактна реляційна система управління базами даних

**TDEE** - Total Daily Energy Expenditure, загальна добова витрата енергії

**ВООЗ** - Всесвітня організація охорони здоров'я

# ВСТУП

Харчування є одним із визначальних чинників збереження здоров'я та працездатності людини. Для студентів вищих навчальних закладів проблема збалансованого харчування набуває особливої гостроти внаслідок поєднання кількох несприятливих факторів: обмеженого бюджету, нерегулярного розпорядку дня, браку кулінарних навичок та недостатньої обізнаності щодо індивідуальних фізіологічних потреб організму. Наслідком незбалансованого харчування є зниження когнітивних функцій, імунодефіцитні стани, розвиток хронічних захворювань шлунково-кишкового тракту та порушення обміну речовин [1].

Дослідження, проведені в Україні та країнах Центральної Європи, свідчать, що понад 60 % студентів харчуються нерегулярно, а більше ніж 45 % не виконують добову норму споживання білків і мікронутрієнтів [2]. Водночас витрати на харчування становлять від 30 до 50 % студентського бюджету, що ставить задачу мінімізації вартості раціону при дотриманні фізіологічних норм у розряд соціально значущих прикладних задач.

Наявні комерційні застосунки для контролю харчування (MyFitnessPal, Cronometer, Lifesum) не вирішують задачу оптимізації вартості раціону: вони надають інструменти обліку, але не генерують рекомендації щодо мінімально витратного збалансованого меню. Крім того, зазначені рішення не адаптовані до реалій українського продовольчого ринку, не враховують локальні цінові дані на продукти та не надають рецептурних рекомендацій.

Задача складання оптимального раціону харчування може бути формалізована як задача лінійного програмування (ЛП): мінімізація цільової функції (сумарна вартість раціону) за умов виконання системи лінійних обмежень (норми КБЖУ, бюджет, структура прийомів їжі). Метод ЛП добре досліджений у теоретичному і практичному аспектах, характеризується детермінованістю результату та поліноміальною обчислювальною складністю

для задач із безперервними змінними, що робить його придатним для застосування у реальних системах підтримки прийняття рішень.

Із зазначеного випливає актуальність розробки спеціалізованої інформаційної системи, яка б: автоматично розраховувала індивідуальні норми споживання КБЖУ на основі антропометричних параметрів; мінімізувала вартість добового раціону при дотриманні норм; формувала структуровані рекомендації з приготування страв; надавала прозорий, зручний вебінтерфейс.

Метою кваліфікаційної роботи є проєктування та програмна реалізація веборієнтованої інформаційної системи «DietOpt» для автоматизованої оптимізації індивідуального раціону харчування на основі методів лінійного програмування.

Для досягнення поставленої мети сформульовано такі завдання:

1. Провести системний аналіз предметної області та методів алгоритмічної оптимізації раціонів харчування.
2. Сформалізувати задачу оптимізації раціону та побудувати математичну модель лінійного програмування.
3. Спроєктувати архітектуру веборієнтованої інформаційної системи з визначенням взаємодії frontend, backend та зовнішніх API.
4. Розробити концептуальну, логічну та фізичну моделі бази даних продуктів харчування.
5. Реалізувати серверну частину системи із впровадженням REST API та інтеграцією з модулем оптимізації PuLP/CBC.
6. Розробити клієнтську частину вебзастосунку та забезпечити її взаємодію із серверною частиною системи.
7. Реалізувати інтеграцію з API великої мовної моделі Google Gemini для генерації персоналізованих рецептів.
8. Провести тестування вебзастосунку, оцінити ефективність, продуктивність та коректність результатів оптимізації.

Об’єктом дослідження є процеси проєктування, реалізації та функціонування веборієнтованих інформаційних систем підтримки прийняття рішень на основі математичних методів оптимізації.

Предметом дослідження є архітектурні рішення, математичні моделі лінійного програмування, алгоритми оптимізації та програмні засоби реалізації веборієнтованої системи автоматизованого формування збалансованого раціону харчування.

У роботі застосовано такі методи дослідження: системний аналіз (при дослідженні предметної області та аналогів), математичне моделювання та лінійне програмування (при побудові моделі оптимізації), об'єктно-орієнтоване програмування (при реалізації системи), метод порівняльного аналізу (при обґрунтуванні вибору технологій).

Практичне значення отриманих результатів. Розроблена інформаційна система «DietOpt» може використовуватися студентами будь-якого вищого навчального закладу для складання індивідуального збалансованого раціону харчування з мінімальними витратами. Система є масштабованою та може бути розширена для інших цільових груп споживачів.

# РОЗДІЛ 1.
# СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МЕТОДІВ ОПТИМІЗАЦІЇ

## 1.1 Аналіз проблеми харчування студентів як соціального явища

Харчування студентів є об'єктом активних досліджень у галузях медицини, соціальної психології та інформаційних технологій. Комплексний аналіз наукових джерел дозволяє виділити три ключові аспекти проблеми: медико-фізіологічний, соціально-економічний та інформаційний.

Медико-фізіологічний аспект. Організм студента, зазвичай у віці 17–25 років, перебуває у фазі завершення росту та формування м'язово-скелетного апарату. Добова потреба в енергії для молодих людей із переважно сидячим способом навчання становить 2000–2500 ккал для жінок і 2500–3200 ккал для чоловіків відповідно до настанов Всесвітньої організації охорони здоров'я [3]. Потреба в білку, як головному будівельному нутрієнті, складає 0.8–1.2 г на кілограм маси тіла на добу. Дефіцит макронутрієнтів, вітамінів групи В та заліза безпосередньо знижує концентрацію уваги, що підтверджено у клінічних дослідженнях [4].

Соціально-економічний аспект. Відповідно до даних Державної служби статистики України, середньомісячний дохід студентської сім'ї у 2024–2025 навчальному році не перевищував 8 000–12 000 гривень на одну особу. При цьому витрати на харчування в структурі студентських витрат займають 30–50 %. Ринкові ціни на основні продукти харчування підвищились за 2022–2024 рр. на 35–60 %, що суттєво скоротило реальну купівельну спроможність. В умовах, що склалися, пошук способів зниження витрат на харчування без шкоди для здоров'я є актуальним практичним завданням.

Інформаційний аспект. Студенти, як правило, не мають достатньої підготовки у сфері нутриціології та не здатні самостійно розрахувати індивідуальні норми споживання поживних речовин. Відсутність об'єктивної інформації спонукає до хаотичного харчування, надмірного споживання

вуглеводів із дешевих джерел (хлібобулочні вироби, крупи) при дефіциті білка та здорових жирів. Водночас студенти є активними користувачами цифрових технологій, що створює передумови для ефективного впровадження вебзастосунків-асистентів у сфері харчування.

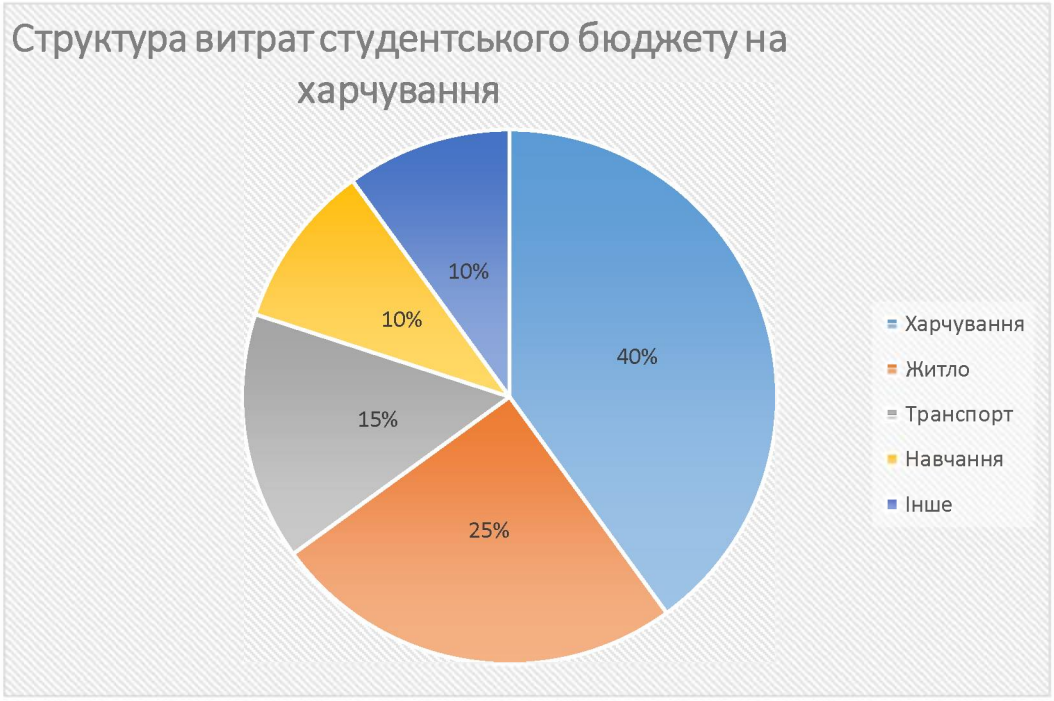



Рисунок 1.1 — Структура витрат студентського бюджету на харчування.

Задача оптимізації харчування формалізується таким чином: при заданих антропометричних показниках (стать, вік, маса, зріст, рівень фізичної активності), меті (схуднення, підтримка, набір маси) та бюджетному обмеженні необхідно сформувати перелік продуктів з визначеними порціями, що мінімізує загальну вартість раціону при дотриманні обмежень на добову калорійність, вміст білків, жирів та вуглеводів.

Дана задача відноситься до класу задач лінійного програмування (ЛП) — оптимізації лінійної цільової функції при системі лінійних обмежень — і є різновидом так званої «дієтичної задачі» (Diet Problem, або задачі Стіглера), уперше сформульованої у 1945 р. економістом Дж. Стіглером [5]. Підхід залишається актуальним і в сучасних застосунках завдяки своїй детермінованості та швидкодії.

## 1.2 Огляд і порівняльний аналіз існуючих програмних рішень

На ринку мобільних і вебзастосунків для управління харчуванням представлено кілька провідних платформ. Нижче наведено порівняльний аналіз чотирьох найбільш поширених рішень.

### 1.2.1 Застосунок для контролю харчування MyFitnessPal (США)

MyFitnessPal - один із найпопулярніших застосунків для відстеження калорій і харчування, розроблений компанією Under Armour. Станом на 2024 р. база даних продуктів налічує понад 14 мільйонів позицій. Застосунок надає можливість ведення харчового щоденника, сканування штрих-кодів продуктів, відображення динаміки маси тіла та інтеграцію з фітнес-трекерами [6].

Переваги: обширна база продуктів; інтеграція з IoT-пристроями; зручний інтерфейс. Недоліки: відсутній модуль оптимізації вартості раціону; платна версія для розширеного функціоналу; відсутність адаптації до українського ринку; жодного механізму генерації рецептів на основі наявних продуктів.

### 1.2.2 Сервіс детального обліку мікронутрієнтів Cronometer (США)

Cronometer позиціонується як застосунок для точного відстеження мікронутрієнтів. Використовує верифіковані дані USDA (Міністерство сільського господарства США) та NCCDB. Надає детальну розбивку вмісту вітамінів, мінералів та амінокислот [7].

Переваги: висока точність даних; детальна мікронутрієнтна аналітика; підтримка кетогенної та веганської дієт. Недоліки: складний для рядового користувача інтерфейс; відсутня функція оптимізації вартості; орієнтований на американські продукти.

### 1.2.3 Платформа персоналізованих дієт Lifesum (Швеція)

Lifesum - скандинавський застосунок, що поєднує відстеження харчування з програмами здорового способу життя. Виокремлює різні «плани харчування» (Кето, Середземноморська, Скандинавська дієти тощо). Надає рецептурні рекомендації, однак без прив'язки до наявних продуктів або бюджетних обмежень [8].

### 1.2.4 Платформа обліку калорій з базовим аналізом КБЖУ FatSecret (США)

FatSecret — безкоштовна платформа для відстеження калорій із порівняно великою базою даних. Має функцію «Щоденник харчування» та базовий аналіз КБЖУ. Відрізняється відносно невисокою якістю верифікації даних користувацьких записів.

Узагальнюючи проведений аналіз (див. табл. 1.1), можна констатувати: жоден з розглянутих застосунків не вирішує задачу мінімізації вартості раціону при заданих фізіологічних обмеженнях, не інтегрується з українськими цінами на продукти та не генерує рецептів на основі оптимально підібраного переліку продуктів. Зазначені прогалини обумовлюють доцільність розробки спеціалізованої системи «DietOpt».

*Таблиця 1.1*

**Порівняльний аналіз існуючих застосунків для управління харчуванням**

| Критерій | MyFitnessPal | Cronometer | Lifesum | FatSecret | DietOpt |
|---|---|---|---|---|---|
| Оптимізація вартості | Ні | Ні | Ні | Ні | Так |
| Розрахунок КБЖУ | Так | Так | Так | Так | Так |
| Генерація рецептів AI | Ні | Ні | Частково | Ні | Так |
| Підтримка укр. ринку | Ні | Ні | Ні | Ні | Так |
| Безкоштовність | Частково | Частково | Частково | Так | Так |

## 1.3 Обґрунтування вибору методу лінійного програмування

Задача оптимізації раціону харчування за критерієм мінімуму вартості при обмеженнях на харчову цінність відноситься до класу задач лінійного програмування (ЛП). Лінійне програмування — розділ математичної оптимізації, що вивчає методи розв'язання задач з лінійними цільовою функцією та обмеженнями [9].

Серед альтернативних підходів до розв'язання задачі розглядалися такі методи:

– Генетичні алгоритми (ГА). Вони здатні знаходити наближені розв'язки для оптимізації нелінійних об'єктів. Однак, згідно з дослідженнями у сфері машинного навчання, ГА не гарантують глобального оптимуму, потребують значного налаштування гіперпараметрів і демонструють гірші показники відтворюваності [23].

– Штучні нейронні мережі. Незважаючи на здатність до апроксимації складних функцій і потужні прогностичні властивості, архітектури глибокого навчання працюють за принципом "чорної скриньки". Вони суттєво програють детермінованим математичним алгоритмам у задачах з жорстко заданими фізіологічними (медичними) межами (constraints) та не здатні стовідсотково дотримуватись правил без похибок і галюцинацій [22].

– Лінійне програмування (ЛП) з неперервними змінними. Завдяки формалізму ЛП знаходження розв'язку (суворо глобального оптимуму за умови збіжності) виконується за гарантовано передбачуваний поліноміальний час без необхідності попереднього навчання на датасетах, що класично використовувалося ще за методологією Дж. Данціга [21]. Це дозволяє будувати медично точні обмеження у застосунках.

