МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: Інтелектуальний чат-бот послуг для підтримки здорового способу життя

Здобувача вищої освіти другого (магістерського) рівня,
ІІ курсу, КМІ-24-1 групи, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючісистеми і технології в економіці»

Гулевич
(підпис)

Гулевича Олексія Олексійовича
(прізвище, ім'я, по батькові)

Науковий керівник

(підпис)

Грищенко Світлана Миколаївна, канд.пед.наук, ст. досл.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми

(підпис)

Омельчук Антон Анатолійович, канд.техн.наук
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри протокол від «02» січня 2026 № 10

Завідувач кафедри

(підпис)

Філоненко Михайло Миколайович, канд. фіз.-мат. наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2026

Реєстрація №15 20.01.2026 (дата) (підпис відповідальної особи)

Результати перевірки до захисту (до захисту, на доопрацювання) 02.01.2026 (дата) (підпис)

Результат захисту 5 (відмінно) (оцінка) 10.02.2026 (дата) (підпис)

Голова комісії: (підпис) О.С. Дюбанов

Члени комісії:

(підпис) А.Ю. Горбовий

(підпис) М.М. Філоненко

(підпис) І.Д. Погорєловська

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст кваліфікаційної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	**від 0 до 30**	28		10.02.2026 Дюбанов О.С. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	**від 0 до 10**	9		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	**від 0 до 5**	5		
4. Рецензент	**від 0 до 5**	5		
РАЗОМ БАЛІВ	**від 0 до 50**	47		

Оцінювання кваліфікаційної роботи науковим керівником - 46 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
93	A	5	10.02.2026	Дюбанов О.С. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Інформаційні управляючі системи і технології в економіці»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«11» грудня 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Гулевичу Олексію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи Інтелектуальний чат-бот послуг для підтримки здорового способу життя
Затверджена наказом ректора від «10» грудня 2024 № 1852
2. Строк здачі закінченої роботи «02» січня 2026
3. Мета роботи Проєктування та програмна реалізація інтелектуального чат-бота для підтримки здорового способу життя з метою автоматизації моніторингу фізіологічних показників користувача та надання персоналізованих рекомендацій.
Об'єкт дослідження Процес підтримки здорового способу життя людини з використанням цифрових технологій.
Предмет дослідження Методи, моделі та програмні засоби розробки інтелектуального чат-бота для трекінгу фізіологічних показників та надання персоналізованих рекомендацій на основі технологій обробки природної мови.
Орієнтовна база дослідження Інтелектуальний чат-бот послуг з підримки здорового способу життя.
4. Вихідні дані до роботи Методи та підходи до розробки та створений інтелектуальний чат-бот.
5. Перелік графічного матеріалу Структурна схема системи, UML-діаграма станів та діяльності для моделювання логіки діалогу, ER-діаграма бази даних, результати оцінки точності NLU-моделі та юзабіліті-тестування інтерфейсу.
6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
Вступ
Розділ 1. Огляд та аналіз предметної області
1.1. Сучасні технології у галузі підтримки здорового способу життя
1.2. Тенденції в розробці чат-ботів та рекомендаційних систем
Розділ 2. Проєктування та архітектура чат-бота
2.1. Системний аналіз та архітектурна організація комплексу
2.2. Проєктування логіки функціонування та інтелектуальних компонентів
Розділ 3. Дослідження якості та зручності використання (юзабіліті) системи
Висновки
Список використаних джерел
Додатки

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.12.2024	До 15.12.2024
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	28.05.2025	28.05.2025
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	08.09.2025	08.09.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	17.11.2025	17.11.2025
5	**Висновки**	08.12.2025	08.12.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	22.12.2025	22.12.2025
7	Отримання відгуку керівника	20.01.2026	20.01.2026
8	Отримання зовнішньої рецензії	23.01.2026	23.01.2026

8. Дата видачі завдання «11» грудня 2024

9. Керівник кваліфікаційної роботи

(підпис) Грищенко С.М
(прізвище, ініціали)

10. Керівник освітньо-професійної програми

(підпис) Омельчук А.А.
(прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти

Гулевич (підпис) Гулевич О.О.
(прізвище, ініціали)

АНОТАЦІЯ

Інтелектуальний чат-бот послуг для підтримки здорового способу життя

Мета роботи полягає у проєктуванні та програмній реалізації інтелектуального чат-бота для підтримки здорового способу життя.

Завданнями роботи є:

- аналіз предметної області та риноук mHealth, виявлення недоліків існуючих аналогів;
- проектування архітектури чат-бота та структури бази даних, обґрунтування вибіру технологічного стека (Python, Rasa, PostgreSQL);
- розробка програмного модуля системи;
- тестування розробленого комплексу, оцінка точності роботи NLU-моделі та перевірка рівня юзабіліті;
- визначення ефективності впровадження системи.

Предметом дослідження є методи, моделі та програмні засоби розробки інтелектуального чат-бота.

Об'єктом дослідження є процес підтримки здорового способу життя людини з використанням цифрових технологій.

Базою дослідження виступає інтелектуальний чат-бот послуг з підримки здорового способу життя.

За результатами дослідження сформульовано концептуальні засади та етапи розробки інтелектуального чат-бота, що базуються на методах обробки природної мови (NLP) та системному моделюванні процесів підтримки здорового способу життя.

Одержані результати можуть бути використані для вдосконалення архітектури існуючих mHealth-рішень, зокрема в частині покращення точності розпізнавання природномовних запитів та підвищення рівня залученості користувачів у процеси самоконтролю.

Кваліфікаційна робота містить 61 сторінку, 8 рисунків, 4 таблиці, список використаних джерел з 51 найменування, 4 додатки.

Ключові слова: цифрова трансформація охорони здоров'я, mHealth, інтелектуальний чат-бот, обробка природної мови (NLP), Rasa Framework, Python, архітектура діалогових систем, гейміфікація, юзабіліті-тестування (SUS).

Рік виконання кваліфікаційної роботи 2025

Рік захисту роботи 2026

ABSTRACT

Intelligent Chatbot Services for Healthy Lifestyle Support

The purpose of the work is the design and software implementation of an intelligent chatbot for supporting a healthy lifestyle.

The tasks of the work are as follows:

- analysis of the subject area and the mHealth market, identifying the shortcomings of existing analogues;
- designing the chatbot architecture and database structure, justifying the choice of the technology stack (Python, Rasa, PostgreSQL);
- development of the system software modules;
- testing the developed complex, evaluating the accuracy of the NLU model, and checking the usability level;
- determining the efficiency of the system implementation.

The subject of the research is he methods, models, and software tools for the development of an intelligent chatbot.

The object of the research is the process of supporting a healthy human lifestyle using digital technologies.

The basis of the research is an intelligent chatbot for healthy lifestyle support services.

The results of the research formulate the conceptual foundations and development stages of an intelligent chatbot based on Natural Language Processing (NLP) methods and system modeling of healthy lifestyle support processes.

The results obtained can be used to improve the architecture of existing mHealth solutions, particularly in terms of enhancing the accuracy of natural language query recognition and increasing the level of user engagement in self-control processes.

The qualification work contains 61 pages, 8 figures, 4 tables, a list of references with 51 items, 4 appendices.

Keywords: digital health transformation, mHealth, intelligent chatbot, Natural Language Processing (NLP), Rasa Framework, Python, dialogue system architecture, gamification, usability testing (SUS).

Year of qualification work: 2025

Year of work defense: 2026

ЗМІСТ

ВСТУП..... 3
РОЗДІЛ 1. ОГЛЯД ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ..... 7
1.1. Сучасні технології у галузі підтримки здорового способу життя 7
1.2. Тенденції в розробці чат-ботів та рекомендаційних систем 16
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА АРХІТЕКТУРА ЧАТ-БОТА 25
2.1. Системний аналіз та архітектурна організація комплексу 25
2.2. Проєктування логіки функціонування та інтелектуальних компонентів..... 35
РОЗДІЛ 3. ДОСЛІДЖЕННЯ ЯКОСТІ ТА ЗРУЧНОСТІ ВИКОРИСТАННЯ (ЮЗАБІЛІТІ) СИСТЕМИ..... 45
ВИСНОВКИ 52
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 55
ДОДАТКИ..... 62

ВСТУП

Актуальність теми дослідження. Сучасна охорона здоров’я глобально переходить від реактивної моделі лікування до превентивної. За даними ВООЗ, хвороби, пов'язані зі способом життя, спричиняють понад 70% смертності у світі, що робить модифікацію поведінки та самоконтроль ключовими інструментами збереження здоров'я.

Водночас виникає об’єктивна суперечність: з одного боку, спостерігається стрімкий розвиток ринку mHealth та зростання попиту на цифрові інструменти, а з іншого – традиційні фітнес-додатки демонструють низький рівень утримання користувачів. Причиною цього є складність інтерфейсів, необхідність ручного введення даних та відсутність мотивації, що призводить до швидкої втрати інтересу з боку аудиторії.

Вирішенням цієї суперечності є впровадження інтелектуальних чат-ботів. Завдяки використанню NLP та машинного навчання, такі системи забезпечують природну взаємодію, розуміють контекст та надають персоналізовані рекомендації в режимі 24/7, імітуючи спілкування з фахівцем.

Для України розробка таких рішень набуває особливого значення в умовах воєнного стану. Високий рівень стресу та обмежений доступ до медичної інфраструктури формують гостру потребу в доступних та автоматизованих інструментах підтримки. Таким чином, створення вітчизняного інтелектуального чат-бота є своєчасним і науково значущим завданням, спрямованим на покращення якості життя населення.

Теоретико-методологічну основу дослідження складають наукові розробки вітчизняних та зарубіжних учених, які вивчали проблематику цифрової трансформації сфери охорони здоров’я та впровадження технологій здорового способу життя.

Зокрема, соціально-управлінські та організаційні аспекти цифровізації медицини, питання розвитку телемедицини та її впливу на якість життя населення ґрунтовно висвітлено у працях О. Адамовича, С. Квітки,

М. Коцаренка та М. Миргородської. Технічний базис роботи, що охоплює еволюцію чат-ботів, функціонування великих мовних моделей (LLM) та мультимодального штучного інтелекту, досліджували Дж. Акоста, К. Ванг, Дж. Вейценбаум, Дж. Девлін, А. Крупа та М. Хаді.

Окрему увагу в науковій літературі приділено інструментальним засобам моніторингу: роль фітнес-технологій, потенціал носимих пристроїв та споживчі властивості смарт-гаджетів у контексті формування здорового способу життя молоді проаналізовано у доробках І. Баранової, С. Бреусової, О. Гончаренка, Дж. Лонгіні, А. Приймака та Т. Фергюсона. Питання практичної результативності таких рішень, зокрема клінічна ефективність цифрових інтервенцій, вплив гейміфікації та критична проблема утримання користувачів, розкриті в роботах А. Аггарвала, М. Ештевана, А. Мазеаса, А. Мьоннінгхоффа та А. Пратапа. Водночас специфіку застосування цифрових інструментів для надання психологічної підтримки, модифікації поведінкових звичок та точної діагностики обґрунтовано у дослідженнях М. Гуаш-Ферре, М. Житинської, Р. Рушендера та Т. Хіросави.

Важливе значення для формування нормативно-правового та стратегічного підґрунтя дослідження мають програмні документи та рекомендації міжнародних інституцій – ВООЗ, Європейської комісії, ЮНЕСКО та ОЕСР.

Метою кваліфікаційної роботи є проєктування та програмна реалізація інтелектуального чат-бота для підтримки здорового способу життя з метою автоматизації моніторингу фізіологічних показників користувача та надання персоналізованих рекомендацій.

Для досягнення мети необхідно вирішити такі **завдання**:

1. Проаналізувати предметну область та ринок mHealth, виявити недоліки існуючих аналогів та сформувати вимоги до системи.
2. Спроєктувати архітектуру чат-бота та структуру бази даних, обґрунтувавши вибір технологічного стека (Python, Rasa, PostgreSQL).
3. Розробити програмні модулі системи: розпізнавання інтенцій та сутностей (NLP), розрахунок нутрієнтів, сканування штрих-кодів та ведення щоденника активності.

4. Провести тестування розробленого комплексу, оцінити точність роботи NLU-моделі та перевірити рівень юзабіліті.

5. Визначити ефективність впровадження системи та окреслити перспективи її подальшого розвитку.

Об’єктом кваліфікаційної роботи є процес підтримки здорового способу життя людини з використанням цифрових технологій.

Предметом дослідження є методи, моделі та програмні засоби розробки інтелектуального чат-бота для трекінгу фізіологічних показників та надання персоналізованих рекомендацій на основі технологій обробки природної мови.

Методи дослідження. Для розв’язання поставлених завдань використано комплекс загальнонаукових та спеціальних методів:

- ✓ теоретичні: аналіз літератури та нормативної бази – для визначення засад цифрової підтримки здоров’я; порівняльний аналіз аналогів – для виявлення недоліків і формування вимог до системи;
- ✓ спеціальні: системний аналіз (проєктування архітектури); методи NLP та машинного навчання (обробка запитів користувача); математичне моделювання (алгоритми розрахунку фізіологічних показників); об’єктно-орієнтоване проєктування (розробка програмних модулів);
- ✓ емпіричні: експериментальне тестування (перевірка інтеграцій та точності роботи); анкетування за методикою SUS (оцінка юзабіліті); методи математичної статистики (обробка результатів).

Базою дослідження виступає інтелектуальний чат-бот послуг з підримки здорового способу життя.

Наукова новизна одержаних результатів полягає в удосконаленні методів людино-машинної взаємодії у системах моніторингу здоров’я шляхом поєднання технологій обробки природної мови та комп’ютерного зору. Це дозволило реалізувати гібридну модель діалогу, яка, на відміну від існуючих аналогів, забезпечує мультимодальне введення даних (текст, штрих-код) та адаптивну підтримку користувача.

Практичне значення роботи визначається створенням повнофункціонального інтелектуального чат-бота на платформі Telegram, який інтегрує математичні алгоритми розрахунку фізіологічних норм із зовнішньою базою даних OpenFoodFacts. Розроблений програмний продукт є готовим інструментом для автоматизованого контролю харчування та фізичної активності, що може бути використаний для профілактики неінфекційних захворювань.

Особистий внесок здобувача. Усі результати, що викладені в роботі, отримані автором самостійно. Здійснено аналіз предметної області, спроєктовано архітектуру системи та базу даних, розроблено програмний код чат-бота мовою Python, налаштовано NLU-модель та проведено тестування системи.

Публікації. За темою кваліфікаційної роботи опубліковано 5 одноосібних публікацій, 1 у співавторстві у збірниках матеріалів науково-практичних конференцій та 2 одноосібні статті в науковому періодичному електронному виданні «Universum».

Результати дослідження висвітлено на XVIII Міжнародній науково-практичній конференції «Інформаційні технології і автоматизація – 2025» (30-31жовтня 2025 р., м. Одеса), Міжнародній науково-практичній конференції здобувачів вищої освіти та молодих вчених «Економіка і бізнес:нові реалії, технологічні тренди, цифрові інновації» (15 травня 2025 р., Львів – Ірпінь), XVI Всеукраїнській науково-практичній конференції для молодих учених та здобувачів освіти «Сучасні інформаційні технології в освіті і науці» (10-11 квітня 2025 р., м. Умань), XXV Всеукраїнській науково-технічній конференції молодих вчених, аспірантів та студентів «Стан, досягнення і перспективи інформаційних систем і технологій» (17-18 квітня 2025 р., м. Одеса), V Науково-практичному форумі «Цифрова трансформація фінансового сектору» (30 квітня 2025 р., м. Ірпінь), XIII Міжнародній науково-практичній конференції, присвяченої 30-річчю Харківського національного університету внутрішніх справ у статусі університету «Сучасні проблеми правового, економічного та соціального розвитку держави (22 листопада 2024 р., м. Вінниця), XVII Міжнародній науково-практичній конференції «Інформаційні технології і автоматизація – 2024» (31жовтня-1 листопада 2024 р., м.Одеса).

РОЗДІЛ 1. ОГЛЯД ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Сучасні технології у галузі підтримки здорового способу життя

Сучасні технології відіграють важливу роль у підтримці здорового способу життя, надаючи інструменти для моніторингу, аналізу та покращення фізичного і психічного стану. Цифрові рішення дозволяють отримувати персоналізовані поради, попереджати хвороби, контролювати сну, стрес, харчування та фізичну активність, що сприяє розвитку культури здоров'я і підвищенню мотивації до змін.

Здоровий спосіб життя (далі – ЗСЖ) постає як міждисциплінарний об'єкт дослідження, що інтегрує напрацювання медицини, поведінкових наук, інформатики та інженерії. Упродовж 2020–2025 рр. відбулося істотне зростання ринку мобільної медицини (mHealth), цифрових терапій (DTx), носимих сенсорних пристроїв і розмовних агентів, орієнтованих на підтримку корисних звичок у сферах харчування, фізичної активності, сну та психоемоційного благополуччя[44; 50; 51]. Вказані рішення дедалі частіше використовують методи штучного інтелекту (ШІ), зокрема глибинне навчання та великі мовні моделі (Large Language Models, LLM), що дає змогу забезпечити персоналізацію рекомендацій, мотиваційний коучинг і природномовну взаємодію з користувачем[31; 35; 38].

Сучасне розуміння здоров'я охоплює не лише фізичний, але й соматичний, психоемоційний, духовний, інтелектуальний і соціальний аспекти, що спільно формують феномен ЗСЖ. Вітчизняна наука розглядає здоров'я як динамічний процес. Зокрема, у сучасній літературі[цит. за 1] широко використовується визначення В. Войтенка, згідно з яким здоров'я – це «здатність пристосовуватись до середовища... протистояти старінню... та розширювати свої можливості». Такий підхід ідеально корелює з метою створення чат-бота, який допомагає користувачеві саме «розширювати можливості» контролю над власним організмом.

Варто зазначити, що у вітчизняній науковій думці здоровий спосіб життя розглядається не як статика, а як динамічний процес. Зокрема, А. Приймак та О. Гончаренко[25] визначають його як усвідомлену діяльність особистості,

спрямовану на збереження здоров’я та виконання соціальних функцій. Автори наголошують, що стрижнем цієї діяльності для сучасної молоді виступають фітнес-технології, які дозволяють інтегрувати рухову активність у повсякденний графік студента.

Попри суспільну значущість формування ЗСЖ, ефективність фізкультурно-оздоровчих заходів залишається недостатньою: фізична активність охоплює обмежені групи населення, що збільшує ризики неінфекційних захворювань (серцево−судинні хвороби, діабет, ожиріння). Цифрові технології здатні підвищити залученість і ефективність профілактичних практик, інтегруючись у повсякденність користувачів.

Авторитетні організації – Європейська комісія, ЮНЕСКО, ОЕСР і ВООЗ – у своїх дослідженнях та рекомендаціях визначають основні напрямки розвитку та впровадження інновацій у сфері медицини та здоров'я[3; 12; 22; 26]. Ці організації не лише займаються розробкою глобальних стандартів, але й активно впливають на формування політики в охороні здоров'я, що робить їхнє бачення важливим для оцінки поточних тенденцій та прогнозів щодо впливу цифрових технологій на здоров’я населення.

Європейська комісія підкреслює потенціал цифрових технологій для покращення добробуту, ефективнішого надання послуг, підтримки ЗСЖ та самостійності громадян[12]. ЮНЕСКО акцентує важливість ШІ та цифрових інструментів, що розширюють автономію людей у контролі над здоров’ям і формуванні здорової поведінки через персоналізовані рекомендації та зворотний зв’язок[26]. ОЕСР відзначає, що інновації у сфері цифрового здоров’я (мобільні застосунки, носимі пристрої, віртуальні консультації) змінюють підходи до профілактики та раннього втручання, роблячи їх більш доступними[22]. ВООЗ визначає цифрове здоров’я як використання технологій для підвищення ефективності медичної допомоги та підтримки ЗСЖ – від мобільних застосунків і носимих пристроїв до телемедицини, ШІ та «великих даних»[2].

