Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

Бакалаврська кваліфікаційна робота на тему:

«ІНФОРМАЦІЙНА СИСТЕМА АВТОМАТИЗОВАНОГО ПІДБОРУ ДІЄТИ ТА ХАРЧОВОГО ПЛАНУ»

Виконав: студент 4 курсу, групи 2ІСТ-21б
спеціальності 126 - «Інформаційні системи та технології»
_____Ілля ПЕРОВ
Керівник: Ph.D., ст. викл. каф. САІТ
_____ Ольга ВОЙЦЕХОВСЬКА
«05» ____06_____ 2025 р.
Рецензент: к.т.н., доцент кафедри КН
_____ Ігор АРСЕНЮК
«12» ____06_____ 2025 р.

Допущено до захисту

Завідувач кафедри САІТ

_____д.т.н., проф. Віталій МОКІН

«01» ____06_____2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – перший (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність 126 – «Інформаційні системи та технології»
Освітньо-професійна програма – Прикладні інформаційні технології

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
д.т.н., проф. Віталій МОКІН
"12" 05 2025 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Перову Іллі Олександровичу

1. Тема роботи: «Інформаційна система автоматизованого підбору дієти та харчового плану»
керівник роботи: Войцеховська О. О., Ph.D., ст. викл. каф. САІТ
затверджені наказом ВНТУ від «20» 05 2025 року №97
2. Термін подання студентом роботи 31.05.2025 р.
3. Вихідні дані до роботи:
1) Дані про калорійність харчових продуктів;
2) Дані про рецепти страв здорового харчування.
4. Зміст текстової частини:
1) Аналіз предметної області.
2) Вибір оптимальних інформаційних технологій.
3) Розроблення інформаційної системи для автоматизованого підбору дієти та харчового плану.
5. Перелік ілюстративного матеріалу:
1) Архітектура інформаційної системи;
2) Блок-схема алгоритму роботи додатку;
3) Структурна схема функціонування мобільного додатку;
4) Діаграма випадків використання додатку;
5) Діаграма послідовності автоматичного підбору дієтичного плану;
6) Діаграма класів додатку;
7) Інтерфейс додатку.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 28.03.2025р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	01.04	15.04	
2	Огляд аналогів	15.04	30.04	
3	Вибір оптимальних інформаційних технологій	01.05	10.05	
4	Розроблення інформаційної системи автоматизованого підбору дієти та харчового плану	10.05	20.05	
5	Оформлення матеріалів до захисту БКР	20.05	30.05	

Студент Ілля ПЕРОВ

Керівник роботи Ольга ВОЙЦЕХОВСЬКА

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 81 сторінок формату А4, на яких є 26 рисунків, список використаних джерел містить 22 найменувань.

Метою роботи є розроблення інформаційної системи автоматизованого підбору дієти та харчового плану.

В роботі наведено розроблення інформаційної системи у вигляді мобільного додатку для платформ Android та iOS, за допомогою мови програмування TypeScript у поєднанні з фреймворком React та Capacitor. Розроблено дизайн та функціонал мобільного додатку, включаючи механізми інтерактивної взаємодії з користувачем для персоналізованого підбору раціону відповідно до його потреб. Реалізована система інтегрує AI-компонент Google GenAI, що дає змогу динамічно коригувати раціон у випадку відхилення від запропонованого меню без зміни загальної структури розрахунків. Розроблена інформаційна система забезпечує можливість автоматизованого планування харчування з використанням сучасних AI-технологій.

Ключові слова: інформаційна система, мобільний додаток, автоматизований підбір дієти, харчовий план, React, TypeScript, штучний інтелект.

ABSTRACT

The bachelor's thesis consists of 81 A4 pages with 26 figures and a list of references containing 22 titles.

The aim of the work is to develop an information system for automated selection of a diet and food plan.

The paper presents the development of an information system in the form of a mobile application for Android and iOS platforms, using the TypeScript programming language in combination with the React and Capacitor frameworks. The design and functionality of the mobile application, including mechanisms for interactive interaction with the user for personalised dietary selection according to their needs, have been developed. The implemented system integrates the AI component of Google GenAI, which allows for dynamic dietary adjustments in case of deviations from the proposed menu without changing the overall calculation structure. The developed information system enables automated meal planning using modern AI technologies.

Keywords: information system, mobile application, automated diet selection, meal plan, React, TypeScript, artificial intelligence.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному суспільстві, де рівень стресу, малорухливий спосіб життя і нераціональне харчування стали поширеними явищами, питання підтримки здорового способу життя, правильного харчування та контролю ваги набувають особливого значення. Все більше людей прагнуть дотримуватись збалансованої дієти, однак через брак часу, знань або професійної консультації не можуть самостійно сформувати ефективний і безпечний раціон. З іншого боку, зростання обсягів даних у сфері нутриціології, розвиток технологій штучного інтелекту та персоналізованої медицини створюють передумови для автоматизації процесу підбору дієти. Інформаційні системи, що враховують індивідуальні характеристики користувача (вік, стать, вага, фізична активність, стан здоров’я), дозволяють формувати персоналізовані харчові плани з урахуванням потреб організму. Розробка такої системи сприятиме покращенню якості життя користувачів, профілактиці хронічних захворювань, підвищенню ефективності фітнес- або лікувальних програм. Саме тому актуальною є задача по створенню інформаційної системи для автоматизованого підбору дієти та харчового плану.

Мета і задачі дослідження. Метою дослідження є розроблення інформаційної системи автоматизованого підбору дієти та харчового плану.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

– провести аналіз предметної області та існуючих рішень щодо автоматизованого підбору дієти та харчового плану;

– здійснити вибір оптимальних інформаційних технологій для розробки інформаційної системи;

– здійснити вибір оптимальної IT-інфраструктури;

– розробити інформаційну систему автоматизованого підбору дієти та харчового плану у вигляді мобільного додатку;

– провести тестування розробленого додатку для оцінки його ефективності та надійності.

Об’єктом дослідження є процес розроблення інформаційної системи автоматизованого підбору дієти та харчового плану.

Предметом дослідження є методи та програмні засоби розроблення інформаційної системи автоматизованого підбору дієти та харчового плану.

Публікації. За результатами даної роботи були опубліковані тези «Інформаційна система автоматизованого підбору дієти та харчового плану» на Міжнародній науково-практичній інтернет-конференції студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2024-2025 рр.) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

У сучасних умовах, коли проблеми здорового харчування набувають все більшої актуальності, створення ефективних інструментів для планування раціону стає особливо важливим. Згідно з дослідженням, опублікованим у журналі «Український журнал клінічної та лабораторної медицини» (2023), понад 60% населення України стикаються з труднощами при дотриманні збалансованого харчування через відсутність персоналізованих підходів [2]. Актуальність роботи підтверджується даними МОЗ України, які свідчать про зростання захворювань, пов'язаних з неправильним харчуванням [3].

Сучасний світ характеризується високими темпами життя, постійною зайнятістю та зростаючим інтересом до здорового харчування. Однак значна частина людей стикається з проблемою браку часу для самостійного планування раціону, аналізу продуктів та розрахунку необхідної кількості калорій і макронутрієнтів. У цьому контексті розробка автоматизованої інформаційної системи для підбору дієти та харчового плану набуває особливої актуальності.

Здоровий спосіб життя набуває все більшого значення, і правильне харчування є його невід’ємною частиною, проте складання раціону залишається складним завданням, що потребує спеціальних знань у дієтології. Згідно з даними міжнародних досліджень, велика кількість людей споживає невідповідну кількість калорій та макронутрієнтів, що призводить до проблем зі здоров’ям, таких як ожиріння, дефіцит нутрієнтів або порушення метаболізму. Автоматизована система, яка враховує індивідуальні фізіологічні параметри користувача та його дієтичні вподобання, може значно спростити процес планування харчування.

Сучасні технології дозволяють створювати більш точні та персоналізовані дієтичні рекомендації завдяки інтеграції алгоритмічних розрахунків та штучного інтелекту. Використання класичних методів визначення добової потреби у

калоріях (зокрема формули Гарріса-Бенедикта) [4] у поєднанні з AI-технологіями дозволяє адаптувати харчування до індивідуальних потреб кожного користувача. Розроблений додаток інтегрує штучний інтелект для корекції меню, що забезпечує гнучкість системи та зменшує необхідність ручного редагування запропонованих страв.

Розвиток технологій дозволяє мінімізувати складність планування раціону завдяки автоматизованим алгоритмам обчислення калорійності та нутрієнтів. Традиційне планування харчування передбачає ручне розрахування калорійності та нутрієнтів, що може бути складним і неточним без належних знань. Автоматизований підхід дозволяє мінімізувати ризик помилок та оптимізувати процес формування раціону. Розробка мобільного додатку спрямована на створення зручної та швидкої системи, яка в режимі реального часу аналізує введені дані та надає точні результати відповідно до цілей користувача.

Важливим фактором сучасних технологічних рішень є їх доступність для широкої аудиторії. Використання кросплатформених технологій (React [5], TypeScript [6], Capacitor [7]) дозволяє створити єдиний застосунок для мобільних пристроїв з можливістю швидкого оновлення та впровадження нових функцій. Завдяки цьому розроблений додаток буде доступним для різних типів користувачів – незалежно від їх платформи або рівня технічної підготовки.

Автоматизовані системи управління харчуванням мають значний потенціал для впровадження у сфері охорони здоров’я, спортивного харчування та нутриціології. Далі може бути інтеграція розширених AI-модулів для ще точнішої персоналізації харчового плану, а також інтеграція з базами даних продуктів для аналізу їх харчової цінності. Таким чином, розроблений мобільний додаток може стати не лише персональним інструментом для користувачів, а й частиною загальної екосистеми здорового способу життя.

1.2 Сучасні підходи до планування харчування

Сучасні підходи до планування харчування є результатом інтеграції класичних дієтичних методик із новітніми цифровими технологіями та алгоритмами аналізу даних (рис.1.1). Завдяки можливостям машинного навчання та обробці великих обсягів інформації створюються персоналізовані плани, які враховують індивідуальні особливості користувача: анатомічні показники, фізичну активність, генетичні чинники, способи життя та харчові вподобання. Такий підхід сприяє не лише оптимізації добової калорійності, але й профілактиці хронічних захворювань, адже раціон формується з урахуванням постійних змін у параметрах організму.

Рисунок 1.1 – Сучасні підходи до планування харчування з використанням інформаційних технологій

Історично планування харчування базувалося на універсальних нормах, сформованих на основі середніх показників великих груп населення. Такі традиційні методики, як розрахунок базальної метаболічної потреби за формулою Гарріса-Бенедикта, дозволяли оцінити загальний енергетичний

баланс. Проте універсальний підхід не враховує специфіки обміну речовин у окремих осіб, їх індивідуальних потреб та змін у способі життя, що зробило необхідним перехід до персоналізованих моделей [4, 8].

З появою мобільних технологій і сенсорних пристроїв з'явилася можливість інтегрувати дані, що збираються в режимі реального часу, з хмарними сервісами. Завдяки цьому дані про фізичну активність, температуру тіла, серцевий ритм або якість сну можуть автоматично використовуватися для коригування дієтичного плану. Системи, які здатні адаптувати харчування під поточний стан організму, роблять процес планування більш гнучким і точно налаштованим на індивідуум.

Одним із ключових напрямків сучасних досліджень є вдосконалення класичних математичних моделей. Формула Гарріса-Бенедикта, яка використовується для розрахунку базальної метаболічної потреби, сьогодні доповнюється додатковими коефіцієнтами, що враховують рівень фізичного навантаження, сезонні зміни у активності, вікові особливості та інші фактори. В результаті отримуються моделі, які дозволяють більш точно визначати необхідну калорійність і розподіл макронутрієнтів для кожного користувача [9].

Інтеграція технологій штучного інтелекту відкриває додаткові можливості для оптимізації харчового планування. Алгоритми машинного навчання можуть аналізувати історичні дані про вага, фізичну активність та інші параметри, що дає змогу прогнозувати зміни у харчових потребах та генерувати персоналізовані рекомендації. Наприклад, використання платформ на зразок Google GenAI дозволяє створювати альтернативні варіанти меню, адаптовані до змін у стані користувача та враховувати індивідуальні харчові вподобання [10].

Сучасні мобільні додатки для планування харчування характеризуються інтуїтивно зрозумілими інтерфейсами, що сприяють легкому введенню користувацьких даних та перегляду рекомендацій. Вони можуть інтегруватися з іншими сервісами охорони здоров’я, наприклад, з пристроями для моніторингу активності або додатками для вимірювання параметрів здоров’я, що підвищує

точність формування індивідуальних раціонів. Це забезпечує широкий доступ до інноваційних дієтичних технологій для різних груп населення, включаючи як спортсменів, так і осіб із хронічними захворюваннями або бажаючих вести здоровий спосіб життя.

Перспективи розвитку сучасних підходів зосереджені на розширенні функціоналу систем через впровадження мультимодальних технологій, що дозволяють більш глибоко аналізувати дані про стан здоров'я, а також на інтеграції з медичними базами даних і статистичними моделями. Це сприятиме подальшій персоналізації рекомендацій і надасть можливість не лише реагувати на зміни в режимі реального часу, але й прогнозувати потенційні ризики для здоров'я. Крім того, важливо враховувати соціокультурні аспекти харчування, що дозволяє адаптувати системи під регіональні особливості та традиції.

Таким чином, сучасні підходи до планування харчування поєднують класичні наукові методики з передовими технологіями аналізу даних і штучного інтелекту. Це дозволяє створити динамічну систему, здатну оперативно адаптувати харчові плани до змін у способі життя та фізіології користувача, що є важливим для підтримки оптимального здоров'я і профілактики хронічних захворювань.

1.3 Огляд аналогів

Сучасний ринок додатків для планування харчування характеризується великим різноманіттям рішень, орієнтованих на підвищення свідомості щодо власного раціону, відстеження споживання калорій та аналізу нутрієнтів. З огляду на актуальність питань здорового харчування та зменшення ризику виникнення хронічних захворювань, популярність подібних сервісів значно зросла. Серед аналогів на ринку мобільних додатків перебувають декілька популярних продуктів, а саме:

– MyFitnessPal;
– Nutritionix;
– Lose It!.

MyFitnessPal [11] є одним із найпопулярніших мобільних додатків, що дозволяє користувачам відстежувати споживання їжі, планувати раціон і аналізувати макро- та мікронутрієнти. Однією з основних переваг цього додатку є наявність величезної бази даних продуктів, яка надає можливість точно визначати калорійність і харчову цінність продуктів практично з будь-якого регіону світу. Крім того, MyFitnessPal (рис.1.2) інтегрується з численними фітнес-трекерами та іншими додатками, що дозволяє користувачам автоматично імпортувати дані про фізичну активність та коригувати раціон на основі поточної активності. Також варто відзначити інтуїтивно зрозумілий інтерфейс, який сприяє широкому прийняттю серед користувачів різного віку і рівня підготовки. Проте серед недоліків MyFitnessPal іноді зазначають недостатню локалізацію для неангломовних користувачів, а також залежність якості даних від точності самостійного введення інформації, що може впливати на кінцеві розрахунки.

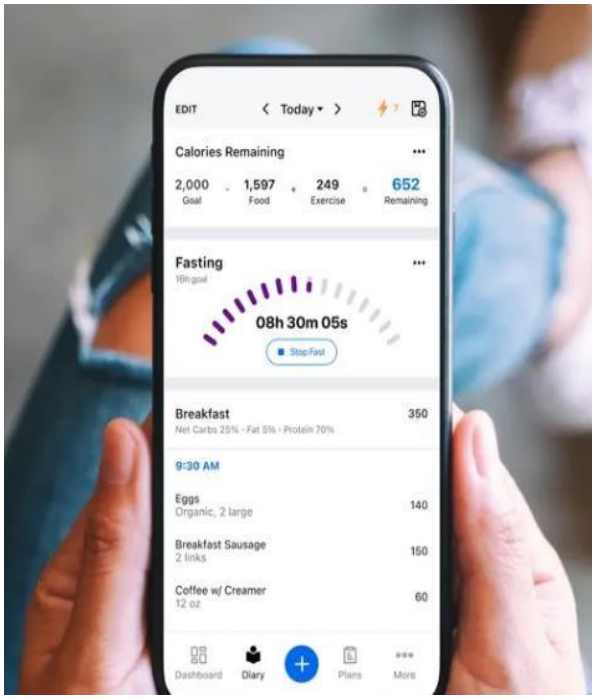


Рисунок 1.2 – Інтерфейс додатку «MyFitnessPal»

Nutritionix [12] позиціонується як потужна платформа для аналізу харчових даних, яка орієнтована не лише на кінцевих користувачів, але й на розробників, завдяки розвинутому API. Основна перевага Nutritionix (рис.1.3) полягає у великій та регулярно оновлюваній базі даних з інформацією про харчову цінність продуктів, що дозволяє досягти високої точності розрахунків. Завдяки відкритому API, це рішення стає привабливим для інтеграції у власні проекти, де потрібна обробка та аналіз харчових даних у реальному часі. Інтерфейс платформи орієнтований на підтримку складних запитів, що дозволяє розробникам гнучко використовувати функціонал Nutritionix у комерційних або наукових дослідженнях. Проте для звичайного споживача додаток може здатися дещо «технічним», а процес налаштування і використання API вимагає певних знань, що обмежує його застосування поза професійною аудиторією.

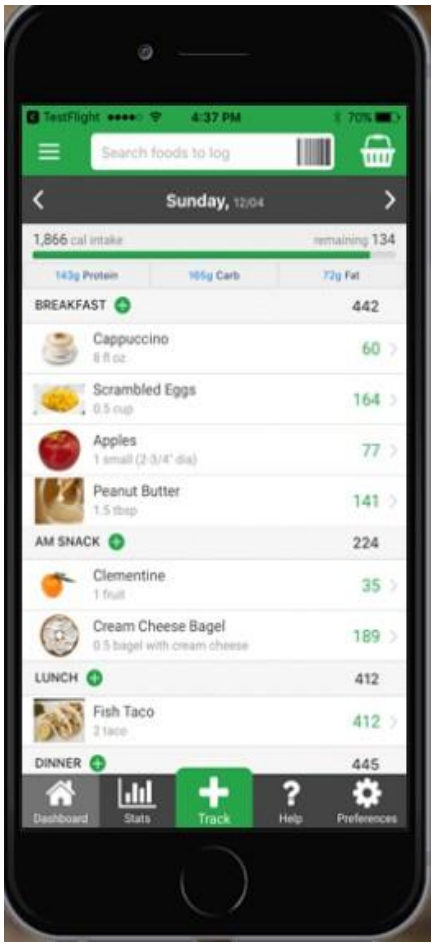


Рисунок 1.3 – Інтерфейс додатку «Nutritionix»

Lose It! [13] – це додаток, розроблений із акцентом на простоту використання та мотивацію користувачів до досягнення цілей зі схуднення. Основною ідеєю Lose It! (рис.1.4) є легкий та зручний інтерфейс для введення даних про споживання їжі, що дозволяє швидко формувати щоденний план харчування та відстежувати калорійність раціону. Додаток пропонує можливість встановлення індивідуальних цілей схуднення, а також містить елементи соціальної взаємодії, завдяки яким користувачі можуть порівнювати свої досягнення з іншими. Така соціальна структура сприяє підвищенню мотивації і регулярності використання сервісу. Крім того, Lose It! регулярно оновлює свою базу даних про харчові продукти, що дозволяє отримувати актуальні дані для аналізу споживання. Проте деякі користувачі відзначають, що база даних і продуктова інформація іноді недостатньо деталізовані, а інтеграція з іншими пристроями може бути обмеженою, що є потенційним недоліком порівняно з більш розвиненими аналогами.

