МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет Навчально-науковий інститут економіки та бізнес-освіти

Кафедра Економіки та цифрового бізнесу

Спеціальність 122 «Комп'ютерні науки»

Форма навчання Денна

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Будяка Ігоря Анатолійовича
(прізвище, ім'я, по батькові здобувача)

на тему Розробка Web-ресурсу для планування тренувань та аналізу динаміки фізичних показників
(повна назва теми)

за матеріалами

(повна назва бази дослідження)

науковий керівник к.т.н. _____ Селезньов М.Є.
(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 09 червня 2025р. № 12
Завідувач кафедри _____
(підпис)
к.е.н., доцент Радько В.М.
наук. ступень, вчене звання прізвище, ініціали

Кривий Ріг – 2025

ЗАТВЕРДЖЕНО
Наказ Міністерства освіти і науки, молоді та спорту України
29 березня 2012 року № 384

Форма № Н-9.01

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ ТА БІЗНЕС-ОСВІТИ
(повне найменування вищого навчального закладу)

Кафедра економіки та цифрового бізнесу
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри _____В.М. Радько

"07" квітня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА ЗДОБУВАЧУ

_____Будяку Ігорю Анатолійовичу_____

1. Тема роботи «Розробка Web-ресурсу для планування тренувань та аналізу динаміки фізичних показників»

науковий керівник роботи _____Селезньов Максим Євгенович_____,
затверджені наказом вищого навчального закладу від «04» квітня 2025 р. № 224-ст (д/ф)
№ 151-ст (з/ф)

2. Строк подання здобувачем роботи 31.05.2025р.

3. Зміст кваліфікаційної роботи бакалавра, об'єкт, предмет та мета дослідження:
Розділ 1 ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ WEB-РЕСУРСУ ДЛЯ ПЛАНУВАННЯ ТРЕНУВАНЬ ТА АНАЛІЗУ ДИНАМІКИ ФІЗИЧНИХ ПОКАЗНИКІВ
Розділ 2 ПРОЄКТУВАННЯ ТА РОЗРОБКА WEB-РЕСУРСУ ДЛЯ ПЛАНУВАННЯ ТРЕНУВАНЬ
Розділ 3 ФУНКЦІОНАЛ ТА ЕЛЕМЕНТИ ІНТЕРФЕЙСУ СТВОРЕНОГО WEB-РЕСУРСУ

Об'єкт дослідження - процес розробки Web-орієнтованих інформаційних систем для підтримки фізичної активності та аналізу фізіологічних показників користувачів

Предмет дослідження - функціональні та технічні аспекти розробки Web-ресурсу для планування тренувань та аналізу динаміки фізичних показників

Мета кваліфікаційної роботи бакалавра – створення інтерактивного Web-ресурсу для індивідуального планування тренувань, фіксування отриманих результатів та аналізу змін фізичних показників користувача

4. Дата видачі завдання 04.04.2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	до 28.04.2025р.	27.04.2025
2	Підготовка розділу 2	до 16.05.2025р.	15.05.2025
3	Підготовка розділу 3	до 30.05.2025р.	28.05.2025
4	Реєстрація завершеної дипломної роботи	до 31.05.2025р.	30.05.2025
5	Отримання відгуку від наукового керівника	03-04.06.2025р.	04.06.2025
6	Отримання зовнішньої рецензії	05-06.06.2025р.	06.06.2025
7	Перевірка кваліфікаційної роботи на плагіат	02-09.06.2025р.	04.06.2025
8	Попередній захист кваліфікаційної роботи на кафедрі	03.06.2025р.	03.06.2025
9	Допуск кафедрою кваліфікаційної роботи до захисту	09.06.2025р.	09.06.2025
10	Підготовка студента до захисту в ЕК	до 17.06.2025р.	17.06.2025

Завдання підготував науковий керівник _____ **Селезньов М. Є**
(підпис) (прізвище та ініціали)

Завдання одержав здобувач _____ **Будяк І. А.**
(підпис) (прізвище та ініціали)

Примітки:

1. Форму призначено для видачі завдання здобувачу на виконання кваліфікаційної роботи бакалавра і контролю за ходом роботи з боку кафедри.
2. Розробляється керівником кваліфікаційної роботи. Видається кафедрою.
3. Формат бланка А4 (210×297 мм), 2 сторінки.

РЕФЕРАТ

Робота містить 54 сторінки, 20 рисунків, 30 джерел, 2 додатки.

Об'єкт дослідження: процес розробки Web-орієнтованих інформаційних систем для підтримки фізичної активності та аналізу фізіологічних показників користувачів.

Предмет дослідження: функціональні та технічні аспекти розробки Web-ресурсу для планування тренувань та аналізу динаміки фізичних показників.

Мета дослідження: створення інтерактивного Web-ресурсу для індивідуального планування тренувань, фіксування отриманих результатів та аналізу змін фізичних показників користувача.