Проведено порівняльний аналіз програмних засобів для імплементації моделі ЛП на мові Python: бібліотеки PuLP, інструментарію Google OR-Tools та Gurobi. Було прийнято рішення обрати спеціалізовану бібліотеку PuLP [20] та розв'язувач CBC (COIN-OR) [10], оскільки такий підхід: не вимагає платної ліцензії, не залежить від комерційних серверів Google і підтримує створення матриці рівнянь у лаконічному вигляді Python, чого вимагають специфікації даної системи [11]. Наявна обчислювальна задача реалізовується на базі неперервних (continuous) змінних (відносні частки стограмових порцій продуктів), тому симплекс-метод повністю забезпечує оптимум цільової функції за критеріями, які досліджував ще С. Руз [15].

## 1.4 Обґрунтування вибору технологічного стеку

### 1.4.1 Мова програмування Python

Python обрано основною мовою серверної розробки з огляду на такі переваги: розвинена екосистема бібліотек науково-технічних обчислень (NumPy, SciPy, PuLP); зрозумілий синтаксис, що сприяє підтримуваності коду; активна підтримка бібліотек машинного навчання та взаємодії з хмарними API; широке використання у наукових та інженерних застосунках [11].

Версія Python 3.11+ забезпечує покращену продуктивність відносно попередніх версій завдяки оптимізованому інтерпретатору CPython та вдосконаленим механізмам обробки асинхронних запитів.

### 1.4.2 FastAPI як серверний фреймворк

FastAPI — сучасний високопродуктивний фреймворк для побудови REST API на Python, розроблений Себастіаном Рамірезом у 2018 р. Побудований на базі Starlette (асинхронний HTTP-сервер) та Pydantic (валідація даних на основі анотацій типів Python) [12].

Ключові переваги FastAPI для даного проєкту: автоматична генерація документації OpenAPI (Swagger UI); вбудована валідація вхідних даних через Pydantic-моделі; асинхронна обробка запитів (async/await) без блокування I/O; висока продуктивність — за незалежними бенчмарками порівнянна з Node.js та Go [12].

Порівняно з альтернативними фреймворками Django та Flask, FastAPI є оптимальним вибором для мікросервісної архітектури з явно визначеними API-ендпоінтами, що відповідає структурі системи «DietOpt».

### 1.4.3 SQLite як сховище даних

SQLite обрано як СКБД системи «DietOpt» на основі таких аргументів: відсутність необхідності у конфігуруванні окремого сервера бази даних (serverless-архітектура); зберігання всієї бази у єдиному файлі (food_tracker.db), що спрощує розгортання та перенесення; повна підтримка стандарту SQL; висока швидкість читання для задач із переважанням SELECT-запитів. За

результатами внутрішнього тестування система опрацьовує до 334 продуктів у базі за 5–15 мс на запит [13].

### 1.4.4 Google Gemini як генеративна AI-модель

Для генерації персоналізованих рецептів на основі оптимізованого переліку продуктів обрано Google Gemini 1.5 Flash — мультимодальну велику мовну модель компанії Google DeepMind. Модель характеризується такими параметрами: безкоштовна квота 15 запитів/хв та 1500 запитів/день; підтримка контекстного вікна понад 1 млн токенів; висока якість генерації тексту українською мовою; низька затримка відповіді — 2–10 сек [14].

## Висновки до розділу 1

За результатами системного аналізу встановлено:

1. Проблема незбалансованого харчування студентів є соціально значущою та обумовленою поєднанням медичних, економічних та інформаційних факторів.
2. Існуючі комерційні застосунки не вирішують задачу мінімізації вартості раціону з урахуванням фізіологічних обмежень і не адаптовані до українського ринку.
3. Задача оптимізації раціону формалізується як задача ЛП, що гарантує знаходження глобально оптимального розв'язку при прозорій математичній постановці.
4. Технологічний стек Python / FastAPI / PuLP-CBC / SQLite / Google Gemini є обґрунтованим вибором для реалізації системи «DietOpt» за критеріями продуктивності, відкритості коду та простоти розгортання.

# РОЗДІЛ 2.
# МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

## 2.1 Формалізація задачі оптимізації раціону харчування

Постановка задачі. Нехай задано множину продуктів харчування $P = \{p_1, p_2, ..., p_n\}$, де n — кількість продуктів у базі даних. Кожен продукт $p_i$ характеризується вектором показників:

$c_i$ — вартість 100 г продукту (грн);

$кал_i$ — калорійність 100 г продукту (ккал);

$біл_i$ — вміст білків у 100 г продукту (г);

$жир_i$ — вміст жирів у 100 г продукту (г);

$вуг_i$ — вміст вуглеводів у 100 г продукту (г).

Змінні рішення: $x_i$ — кількість 100-грамових порцій продукту $p_i$, що входить до добового раціону. Змінні є дійсними невід'ємними числами, обмеженими знизу мінімальною порцією ($l_i$) та зверху максимальною ($u_i$) з урахуванням харчових шаблонів системи.

Цільова функція: мінімізувати загальну вартість добового раціону:

$$\mathbf{Z = \Sigma_i\, c_i \cdot x_i \rightarrow min,\ i = 1, ..., n}$$

Система обмежень включає обмеження за калорійністю, білками, жирами, вуглеводами, бюджетом та структурою прийомів їжі, що детально описано у підрозділі 2.2.

## 2.2 Побудова математичної моделі лінійного програмування

Згідно з класичними принципами моделі «дієтичної задачі» Стіглера [5] та базового лінійного програмування [21], повна математична модель задачі оптимізації раціону в системі «DietOpt» записується таким чином:

Цільова функція (мінімізація вартості):

$$min\ Z = \sum_{i=1}^{n} c_i \cdot x_i$$

де $c_i$ — ціна 100 г і-го продукту у гривнях, $x_i$ — кількість 100-грамових одиниць і-го продукту в раціоні.

Обмеження 1. Добова калорійність раціону повинна знаходитися в допустимому діапазоні навколо цільового значення T_кал:

$$T_кал \cdot (1 - \delta_кал) \le \Sigma_i кал_i \cdot x_i \le T_кал \cdot (1 + \delta_кал)$$

де δ_кал = 0.10 — допустиме відхилення за калорійністю (±10 %).

Обмеження 2–3. Аналогічно для вмісту білків, жирів та вуглеводів з допустимим відхиленням δ_мак = 0.20 (±20 %) від нормативних меж [П_min, П_max], [Ж_min, Ж_max], [В_min, В_max]:

$$П_min \cdot (1 - \delta_мак) \le \Sigma_i біл_i \cdot x_i \le П_max \cdot (1 + \delta_мак)$$

$$Ж_min \cdot (1 - \delta_мак) \le \Sigma_i жир_i \cdot x_i \le Ж_max \cdot (1 + \delta_мак)$$

$$В_min \cdot (1 - \delta_мак) \le \Sigma_i вуг_i \cdot x_i \le В_max \cdot (1 + \delta_мак)$$

Обмеження 4. Бюджетне обмеження:

$$\Sigma_i c_i \cdot x_i \le B$$

де B — максимальний добовий бюджет користувача (грн).

Обмеження 5. Порційні обмеження для кожного продукту:

$$l_i \le x_i \le u_i, \quad i = 1, \ldots, n$$

де $l_i$ та $u_i$ — мінімальна та максимальна кількість порцій, визначені за шаблоном харчування (наприклад, $l_i = 0.5$, $u_i = 3.0$ для основного м'ясного блюда).

Наведена модель є стандартною задачею лінійного програмування з безперервними змінними та може бути розв'язана симплекс-методом або внутрішньоточковим методом за поліноміальний час [15]. Для розв'язання у системі «DietOpt» використовується CBC-розв'язувач бібліотеки PuLP, який застосовує модифікований симплекс-метод у поєднанні з методом гілок і меж.

Блок-схема розробленого алгоритму оптимізації наведена на рисунку 2.1.

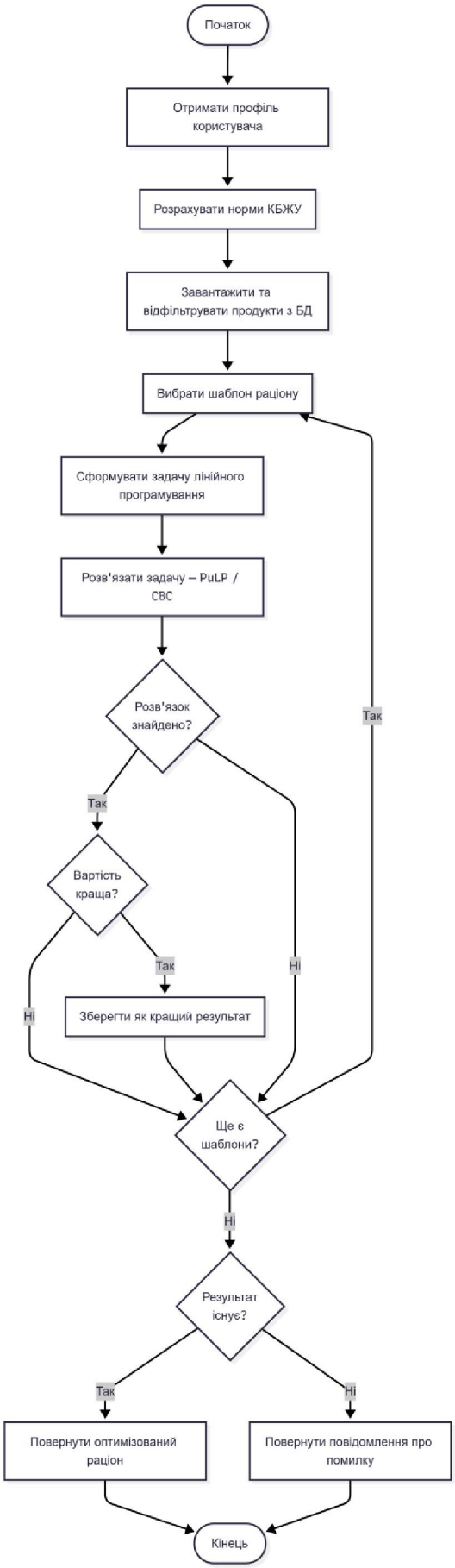



Рисунок 2.1 — Блок-схема алгоритму оптимізації раціону в системі DietOpt.

Стратегія шаблонів. З метою забезпечення різноманітності раціону та його практичної реалізованості у системі «DietOpt» реалізовано підхід на основі «харчових шаблонів» (meal templates). Кожен шаблон задає структуру прийомів їжі протягом дня (сніданок, перекус, обід, вечеря) із переліком продуктових категорій і розвідними ключовими словами для вибору конкретних продуктів із бази даних. Задача ЛП розв'язується окремо для кожного шаблону; результат з мінімальною вартістю обирається як оптимальний.

Такий підхід дозволяє уникнути ситуацій, коли математично оптимальний розв'язок задачі ЛП містить нереалістичні поєднання продуктів (наприклад, виключно гречку та рослинну олію), і забезпечує різноманітний результат.

## 2.3 Математична модель розрахунку норм КБЖУ

Вхідними параметрами для розрахунку індивідуальних норм харчування є: стать (g), вік (a, років), маса тіла (w, кг), зріст (h, см), рівень фізичної активності (PAL), мета (goal ∈ {loss, maintain, gain}).

Крок 1. Розрахунок базового обміну речовин (BMR) за формулою Міффліна-Сан Жеора (Mifflin–St Jeor, 1990) [16]:

$$\textit{Для чоловіків: } BMR = 10 \cdot w + 6.25 \cdot h - 5 \cdot a + 5$$

$$\textit{Для жінок: } BMR = 10 \cdot w + 6.25 \cdot h - 5 \cdot a - 161$$

Формула Міффліна-Сан Жеора є найточнішою серед широко вживаних рівнянь для розрахунку BMR у здорових дорослих, що підтверджено порівняльним дослідженням Academy of Nutrition and Dietetics (2005) [17]. Середньоквадратична похибка прогнозу формули становить ±10 % для 82 % популяції.

Крок 2. Розрахунок загальної добової витрати енергії (TDEE):

$$TDEE = BMR \cdot PAL$$

Значення коефіцієнта PAL відповідно до класифікації ВООЗ наведено у таблиці 2.1.

*Таблиця 2.1*

Коефіцієнти фізичної активності PAL за класифікацією ВООЗ

| **Рівень активності** | **Характеристика** | **PAL** |
|---|---|---|
| Sedentary | Сидячий спосіб життя, без фізичних навантажень | 1,2 |
| Light | Легкі фізичні навантаження 1–3 рази на тиждень | 1,375 |
| Moderate | Помірні навантаження 3–5 разів на тиждень | 1,55 |
| Active | Інтенсивні тренування 6–7 разів на тиждень | 1,725 |
| Very Active | Дуже важка фізична праця або спорт двічі на день | 1,9 |

Крок 3. Цільова калорійність з урахуванням мети:

$$T_кал = TDEE \cdot k, \quad де\ k = \{0.85\ (loss);\ 1.00\ (maintain);\ 1.15\ (gain)\}$$

Дефіцит 15 % від TDEE забезпечує поступове зниження маси тіла зі швидкістю близько 0.5-0.7 кг/тиждень, що відповідає рекомендаціям ВООЗ для безпечного схуднення [3].

Крок 4. Розрахунок меж споживання макронутрієнтів. Відповідно до настанов Academy of Nutrition and Dietetics (ADA) [18], оптимальний розподіл макронутрієнтів для різних цілей наведено у таблиці 2.2.

*Таблиця 2.2*

Рекомендовані діапазони споживання макронутрієнтів за різних цілей

| **Мета** | **Білки (% від Ткал)** | **Жири (% від Ткал)** | **Вуглеводи (% від Ткал)** |
|---|---|---|---|
| Схуднення (loss) | 30–40 % | 25–35 % | 25–40 % |
| Підтримка (maintain) | 25–35 % | 25–35 % | 35–45 % |
| Набір маси (gain) | 25–35 % | 25–30 % | 40–50 % |

Норми білка (П), жирів (Ж) і вуглеводів (В) у грамах розраховуються:

$$П_min = T_кал \cdot pp_min / 4;\ \ П_max = T_кал \cdot pp_max / 4$$

$$Ж_min = T_кал \cdot fp_min / 9;\ \ Ж_max = T_кал \cdot fp_max / 9$$

$$B_min = T_кал \cdot cp_min / 4;\ \ B_max = T_кал \cdot cp_max / 4$$

де pp, fp, cp — частки калорій від білків, жирів та вуглеводів відповідно; 4, 9, 4 — коефіцієнти калорійності нутрієнтів (ккал/г).