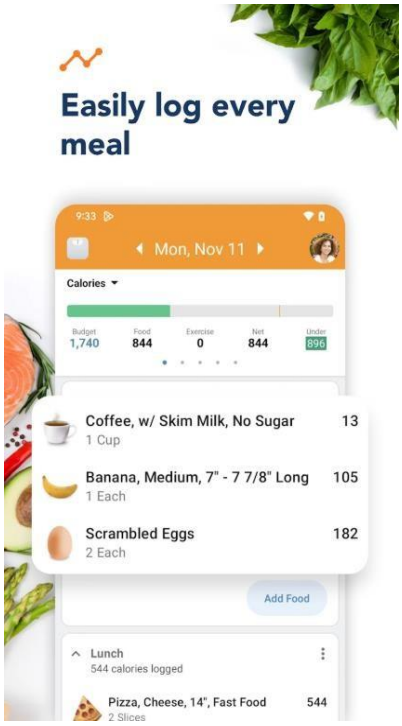


Рисунок 1.4 – Інтерфейс додатку «Lose It!»

Проведений аналіз показує, що кожен із розглянутих аналогів має свої сильні та слабкі сторони, що залежить від цільової аудиторії та специфіки застосування. MyFitnessPal вирізняється великою базою даних та інтеграцією з фітнес-трекерами, що робить його універсальним інструментом для кінцевих користувачів. Nutritionix, завдяки своєму відкритому API і точній продуктій інформації, підходить для інтеграції в корпоративні проекти та наукові дослідження з широкою можливістю кастомізації. Lose It! орієнтується на користувачів, які віддають перевагу простоті використання та соціальній мотивації, що сприяє регулярному контролю над власним раціоном.

При розробці інформаційної системи автоматизованого підбору дієти важливо врахувати переваги кожного з аналізованих рішень. Наприклад, інтеграція елементів гнучкої бази даних продуктів із можливістю автоматичного збирання даних з пристроїв може стати базою для створення високоточного та адаптивного алгоритму розрахунку харчових потреб. Також важливо впровадити інтерфейс, що забезпечує простоту введення даних та користувацьку взаємодію, зберігаючи високий рівень персоналізації, що характерний для Lose It!. З іншого боку, використання відкритого API, подібного Nutritionix, може значно розширити можливості інтеграції системи з іншими цифровими платформами та сервісами охорони здоров’я.

Отже, проведений огляд аналогів дозволяє зробити висновок про необхідність гармонійного поєднання функціоналу, адаптивності та зручності використання при розробці нової інформаційної системи. Кожне з розглянутих рішень, має унікальний набір переваг, які можуть бути використані як референс для подальшої розробки власного продукту. Застосування сучасних технологій і адаптація кращих практик з ринку дозволить створити конкурентоспроможний додаток, що відповідає сучасним вимогам персоналізованого планування харчування [4].

1.4 Постановка задачі дослідження

Розробка інформаційної системи автоматизованого підбору дієти та харчового плану є актуальним завданням, яке потребує комплексного підходу. Основна мета дослідження – створити мобільний додаток для інформаційної системи, що дозволяє користувачам отримувати персоналізовані рекомендації щодо харчування з урахуванням їхніх фізичних параметрів, дієтичних обмежень та поставлених цілей. Для досягнення цієї мети необхідно вирішити низку ключових завдань, що визначають структуру та функціонал системи.

Основні задачі дослідження включають:

1. Аналіз існуючих методів розрахунку харчового плану. Дослідження класичних математичних моделей розрахунку базального обміну речовин, зокрема формули Гарріса-Бенедикта, для встановлення найбільш точного алгоритму визначення добової калорійності користувача;
2. Впровадження обчислювального алгоритму, що враховує фізіологічні параметри людини (вік, вага, зріст, стать), рівень фізичної активності та мету (схуднення, підтримка ваги, набір маси);
3. Інтеграція AI-модуля для адаптації харчового плану. Використання зовнішнього AI-сервісу (Google GenAI) для корекції запропонованого меню в режимі реального часу при необхідності користувача змінити окремі страви;
4. Розробка зручного та інтуїтивно зрозумілого користувацького інтерфейсу. Впровадження сучасного дизайну з використанням React та TypeScript, що забезпечує швидку навігацію та адаптивність для різних мобільних платформ;
5. Забезпечення кросплатформної сумісності застосунку. Використання технології Capacitor для підтримки роботи мобільного додатку як нативного застосунку на Android [14] та iOS [15];

6. Розробка модуля збереження даних користувача. Створення механізму зберігання даних, що дозволяє запам’ятовувати параметри користувача та історію змін у харчовому плані;

7. Проведення тестування для оцінки продуктивності та точності розрахунків. Виконання модульного та інтеграційного тестування для перевірки коректності роботи алгоритмів, стабільності додатку та відповідності отриманих результатів науковим даним.

1.5 Висновки

Аналіз предметної області підтвердив необхідність автоматизації підбору дієти та харчового плану з урахуванням індивідуальних параметрів користувача. Вивчення існуючих рішень показало, що більшість аналогів пропонують статичні раціони без можливості адаптації в режимі реального часу, що знижує їхню ефективність. Запропонована система вирішує цю проблему завдяки інтеграції AI-компонента Google GenAI, який дозволяє гнучко змінювати меню відповідно до уподобань користувача, зберігаючи баланс макронутрієнтів. Застосування математичних моделей, зокрема формули Гарріса-Бенедикта, гарантує точність розрахунку добової калорійності, а використання кросплатформних технологій React та Capacitor забезпечує доступність додатку для Android та iOS.

Попри переваги системи, ключовим обмеженням залишається необхідність початкового введення параметрів користувача, що впливає на точність прогнозів на перших етапах використання. Для подальшого розвитку можливе впровадження алгоритмів машинного навчання, які аналізуватимуть харчові звички користувачів та динамічно коригуватимуть рекомендації на основі накопичених даних. Загалом проведений аналіз підтвердив перспективність розробки, окреслив ключові технологічні рішення та визначив напрями вдосконалення інформаційної системи.

2 ВИБІР ОПТИМАЛЬНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

2.1 Обґрунтування вибору мови програмування

Розробка мобільного додатку потребує ретельного вибору технологій, зокрема мови програмування, яка забезпечуватиме високу продуктивність, масштабованість та простоту підтримки. Сучасні мобільні рішення повинні враховувати не лише базові вимоги до кодування, а й інтеграцію з різноманітними сервісами, можливість використання AI-технологій, а також адаптацію до змінних вимог користувачів. Саме від цього залежить, наскільки швидко продукт зможе реагувати на зворотний зв’язок, впроваджувати нові функції та забезпечувати стабільну роботу навіть при високих навантаженнях. Крім того, вибір мови програмування має суттєвий вплив на якість та доступність кінцевого продукту для широкої аудиторії користувачів.

Аналіз популярних мов програмування для мобільних застосунків дозволяє виділити кілька основних варіантів, кожен з яких має свої особливості в залежності від платформної орієнтації, функціональних можливостей та рівня підтримки спільнотою:

– Java [16] є однією з найстаріших мов програмування, що використовується для розробки застосунків під Android. Її основними перевагами є стабільність, випробуваність часом та широке використання у корпоративних системах. Сильна сторона мови полягає у її надійності та потужній підтримці розробницької спільноти, що гарантує наявність багатьох готових рішень і бібліотек. Проте слід зазначити, що Java характеризується відносно великим обсягом коду, що може затримувати розробку та ускладнювати впровадження змін, особливо коли йдеться про швидке оновлення застосунку;

– Swift [17] – це сучасна мова програмування, спеціально створена компанією Apple для розробки застосунків на платформі iOS. Вона вирізняється зручним, інтуїтивним синтаксисом і високою продуктивністю. Swift оптимізована для пристроїв Apple, що дозволяє досягати чудових результатів у

швидкодії і ефективному використанні ресурсів. Однак її застосування обмежене екосистемою iOS, що створює певні труднощі при розробці кросплатформених рішень, якщо потрібно охопити і користувачів Android;

– Kotlin [18] є сучасною мовою для розробки застосунків під Android і розглядається як суттєве вдосконалення Java. Його компактний та читабельний синтаксис дозволяє значно зменшити обсяг коду, що сприяє простоті підтримки та зниженню кількості помилок. Крім того, Kotlin має потужну підтримку в Android-екосистемі, проте, як і Swift, орієнтований переважно на одну платформу, що обмежує його застосування в умовах, коли потрібно створити єдине кросплатформене рішення;

– TypeScript – це надмножина JavaScript з підтримкою статичної типізації, яка дозволяє попередньо виявляти помилки та підвищувати надійність коду. Використання мови TypeScript особливо актуальне при створенні кросплатформених мобільних застосунків на базі фреймворку React. Такий підхід дозволяє розробляти єдину кодову базу для Android та iOS, що значно скорочує витрати на розробку та забезпечує швидку інтеграцію з іншими сервісами. Крім того, TypeScript сприяє підвищенню якості кінцевого продукту за рахунок суворого контролю типів та більш структурованої організації коду, що є особливо важливим для масштабованих систем.

Враховуючи поставлені завдання та специфіку проекту, було обрано TypeScript у поєднанні з React та Capacitor. Розглянемо основні причини такого вибору:

1. Кросплатформність. Мобільний додаток має працювати на як Android, так і iOS. Завдяки використанню TypeScript з React розробники можуть писати загальний код, який виконуватиметься без змін на обох платформах. Це дозволяє спростити процес розробки, знизити витрати на тестування та подальшу підтримку, а також уникнути дублювання функціональності в різних кодових базах;

2. Простота підтримки та розширення функціоналу. Строгий контроль типів, який забезпечує TypeScript, дозволяє значно зменшити кількість програмних помилок на етапі компіляції, що позитивно впливає на стабільність застосунку. Така передбачуваність допомагає легше модифікувати код при зміні вимог і додавати нові функції без ризику порушити існуючу логіку роботи системи;

3. Швидкість розробки та модульність. Фреймворк React дозволяє створювати незалежні компоненти інтерфейсу, які можна повторно використовувати у різних частинах застосунку. Це сприяє ефективному розподілу розробницьких завдань у команді, дозволяє швидко впроваджувати інновації та легко адаптувати інтерфейс до змінних вимог ринку;

4. Інтеграція з AI-сервісами. Використання Google GenAI для адаптації дієтичного плану під вподобання користувача вимагає надійної взаємодії з зовнішніми API. Завдяки зручній підтримці робочих процесів в TypeScript забезпечується швидка інтеграція з AI-сервісами, що сприяє покращенню користувацького досвіду та оптимізації алгоритмів роботи додатку;

5. Взаємодія з апаратними функціями пристроїв. Інтеграція з Capacitor дозволяє отримувати доступ до різноманітних апаратних функцій, таких як камера, GPS, акселерометр та інші сенсори. Це забезпечує можливість розширити функціональність застосунку без необхідності розробляти окремо нативні модулі для різних платформ, що суттєво економить час та ресурси;

6. Гнучкість та адаптивність. Використання React дає змогу ефективно адаптувати застосунок під різні розміри екранів і технологічні можливості пристроїв, що робить його зручним у використанні для ширшої аудиторії. Такий підхід дозволяє досягти високої якості інтерфейсу незалежно від особливостей апаратного забезпечення;

7. Велика спільнота розробників. Широкий спектр підтримки та наявність численних бібліотек для TypeScript і React забезпечує оперативне вирішення потенційних проблем, постійне оновлення інструментів та доступ до

сучасних технологічних рішень. Це є важливим фактором для забезпечення довгострокової підтримки та розвитку застосунку.

2.2 Аналіз середовища розробки Visual Studio Code та Android Studio

Розробку системи автоматизованого підбору дієти та харчового плану було здійснено за допомогою інтегрованих середовищ, які забезпечують ефективну організацію кодування, налагодження та тестування. Вибір середовища розробки відіграє важливу роль у забезпеченні продуктивності розробки, гнучкості процесу та оптимізації програмного забезпечення під вимоги користувачів.

Visual Studio Code [19] – гнучке середовище для розробки фронтенду. Основним редактором коду, який використовувався протягом роботи над додатком, є Visual Studio Code (VS Code). Це сучасне середовище розробки, яке відзначається високою продуктивністю, гнучкістю, численними розширеннями та зручною інтеграцією з системами контролю версій. Його легкість у використанні та широкий спектр можливостей робить його оптимальним вибором для розробки веб-інтерфейсів та мобільних застосунків.

Значні переваги VS Code:

– підтримка TypeScript і JavaScript — дозволяє розробникам використовувати статичну типізацію для зменшення ймовірності помилок та покращення читабельності коду;

– автоматичне форматування коду — інтеграція розширень, таких як Prettier, допомагає підтримувати єдиний стиль програмування в проєкті;

– інтелектуальне доповнення коду — завдяки вбудованій функції IntelliSense VS Code пропонує варіанти коду, що значно покращує швидкість написання програмного забезпечення;

– інтеграція з Git — дозволяє зручне управління системою контролю версій без необхідності використовувати окремі інструменти;

– підтримка фреймворків — середовище чудово адаптоване для роботи з React, що було ключовим аспектом під час реалізації інтерфейсу мобільного застосунку.

Крім того, VS Code широко використовується для роботи з системою контролю версій Git, що дозволяло постійно відстежувати зміни в коді, здійснювати ревізії та працювати над оновленнями в єдиній кодовій базі. Такий підхід забезпечував плавну інтеграцію нових функціональних можливостей та створення адаптивних інтерфейсів, що є особливо важливим для проектів, орієнтованих на персоналізований підбір харчового плану.

Ще одним важливим інструментом, який активно використовувався під час розробки, був Android Studio [20]. Android Studio – нативне середовище для розробки мобільних додатків. Це потужне середовище розробки, яке надає можливість не лише створювати застосунки для Android, але й ефективно тестувати їх за допомогою вбудованого емулятора.

Ключові можливості Android Studio:

– вбудований емулятор — дозволяє тестувати мобільні застосунки без фізичних пристроїв, моделюючи роботу на смартфонах та планшетах з різними характеристиками;

– профайлери продуктивності — дозволяють аналізувати використання оперативної пам’яті, швидкість виконання застосунку та завантаженість процесора;

– Live Edit — функція, яка дозволяє розробникам вносити зміни до UI в реальному часі без необхідності перевантаження застосунку;

– моніторинг споживання ресурсів — дозволяє оцінити ефективність додатка та оптимізувати його продуктивність.

Емулятор Android дозволяв перевіряти роботу додатку на різних моделях пристроїв, конфігураціях роздільної здатності і версіях операційної системи, що допомагало виявляти та виправляти помилки на ранніх етапах розробки. За допомогою інструментів налагодження, включаючи профайлери продуктивності

і монітори ресурсів, розробники могли оптимізувати роботу застосунку для забезпечення високої стабільності та швидкодії.

На практиці поєднання роботи в VS Code та Android Studio дозволило досягти високої ефективності в процесі тестування і відлагодження. Розробники отримали можливість оперативно запускати емулятор, проводити інтерактивне тестування та переглядати результати роботи застосунку в режимі реального часу. Це дозволяло коригувати будь-які недоліки та оптимізувати користувацький досвід, враховуючи специфіку як веб-інтерфейсу, так і нативних функцій пристроїв.

Інтеграція VS Code та Android Studio у розробці мобільного застосунку. Використання цих двох середовищ розробки дозволило:

– ефективно писати код у Visual Studio Code, використовуючи всі переваги TypeScript та React;

– запускати мобільний застосунок у Android Studio та тестувати його на віртуальних пристроях;

– впроваджувати зміни швидко та відстежувати їх у Git безпосередньо з середовища розробки;

– оптимізувати продуктивність мобільного застосунку за допомогою Android Studio.

Використання розширень та інтегрованих інструментів у VS Code сприяло підтриманню чистоти та зрозумілості коду, а застосування Android Studio — забезпеченню відповідності роботи додатку вимогам мобільного ринку. Така організація робочого процесу дозволила ефективно управляти розробкою, швидко реагувати на зворотний зв’язок та інтегрувати нові можливості в проект.

У підсумку, поєднання цих інструментів сприяло створенню високоякісного застосунку, який демонструє стабільну роботу в умовах реальної експлуатації. Саме тому використання Visual Studio Code для розробки та Android Studio для тестування і налагодження становить важливу складову успішної

реалізації проекту, забезпечуючи оптимізацію циклів розробки, підвищення продуктивності команди та гарантування високої якості кінцевого продукту.

2.3 Інтеграція та використання Google GenAI

У рамках дослідження було розроблено систему автоматизованого підбору дієти та харчового плану, яка поєднує науково обґрунтовані алгоритми розрахунку з можливостями штучного інтелекту. Основний план формується на основі індивідуальних параметрів користувача, таких як вік, вага, зріст, стать, рівень фізичної активності та харчові вподобання. Це дозволяє системі визначати добову калорійність раціону та оптимальний розподіл макронутрієнтів (білків, жирів, вуглеводів) відповідно до потреб кожного користувача.

Роль штучного інтелекту у персоналізації харчування. Оскільки раціональні алгоритми можуть лише запропонувати загальну структуру харчового плану, а індивідуальні смакові уподобання користувачів можуть значно варіюватися, було реалізовано механізм динамічної адаптації меню за допомогою Google GenAI. Цей штучний інтелект здатний генерувати альтернативні варіанти страв у реальному часі, зберігаючи при цьому загальну харчову балансованість раціону.