Цифрові технології підтримки здорового способу життя (ЗСЖ), що охоплюють мобільні застосунки, носимі пристрої, дистанційний моніторинг та телемедичні сервіси, в Україні на законодавчому рівні регламентуються

переважно загальними нормами у сферах охорони здоров'я, захисту персональних даних та електронної системи охорони здоров'я (ЕСОЗ). Спеціалізованого законодавства, спрямованого виключно на «wellness»-технології або профілактичні цифрові інструменти, поки що не існує. Відтак, такі технології регулюються залежно від їх функціонального призначення, сфери використання та способу обробки даних.

Згідно із Законом України «Про охорону здоров'я», цифрові інструменти можуть розглядатися як частина електронної системи охорони здоров'я або як елемент телемедичних послуг. Стаття 35-1 визначає, що ЕСОЗ є комплексною телекомунікаційною системою для обробки медичної інформації, яка «забезпечує автоматизований збір, зберігання, обробку, використання, захист та поширення інформації про стан здоров'я населення»[15].

Будь-які цифрові рішення, що зберігають або обробляють дані про стан здоров'я користувачів, підпадають під дію Закону України «Про захист персональних даних». Згідно зі статтею 7, персональні дані про здоров'я належать до особливих (чутливих) категорій, і їх обробка дозволяється лише за наявності письмової згоди суб'єкта або в інших випадках, визначених законодавством[14].

В окремих випадках програмне забезпечення, що використовується для моніторингу фізіологічних показників, оцінки стану здоров'я або прогнозування ризиків, може підпадати під визначення медичного виробу. Хоча окремий Закон «Про медичні вироби» перебуває на етапі розробки, чинним наразі є Технічний регламент щодо медичних виробів (постанова КМУ № 753 від 02.10.2013), який передбачає, що медичний виріб – це також «програмне забезпечення, призначене виробником для застосування з медичною метою»[24].

Отже, у разі, якщо мобільний додаток чи цифрова система виконує функції діагностики, лікування або моніторингу стану здоров'я з медичним призначенням, він має проходити оцінку відповідності, реєстрацію та маркування.

На рівні державної політики стратегічним документом є «Національна стратегія розвитку електронної охорони здоров'я на період до 2025 року», яка передбачає розвиток персоналізованої медицини, впровадження телемедицини,

інтеграцію мобільних додатків, пристроїв самообслуговування, а також створення електронного щоденника здоров'я[21].

Таким чином, цифрові технології підтримки ЗСЖ в Україні функціонують у межах ширших нормативних рамок електронної охорони здоров'я, захисту персональних даних і технічного регулювання медичних виробів. Залежно від функціональності і сфери застосування, окремі рішення можуть підпадати під відповідні законодавчі акти. Водночас, з огляду на стрімкий розвиток цифрових інструментів у сфері профілактики, актуальним залишається завдання створення спеціалізованих нормативних документів, що враховують специфіку застосунків «mHealth», носимих пристроїв та систем самоспостереження.

Сучасні цифрові технології відіграють ключову роль у підтримці здорового способу життя (ЗСЖ), забезпечуючи безперервний моніторинг фізичного стану, доступ до медичних консультацій, персоналізовані рекомендації та інструменти поведінкової модифікації.

Цей технологічний підхід узгоджується з позицією А. Приймака та О. Гончаренка[25], які розглядають сучасні фітнес-технології як інноваційну систему засобів, здатну ефективно вирішити проблему низької рухової активності молоді. Автори наголошують, що ключовою перевагою таких інструментів є можливість повної індивідуалізації навантажень та підвищення емоційної привабливості занять. Саме завдяки інтеграції цифрових рішень фізична активність трансформується з рутинного обов'язку у вмотивований та цікавий для користувача процес самовдосконалення.

Нижче розглянуто основні класи технологічних рішень, що мають підтверджену ефективність згідно з науковими дослідженнями.

1. Фітнес-трекери та носимі пристрої

Носимі пристрої, такі як фітнес-браслети та розумні годинники, дозволяють відстежувати фізичну активність, кількість кроків, частоту серцевих скорочень, рівень кисню в крові, якість сну тощо. Згідно з метааналізом Т. Фергюсона та співавт.[33], використання сучасних носимих пристроїв сприяє збільшенню щоденної активності на понад 2000 кроків і зниженню індексу маси тіла. Як свідчить парасольковий огляд Дж. Лонгіні та співавт.[39], використання цифрових

трекерів призводить до статистично значущого підвищення рівня фізичної активності (в середньому на 1312 кроків/день та 58 хвилин інтенсивних навантажень на тиждень). Автори підкреслюють, що отримані дані є беззаперечним доказом ефективності автоматизованого моніторингу, що обґрунтовує необхідність технологічної інтеграції чат-бота з платформами Google Fit та Apple Health для забезпечення об'єктивного контролю прогресу користувача. У зв'язку з цим А. Пратап та співавт.[44], базуючись на аналізі поведінки понад 100 тисяч учасників цифрових програм здоров'я, констатують критичну проблему низького утримання (retention): значна частина користувачів припиняє взаємодію з інтервенціями вже протягом перших тижнів, що суттєво обмежує клінічну ефективність таких рішень. Цю тезу підтверджують і вітчизняні автори, вказуючи на необхідність пошуку інструментів, що здатні утримувати увагу пацієнта тривалий час. Ефективним шляхом подолання цього бар'єру стає інтеграція програмних рішень із носимними пристроями, які забезпечують безперервний зворотний зв'язок. Як зазначають дослідники НФаУ С. Бреусова та співавт.[2], сучасна носима електроніка трансформувалася з іміджевих аксесуарів у дієві інструменти об'єктивного самоконтролю. Автори підкреслюють, що можливість моніторингу фізіологічних показників у реальному часі сприяє формуванню усвідомленого ставлення до здоров'я та мотивує користувачів до корекції поведінкових звичок на основі точних даних, а не суб'єктивних відчуттів.

2. Мобільні додатки для моніторингу здоров'я

Мобільні додатки (mHealth) дозволяють користувачам вести облік харчування, сну, фізичної активності, настрою тощо. Вони інтегрують алгоритми змін поведінки, моніторинг та персоналізовані поради. Як наголошують А. Аггарвал та співавт.[28], сучасні діалогові агенти виступають ефективним інструментом для реалізації поведінкових інтервенцій, забезпечуючи безперервний моніторинг, адаптивний зворотний зв'язок та персоналізовану мотивацію у режимі реального часу.

Науково доведено, що цифрові інструменти демонструють статистично значущий позитивний вплив на формування здорового способу життя, перевершуючи традиційні епізодичні консультації завдяки безперервності

підтримки. У сфері фізичної активності це підтверджує масштабний мета-аналіз А. Мьоннінгхоффа та співавт.[41], який охопив понад 21 тис. учасників і довів, що mHealth-інтервенції забезпечують стійке підвищення рухової активності навіть у довгостроковій перспективі. Щодо збалансованого харчування, то, за даними М. Гуаш-Ферре та співавт.[34], алгоритмічні системи прецизійного харчування дозволяють досягти кращого глікемічного контролю порівняно зі стандартними дієтологічними рекомендаціями.»

У сфері відмови від куріння потенціал цифрових інтервенцій також знаходить наукове підтвердження. Ефективність цифрових інструментів підтверджено на найвищому рівні доказовості у парасольковому огляді Р. Рушендера та співавт.[45]. Проаналізувавши 11 систематичних досліджень, автори дійшли висновку, що мобільні застосунки відіграють перспективну роль у відмові від куріння. При цьому наголошується, що максимальну результативність демонструють не прості трекери, а комплексні рішення, побудовані на наукових моделях зміни поведінки та глибокій персоналізації взаємодії з користувачем. В українському контексті важливим є аспект психічного здоров’я молоді, особливо в умовах сучасних кризових викликів. Як зазначає М. Житинська[13], цифрові інструменти сьогодні виступають ключовим каналом надання соціально-психологічної підтримки, забезпечуючи її оперативність та безбар'єрність. Авторка підкреслює, що для молодого покоління цифрова взаємодія (через чат-боти чи мобільні застосунки) має низку критичних переваг: вона гарантує анонімність, знижує стигматизацію звернення за допомогою та дозволяє отримати підтримку у гнучкому форматі «тут і зараз». Таким чином, інтеграція психологічних модулів у розроблюваний сервіс відповідає нагальним потребам користувачів, які шукають доступні та безпечні інструменти саморегуляції в умовах невизначеності.

3. Телемедицина

Телемедицина забезпечує доступ до лікарів у режимі відеозв’язку, асинхронної комунікації або мобільних сервісів, що значно розширює можливості регулярного медичного спостереження. У звіті Всесвітньої організації охорони здоров’я[51] зазначається: «телемедицина покращує доступ

до медичної допомоги та скорочує час до початку втручання, особливо для сільських або віддалених груп населення». Необхідність створення інтегрованих цифрових рішень підтверджується найновішими дослідженнями 2025 року. У оглядовій праці М. Ештевана та співавт.[32] доведено, що найбільшу клінічну цінність мають телемедичні системи, які поєднують менеджмент серцево-судинних ризиків та контроль діабету. Автори встановили, що використання мобільних платформ та AI-інструментів дозволяє одночасно покращувати показники артеріального тиску, ліпідного профілю та глікемічного контролю. Це свідчить про те, що архітектура сучасного чат-бота повинна базуватися на холістичному підході, забезпечуючи моніторинг множинних параметрів здоров'я для ефективної профілактики коморбідних станів. Цей технологічний вектор розвитку цілком узгоджується з позицією вітчизняних науковців, які розглядають телемедицину як системний інструмент трансформації галузі. Зокрема, у праці М. Коцаренка та співавт.[17] підкреслено, що сучасні телемедичні рішення виступають не лише засобом комунікації, а й дієвим механізмом оптимізації надання медичної допомоги. Автори наголошують, що для реалізації комплексного моніторингу критично важливою є глибока технологічна інтеграція та підтримка стандартів обміну медичними даними (таких як DICOM). Саме такий підхід дозволяє перейти від поверхневого консультування до повноцінної віддаленої діагностики та аналізу динаміки стану пацієнта, що значно скорочує час прийняття клінічних рішень.

4. Штучний інтелект та персоналізовані рекомендації

Штучний інтелект (ШІ) дозволяє аналізувати великі обсяги персональних даних – такі як історія захворювань, біомаркери, генетична інформація чи патерни поведінки – для побудови індивідуалізованих інтервенцій. Розвиваючи цю думку, Дж. Н. Акоста та співавт.[27] у своєму дослідженні мультимодального біомедичного ШІ зазначають, що саме інтеграція статичних клінічних даних із динамічними показниками дозволяє реалізувати модель «прецизійного здоров'я» (precision health). Такий підхід гарантує, що рекомендації не є статичними, а адаптуються під фізіологічні зміни користувача в режимі реального часу, випереджаючи розвиток патологічних станів.

Практичну ефективність такої методології підтверджують новітні дослідження. Зокрема, у програмній праці М. Гуаш-Ферре та співавт.[34] доведено, що персоналізовані стратегії харчування, побудовані на глибокому аналізі мікробіому та метаболічних реакцій, значно перевершують за результативністю традиційні універсальні рекомендації («one-size-fits-all») у контролі глікемічного статусу та профілактиці метаболічних порушень.

5. Інтернет речей (IoT) у моніторингу здоров'я

IoT-пристрої – смарт-ваги, тонометри, пульсоксиметри, монітори сну тощо – забезпечують безперервний збір даних про стан здоров'я та передачу їх у хмарні системи або мобільні застосунки. Такі технології часто використовуються для тривалого самоспостереження та раннього виявлення змін. М. Джавайд, А. Халім та співавт.[36] у своєму дослідженні технологій Medical 4.0 зазначають, що «Інтернет медичних речей» (IoMT) виступає фундаментом цієї трансформації, забезпечуючи перехід від епізодичного лікування до безперервного дистанційного моніторингу пацієнтів. Разом з тим, автори наголошують, що масова інтеграція сенсорних пристроїв створює критичні виклики у сфері кібербезпеки та приватності даних, що вимагає впровадження стандартизованих протоколів шифрування та верифікації точності вимірювань.

6. Віртуальні фітнес-платформи та онлайн-тренери

Онлайн-платформи для фізичної активності забезпечують доступ до тренувань у зручному форматі (відео, стріми, інтерактивні програми) та дозволяють індивідуалізувати навантаження. Часто інтегрується гейміфікація або соціальні функції. Ефективність ігрових механік підтверджує систематичний огляд А. Мазеаса та співавт.[40]. Дослідники встановили, що інтеграція елементів гейміфікації (таких як віртуальні нагороди, рейтингові таблиці та змагальні челенджі) у цифрові платформи здоров'я забезпечує статистично значуще підвищення рівня фізичної активності користувачів, сприяючи формуванню стійкої звички до регулярних тренувань.

Узагальнюючи аналіз різних інструментів, можна стверджувати, що сучасні цифрові технології охоплюють широкий спектр взаємопов'язаних рішень: від носимої електроніки і мобільних застосунків до телемедицини та

штучного інтелекту. Вони сприяють формуванню здорових звичок, розширюють доступ до профілактики та забезпечують персоналізовану підтримку в режимі реального часу.

Разом з тим, впровадження цих технологій має не лише технічний, а й глибокий соціальний вимір. Як зазначають С. Квітка та М. Миргородська[16], цифрова трансформація охорони здоров’я спрямована передусім на створення комфортного середовища та підвищення якості життя населення. Автори підкреслюють, що в умовах сучасного мережевого суспільства формується новий тип користувача – «цифровий пацієнт». Для нього ключовим фактором ефективності стає довіра до цифрових сервісів, яка здатна забезпечити більш продуктивні відносини між людиною та системою охорони здоров'я, компенсуючи ослаблення традиційних інститутів.

Аналіз сучасних цифрових інструментів для ЗСЖ виявив низку системних проблем, які ефективно вирішуються впровадженням інтелектуального чат-бота. По-перше, він долає фрагментованість сервісів, створюючи універсальний інтерфейс для моніторингу активності, сну та харчування в одному середовищі. По-друге, використання алгоритмів персоналізованого коучингу та гейміфікації підвищує мотивацію користувачів, запобігаючи швидкій відмові від цифрових інтервенцій. По-третє, завдяки штучному інтелекту та великим мовним моделям забезпечується глибока індивідуалізація рекомендацій на основі реальних поведінкових патернів. Крім того, природномовний інтерфейс і підтримка голосових запитів роблять технологію доступною для категорій з низькою цифровою компетентністю, а повна мовна та нормативна локалізація гарантує відповідність українському законодавству та культурному контексту. Таким чином, інтелектуальний чат-бот трансформує розрізнені цифрові рішення у цілісну, доступну та персоналізовану платформу підтримки здоров’я.

Отже, інтелектуальний чат-бот, орієнтований на підтримку здорового способу життя, може стати ефективним інструментом об’єднання цифрових сервісів у персоналізовану, доступну та інтерактивну платформу. Його впровадження сприятиме подоланню фрагментації цифрового простору ЗСЖ,

підвищенню залученості користувачів та розширенню доступу до профілактичних послуг у межах сучасної системи охорони здоров’я.

1.2. Тенденції в розробці чат-ботів та рекомендаційних систем

У сучасному світі, коли технології штучного інтелекту (ШІ) та автоматизації розвиваються з неймовірною швидкістю, інтелектуальні чат-боти стають важливим інструментом у багатьох сферах життя, використовуючись для автоматизації комунікацій, підтримки клієнтів, оптимізації бізнес-процесів, а також у галузях медицини, освіти та державного управління. Актуальність дослідження розвитку інтелектуальних чат-ботів обумовлена не лише їхнім широким впровадженням, а й постійним удосконаленням технологій, які дозволяють підвищити їхню ефективність та функціональність. Сучасні чат-боти здатні виконувати складні завдання, інтегруватися з іншими цифровими системами, забезпечувати зручну та швидку взаємодію з користувачами, відкриваючи нові можливості для бізнесу та державних установ, зокрема у сфері обслуговування клієнтів, оптимізації робочих процесів та забезпечення доступу до важливої інформації.

Виникнення чат-ботів можна вважати частиною реалізації бачення Алана Тюрінга про інтелектуальні машини, яке він представив ще в 1950-х роках у своїй знаменитій статті «Computing Machinery and Intelligence»[47]. Тюрінг запропонував так званий «Тест Тюрінга», який став основою для оцінки здатності машини до розумного (інтелектуального) поводження. Він припустив, що якщо машина може так успішно імітувати людину в розмові, що спостерігач не зможе відрізнити її від живої особи, то така машина може вважатися «інтелектуальною». Ідея тесту полягала в тому, щоб оцінити, чи може машина успішно взаємодіяти з людиною за допомогою текстового чату, зберігаючи таємницю своєї «машинної» природи, що стало основою для створення сучасних інтелектуальних чат-ботів.

Першим реальним кроком у напрямку створення чат-ботів став ELIZA (1966 р.), розроблений Джозефом Вейценбаумом у Массачусетському технологічному інституті. ELIZA була програмою, яка могла імітувати терапевта, виконуючи прості шаблонні заміни фраз[49]. Вона мала обмежений

рівень «інтелекту», але вже тоді показала, що комп'ютери можуть симулювати людське спілкування в певних межах, що стало основою для подальших досліджень в області штучного інтелекту та чат-ботів.

Пізніше, у 1970-х роках, з'явився чат-бот PARRY, який міг імітувати поведінку людини з шизофренією, розширивши можливості симуляції психоемоційних станів у комп'ютерних програмах[48]. PARRY був здатний вести прості розмови, що набагато більше імітувало людську взаємодію в порівнянні з ELIZA.

Кінець 1990-х років характеризується появою комерційних чат-ботів. Одним із таких ботів став SmarterChild, який використовувався в чатах на платформах AOL Instant Messenger і MSN Messenger. SmarterChild міг виконувати прості запити, такі як надання інформації про погоду чи новини, і став популярним серед користувачів. Його можливості стали основою для розвитку більш складних чат-ботів у майбутньому[46].

На початку 21-го століття з'явилися більш складні віртуальні помічники, такі як Siri від Apple, Google Assistant, Amazon Alexa. Вони стали інтегрованими у мобільні пристрої та мали можливість голосового введення, що дозволило чат-ботам ще більше наблизитися до реальної людської взаємодії. Штучний інтелект цих систем почав використовувати передові алгоритми для обробки природної мови, що дозволило значно покращити здатність чат-ботів до розуміння і генерації тексту.

Одним із найбільш революційних досягнень у розвитку чат-ботів став GPT-3 від OpenAI, який здатний генерувати тексти, вести осмислені бесіди та навіть вирішувати складні завдання за допомогою передових алгоритмів глибокого навчання[42].

Отже, ці технології свідчать про те, що чат-боти розвиваються від простих інструментів до потужних систем штучного інтелекту, які можуть виконувати складні функції в різних сферах діяльності. Чат-боти є одним з найважливіших інструментів, які втілюють принципи штучного інтелекту, застосовані до взаємодії між людиною та машиною. Вони з'явилися як результат прогресу в галузі обробки природної мови, машинного навчання та автоматизації процесів.