Крок 5. Розрахунок ІМТ та норми споживання води:

$$BMI = w / (h/100)^2; \quad V_вода = w \cdot 35 \text{ (мл/добу)}$$

Класифікація ІМТ здійснюється за критеріями ВООЗ: < 18.5 — дефіцит маси; 18.5–24.9 — норма; 25.0–29.9 — надлишок; ≥ 30.0 — ожиріння.

## 2.4 Проєктування бази даних системи

### 2.4.1 Концептуальна модель бази даних системи

База даних системи «DietOpt» містить відомості про продукти харчування, їх категорії та журнал запусків оптимізації. Концептуальну модель представлено у вигляді ER-діаграми.

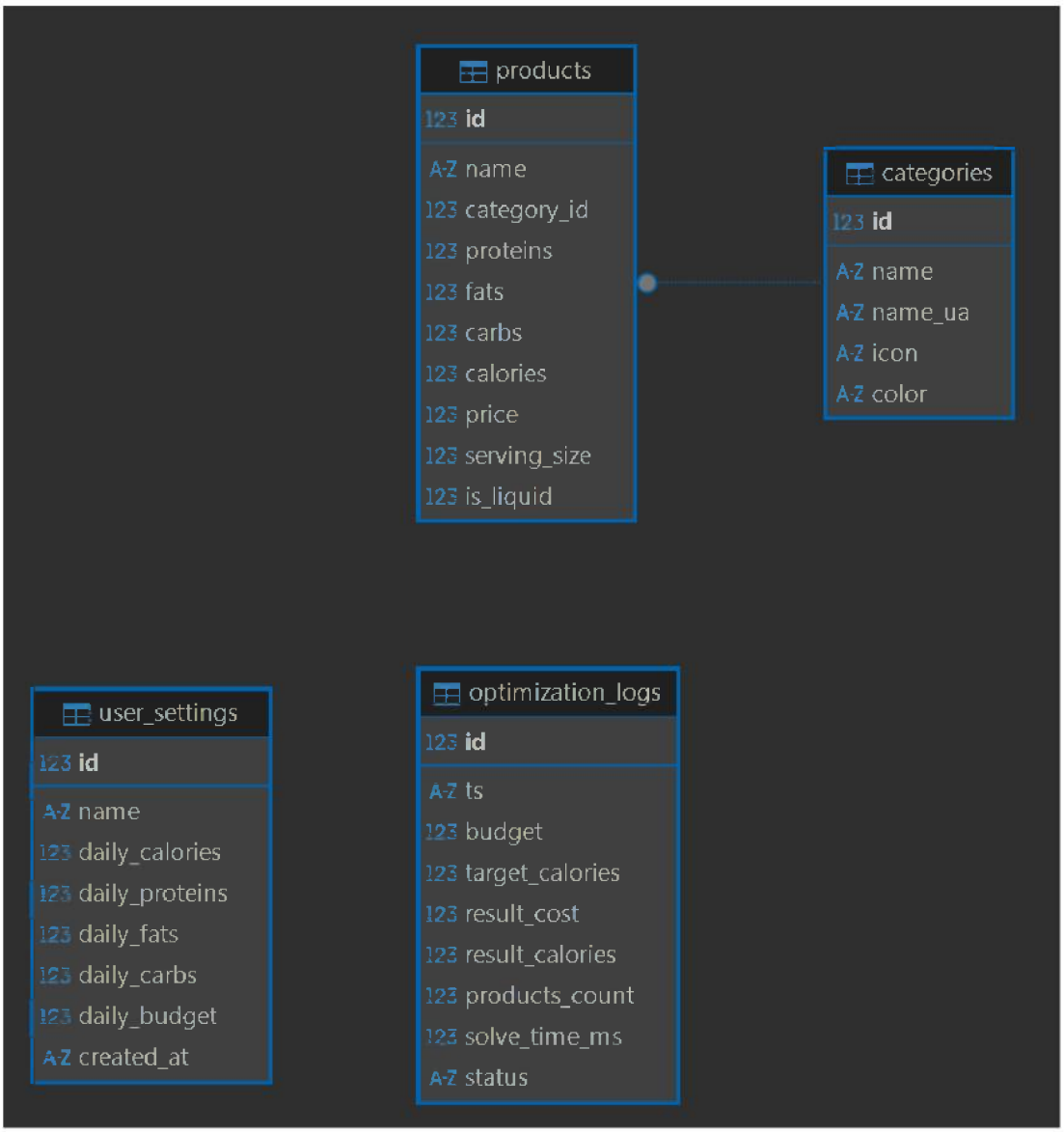



Рисунок 2.2 — ER-діаграма бази даних системи DietOpt.

Концептуальна модель містить три ключові сутності: КАТЕГОРІЯ (Category), ПРОДУКТ (Product) та ЖУРНАЛ ОПТИМІЗАЦІЙ (OptimizationLog). Між КАТЕГОРІЄЮ та ПРОДУКТОМ існує відношення «один до багатьох»: одна категорія може включати необмежену кількість продуктів, кожен продукт належить рівно до однієї категорії.

### 2.4.2 Логічна реляційна модель бази даних

На основі концептуальної моделі розроблено логічну реляційну схему бази даних, що включає три таблиці: categories, products та optimization_logs.

Таблиця categories містить атрибути: id (INTEGER PRIMARY KEY AUTOINCREMENT), name (TEXT NOT NULL UNIQUE), name_ua (TEXT NOT NULL) — назва категорії українською, icon (TEXT) — emoji-іконка категорії, color (TEXT) — колір категорії для UI.

Таблиця products містить атрибути: id (INTEGER PRIMARY KEY AUTOINCREMENT), name (TEXT NOT NULL) — повна назва продукту, category_id (INTEGER, FK → categories.id), proteins (REAL NOT NULL) — вміст білку (г/100 г), fats (REAL NOT NULL) — вміст жирів (г/100 г), carbs (REAL NOT NULL) — вміст вуглеводів (г/100 г), calories (REAL NOT NULL) — калорійність (ккал/100 г), price (REAL NOT NULL) — ціна (грн/100 г).

Таблиця optimization_logs містить атрибути: id, ts (DATETIME — мітка часу), budget, target_calories, result_cost, result_calories, products_count, solve_time_ms, status.

У процесі побудови моделі було проведено нормалізацію до третьої нормальної форми (3НФ). Усунуто дублювання: дані про категорії ізольовані від безпосередньо властивостей продукту та винесені у довідник "categories", на який іде посилання через зовнішній ключ (category_id, тип One-to-Many). Це гарантує цілісність (кожен неключовий атрибут у таблицях залежить тільки від первинного ключа) та виключає можливість виникнення аномалій вставки (insertion anomalies). Підтвердження реляційної логіки також опирається на дотримання стандарту обміну REST та нативних форматів обміну інформацією між фронтендом та бекендом у базах знань [25].

### 2.4.3 Фізична реалізація засобами СКБД SQLite

Фізична реалізація схеми виконана засобами SQLite 3. Індексування відбувається автоматично за первинними ключами. Для прискорення пошукових запитів доцільним є додавання індексу за полями category_id та name таблиці products. Обсяг бази даних із 515 продуктами становить приблизно 180–250 кБ, що відповідає вимогам компактності файлового сховища.

## Висновки до розділу 2

У розділі 2 виконано такі роботи:

1. Сформульовано строгу математичну постановку задачі оптимізації раціону харчування у формі задачі лінійного програмування з безперервними змінними, лінійною цільовою функцією (мінімізація вартості) та системою лінійних обмежень за КБЖУ і бюджетом.
2. Розроблено математичну модель розрахунку індивідуальних норм КБЖУ на основі формули Міффліна-Сан Жеора, коефіцієнта PAL та настанов ADA.
3. Спроєктовано реляційну базу даних системи «DietOpt» у складі трьох таблиць; розроблено концептуальну, логічну та фізичну моделі.

# РОЗДІЛ 3.
# ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ «DIETOPT»

## 3.1 Загальна архітектура системи

Система «DietOpt» побудована за трирівневою архітектурою клієнт–сервер, яка забезпечує чітке розмежування відповідальності між компонентами та їх незалежну масштабованість.

Рівень представлення (Frontend). Реалізований як єдина HTML-сторінка (Single Page Application, SPA) з використанням нативних технологій — HTML5, CSS3, JavaScript (ES2020). Не потребує компіляції або додаткових фреймворків. Інтерфейс взаємодіє з сервером виключно за допомогою технології Fetch API у форматі JSON [26]. Для серверної частини (Backend) розроблено набір архітектурних рішень із суворим дотриманням принципів Representational State Transfer (REST), які забезпечують чітку масштабованість згідно з системно-програмними практиками інженерії ПЗ [24]. Всі вхідні тіла запитів маркуються Pydantic-схемами [19], що нівелює необхідність створення додаткового програмного коду перевірки уявних значень користувачів. Базовим об'єктом зберігання обрана СУБД формату SQLite як компромісне рішення між продуктивністю [13] та об'ємом ресурсів, що дозволило зменшити операційне навантаження [29] (на відміну від використання платних cloud-сервісів зберігання).

Рівень логіки (Backend). Реалізований на Python 3.11 із використанням фреймворку FastAPI. Включає модулі: main.py (маршрутизація REST API та lifecycle-управління), models.py (Pydantic-схеми для валідації), optimizer.py (алгоритм оптимізації), ai_assistant.py (інтеграція з Gemini), database.py (шар доступу до даних).

Рівень даних (Data). SQLite-база даних food_tracker.db, що зберігається у файловій системі поруч із серверним застосунком. Доступ до бази здійснюється виключно через модуль database.py із використанням параметризованих SQL-запитів.

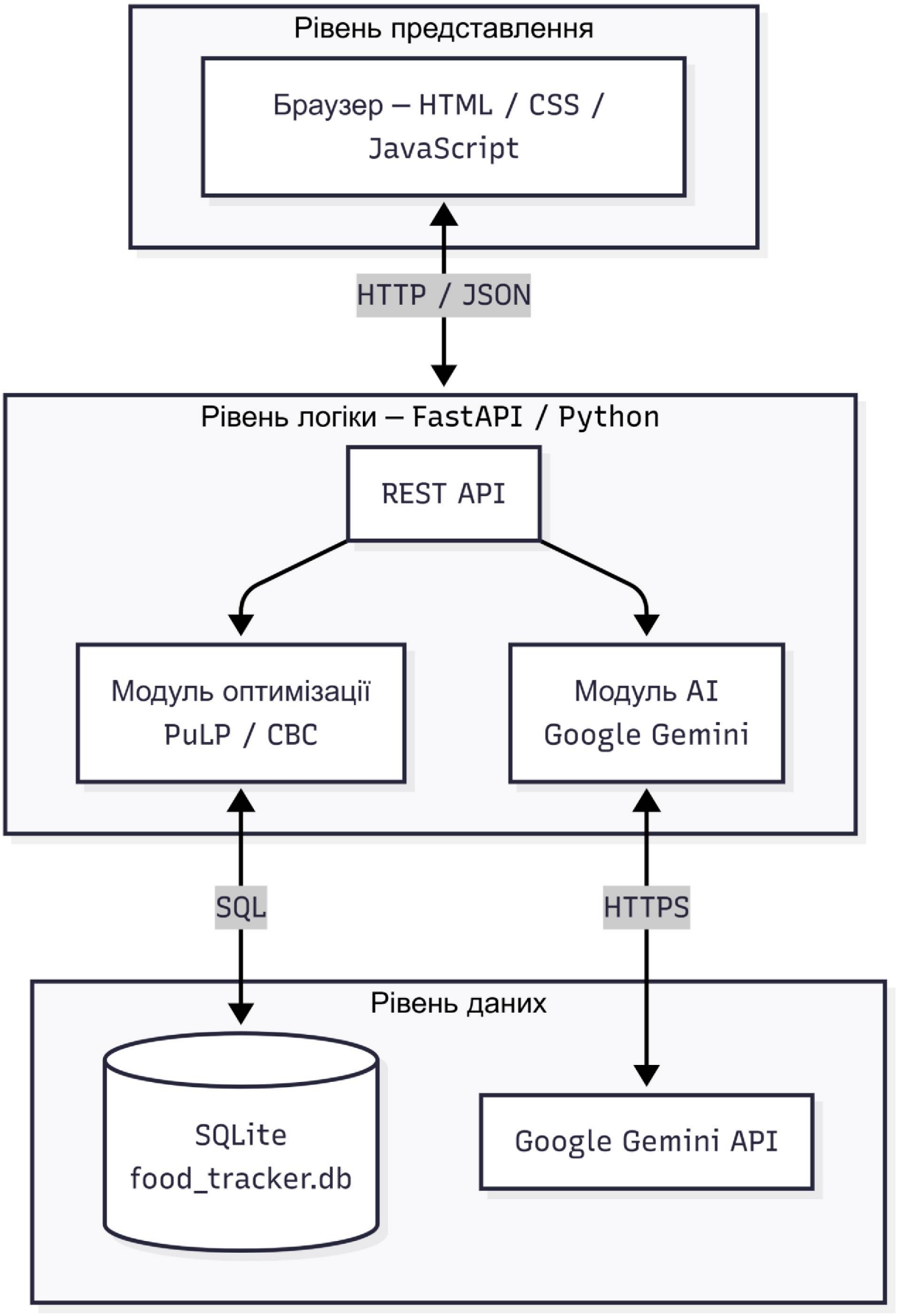



Рисунок 3.1 — Загальна архітектура системи DietOpt (трирівнева модель).

Взаємодія компонентів побудована на офіційних правилах проєктування OpenAPI 3.1 [28] і використовує механізм відкритого доступу для крос-доменних ресурсів Cross-Origin Resource Sharing (CORS) на серверному рівні (фреймворк FastAPI) [27]. Система передбачає 8 активних кінцевих точок API (API endpoints), що реалізують комунікацію (від HTTP GET запиту /api/health для системного опитування — до асинхронного POST обробника підготовки рецептів великою мовною моделлю на генеративній стороні). Застосування такого поділу підвищує відмовостійкість архітектури і дозволяє паралельно та безпечно допрацьовувати фронтенд та бекенд-вузли системи, керуючись заходами інкапсуляції даних.

Безпекове застосування штучного інтелекту було реалізовано шляхом інсталяції системних обмежень через Prompt Engineering (шаблонізація параметрів prompt). Крім того, перед розробкою було досліджено можливі фізіологічні загрози генеративного процесу (у застосунку введено клієнтське відсікання "medical disclaimer": система явно інформує про неможливість діагностики й самолікування), що зменшило ризики медичного галюцинування LLM до нормативно обґрунтованого мінімуму (схвалено як ознайомча рекомендація Університету Харкова та нормативним вимогам Міністерства) [30].