Користувач має можливість переглядати запропонований йому харчовий план та оцінювати відповідність рекомендованих страв своїм уподобанням. Якщо певна страва не відповідає його смаковим очікуванням або дієтичним обмеженням, він може скористатися функцією заміни страви за допомогою AI. У додатку передбачено інтерактивний механізм, що дозволяє швидко здійснювати модифікації без втрати точності розрахунків.

Технологічна реалізація інтеграції Google GenAI. Google GenAI інтегрується в систему через API, який обробляє запити користувача та генерує альтернативні варіанти страв на основі заданих критеріїв. Основні етапи взаємодії між мобільним додатком та Google GenAI:

– формування початкового плану – система на основі вхідних параметрів користувача створює базову модель харчування;

– запит на зміну страви – якщо користувач бажає змінити певну страву, система передає запит до AI із характеристиками бажаної альтернативи;

– аналіз AI та генерація нового варіанту – алгоритм AI обробляє запит і формує нову страву, яка відповідає калорійним та нутрієнтним показникам оригінальної;

– оновлення харчового плану – система інтегрує отриману інформацію та оновлює меню відповідно до змін, збережених користувачем.

Такий підхід гарантує гнучкість системи: основна структура харчового плану залишається стабільною, а кожен користувач має можливість адаптувати меню відповідно до своїх вподобань без порушення загального балансу поживних речовин.

Переваги використання AI для адаптації харчування. Застосування штучного інтелекту для модифікації меню має кілька важливих переваг:

– індивідуальний підхід – система підбирає альтернативи, враховуючи конкретні дієтичні вподобання та обмеження користувача;

– збереження харчового балансу – AI аналізує нутрієнтний склад кожної страви та забезпечує відповідність рекомендаціям щодо здорового харчування;

– автоматична корекція – якщо користувач регулярно змінює певні продукти, система може адаптувати майбутні рекомендації на основі його історії вибору;

– зручність та швидкість – вся процедура зміни страви відбувається за лічені секунди, що значно економить час користувача.

Інтеграція з Google GenAI у розробленому мобільному додатку стала ключовим фактором для покращення користувацького досвіду, надаючи можливість динамічної модифікації раціону з урахуванням персональних вподобань. Завдяки використанню AI технологій система не лише спрощує

процес планування харчування, але й робить його більш персоналізованим, адаптивним та ефективним.

Таким чином, поєднання алгоритмічного розрахунку добової калорійності та AI-персоналізації забезпечує високу точність, індивідуальний підхід та максимальний комфорт для користувача.

2.4 Метод розрахунку дієтичного плану

Формула Гарріса-Бенедикта [22] – один із найбільш поширених методів розрахунку базального обміну речовин (БОР – базальний обміну речовин), тобто кількості калорій, необхідних організму на підтримку важливих функцій в стані спокою. Розроблена фахівцями Дж. Артуром Гаррісом та Френсісом Г. Бенедиктом у 1919 році, ця формула використовується в дієтології для визначення індивідуальних калорійних потреб. У мобільному додатку вона буде основою для генерування персоналізованого плану харчування.

Формула Гарріса-Бенедикта відрізняється для чоловіків і жінок:

Для чоловіків: БОР = 66.5 + (13.75 × (вага в кг)) + (5.003 × (зріст в см)) – (6.775 × (вік в роках)).

Для жінок: БОР = 655.1 + (9.563 × (вага в кг)) + (1.85 × (зріст в см)) - (4.676 × (вік в роках)).

Ці рівняння дають початкову оцінку кількості калорій, які організм використовує для базових функцій.

Коефіцієнти активності: після розрахунку БОР отримане значення множиться на коефіцієнт фізичної активності, що дозволяє врахувати додаткове енергоспоживання при виконанні фізичних вправ:

- Низька активність (малорухливий спосіб життя): 1.2;
- Легка активність (легкі тренування 1-3 рази на тиждень): 1.375;
- Середня активність (помірні тренування 3-5 разів на тиждень): 1.55;

– Висока активність (інтенсивні тренування 6-7 разів на тиждень): 1.725;

– Дуже висока активність (важка фізична праця, інтенсивні тренування кілька разів на день): 1.9.

Коригування для цілей: після визначення загальної потреби в калоріях значення коригується відповідно до харчової мети користувача [22]:

– Схуднення: зменшення на 15–20% від загальної потреби;

– Підтримка ваги: без змін;

– Набір маси: збільшення на 15–20% від загальної потреби.

Розрахунок макронутрієнтів: на основі отриманої цільової кількості калорій мобільний додаток автоматично розподіляє калорії між основними групами нутрієнтів:

– Білки: 30% від загальних калорій (1 (г) білка = 4 (ккал));

– Жири: 30% від загальних калорій (1 (г) жиру = 9 (ккал));

– Вуглеводи: 40% від загальних калорій (1 (г) вуглеводів = 4 (ккал)).

Такий розподіл дозволяє забезпечити необхідний рівень енергії та оптимальне співвідношення макроелементів, що сприяє досягненню поставлених харчових цілей.

Приклади обрахунків:

Приклад 1: Чоловік, 30 років, 80 (кг), 180 (см), середня активність, мета – схуднення.

1. Розрахунок БОР:

$$БОР=66.5+(13.75\times80)+(5.003\times180)-(6.775\times30);$$

$$БОР=66.5+1100+900.54-203.25=1863.79 \text{ (ккал)}.$$

2. Врахування рівня активності:

$$\text{Загальна потреба}=1863.79\times1.55=2888.87 \text{ (ккал)}.$$

3. Коригування для мети схуднення (-20%):

$$\text{Цільова кількість калорій}=2888.87\times0.8=2311.1 \text{ (ккал)}.$$

4. Розрахунок макронутрієнтів:
   - Білки: $\frac{2311.1\times0.3}{4} \approx 173.33(\text{г})$;
   - Жири: $\frac{2311.1\times0.3}{9} \approx 77.04(\text{г})$;
   - Вуглеводи: $\frac{2311.1\times0.4}{4} \approx 231.11(\text{г})$.

Приклад 2: Жінка, 25 років, 60 (кг), 165 (см), низька активність, мета – підтримка ваги.

1. Розрахунок БОР:

$$\text{БОР}=655.1+(9.563\times60)+(1.85\times165)-(4.676\times25);$$

$$\text{БОР}=655.1+573.78+305.25-116.9=1417.23 \text{ (ккал)}.$$

2. Врахування рівняння активності:

$$\text{Загальна потреба}=1417.23\times1.2=1700.68 \text{ (ккал)}.$$

3. Коригування для підтримки ваги (без змін):

$$\text{Цільова кількість калорій}=1700.68 \text{ (ккал)}.$$

4. Розрахунок макронутрієнтів:
   - Білки: $\frac{1700.68\times0.3}{4} \approx 127.55(\text{г})$;
   - Жири: $\frac{1700.68\times0.3}{9} \approx 56.69(\text{г})$;
   - Вуглеводи: $\frac{1700.68\times0.4}{4} \approx 170.07(\text{г})$.

2.5 Вибір IT-інфраструктури

Розробка мобільного додатку для інформаційної системи автоматизованого підбору дієти та харчового плану потребує ретельного аналізу IT-інфраструктури, яка забезпечить ефективне функціонування додатку. Вибір технологій має відповідати критеріям продуктивності, масштабованості та доступності для користувачів.

IT-інфраструктура застосунку включає три основні компоненти:

1. Апаратне забезпечення – вимоги до пристроїв, на яких працюватиме додаток;

2. Програмне забезпечення – середовище розробки, мови програмування, фреймворки та інструменти;

3. Архітектура та модульність – взаємодія компонентів для забезпечення стабільної роботи та гнучкості системи.

Розроблена система складається з клієнтської та серверної частин, які взаємодіють для забезпечення автоматизованого підбору дієти та харчового плану. Архітектура побудована на принципах модульності, що забезпечує гнучкість та розширюваність системи.

На рисунку 2.1 зображено архітектуру інформаційної системи.

Рисунок 2.1 – Архітектура інформаційної системи

Система включає наступні ключові компоненти:

1. Клієнтська частина (Frontend):

   – Мобільний додаток, реалізований за допомогою React Native та TypeScript, що забезпечує кросплатформну сумісність (Android, iOS);

– Голосове керування для взаємодії з додатком;

– Інтеграція AI-компонента Google GenAI для динамічного коригування меню.

2. Серверна частина (Backend):

– Хмарний сервер на базі Google Cloud, що обробляє запити користувача;

– База даних для збереження профілів користувачів, історії змін та налаштувань;

– API для комунікації між мобільним додатком та сервером.

3. Хмарні сервіси та інфраструктура:

– Google GenAI — AI-компонент, що генерує альтернативні варіанти страв у реальному часі;

– Cloud Firestore — база даних у реальному часі для збереження інформації про користувача та його дієтичний план.

4. Безпека системи:

– SSL-шифрування для безпечного передавання даних між клієнтом і сервером;

– Обмеження доступу до даних через рольову модель прав доступу;

– Регулярне резервне копіювання бази даних для захисту від втрати інформації.

5. ПЗ та інструменти моніторингу:

– Visual Studio Code — середовище розробки коду;

– Android Studio — тестування мобільного додатку на Android;

– Cloud Logging — інструмент для ведення журналу роботи сервера.

Апаратне забезпечення даної інформаційної системи включає мобільні пристрої користувачів та середовище розробки, що є критично важливими для стабільної та ефективної роботи застосунку. Мобільний додаток розроблено для роботи на смартфонах та планшетах під керуванням операційних систем Android

і iOS. Завдяки цьому користувачі отримують можливість комфортного доступу до системи незалежно від того, яку платформу вони використовують.

Основним принципом розробки є використання локального збереження даних, що дозволяє зберігати всі параметри користувача, історію змін та налаштування без необхідності постійного з'єднання із сервером. Використання внутрішніх механізмів, таких як LocalStorage, гарантує швидкий доступ до інформації, забезпечуючи при цьому високий рівень конфіденційності. Такий підхід не лише мінімізує час відповіді програми, але й знижує залежність від мережевих з'єднань, що може бути критичним у умовах поганого зв'язку.

Для забезпечення стабільної роботи мобільного додатку рекомендується використовувати пристрої, які відповідають певним апаратним вимогам. Зокрема, оптимальним варіантом є пристрої з мінімум 3 ГБ оперативної пам'яті та чотириядерним процесором. Ці технічні характеристики забезпечують плавне виконання алгоритмів розрахунку добової калорійності та оперативного коригування дієтичного раціону, що є важливим для безперервної та ефективної роботи системи. Додатково, стабільне інтернет-з'єднання є необхідністю для взаємодії із зовнішніми сервісами, зокрема з AI-сервісами, як-от Google GenAI, що дозволяє в режимі реального часу надавати користувачу персоналізовані рекомендації.

Розробка додатку здійснювалася із використанням сучасних інструментів, що забезпечують кросплатформену сумісність і дозволяють ефективно управляти кодом. Основним середовищем розробки став Visual Studio Code, де застосовано мову програмування TypeScript у поєднанні з фреймворком React. Такий комбінаційний підхід дозволив створити єдину кодову базу для Android та iOS, що значно спрощує підтримку та розширення функціоналу системи. Вбудована підтримка систем контролю версій і численні розширення редактора сприяють швидкому впровадженню нових можливостей, а також оперативному виправленню виявлених помилок.

Тестування додатку виконувалося із використанням емуляторів Android Studio. Поєднання реальних умов експлуатації із симульованими середовищами дозволило провести комплексну перевірку продуктивності, стабільності роботи застосунку та правильності функціонування користувацького інтерфейсу. Особливу увагу приділено перевірці алгоритмів, що обраховують добову калорійність і здійснюють корекцію раціону, щоб забезпечити максимальну точність розрахунків та відповідність даних вимогам користувачів.

Отже, апаратне забезпечення інформаційної системи включає добре злагоджену інфраструктуру мобільних пристроїв і сучасне середовище розробки, що разом сприяють створенню високоефективного, надійного і зручного у використанні додатку. Обраний підхід до збереження даних та тестування дозволяє гарантувати як високий рівень конфіденційності, так і оперативність при обробці запитів, що надзвичайно важливо для систем, орієнтованих на персоналізований підхід у сфері харчування.

2.6 Висновки

Аналіз альтернативних рішень для розробки інформаційної системи показав, що традиційні методи планування харчування здебільшого базуються на статичних дієтах, що не враховують зміни у способі життя користувачів. Порівняння доступних технологій підтвердило, що використання TypeScript у поєднанні з React та Capacitor є оптимальним для створення кросплатформного мобільного додатку, оскільки забезпечує сумісність з Android та iOS, високу продуктивність та легкість розширення функціоналу. Середовище розробки Visual Studio Code було обране через його інтеграцію з системами контролю версій та підтримку модульного програмування.

Запропонована система використовує формулу Гарріса-Бенедикта для розрахунку добової потреби в калоріях, що дозволяє точно визначати раціон відповідно до фізичних параметрів користувача. Вибір IT-інфраструктури

ґрунтувався на критеріях швидкодії, масштабованості та зручності інтеграції. Cloud Firestore було обрано для зберігання користувацьких даних, а Google GenAI—для адаптивного коригування харчового плану. Основним обмеженням є залежність AI-модуля від стабільного інтернет-з’єднання, що може впливати на доступність корекції раціону в режимі реального часу. Подальше вдосконалення може включати розширення можливостей локального збереження даних та впровадження алгоритмів машинного навчання для більш точного прогнозування харчових потреб.

3 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ АВТОМАТИЗОВАНОГО ПІДБОРУ ДІЄТИ ТА ХАРЧОВОГО ПЛАНУ

3.1 Розроблення структури інформаційної системи

Додаток для автоматизованого підбору дієти та харчового плану розроблений для вирішення проблеми правильного та збалансованого харчування користувачів. Основна мета додатку – допомогти людям досягти їхніх цілей щодо здоров'я та фізичної форми шляхом створення персоналізованих планів харчування на основі індивідуальних характеристик та потреб.

Додаток призначений для:

– людей, які прагнуть контролювати свою вагу (схуднення або набір маси);

– спортсменів, які потребують спеціального харчування для підтримки фізичної активності;

– людей з особливими дієтичними потребами та обмеженнями;

– тих, хто хоче покращити своє здоров'я через збалансоване харчування;

– людей, які не мають часу або знань для самостійного планування раціону.

На рисунку 3.1 представлена діаграма випадків використання додатку.

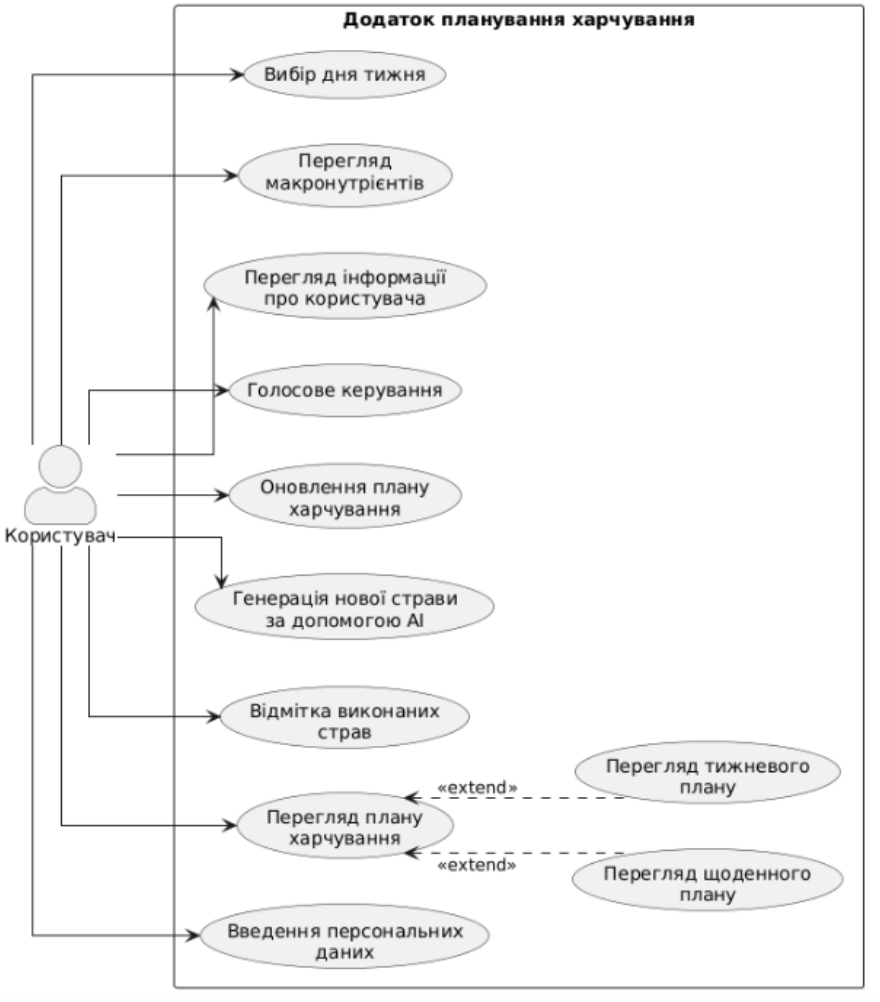


Рисунок 3.1 – Діаграма випадків використання додатку (Use Case Diagram)

Випадки використання додатку:

– введення персональних даних: користувач вводить свої фізичні параметри (вік, стать, вага, зріст), рівень фізичної активності та дієтичні вподобання. Ця інформація використовується для розрахунку оптимальної кількості калорій та макронутрієнтів, необхідних для досягнення поставлених цілей;

– перегляд плану харчування: після введення даних користувач отримує доступ до персоналізованого плану харчування. Він може переглядати щоденний план, який включає сніданок, обід, вечерю та перекуски, а також тижневий план для планування раціону наперед. Кожна страва супроводжується інформацією про калорійність та вміст макронутрієнтів;

– відмітка виконаних страв: користувач може відмічати страви, які він вже спожив протягом дня. Це дозволяє відстежувати прогрес та контролювати дотримання плану харчування;

– генерація нової страви за допомогою AI: якщо запропонована страва не подобається користувачу, він може скористатися функцією генерації нової страви за допомогою штучного інтелекту. Система враховує дієтичні вподобання та необхідні макронутрієнти при генерації нових варіантів;

– оновлення плану харчування: користувач може оновити весь план харчування, щоб отримати нові варіанти страв, які відповідають його потребам та вподобанням;

– голосове керування: для зручності використання додаток підтримує голосове керування, що дозволяє відмічати виконані страви за допомогою голосових команд;

– перегляд інформації про користувача: користувач може переглядати свої персональні дані та розраховані на їх основі потреби в калоріях та макронутрієнтах;

– перегляд макронутрієнтів: додаток надає детальну інформацію про розподіл макронутрієнтів (білків, жирів, вуглеводів) у раціоні, що допомагає користувачу дотримуватися збалансованого харчування;

– вибір дня тижня: користувач може вибирати різні дні тижня для перегляду відповідного плану харчування, що дозволяє планувати раціон наперед.