А. Крупа, розглядаючи технологію чат-боту як чинник комп'ютерно-посередницької комунікації цифрового суспільства, визначає його як «програму-співрозмовник, що імітує людське спілкування за допомогою тексту чи голосу, вони допомагають автоматизувати завдання, працюючи за заданим алгоритмом, ведуть діалог із користувачем, виконуючи його прохання, відповідають на запити або розважають своїми відповідями»[18].

Згідно з визначенням К. Ванга[48], чат-боти – це інтелектуальні програмні системи, що використовують алгоритми машинного навчання та обробки природної мови (NLP) для симуляції людської комунікації через текстові або голосові інтерфейси. Автор підкреслює, що завдяки стрімкій еволюції ШІ сучасні рішення трансформувалися з простих шаблонних механізмів у складні адаптивні моделі, здатні глибоко аналізувати контекст запитів та забезпечувати високоточну, персоналізовану взаємодію з користувачем.

Чат-бот – це комп'ютерна програма, яка імітує та обробляє людську розмову (письмово чи усно), дозволяючи людям взаємодіяти з цифровими пристроями, начебто вони спілкуються з реальною людиною. Чат-боти можуть бути такими ж простими, як елементарні програми, які відповідають на простий запит однорядковою відповіддю, або настільки ж складними, як цифрові помічники, які навчаються та розвиваються для забезпечення більш високого рівня персоналізації для збору та обробки інформації[43].

Інтелектуальний чат-бот – це програмне забезпечення, яке використовує технології штучного інтелекту (ШІ), машинного навчання (ML) і обробки природної мови (NLP) для симулювання розмови з користувачем. На відміну від простих чат-ботів, інтелектуальні чат-боти можуть вести більш складні, осмислені діалоги, розуміти контекст запитів та надавати персоналізовані відповіді.

Оскільки інтелектуальні чат-боти стають важливим інструментом для автоматизації комунікацій, підтримки клієнтів і оптимізації бізнес-процесів, їхній попит постійно зростає. Це зумовлено як технологічними інноваціями, так і змінами в потребах ринку, зокрема, через підвищення вимог до оперативності та персоналізації взаємодії з користувачами. Враховуючи все більш активне використання чат-ботів у різних сферах – від обслуговування клієнтів до

медичних консультацій, прогнозується значне збільшення попиту на ці технології в найближчі роки.

Попит ринку чат-ботів, за дослідженням Credence Research, у 2023 році оцінювався у 4951,2 млн доларів США, і як очікується, досягне 37 423,93 млн доларів США у 2032 році[30]. Прогнози щодо розвитку ринку чат-ботів показують, що CAGR (складний середньорічний темп зростання) впродовж наступних років буде значним, зростаючи зі середньорічним темпом зростання 25,2 % між 2024 та 2032 роками[30], що підтверджує їхню важливість та перспективність у сфері технологій і бізнесу.

Отже, розвиток інтелектуальних чат-ботів, які базуються на технологіях штучного інтелекту, обробки природної мови та машинного навчання, є важливим кроком у трансформації комунікаційних процесів. Завдяки здатності до персоналізації взаємодії та адаптації до запитів користувачів, чат-боти сьогодні займають провідне місце в автоматизації бізнес-процесів, підтримці клієнтів та інших сферах. Зростаючий попит на ці технології, а також їхня інтеграція з іншими цифровими системами, свідчать про їхнє значення для подальшого розвитку різних галузей. Прогнози щодо зростання ринку чат-ботів підтверджують високий потенціал цих систем у майбутньому.

У сфері цифрової підтримки здорового способу життя активно розвиваються два ключові напрямки: 1) чат-боти, орієнтовані на взаємодію з користувачем; 2) рекомендаційні системи, що забезпечують персоналізовані поради на основі даних. У сучасних рішеннях ці компоненти часто інтегруються в гібридні платформи з підтримкою штучного інтелекту.

Чат-боти умовно поділяються на інформаційні, навчальні, мотиваційні, діалогові та гібридні. Інформаційні чат-боти надають користувачеві структуровану довідкову інформацію (про симптоми, ліки, послуги, режим харчування тощо). Працюють за сценарієм «запит – відповідь». Навчальні чат-боти спрямовані на формування здорових звичок через мікронавчання, пояснення та вправи. Часто використовуються у сфері психоосвіти або фітнес-освіти. Мотиваційні чат-боти зосереджені на підтримці користувача в досягненні цілей: нагадують, надихають, фіксують прогрес. Використовують техніки

позитивного підкріплення. Діалогові чат-боти мають розвинену діалогову модель з елементами емоційного резонансу. Імітують розмову з людиною, можуть «вести» бесіду в режимі вільного тексту. Гібридні чат-боти поєднують кілька функцій: інформування, мотивацію, навчання та підтримку. Вони адаптивні, персоналізовані, часто використовують штучний інтелект.

Кожен тип має свою функціональну орієнтацію – від інформування користувача до підтримки емоційного стану та генерації персоналізованих рекомендацій. Типологія чат-ботів представлена у табл. 1.1.

Таблиця 1.1

Типологія чат-ботів для ЗСЖ

Тип бота	**Основна функція**	**Приклад / Джерело**
Інформаційні	Надання довідкової інформації, порад	WHO Health Alert (WhatsApp)
Навчальні	Освітні модулі, підвищення обізнаності	Healthily AI Bot
Мотиваційні	Коучинг, нагадування, трекінг цілей	Lark, Woebot
Діалогові	Підтримка у режимі реального часу, Q&A	Florence, Ada Health
Гібридні	Поєднання підтримки, трекінгу та порад	Replika, Wysa, Youpe

Коротка характеристика відомих рішень, що застосовуються для підтримки здорового способу життя, психічного благополуччя та первинної медичної допомоги представлено у Додатку А.

Рекомендаційні системи в контексті ЗСЖ можуть бути контентно-орієнтованими, колаборативними або гібридними. Вони використовують алгоритми машинного навчання для персоналізації контенту. Прикладом є Youper, Ada Health, що формують поради на основі стану користувача.

Контентно-орієнтовані: поради формуються на основі характеристик самого користувача (вік, стать, рівень активності, цілі) та властивостей об’єктів (тип тренування, склад продукту, рівень інтенсивності). Колаборативні: рекомендації використовують досвід і вподобання інших користувачів зі схожими профілями. Система прогнозує, що може сподобатись людині на основі поведінки «схожих» людей. Гібридні рекомендації поєднують елементи контентного та колаборативного підходів, щоб підвищити точність, релевантність і персоналізацію.

У контексті чат-ботів ЗСЖ гібридні системи найчастіше реалізуються із використанням штучного інтелекту, що забезпечує динамічну адаптацію до змін у поведінці, настрої чи стані користувача.

У табл. 1.2. наведено порівняльний аналіз ключових інтелектуальних чат-ботів, які застосовуються для підтримки здорового способу життя, психічного благополуччя та первинної медичної допомоги.

Таблиця 1.2

Порівняльна таблиця відомих рішень

Чат-бот / Платформа	**Основна функція**	**Особливості**	**Цільова аудиторія**	**Технології**	**Посилання / Джерело**
Healthily (LiveHealthily)	Оцінка симптомів, поради щодо ЗСЖ	AI-опитувач, персоналізовані плани, енциклопедія	Загальне населення	NLP, decision tree AI	https://www.livehealthily.com
Florence	Нагадування про ліки, моніторинг ЗСЖ	Працює у Facebook Messenger, Telegram, Skype	Пацієнти з хронічними станами	Chatbot API, інтеграції	https://florence.chat
Wysa	Психічне здоров'я	CBT-методи, анонімне консультування	Підлітки, молодь, дорослі з тривожністю	AI + психологи	https://www.wysa.io
Replika	Емоційна підтримка, віртуальний друг	Діалогова емпатія, генерація відповідей	Молодь, користувачі з самотністю	Глибоке навчання, NLP	https://replika.ai
Ada Health	Попередня діагностика	Симптом-чекер, клінічна база даних	Широке коло користувачів	AI, медичні алгоритми	https://ada.com
Lark	Діабет-профілактика, хронічні хвороби	24/7 коучинг, інтеграція зі страховками	Пацієнти з ризиками діабету/ожиріння	AI, мобільні сенсори	https://www.lark.com

Попри активний розвиток ринку цифрових медичних технологій в Україні, повноцінні інтелектуальні чат-боти, спрямовані на комплексну підтримку здорового способу життя (ЗСЖ), перебувають на початковій стадії впровадження. Наявні рішення охоплюють окремі аспекти (інформування,

ментальне здоров’я, запис до лікаря), але не забезпечують персоналізованої підтримки на основі штучного інтелекту.

Загальний огляд функціоналу та призначення актуальних українських чат-ботів для підтримки ЗСЖ представлено у Додатку Б. Наведена інформація слугує для ілюстрації поточного стану вітчизняного ринку цифрових помічників.

У табл. 1.3 наведено систематизований огляд українських платформ, що підтримують здоровий спосіб життя.

Таблиця 1.3

Приклади чат-ботів та цифрових сервісів ЗСЖ в Україні

Критерій	**Helsi-бот**	**Лікар Бот (DoctorBot)**	**MedBot**	**Mindly**	**ЗдоровІнфо**
Основна мета	Доступ до медичних сервісів	Інформація про COVID-19, самодіагностика	Пошук аптек, лікарень	Психоемоційна самодопомога	Освіта про ЗСЖ
Тип чат-бота	Інформаційний	Інформаційно-мотиваційний	Інформаційний	Мотиваційно-навчальний	Інформаційний
Інтелектуальність	Rule-based (без AI)	Сценарний, напівавтомат	Rule-based	Частковий AI	Rule-based
Платформи	Telegram, Viber	Telegram, Messenger	Telegram	Мобільний застосунок	Telegram
Функції персоналізації	Обмежені	Обмежені	Відсутні	Є часткова адаптація	Відсутні
Джерело даних	ЕСОЗ	Публічні джерела МОЗ	Аптечні бази	Внутрішні експертні моделі	Партнерські дані МОЗ
Оновлення/ Актуальність	Постійне	Переважно 2020–2022	Нерегулярне	Підтримується активно	Переважно статичне

Отже, в Україні відсутній повноцінний інтелектуальний чат-бот, орієнтований комплексно на підтримку ЗСЖ (фізичне, харчування, сон, психічне здоров’я) з використанням ШІ. Наявні рішення є або інформаційними, або мають вузьку функціональність (COVID, психоемоційний стан, медична адміністрація). Це створює інноваційну нішу для розробки вітчизняного україномовного AI-чат-бота для ЗСЖ з елементами рекомендаційної системи, трекінгу й коучингу.

З урахуванням аналізу вітчизняних рішень, таких як Helsi-бот (що орієнтований переважно на інформаційне обслуговування в межах ЕСОЗ) та

ЗдоровІнфо (що реалізує просвітницьку функцію у форматі rule-based логіки), виявлено низку обмежень: відсутність адаптивного діалогу, персоналізації, контекстного розуміння запитів користувача та гнучкої інтеграції рекомендаційних сервісів. Це зумовлює доцільність застосування інтелектуальних підходів до архітектури майбутнього чат-бота.

Для розробки інтелектуального чат-бота підтримки здорового способу життя були обрані наступні інструменти: Python як основна мова програмування, що забезпечує широкі можливості для реалізації штучного інтелекту, обробки природної мови (NLP) та інтеграції з зовнішніми сервісами; Rasa Framework – фреймворк з відкритим кодом для створення чат-ботів на основі машинного навчання, що дає змогу будувати гнучкі діалоги, обробляти інтенції користувача, керувати контекстом розмови та підключати рекомендаційні системи; PostgreSQL як система управління базами даних для зберігання структурованої інформації про користувачів, їхні параметри здоров’я та історію взаємодій; Visual Studio Code – як зручне та потужне середовище розробки з підтримкою Python, інтеграції з Git та розширень для Rasa; Telegram API – для реалізації взаємодії чат-бота з користувачами у популярному месенджері; Git – для керування версіями коду і командної розробки. Таке поєднання технологій забезпечує високу масштабованість, інтерактивність, персоналізовану взаємодію з користувачем та відповідність вимогам у сфері цифрового здоров’я.

Вибір мови програмування Python є обґрунтованим з огляду на широку підтримку бібліотек для NLU та машинного навчання (spaCy, Transformers, TensorFlow, scikit-learn); інтеграцію з API сторонніх сервісів охорони здоров’я (HL7 FHIR, RESTful API ECO3); активну спільноту у сфері розробки діалогових агентів.

Фреймворк: Rasa Framework (Rasa Open Source) на відміну від сценарних або rule-based рішень забезпечує підтримку intents & entities (класифікація намірів і витяг інформаційних об’єктів); диалоговий менеджмент на основі ML-політик; вбудовану систему трекінгу станів користувача, що дає змогу враховувати історію взаємодії; можливість інтеграції рекомендаційних систем та зовнішніх баз знань.

Модель обробки природної мови Transformers (BERT) на відміну від традиційних чат-ботів, дозволить реалізувати контекстну обробку запитів; впровадити персоналізовану комунікацію, що враховує історію здоров’я користувача; підвищити точність розпізнавання намірів у сфері ЗСЖ, де часто використовуються побутові або емоційно забарвлені формулювання.

Буде реалізовано гібридну рекомендаційну систему, що поєднує контентно-орієнтовані рекомендації (на основі параметрів здоров’я, цілей користувача); колаборативний підхід (на основі кластеризації схожих профілів користувачів); адаптивну логіку для трекінгу змін поведінки.

Планується використання модульної мікросервісної архітектури, що дозволить масштабувати систему за напрямками ЗСЖ (харчування, активність, сон, стрес); додавати нові канали (Telegram, Viber, Web-widget) без зміни ядра; реалізувати аналітичний модуль для збору анонімізованих даних про ефективність рекомендацій.

Таким чином, запропонована архітектура базується на сучасному стеку технологій з відкритим кодом, орієнтованому на реалізацію персоналізованої, україномовної платформи підтримки ЗСЖ. Обрані інструменти забезпечують ефективну обробку природної мови, масштабованість і відповідність сучасним вимогам безпеки та гнучкості розробки.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА АРХІТЕКТУРА ЧАТ-БОТА

2.1. Системний аналіз та архітектурна організація комплексу

Предметною областю дослідження є автоматизація процесів моніторингу фізіологічних показників та корекції способу життя людини. Ключові бізнес-процеси охоплюють збір, обробку та аналіз даних щодо харчової поведінки (нутриціологія), фізичної активності (кінезіологія) та гідратації організму.

На основі аналізу існуючих аналогів та специфіки взаємодії у форматі «людина-машина» (HCI) через месенджер-інтерфейс, було сформульовано перелік вимог (функціональних та нефункціональних) до програмного комплексу.

Функціональні вимоги формалізують поведінку системи та набір операцій, які вона повинна виконувати для задоволення потреб користувача.

1. Параметрична ідентифікація та управління профілем користувача.

Система повинна забезпечувати ініціалізацію сесії користувача та збір антропометричних даних, необхідних для розрахунку метаболічних показників:

- Вхідні дані: вік (роки), вага (кг), зріст (см), стать, рівень фізичної активності, мета (схуднення/підтримка/набір маси).
- Обробка: автоматичний розрахунок рівня базального метаболізму (BMR) та добової норми калорій, білків, жирів, вуглеводів (КБЖВ) та води.
- Алгоритм: використання валідованих формул, зокрема рівняння Міффліна-Сан Жеора, з урахуванням коефіцієнтів активності. Дані зберігаються у сутності `users` реляційної бази даних.

2. Інтелектуальний трекінг харчування.

Система повинна підтримувати мультимодальний ввід даних про споживання їжі:

- Обробка природної мови (NLU): розпізнавання намірів користувача (Intent Classification) та вилучення іменованих сутностей (NER) з текстових повідомлень (наприклад, «Я з'їв 200 грамів рису»). Пошук здійснюється у локальній базі даних (`food_products`) з використанням нечіткого пошуку (`fuzzy search`) за назвою та масивом псевдонімів (`aliases`).

- Комп'ютерний зір (Computer Vision): автоматизована ідентифікація продуктів за штрих-кодом. Процес включає отримання зображення через Telegram API, декодування стандарту EAN/UPC за допомогою бібліотеки `pyzbar`, відправку API-запиту до зовнішньої бази даних OpenFoodFacts та парсинг отриманого JSON-об'єкта для вилучення нутрієнтів.
- Розрахунок енергетичної цінності: автоматична калькуляція калорійності та макронутрієнтів пропорційно до вказаної маси продукту.

3. Моніторинг та контроль гідратації.

Система повинна забезпечувати дискретну фіксацію подій споживання рідини.

- Механізм: оновлення агрегованого показника `consumed_water_ml` у таблиці `daily_stats` в режимі реального часу.
- Візуалізація: надання користувачеві зворотного зв'язку щодо відсотка виконання добової норми.

4. Облік енерговитрат фізичної активності.

Система повинна розраховувати витрачені калорії на основі типу активності та її тривалості або інтенсивності.

- База знань: використання довідника активностей (`activity_types`), що містить коефіцієнти енерговитрат (аналог метаболічного еквівалента завдання – MET).
- Розрахунок: $E = T \times K$, де E – спалені калорії, T – тривалість/кількість, K – коефіцієнт інтенсивності (`kcal_factor`).

5. Аналітична звітність та візуалізація даних.

Система повинна агрегувати історичні дані та представляти їх у графічному вигляді.

- Генерація звітів: формування діаграм (стовпчикових, лінійних) розподілу КБЖВ та динаміки ваги з використанням бібліотеки `matplotlib`.
- Експорт даних: серіалізація вибірки з бази даних (`food_log`) у табличний формат `.xlsx` (Excel) за допомогою бібліотеки `pandas` для подальшого аналізу користувачем.

6. Автоматизована система гейміфікації.

Система повинна містити механізми поведінкового підкріплення для підвищення залученості користувача.

- Логіка: реалізація тригерної системи перевірки умов (`check_achievements_trigger`), яка активується після кожної транзакції (логування їжі, води, ваги).
- Механіки: нарахування досягнень (Achievements) за виконання специфічних умов (наприклад, «Vitamin King» за споживання 500г овочів) та відстеження безперервних серій використання (Streaks).

Нефункціональні вимоги визначають атрибути якості програмного забезпечення та обмеження архітектурного рішення.

1. *Доступність (Availability):* система повинна функціонувати у режимі 24/7, забезпечуючи асинхронну обробку запитів через черги повідомлень Telegram.
2. *Інтернаціоналізація (i18n):* архітектура повинна підтримувати динамічне перемикання мови інтерфейсу (українська/англійська) на рівні сесії користувача, використовуючи відповідні слоти (`language`) для вибору шаблонів відповідей.
3. *Масштабованість (Scalability):* модульна структура на базі Rasa Open Source повинна дозволяти горизонтальне масштабування та додавання нових сценаріїв діалогу (Stories/Rules) без зміни ядра системи.
4. *Цілісність та безпека даних (Security & Integrity):*

- Використання параметризованих SQL-запитів (бібліотека psycopg2) для запобігання атакам типу SQL Injection.
- Зберігання даних у нормалізованій реляційній базі даних PostgreSQL для забезпечення ACID-властивостей транзакцій.
- Валідація вхідних даних (типів даних, діапазонів значень) на стороні FormValidationAction.

Для формалізації взаємодії акторів із системою розроблено діаграму прецедентів (Use Case Diagram). Ключовим актором виступає «Користувач», який ініціює сценарії використання (Use Cases), такі як «Реєстрація профілю», «Логування харчування», «Сканування штрих-коду» та «Перегляд статистики».

Система виступає як актор-виконавець, що обробляє запити та повертає результат.