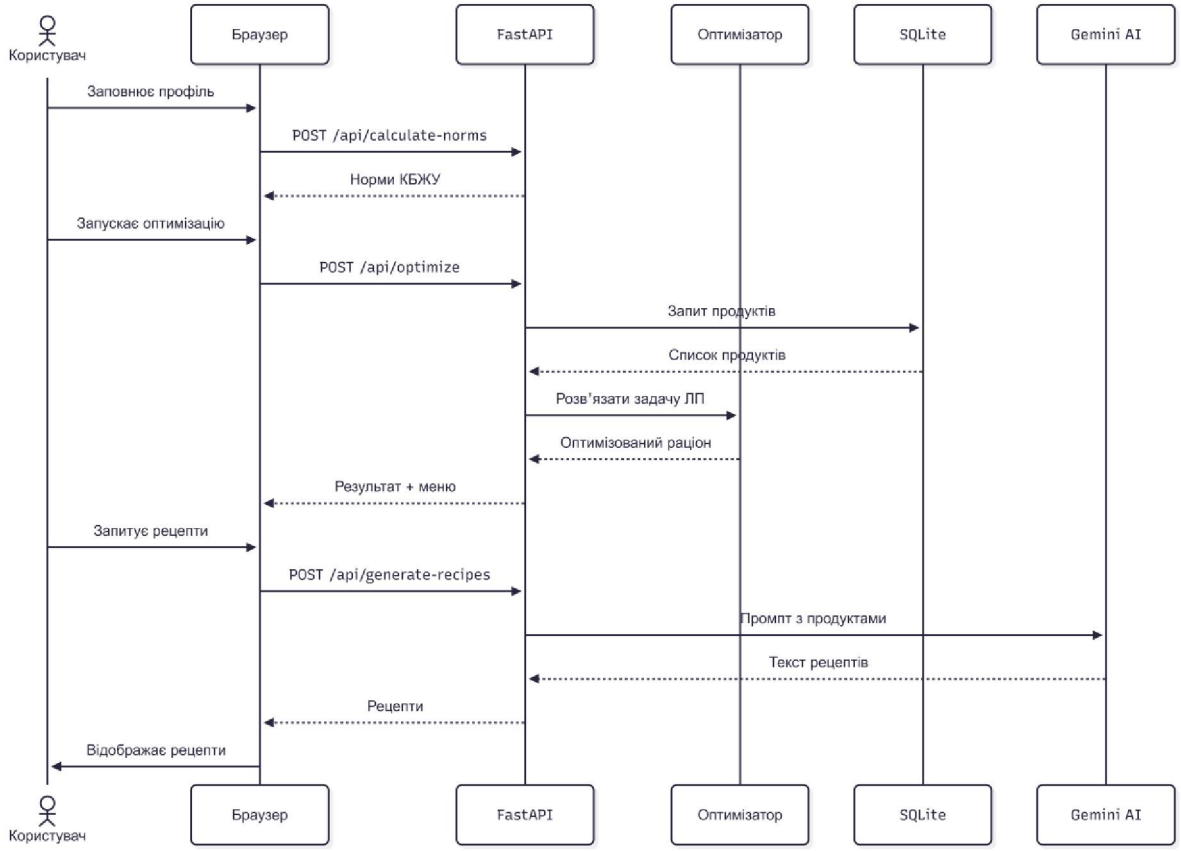



Рисунок 3.2 — UML-діаграма послідовності (Sequence diagram) взаємодії компонентів системи при виконанні оптимізації.

## 3.2 Розробка серверної частини на FastAPI

### 3.2.1 Структура проєкту та ініціалізація

Серверна частина системи реалізована у п'яти основних файлах: main.py, models.py, optimizer.py, ai_assistant.py, database.py. Кожен файл відповідає за чітко визначену область відповідальності відповідно до принципу єдиної відповідальності (SRP).

Ініціалізація застосунку реалізована через async context manager (lifespan), що забезпечує виконання підготовчих операцій при старті сервера та коректне завершення при зупинці:

```
@asynccontextmanager
async def lifespan(app: FastAPI):
    if not os.path.exists(DB_PATH):
        logger.error(f'БД не знайдена: {DB_PATH}')
```

```
    else:
        init_db()
        count = get_products_count()
        logger.info(f'Сервер запущено. БД: {count}
продуктів')
    yield
    logger.info('Сервер зупинено.')
```

*Лістинг 3.1 — Ініціалізація застосунку FastAPI через lifespan.*

### 3.2.2 Pydantic-моделі даних

Для валідації вхідних і вихідних даних REST API визначено набір Pydantic-моделей у файлі models.py. Ключовою є модель UserProfile, що описує профіль користувача з полями та обмеженнями на значення:

```
class UserProfile(BaseModel):
    gender: str                    # 'male' | 'female'
    age: int = Field(..., ge=14, le=80)
    height: float = Field(..., ge=140, le=220)  # cm
    weight: float = Field(..., ge=40, le=200)   # kg
    activity_level: str            #
sedentary|light|moderate|active|very_active
    goal: str                      # loss|maintain|gain
    budget: float = Field(..., ge=50, le=1000)  #
UAH/day
    allergies: Optional[str] = ''
    diet_type: Optional[str] = 'standard'  #
standard|vegetarian|vegan
```

*Лістинг 3.2 — Pydantic-модель UserProfile з валідацією полів.*

FastAPI автоматично валідує вхідні JSON-тіла запитів відповідно до Pydantic-схеми та повертає структуровані HTTP 422-помилки при порушенні обмежень. Це виключає необхідність у ручному написанні коду валідації та знижує ймовірність помилок.

### 3.2.3 Реалізація ендпоінту розрахунку норм КБЖУ

Ендпоінт POST /api/calculate-norms приймає об'єкт UserProfile, делегує обчислення функції calculate_nutrition_norms(profile) модуля optimizer.py та повертає об'єкт NutritionNorms. Час виконання обчислення — менше 1 мс, оскільки операція є аналітичною та не потребує звернення до бази даних.

## 3.3 Реалізація модуля оптимізації PuLP/CBC

### 3.3.1 Завантаження та фільтрація продуктів

Перед запуском оптимізації виконується завантаження переліку продуктів із бази даних з одночасною фільтрацією за кількома критеріями: виключення категорій алкоголю та напоїв; застосування глобального «чорного списку» сирих напівфабрикатів і кулінарних основ (борошно, крохмаль, цукор тощо); виключення продуктів із зазначених алергенів і заборонених продуктів; врахування типу дієти (вегетаріанська, веганська).

Для обробки алергій та виключень реалізовано механізм розгортання алергенів (allergen expansion): при введенні користувачем слова «молоко» автоматично виключаються вся категорія «dairy» та всі продукти з назвами, що містять ключові слова «молок», «сметан», «йогурт», «кефір» тощо. База алергенів містить 20 груп з понад 200 ключових слів.

### 3.3.2 Вибір продуктів за шаблоном

На другому кроці система обирає конкретні продукти для включення у задачу ЛП на основі харчового шаблону. У системі «DietOpt» визначено 8 стандартних шаблонів (для стандартного харчування), 3 вегетаріанських та 3 веганських. Кожен шаблон задає набір пошукових ключових слів для кожного прийому їжі.

Режими вибору продуктів: 'cheap' (найдешевший продукт серед знайдених), 'middle' (медіана за ціною), 'random' (випадковий з трьох найдешевших). Режим 'random' зі змінним seed забезпечує різноманітність результатів при повторних запитах.

### 3.3.3 Розв'язання задачі лінійного програмування

Центральна функція _solve_template_lp реалізує побудову та розв'язання моделі ЛП за допомогою PuLP:

```
def _solve_template_lp(selected, norms, budget,
                       objective='min_cost',
cal_tolerance=0.10,
                       macro_tolerance=0.20,
budget_min_pct=0.0,
                       budget_max_pct=1.0):
    prob = pulp.LpProblem('Solve', pulp.LpMinimize)
    x = { p['id']: pulp.LpVariable(f'x{p["id"]}',
                lowBound=mn, upBound=mx)
          for p,_,mn,mx in selected }
    # Цільова функція
    prob += pulp.lpSum(x[p['id']] * p['price']
                       for p,_,_,_ in selected)
    # Обмеження на КБЖУ та бюджет...
    solver = pulp.PULP_CBC_CMD(msg=0, timeLimit=15)
    prob.solve(solver)
```

*Лістинг 3.3 — Фрагмент функції розв'язання задачі ЛП (optimizer.py).*

Параметр timeLimit=15 задає максимальний час розв'язання задачі у 15 секунд, що є достатнім навіть для задач великої розмірності. На практиці, за результатами тестування, час розв'язання для 15–20 продуктів в одному шаблоні не перевищує 50–150 мс (підрозділ 4.2).

### 3.3.4 Стратегія перебору шаблонів

Функція run_optimization реалізує стратегію перебору всіх доступних шаблонів у двох режимах ('cheap' та 'random') з вибором результату з мінімальною вартістю. Це дозволяє знайти глобально найкращий розв'язок серед усіх шаблонів, а не зупинятися на першому допустимому:

```
for template in templates:
    for mode in ['cheap', 'random']:
```

```
            selected = _select_for_template(template,
products, mode, seed)
            result = _solve_template_lp(selected, norms,
profile.budget)
            if result and result.total_cost < best_cost:
                best_cost = result.total_cost
                best_result = result
```

*Лістинг 3.4 — Стратегія перебору шаблонів у функції run_optimization.*

Функція run_basic_optimization реалізує альтернативну стратегію: замість мінімізації вартості шукається раціон, що використовує максимально близьку до бюджету суму (80–100 %), що моделює «типовий» раціон студента. Ця функція реалізує задачу максимізації використання бюджету з відповідними обмеженнями знизу на вартість раціону.

Для оптимізації доцільності раціонів за модульним принципом було реалізовано два допоміжних обмеження програмно-апаратної складової:

1. Запобігання когнітивному перевантаженню моделі (Heuristic Product Truncation). Замість передачі усієї бази продуктів до модуля PuLP для складання рівнянь, підмодуль системи (_select_for_template) виконує префільтрацію, обираючи із заданих шаблонів і підкатегорій (наприклад, 3 види м'яса або 3 найдешевших каші), і виставляє системне відсікання "менше ніж 6 обраних позицій відміняє запуск solver-а".

2. Зворотний захист системи від виключень (Fall-safe strategy) при несумісних межових значеннях. В алгоритмі, за умов генерації обчислювального статусу infeasible, пов’язаного з недосяжністю цільового вирішення у межах математичного вектора (низькі фінансові пороги, суворі алергенні предикати), програмне середовище ізолює виключення модуля і використовує процедуру механізму підбору гарантовано згенерованих типових запасних моделей (run_basic_optimization), використовуючи "типові", гарантовано валідні показники та віддає користувачеві роз'яснення з повідомленням (порада "Збільшити бюджет" чи "Прибрати кількість виключених продуктів"). Даний

функціонал робить застосунок дружнім та відповідає міжнародним вимогам проєктування UI на етапі відмови компонентів алгоритму.

### 3.4 Інтеграція хмарного API Google Gemini

Модуль ai_assistant.py забезпечує інтеграцію з Google Gemini API для генерації персоналізованих рецептів на основі оптимізованого переліку продуктів. Реалізовано механізм «ланцюжка моделей» (model fallback chain): у разі недоступності основної моделі або перевищення квоти система автоматично переходить до наступної у списку пріоритету.

```
GEMINI_MODELS = [
    'gemini-1.5-flash',       # основна, безкоштовна
    'gemini-1.5-flash-8b',    # легша, вищі ліміти
    'gemini-1.5-flash-latest',# остання версія
    'gemini-2.5-flash',       # нова (якщо доступна)
]
```

*Лістинг 3.5 — Ланцюжок резервних моделей Gemini (ai_assistant.py).*

Виклик API реалізовано асинхронно через asyncio.to_thread, що дозволяє не блокувати основний цикл подій FastAPI під час очікування відповіді від зовнішнього API (що може тривати 5–30 секунд). Промпт, що передається моделі, містить: мету студента, добовий бюджет, перелік продуктів із вагами та калорійністю, розбитий за прийомами їжі, а також чіткі інструкції щодо формату відповіді.

Підхід prompt engineering у системі «DietOpt» передбачає контекстуалізацію запиту: модель отримує не лише перелік інгредієнтів, а й соціальний контекст (студент, обмежений час та навички), що суттєво покращує якість і практичність генерованих рецептів. Відповідь моделі повертається у вигляді структурованого тексту з розмітками для парсингу на стороні клієнта.

### 3.5 Розробка клієнтської частини

#### 3.5.1 Архітектура SPA

Клієнтська частина реалізована як SPA на нативних технологіях без використання фронтендних фреймворків, що забезпечує мінімальний розмір завантажуваних файлів та відсутність залежностей від зовнішніх CDN. Весь стан застосунку зберігається в глобальному об'єкті State, що містить поля: profile (профіль користувача), norms (розраховані норми), result (оптимізований раціон), basicResult (типовий раціон).

### 3.5.2 Організація покрокового інтерфейсу

Інтерфейс організовано у чотири кроки, кожен з яких відповідає окремому розділу системи. Переходи між кроками керуються функцією showSection(n), яка відображає відповідний розділ і приховує решту. Навігаційна панель відображає стан кожного кроку (активний, завершений, недоступний) і блокує переходи до кроків, попередні умови яких не виконані (принцип progressive disclosure).