Додаток розроблено з використанням сучасних технологій, що забезпечує зручний та інтуїтивно зрозумілий інтерфейс. Всі дані користувача зберігаються локально на пристрої, що гарантує конфіденційність. Функція генерації нових страв використовує AI-технології для створення персоналізованих рекомендацій з урахуванням індивідуальних потреб та вподобань користувача.

На рисунку 3.2 зображена діаграма послідовності автоматичного підбору дієти, яка ілюструє порядок взаємодії між користувачем і мобільним додатком під час автоматичного вибору оптимального раціону.

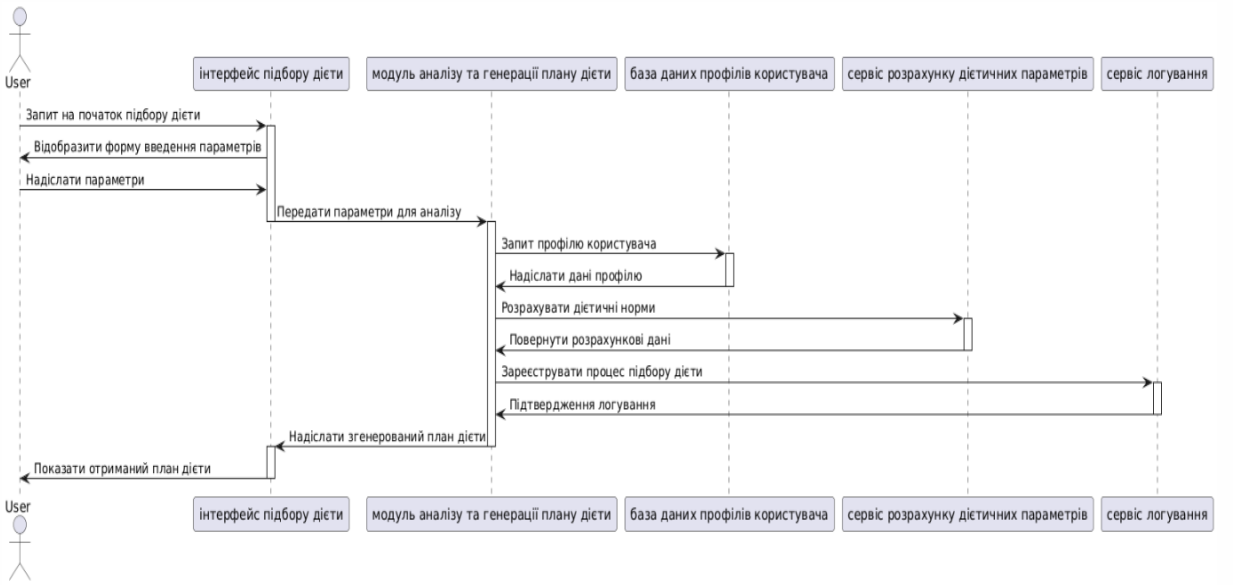


Рисунок 3.2 – Діаграма послідовності автоматичного підбору дієтичного плану

Опис процесу послідовності автоматичного підбору дієтичного плану наступний:

– запит користувача: користувач ініціює процес підбору дієти через інтерфейс застосунку;

– відображення форми введення: система показує користувачеві форму для заповнення параметрів (вік, вага, зріст, рівень активності);

– заповнення та надсилання параметрів: користувач вводить дані та відправляє їх для подальшого аналізу;

– передача даних до модуля аналізу: інтерфейс передає отриману інформацію до модуля аналізу та генерації дієтичного плану;

– запит даних профілю користувача: модуль аналізу звертається до бази даних профілів користувачів, щоб врахувати додаткову інформацію, збережену раніше;

– отримання профільних даних: база даних надсилає релевантні дані користувача у відповідь на запит;

– розрахунок дієтичних параметрів: модуль аналізу виконує математичні розрахунки добової потреби в калоріях, розподілу макронутрієнтів і персоналізованого меню;

– запит до сервісу розрахунку: додаткові обчислення проводяться за допомогою спеціального сервісу розрахунку дієтичних параметрів;

– передача розрахованих даних: сервіс повертає результати розрахунку, які використовуються для формування дієтичного плану;

– логування процесу підбору дієти: модуль аналізу фіксує подію підбору дієти в системі логування, щоб зберегти історію запитів користувачів;

– підтвердження запису в логах: сервіс логування надсилає підтвердження про успішний запис;

– надсилання згенерованого плану дієти: модуль аналізу передає результати до інтерфейсу користувача;

– відображення отриманого плану: користувач отримує згенерований план дієти та може переглянути його в інтерфейсі застосунку.

Ця діаграма послідовності демонструє чітку логіку автоматичного підбору дієти, забезпечуючи персоналізовані рекомендації на основі збережених параметрів та динамічного розрахунку харчових норм.

Діаграма класів, представлена на рисунку 3.3, відображає структуру додатку для автоматизованого підбору дієти та харчового плану.

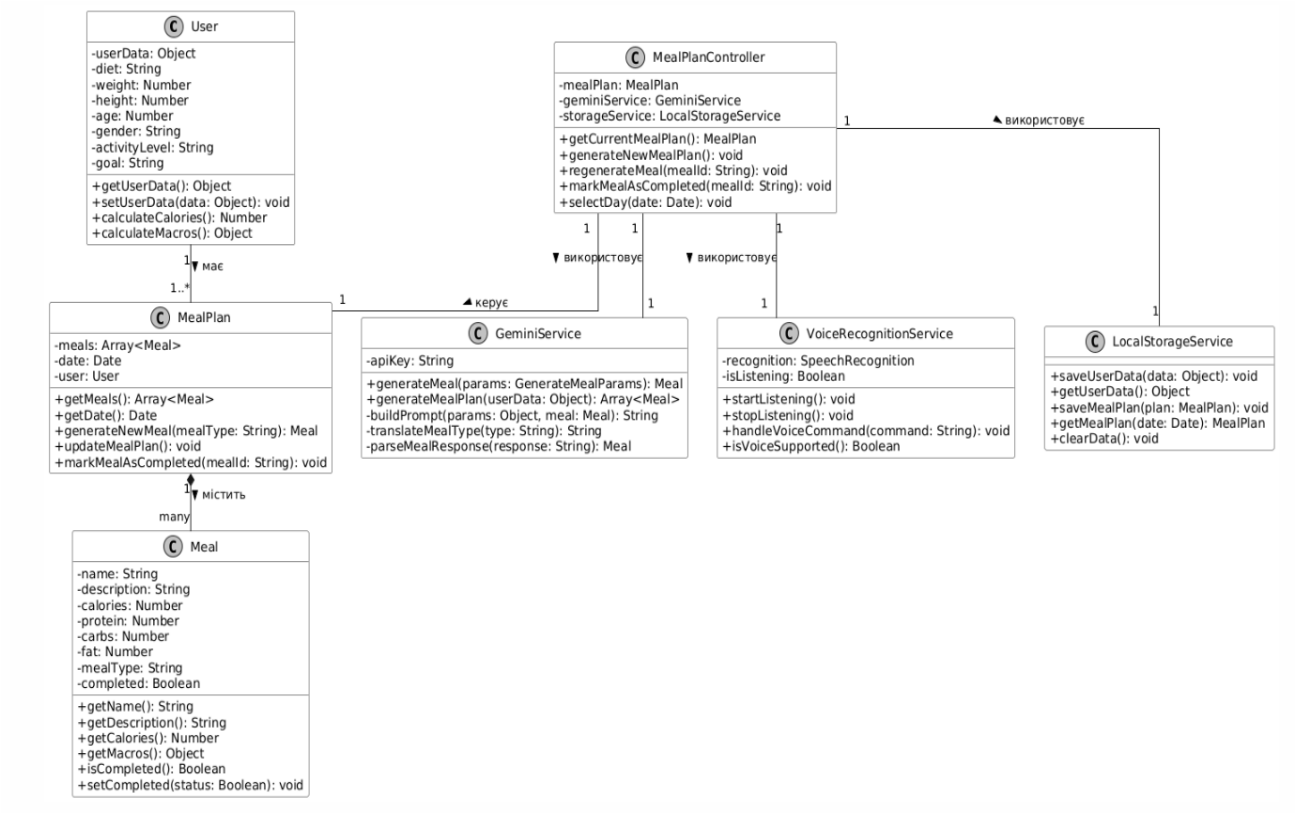


Рисунок 3.3 – Діаграма класів додатку

Основні класи та їх взаємозв'язки:

– User – клас, що представляє користувача системи з його персональними даними (вік, вага, зріст, стать, рівень активності, цілі та дієтичні вподобання). Містить методи для розрахунку калорій та макронутрієнтів;

– Meal – клас, що представляє окрему страву з інформацією про назву, опис, калорійність, макронутрієнти та статус виконання;

– MealPlan – клас, що представляє план харчування на певний день. Містить колекцію страв та методи для оновлення плану;

– GeminiService – сервіс для взаємодії з API Gemini для генерації нових страв та планів харчування;

– VoiceRecognitionService – сервіс для розпізнавання голосових команд користувача;

– LocalStorageService – сервіс для збереження та отримання даних з локального сховища пристрою;

– MealPlanController – контролер, що координує взаємодію між моделями даних та сервісами.

3.2 Розроблення загального алгоритму роботи додатку

Розробка мобільного додатку вимагає ретельного планування та визначення ключових етапів його роботи. Важливо забезпечити інтуїтивність взаємодії користувача із системою та оптимізувати процеси обробки інформації. На рисунку 3.4 представлена блок-схема, що візуалізує етапи функціонування мобільного додатку.

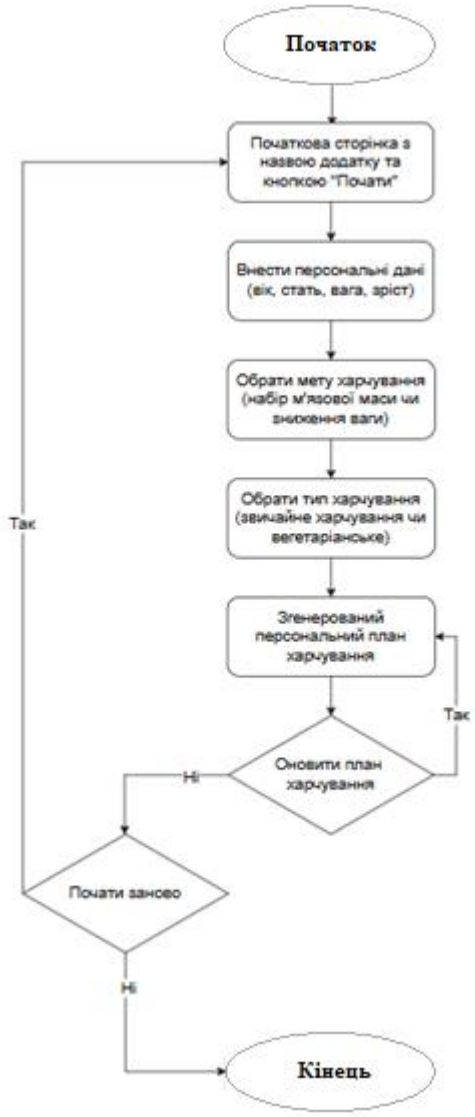


Рисунок 3.4 – Блок-схема алгоритму роботи додатку

Опис загального алгоритму роботи додатку:

Додаток працює за принципом автоматизованого підбору харчового плану, який базується на введених користувачем даних. Основні етапи його функціонування включають:

1. Запуск додатку:
   - Користувач відкриває додаток і потрапляє на початковий екран, де представлена назва застосунку та кнопка «Почати».
2. Введення персональних даних:
   - Користувач вводить основні параметри: вік, стать, вага та зріст;
   - Інформація проходить перевірку на коректність та передається до наступного етапу.
3. Вибір мети харчування:
   - Користувач визначає, чи його мета – схуднення, підтримка ваги чи набір м'язової маси;
   - На основі цього параметра буде здійснюватися коригування калорійності.
4. Вибір типу харчування:
   - Додаток дозволяє користувачеві вибрати між звичайним харчуванням та вегетаріанським раціоном, що впливає на набір доступних продуктів у системі.
5. Генерація персонального плану харчування:
   - Виконується автоматичний розрахунок базального обміну речовин за формулою Гарріса-Бенедикта;
   - Враховуються коефіцієнти фізичної активності;
   - Визначається добова потреба в калоріях та розподіл макронутрієнтів (білки, жири, вуглеводи);
   - Генерується початковий варіант харчового плану.

6. Відображення плану харчування:
   – Користувач отримує згенерований список прийомів їжі з детальною інформацією про калорійність кожної страви.
7. Адаптація меню за допомогою AI:
   – Якщо запропоноване меню не відповідає вподобанням користувача, він може скористатися функцією заміни страви;
   – Генерується запит до Google GenAI, який формує новий варіант відповідно до індивідуальних параметрів.
8. Оновлення або початок нового плану:
   – Користувач може зберегти поточний план або згенерувати новий, змінивши основні параметри;
   – Усі дані зберігаються в пам'яті для подальшого використання.

Ця схема демонструє ключові етапи процесу та взаємодію між користувачем і системою, що забезпечує плавну та ефективну роботу додатку.

Мобільний додаток поєднує перевірені математичні методи та сучасні технології для створення персоналізованого та адаптивного харчового плану, що враховує індивідуальні потреби кожного користувача.

3.3 Розроблення загальної структурної схеми функціонування мобільного додатку автоматизованого підбору дієти та харчового план

Розробка мобільного додатку вимагає чітко продуманої архітектури, яка забезпечує стабільність, гнучкість і ефективність його роботи. Одним із ключових аспектів проектування є визначення структурної схеми функціонування додатку, що дозволяє візуалізувати основні процеси та їхню взаємодію.

Структурна схема додатку демонструє логіку роботи системи: від першого запуску до створення персоналізованого харчового плану та його адаптації за допомогою AI. Вона включає всі основні кроки користувача та автоматизовані дії

системи, що забезпечують точний і зручний підбір раціону відповідно до індивідуальних параметрів. На рисунку 3.5 представлена загальна структурна схема функціонування мобільного додатку.

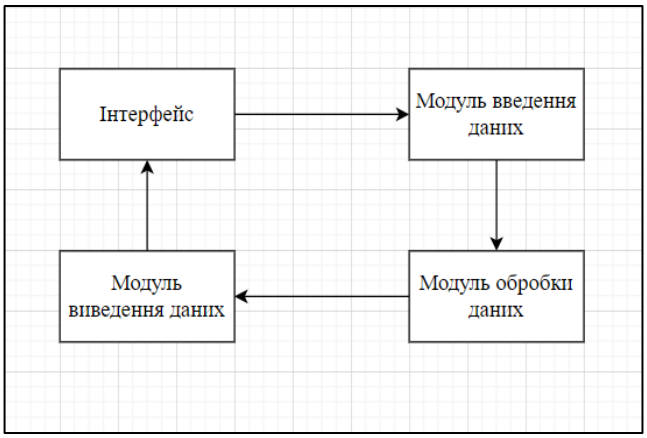


Рисунок 3.5 – Структурна схема функціонування мобільного додатку

Опис структурної схеми функціонування мобільного додатку:

1. Запуск додатку:
   - Відображення головного екрану з кнопкою «Почати»;
   - Перехід до форми введення персональних параметрів.
2. Введення даних користувача:
   - Користувач вводить інформацію про стать, вік, вагу, зріст, рівень фізичної активності та харчові уподобання;
   - Всі введені параметри проходять перевірку на коректність.
3. Розрахунок харчового плану:
   - Використання формули Гарріса-Бенедикта для обчислення базальний обміну речовин (БОР) [18];
   - Коригування добової потреби в калоріях відповідно до рівня активності;
   - Розрахунок оптимального співвідношення макронутрієнтів (білків, жирів, вуглеводів);
   - Генерація індивідуального харчового плану.

4. Відображення плану харчування:
   - Користувач отримує рекомендації щодо раціону на день або тиждень;
   - Усі страви супроводжуються інформацією про калорійність та макронутрієнти.
5. Взаємодія з користувачем:
   - Відмітка виконаних прийомів їжі;
   - Перегляд плану на інші дні тижня;
   - Використання голосового керування для взаємодії з додатком.
6. Оновлення даних та завершення роботи:
   - Користувач може згенерувати новий план або оновити профільні параметри.

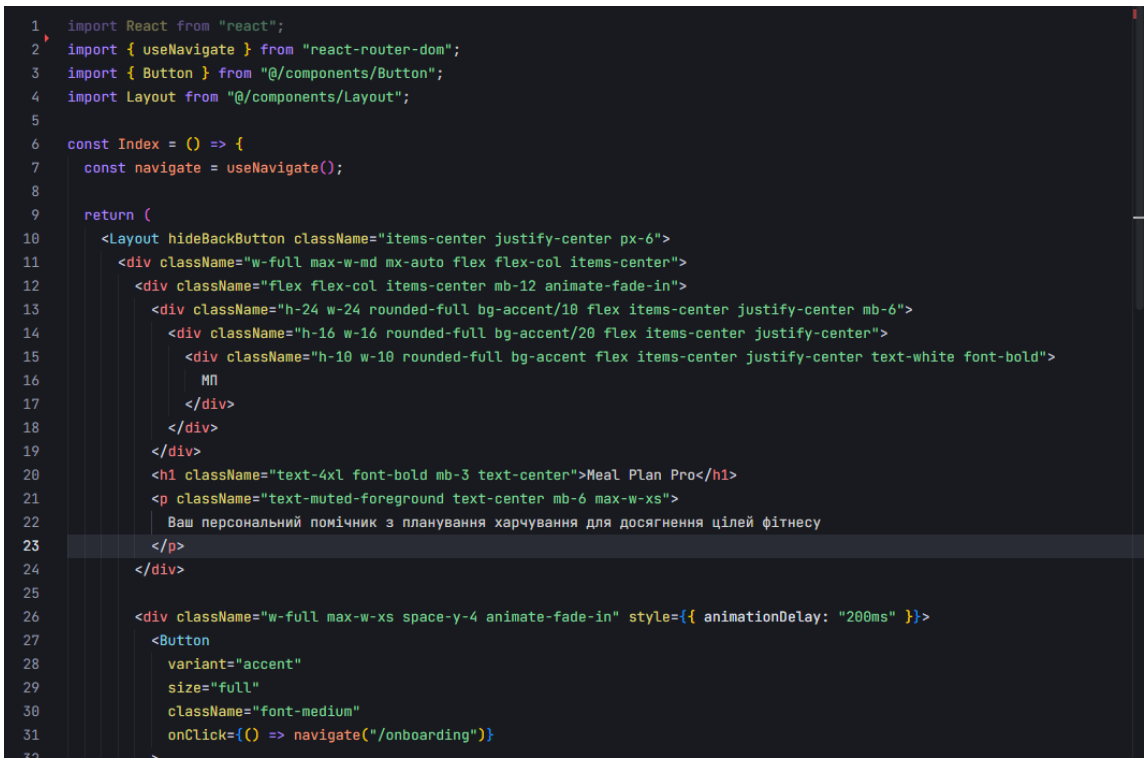
Дана схема демонструє зв’язки між основними модулями додатку та описує логіку його функціонування. Завдяки такому підходу мобільний додаток може обробляти користувацькі запити в режимі реального часу, адаптувати рекомендації до потреб користувачів та забезпечувати максимально точний підбір раціону.