Проєктування програмного комплексу базується на принципах ***сервіс-орієнтованої архітектури (Service-Oriented Architecture, SOA)***. Такий підхід дозволяє декомпозувати систему на набір слабко пов'язаних модулів, що взаємодіють через стандартизовані протоколи, забезпечуючи модульність, масштабованість та незалежне розгортання функціональних блоків.

Вибір інструментальних засобів розробки здійснено на основі аналізу критеріїв продуктивності, наявності спеціалізованих бібліотек для розв'язання задач NLP (Natural Language Processing) та підтримки спільноти.

1. Мова програмування Python 3.8+

Обрана як основне середовище розробки завдяки її домінуючій ролі у сферах Data Science та Machine Learning. Високорівневий синтаксис та динамічна типізація дозволяють реалізувати концепцію швидкої розробки застосунків (RAD).

У проєкті задіяно наступну екосистему бібліотек:

Pandas: Використовується у модулі `health_actions.py` для маніпуляцій з табличними даними та підготовки їх до експорту. Наприклад, функція `ActionExportData` використовує `pandas.DataFrame` для серіалізації логів харчування у формат `.xlsx`.

Matplotlib: Забезпечує візуалізацію статистичних даних. У класі `ActionShowWaterProgress` (файл `health_actions.py`) реалізовано генерацію стовпчикових діаграм динаміки гідратації, які конвертуються у байтовий потік для відправки користувачу.

Pyzbar: Інтегрована у модуль `telegram_bridge.py` для реалізації функцій комп'ютерного зору. Вона забезпечує декодування одновимірних штрих-кодів стандартів EAN-13/UPC із зображень, отриманих від користувача, що дозволяє автоматизувати введення продуктів.

Psycopg2: Виконує роль драйвера для асинхронної взаємодії з базою даних. У файлі `db_utils.py` функція `get_pg_connection` ініціалізує з’єднання, забезпечуючи виконання транзакцій.

2. Платформа Rasa Open Source

Використовується як семантичне ядро системи. На відміну від хмарних рішень (Dialogflow, Microsoft LUIS), Rasa дозволяє розгортати NLU-модель локально (on-premise), гарантуючи повний контроль над конфіденційністю даних.

Архітектура Rasa у проєкті включає:

Rasa NLU: Відповідає за класифікацію намірів (Intents), визначених у файлі `nlu.yml` (наприклад, `log_food, scan_barcode`), та вилучення сутностей (Entities) із неструктурованого тексту.

Rasa Core: Керує контекстом діалогу на основі політик та правил, описаних у `rules.yml`, забезпечуючи детерміновану поведінку у критичних сценаріях (наприклад, `Start new session and show main menu`).

3. СУБД PostgreSQL

Виступає сховищем персистентних даних. Це об’єктно-реляційна система (ОРСКБД), що підтримує властивості ACID (Atomicity, Consistency, Isolation, Durability). У проєкті вона використовується для виконання складних аналітичних запитів, зокрема агрегації статистики споживання калорій та води у таблиці `daily_stats` (схема описана у `db_utils.py`).

4. Telegram Bot API

Обраний як інтерфейс взаємодії «людина-машина» (HCI). API надає можливості створення інтерактивних елементів (`ReplyKeyboardMarkup, InlineKeyboardMarkup)`, що реалізовано у методі `build_keyboard_from_buttons` файлу `telegram_bridge.py.` Також API є критичним для передачі мультимедійних об'єктів (фотографій) для модуля сканування.

Архітектура програмного комплексу побудована за модульним принципом і складається з чотирьох логічно відокремлених компонентів:

1. Комунікаційний шлюз (Communication Gateway / Telegram Bridge) реалізований у модулі `telegram_bridge.py.` Цей компонент виступає посередником між Telegram API та ядром Rasa, реалізуючи патерн Adapter.

Функції: отримання оновлень (Updates) через механізм *Long Polling* за допомогою класу `Updater;` попередня обробка медіа-даних: у функції `handle_photo` реалізовано алгоритм завантаження зображення у пам'ять

(`io.BytesIO`), його конвертація бібліотекою PIL та передача у декодер `pyzbar`; маршрутизація запитів до Rasa Server через HTTP REST API у форматі JSON; рендеринг відповідей (текст, кнопки, графіки).

Власна реалізація моста дозволила обійти обмеження стандартних конекторів Rasa (зокрема, щодо розміру файлів та специфічної обробки бінарних даних зображень).

2. Сервер обробки природної мови (Rasa Server), центральний компонент, що обробляє текстові вхідні дані.

Вхідний текст проходить через налаштований у `config.yml` конвеєр (Pipeline): токенізація $\rightarrow$ векторизація $\rightarrow$ класифікація наміру. Далі модуль Rasa Core на основі правил (`rules.yml`) визначає необхідну дію (Action) – наприклад, перехід до слоту форми або виклик зовнішньої дії `action_log_food`.

3. Сервер бізнес-логіки (Rasa Action Server), реалізований у файлі health_actions.py. Це окремий мікросервіс, що виконує Python-код у відповідь на RPC-виклики від Rasa Core.

Функціонал: валідація даних – класи, успадковані від `FormValidationAction` (наприклад, `ValidateProfileForm`), перевіряють коректність введених параметрів (вік, вага); математичне моделювання: розрахунок норм КБЖВ за формулою Міффліна-Сан Жеора; інтеграція: виконання HTTP-запитів до зовнішніх API (OpenFoodFacts) у класі `ActionScanBarcode`; генерація звітів: створення графіків за допомогою `matplotlib` та їх кодування у Base64 для передачі через JSON.

4. Шар даних (Data Persistence Layer), реалізований через модуль абстракції db_utils.py, який інкапсулює SQL-запити.

Забезпечує CRUD-операції. Шар даних ізолює бізнес-логіку від прямої роботи з драйвером. Наприклад, функція `log_food_entry` виконує параметризовану вставку даних у таблицю `food_log`, запобігаючи SQL-ін'єкціям та забезпечуючи цілісність посилань на користувача (`user_id`).

Така архітектура забезпечує чітке розділення відповідальності (Separation of Concerns): `telegram_bridge.py` відповідає за *View* (представлення), Rasa – за *Controller* (керування потоком), `health_actions.py` – за *Model Logic* (бізнес-правила), а PostgreSQL – за *Model Data* (дані). Це дозволяє масштабувати кожен компонент незалежно та спрощує процес налагодження системи. Структурна схема взаємодії компонентів системи представлена у додатку В, рис. В.1.

Ключовим аспектом побудови надійної інформаційної системи є розробка ефективної моделі даних, яка забезпечує не лише збереження стану діалогу (state persistence), але й швидкий доступ до історичних даних для генерації аналітики. У рамках дослідження було спроєктовано реляційну модель бази даних, що відповідає третій нормальній формі (3NF) для мінімізації надлишковості даних, з елементами контрольованої денормалізації в аналітичних таблицях для підвищення продуктивності.

На концептуальному рівні інформаційна модель системи представляє собою сукупність сутностей, об'єднаних семантичними зв'язками. Центральною сутністю є `User` (Користувач), навколо якої будуються транзакційні дані (логи харчування, активності) та нормативно-довідкова інформація (каталоги продуктів, типи досягнень).

Модель характеризується наступними типами кардинальності зв'язків:

1. «Один-до-багатьох» (1:N): Реалізовано між сутністю `User` та журналами подій. Один користувач генерує множину записів у таблицях `food_log` та `activity_log`.
2. «Багато-до-багатьох» (M:N): Реалізовано між користувачами та досягненнями. Оскільки реляційна модель не підтримує прямих зв'язків M:N, вони декомпозовані через асоціативну сутність `user_achievements`.

Схему базу даних унаочнено на рис. 2.1.

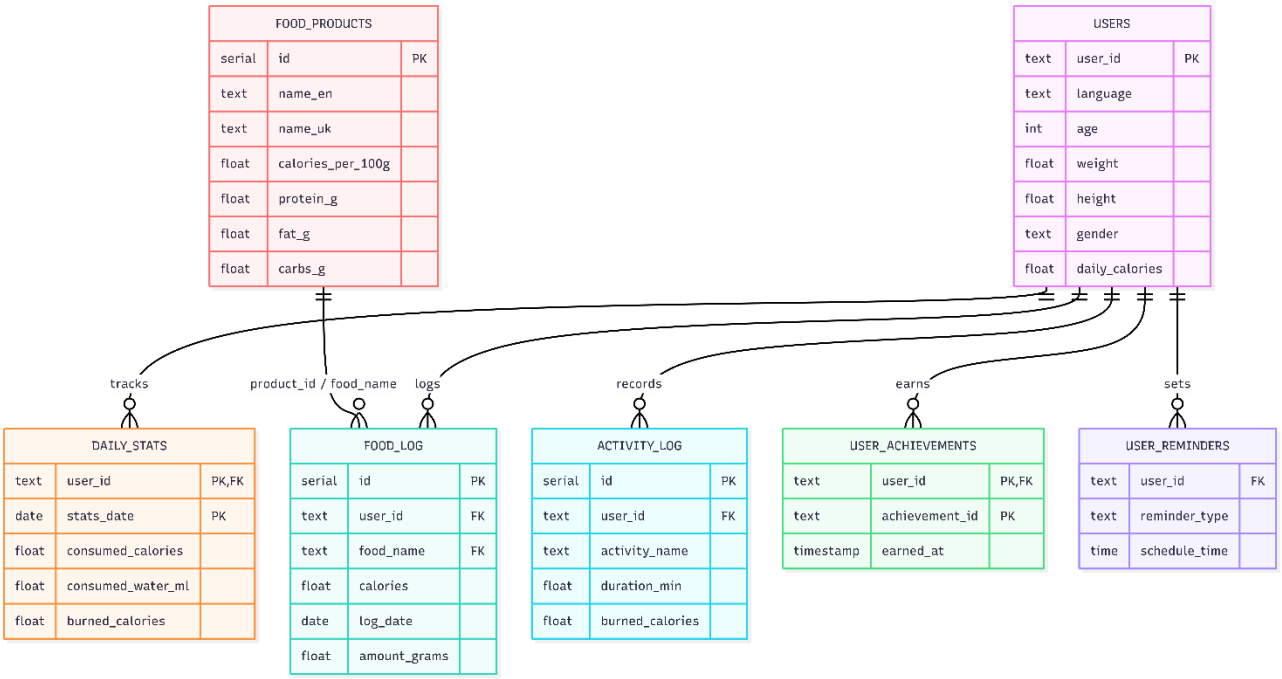


Рис. 2.1. ER-діаграма (Схема бази даних)

Фізична реалізація бази даних виконана у середовищі СУБД PostgreSQL. Вибір даної системи зумовлений її підтримкою розширених типів даних (масиви, JSONB), транзакційної цілісності (ACID) та високою продуктивністю при виконанні складних `JOIN`-запитів.

Схема бази даних визначена у модулі `db_utils.py`. Розглянемо детально структуру ключових таблиць та їх програмну реалізацію.

1. Таблиця профілів користувачів (`users`*)* зберігає стан та налаштування суб'єкта системи. Як первинний ключ (`PRIMARY KEY`) використовується `user_id` типу `TEXT`, що відповідає унікальному ідентифікатору Telegram ID. Це дозволяє уникнути зайвих операцій `JOIN` при ідентифікації користувача під час обробки вебхуків, що унаочнено у ліст. 2.1.

Лістинг 2.1

DDL-запит створення таблиці користувачів (`db_utils.py`)

```
cur.execute("""
    CREATE TABLE IF NOT EXISTS users (
        user_id TEXT PRIMARY KEY,
        language TEXT DEFAULT 'en',
        age INTEGER,
        weight REAL,
        height REAL,
        gender TEXT,
```

```
        goal TEXT,
        daily_calories REAL,
        created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    )
""")
```

Атрибут daily_calories є розрахунковим полем, що зберігається персистентно. Це рішення зменшує навантаження на CPU, усуваючи необхідність перерахунку формули Міффліна-Сан Жеора при кожному запиті меню.

2. Довідник продуктів харчування (food_products) містить нормативно-довідкову інформацію. Використовує сурогатний первинний ключ id (SERIAL). Особливістю реалізації є підтримка інтернаціоналізації на рівні схеми даних (name_en, name_uk) та використання масиву aliases для зберігання синонімів, що покращує релевантність пошуку. Це представлено у ліст. 2.2.

Лістинг 2.2

Структура довідника продуктів

```
cur.execute("""
    CREATE TABLE IF NOT EXISTS food_products (
        id SERIAL PRIMARY KEY,
        name_en TEXT,
        name_uk TEXT,
        calories_per_100g REAL,
        protein_g REAL,
        fat_g REAL,
        carbs_g REAL,
        aliases TEXT[]  -- Масив синонімів для fuzzy search
    )
""")
```

3. Журнал споживання їжі **(**food_log) – основна транзакційна таблиця. Для забезпечення посилальної цілісності (Referential Integrity) використано обмеження FOREIGN KEY з каскадним видаленням (ON DELETE CASCADE). Це гарантує, що при видаленні акаунту користувача вся його історія харчування буде автоматично знищена відповідно до вимог GDPR.

Важливим архітектурним рішенням є snapshotting (фіксація даних): нутрієнти (calories, protein тощо) розраховуються для конкретної порції та зберігаються у лозі в момент запису. Це захищає історичні дані користувача від майбутніх змін у рецептурі продуктів у довіднику.

4. Таблиця агрегованої статистики (daily_stats) реалізує патерн денормалізації для оптимізації операцій читання. Замість виконання ресурсномістких агрегатних функцій SUM() по всій історії логів при кожному запиті, система підтримує актуальний стан лічильників у таблиці daily_stats. Використовується складений первинний ключ (user_id, stats_date), що забезпечує унікальність запису для кожного користувача в межах доби.

На ліст. 2.3 відтворено оновлення даних, яке реалізовано через механізм Upsert (INSERT ... ON CONFLICT DO UPDATE), що забезпечує атомарність операції.

Лістинг 2.3

Реалізація Upsert для статистики (db_utils.py)

```
def add_daily_food(user_id, calories):
    """Оновлює лічильник калорій, створюючи запис при необхідності."""
    sql = """
        INSERT INTO daily_stats (user_id, stats_date, consumed_calories)
        VALUES (%s, CURRENT_DATE, %s)
        ON CONFLICT (user_id, stats_date)
        DO UPDATE SET
            consumed_calories    =    daily_stats.consumed_calories    +
EXCLUDED.consumed_calories
    """
    # Виконання запиту...
```

5. Підсистема гейміфікації (user_achievements) реалізує зв'язок M:N між користувачами та типами нагород. Атрибут earned_at дозволяє будувати часові ряди отримання досягнень, що необхідно для аналізу залученості (Retention).

Для дотримання нефункціональних вимог щодо часу відгуку системи (Response Time < 200ms) реалізовано стратегію індексації:

1. B-Tree індекси: автоматично створені для всіх первинних ключів. Додатково рекомендовано створення індексів для зовнішніх ключів (user_id) у таблицях логів для прискорення операцій з'єднання (JOIN).

2. Текстовий пошук: для реалізації функції пошуку продукту за назвою використано оператор ILIKE (case-insensitive matching). У функції search_food_by_name (файл db_utils.py) запит формується наступним чином:

```
cur.execute("""
    SELECT * FROM food_products
    WHERE name_uk ILIKE %s OR name_en ILIKE %s OR %s = ANY(aliases)
    LIMIT 10
""", (pattern, pattern, query))
```

Використання оператора ANY(aliases) дозволяє ефективно шукати входження у масивах синонімів.

Така структура бази даних забезпечує баланс між нормалізацією (для запобігання аномаліям даних) та денормалізацією (для прискорення аналітичних запитів), створюючи надійний фундамент для функціонування інтелектуального чат-бота.

2.2. Проєктування логіки функціонування та інтелектуальних компонентів

Архітектура інтелектуальної складової програмного комплексу базується на фреймворку Rasa Open Source і реалізує конвеєрну (pipeline) обробку запитів. Система функціонує на основі двох взаємопов'язаних модулів: Rasa NLU, що відповідає за семантичний аналіз і перетворення неструктурованого природного тексту у структурований формат (Intent/Entity), та Rasa Core, що забезпечує керування контекстом і прийняття рішень (Dialogue Policy) щодо наступної дії агента.

Завданям модуля NLU є класифікація наміру користувача (Intent Classification) та вилучення іменованих сутностей (Named Entity Recognition – NER). Навчальна вибірка, сформована у файлі `nlu.yml`, містить анотовані приклади фраз для понад 20 класів інтенцій.

Семантичний простір намірів структуровано за функціональним призначенням:

1. *Навігаційні та системні інтенції:* забезпечують керування сесією (`start_over, goodbye`) та навігацію по меню (`menu_main, help`).
2. *Транзакційні інтенції (Data Entry):* ініціюють процеси запису даних до бази. Наприклад, інтент `log_food` розпізнає спроби користувача зафіксувати прийом їжі природною мовою.
3. *Інформаційні запити:* (`show_water_progress, view_profile`) – активують генерацію аналітичних звітів.
4. *Мультимедійні тригери:* інтент `scan_barcode`, який активується не текстовим вводом, а програмним викликом (API trigger) з боку `telegram_bridge.py` після успішного декодування зображення.

Для параметризації запитів застосовано механізм вилучення сутностей, що представлено у ліст. 2.4. У файлі `nlu.yml` визначено сутності `food_name` (назва

продукту) та `amount` (кількість). Для нормалізації одиниць виміру впроваджено механізм Entity Synonyms, що дозволяє зводити варіативні лексичні форми до канонічного вигляду.

Лістинг 2.4

Фрагмент конфігурації NLU (`nlu.yml`)

```
- intent: log_food
  examples: |
    - я з'їв [яблуко](food_name)
    - додай [200](amount) [грам](unit) [гречки](food_name)
    - запиши обід: [борщ](food_name) [300 г](amount)
- synonym: g
  examples: |
    - грам
    - грамів
    - гр
    - г.
```

Такий підхід дозволяє системі коректно інтерпретувати фразу *«200 грамів»* як числове значення 200 та одиницю виміру g, що є критичним для подальших математичних розрахунків калорійності.

Керування контекстом та станом діалогу (Domain Ontology)

Онтологія предметної області діалогової системи визначена у файлі `domain.yml`. Збереження контексту (Context Persistence) реалізовано через механізм слотів (Slots), які виконують функцію короткострокової пам'яті бота.

Виділено наступні категорії слотів залежно від їх впливу на діалог:

1. *Інформаційні слоти* (Text/Any): Зберігають дані профілю (`user_id, language, age`), які не впливають на хід діалогу безпосередньо, але використовуються у бізнес-логіці (`health_actions.py`).

2. *Логічні слоти* (Float/Categorical): Впливають на розгалуження сценаріїв. Наприклад, слот goal (мета: схуднення/набір) визначає коефіцієнти розрахунку КБЖВ.

3. *Тимчасові буферні слоти:* (`temp_food_id, image_data`) – використовуються для передачі даних між кроками багатоетапних діалогів.

Стратегія керування діалогом (Dialogue Policies)

Модуль Rasa Core використовує гібридну стратегію керування діалогом, поєднуючи детерміновані правила та імовірнісні моделі.

1. Rule-based Policy (Детермінована логіка)

Правила (rules.yml) використовуються для обробки однозначних сценаріїв («If-Then» logic), які не залежать від історії діалогу. Це забезпечує надійність виконання критичних команд. Приклад правила обробки нагадування (`rules.yml`) представлено у ліст. 2.5.

Наприклад, правило Trigger Water Check гарантує миттєву реакцію на запит перевірки води, незалежно від попереднього контексту.