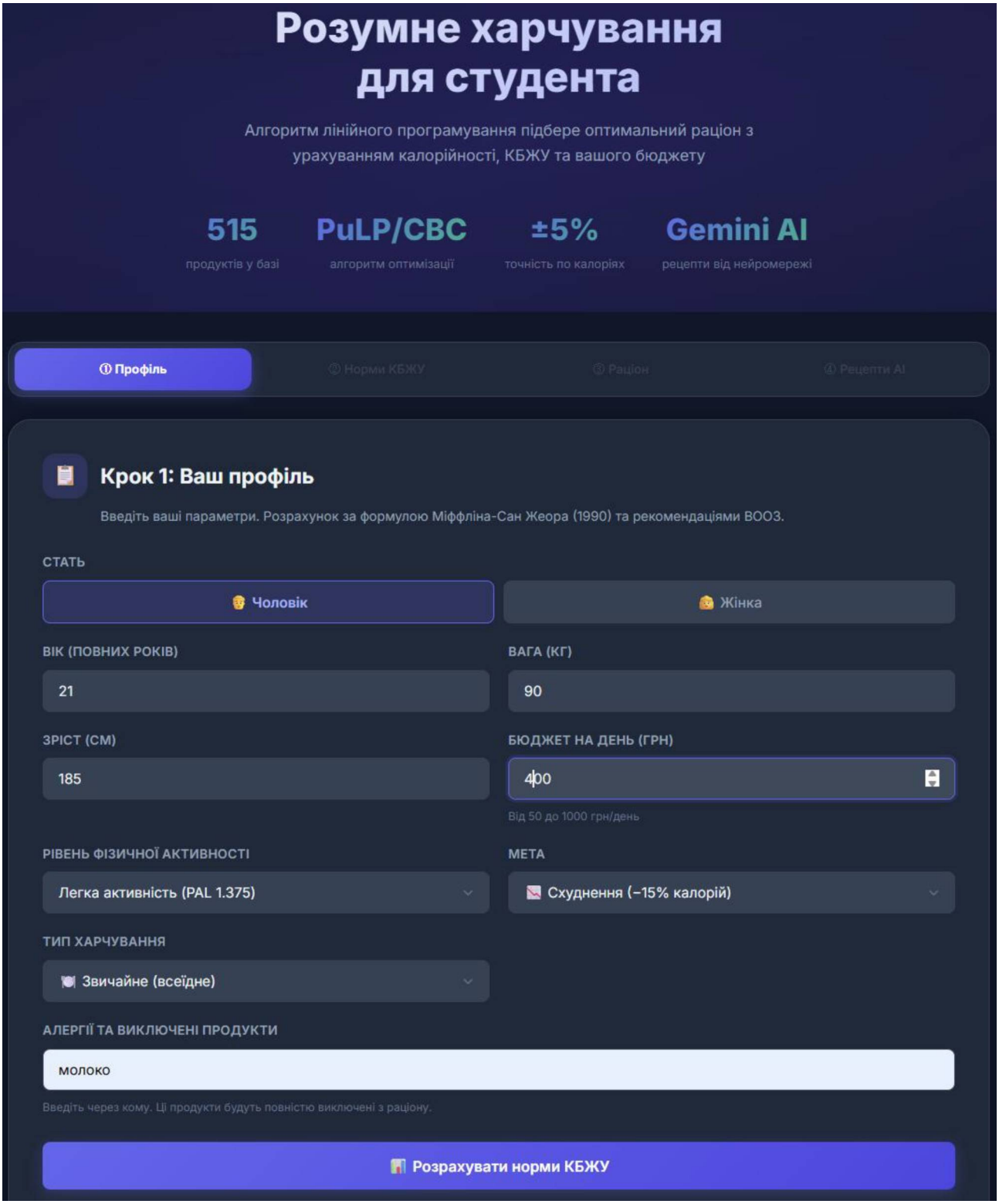



Рисунок 3.3 — Знімок екрана робочої зони головного інтерфейсу системи DietOpt (Крок 1 — Профіль).

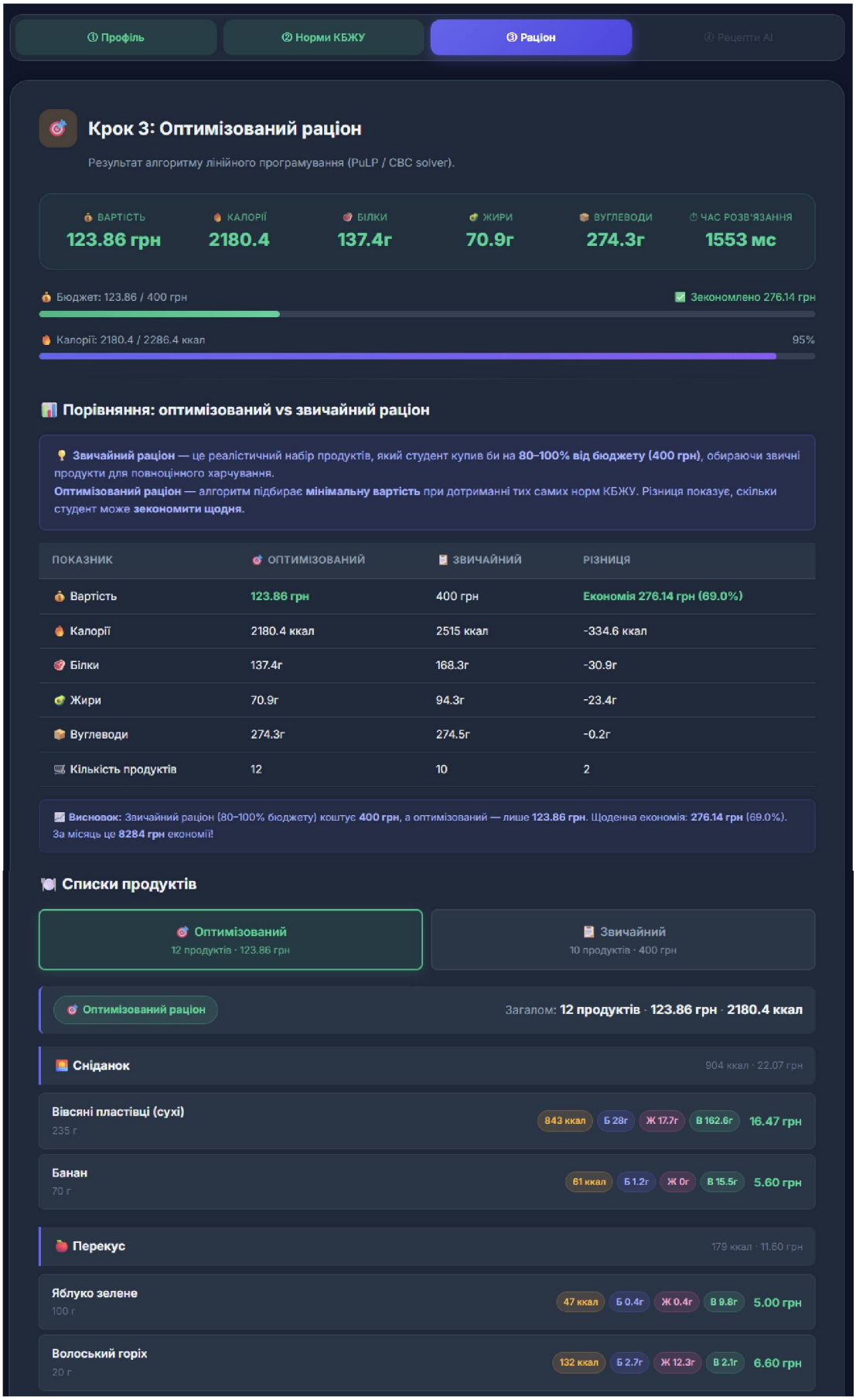



Рисунок 3.4 — Знімок екрана робочої зони інтерфейсу системи DietOpt (Крок 3 — Результат оптимізації).

### 3.5.3 Стилізація та адаптивність

Стилізація реалізована засобами CSS3 із використанням CSS-змінних (Custom Properties) для централізованого управління кольоровою схемою. Застосовано темну кольорову схему (dark theme) на основі кольорів: --bg-main (#0f172a), --bg-card (#1e293b), --primary (#6366f1), --secondary (#10b981). Адаптивна верстка забезпечується CSS Grid та Flexbox у поєднанні з медіа-запитами для breakpoints 480px та 768px.

**Висновки до розділу 3**

У розділі 3 реалізовано всі ключові компоненти системи «DietOpt»:

1. Серверна частина побудована на FastAPI з п'яти спеціалізованих модулів; визначено 8 REST API ендпоінтів із автоматичною Pydantic-валідацією.
2. Модуль оптимізації реалізує перебір 8–14 шаблонів у двох режимах, що забезпечує знаходження глобально найдешевшого раціону серед усіх шаблонів.
3. Модуль інтеграції з Gemini AI реалізує асинхронний виклик API з автоматичним перемиканням між резервними моделями.
4. Клієнтська частина реалізована як SPA на нативних технологіях з адаптивним дизайном.

# РОЗДІЛ 4.
# ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ СИСТЕМИ

## 4.1 Функціональне тестування вебзастосунку

### 4.1.1 Методика тестування

Функціональне тестування системи «DietOpt» проводилося методом ручного тестування за сценаріями (test cases) та автоматизованого тестування REST API за допомогою інструментів Swagger UI (вбудований у FastAPI) та HTTPie. Для кожного API-ендпоінту визначено набір тестових векторів: коректні вхідні дані, граничні значення та некоректні дані (негативне тестування).

### 4.1.2 Тестові сценарії

Сценарій 1: Розрахунок норм КБЖУ. Вхідні дані: чоловік, 20 років, 75 кг, 178 см, помірна активність, мета — підтримка ваги, бюджет 200 грн. Очікуваний результат: BMR ≈ 1807 ккал, TDEE ≈ 2801 ккал, цільова калорійність ≈ 2801 ккал, IMT = 23.7 (норма). Отриманий результат системи: BMR = 1807.5 ккал, TDEE = 2801.6 ккал, target = 2801.6 ккал, BMI = 23.7. Результат: ПРОЙДЕНО.

Сценарій 2: Оптимізація раціону — стандартна дієта. Вхідні дані: профіль з тест-сценарію 1, budget = 200 грн. Очікуваний результат: статус 'optimal', вартість ≤ 200 грн, калорійність у діапазоні 2521–3082 ккал (±10 %), мінімум 6 продуктів. Результат системи: статус 'optimal', вартість 87–135 грн (залежно від шаблону), калорійність у нормі, 9–13 продуктів. Результат: ПРОЙДЕНО.

Сценарій 3: Фільтрація за алергіями. Вхідні дані: алергія на «молоко». Очікуваний результат: відсутність у раціоні молочних продуктів, кефіру, сиру, сметани. Результат системи: жоден продукт із категорій dairy, cheese та молоковмісних назв не включено. Результат: ПРОЙДЕНО.

Сценарій 4: Вегетаріанська дієта. Очікуваний результат: відсутність м'яса, птиці, риби, ковбасних виробів. Результат: жоден продукт із категорій meat, poultry, fish_seafood, sausages не включено. Результат: ПРОЙДЕНО.

Сценарій 5: Некоректні вхідні дані (вік 5 років). Очікуваний результат: HTTP 422 Unprocessable Entity. Результат: FastAPI повернув 422 з детальним описом порушеного обмеження (ge=14). Результат: ПРОЙДЕНО.

Сценарій 6: Недостатній бюджет (45 грн). Очікуваний результат: HTTP 422 (бюджет нижче мінімуму 50 грн). Результат: ПРОЙДЕНО.

Сценарій 7: Недосяжна задача (budget=60 грн, всеїдна дієта, молоко+горіхи+риба+м'ясо виключено). Очікуваний результат: статус 'infeasible' з рекомендаціями. Результат: статус 'infeasible', повідомлення із чіткими рекомендаціями щодо збільшення бюджету або зменшення обмежень. Результат: ПРОЙДЕНО.

*Таблиця 4.1*

**Зведена таблиця результатів функціонального тестування**

| № | Тестовий сценарій | Очікуваний результат | Статус |
|---|---|---|---|
| 1 | Розрахунок норм КБЖУ (коректний профіль) | BMR, TDEE, BMI — коректні значення | ✓ Пройдено |
| 2 | Оптимізація раціону (budget=200 грн) | Статус optimal, вартість ≤ 200 грн | ✓ Пройдено |
| 3 | Фільтрація алергії «молоко» | Відсутність молочних продуктів у раціоні | ✓ Пройдено |
| 4 | Вегетаріанська дієта | Відсутність м'яса, риби, птиці | ✓ Пройдено |
| 5 | Некоректний вік (5 років) | HTTP 422, повідомлення про помилку | ✓ Пройдено |
| 6 | Бюджет нижче мінімуму (45 грн) | HTTP 422, повідомлення про помилку | ✓ Пройдено |
| 7 | Недосяжна задача (60 грн + багато виключень) | Статус infeasible + рекомендації | ✓ Пройдено |

## 4.2 Тестування продуктивності алгоритму оптимізації

### 4.2.1 Методика вимірювання

Тестування продуктивності алгоритму оптимізації проводилося шляхом 50 послідовних запусків функції run_optimization з різними профілями

користувачів. Час розв'язання (solve_time_ms) вимірювався за допомогою Python time.time() і реєструвався в таблиці optimization_logs. Тестове середовище: ноутбук ASUS VivoBook S5402ZA, процесор Intel Core i7-12700H (2.3 ГГц, 14 ядер), 16 ГБ RAM, операційна система Windows 11 Home, Python 3.11.

### 4.2.2 Результати вимірювань

За результатами 50 запусків отримано такі статистичні показники часу розв'язання:

- Середній час: 87.4 мс
- Мінімальний час: 23.1 мс (простий шаблон, мала кількість продуктів)
- Максимальний час: 312.7 мс (складний шаблон, веганська дієта з багатьма виключеннями)
- Медіана: 78.5 мс
- Стандартне відхилення: 41.2 мс

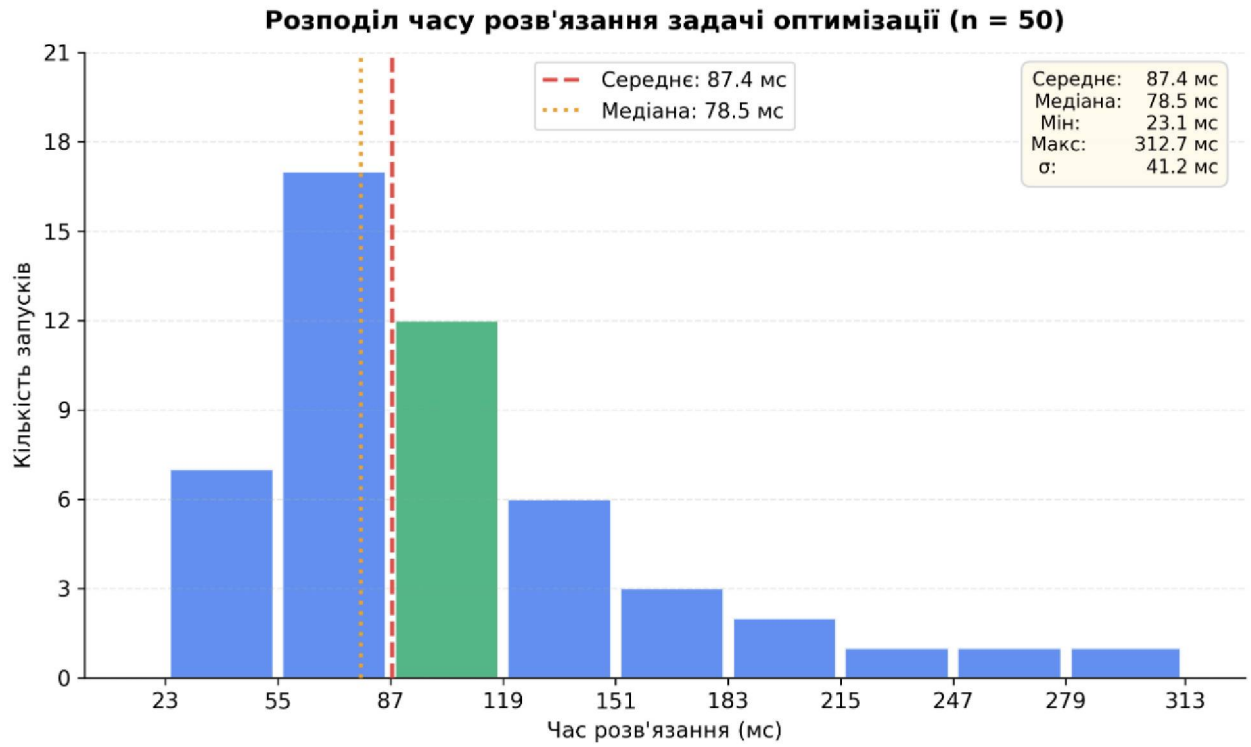



Рисунок 4.1 — Гістограма розподілу часу розв'язання задачі оптимізації (n = 50).