Архітектура системи побудована з урахуванням модульності, що дозволяє легко масштабувати продукт, додавати нові функції та інтегрувати додаткові сервіси без необхідності кардинальних змін у кодовій базі.

3.4 Програмна реалізація інформаційної системи

Розробка мобільного додатку здійснювалася у середовищі Visual Studio Code із застосуванням бібліотеки Capacitor.

На рисунку 3.6 зображено код головної сторінки Index.tsx, початкова сторінка з назвою додатка та кнопкою ”Почати”.



```tsx
import React from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import Layout from "@/components/Layout";

const Index = () => {
  const navigate = useNavigate();

  return (
    <Layout hideBackButton className="items-center justify-center px-6">
      <div className="w-full max-w-md mx-auto flex flex-col items-center">
        <div className="flex flex-col items-center mb-12 animate-fade-in">
          <div className="h-24 w-24 rounded-full bg-accent/10 flex items-center justify-center mb-6">
            <div className="h-16 w-16 rounded-full bg-accent/20 flex items-center justify-center">
              <div className="h-10 w-10 rounded-full bg-accent flex items-center justify-center text-white font-bold">
                MP
              </div>
            </div>
          </div>
          <h1 className="text-4xl font-bold mb-3 text-center">Meal Plan Pro</h1>
          <p className="text-muted-foreground text-center mb-6 max-w-xs">
            Ваш персональний помічник з планування харчування для досягнення цілей фітнесу
          </p>
        </div>

        <div className="w-full max-w-xs space-y-4 animate-fade-in" style={{ animationDelay: "200ms" }}>
          <Button
            variant="accent"
            size="full"
            className="font-medium"
            onClick={() => navigate("/onboarding")}
```

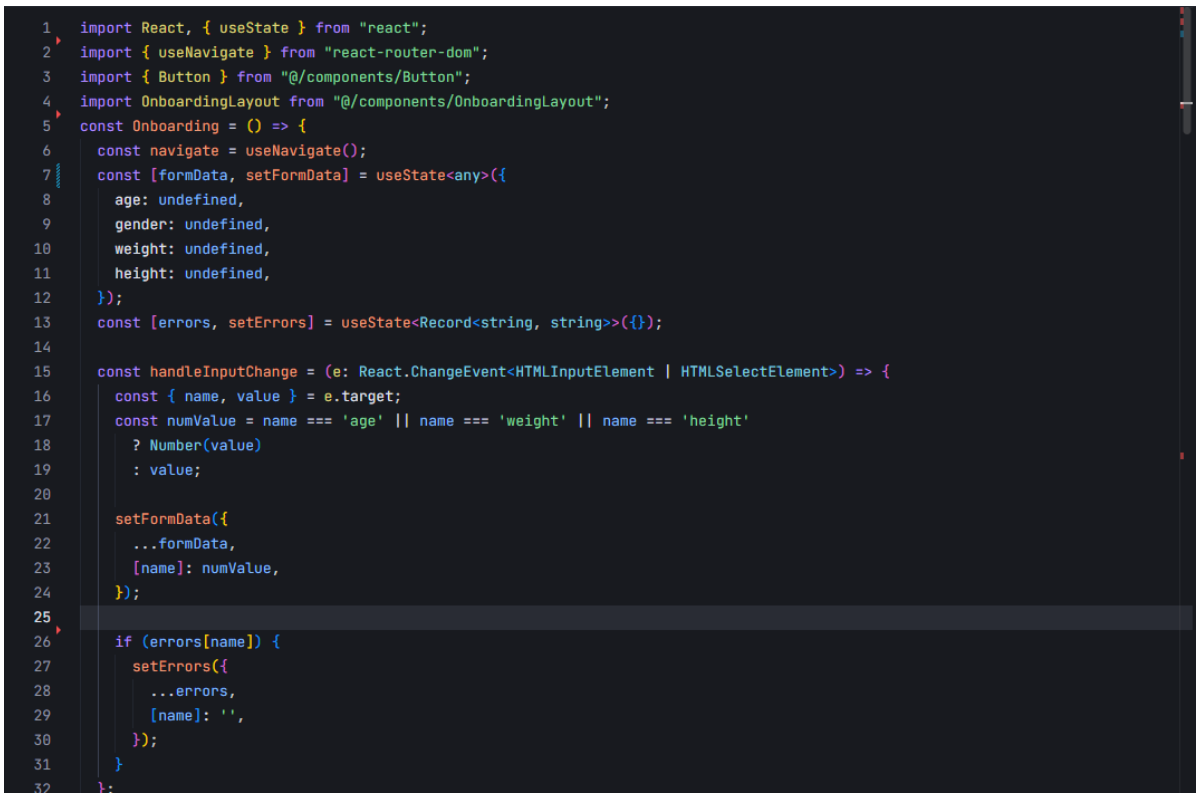
Рисунок 3.6 – Код сторінки Index.tsx

На рисунку 3.7 представлений код інтерфейсу заповнення фізичних показників, а саме:

– вибір статі: користувач обирає свою стать, що впливає на розрахунки метаболізму;

– введення віку: користувач вказує свій вік;

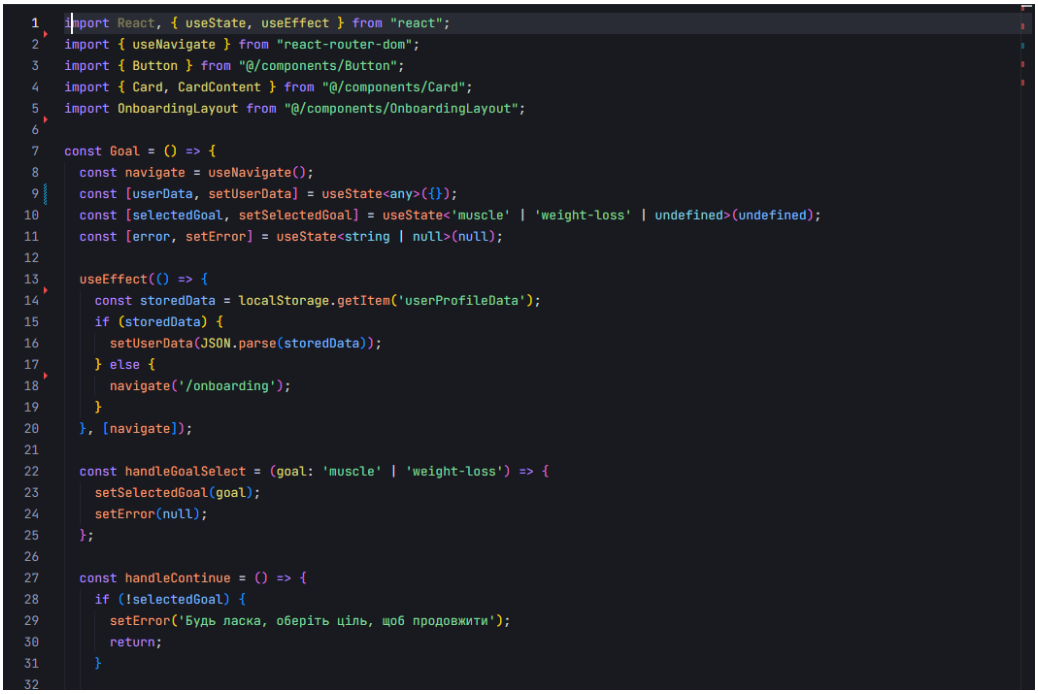
– введення ваги та зросту: користувач вводить свої фізичні параметри для розрахунку індивідуальних потреб.



```tsx
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import OnboardingLayout from "@/components/OnboardingLayout";
const Onboarding = () => {
  const navigate = useNavigate();
  const [formData, setFormData] = useState<any>({
    age: undefined,
    gender: undefined,
    weight: undefined,
    height: undefined,
  });
  const [errors, setErrors] = useState<Record<string, string>>({});

  const handleInputChange = (e: React.ChangeEvent<HTMLInputElement | HTMLSelectElement>) => {
    const { name, value } = e.target;
    const numValue = name === 'age' || name === 'weight' || name === 'height'
      ? Number(value)
      : value;

    setFormData({
      ...formData,
      [name]: numValue,
    });

    if (errors[name]) {
      setErrors({
        ...errors,
        [name]: '',
      });
    }
  };
```

Рисунок 3.7 – Код сторінки Onboarding.tsx

На рисунку 3.8 зображено код сторінки Goal.tsx, де користувач обирає мету харчування (схуднення, підтримка ваги, набір маси).



```tsx
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import { Card, CardContent } from "@/components/Card";
import OnboardingLayout from "@/components/OnboardingLayout";

const Goal = () => {
  const navigate = useNavigate();
  const [userData, setUserData] = useState<any>({});
  const [selectedGoal, setSelectedGoal] = useState<'muscle' | 'weight-loss' | undefined>(undefined);
  const [error, setError] = useState<string | null>(null);

  useEffect(() => {
    const storedData = localStorage.getItem('userProfileData');
    if (storedData) {
      setUserData(JSON.parse(storedData));
    } else {
      navigate('/onboarding');
    }
  }, [navigate]);

  const handleGoalSelect = (goal: 'muscle' | 'weight-loss') => {
    setSelectedGoal(goal);
    setError(null);
  };

  const handleContinue = () => {
    if (!selectedGoal) {
      setError('Будь ласка, оберіть ціль, щоб продовжити');
      return;
    }
```

Рисунок 3.8 – Код сторінки Goal.tsx

Код сторінки Diet.tsx, зображений на рисунку 3.9, призначений для вибору типу дієти, де користувач обирає дієтичні вподобання.

```tsx
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import { Card, CardContent } from "@/components/Card";
import OnboardingLayout from "@/components/OnboardingLayout";


const Diet = () => {
  const navigate = useNavigate();
  const [userData, setUserData] = useState<any>({});
  const [selectedDiet, setSelectedDiet] = useState<'omnivore' | 'vegetarian' | undefined>(undefined);
  const [error, setError] = useState<string | null>(null);

  useEffect(() => {
    const storedData = localStorage.getItem('userProfileData');
    if (storedData) {
      setUserData(JSON.parse(storedData));
    } else {
      navigate('/onboarding');
    }
  }, [navigate]);

  const handleDietSelect = (diet: 'omnivore' | 'vegetarian') => {
    setSelectedDiet(diet);
    setError(null);
  };

  const handleContinue = () => {
    if (!selectedDiet) {
      setError('Будь ласка, оберіть тип харчування, щоб продовжити');
      return;
```

Рисунок 3.9 – Код сторінки Diet.tsx

Головна сторінка з планом харчування. MealPlan.ts – це основна сторінка додатку (рис. 3.10), де відображається персоналізований план харчування користувача. На цій сторінці представлені:

– картки прийомів їжі: сніданок, обід, вечеря та перекуски з назвою страви, описом та інформацією про поживну цінність (калорії, білки, жири, вуглеводи);

– селектор дня тижня: дозволяє переглядати план на різні дні тижня;

– індикатори виконання: показують, які прийоми їжі вже відмічені як виконані;

– кнопка оновлення плану: дозволяє згенерувати новий план харчування;

– кнопка AI: на кожній картці страви, дозволяє згенерувати альтернативний варіант для конкретного прийому їжі;

– користувач може натиснути на картку страви, щоб відмітити її як виконану, або використати голосове керування для цієї дії.

```tsx
import { GoogleGenAI } from "@google/genai";
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import Layout from "@/components/Layout";
import MealCard, { Meal } from "@/components/MealCard";
import { useMealPlan } from "@/hooks/useMealPlan";
import { Check, ArrowRight, Calendar, Mic, Sparkles } from "lucide-react";
import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue } from "@/components/ui/select";
import geminiService from "@/services/geminiService";
import ApiKeyService from "@/services/apiKeyService";
import { toast } from "@/hooks/use-toast";
const MealPlan = () => {
  const navigate = useNavigate();
  const [userData, setUserData] = useState<any>({});

  const [loading, setLoading] = useState<boolean>(true);
  const [weeklyView, setWeeklyView] = useState<boolean>(false);
  const [selectedDay, setSelectedDay] = useState<string>("Today");
  const [completedMeals, setCompletedMeals] = useState<string[]>([]);
  const [isListening, setIsListening] = useState<boolean>(false);
  const [recognition, setRecognition] = useState<any>(null);
  const [generatingMeal, setGeneratingMeal] = useState<string | null>(null);

  useEffect(() => {
    const storedData = localStorage.getItem('userProfileData');
    if (storedData) {
      setUserData(JSON.parse(storedData));
      console.log("storedData", JSON.parse(storedData))
      setTimeout(() => {
        setLoading(false);
      }, 800);
```

Рисунок 3.10 – Код сторінки MealPlan.tsx

За результатами даної роботи були опубліковані тези «Інформаційна система автоматизованого підбору дієти та харчового плану» на Міжнародній науково-практичній інтернет-конференції студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2024-2025 рр.) [3].

3.5 Тестування роботи системи

Після вдалого встановлення додатку на екрані з’явиться його значок (рис. 3.11).

Рисунок 3.11 – Іконка додатку

Стартове вікно додатку: коли користувач відкриває додаток, він бачить стартову сторінку (рис. 3.12) з привітанням та кнопкою для початку роботи.

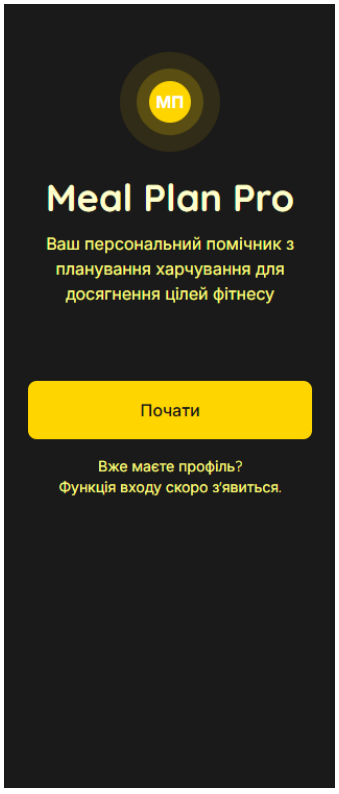


Рисунок 3.12 – Стартове вікно додатку

Ця сторінка є точкою входу в додаток і пропонує користувачу розпочати процес створення персоналізованого плану харчування.

Вікно для введення даних користувача: після натискання кнопки "Почати", користувач переходить на сторінку введення особистих даних (рис. 3.13)

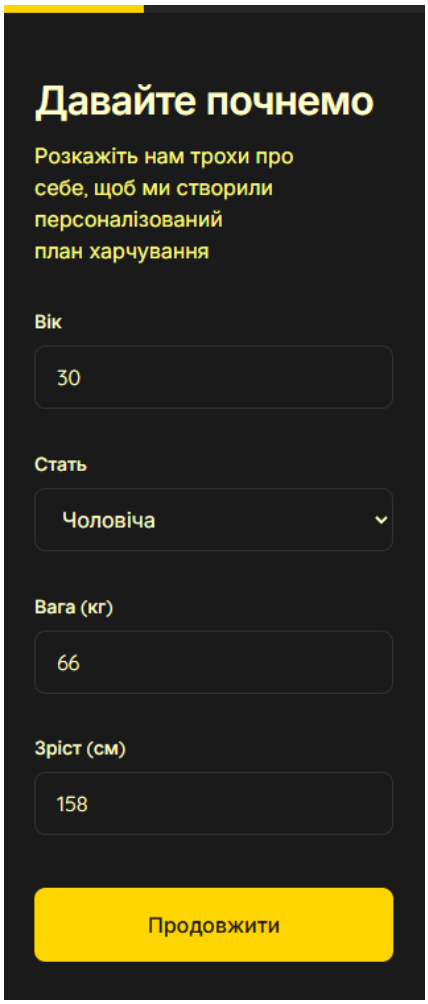


Рисунок 3.13 – Вікно для вводу персональних даних користувача

В цьому вікні пропонується заповнити форму з такими параметрами:

– стать (чоловіча/жіноча);
– вік;
– вага (в кілограмах);
– зріст (в сантиметрах).

Ці дані необхідні для розрахунку індивідуальних потреб користувача в калоріях та макронутрієнтах. Важливо перевірити, що всі поля форми працюють

коректно, валідація даних працює правильно, і користувач може легко перейти до наступного кроку.

На рисунку 3.14 продемонстроване вікно, де користувач повинен обрати мету фітнес-цілей.

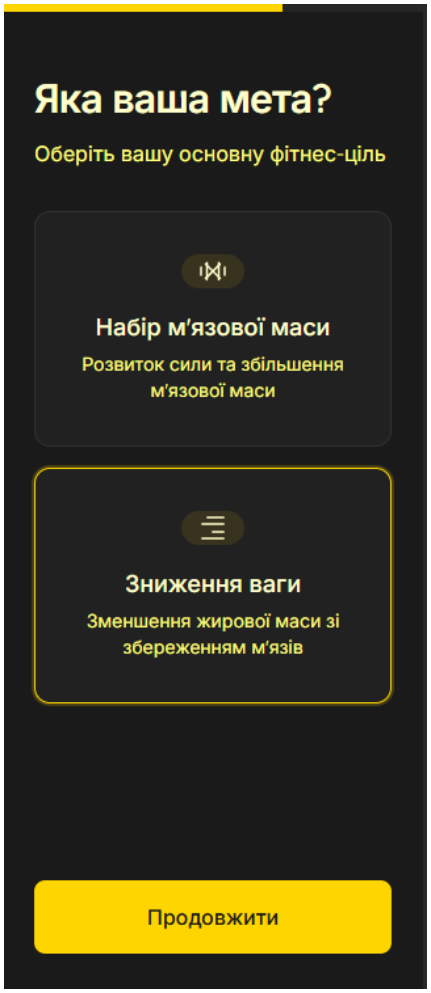


Рисунок 3.14 – Вікно для вказання мети користувача

Користувачу необхідно обрати один з варіантів для того, щоб продовжити заповнення форми:

– набір м’язової маси;
– зниження ваги.