Лістинг 2.5

Приклад правила обробки нагадування (`rules.yml`)

```
rules:
  - rule: Trigger Water Check
    steps:
      - intent: show_water_progress   # Вхідний намір
      - action: action_show_water_progress # Виклик Python-функції
```

2. Slot Filling & Form Policy (Обробка форм)

Для реалізації складних сценаріїв збору даних (наприклад, реєстрація користувача) використано механізм форм (Forms). Форма активує цикл `active_loop`, який утримує фокус діалогу в межах одного завдання, доки не будуть заповнені всі обов'язкові слоти.

У системі реалізовано форму `profile_form`, яка послідовно запитує параметри: `age` $\rightarrow$ `weight` $\rightarrow$ `height` $\rightarrow$ `gender`. Валідація вхідних даних здійснюється класом `ValidateProfileForm` у файлі `health_actions.py`, що дозволяє відсіювати некоректні значення (наприклад, вік < 10 або вага < 30 кг) до моменту запису в базу даних.

Діаграму станів діалогу (State Machine Diagram) унаочнено у додатку В, рис. В.2.

Взаємодія NLU та Core компонентів забезпечує гнучкість системи: вона здатна як чітко слідувати інструкціям (Rules), так і адаптуватися до вільного вводу користувача (Intents/Forms).

Функціональне ядро системи реалізовано на рівні Rasa Action Server (`health_actions.py`). Цей компонент відповідає за виконання детермінованих алгоритмів, математичні розрахунки та взаємодію з базою даних, забезпечуючи перетворення вхідних даних у значущу інформацію для користувача.

1. Алгоритм розрахунку добових енергетичних потреб (КБЖВ)

Для визначення індивідуальних фізіологічних норм користувача імплементовано алгоритм розрахунку базального метаболізму (Basal Metabolic Rate, BMR) та загальних добових енерговитрат (Total Daily Energy Expenditure, TDEE).

В основу покладено рівняння Міффліна-Сан Жеора, яке, згідно з актуальними дослідженнями в галузі дієтології, демонструє найменшу похибку (в межах 10%) для осіб з різним типом тілобудови. Реалізація алгоритму міститься у методі `ActionSaveProfile.run` та допоміжних функціях.

Математична модель розрахунку описується системою рівнянь:

$$BMR = (10 \cdot W) + (6.25 \cdot H) - (5 \cdot A) + S$$

де:

- W – вага тіла (кг),
- H – зріст (см),
- A – вік (років),
- S – статевий коефіцієнт ($S=+5$ для чоловіків, $S=-161$ для жінок).

Фінальний розрахунок добової калорійності (DC) враховує коефіцієнт фізичної активності (PAL) та мету користувача ($Target$):

$$DC = BMR \cdot PAL \cdot Target$$

Технічні деталі реалізації цих функцій у межах файлу `health_actions.py` продемонстровано в ліст. 2.6.

Лістинг 2.6

Реалізація розрахунку BMR та TDEE

```
# Базовий розрахунок BMR
bmr = (10 * weight) + (6.25 * height) - (5 * age)
if gender == 'male':
    bmr += 5
else:
    bmr -= 161
# Визначення коефіцієнта активності (PAL)
activity_multipliers = {
    "sedentary": 1.2,      # Малорухливий
    "light": 1.375,        # Легкі тренування
    "moderate": 1.55,      # Помірні тренування
    "active": 1.725,       # Інтенсивні тренування
    "very_active": 1.9     # Професійний спорт
}
```

```
tdee = bmr * activity_multipliers.get(activity_level, 1.2)
# Коригування відповідно до мети
if goal == "lose_weight":
    daily_calories = tdee * 0.85  # Дефіцит 15%
elif goal == "gain_muscle":
    daily_calories = tdee * 1.15  # Профіцит 15%
else:
    daily_calories = tdee         # Підтримка ваги
```

Система автоматично розраховує макронутрієнти (БЖВ) на основі класичної пропорції: Білки (30%), Жири (30%), Вуглеводи (40%) від загального калоражу.

2. Логіка автоматизованої гейміфікації (Event-Driven Architecture)

Для підвищення показника утримання користувачів (Retention Rate) розроблено підсистему гейміфікації, яка базується на подійно-орієнтованій архітектурі. Алгоритм перевірки досягнень (`check_achievements_trigger`) виконується асинхронно після кожної транзакції (логування їжі, води або ваги).

Алгоритм складається з чотирьох етапів: 1) агрегація даних: вибірка актуальної статистики користувача з таблиць `daily_stats`, `food_log` та `activity_log` через модуль `db_utils.py`; 2) верифікація умов (Conditional Check): порівняння поточних показників із пороговими значеннями, визначеними для кожного типу досягнення; 3) фільтрація дублікатів: перевірка таблиці `user_achievements` для запобігання повторному нарахуванню вже отриманих нагород; 4) персистентність та нотифікація: запис факту отримання нагороди в БД та формування повідомлення для користувача.

У коді це реалізовано через каскад умов, що відтворено у ліст. 2.7.

Лістинг 2.7

Фрагмент алгоритму перевірки досягнень

```
def check_achievements_trigger(user_id: str, dispatcher: CollectingDispatcher, lang: str):
    # Отримання агрегованих даних
    stats = get_today_stats(user_id)
    water_ml = stats.get("water_ml", 0)
    existing_achievements = get_user_achievements(user_id) # Set of IDs
    new_unlocks = []
    # --- Перевірка умов (приклад для води) ---
    # Досягнення "Water Beginner": випити 2000 мл води
    if "water_beginner" not in existing_achievements and water_ml >= 2000:
        award_achievement(user_id, "water_beginner")
        new_unlocks.append("💧 Water Beginner")
```

```
    # --- Перевірка серій (Streaks) ---
    streak_days = get_streak_days(user_id)
    if "streak_3" not in existing_achievements and streak_days >= 3:
        award_achievement(user_id, "streak_3")
        new_unlocks.append("🔥 3 Days Streak")
    # ... перевірка інших 30+ умов ...
    # Нотифікація
    if new_unlocks:
        msg = "🏆 New Achievements Unlocked!\n" + "\n".join(new_unlocks)
        dispatcher.utter_message(text=msg)
```

Такий підхід дозволяє масштабувати систему гейміфікації шляхом додавання нових блоків `if` без зміни архітектури бази даних.

3. Алгоритм обробки мультимедійних даних (Computer Vision Pipeline)

Функціонал сканування штрих-кодів реалізує сценарій швидкого введення даних, мінімізуючи когнітивне навантаження на користувача. Обробка зображення відбувається у конвеєрному режимі (Pipeline), розподіленому між `telegram_bridge.py` (клієнтська обробка) та `health_actions.py` (бізнес-логіка).

Етап А: Попередня обробка та декодування (Image Pre-processing). Цей етап винесено у шар `telegram_bridge.py` для розвантаження основного сервера Rasa. Отримане зображення конвертується у потік байтів, після чого бібліотека pyzbar аналізує растр на наявність лінійних кодів стандартів EAN-13 або UPC-A (ліст. 2.8).

Лістинг 2.8

Декодування зображення

```
# Завантаження фото в пам'ять
photo_file = update.message.photo[-1].get_file()
bio = io.BytesIO()
photo_file.download(out=bio)
img = Image.open(bio)
# Декодування
decoded_objects = decode(img)
if decoded_objects:
    barcode_data = decoded_objects[0].data.decode("utf-8")
    # Формування payload для Rasa
    payload = f'/scan_barcode{{"barcode": "{barcode_data}"}}'
    forward_to_rasa(payload, str(chat_id))
```

Етап Б: Запит до зовнішнього API та нормалізація даних. Отримавши інтент `scan_barcode` із сутністю `barcode`, клас `ActionScanBarcode` виконує HTTP-запит до відкритої бази даних OpenFoodFacts. Отримана JSON-відповідь парситься, а нутрієнтні дані нормалізуються (приводяться до стандартних

ключів: `proteins_100g, energy-kcal_100g`) та зберігаються у тимчасових слотах для подальшого діалогу (запиту ваги спожитої порції). Унаочнення зазначеного представлено у ліст. 2.9 та на лис. 2.2.

Лістинг 2.9

Програмна реалізація взаємодії із зовнішнім сервісом OpenFoodFacts у класі `ActionScanBarcode`

```
f"https://world.openfoodfacts.org/api/v0/product/{barcode}.json"
response = requests.get(url, timeout=5)
data = response.json()
if data.get("status") == 1:
    product = data.get("product", {})
    nutriments = product.get("nutriments", {})
    # Збереження у тимчасові слоти для наступного кроку (ActionAskAmount)
    return [
        SlotSet("temp_scan_name", product.get("product_name")),
        SlotSet("temp_scan_calories",  nutriments.get("energy-kcal_100g",
0)),
        SlotSet("temp_food_id", "SCAN_EXTERNAL") # Маркер джерела даних
    ]
```

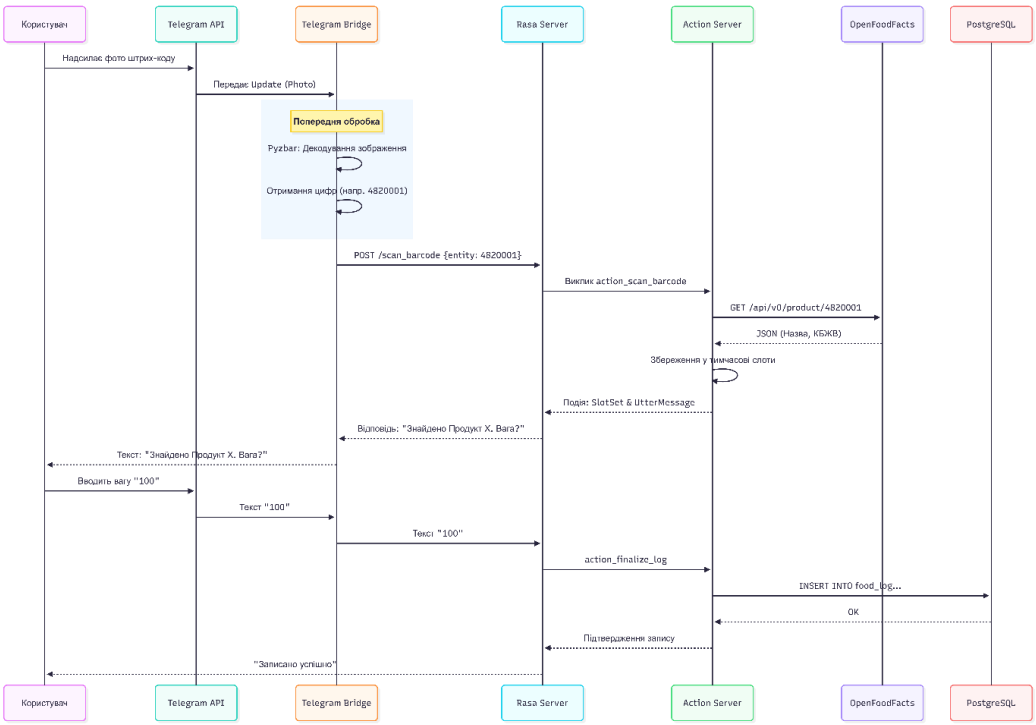


Рис. 2.2. Діаграма послідовності (Sequence Diagram)

Описана алгоритмічна структура забезпечує високу автономність системи, дозволяючи виконувати складні обчислення та інтеграції «прозоро» для кінцевого користувача.

Проєктування інтерфейсу користувача (UI) та досвіду взаємодії (UX) виконано з урахуванням обмежень та особливостей месенджера Telegram. Архітектура клієнтської частини реалізована за принципом «тонкого клієнта» (Thin Client), де вся логіка відображення та навігації керується сервером, а клієнтський додаток виконує лише функцію рендерингу отриманих JSON-структур.

Динамічна генерація інтерфейсу (Dynamic UI Rendering)

Інтерфейс взаємодії базується на використанні персистентних клавіатур (`ReplyKeyboardMarkup`), що забезпечує інтуїтивну навігацію та мінімізує необхідність ручного введення текстових команд. На відміну від статичного визначення меню у коді бота, у розробленій системі застосовано підхід Server-Driven UI.

Структура меню та навігаційні переходи визначені декларативно у файлі конфігурації Rasa `domain.yml`. Кожна відповідь бота (Response) може містити масив кнопок `buttons`. Модуль `telegram_bridge.py` виступає як адаптер, що десеріалізує JSON-відповідь Rasa та трансформує її у об'єкти Telegram API (ліст.2.10).

Лістинг 2.10

Метод побудови динамічної клавіатури (`telegram_bridge.py`)

```
def build_keyboard_from_buttons(chat_id: int, buttons: List[Dict[str, Any]]) -> ReplyKeyboardMarkup:
    # Динамічне мапування: Текст кнопки -> Payload (команда)
    mapping: Dict[str, str] = {}
    rows: List[List[KeyboardButton]] = []
    row: List[KeyboardButton] = []
    for btn in buttons:
        title = btn.get("title", "")
        payload = btn.get("payload") or title
        # Формування сітки кнопок (Grid Layout) по 2 в ряд
        row.append(KeyboardButton(title))
        if len(row) == 2:
            rows.append(row)
            row = []
    # Збереження мапінгу для обробки натискань
    BUTTON_PAYLOADS[chat_id] = mapping
    return ReplyKeyboardMarkup(rows, resize_keyboard=True)
```

Такий підхід забезпечує відокремлення представлення від логіки (Separation of Concerns): додавання нового пункту меню не потребує зміни коду Python, а лише оновлення конфігурації YAML.

Ієрархія навігації: реалізовано деревоподібну структуру меню, кореневим вузлом якої є інтент `menu_main`. Навігаційний граф включає наступні гілки:

1. Харчування (`menu_nutrition`): доступ до сценаріїв логування (`/log_food`), пошуку (`/search_food`) та сканування (`/scan_barcode`).

2. Гідратація (`menu_water`): швидке додавання води (+250мл) та перегляд графіків.

3. Фізична активність (`menu_fitness`): селектор категорій спорту.

4. Аналітика та налаштування: генерація звітів та керування профілем.

Реалізація сценарію додавання прийому їжі (Multimodal Input Handling)

Ключовий бізнес-процес системи – фіксація спожитої їжі – спроєктовано з підтримкою мультимодального вводу, що продемонстровано у ліст. 2.11. Система реалізує патерн «*Поліморфізм вхідних даних»*, де різні канали інформації (текст, клік, зображення) зводяться до єдиного формату обробки.

Процес складається з трьох етапів:

1. *Ініціалізація (Input Acquisition):*
   - *Текстовий ввід:* NLU класифікує інтент `log_food` та виділяє сутності `food_name, amount`.
   - *Кнопковий ввід:* Вибір зі списку улюблених або результатів пошуку активує payload `/select_food`.
   - *Візуальний ввід:* Сканування штрих-коду активує `ActionScanBarcode`, який заповнює тимчасові слоти даними із зовнішнього API.

2. *Валідація та уточнення (Slot Filling):* якщо отриманих даних недостатньо (наприклад, не вказана вага), система ініціює уточнюючий діалог (`action_ask_amount`).

3. *Фіналізація транзакції (Transaction Commit):* незалежно від джерела даних, процес завершується викликом універсального методу

`ActionFinalizeLog`. Цей клас виконує нормалізацію даних, розрахунок нутрієнтів та атомарний запис у таблицю `food_log`.

Лістинг 2.11

Уніфікація логування (health_actions.py)

```
class ActionFinalizeLog(Action):
    def run(self, dispatcher, tracker, domain):
        # Отримання даних з різних джерел (сканер або пошук БД)
        food_id = tracker.get_slot("temp_food_id")
        amount = tracker.get_slot("amount")
        if food_id == "SCAN_EXTERNAL":
            # Обробка даних зі сканера
            name = tracker.get_slot("temp_scan_name")
            cal = tracker.get_slot("temp_scan_calories") * (amount / 100)
            log_source = "barcode_scan"
        else:
            # Обробка даних з локальної БД
            product = get_food_by_id(int(food_id))
            cal = product['calories'] * (amount / 100)
            log_source = "manual_search"
        # Єдина точка запису в БД
        log_food_entry(user_id, name, cal, ..., log_source)
```

У додатку В, рис. В.3 наведено діаграму діяльності, що візуалізує алгоритм обробки мультимодального вводу.

Така організація інтерфейсу дозволяє забезпечити високу гнучкість системи, підтримку різних сценаріїв використання та зручність для кінцевого користувача, приховуючи складність внутрішньої логіки за простими візуальними елементами.

РОЗДІЛ 3. ДОСЛІДЖЕННЯ ЯКОСТІ ТА ЗРУЧНОСТІ ВИКОРИСТАННЯ (ЮЗАБІЛІТІ) СИСТЕМИ

Оцінка якості розробленої системи вимагає комплексного підходу, оскільки чат-бот є гібридною інтелектуальною платформою, що поєднує методи машинного навчання для обробки природної мови (NLP) та детерміновані алгоритми для математичних розрахунків і взаємодії з API.

У даному дослідженні якість системи оцінювалася за двома основними напрямками:

1. *Технічна ефективність:* швидкість реакції на запити, стабільність інтеграції з API OpenFoodFacts, коректність розрахунків енергетичної цінності.
2. *Юзабіліті (Usability):* суб'єктивна задоволеність користувачів, легкість навчання роботі з ботом та ефективність виконання цільових сценаріїв.

Для кількісного вимірювання було обрано наступні метрики:

- *Task Completion Rate (TCR):* відсоток успішно завершених діалогів (наприклад, успішне отримання даних про калорії після сканування).
- *Average Response Time (ART):* середній час очікування відповіді від бота.
- *System Usability Scale (SUS):* стандартизована анкета для оцінки зручності інтерфейсу.

Для проведення тестування було сформовано фокус-групу з 12 респондентів віком від 20 до 35 років. Учасники мали різний рівень технічної підготовки, що дозволило перевірити інтуїтивність інтерфейсу.

Сценарій експерименту: кожному учаснику було запропоновано виконати послідовність дій, що імітує реальне використання продукту:

1. Розпочати діалог та привітатися.
2. Активувати функцію сканування штрих-коду.
3. Надіслати боту валідний штрих-код продукту (тестовий набір: EAN-13 коди популярних продуктів).
4. Вказати вагу спожитої порції (наприклад, «200 грам» або «0.5 кг»).
5. Запитати загальну кількість спожитих калорій за день.
6. Спробувати надіслати неіснуючий код або текст замість коду (перевірка обробки помилок).

Частково результати експерименту ініціалізація профілю користувача, валідація антропометричних даних та ідентифікація продукту за штрих-кодом й розрахунок калоражу порції представлено на рис. 3.1. та рис. 3.2.



Рис. 3.1. Початковий етап взаємодії: ініціалізація профілю користувача та валідація антропометричних даних



Рис. 3.2. Приклад успішного діалогу: ідентифікація продукту за штрих-кодом та розрахунок калоражу порції

Часові характеристики системи

Критичним параметром для Chatbot-систем є час затримки (latency). Оскільки бот виконує синхронний запит до зовнішнього сервера `world.openfoodfacts.org`, час відповіді залежить від швидкості мережі та часу обробки запиту API.

У ході експерименту було здійснено 50 контрольних запитів. Результати вимірювань наведено в табл. 3.1.

Таблиця 3.1

Часові характеристики обробки запитів

Етап обробки запиту	Середній час (мс)	Максимальний час (мс)	Примітки
Класифікація інтенту (NLU)	45	120	Локальна обробка (Rasa)
HTTP-запит до OpenFoodFacts	850	2500	Залежить від зовнішнього API
Нормалізація даних та відповідь	15	30	Python-логіка (`ActionScanBarcode`)
Сумарний час реакції	~910	~2650	

Середній час реакції менше 1 секунди є комфортним для користувача. Випадки затримки до 2.5 секунд пов'язані з «холодним стартом» зовнішнього API або проблемами мережі. Для покращення UX у таких випадках бот налаштований виводити індикатор «typing...» (імітація набору тексту).