Методологічну основу розробки склали сучасні підходи до проєктування Web-систем, включаючи об’єктно-орієнтоване програмування, використання шаблонізаторів для створення динамічного інтерфейсу, структурування проєкту за принципами MVC, а також застосування бібліотек для візуалізації та взаємодії з користувачем. Для реалізації серверної логіки використано Flask - легкий фреймворк на Python, що забезпечує достатню гнучкість і простоту впровадження, а як СУБД обрано SQLite у поєднанні з SQLAlchemy, що забезпечило зручну й безпечну роботу з даними.

Область застосування: результати цієї роботи можуть бути використані для створення інтерактивних онлайн-платформ, що забезпечують користувачам інструменти для планування фізичної активності, контролю навантажень, аналізу змін фізичних показників та формування персоналізованих рекомендацій.

ІНФОРМАЦІЙНА СИСТЕМА, ПЛАНУВАННЯ, ТРЕНУВАННЯ, ФІЗИЧНА АКТИВНІСТЬ, WEB-РЕСУРС, WEB-РОЗРОБКА, WEB-ІНТЕРФЕЙС.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – База Даних;

СКБД – Система Керування Базами Даних;

API – Application Programming Interface;

HTML – HyperText Markup Language;

MVC – Model-View-Controller

ORM – Object-Relational Mapping;

REST – Representational State Transfer;

UML – Unified Modeling Language;

UX – User Experience.

ВСТУП

Сучасне суспільство перебуває в умовах швидкої цифрової трансформації, яка охоплює практично всі сфери життя - від економіки та освіти до особистого здоров’я та спорту. Зростаючий інтерес до здорового способу життя супроводжується активним впровадженням цифрових інструментів для моніторингу фізичної активності, планування тренувань і аналізу індивідуальних показників здоров’я. У цьому контексті Web-ресурси відіграють все більш важливу роль, надаючи користувачам можливість ефективно керувати власним фізичним станом у зручному, доступному та персоналізованому форматі.

Незважаючи на стрімкий темп цифровізації занять спортом, ринок все ще залишається досить фрагментованим. Більшість існуючих рішень орієнтовані лише на окремі функціональні аспекти, такі як підрахунок калорій, перегляд відеотренувань чи фіксація результатів отриманих і не забезпечують комплексного підходу до оцінки користувачем власної фізичної активності. Багато з існуючих рішень не передбачають можливості побудови гнучкого індивідуального плану тренувань, фіксації фізіологічних параметрів, відстеження динаміки прогресу або взаємодії з тренером. Таким чином, виникає об'єктивна потреба у створенні цілісного Web-ресурсу, який об'єднує всі ці функції в одному інтерфейсі.

Об'єкт дослідження: процес розробки Web-орієнтованих інформаційних систем для підтримки фізичної активності та аналізу фізіологічних показників користувачів.

Предмет дослідження: функціональні та технічні аспекти розробки Web-ресурсу для планування тренувань та аналізу динаміки фізичних показників.

Метою дослідження є створення інтерактивного Web-ресурсу для індивідуального планування тренувань, фіксування отриманих результатів та аналізу змін фізичних показників користувача.

В процесі дослідження вирішувалися наступні завдання:

- виконати аналіз предметної області та оцінити актуальність розробки Web-ресурсу;

- проаналізувати існуючі Web-ресурси для планування тренувань;

- обґрунтувати вибір архітектури та технологічного стеку розробки Web-ресурсу;

- розробити інформаційну модель та спроєктувати базу даних Web-ресурсу;

- розробити серверну та клієнтську частини Web-ресурсу;

- виконати загальний аналіз створеного Web-ресурсу.

В процесі проведення дослідження використовувалися наступні методики:

- аналіз предметної області;

- порівняльний аналіз;

- моделювання з використанням UML;

- об’єктно-орієнтований підхід;

- системний аналіз.

Практична значущість результатів дослідження полягає у створеному повноцінному Web-ресурсу, який дозволяє користувачам системно організовувати фізичну активність, здійснювати її облік і моніторинг, та може бути використаний як основа для подальшого розширення або впровадження в реальні умови.

РОЗДІЛ 1
ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ WEB-РЕСУРСУ ДЛЯ ПЛАНУВАННЯ ТРЕНУВАНЬ ТА АНАЛІЗУ ДИНАМІКИ ФІЗИЧНИХ ПОКАЗНИКІВ

1.1 Аналіз предметної області та актуальності розробки Web-ресурсу

У сучасному світі дедалі більше людей усвідомлюють важливість ведення здорового способу життя, регулярної фізичної активності та контролю за власним фізичним станом. Цей тренд зумовлений як загальним підвищенням обізнаності щодо користі фізичних вправ, так і стрімким розвитком цифрових технологій, які надають користувачам нові інструменти для зручного моніторингу, аналізу та планування власної активності. Суспільство перебуває в активному процесі цифрової трансформації, що охоплює майже всі сфери життя - від освіти та медицини до спорту та особистого розвитку. Тож не дивно, що технології проникають і у повсякденну фізичну активність, змінюючи уявлення людей про тренування, контроль результатів та досягнення поставлених фітнес-цілей.