На рисунку 3.15 показано фінальне вікно форми з назвою тип харчування.

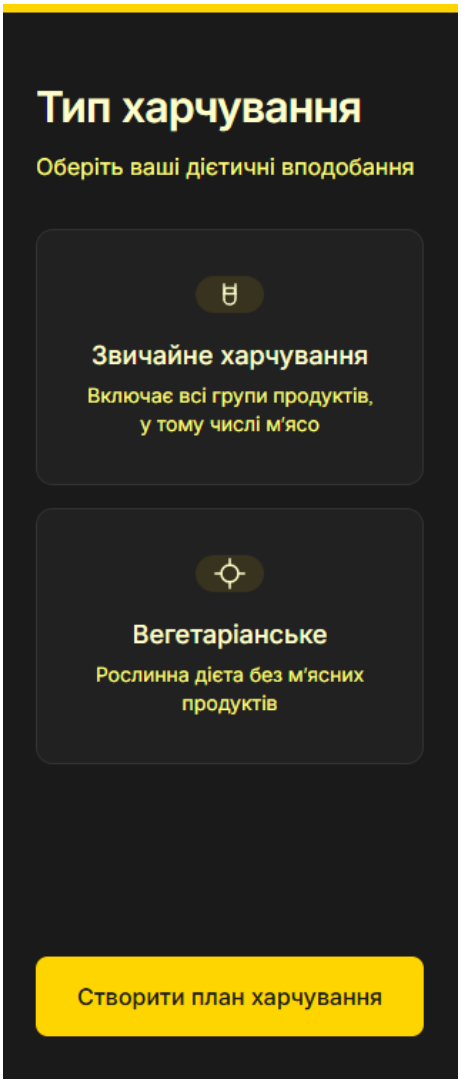


Рисунок 3.15 – Вікно для вказання типу харчування користувача

Користувачу потрібно обрати його власні дієтичні вподобання з запропонованих двох варіантів:

– звичайне харчування;
– вегетаріанське.

Після заповнення всіх необхідних параметрів користувач переходить до основної сторінки додатку – сторінки плану харчування (рис. 3.16), яка містить детальну інформацію про сформований раціон.

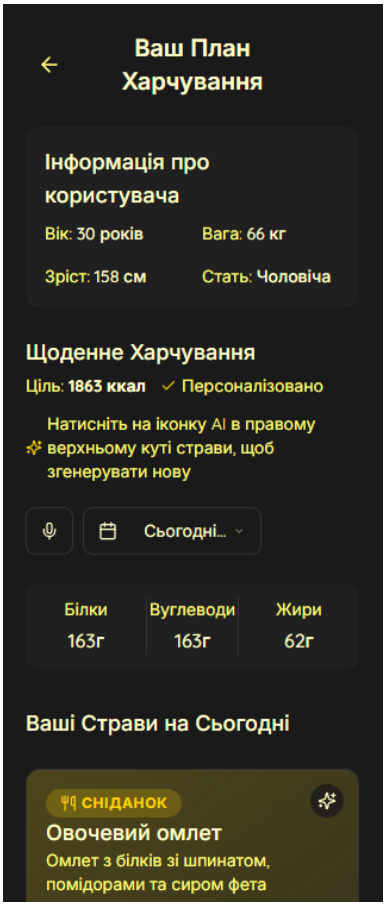


Рисунок 3.16 – Вікно згенерованого харчового плану з переглядом інформації про дані користувача

Інтерфейс цієї сторінки структурований таким чином, щоб забезпечити зручну навігацію, швидкий доступ до ключової інформації та можливість модифікації плану відповідно до персональних вподобань. На верхній частині сторінки розташований інформаційний блок, який містить персональні параметри користувача: вік, вага, зріст, рівень активності, а також щоденну норму калорій згідно з розрахованою моделлю харчування.

На рисунку 3.17 продемонстровано, що додаток підтримує перегляд плану харчування на різні дні тижня.

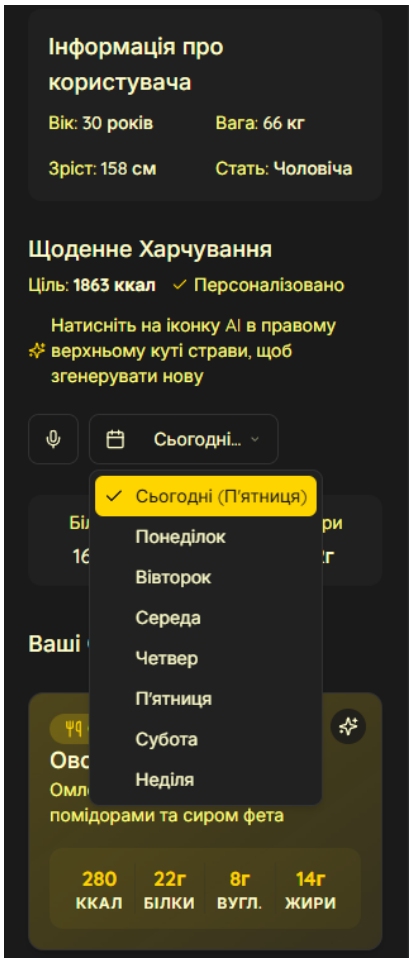


Рисунок 3.17 – Список, де можна обрати день тижня та переглянути харчовий план на обраний день

Вбудований селектор дати дозволяє вибрати потрібний день і переглянути заплановане меню. Всі розрахунки здійснюються автоматично з урахуванням змін у параметрах користувача та його раціону.

У додатку реалізовано підтримку голосових команд для взаємодії з планом харчування (рис. 3.18).

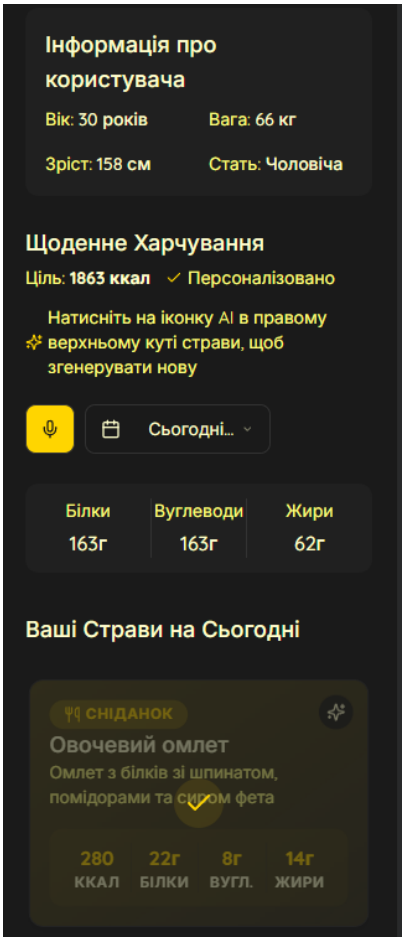


Рисунок 3.18 – Демонстрація роботи голосового вводу для відмітки про вже вживану страву

Користувач може вимовити назву страви, і система автоматично відмітить її як виконану. Голосове керування підвищує зручність користування, особливо якщо користувач не може фізично взаємодіяти з екраном.

На рисунку 3.19 зображено вигляд створених карток зі стравами.

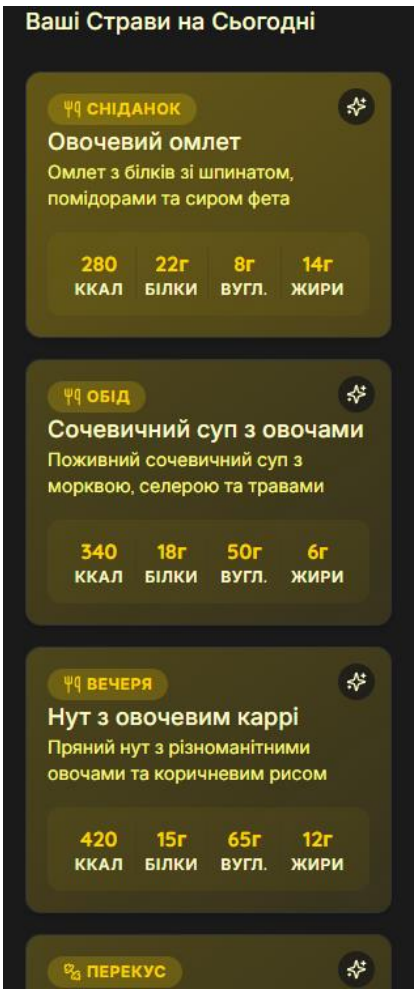


Рисунок 3.19 – Вигляд створених карток зі стравами

Як видно з рисунка 3.19, кожна картка містить назву страви та її категорію (сніданок, обід, вечеря, перекус), калорійність кожної порції, розподіл макронутрієнтів (грам білків, жирів, вуглеводів), короткий опис страви та її склад.

Якщо запропоноване меню не відповідає вподобанням користувача, він може натиснути іконку AI на картці страви (рис. 3.20).

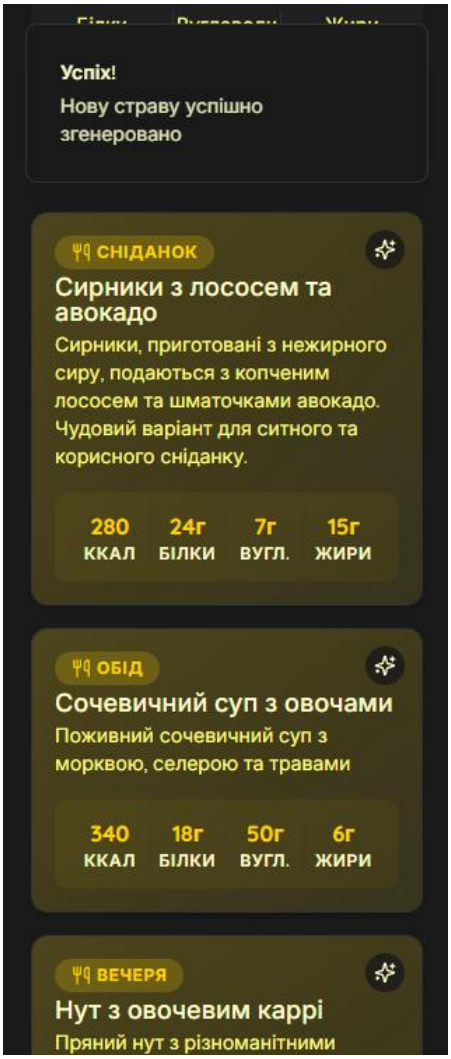


Рисунок 3.20 – Демонстрація зміни страви за допомогою AI

Система відправить запит до Google GenAI, який створить альтернативну страву з подібним рівнем калорійності та макронутрієнтів. Нова страва автоматично замінить попередню, зберігаючи баланс харчування.

Якщо користувач хоче повністю змінити свій раціон, він може натиснути кнопку "Оновити план харчування" (рис. 3.21).

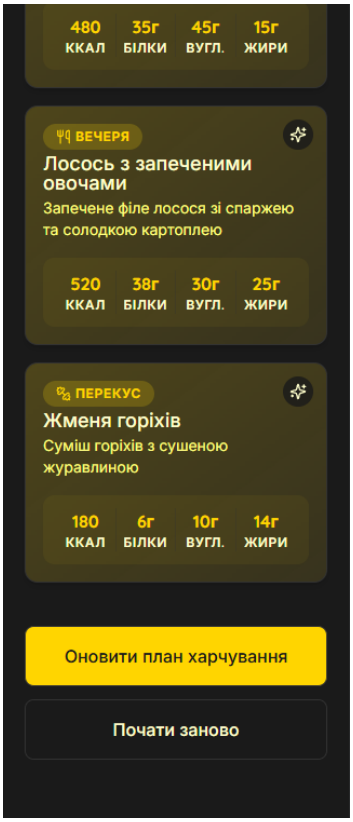


Рисунок 3.21 – Можливість оновити весь харчовий план уразі, якщо користувачу він не підійшов або не сподобався

Система згенерує новий варіант меню, адаптований до його поточних параметрів та вподобань. Також передбачена можливість повного скидання даних користувача, що дозволяє повернутися до стартового екрана та розпочати нове планування харчування.

Інтерфейс додатку має такі особливості:

— адаптивний дизайн: інтерфейс адаптується до різних розмірів екрану, забезпечуючи зручне використання як на мобільних пристроях, так і на комп'ютерах;

— анімації: елементи інтерфейсу мають плавні анімації, які покращують користувацький досвід;

— індикатори завантаження: під час завантаження даних або генерації нової страви відображаються індикатори завантаження;

— сповіщення: додаток використовує систему сповіщень для інформування користувача про успішні дії або помилки.

3.6 Висновки

Розроблена структура інформаційної системи забезпечує чітку організацію взаємодії між компонентами, що підвищує її модульність та масштабованість. Використання UML-діаграм дозволило візуалізувати процеси, зокрема алгоритм автоматизованого підбору дієтичного плану та основні сценарії використання додатку. Запропонована архітектура відповідає вимогам персоналізованого планування харчування, що гарантує точність розрахунку та зручність взаємодії для користувачів.

Розроблений алгоритм роботи додатку враховує етапи введення даних, розрахунку оптимального раціону та його корекції за допомогою AI. Використання адаптивного механізму модифікації меню забезпечує високий рівень персоналізації без втручання користувача, а автоматичне оновлення параметрів сприяє точному прогнозуванню змін у харчових звичках. Тестування підтвердило коректність реалізації, стабільність роботи мобільного додатку та відповідність заявленим функціональним вимогам. Основним напрямом розвитку може стати розширення можливостей аналізу харчової цінності продуктів та впровадження додаткових рівнів інтерактивності для користувачів.

ВИСНОВКИ

В ході виконання бакалаврської кваліфікаційної роботи було проведено аналіз предметної області, що підтвердило актуальність розробки інформаційної системи для автоматизованого підбору дієти та харчового плану.

Розглянуто сучасні рішення, їх переваги та недоліки, що дало змогу визначити ключові вимоги до розроблюваного додатку.

Проведений аналіз та вибір оптимальних інформаційних технологій, зокрема застосування TypeScript, React та Capacitor, дозволив забезпечити кросплатформну сумісність і продуктивність інформаційної системи. Інтеграція Google GenAI підвищила рівень персоналізації, адаптуючи раціон під індивідуальні вподобання користувача без порушення балансу поживних речовин.

Запропонована IT-інфраструктура інформаційної системи ґрунтується на використанні хмарних сервісів Google Cloud, що забезпечують масштабованість та безпечне зберігання даних. Взаємодія між клієнтською та серверною частинами здійснюється через API, що дозволяє ефективно обробляти запити користувачів у режимі реального часу. Впровадження механізмів безпеки, таких як SSL-шифрування та рольова модель доступу, гарантує захист персональних даних.

Розроблена інформаційна система автоматизованого підбору дієти та харчового плану забезпечує високий рівень персоналізації та адаптивності для користувачів. Використання математичної моделі, заснованої на формулі Гарріса-Бенедикта, гарантує точність розрахунків калорійності та нутрієнтів, що робить систему ефективним інструментом для оптимізації харчового раціону.

Порівняно з існуючими рішеннями, система демонструє гнучкість у налаштуванні раціону, можливість корекції страв за допомогою AI та використання голосового керування. Це робить її зручною у використанні та дозволяє адаптувати харчовий план в режимі реального часу.

Проведено тестування додатку, що підтвердило ефективність роботи системи та відповідність поставленим задачам. Додаток демонструє стабільну роботу, забезпечує точність розрахунків та високий рівень персоналізації раціону.

Серед обмежень можна виділити відсутність онлайн-синхронізації між пристроями, що могло б покращити користувацький досвід. Перспективним напрямом розвитку є інтеграція розширених AI-модулів для аналізу харчової цінності продуктів та додаткові функції з урахуванням мікронутрієнтів та харчових обмежень.

Отже, розроблена інформаційна система демонструє ефективність у сфері автоматизованого планування харчування, поєднуючи точність алгоритмів, інтеграцію сучасних технологій та можливості персоналізації.

За результатами виконаної роботи опубліковано тези доповідей на тему «Інформаційна система автоматизованого підбору дієти та харчового плану» на Міжнародній науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2024-2025 рр.).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Перов І. О., Войцеховська О. О. Інформаційна система автоматизованого підбору дієти та харчового плану. *Міжнародна науково-практична інтернет-конференція студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи»:* збірник матеріалів конференції. Вінниця, 2025. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/viewFile/24960/20773.

2. Бужанська М. В., Давидович О. Я. Глобальні тренди харчування: переваги, недоліки та вплив на здоров'я. *Вісник ЛТЕУ. Технічні науки*. 2024. Т. 39. с.38-48. DOI: http://journals-lute.lviv.ua/index.php/visnyk-tech/article/view/1661. (дата звернення: 04.05.2025).

3. Rimal A. P., Fletcher S. M., McWatters K. H. Nutrition considerations in food selection. *The International Food and Agribusiness Management R\їview*. 2000. Vol. 3, Issue 1, P. 55-70. DOI: https://www.sciencedirect.com/science/article/abs/pii/S1096750800000264.

4. Harris, J. A., Benedict, F. G. A Biometric Study of Human Basal Metabolism. *Proceedings of the National Academy of Sciences*. 1918. № 4(12). С. 370–373. DOI: https://doi.org/10.1073/pnas.4.12.370 (дата звернення: 04.05.2025).

5. React. URL: https://react.dev/ (дата звернення: 04.05.2025).

6. TypeScript. URL: https://www.typescriptlang.org/ (дата звернення: 04.05.2025).

7. Capacitor. URL: https://capacitorjs.com/ (дата звернення: 04.05.2025).

8. World Health Organization. Healthy diet. 2018. URL: https://www.who.int/news-room/fact-sheets/detail/healthy-diet (дата звернення: 04.05.2025).

9. Khatiwada, P., Chatterjee, A., Bajpai, R. Analyzing the Impact of Healthy Behavior on Weight Change with a Provable Mathematical Model (Preprint). *Journal*

of Medical Internet Research. 2021, vol.1. PP. 4-27. DOI: https://doi.org/10.2196/preprints.25201 (дата звернення: 04.05.2025).