Надійність виконання сценаріїв (Task Completion Rate)

Було проаналізовано успішність проходження повного циклу «Сканування -> Вага -> Калорії».

- Успішно завершено: 92% діалогів.
- Помилки користувача: 5% (введення тексту замість ваги, нечіткі формулювання).
- Системні помилки: 3% (відсутність продукту в базі OpenFoodFacts).

На рис. 3.2 продемонстровано приклад виключної ситуації: реакція системи на невалідний або відсутній у базі штрих-код.



Рис. 3.3. Обробка виключної ситуації: реакція системи на невалідний або відсутній у базі штрих-код

Високий показник TCR (92%) свідчить про коректно побудовану логіку діалогу (Rasa Stories) та надійну роботу `SlotSet` механізмів.

Оцінка юзабіліті за методикою SUS

Для об'єктивізації оцінки зручності використання (usability) системи було проведено анкетування респондентів за стандартизованою шкалою System Usability Scale (SUS). Опитувальник містить 10 тверджень: 5 із позитивним формулюванням (непарні номери) та 5 із негативним (парні номери).

Кожен респондент оцінював твердження за шкалою Лайкерта від 1 (Повністю не згоден) до 5 (Повністю згоден). Усереднені результати опитування фокус-групи (12 осіб) наведено в табл. 3.2.

Таблиця 3.2

Зведені результати анкетування користувачів (SUS)

№	Питання анкети (SUS Item)	Тип питання	Середній бал (M), σ≈0.5
1	Я хотів би користуватися цією системою часто	Позитивне	4.2
2	Я вважаю систему невиправдано складною	Негативне	1.8
3	Я вважаю, що системою легко користуватися	Позитивне	4.6
4	Мені потрібна допомога технічного спеціаліста для використання системи	Негативне	1.3
5	Я вважаю, що різні функції системи добре інтегровані	Позитивне	3.9
6	Я вважаю, що в системі багато суперечностей (непослідовностей)	Негативне	2.1
7	Я вважаю, що більшість людей швидко навчиться користуватися системою	Позитивне	4.5
8	Система здалася мені дуже громіздкою у використанні	Негативне	1.7
9	Я почувався впевнено, використовуючи систему	Позитивне	4.3
10	Мені довелося вивчити багато нового, перш ніж я зміг почати роботу	Негативне	1.4

Примітка: Для позитивних питань ідеальним є бал 5, для негативних – 1.

Розрахунок інтегрального показника:

Для отримання фінального рейтингу використовувалася стандартна методика розрахунку SUS Score.

1. Для непарних питань ($Q_1, Q_3...$): $Score = ScalePosition - 1$.

2. Для парних питань ($Q_2, Q_4...$): $Score = 5 – ScalePosition$.
3. Сума всіх нормалізованих балів множиться на 2.5.

$$SUS_{total} = \left(\sum_{i=1}^{5} (M_{odd_i} – 1) + \sum_{j=1}^{5} (5 – M_{even_j}) \right) \times 2.5$$

Підставивши отримані середні значення:

- Сума позитивних аспектів: $(4.2-1) + (4.6-1) + (3.9-1) + (4.5-1) + (4.3-1) = 16.5$
- Сума негативних аспектів (інвертована): $(5-1.8) + (5-1.3) + (5-2.1) + (5-1.7) + (5-1.4) = 16.7$
- Загальна сума: 33.2

$$SUS_{score} = 33.2 \times 2.5 = \mathbf{83.0}$$

Інтерпретація результатів: отриманий показник 83.0 значно перевищує середньостатистичний рівень для програмних продуктів (який становить 68 балів).

Згідно з табл. 3.2, найвищі оцінки отримали пункти №3 (егкість використання, 4.6) та №7 (Швидкість навчання, 4.5). Це пояснюється використанням природно-мовного інтерфейсу (Chatbot), який є інтуїтивнішим за графічні меню. Дещо нижчий бал отримав пункт №5 (Інтеграція функцій, 3.9) та №6 (Суперечності, 2.1), що корелює з виявленими під час тестування затримками при отриманні відповіді від зовнішнього API, які користувачі могли сприймати як нестабільність роботи.

Високий показник легкості використання (4.6 за пунктом №3) та впевненості користувачів зумовлений мінімізацією когнітивного навантаження. Зокрема, для того, щоб користувачеві не доводилося вручну вводити текстові запити для типових дій, у системі було реалізовано графічний інтерфейс швидких команд, представлений на рис. 3.4.

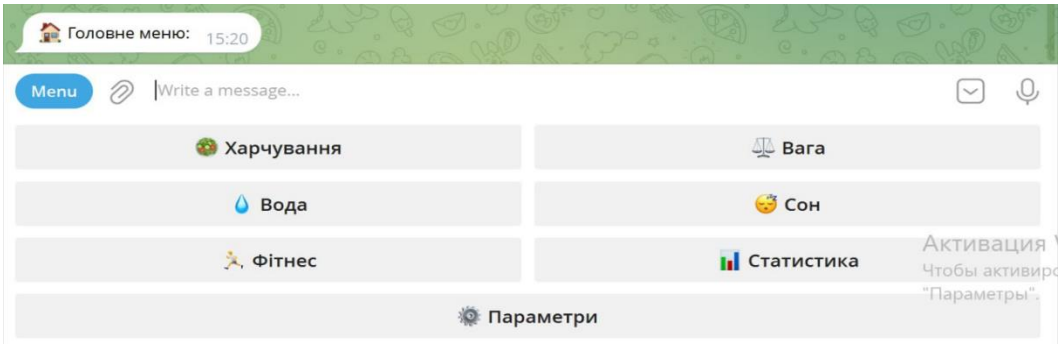


Рис. 3.4. Реалізація графічного інтерфейсу швидких команд (`Reply Keyboard`) для спрощення навігації

Візуальна інтерпретація результатів анкетування, що відображає кількісний розподіл відповідей респондентів за кожним із десяти параметрів шкали SUS, наведена на рис. 3.5.

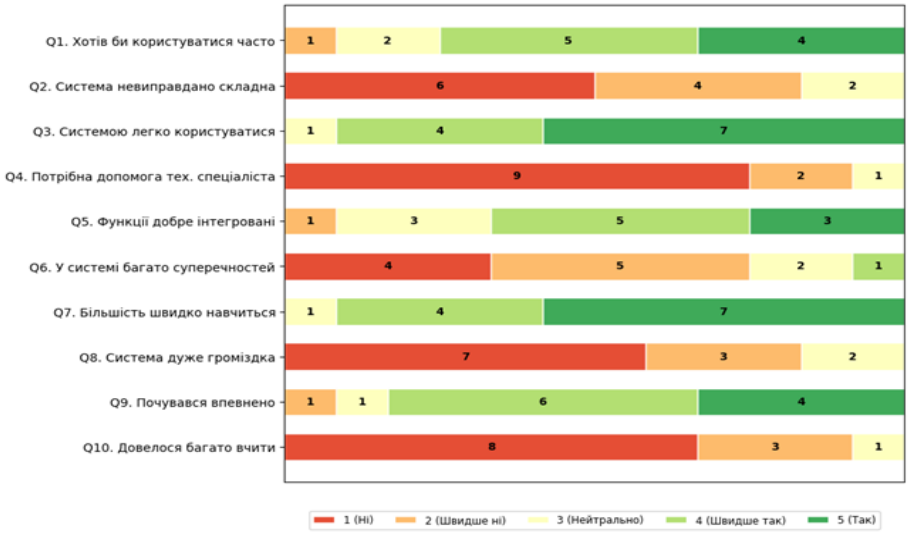


Рис. 3.5. Розподіл відповідей користувачів за шкалою SUS

Окремим етапом було перевірено математичну точність модуля калькуляції калорій, що представлено на рис. 3.6. Було порівняно результати ручного розрахунку (за даними на етикетці) та результати, видані ботом.

Приклад перевірки: продукт: Йогурт (штрих-код 300001); дані на упаковці: 89 ккал / 100 г; введена вага: 150 г; еталонне значення: (89×150)/100=133.5 ккал; відповідь бота: «133.5 ккал».

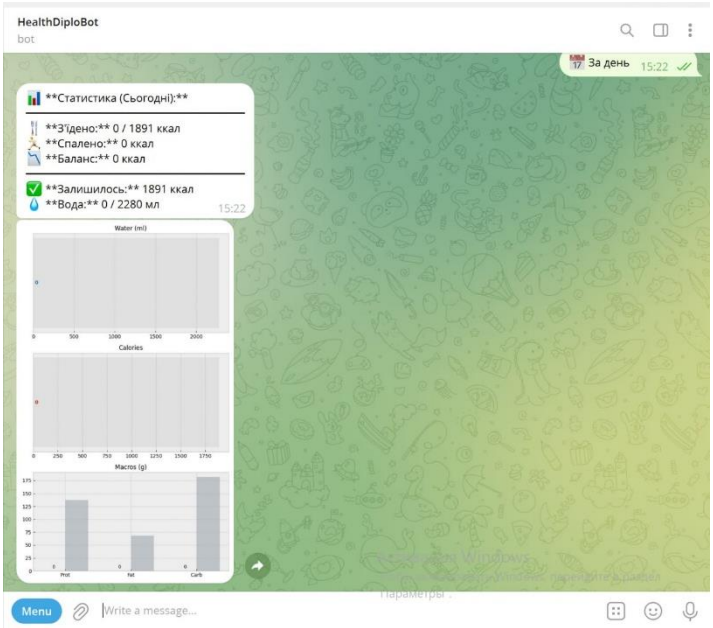


Рис. 3.6. Візуалізація агрегованих даних моніторингу: відображення спожитих калорій та водного балансу

Відхилень у розрахунках виявлено не було, що підтверджує коректність реалізації формули у методі run класу `ActionAskAmount`.

На основі проведеного тестування встановлено, що розроблена система демонструє високу стабільність роботи, забезпечуючи коректну обробку 92% запитів із середнім часом відгуку 910 мс. Отримана оцінка юзабіліті на рівні 83.0 балів за шкалою SUS, разом із підтвердженою математичною точністю розрахункових алгоритмів та інтеграційних модулів, свідчить про повну відповідність програмного продукту поставленим вимогам та його готовність до практичного впровадження.

Ознайомитися з розробленим чат-ботом «HeathDiploBot» можна за ідентифікатором @HeathDiploBot у мережі Telegram.

ВИСНОВКИ

Аналіз сучасних тенденцій цифровізації сфери охорони здоров’я засвідчив, що попри значне поширення технологій mHealth та IoT, наявні рішення характеризуються фрагментарністю функціоналу та недостатнім рівнем персоналізації, що знижує мотивацію користувачів до їх тривалого використання. Обґрунтовано, що найбільш ефективним інструментом подолання цих бар’єрів є розробка інтелектуального чат-бота, який завдяки використанню алгоритмів штучного інтелекту та природно-мовного інтерфейсу забезпечує консолідацію сервісів моніторингу, адаптивність рекомендацій та доступність для широких верств населення, відповідаючи актуальним вимогам превентивної медицини.

Аналіз еволюції діалогових систем засвідчив стійку тенденцію переходу від сценарних алгоритмів до інтелектуальних агентів на базі NLP та машинного навчання. Встановлено, що попри наявність глобальних гібридних рішень, вітчизняний ринок цифрового здоров’я характеризується відсутністю комплексних україномовних AI-чат-ботів для підтримки здорового способу життя, оскільки існуючі платформи (Helsi, DoctorBot) мають переважно інформаційно-довідковий характер. Для заповнення цієї ніші обґрунтовано вибір технологічного стека на основі Python, Rasa Framework та моделей трансформерів, що дозволяє реалізувати архітектуру з глибоким розумінням контексту, персоналізацією рекомендацій та адаптивним управлінням діалогом.

Здійснено системний аналіз предметної області та формалізовано вимоги до програмного комплексу моніторингу здорового способу життя. На основі визначених функціональних потреб, таких як мультимодальний ввід даних (NLP, Computer Vision) та гейміфікація, було спроєктовано сервіс-орієнтовану архітектуру (SOA) системи.

Обґрунтовано вибір технологічного стека, що включає мову програмування Python, фреймворк Rasa Open Source для локальної обробки природної мови, СУБД PostgreSQL для надійного зберігання даних та Telegram

Bot API як інтерфейс взаємодії. Розроблена модульна структура, що складається з комунікаційного шлюзу, сервера NLU, сервера бізнес-логіки та шару даних, забезпечує відповідність нефункціональним вимогам щодо масштабованості, безпеки та швидкодії. Спроєктована реляційна модель бази даних, нормалізована до третьої форми з елементами оптимізації для аналітики, створює надійний фундамент для ефективного функціонування інтелектуального помічника.

Спроєктовано інтелектуальну складову та функціональну логіку програмного комплексу. Обґрунтовано використання фреймворку Rasa Open Source для побудови конвеєра обробки природної мови, що включає семантичний аналіз (NLU) та гібридну стратегію керування діалогом (Rasa Core) на основі правил і форм. Реалізовано сервер бізнес-логіки, який інтегрує математичні моделі розрахунку енергетичних потреб (рівняння Міффліна-Сан Жеора), подійно-орієнтовану систему гейміфікації та алгоритми комп’ютерного зору для автоматизованої ідентифікації продуктів за штрих-кодами. Розроблена архітектура інтерфейсу за принципом Server-Driven UI та підтримка мультимодального вводу забезпечують гнучкість взаємодії з користувачем, високу точність обробки даних та адаптивність системи до різних сценаріїв використання.

Проведено комплексне експериментальне дослідження розробленої діалогової системи, спрямоване на оцінку її технічної ефективності, надійності та зручності для кінцевого користувача. Аналіз часових характеристик показав, що середній час реакції системи становить близько 910 мс, що є комфортним показником для ведення діалогу. Затримки, пов'язані з роботою зовнішнього API OpenFoodFacts (до 2.5 с), успішно нівелюються впровадженням UX-механізмів індикації набору тексту, що дозволяє утримувати увагу користувача. Високий показник успішності завершення діалогів (Task Completion Rate – 92%) підтверджує коректність побудови архітектури діалогу на базі Rasa та ефективність роботи слотів пам'яті. Система стійка до помилок користувача та здатна адекватно обробляти відсутність даних у зовнішній базі. Анкетування фокус-групи за методикою SUS (System Usability Scale) продемонструвало

високий рівень задоволеності користувачів. Інтегральна оцінка системи склала 83.0 бали, що згідно зі шкалою інтерпретації значно перевищує середньостатистичний показник для програмних продуктів (68 балів) і відповідає рівню «Відмінно». Респонденти високо оцінили легкість навчання та інтуїтивність природно-мовного інтерфейсу. Перевірка математичного модуля підтвердила абсолютну точність розрахунку енергетичної цінності порцій. Розбіжностей між еталонними значеннями та відповідями чат-бота не виявлено.

Економічна ефективність розробленого програмного продукту ґрунтується на мінімізації капітальних та операційних витрат завдяки використанню технологій з відкритим вихідним кодом (Open Source) та оптимізації архітектурних рішень. Застосування безкоштовного технологічного стека, що включає мову програмування Python, фреймворк Rasa та СУБД PostgreSQL, дозволило уникнути витрат на придбання пропрієтарних ліцензій, а вибір платформи Telegram як інтерфейсу користувача виключив необхідність високовартісної розробки, дизайну та підтримки окремих нативних мобільних застосунків для операційних систем iOS та Android. Додаткова економічна вигода досягнута шляхом інтеграції з відкритим API OpenFoodFacts, що забезпечило доступ до верифікованої бази даних продуктів без необхідності інвестицій у створення власного контенту чи купівлю доступу до комерційних реєстрів. Висока масштабованість розробленої системи дозволяє обслуговувати значну кількість користувачів з мінімальними витратами на серверну інфраструктуру, а соціально-економічний ефект проєкту полягає у забезпеченні широких верств населення доступним та безкоштовним інструментом для профілактики неінфекційних захворювань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бобрицька В. І., Гриньова М. В., Лазарєв Ф. В. Валеологія : підручник. Полтава : Скайтек, 2023. 368 с.

2. Бреусова С. В., Баранова І. І., Запорожська С. М. та ін. Дослідження споживчих властивостей «розумних» годинників та фітнес-браслетів. *Матеріали науково-практичної конференції НФаУ*. Харків : НФаУ. URL: https://dspace.nuph.edu.ua/bitstream/123456789/28121/1/98-101.pdf (дата звернення: 03.09.2025).

3. Всесвітня організація охорони здоров'я. Digital Health. [Електронний ресурс]. URL: https://www.who.int/health-topics/digital-health (дата звернення: 06.09.2025).

4. Грищенко С. М., Гулевич О. О. Правове забезпечення становлення інформаційного суспільства в Україні. *Сучасні проблеми правового, економічного та соціального розвитку держави* : тези доп. XIII Міжнар. наук.-практ. конф. (м. Вінниця, 22 листоп. 2024 р.). Вінниця : ХНУВС, 2024. С. 252–255. URL: https://dspace.univd.edu.ua/items/1a606fc2-a6e8-4867-9bc5-825beee02a92.

5. Гулевич О. О. Інтелектуальні чат-боти: трансформація комунікацій з допомогою технологій. *Економіка і бізнес: нові реалії, технологічні тренди, цифрові інновації* : зб. матеріалів Міжнар. наук.-практ. конф. (м. Львів — Ірпінь, 15 трав. 2025 р.). Львів : СПДФО Омельченко В. Г., 2025. С. 61–67. URL: https://financial.lnu.edu.ua/wp-content/uploads/2025/06/Zbirnyk-konferentsii-15-travnia-2025.pdf.

6. Гулевич О. О. Історія розвитку штучного інтелекту. *Інформаційні технології і автоматизація – 2024* : матеріали XVII Міжнар. наук.-практ. конф. (м. Одеса, 31 жовт. – 1 листоп. 2024 р.). Одеса : ОНТУ, 2024. С. 598–601. URL: https://www.researchgate.net/publication/387318817.

7. Гулевич О. О. Моделювання механізмів реєстрації користувачів для автентифікації в чат-боті здорового способу життя. *Universum*. 2025. № 22. С. 56–63. URL: https://archive.liga.science/index.php/universum/issue/view/july2025/143.

8. Гулевич О. О. Смарт-технології як засіб підтримки здорового способу життя. *Інформаційні технології і автоматизація – 2025* : матеріали XVIII Міжнар. наук.-практ. конф., м. Одеса, 30–31 жовт. 2025 р. Одеса : Вид-во ОНТУ, 2025. С. 1225–1227. URL: https://ontu.edu.ua/download/konfi/2025/Collection-of-abstracts-of-the-conference-ITIA-2025.pdf.

9. Гулевич О. О. Сучасні технології підтримки здорового способу життя. *Стан, досягнення і перспективи інформаційних систем і технологій* : матеріали XXV Всеукр. наук.-техн. конф. молодих вчених, аспірантів та студентів (м. Одеса, 17–18 квіт. 2025 р.). Одеса : Вид-во ОНТУ, 2025. С. 252–255. URL: https://ontu.edu.ua/download/konfi/2025/Conference_abstract-IT-2025.pdf.

10. Гулевич О. О. Технологічні тренди у сфері персоналізованої взаємодії: чат-боти та системи рекомендацій. *Universum*. 2025. № 25. С. 284–292. URL: https://archive.liga.science/index.php/universum/issue/view/october2025/152.