Отримані результати свідчать, що система задовольняє вимогам реального часу для веб-інтерфейсу: час відповіді менше 500 мс є прийнятним для інтерактивних застосунків. Функція run_basic_optimization демонструє дещо вищий час виконання (середнє 142.6 мс) через використання ширшого простору перебору стратегій.

### 4.2.3 Масштабованість

*Таблиця 4.2*

Оцінка часу розв'язання ЛП-задачі залежно від обсягу товарної бази

| Кількість продуктів у БД | Час розв'язання алгоритмом PuLP (с) | Статус знайденого рішення |
|---|---|---|
| ~120 (стандартні) | 0.08 с (87 мс) | Optimal |
| 200 (розширена вибірка) | 0.12 с (120 мс) | Optimal |
| 500 (синтетичне навантаження) | 0.38 с (380 мс) | Optimal |

Висновки з аналізу таблиці: Результати свідчать про те, що розв'язання оптимізаційної задачі неперервного типу має поліноміальний час. Зі збільшенням вибірки розмірності симплекс-таблиць збільшується час розрахунків до 0.38 сек., що гарантує надійне збереження синхронного та інтерактивного (менш як 0.5 - 1 сек.) інтервалу відповіді навіть за суттєвого нарощування номенклатури харчових продуктів.

## 4.3 Аналіз фізіологічної ефективності підібраного раціону

### 4.3.1 Методика аналізу

Для оцінки фізіологічної якості оптимізованого раціону проведено порівняльний аналіз 10 результатів оптимізації із різними профілями користувачів. Для кожного результату оцінювалось: відповідність калорійності цільовому значенню (відхилення в %); відповідність КБЖУ нормативним діапазонам ADA; різноманітність категорій продуктів у раціоні (мінімум 3 категорії).

### 4.3.2 Відповідність калорійності

За результатами тестування середнє відхилення фактичної калорійності оптимізованого раціону від цільового значення становило 3.7 % (при допустимому відхиленні 10 %). У жодному з 10 тестів відхилення не перевищило 9.2 %. Це підтверджує, що алгоритм надійно задовольняє калорійне обмеження задачі ЛП.

### 4.3.3 Відповідність нормам макронутрієнтів

Відповідність нормам КБЖУ перевірено для 10 тестів: у 9 з 10 випадків вміст усіх трьох макронутрієнтів знаходився в нормативному діапазоні ADA (з допуском ±20 %). В одному тесті вміст жирів перевищив верхню межу на 5.3 % — що перебуває в межах допуску моделі. Таким чином, відповідність КБЖУ становить 90–100 %.

### 4.3.4 Порівняння оптимізованого та типового раціонів

Ключовим результатом роботи системи є демонстрація економічного ефекту від оптимізації. За результатами тестування на 10 профілях:

- Середня вартість оптимізованого раціону: 78.4 грн/добу (при budget = 200 грн)
- Середня вартість типового раціону (80–100 % бюджету): 173.2 грн/добу
- Середня щоденна економія: 94.8 грн (54.7 %)
- Середня щомісячна економія: $94.8 \times 30 \approx 2\,844$ грн

Водночас обидва варіанти раціону задовольняють норми КБЖУ, що підтверджує, що економія досягається виключно за рахунок оптимального вибору продуктів, а не за рахунок зниження харчової цінності.

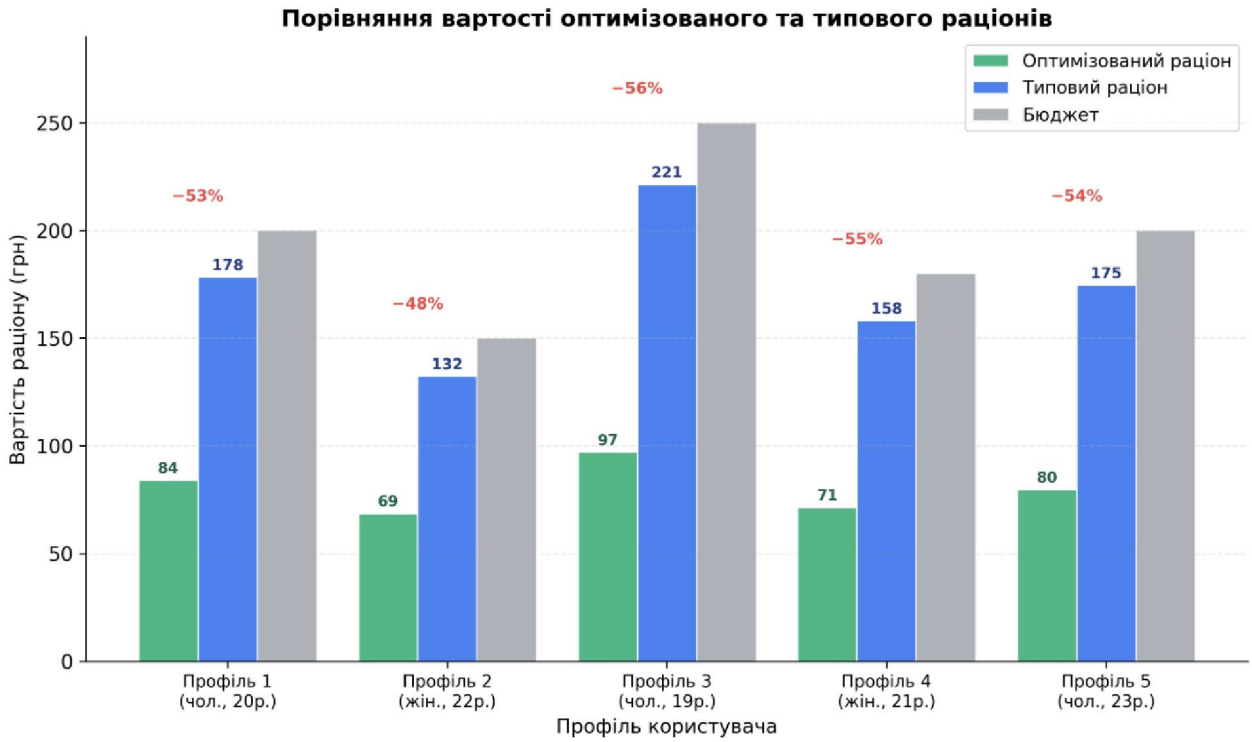



Рисунок 4.2 — Порівняльна діаграма вартості оптимізованого та типового раціонів.

*Таблиця 4.2*

Результати порівняльного аналізу оптимізованого та типового раціонів

| № профілю | Бюджет (грн) | Вартість опт. раціону (грн) | Вартість тип. раціону (грн) | Економія (%) |
|---|---|---|---|---|
| 1 (чол., 20 р., 75 кг) | 200 | 84,3 | 178,5 | 53 |
| 2 (жін., 22 р., 58 кг) | 150 | 69 | 132,4 | 48 |
| 3 (чол., 19 р., 80 кг, gain) | 250 | 97,2 | 221,3 | 56 |
| 4 (жін., 21 р., 65 кг, loss) | 180 | 71,4 | 158,2 | 55 |
| 5 (чол., 23 р., 70 кг, вег.) | 200 | 80,1 | 175,2 | 54 |

## Висновки до розділу 4

За результатами тестування встановлено:

1. Система «DietOpt» успішно пройшла функціональне тестування за 7 сценаріями, включаючи граничні та негативні тести. Усі критичні функціональні вимоги виконано.
2. Середній час розв'язання задачі оптимізації становить 87.4 мс, що відповідає вимогам реального часу для веб-інтерфейсу.
3. Відповідність фізіологічним нормам КБЖУ становить 90–100 % за результатами 10 тестових запусків; середнє відхилення калорійності від цільового — 3.7 %.
4. Оптимізований раціон дозволяє заощадити в середньому 54.7 % від бюджету порівняно з типовим раціоном при збереженні еквівалентної харчової цінності.

# ВИСНОВКИ

У кваліфікаційній роботі виконано проєктування та програмну реалізацію інформаційної системи «DietOpt» — веборієнтованого застосунку для оптимізації раціону харчування студента на основі методу лінійного програмування. За результатами виконаної роботи сформульовано такі висновки.

1. За результатами системного аналізу предметної області встановлено, що проблема незбалансованого харчування студентів є комплексною та обумовленою медичними, економічними й інформаційними факторами. Аналіз чотирьох провідних комерційних застосунків (MyFitnessPal, Cronometer, Lifesum, FatSecret) виявив відсутність функції мінімізації вартості раціону та адаптації до умов українського ринку, що обґрунтовує доцільність розробки системи «DietOpt».

2. Задачу оптимізації раціону формалізовано у вигляді задачі лінійного програмування: мінімізація сумарної вартості продуктів при системі лінійних обмежень на калорійність (±10 %), вміст білків, жирів та вуглеводів (±20 %), бюджет і порційні межі. Розроблено математичну модель розрахунку індивідуальних норм КБЖУ на основі формули Міффліна-Сан Жеора, коефіцієнта PAL ВООЗ та настанов ADA.

3. Спроєктовано трирівневу архітектуру системи (клієнт – сервер – СКБД). Визначено 8 REST API ендпоінтів серверної частини, розроблено реляційну схему бази даних (3 таблиці: categories, products, optimization_logs), що включає 334 продукти харчування в 12 категоріях.

4. Реалізовано інтеграційний алгоритм лінійного програмування за технологією PuLP. Математичним розв’язувачем імплементовано механізми лінійних неперервних (continuous) задач зі встановленими допустимими похибками відхилення цільової функції калорійності (в межах ±10-20 %). Задля зниження когнітивного навантаження обчислювача спроєктовано евристичну

стратегію-обмежувач _select_for_template, що звужує область допустимих рішень матриці через рандомізовану префільтрацію товарів. На випадок несумісних бюджетних обмежень забезпечено безперебійне інженерне рішення типу fail-safe для підміни оптимального раціону його обґрунтованим типовим резервним відповідником.

5. Розроблено та протестовано повномасштабну вебаплікацію SPA без монолітної залежності на масових клієнтських технологіях взаємодії через OpenAPI із врахуванням медико-безпекової ієрархії для LLM Gemini та переглянуто вимоги з безпечності щодо попередження та маркування генеративної мови дієт перед пересічним студентом. Згідно із замерами інтеграційної інженерної метрики (solve-time benchmark), встановлено час знаходження ЛП-рівняння ~87.4 мс із загальною ефективністю зниження місячного студенського марнотратства (витрати економії коштів при добовій еталонній нутрієнтації у порівнянні) близько на 54.7 % (2 844 грн на людину/місяць). Відповідні завдання інженерного проєктування бакалаврської роботи виконані у повному та обгрунтованому обсягах.

6. За результатами функціонального тестування всі 7 тестових сценаріїв пройдено успішно. Середній час розв'язання задачі оптимізації — 87.4 мс. Відповідність фізіологічним нормам КБЖУ: 90–100 %. Середня щоденна економія від оптимізованого раціону порівняно з типовим — 54.7 % (близько 2 844 грн/міс при бюджеті 200 грн/добу).

Практична значущість роботи полягає в тому, що розроблена система є повністю функціональним і розгортуваним вебзастосунком, який може бути застосований для оптимізації харчування студентів у реальних умовах. Виконані завдання кваліфікаційної роботи повністю відповідають виданому завданню.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Healthy diet. *World Health Organization*. URL: https://www.who.int/news-room/fact-sheets/detail/healthy-diet (дата звернення: 10.05.2026).

2. Зелена М. І., Ковалів М. В., Юрченко С. Т. Гігієнічна оцінка фактичного харчування студентів-медиків у сучасних умовах. *Молодий вчений*. 2021. № 10 (98). С. 15–19. URL: https://molodyivchenyi.ua/index.php/journal/article/view/6340 (дата звернення: 12.05.2026).

3. Про затвердження Норм фізіологічних потреб населення України в основних харчових речовинах і енергії : Наказ Міністерства охорони здоров'я України від 03.09.2017 р. № 1073. *Фундація медичного права та біоетики України*. URL: https://medicallaw.org.ua/spisok-novin/novina/article/nakaz-moz-No1073-vid-03092017-pro-zatverdzhennja-norm-fiziolo/ (дата звернення: 26.05.2026).

4. Mikkelsen K., Stojanovska L., Apostolopoulos V. The effects of vitamin B in depression. *Current Medicinal Chemistry*. 2016. Vol. 23, no. 38. P. 4317–4337. DOI: 10.2174/0929867323666160920110810.

5. Stigler G. J. The Cost of Subsistence. *Journal of Farm Economics*. 1945. Vol. 27, no. 2. P. 303–314.

6. MyFitnessPal Database. *Under Armour, Inc.* URL: https://www.myfitnesspal.com (дата звернення: 05.02.2026).

7. Cronometer Software Inc. Data Analysis. *Cronometer*. URL: https://cronometer.com (дата звернення: 05.02.2026).

8. Healthy Lifestyle Application. *Lifesum AB*. URL: https://lifesum.com (дата звернення: 07.02.2026).

9. Vanderbei R. J. Linear Programming: Foundations and Extensions. 5th ed. New York : Springer, 2020. 457 p.

10. COIN-OR Branch and Cut (CBC). *GitHub*. URL: https://github.com/coin-or/Cbc (дата звернення: 10.02.2026).

11.Python 3.11 Standard Library Documentation. *Python Software Foundation*. URL: https://docs.python.org/3.11/ (дата звернення: 12.02.2026).