10. Zhang, Y., Jiang, L., Ping, H. Artificial Intelligence in Nutrition Research: Potential and Challenges. *Nutrition Reviews*. 2020. № 78(6). С. 576–583. DOI: https://doi.org/10.1093/nutrit/nuz068. (дата звернення: 04.05.2025).

11. MyFitnessPal. URL: https://www.myfitnesspal.com/ (дата звернення: 05.05.2025).

12. Nutritionix. URL: https://www.nutritionix.com/app (дата звернення: 05.05.2025).

13. Lose it!. URL: https://www.loseit.com/ (дата звернення: 05.05.2025).

14. ANDROID. URL: https://developer.android.com/develop (дата звернення: 06.09.2025).

15. IOS. URL: https://developer.apple.com/documentation/ (дата звернення: 06.05.2025).

16. Васильєв О. М. Програмування мовою Java. – Bohdan Books, 2022. – 696 с. – ISBN: 978-966-10-5879-7 URL: https://books.google.com.ua/books?id=D1lcEAAAQBAJ&printsec=frontcover&hl=ru&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false (дата звернення: 06.05.2025).

17. Hoffman J. Mastering Swift 5.3: Upgrade your knowledge and become an expert in the latest version of the Swift programming language. Birmingham: Packt Publishing, 2020. 418 с. ISBN 9781800569973, 1800569971 URL: https://www.google.com.ua/books/edition/Mastering_Swift_5_3/ow4LEAAAQBAJ?hl=ru&gbpv=1 (дата звернення: 06.05.2025).

18. Rajagopal S., Popat K., Meva D., Bajeja S. Advancements in Smart Computing and Information Security: *in Second International Conference, ASCIS 2023, Rajkot, India, December 7–9, 2023, Revised Selected Papers, Part IV*. Cham: Springer Nature Switzerland, 2024. 464 с. URL:

https://www.google.com.ua/books/edition/Advancements_in_Smart_Computing_and_Info/c5cFEQAAQBAJ?hl=ru&gbpv=0 (дата звернення: 06.05.2025).

19. Visual Studio Code. URL: https://code.visualstudio.com/ (дата звернення: 07.05.2025).

20. Android Studio. URL: https://developer.android.com/studio (дата звернення: 10.05.2025).

21. Google GenAI. URL: https://ai.google/ (дата звернення: 11.05.2025).

22. Формула Гарріса-Бенедикта. URL: https://www.thecalculatorsite.com/articles/health/bmr-formula.php (дата звернення: 08.05.2025).

Додаток А

(обов’язковий)

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ

Завідувач кафедри САІТ

_____ д.т.н., проф. Віталій МОКІН

“____”_____ 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на бакалаврську кваліфікаційну роботу
ІНФОРМАЦІЙНА СИСТЕМА АВТОМАТИЗОВАНОГО ПІДБОРУ ДІЄТИ ТА ХАРЧОВОГО ПЛАНУ
08-34.БКР.016.00.000 ТЗ

Керівник: Ph.D., ст. викл. каф.САІТ

_____ Ольга ВОЙЦЕХОВСЬКА

«__» _____ 2025 р.

Розробив студент гр. 2ІСТ-21б

_____ Ілля ПЕРОВ

«__» _____ 2025 р.

Вінниця 2025

1. Підстава для проведення робіт.

Підставою для виконання роботи є наказ №__по ВНТУ від «__» _____2025р., та індивідуальне завдання на БКР, затверджене протоколом №_ засідання кафедри САІТ від «__» _____2025р.

2. Джерела розробки:

1) Бужанська М. В., Давидович О. Я. Глобальні тренди харчування: переваги, недоліки та вплив на здоров'я. *Вісник ЛТЕУ. Технічні науки*. 2024. Т. 39. с.38-48. DOI: http://journals-lute.lviv.ua/index.php/visnyk-tech/article/view/1661.

2) Rimal A. P., Fletcher S. M., McWatters K. H. Nutrition considerations in food selection. *The International Food and Agribusiness Management Review*. 2000. Vol. 3, Issue 1, P. 55-70. DOI: https://www.sciencedirect.com/science/article/abs/pii/S1096750800000264.

3. Мета і призначення роботи.

Метою дослідження є розроблення інформаційної системи автоматизованого підбору дієти та харчового плану.

4. Вхідні дані для проведення робіт:

1) Дані про калорійність харчових продуктів;
2) Дані про рецепти страв здорового харчування.

5. Методи дослідження:

1) Аналіз вимог;
2) Аналіз потреб користувачів;
3) Математичне моделювання;
4) Програмна реалізація та тестування;
5) Інтеграція штучного інтелекту.

6. Етапи роботи і терміни їх виконання:

a) Аналіз предметної області _____–_____
b) Огляд аналогів _____–_____
c) Вибір оптимальних інформаційних технологій _____–_____
d) Розроблення інформаційної системи автоматизованого підбору дієти та харчового плану _____–_____
e) Оформлення матеріалів до захисту БКР _____–_____

7. Очікувані результати та порядок реалізації

Отримання інформаційної системи автоматизованого підбору дієти та харчового плану.

8. Вимоги до розробленої документації

Текстова та ілюстративна частини роботи оформлені у відповідності до вимог «Методичних вказівок до виконання бакалаврських кваліфікаційних робіт для студентів спеціальностей: 124 «Системний аналіз», 126 «Інформаційні системи та технології» (освітня програма «Прикладні інформаційні технології»)».

9. Порядок приймання роботи

Публічний захист «___» _____ 2025 р.
Початок роботи «___» _____ 2025 р.

Граничні терміни виконання БКР «___» _____ 2025 р.

Розробив студент групи 2ІСТ-21б _____ Ілля ПЕРОВ

Додаток Б
(обов'язковий)

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: «Інформаційна система автоматизованого підбору дієти та харчового плану»

Тип роботи: бакалаврська кваліфікаційна робота

Підрозділ: кафедра САІТ, ФІІТА, гр. 2ІСТ-21б

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 2,25 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

- ■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- □ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- □ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Віталій МОКІН, зав. каф. САІТ _____ (підпис)

Євгеній КРИЖАНОВСЬКИЙ, доц. каф. САІТ _____ (підпис)

Особа, відповідальна за перевірку _____ (підпис) Сергій ЖУКОВ

З висновком експертної комісії ознайомлений(-на)

Керівник _____ (підпис) Ольга ВОЙЦЕХОВСЬКА, PhD, ст. викл. каф. САІТ

Здобувач _____ (підпис) Ілля ПЕРОВ

Додаток В
(довідниковий)
Фрагмент лістингу програми

Код файлу додатку Index.tsx

```
import React from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import Layout from "@/components/Layout";

const Index = () => {
  const navigate = useNavigate();

  return (
    <Layout hideBackButton className="items-center justify-center px-6">
      <div className="w-full max-w-md mx-auto flex flex-col items-center">
        <div className="flex flex-col items-center mb-12 animate-fade-in">
          <div className="h-24 w-24 rounded-full bg-accent/10 flex items-center justify-center mb-6">
            <div className="h-16 w-16 rounded-full bg-accent/20 flex items-center justify-center">
              <div className="h-10 w-10 rounded-full bg-accent flex items-center justify-center text-white font-bold">
                МП
              </div>
            </div>
          </div>
          <h1 className="text-4xl font-bold mb-3 text-center">Meal Plan Pro</h1>
          <p className="text-muted-foreground text-center mb-6 max-w-xs">
            Ваш персональний помічник з планування харчування для досягнення цілей фітнесу
          </p>
        </div>

        <div className="w-full max-w-xs space-y-4 animate-fade-in" style={{ animationDelay: "200ms" }}>
          <Button
            variant="accent"
            size="full"
            className="font-medium"
            onClick={() => navigate("/onboarding")}
          >
            Почати
          </Button>

          <div className="text-center mt-8">
            <p className="text-sm text-muted-foreground">
              Вже маєте профіль? <br />
              Функція входу скоро з'явиться.
            </p>
          </div>
        </div>
      </div>
    </Layout>
  );
};

export default Index;
```

Код файлу додатку MealPlan.tsx

```
import { GoogleGenAI } from "@google/genai";
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { Button } from "@/components/Button";
import Layout from "@/components/Layout";
import MealCard, { Meal } from "@/components/MealCard";
import { useMealPlan } from "@/hooks/useMealPlan";
import { Check, ArrowRight, Calendar, Mic, Sparkles } from "lucide-react";
import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue } from "@/components/ui/select";
import geminiService from "@/services/geminiService";
import ApiKeyService from "@/services/apiKeyService";
import { toast } from "@/hooks/use-toast";
const MealPlan = () => {
  const navigate = useNavigate();
  const [userData, setUserData] = useState<any>({});

  const [loading, setLoading] = useState<boolean>(true);
  const [weeklyView, setWeeklyView] = useState<boolean>(false);
  const [selectedDay, setSelectedDay] = useState<string>("Today");
  const [completedMeals, setCompletedMeals] = useState<string[]>([]);
  const [isListening, setIsListening] = useState<boolean>(false);
  const [recognition, setRecognition] = useState<any>(null);
  const [generatingMeal, setGeneratingMeal] = useState<string | null>(null);

  useEffect(() => {
    const storedData = localStorage.getItem('userProfileData');
    if (storedData) {
      setUserData(JSON.parse(storedData));
      console.log("storedData", JSON.parse(storedData))
      setTimeout(() => {
        setLoading(false);
      }, 800);
    } else {
      navigate('/onboarding');
    }
  }, [navigate]);

  const { mealPlan, setMealPlan} = useMealPlan(userData || {});
  console.log("mealPlan", mealPlan)
  const weeklyMealPlan = mealPlan?.weeklyPlan || {};

  const getCurrentDayName = () => {
    const days = ["Неділя", "Понеділок", "Вівторок", "Середа", "Четвер", "П'ятниця", "Субота"];
    const today = new Date().getDay();
    return days[today];
  };

  const handleRegenerate = () => {
    setLoading(true);
    const updatedUserData = {
      ...userData,
      timestamp: Date.now()
    };
    setUserData(updatedUserData);
    localStorage.setItem('userProfileData', JSON.stringify(updatedUserData));
```

```
  setTimeout(() => {
    setLoading(false);
  }, 800);
};

const handleReset = () => {
  localStorage.removeItem('userProfileData');
  navigate('/');
};

const toggleView = () => {
  setWeeklyView(!weeklyView);
  setSelectedDay("Today");
};

const handleDaySelect = (day: string) => {
  setSelectedDay(day);
  setWeeklyView(false);
const getSelectedDayMeals = () => {
  if (!weeklyMealPlan) return mealPlan?.meals || [];
  if (selectedDay === "Today") {
    return mealPlan?.meals || [];
  }
  return weeklyMealPlan[selectedDay] || [];
};

const toggleMealCompletion = (mealId: string) => {
  setCompletedMeals(prev => {
    const newCompletedMeals = prev.includes(mealId)
      ? prev.filter(id => id !== mealId)
      : [...prev, mealId];
    return newCompletedMeals;
  });
};

const handleGenerateNewMeal = async (meal: Meal) => {
  try {
    setGeneratingMeal(meal.id);
    const apiKey = "AIzaSyBspMmDj1n1YXtWOi16wui04u1ZTW1kZyk";
    ApiKeyService.saveGeminiApiKey(apiKey);
    geminiService.setApiKey(apiKey);
    const params = {
      mealType: meal.type,
      calories: meal.calories,
      protein: meal.protein,
      carbs: meal.carbs,
      fat: meal.fat,
      diet: userData.diet
    };
    const newMeal = await geminiService.generateMeal(params, meal);
    console.log("newMeal", {newMeal, meals: getSelectedDayMeals()})
    if (!newMeal) {
      throw new Error('Не вдалося згенерувати нову страву');
    }
    const updatedMeals = getSelectedDayMeals().map(m =>
      m.id === meal.id ? newMeal : m
```

```
    );
    console.log("updatedMeals", updatedMeals)
    if (selectedDay === "Today") {
      const updatedUserData = {
        ...userData,
        mealPlanOverride: {
          ...userData.mealPlanOverride || {},
          meals: updatedMeals
        }
      };
    console.log("updatedUserData", updatedUserData)
    setMealPlan((prev) =>({...prev, meals:updatedMeals}) )
      setUserData(updatedUserData);
      localStorage.setItem('userProfileData', JSON.stringify(updatedUserData));
    } else {
      const updatedUserData = {
        ...userData,
        weeklyPlanOverride: {
          ...userData.weeklyPlanOverride || {},
          [selectedDay]: updatedMeals
        }
      };
      setUserData(updatedUserData);
      localStorage.setItem('userProfileData', JSON.stringify(updatedUserData));
    }

    toast({
      title: 'Успіх!',
      description: 'Нову страву успішно згенеровано',
    });
  } catch (error) {
    console.error('Помилка при генерації нової страви:', error);
    let errorMessage = 'Не вдалося згенерувати нову страву. Спробуйте ще раз.';
    if (error instanceof Error) {
      if (error.message.includes('API ключ не встановлено')) {
        errorMessage = 'API ключ Gemini не встановлено або недійсний. Будь ласка, введіть правильний ключ.';
        ApiKeyService.removeGeminiApiKey();
      } else if (error.message.includes('Помилка API: 400')) {
        errorMessage = 'Помилка запиту до Gemini API. Перевірте правильність API ключа.';
      } else if (error.message.includes('Помилка API: 401') || error.message.includes('Помилка API: 403')) {
        errorMessage = 'Помилка авторизації в Gemini API. Ваш API ключ недійсний або закінчився термін його
дії.';
        ApiKeyService.removeGeminiApiKey();
      } else if (error.message.includes('Помилка API: 429')) {
        errorMessage = 'Перевищено ліміт запитів до Gemini API. Спробуйте пізніше.';
      }
    }

    toast({
      title: 'Помилка',
      description: errorMessage,
      variant: 'destructive'
    });

    if (errorMessage.includes('API ключ') || errorMessage.includes('авторизації')) {
      setTimeout(() => {
        handleGenerateNewMeal(meal);
```

```
} finally {
setGeneratingMeal(null);
}
};

const startListening = () =&gt; {
if (&#39;webkitSpeechRecognition&#39; in window) {
const recognition = new (window as any).webkitSpeechRecognition();
recognition.continuous = false;
recognition.interimResults = false;
recognition.lang = &#39;uk-UA&#39;;
recognition.onstart = () =&gt; {
setIsListening(true);
};
```

Додаток Г
(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА СИСТЕМА АВТОМАТИЗОВАНОГО ПІДБОРУ ДІЄТИ ТА ХАРЧОВОГО ПЛАНУ

Нормоконтроль: к.т.н., доцент
_____Сергій ЖУКОВ
«_____» _____2025 р.