11. Гулевич О. О. Цифрові технології та креативна економіка як драйвери розвитку України. *Сучасні інформаційні технології в освіті і науці* : зб. матеріалів XVI Всеукр. наук.-практ. конф. (м. Умань, 10–11 квіт. 2025 р.). Умань : Візаві, 2025. С. 52–55. URL: https://if.uu.edu.ua/wp-content/uploads/2025/05/zbirnyk_kviten_2025.pdf.

12. Європейська комісія. Цифрове здоров’я та догляд. [Електронний ресурс]. URL: https://digital-strategy.ec.europa.eu/en/policies/digital-health-and-care (дата звернення: 03.09.2025).

13. Житинська М. О. Місце цифрових технологій та комунікацій у наданні соціально-психологічної підтримки молоді в умовах сьогодення. *Ментальне здоров’я*. 2025. № 2. С. 30–37. DOI: https://doi.org/10.32782/3041-2005/2025-2.6 (дата звернення: 05.09.2025).

14. Закон України «Про захист персональних даних». № 2297-VI від 01.06.2010. [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/2297-17#Text (дата звернення: 04.09.2025).

15. Закон України «Про охорону здоров'я». № 2801-XII від 19.11.1992. [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/2801-12#Text (дата звернення: 04.09.2025).

16. Квітка С., Миргородська М. Цифрова трансформація системи охорони здоров'я: фактори впливу на якість життя населення. *Аспекти публічного управління*. 2024. № 12(1). С. 14–21. DOI: https://doi.org/10.15421/152402 (дата звернення: 05.09.2025).

17. Коцаренко М. В., Адамович О. О., Адамович О. П. Телемедицина як інструмент оптимізації та удосконалення способів надання медичної допомоги населенню. *Буковинський медичний вісник*. 2023. Т. 27, № 1 (105). С. 73–78. DOI: https://doi.org/10.24061/2413-0737.27.1.105.2023.13.

18. Крупа А. Технологія чат-бот як чинник комп'ютерно-посередницької комунікації цифрового суспільства. *Humanities studies: Collection of Scientific Papers*. Запоріжжя : Видавничий дім «Helvetica», 2022. № 12 (89). С. 130–141. URL: http://humstudies.com.ua/article/view/266484/262502 (дата звернення: 05.09.2025).

19. Наказ МОЗ № 947 від 01.06.2021 «Про затвердження Вимог до електронної системи охорони здоров'я». [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/z0959-21#Text (дата звернення: 04.09.2025).

20. Наказ МОЗ № 1977 від 31.08.2020 «Про затвердження Вимог до ІС, що використовуються в системі охорони здоров'я». [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/z1113-20#Text (дата звернення: 04.09.2025).

21. Національна стратегія розвитку електронної охорони здоров'я в Україні до 2025 року. [Електронний ресурс]. URL: https://moz.gov.ua/article/reform-plan/nacionalna-strategija-rozvitku-elektronnoi-ohoroni-zdorovja (дата звернення: 04.09.2025).

22. Організація економічного співробітництва і розвитку (ОЕСР). Огляд здоров'я 2023: Показники ОЕСР. [Електронний ресурс]. URL: https://www.oecd.org/health/health-at-a-glance.htm (дата звернення: 06.09.2025).

23. Постанова КМУ № 411 від 25.04.2018 «Про деякі питання електронної системи охорони здоров'я». [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/411-2018-%D0%BF#Text (дата звернення: 04.09.2025).

24. Постанова КМУ № 753 від 02.10.2013 «Про затвердження Технічного регламенту щодо медичних виробів». [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/753-2013-%D0%BF#Text (дата звернення: 04.09.2025).

25. Приймак А. Ю., Гончаренко О. С. Сутність та потенціал фітнес-технологій у контексті формування здорового способу життя сучасного студентства. *Перспективи та інновації науки*. 2023. № 14 (32). С. 378–390. URL: http://perspectives.pp.ua/index.php/pis/issue/view/183/274 (дата звернення: 13.09.2025).

26. ЮНЕСКО. Штучний інтелект і здоров'я: можливості та виклики. [Електронний ресурс]. URL: https://en.unesco.org/artificial-intelligence/health (дата звернення: 04.09.2025).

27. Acosta J. N., Falcone G. J., Rajpurkar P., Topol E. J. Multimodal biomedical AI. *Nature Medicine*. 2022. Vol. 28. P. 1773–1784. DOI: https://doi.org/10.1038/s41591-022-01981-2.

28. Aggarwal A. et al. Chatbots for lifestyle behaviour change: a systematic review. *Journal of Medical Systems*. 2023. Vol. 47, No. 1. Art. 68. DOI: https://doi.org/10.1007/s10916-023-01960-7.

29. Centers for Disease Control and Prevention (CDC). Lark for Diabetes Prevention Program. 2021. [Electronic resource]. URL: https://www.cdc.gov/diabetes/prevention/index.html (date of access: 12.10.2025).

30. Credence Research. Chatbots Market. [Electronic resource]. URL: https://www.credenceresearch.com/report/chatbots-market (date of access: 08.10.2025).

31. Devlin J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*. 2019. P. 4171–4186. URL: https://aclanthology.org/N19-1423/ (date of access: 15.10.2025).

32. Estevam M. D. et al. Integrated Cardiovascular and Diabetes Management: A Scoping Review. *Journal of Diabetes Research*. 2025. Vol. 2025. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC12688643/ (date of access: 18.10.2025).

33. Ferguson T. et al. Effectiveness of wearable activity trackers to increase physical activity and improve health: a systematic review of systematic reviews and meta-analyses. *The Lancet Digital Health*. 2022. Vol. 4, No. 8. P. e615–e626. URL: https://www.thelancet.com/journals/landig/article/PIIS2589-75002200111-X/fulltext (date of access: 18.10.2025).

34. Guasch-Ferré M., Wittenbecher C., Palmnäs M., Ben-Yacov O. et al. Precision nutrition for cardiometabolic diseases. *Nature Medicine*. 2025. Vol. 31. P. 1444–1453. URL: https://pubmed.ncbi.nlm.nih.gov/40307513/ (date of access: 12.11.2025).

35. Hadi M. U., Qureshi R., Shah A., Irfan M. et al. Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects. *IEEE Access*. 2024. Vol. 12. P. 19562–19603. URL: https://www.techrxiv.org/users/618307/articles/682263-large-language-models-a-comprehensive-survey-of-its-applications-challenges-limitations-and-future-prospects (date of access: 09.11.2025).

36. Haleem A., Javaid M., Pratap Singh R., Suman R. Medical 4.0 Technologies for Healthcare: Features, Capabilities, and Applications. *Internet of Things and Cyber-Physical Systems*. 2022. Vol. 2. P. 12–30. URL:

https://www.scirp.org/reference/referencespapers?referenceid=3788503 (date of access: 10.10.2025).

37. Healthily AI Chatbot. [Electronic resource]. URL: https://www.livehealthily.com (date of access: 05.10.2025).

38. Hirosawa T. et al. Diagnostic Accuracy of Differential-Diagnosis Lists Generated by Generative Pre-trained Transformer 3 Chatbot for Clinical Vignettes with Common Chief Complaints: A Pilot Study. *International Journal of Environmental Research and Public Health*. 2023. Vol. 20, No. 4. Art. 3378. URL: https://www.semanticscholar.org/paper/Diagnostic-Accuracy-of-Differential-Diagnosis-Lists-Hirosawa-Harada/b026af1b2436fe064e02ca7a00b79713427bc3b2 (date of access: 12.10.2025).

39. Longhini J., Wiener D., Kolb S. et al. Wearable Devices to Improve Physical Activity and Reduce Sedentary Behaviour: An Umbrella Review. *Sports Medicine - Open*. 2024. Vol. 10. Art. 9. URL: https://www.researchgate.net/publication/377399165_Wearable_Devices_to_Improve_Physical_Activity_and_Reduce_Sedentary_Behaviour_An_Umbrella_Review (date of access: 17.10.2025).

40. Mazéas A., Duclos M., Pereira B., Chalabaev A. Evaluating the Effectiveness of Gamification on Physical Activity: Systematic Review and Meta-analysis of Randomized Controlled Trials. *J Med Internet Res*. 2022. Vol. 24(1). Art. e26779. URL: https://pubmed.ncbi.nlm.nih.gov/34982715/ (date of access: 07.10.2025).

41. Mönninghoff A., Kramer J. N., Hess A. J. et al. Long-term Effectiveness of mHealth Physical Activity Interventions: Systematic Review and Meta-analysis of Randomized Controlled Trials. *Journal of Medical Internet Research*. 2021. Vol. 23, No. 4. Art. e26699. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC8122296/ (date of access: 12.10.2025).

42. OpenAI GPT-3. [Electronic resource]. URL: https://openai.com/blog/openai-api (date of access: 02.10.2025).

43. Oracle. What is a Chatbot. [Electronic resource]. URL: https://www.oracle.com/chatbots/what-is-a-chatbot/ (date of access: 02.10.2025).

44. Pratap A., Neto E. C., Snyder P. et al. Indicators of retention in remote digital health studies: a cross-sectional study of 100,000 participants. *NPJ Digital Medicine*. 2020. Vol. 3. Art. 21. URL: https://pubmed.ncbi.nlm.nih.gov/32128451/ (date of access: 14.10.2025).

45. Rushender R., Logaraj M., Krishnamoorthy Y. Effectiveness of mobile phone applications for tobacco cessation: An umbrella review. *Drug and Alcohol Dependence*. 2024. Vol. 263. Art. 112425. URL: https://pubmed.ncbi.nlm.nih.gov/39216199/ (date of access: 12.10.2025).

46. SmarterChild. [Electronic resource]. URL: https://en.wikipedia.org/wiki/SmarterChild (date of access: 02.10.2025).

47. Turing A. M. Computing Machinery and Intelligence. *Mind*. 1950. Vol. 59, No. 236. P. 433–460.

48. Wang K. From ELIZA to ChatGPT: A brief history of chatbots and their evolution. *Proceedings of the 2023 International Conference on Machine Learning and Automation*. 2024. Vol. 39. P. 57–62. URL: https://www.researchgate.net/publication/378316730_From_ELIZA_to_ChatGPT_A_brief_history_of_chatbots_and_their_evolution (date of access: 10.10.2025).

49. Weizenbaum J. ELIZA – A Computer Program for the Study of Natural Language Communication between Man and Machine. *Communications of the ACM*. 1966. Vol. 9, No. 1. P. 36–45.

50. World Health Organization. Digital Health. [Electronic resource]. URL: https://www.who.int/health-topics/digital-health (date of access: 10.09.2025).

51. World Health Organization. Telemedicine: Opportunities and Developments. Geneva : WHO, 2010.

.

ДОДАТКИ

Додаток А

Ключові інтелектуальні чат-боти, які застосовуються для підтримки здорового способу життя, психічного благополуччя та первинної медичної допомоги

Healthily (LiveHealthily) – цифровий асистент, який використовує штучний інтелект для оцінки симптомів, надає персоналізовані поради щодо здоров'я та пропонує інтерактивні плани ЗСЖ. Працює на основі NHS-керівництв [https://www.livehealthily.com].

Florence – простий бот-нагадувач, який допомагає приймати ліки вчасно, відстежує вагу, фізичну активність, тиск. Доступний у месенджерах, має інтуїтивний інтерфейс [https://florence.chat].

Wysa – чат-бот для ментального здоров'я, який використовує когнітивно-поведінкову терапію (CBT) та вправи для зниження тривожності. Має анонімний режим і підтримує користувача щодня [https://www.wysa.io].

Replika – віртуальний емоційний помічник, що імітує дружню бесіду. Розвиває емоційний інтелект, підтримує користувача, навчається на базі діалогів [https://replika.ai].

Ada Health – медичний чат-бот із клінічною базою знань. Збирає інформацію про симптоми, оцінює їх і формує попередні висновки про стан здоров'я [https://ada.com].

Lark – AI-коуч для профілактики діабету, гіпертонії, ожиріння. Пропонує рекомендації на основі аналізу даних про поведінку та стан користувача [https://www.lark.com].

Додаток Б

Коротка характеристика кожного з українських чат-ботів, що підтримують здоровий спосіб життя

Helsi-бот – інформаційний чат-бот, що забезпечує доступ до електронної медичної картки, дозволяє записатися до лікаря, отримати нагадування про візити та переглядати історію вакцинацій. Він працює на платформах Telegram і Viber, інтегрований з електронною системою охорони здоров'я (ЕСОЗ). Бот побудований за rule-based логікою, без використання штучного інтелекту [https://helsi.me].

Лікар Бот (DoctorBot) – чат-бот, створений за підтримки МОЗ і USAID для інформування населення про COVID-19, вакцинацію та основи самодіагностики. Працює в Telegram і Facebook Messenger. Його функції частково мотиваційні: бот допомагає користувачам оцінити ризики та прийняти рішення щодо звернення до лікаря. Логіка побудована на сценаріях з обмеженою адаптацією до користувача [https://t.me/COVID19_Ukraine_Bot].

MedBot – Telegram-бот, що надає довідкову інформацію про найближчі аптеки, лікарні та клініки, а також дозволяє перевіряти наявність і ціни на ліки. Це класичний інформаційний бот із жорстко заданими командами без елементів персоналізації чи машинного навчання. Орієнтований на швидке надання медичної інфраструктурної інформації [https://t.me/medbot_ua_bot].

Mindly – це гібридний інструмент підтримки ментального здоров'я, що поєднує мобільний застосунок та чат-бот. Розроблений українськими психологами, він пропонує дихальні вправи, техніки самозаспокоєння, рекомендації з подолання тривоги й стресу. Бот містить адаптивні елементи логіки і частково використовує AI для налаштування контенту під користувача [https://mindly.space].

ЗдоровІнфо (ZdorovInfoBot) – Telegram-бот, створений Міністерством охорони здоров'я України за підтримки міжнародних партнерів. Його мета – надання перевіреної інформації про вакцинацію, профілактику захворювань, здорове харчування та загальні принципи ЗСЖ. Бот побудований за rule-based принципами, з регулярним оновленням інформації, однак без можливостей персоналізації чи AI [https://t.me/ZdorovInfoBot].

Додаток В

Моделі архітектури та алгоритмів функціонування чат-бота

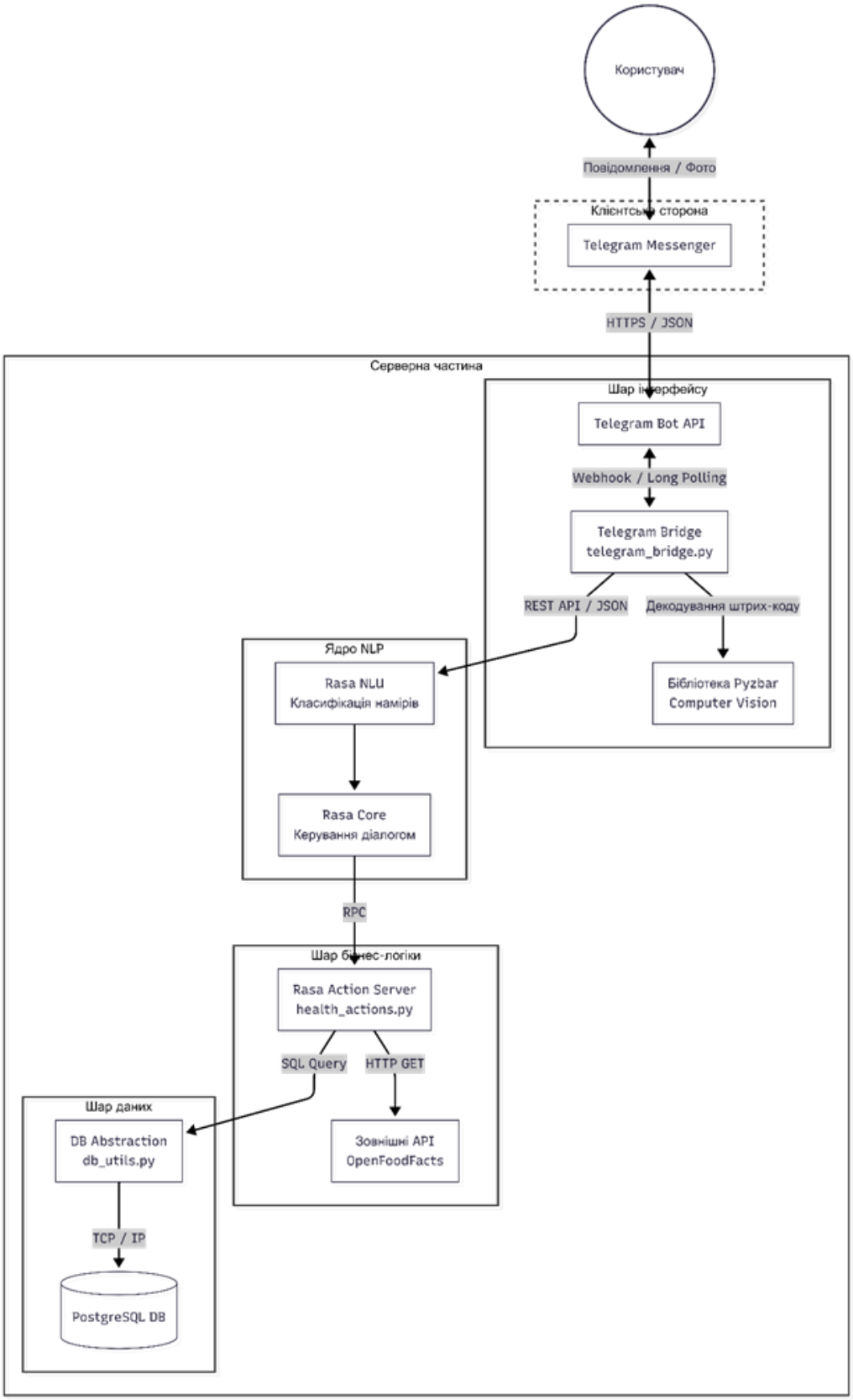


Рис. В.1. Структурна схема взаємодії компонентів системи

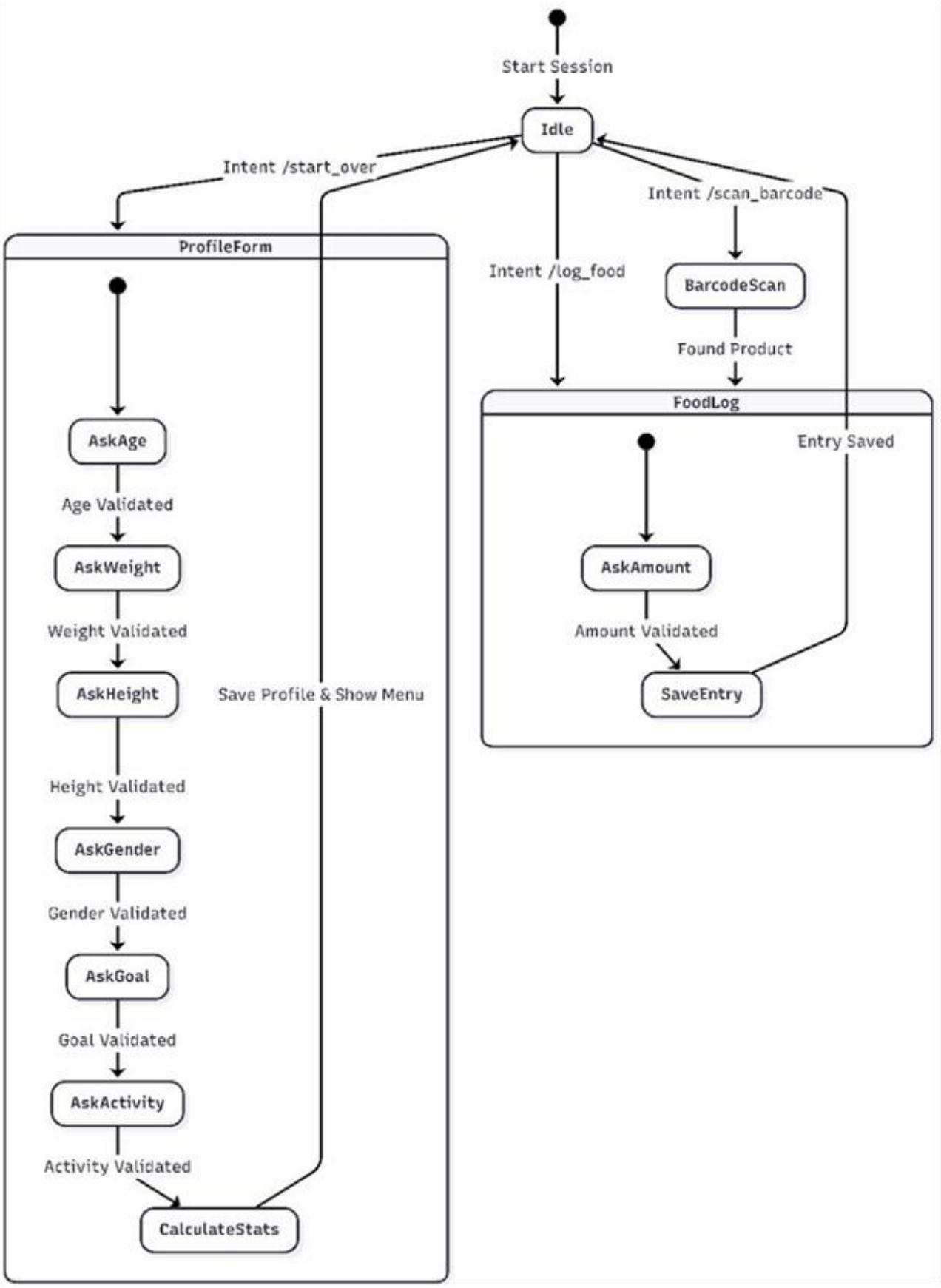


Рис. В.2. Діаграма станів діалогу (State Machine Diagram)