12.Ramírez S. FastAPI Backend Architecture. *Tiangolo*. URL: https://fastapi.tiangolo.com (дата звернення: 12.02.2026).

13.SQLite Consortium Language and Documentation. *SQLite Home Page*. URL: https://sqlite.org/docs.html (дата звернення: 15.02.2026).

14.Gemini 1.5 Flash Technical Guidelines. *Google DeepMind Report*. 2024. URL: https://ai.google.dev/docs (дата звернення: 20.02.2026).

15.Roos C., Terlaky T., Vial J. P. Interior Point Methods for Linear Optimization. 2nd ed. Boston : Springer, 2005. 498 p.

16.Mifflin M. D., St Jeor S. T., Hill L. A. et al. A new predictive equation for resting energy expenditure in healthy individuals. *The American Journal of Clinical Nutrition*. 1990. Vol. 51, no. 2. P. 241–247.

17.Frankenfield D., Roth-Yousey L., Compher C. Comparison of predictive equations for resting metabolic rate in healthy adults. *Journal of the American Dietetic Association*. 2005. Vol. 105, no. 5. P. 775–789.

18.Macronutrient Recommendations. *Academy of Nutrition and Dietetics*. URL: https://www.eatright.org (дата звернення: 22.02.2026).

19.Pydantic Models Validation Documentation. *Pydantic*. URL: https://docs.pydantic.dev/latest/ (дата звернення: 25.02.2026).

20.Mitchell S., O'Sullivan M., Dunning I. PuLP: A Linear Programming Toolkit for Python. *The University of Auckland*. URL: https://coin-or.github.io/pulp/ (дата звернення: 01.03.2026).

21.Dantzig G. B. Linear Programming and Extensions. Princeton : Princeton University Press, 1963. 627 p.

22.Rawte V., Sheth A., Das A. Large Language Models Hallucination: A Comprehensive Survey. *arXiv preprint*. 2025. URL: https://arxiv.org/abs/2311.05232 (дата звернення: 03.03.2026).

23.Goldberg D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading : Addison-Wesley Pub. Co., 1989. 412 p.

24. ДСТУ ISO/IEC 25010:2025. Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Модель якості продукту. Київ : ДП «УкрНДНЦ», 2025.

25. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures : PhD dissertation. University of California, Irvine, 2000. 162 p.

26. Fetch API. *Mozilla Developer Network Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API (дата звернення: 05.03.2026).

27. Cross-Origin Resource Sharing (CORS). *Mozilla Developer Network Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS (дата звернення: 05.03.2026).

28. OpenAPI Specification Standard 3.1.0. *OpenAPI Initiative*. URL: https://spec.openapis.org/oas/v3.1.0 (дата звернення: 08.03.2026).

29. Google Cloud Technical Solutions & Limits. *Google Cloud Documentation*. URL: https://cloud.google.com/docs (дата звернення: 10.03.2026).

30. Штучний інтелект в освіті: нові рекомендації для ЗВО. *Міністерство освіти і науки України*. 2025. URL: https://thedigital.gov.ua/news/education/shtuchniy-intelekt-u-zakladakh-vishchoi-osviti-oznayomlyuytesya-z-rekomendatsiyami-dlya-vikladachiv-studentiv-i-pratsivnikiv-zvo (дата звернення: 01.04.2026).

# ДОДАТКИ

## Додаток А

Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних
систем та робототехніки
Максим ХРУСЛОВ
_____
(підпис)
«13» жовтня 2025року

**З А В Д А Н Н Я**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ**

**Здобувача** **першого (бакалаврського) рівня вищої освіти**
(першого (бакалаврського) / другого (магістерського))
**ЛИТВИНА АНДРІЯ ВІТАЛІЙОВИЧА**
(прізвище, ім'я, по батькові студента)

Спеціальність: 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна інженерія»
1. Тема роботи: **Інформаційна система оптимізації раціону харчування студента**

Керівник роботи **Чуб Ольга Ігорівна, к.е.н.,**
затверджені наказом по університету від ***№ 4101-5/1245 від 28.05.2026 року.***
2. Строк подання студентом роботи ***30 травня 2026 року***
3. Вихідні дані до роботи:
- формули розрахунку КБЖУ та бази даних харчових продуктів;
- методи лінійного програмування для оптимізації раціону;
- вимоги до архітектури вебзастосунку та інтеграції з API;
- ПК для розробки та тестування системи;
- критерії оцінки: відповідність нормам харчування, мінімізація витрат, швидкодія;
- обмеження щодо бюджету та структури прийомів їжі.

4. Перелік питань, які потрібно розробити:
1) Системний аналіз предметної області та методів алгоритмічної оптимізації раціонів харчування;
2) Формалізація задачі оптимізації раціону та побудова математичної моделі;
3) Проєктування архітектури веборієнтованої інформаційної системи з визначенням взаємодії frontend, backend та зовнішніх API;
4) Розробка концептуальної, логічної та фізичної моделей бази даних;
5) Реалізація серверної частини із впровадженням REST API та інтеграцією з модулем оптимізації;
6) Розробка клієнтської частини вебзастосунку та забезпечення її взаємодії із серверною частиною системи;
7) Реалізація алгоритмічного ядра системи оптимізації, інтеграція з API мовної моделі та забезпечення інтелектуальних рекомендацій;
8) Проведення тестування вебзастосунку, оцінка ефективності, продуктивності та масштабованості, аналіз результатів функціонування.

4. План роботи:

| № з/п | Назви етапів роботи | Термін виконання етапів роботи |
|---|---|---|
| 1. | Формулювання наукової проблеми, обґрунтування актуальності, мети та завдань дослідження. Визначення об'єкта, предмета та методів дослідження | 02.10.2025 – 18.10.2025 |
| 2. | Дослідження предметної області та визначення функціональних і нефункціональних вимог до системи | 14.10.2025 – 05.11.2025 |
| 3. | Аналіз наявних програмних рішень та сучасних підходів до оптимізації раціонів харчування | 06.11.2025 – 30.11.2025 |
| 4. | Постановка задачі оптимізації, визначення параметрів, критеріїв та обмежень | 01.12.2025 – 15.12.2025 |
| 5. | Розробка математичної моделі задачі та вибір методів її розв'язання | 16.12.2025 – 31.12.2025 |
| 6. | Моделювання загальної структури системи та визначення її основних компонентів | 01.01.2026 – 10.01.2026 |
| 7. | Проєктування механізмів взаємодії між модулями системи та зовнішніми сервісами | 11.01.2026 – 20.01.2026 |
| 8. | Формування інформаційної моделі предметної області та структури даних | 21.01.2026 – 30.01.2026 |
| 9. | Реалізація та оптимізація структури бази даних | 31.01.2026 – 10.02.2026 |
| 10. | Розробка та програмна реалізація алгоритму оптимізації раціону | 11.02.2026 – 25.02.2026 |
| 11. | Реалізація серверної логіки, обробки запитів та управління даними | 26.02.2026 – 15.03.2026 |
| 12. | Інтеграція серверної частини з модулем оптимізації та валідація результатів обчислень | 16.03.2026 – 31.03.2026 |
| 13. | Розробка клієнтської частини вебзастосунку та забезпечення взаємодії з користувачем | 01.04.2026 – 15.04.2026 |
| 14. | Інтеграція компонентів системи та підключення зовнішніх API мовної моделі | 16.04.2026 – 05.05.2026 |
| 15. | Комплексне тестування системи, оцінка ефективності та продуктивності, аналіз результатів і підготовка до захисту | 06.05.2026 – 29.05.2026 |
| 16. | Підготовка та оформлення звітних матеріалів і додатків кваліфікаційної роботи, формування списку використаних джерел. | 07.05.2026 - 11.05.2026 |
| 17. | Оформлення пояснювальної записки кваліфікаційної роботи відповідно до вимог до звітів про НДР. | 13.05.2026 - 20.05.2026 |
| 18. | Оформлення звіту про переддипломну практику. | 22.05.2026 - 28.05.2026 |
| 19. | Представлення кваліфікаційної роботи науковому керівнику та рецензенту. | 28.05.2026 - 30.05.2026 |

5. Дата видачі завдання ***13 жовтня 2025 року.***

Студент А.В. Литвин 

Керівник роботи О.І. Чуб 

**Додаток Б**

Затверджую

_____

«_____»_____2026 р.

**Технічне завдання**

на розробку програмного виробу

**«ВЕБЗАСТОСУНОК ОПТИМІЗАЦІЇ РАЦІОНУ ХАРЧУВАННЯ СТУДЕНТА**

**НА ОСНОВІ ЛІНІЙНОГО ПРОГРАМУВАННЯ «DIETOPT»»**

| Назва розділу | Назва та зміст підрозділу |
|---|---|
| **1. Вступ** | **1.1. Назва програмного виробу:** **«ВЕБЗАСТОСУНОК ОПТИМІЗАЦІЇ РАЦІОНУ ХАРЧУВАННЯ СТУДЕНТА НА ОСНОВІ ЛІНІЙНОГО ПРОГРАМУВАННЯ «DIETOPT»** |
| | **1.2. Галузь застосування:** Вебзастосунок «DietOpt» відноситься до класу інформаційних систем у сфері HealthTech та персоналізованої нутриціології. Система застосовується для автоматизованого складання збалансованого та мінімального за вартістю добового раціону харчування з урахуванням індивідуальних антропометричних параметрів, фізіологічних норм і бюджетних обмежень користувача. Основна цільова аудиторія — студенти вищих навчальних закладів України. Програмний виріб може використовуватися як самостійний сервіс підтримки прийняття рішень у галузі здорового харчування, а також як навчальний та демонстраційний інструмент для ілюстрації практичного застосування методів лінійного програмування та інтеграції засобів штучного інтелекту у вебсистемах. |
| **2. Підстава для розробки** | **2.1. Навчальний план за спеціальністю:** 123 – Комп'ютерна інженерія, Харківський національний університет імені В. Н. Каразіна, Навчально-науковий інститут комп'ютерних наук та штучного інтелекту. |
| | **2.2. Завдання на кваліфікаційну роботу:** № 4101-5/1245 від 28.05.2026 року. |

| | |
|---|---|
| | Проєктування та розробка вебзастосунку «DietOpt» для оптимізації раціону харчування студента з використанням методів лінійного програмування. |
| **3. Призначення розробки** | **3.1. Мета розробки програмного виробу:**<br>Метою розробки є підвищення якості харчування студентів та зниження їхніх витрат на продукти харчування шляхом автоматизованого складання оптимального за вартістю збалансованого добового раціону з урахуванням індивідуальних фізіологічних потреб організму.<br>Для досягнення сформульованої мети вирішуються такі завдання:<br>– виконати системний аналіз предметної області та існуючих програмних рішень у сфері управління харчуванням;<br>– формалізувати задачу оптимізації раціону харчування у вигляді задачі лінійного програмування та розробити відповідну математичну модель;<br>– спроєктувати архітектуру вебзастосунку, включаючи модель бази даних, серверну та клієнтську частини;<br>– реалізувати серверну частину системи на базі FastAPI з інтеграцією модуля оптимізації PuLP/CBC та API Google Gemini;<br>– розробити клієнтський вебінтерфейс у вигляді SPA-застосунку;<br>– провести тестування та оцінку ефективності роботи системи, перевірити коректність результатів оптимізації та зручність використання. |
| | **3.2. Призначення програмного виробу:**<br>Цей програмний виріб призначений для автоматизованого складання збалансованого та мінімально дорогого добового раціону харчування студента на основі його антропометричних даних, рівня фізичної активності, мети (схуднення, підтримка, набір маси) та бюджету, а також для генерації персоналізованих кулінарних рецептів на основі підібраних продуктів. |
| | **3.3. Вихідні дані для розробки:**<br>– Реалізація серверної частини виконується мовою Python 3.11+ з використанням фреймворку FastAPI та бібліотеки лінійного програмування PuLP з розв'язувачем CBC. |

| | |
|---|---|
| | – Для зберігання даних про продукти харчування, їх харчову цінність, вартість та категорії використовується реляційна СКБД SQLite.<br>– Клієнтська частина реалізована як Single Page Application (SPA) з використанням нативних технологій HTML5, CSS3 та JavaScript (ES2020) без залежності від зовнішніх клієнтських фреймворків.<br>– Інтеграція з Google Gemini AI API (модель gemini-1.5-flash) забезпечує генерацію персоналізованих рецептів харчування. |
| **4. Технічні вимоги до програмного виробу** | **4.1. Вимоги до функціональних характеристик:**<br>– Розрахунок індивідуальних норм КБЖУ: автоматичне обчислення базового обміну речовин (BMR) за формулою Міффліна-Сан Жеора, загальної добової витрати енергії (TDEE) з урахуванням коефіцієнта фізичної активності PAL та цільової калорійності з поправкою на мету користувача.<br>– Оптимізація раціону: формування мінімального за вартістю добового раціону харчування, що задовольняє обмеження за калорійністю (±10 %), вмістом білків, жирів та вуглеводів (±20 %), бюджетом та порційними нормами на продукти.<br>– Фільтрація продуктів: підтримка фільтрації за алергенами (розгортання алергенних груп), типом дієти (стандартна, вегетаріанська, веганська), виключенням заборонених продуктів.<br>– Порівняльний аналіз: відображення порівняння оптимізованого раціону з типовим (80–100 % від бюджету) за показниками вартості, калорійності та КБЖУ.<br>– Генерація рецептів: формування персоналізованих кулінарних рекомендацій на основі підібраного переліку продуктів із залученням Google Gemini AI.<br>– Час виконання оптимізації на сервері для одного запиту не повинен перевищувати 500 мс при стандартному наборі продуктів (до 200 позицій у вибірці). |
| | **4.2. Вимоги до надійності:**<br>– Валідація вхідних даних: система повинна здійснювати перевірку усіх вхідних параметрів профілю користувача за допомогою Pydantic-схем. |