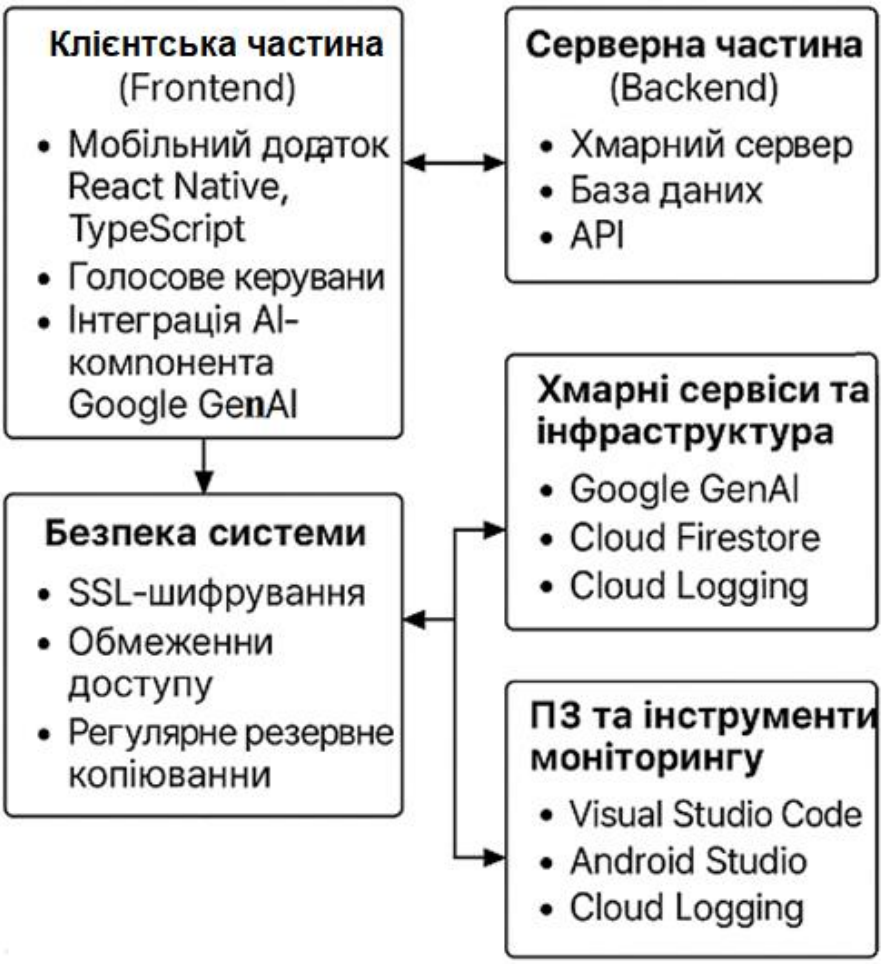


Рисунок Г.1 – Архітектура інформаційної системи

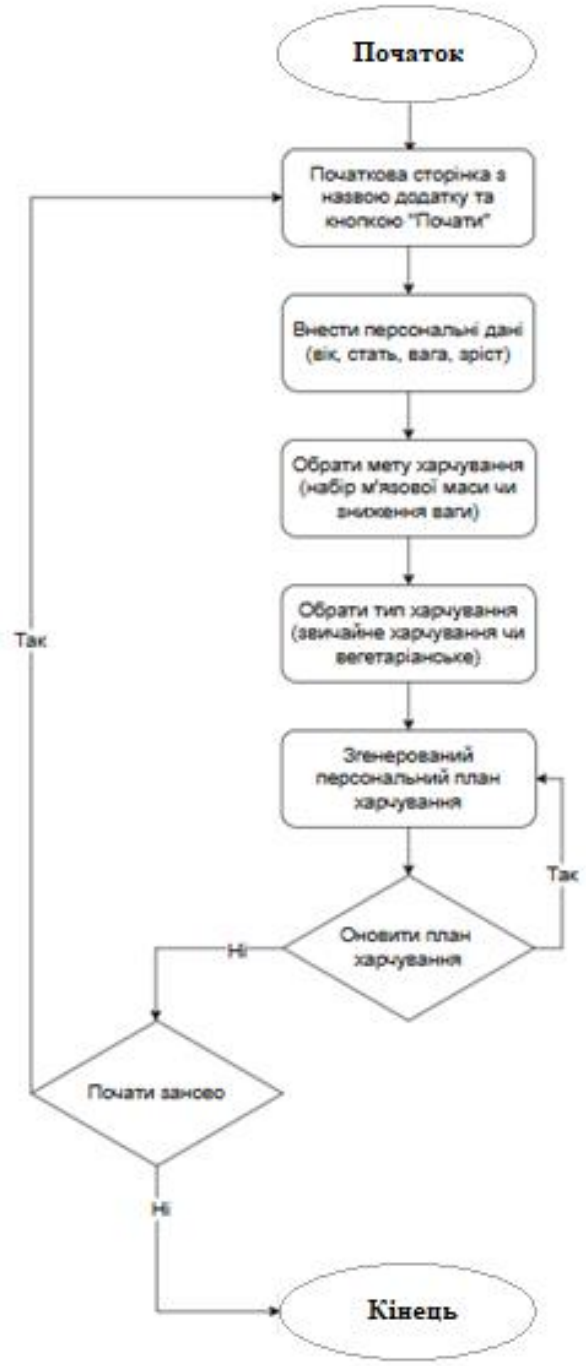


Рисунок Г.2 – Блок-схема алгоритму роботи додатку

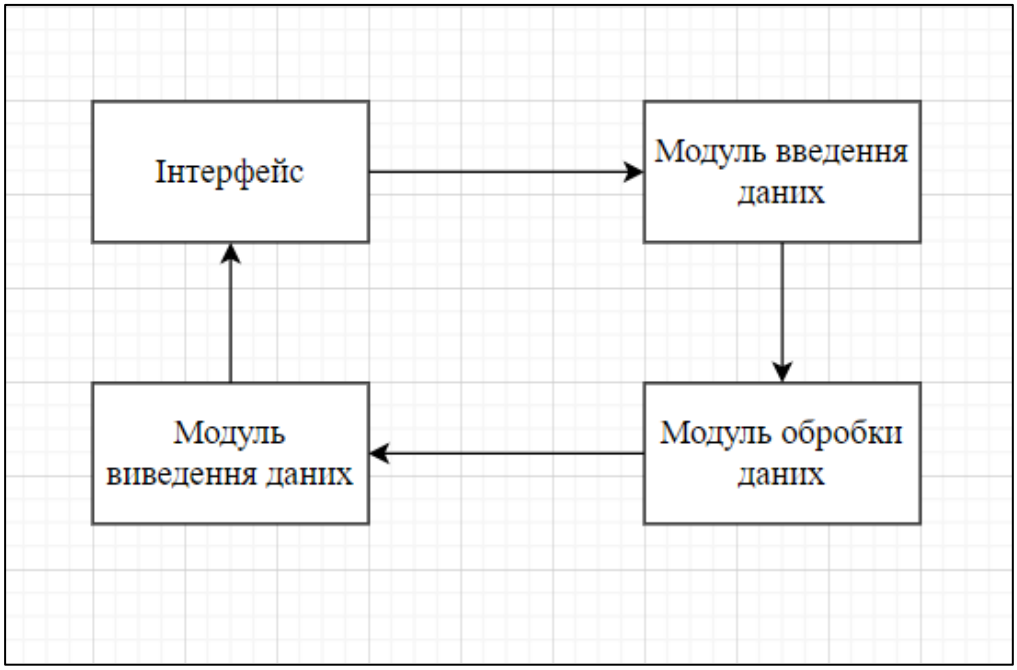


Рисунок Г.3 – Структурна схема функціонування мобільного додатку

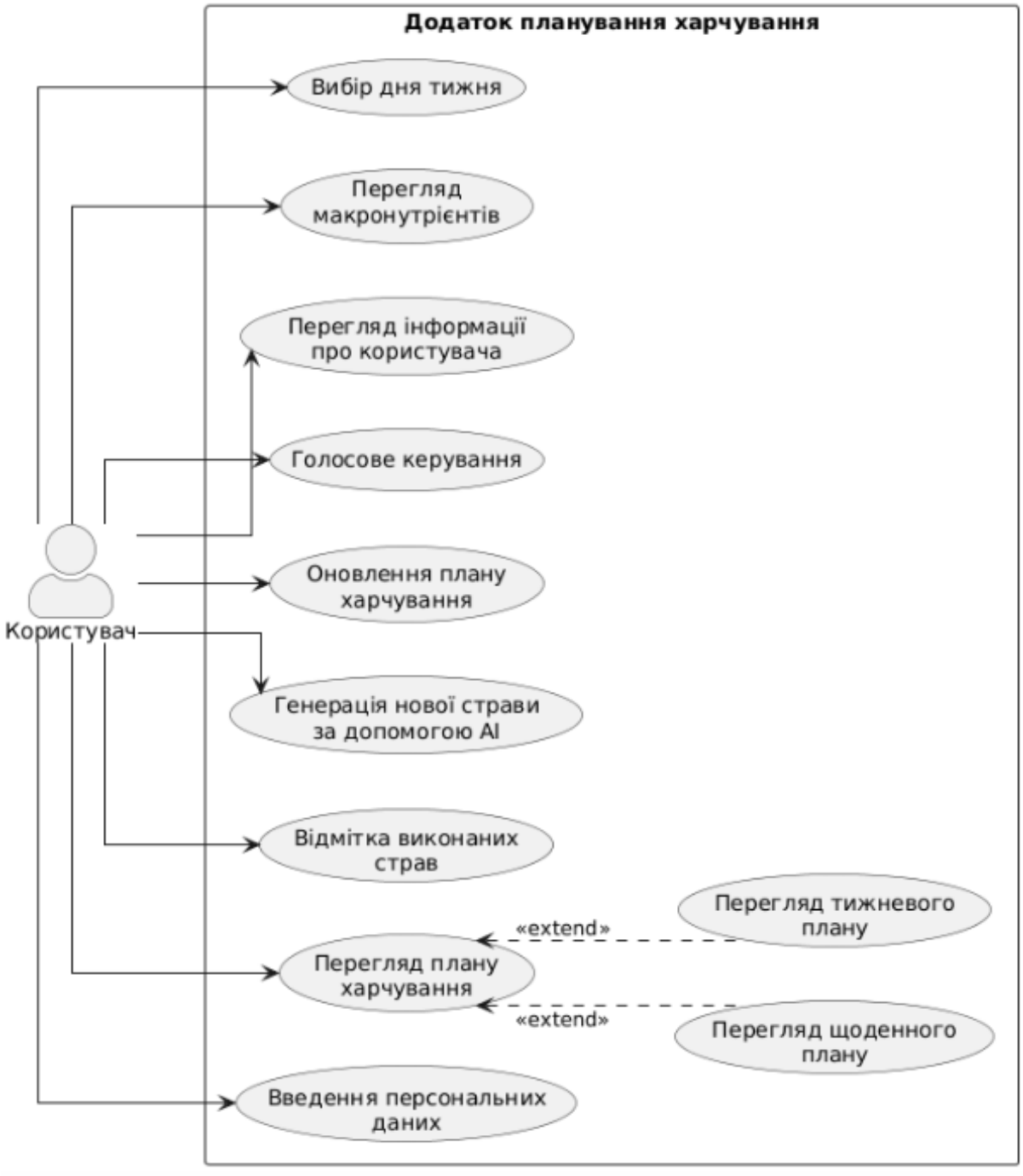


Рисунок Г.4 – Діаграма випадків використання додатку (Use Case Diagram)

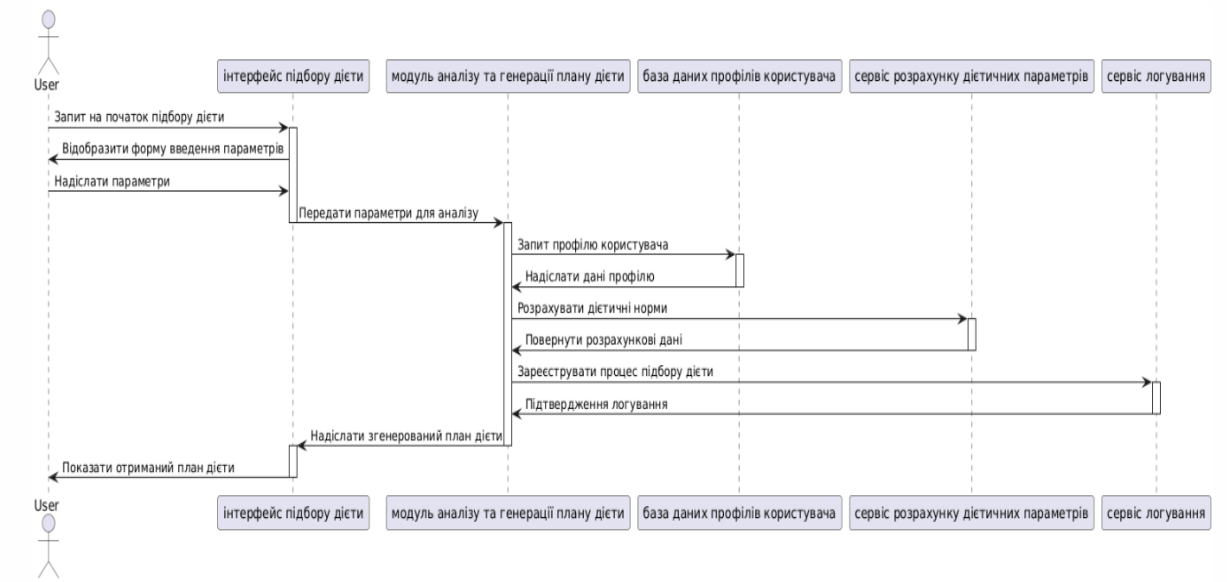


Рисунок Г.5 – Діаграма послідовності автоматичного підбору дієтичного плану

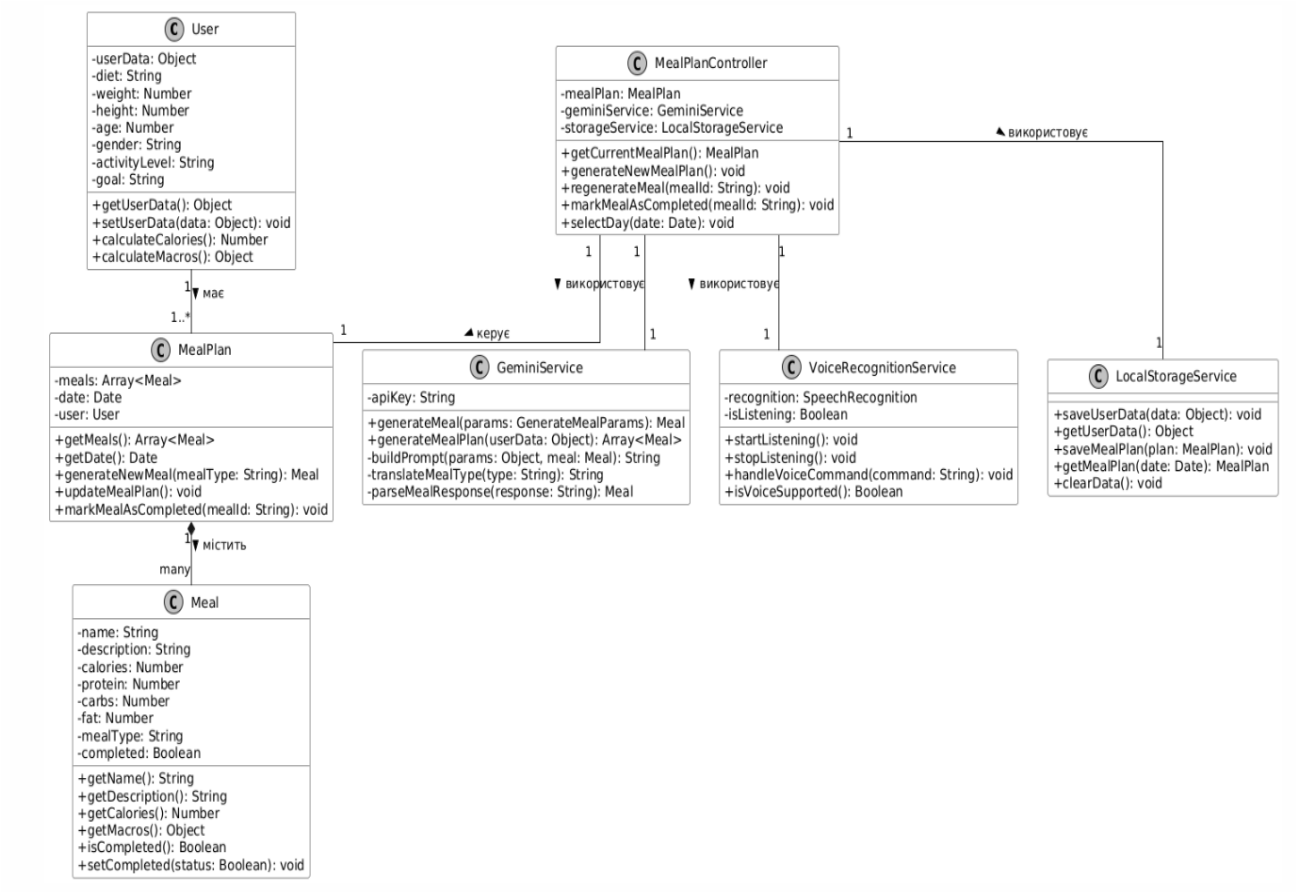


Рисунок Г.6 – Діаграма класів додатку

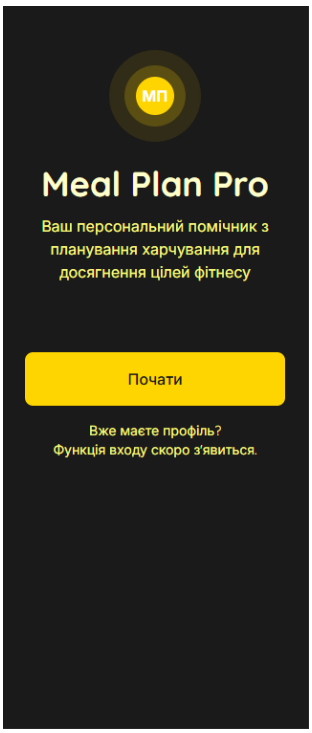


Рисунок Г.7 – Стартове вікно додатку

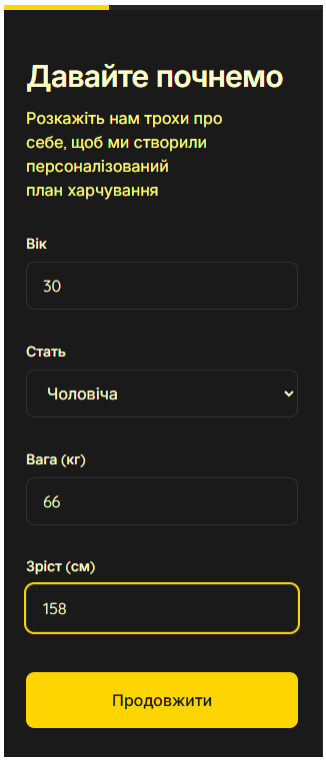


Рисунок Г.8 – Вікно для вводу персональних даних користувача

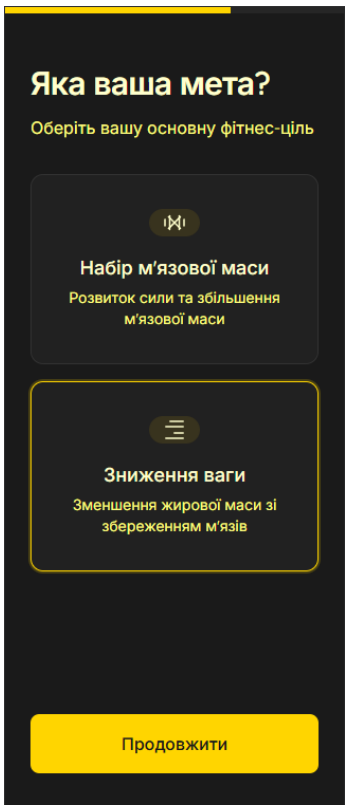


Рисунок Г.9 – Вікно для вказання мети користувача

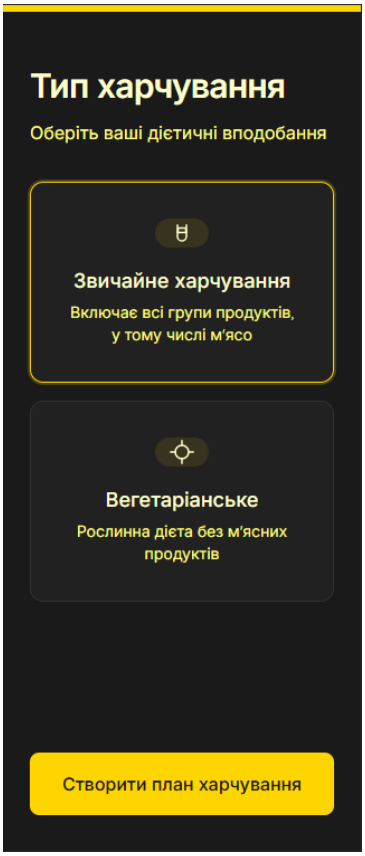


Рисунок Г.10 – Вікно для вказання типу харчування користувача

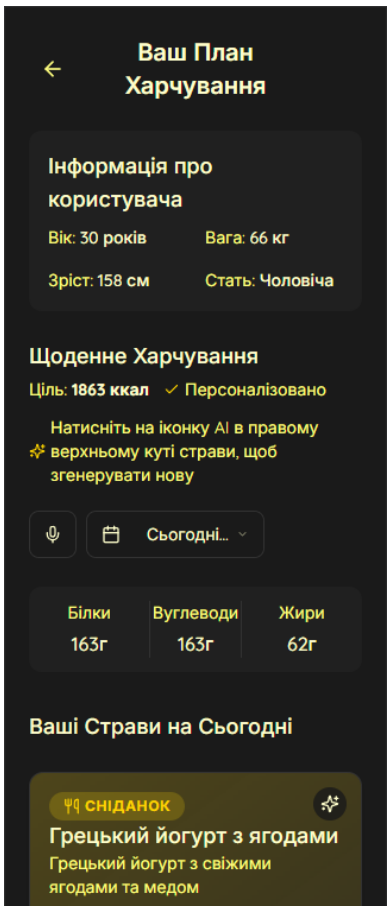


Рисунок Г.11 – Вікно згенерованого харчового плану з переглядом інформації про дані користувача