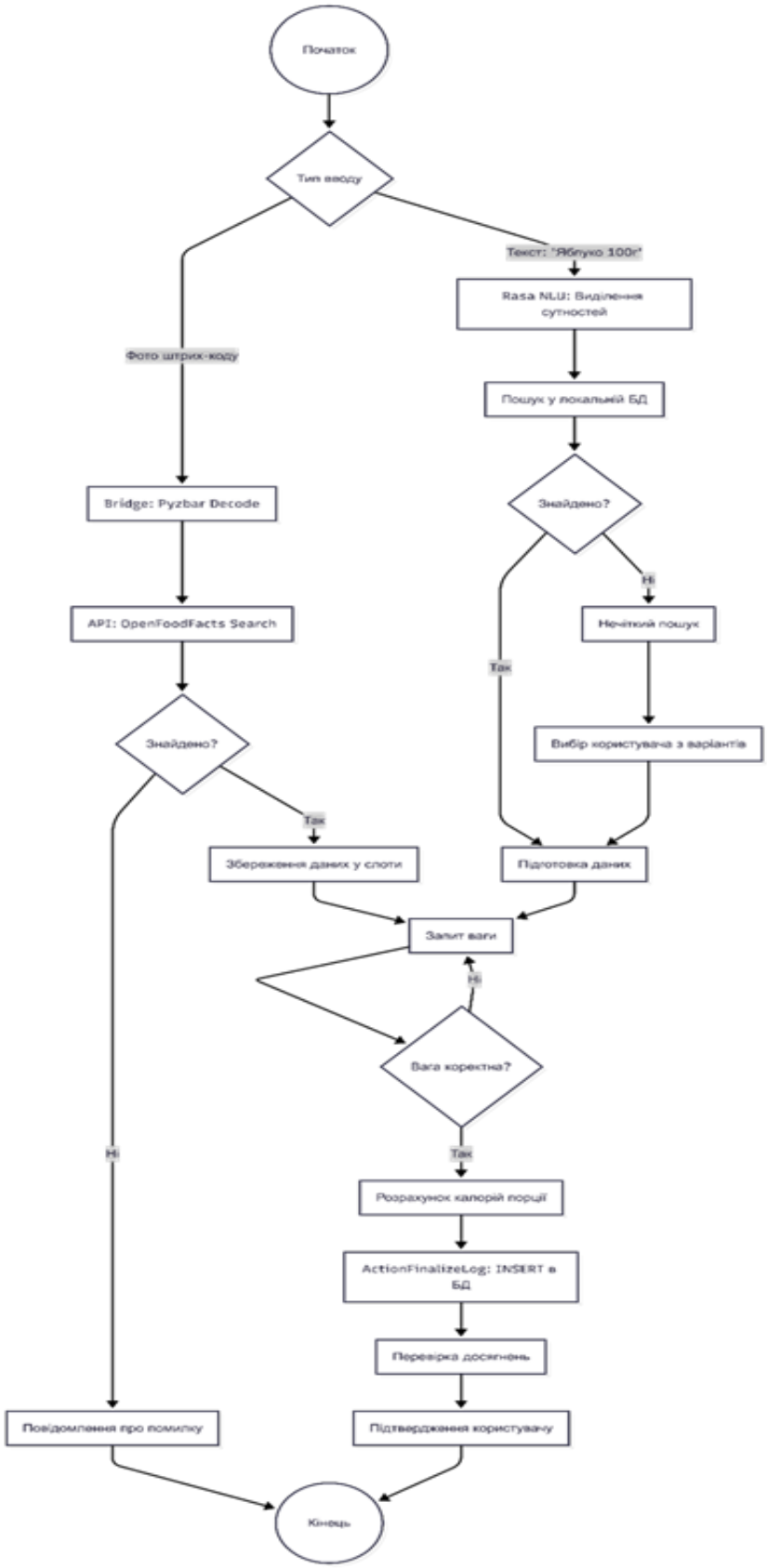


Рис. В.3. Діаграма діяльності процесу додавання продукту

Додаток Г

Лістинг програми

```
# ============================================================
#  1. SYSTEM & SESSION ACTIONS
# ============================================================
class ActionSessionStart(Action):
    def name(self) -> str: return "action_session_start"

    def run(self, dispatcher: CollectingDispatcher, tracker: Tracker,
domain: dict):
        user_id = str(tracker.sender_id)
        profile = get_user_profile(user_id)

        events = [SessionStarted()]

        if profile:
            lang = profile.get("language", "en")
            events.extend([
                SlotSet("language", lang),
                SlotSet("age", profile.get("age")),
                SlotSet("weight", profile.get("weight")),
                SlotSet("height", profile.get("height")),
                SlotSet("gender", profile.get("gender")),
                SlotSet("goal", profile.get("goal", "maintain")),
                SlotSet("daily_calories",   profile.get("daily_calories",
0.0)),
                SlotSet("daily_water_ml",   profile.get("daily_water_ml",
0.0)),
            ])
        else:
            dispatcher.utter_message(text="Привіт!  Для  початку  роботи
заповніть, будь ласка, анкету.")

        events.append(ActionExecuted("action_listen"))
        return events

class ActionCheckProfile(Action):
    def name(self) -> Text: return "action_check_profile"
    def run(self, dispatcher, tracker, domain):
        user_id = str(tracker.sender_id)
        profile = get_user_profile(user_id)
        if profile and profile.get("age"):
            lang = profile.get("language", "en")
            return               [SlotSet("language",               lang),
FollowupAction("action_show_main_menu")]
        return [FollowupAction("action_start_registration")]

class ActionStartRegistration(Action):
    def name(self) -> Text: return "action_start_registration"
    def run(self, dispatcher, tracker, domain):
```

```
        ActionResetProfile().run(dispatcher, tracker, domain)

dispatcher.utter_message(response="utter_start_language_onboarding")
        return []

class ActionSetLanguage(Action):
    def name(self) -> Text: return "action_set_language"

    def run(self, dispatcher, tracker, domain):
        user_id = str(tracker.sender_id)

        # 1. Визначаємо мову
        lang = next(tracker.get_latest_entity_values("language"), None)
        if not lang:
            text = tracker.latest_message.get("text", "").lower()
            if any(w in text for w in ["ua", "ukr", "укр", "україн"]):
lang = "uk"
            elif any(w in text for w in ["en", "eng", "англ", "english"]):
lang = "en"

        if not lang: lang = tracker.get_slot("language")

        # Нормалізація
        if lang: lang = str(lang).strip().lower()
        if lang not in ("uk", "en"): lang = "en"

        # 2. ЗБЕРЕЖЕННЯ В БД (З "захисним" профілем)
        try:
            current_profile = get_user_profile(user_id) or {}
            safe_profile = {
                "age": None, "weight": None, "height": None, "gender":
None, "goal": None,
                "daily_calories": 0, "daily_water_ml": 0,
"daily_proteins_g": 0,
                "daily_fats_g": 0, "daily_carbs_g": 0
            }
            safe_profile.update(current_profile)
            safe_profile["language"] = lang
            upsert_user_profile(user_id, safe_profile)
        except Exception as e:
            print(f"❌ Error saving language: {e}")

        # 3. Встановлюємо слот
        events = [SlotSet("language", lang)]

        # Повідомляємо про зміну
        msg_selected = "Обрано українську мову 🇺🇦" if lang == "uk" else
"English language selected 🇬🇧"
        dispatcher.utter_message(text=msg_selected)

        if tracker.active_loop: return events
```

```
        # 4. ВІДРАЗУ ПОКАЗУЄМО ГОЛОВНЕ МЕНЮ НОВОЮ МОВОЮ
        age = tracker.get_slot("age")
        has_profile = False
        if not age:
             p = get_user_profile(user_id)
             if p and p.get("age"):
                 events.append(SlotSet("age", p["age"]))
                 has_profile = True

        if age or has_profile:
            if lang == 'uk':
                text = "🏠 Головне меню:"
                buttons = [
                    {"title":      "🥗      Харчування",      "payload":
"/menu_nutrition"},
                    {"title": "⚖️ Вага", "payload": "/menu_weight"},
                    {"title": "💧 Вода", "payload": "/menu_water"},
                    {"title": "😴 Сон", "payload": "/menu_sleep"},
                    {"title": "🏃 Фітнес", "payload": "/menu_fitness"},
                    {"title": "📊 Статистика", "payload": "/menu_stats"},
                    {"title": "⚙️ Параметри", "payload": "/options"}
                ]
            else:
                text = "🏠 Main menu:"
                buttons = [
                    {"title":      "🥗      Nutrition",      "payload":
"/menu_nutrition"},
                    {"title": "⚖️ Weight", "payload": "/menu_weight"},
                    {"title": "💧 Water", "payload": "/menu_water"},
                    {"title": "😴 Sleep", "payload": "/menu_sleep"},
                    {"title": "🏃 Fitness", "payload": "/menu_fitness"},
                    {"title": "📊 Stats", "payload": "/menu_stats"},
                    {"title": "⚙️ Options", "payload": "/options"}
                ]
            dispatcher.utter_message(text=text, buttons=buttons)
        else:
            if lang == "uk":
                 msg = "Бажаєте заповнити профіль?"
                 buttons  =  [{"title":  "📝  Так,  почати",  "payload":
"/edit_profile"}]
            else:
                 msg = "Do you want to setup profile?"
                 buttons  =  [{"title":  "📝  Yes,  start",  "payload":
"/edit_profile"}]
            dispatcher.utter_message(text=msg, buttons=buttons)

        return events
```

```
# ============================================================
#  2. PROFILE ACTIONS
# ============================================================

class ActionResetProfile(Action):
    def name(self) -> Text: return "action_reset_profile"
    def run(self, dispatcher, tracker, domain):
        current_lang = _get_language(tracker)
        slots_to_reset = ["age", "weight", "height", "gender", "goal",
"daily_calories", "daily_water_ml", "daily_proteins_g", "daily_fats_g",
"daily_carbs_g"]
        events = [SlotSet(s, None) for s in slots_to_reset]
        events.append(SlotSet("language", current_lang))
        return events

class ActionAskProfileConfirmation(Action):
    def name(self) -> Text: return "action_ask_profile_confirmation"
    def run(self, dispatcher, tracker, domain):
        lang = _get_language(tracker)
        age = tracker.get_slot("age")
        weight = tracker.get_slot("weight")
        height = tracker.get_slot("height")
        gender = tracker.get_slot("gender")
        goal = tracker.get_slot("goal")

        try:
            targets = calculate_daily_targets(
                float(str(age).replace(',', '.')),
                float(str(weight).replace(',', '.')),
                float(str(height).replace(',', '.')),
                gender, goal
            )
            cal_preview = targets["daily_calories"]
        except:
            cal_preview = "?"

        if lang == "uk":
            msg = (f"📝 **Перевірте дані:**\n• Вік: {age}\n• Вага:
{weight} кг\n• Зріст: {height} см\n• Стать: {gender}\n• Ціль: {goal}\n\n🎯
Норма: **{cal_preview} ккал**\nЗберегти?")
            buttons = [{"title": "✅ Зберегти", "payload":
"/confirm_save_profile"}, {"title": "❌ Скасувати", "payload":
"/discard_profile"}]
        else:
            msg = (f"📝 **Review:**\n• Age: {age}\n• Weight: {weight}
kg\n• Height: {height} cm\n• Gender: {gender}\n• Goal: {goal}\n\n🎯
Target: **{cal_preview} kcal**\nSave?")
            buttons = [{"title": "✅ Save", "payload":
"/confirm_save_profile"}, {"title": "❌ Cancel", "payload":
"/discard_profile"}]
```

```
        dispatcher.utter_message(text=msg, buttons=buttons)
        return []

class ActionDiscardProfile(Action):
    def name(self) -> Text: return "action_discard_profile"
    def run(self, dispatcher, tracker, domain):
        lang = _get_language(tracker)
        slots_to_reset = ["age", "weight", "height", "gender", "goal"]
        events = [SlotSet(s, None) for s in slots_to_reset]
        msg = "❌ Скасовано." if lang == "uk" else "❌ Discarded."
        dispatcher.utter_message(text=msg)
        events.append(FollowupAction("action_show_options_menu"))
        return events

class ActionSaveProfile(Action):
    def name(self) -> Text: return "action_save_profile"
    def run(self, dispatcher, tracker, domain):
        user_id = str(tracker.sender_id)
        lang = _get_language(tracker)

        age = tracker.get_slot("age")
        weight = tracker.get_slot("weight")
        height = tracker.get_slot("height")
        gender = tracker.get_slot("gender")
        goal = tracker.get_slot("goal")

        if not (age and weight and height and gender and goal): return []

        try:
            if isinstance(age, list): age = age[0]
            if isinstance(weight, list): weight = weight[0]
            if isinstance(height, list): height = height[0]

            age_f = float(str(age).replace(',', '.'))
            weight_f = float(str(weight).replace(',', '.'))
            height_f = float(str(height).replace(',', '.'))
        except:
            dispatcher.utter_message(text="Format error.")
            return []

        targets = calculate_daily_targets(age_f, weight_f, height_f,
gender, goal)

        profile = {
            "language": lang, "age": int(age_f), "weight": weight_f,
"height": height_f,
            "gender": str(gender), "goal": str(goal),
            **targets
        }
        upsert_user_profile(user_id, profile)
```

```
        log_weight_entry(user_id, weight_f)

        if lang == "uk":
            dispatcher.utter_message(text=f"✅ Профіль збережено! Норма:
{targets['daily_calories']} ккал.")
        else:
            dispatcher.utter_message(text=f"✅       Saved!       Target:
{targets['daily_calories']} kcal.")

        events = [SlotSet("language", lang)]
        for k, v in targets.items():
            events.append(SlotSet(k, v))

        trigger_daily = datetime.utcnow() + timedelta(hours=24)

events.append(ReminderScheduled(intent_name="ACTION_DAILY_STATS_CHECK",
trigger_date_time=trigger_daily,                name="daily_stats_check",
kill_on_user_message=False))
        events.extend(_get_reminders_events(user_id))
        events.append(FollowupAction("action_show_main_menu"))
        return events

class ValidateProfileForm(FormValidationAction):
    def name(self) -> Text: return "validate_profile_form"
    def      validate_age(self,      slot_value:      Any,      dispatcher:
CollectingDispatcher,  tracker:  Tracker,  domain:  Dict[Text,  Any])  ->
Dict[Text, Any]:
        return  self._validate_number(slot_value,  dispatcher,  "age",  10,
100)
    def     validate_weight(self,     slot_value:     Any,     dispatcher:
CollectingDispatcher,  tracker:  Tracker,  domain:  Dict[Text,  Any])  ->
Dict[Text, Any]:
        return self._validate_number(slot_value, dispatcher, "weight", 30,
300)
    def     validate_height(self,     slot_value:     Any,     dispatcher:
CollectingDispatcher,  tracker:  Tracker,  domain:  Dict[Text,  Any])  ->
Dict[Text, Any]:
        return  self._validate_number(slot_value,  dispatcher,  "height",
100, 250)
    def     validate_gender(self,     slot_value:     Any,     dispatcher:
CollectingDispatcher,  tracker:  Tracker,  domain:  Dict[Text,  Any])  ->
Dict[Text, Any]:
        val = str(slot_value).lower().strip()
        if val in ["ч", "м", "m", "male", "чоловік", "хлопець", "man"]:
return {"gender": "m"}
        elif  val  in  ["ж",  "f",  "female",  "жінка",  "дівчина",  "woman"]:
return {"gender": "f"}
        else:
            dispatcher.utter_message(text="Будь  ласка,  вкажіть  стать
(ч/ж).")
            return {"gender": None}
```

```
    def _validate_number(self, value, dispatcher, slot_name, min_val,
max_val):
        try:
            if isinstance(value, list): value = value[0]
            clean_val = float(str(value).replace(",", "."))
            if min_val <= clean_val <= max_val: return {slot_name:
clean_val}
            else:
                dispatcher.utter_message(text=f"Значення має бути між
{min_val} та {max_val}.")
                return {slot_name: None}
        except:
            dispatcher.utter_message(text="Будь ласка, введіть число.")
            return {slot_name: None}

class ActionViewProfile(Action):
    def name(self) -> Text: return "action_view_profile"
    def run(self, dispatcher, tracker, domain):
        user_id = str(tracker.sender_id)
        lang = _get_language(tracker)
        profile = get_user_profile(user_id)

        if not profile or not profile.get("age"):
            msg = "Анкета не заповнена." if lang == 'uk' else "Profile not
set."
            dispatcher.utter_message(text=msg)
            return []

        if lang == 'uk':
            msg = (f"👤 **Твій профіль:**\n"
                   f"▫ Вік: {profile['age']}\n▫ Вага: {profile['weight']}
кг\n"
                   f"▫ Ціль: {profile['goal']}\n🎯 Норма:
{profile['daily_calories']:.0f} ккал")
            buttons = [{"title": "✏ Редагувати", "payload":
"/edit_profile"}, {"title": "⬅ Назад", "payload": "/options"}]
        else:
            msg = (f"👤 **Profile:**\n"
                   f"▫ Age: {profile['age']}\n▫ Weight:
{profile['weight']} kg\n"
                   f"▫ Goal: {profile['goal']}\n🎯 Target:
{profile['daily_calories']:.0f} kcal")
            buttons = [{"title": "✏ Edit", "payload": "/edit_profile"},
{"title": "⬅ Back", "payload": "/options"}]

        dispatcher.utter_message(text=msg, buttons=buttons)
        return []
```