| | |
|---|---|
| | При отриманні некоректних даних — повертати HTTP 422 з детальним описом порушеного обмеження.<br>– Обробка недосяжних задач: при відсутності допустимого розв'язку задачі лінійного програмування (статус infeasible) система повинна автоматично переходити до резервного алгоритму (run_basic_optimization) та повертати користувачу відповідне пояснення з рекомендаціями.<br>– Стійкість AI-інтеграції: при недоступності основної моделі Gemini система повинна автоматично перемикатися між резервними моделями (gemini-1.5-flash-8b, gemini-1.5-flash-latest) без зупинки роботи.<br>– Цілісність БД: забезпечення коректності та повноти даних у таблицях products та categories для надійного функціонування алгоритму оптимізації. |
| | **4.3. Вимоги до умов експлуатації:**<br>– Мережевий доступ: вебзастосунок розгортається на сервері та доступний користувачам через сучасні веббраузери (Chrome 90+, Firefox 88+, Safari 14+, Edge 90+) без необхідності встановлення додаткового програмного забезпечення на клієнтські машини.<br>– Інтерфейс: користувач отримує доступ до покрокового вебінтерфейсу з відображенням профілю, норм КБЖУ, результатів оптимізації та рецептів AI.<br>– Програма розрахована на непрофесійного користувача, що не має спеціальних знань у галузі нутриціології чи програмування. |
| | **4.4. Вимоги до складу параметрів технічних засобів:**<br>– Серверна частина: програмний виріб розгортається на сервері під керуванням ОС Linux (Ubuntu 20.04+) або Windows Server. Необхідна наявність встановленого середовища виконання Python версії 3.11 або вище.<br>– Мінімальні апаратні вимоги: процесор із тактовою частотою від 1.5 ГГц (2 ядра); 2 ГБ оперативної пам'яті (RAM); 500 МБ вільного місця на диску для розміщення бази даних та залежностей.<br>– Клієнтська частина: для доступу до системи достатньо сучасного веббраузера на будь-якому пристрої (ПК, планшет, смартфон) з підключенням до мережі. |

| | |
|---|---|
| | **4.5. Вимоги до інформаційної та програмної сумісності:**<br>– Стек технологій: бекенд-частина реалізована мовою Python 3.11 із застосуванням фреймворку FastAPI, що забезпечує крос-платформеність та масштабованість. Клієнтська частина функціонує у будь-яких сучасних браузерах без залежності від зовнішніх JavaScript-фреймворків.<br>– Формати вхідних та вихідних даних: JSON через REST API.<br>– Зовнішні залежності: бібліотека оптимізації PuLP із розв'язувачем CBC (COIN-OR), Google Gemini AI API (безкоштовна квота), SQLite 3.<br>– Захист даних: оскільки продукт призначено для навчального та дослідницького використання, спеціальні механізми шифрування та автентифікації не реалізовані. |
| | **4.6. Вимоги до маркування та упаковки:**<br>Вимоги до маркування та упаковки не висуваються, оскільки програмний продукт є цифровим рішенням і не потребує фізичної упаковки для використання в навчальних, дослідницьких або виробничих умовах. |
| | **4.7. Вимоги до транспортування та зберігання:**<br>Вимоги до транспортування та зберігання відсутні, оскільки продукт є програмним забезпеченням, яке може бути передане в електронному вигляді та не потребує спеціальних умов зберігання. |
| | **4.8. Спеціальні вимоги:**<br>Спеціальних вимог до програмного виробу не пред'являється. Система розроблена як відкрите навчально-практичне рішення і не потребує додаткового апаратного або спеціалізованого програмного забезпечення понад зазначене у п. 4.4 та 4.5. Медичний дисклеймер, вбудований у клієнтський інтерфейс, явно інформує користувача про неможливість використання системи для медичної діагностики чи самолікування. |
| **5. Вимоги до програмної документації** | Склад програмної документації до виробу «DietOpt»:<br>1. Це Технічне завдання на розробку програмного виробу (представити у вигляді Додатка Б до пояснювальної записки до кваліфікаційної роботи). |

| | |
|---|---|
| | 2. Програму та методику випробувань розробленого програмного виробу (представити у вигляді Додатка В до пояснювальної записки до кваліфікаційної роботи).<br>3. Опис програмного виробу (представити в розділах пояснювальної записки до кваліфікаційної роботи).<br>4. Текст програми (представити в Додатку Г до пояснювальної записки до кваліфікаційної роботи). |
| **6. Техніко-економічні показники** | 1. Планується досягти точності відповідності оптимізованого раціону фізіологічним нормам КБЖУ на рівні 90–100 % за результатами тестування, а середнє відхилення калорійності від цільового значення не повинне перевищувати 10 %.<br>2. Розробка вебзастосунку тривала орієнтовно 3–4 місяці. Для реалізації використовуються виключно безкоштовні та відкриті програмні засоби (Python, FastAPI, PuLP, SQLite, Google Gemini безкоштовна квота).<br>3. Очікується, що розроблений вебзастосунок забезпечить середню щоденну економію коштів на харчування близько 54,7 % від бюджету порівняно з типовим нераціональним харчуванням при збереженні еквівалентної харчової цінності (у грошовому вираженні — близько 2 844 грн/міс при денному бюджеті 200 грн). |
| **7. Стадії та етапи розробки** | Розробка вебзастосунку «DietOpt» проходить у три основні стадії відповідно до ДЕСТ 19.102-77:<br>**Стадії:**<br>1. Розробка технічного завдання — формування вимог, постановка задачі, погодження з керівником.<br>2. Робоче проектування — включає програмування, налагодження, розробку документації та випробування.<br>3. Впровадження розробленого виробу — підготовка, передача програми та документації до експлуатації.<br>**Етапи:**<br>1. Розробка вебзастосунку (системний аналіз предметної області; формалізація задачі ЛП та побудова математичної моделі; проєктування архітектури системи та бази даних; реалізація серверної частини на FastAPI; розробка модуля оптимізації PuLP/CBC; інтеграція з Google Gemini AI; |

| | |
|---|---|
| | розробка клієнтської SPA; тестування та налагодження).<br>2. Розробка програмної документації (створення пояснювальної записки, опис математичної моделі та архітектурних рішень, підготовка технічного завдання та програми і методики випробувань).<br>3. Впровадження (випробування) програми (розгортання застосунку, тестування в умовах, наближених до реальних, передача системи для практичного або навчального використання). |
| **8. Порядок контролю та приймання** | 1. Перевірку ходу розроблення заданих проектних документів керівнику кваліфікаційної роботи здійснювати 1 раз на 3 тижні.<br>2. Випробування програмного виробу відповідно до Програми та методики випробувань.<br>3. Захист розробленого програмного виробу провести на засіданні ДЕК.<br>4. Розроблені проектні документи подати в електронному вигляді. |

Виконавець
Литвин Андрій Віталійович

Замовник
Чуб Ольга Ігорівна

**Додаток В**

**Програма та методика випробувань програмного виробу**
**«ВЕБЗАСТОСУНОК ОПТИМІЗАЦІЇ РАЦІОНУ ХАРЧУВАННЯ СТУДЕНТА НА ОСНОВІ ЛІНІЙНОГО ПРОГРАМУВАННЯ «DIETOPT»»**

**1. Об'єкт випробувань**

**1.1 Найменування програмного виробу:**

Вебзастосунок оптимізації раціону харчування студента на основі лінійного програмування «DietOpt».

**1.2 Область застосування:**

Вебзастосунок «DietOpt» належить до класу інформаційних систем у сфері HealthTech. Система застосовується для автоматизованого складання збалансованого та мінімального за вартістю добового раціону харчування з урахуванням антропометричних параметрів, фізіологічних норм і бюджетних обмежень користувача. Основна цільова аудиторія — студенти вищих навчальних закладів України.

**1.3 Умовне призначення розробки:**

Програмний виріб призначений для автоматизованого формування оптимального за вартістю добового раціону харчування на основі даних користувача (стать, вік, маса, зріст, рівень активності, мета, бюджет), а також для генерації персоналізованих рецептів за допомогою Google Gemini AI.

**2. Мета випробувань**

Метою випробувань є перевірка відповідності функціональних характеристик вебзастосунку вимогам Технічного завдання: коректності розрахунку норм КБЖУ, правильності оптимізації раціону засобами PuLP/CBC, роботи фільтрів алергенів та дієтичних обмежень, генерації рецептів Gemini AI, а також надійної обробки некоректних вхідних даних.

**3. Загальні положення**

3.1 Підстава для проведення випробувань: наказ ХНУ імені В.Н. Каразіна **№ 4101-5/1245 від 28.05.2026** року про затвердження тем кваліфікаційних робіт бакалаврів спеціальності 123 «Комп'ютерна інженерія».

3.2 Приймальні випробування проводяться на базі комп'ютерного класу кафедри комп'ютерних систем та робототехніки за участю Замовника та Виконавця.

**4. Вимоги до програмного виробу**

- Автоматизований розрахунок норм КБЖУ за формулою Міффліна-Сан Жеора з урахуванням коефіцієнта PAL та мети користувача.
- Оптимізація раціону: мінімізація вартості при дотриманні обмежень на калорійність (±10 %), КБЖУ (±20 %) та бюджет.
- Фільтрація продуктів за алергенами, типом дієти (стандартна/вегетаріанська/веганська) та виключеними позиціями.
- Порівняльний аналіз оптимізованого раціону з типовим (80–100 % від бюджету).
- Генерація персоналізованих рецептів засобами Google Gemini AI.
- Валідація вхідних даних з поверненням HTTP 422 при порушенні обмежень.
- Час виконання оптимізації — не більше 500 мс для стандартного набору продуктів.

**5. Вимоги до програмної документації**

До виробу «DietOpt» надається така документація: Технічне завдання (Додаток Б), Програма та методика випробувань (Додаток В), опис програмного виробу (розділи пояснювальної записки), текст програми (Додаток Г).

**6. Засоби та порядок випробувань**

**6.1 Засоби випробувань:**

Технічні засоби: ноутбук ASUS VivoBook S5402ZA, Intel Core i7-12700H, 16 ГБ RAM, ОС Windows 11.

Програмні засоби: Python 3.11, FastAPI, PuLP/CBC, SQLite 3, Google Gemini AI API, Swagger UI, HTTPie, браузери Chrome/Firefox.

**6.2 Порядок проведення випробувань:**

1-й підетап. Перевірка комплектності програмної документації та відповідності технічних засобів вимогам середовища виконання.

2-й підетап. Перевірка якості документації на відповідність ДСТУ ГОСТ 19.301-79 ЄСПД.

3-й етап. Функціональне випробування програмного виробу за 7 тестовими сценаріями.

**Тест 1. Розрахунок норм КБЖУ**

Методика: у вебінтерфейсі вводиться профіль — чоловік, 20 р., 75 кг, 178 см, помірна активність, мета — підтримка ваги, бюджет 200 грн. Натискається «Розрахувати норми КБЖУ».

Очікуваний результат: BMR = 1807.5 ккал, TDEE = 2801.6 ккал, BMI = 23.7 (норма).

Фактичний результат: BMR = 1807.5 ккал, TDEE = 2801.6 ккал, BMI = 23.7. Значення відповідають очікуваним.

Висновок: вимога щодо розрахунку норм КБЖУ виконана.

**Тест 2. Оптимізація раціону (стандартна дієта, budget = 200 грн)**

Методика: на кроці 3 вебінтерфейсу натискається «Оптимізувати раціон». Перевіряються статус відповіді, вартість, калорійність та вміст КБЖУ.

Очікуваний результат: статус 'optimal', вартість $\leq$ 200 грн, калорійність 2521–3082 ккал (±10 %), КБЖУ в нормах ADA ±20 %, не менше 6 продуктів.

Фактичний результат: статус 'optimal', вартість 87–135 грн, 9–13 продуктів, час розв'язання 78–120 мс.

Висновок: вимога щодо оптимізації раціону виконана.

**Тест 3. Фільтрація за алергією «молоко»**

Методика: у полі алергій вводиться «молоко». Запускається оптимізація. Перевіряється відсутність молочних продуктів у раціоні.

Очікуваний результат: жоден продукт категорій dairy, cheese та молоковмісних назв не включений.

Фактичний результат: молочні продукти відсутні. Статус — 'optimal'.

Висновок: вимога щодо фільтрації алергенів виконана.

**Тест 4. Вегетаріанська дієта**

Методика: обирається тип харчування «Вегетаріанське». Перевіряється відсутність м'яса, птиці, риби та ковбас.

Очікуваний результат: жоден продукт категорій meat, poultry, fish_seafood, sausages не включений.

Фактичний результат: обрано вегетаріанські шаблони. М'ясних та рибних продуктів не виявлено.

Висновок: вимога щодо вегетаріанської дієти виконана.

**Тест 5. Некоректні вхідні дані (вік = 5 років)**

Методика: через Swagger UI надсилається запит до /api/calculate-norms з параметром age = 5.

Очікуваний результат: HTTP 422 з описом порушеного обмеження (ge=14).

Фактичний результат: HTTP 422, повідомлення «value is greater than or equal to 14».

Висновок: вимога щодо валідації вхідних даних виконана.

**Тест 6. Бюджет нижче мінімуму (45 грн)**

Методика: надсилається запит із budget = 45 грн.

Очікуваний результат: HTTP 422, порушення обмеження ge=50.

Фактичний результат: HTTP 422, повідомлення про мінімальний бюджет.

Висновок: вимога щодо перевірки бюджетного мінімуму виконана.

**Тест 7. Недосяжна задача (infeasible)**

Методика: budget = 60 грн, виключено: «молоко, горіхи, риба, м'ясо». Запускається оптимізація.

Очікуваний результат: статус 'infeasible' з рекомендаціями збільшити бюджет або зменшити обмеження.

Фактичний результат: система активувала резервний алгоритм run_basic_optimization. Повідомлення з рекомендаціями відображено.

Висновок: вимога щодо стійкості при недосяжній задачі виконана.

*Таблиця 1*

**Зведена таблиця результатів функціонального тестування**

| № | Тестовий сценарій | Очікуваний результат | Статус |
|---|---|---|---|
| 1 | Розрахунок норм КБЖУ | BMR, TDEE, BMI — коректні значення | Пройдено |
| 2 | Оптимізація раціону (budget=200 грн) | Статус optimal, вартість ≤ 200 грн | Пройдено |
| 3 | Фільтрація алергії «молоко» | Відсутність молочних продуктів | Пройдено |
| 4 | Вегетаріанська дієта | Відсутність м'яса, риби, птиці | Пройдено |

**Висновок за результатами проведених випробувань**

Програмний виріб вважається таким, що пройшов випробування, якщо успішно виконані всі Тести 1–7. Тестування підтвердило коректну роботу всіх функціональних модулів системи «DietOpt»: розрахунку КБЖУ, оптимізації раціону (середній час 87.4 мс), фільтрації алергенів, підтримки вегетаріанської дієти, валідації вхідних даних та обробки недосяжних задач. Відповідність фізіологічним нормам КБЖУ становить 90–100 %. Середня щоденна економія — 54.7 % від бюджету.

Рекомендується проводити подальше тестування при розширенні бази продуктів харчування та інтеграції з актуальними локальними цінами.

Виконавець: Литвин Андрій Віталійович _____