Сьогодні фізична активність перестала бути винятковою прерогативою професійних спортсменів чи фітнес-ентузіастів. Все більше звичайних користувачів, незалежно від віку, статі чи рівня підготовки, прагнуть впроваджувати регулярні тренування у своє життя. Водночас зростає потреба не просто виконувати фізичні вправи, а й розуміти їхній вплив на організм, бачити прогрес, аналізувати ефективність методів та своєчасно вносити корективи. Такий підхід свідчить про поширення аналітичного мислення серед населення, яке більше не задовольняється простим виконанням рекомендацій, а прагне до глибшого розуміння процесів, що відбуваються у власному тілі.

Підтвердженням високої актуальності цифрових фітнес-рішень є статистичні дані. За даними аналітичного порталу Statista, у 2023 році кількість користувачів фітнес-додатків у світі перевищила 970 мільйонів осіб, а ринкова вартість цього

сегмента сягнула понад 19 мільярдів доларів США. І ці цифри продовжують зростати. Очікується, що до 2027 року кількість активних користувачів перевищить 1,2 мільярда. Така динаміка свідчить про глибоку інтеграцію цифрових фітнес-інструментів у повсякденне життя людей, про їх високу затребуваність та довіру з боку користувачів.

Пандемія COVID-19 стала додатковим каталізатором цього процесу, різко змінивши звички мільйонів людей у всьому світі. Через закриття спортивних клубів, фітнес-центрів і обмеження на пересування, багато хто був змушений шукати альтернативні способи підтримки фізичної форми. У таких умовах Web-застосунки та мобільні додатки стали справжнім порятунком - вони дозволили проводити тренування вдома, без потреби у спеціальному обладнанні чи участі інструктора, при цьому забезпечуючи інструменти для самостійного моніторингу прогресу. Навіть після зняття карантинних обмежень чимало людей залишилися прихильниками такого формату, відзначаючи його гнучкість, доступність і зручність.

Особливу роль у цьому процесі відіграють Web-платформи, що мають низку суттєвих переваг порівняно з традиційними методами фіксації даних, такими як паперові щоденники або неструктуровані файли. Web-ресурси забезпечують постійний доступ до інформації з будь-якого пристрою, автоматичне збереження та резервне копіювання даних, можливість синхронізації з фітнес-трекерами, смарт-годинниками, іншими сервісами для здоров’я. Вони також надають розширені функції візуалізації, зокрема графіки змін, діаграми, звіти - усе це допомагає краще зрозуміти тенденції, відстежити прогрес або виявити стагнацію.

Крім базових показників, таких як вага чи тривалість тренування, сучасні користувачі дедалі частіше звертають увагу на складніші фізіологічні параметри: частоту серцевих скорочень, рівень стресу, якість і тривалість сну, швидкість відновлення, варіабельність серцевого ритму, насичення крові киснем, рівень активності протягом доби тощо. Це свідчить про зростання вимог до функціоналу

фітнес-систем, адже вони мають бути здатними не лише зберігати широкий спектр даних, а й інтерпретувати їх з максимальною користю для користувача. Важливою є і можливість гнучкого налаштування - щоб користувач міг самостійно обирати, які показники його цікавлять, і як саме він бажає їх бачити та аналізувати.

На жаль, на ринку досі бракує комплексних рішень, які б одночасно поєднували зручність, функціональність і гнучкість. Багато додатків концентруються лише на окремих аспектах: підрахунку калорій, побудові стандартних тренувальних планів, веденні харчових щоденників тощо. Проте для користувача, який хоче бачити повну картину свого фізичного стану та мати реальні інструменти для аналізу і покращення результатів, цього недостатньо. Часто бракує можливості створювати індивідуальні програми, редагувати параметри, отримувати поради щодо корекції навантажень або бачити довготривалу динаміку у зручному вигляді.

Високий рівень зацікавленості користувачів у глибшому аналізі власного прогресу також створює передумови для впровадження більш просунутих технологій, зокрема алгоритмів штучного інтелекту. Наприклад, системи машинного навчання можуть автоматично виявляти плато у тренуваннях, прогнозувати зміну показників при дотриманні певного ритму занять або навіть попереджати про ризики перенавантаження. Це дає змогу користувачу не лише відстежувати процес, а й проактивно впливати на нього, змінювати навантаження, чергувати види активності, коригувати розклад, залежно від змін у власному тілі або самопочутті.

Особливо важливою є демократизація доступу до подібних технологій. Якщо раніше такі інструменти були доступні лише професійним спортсменам і тренерам, то сьогодні кожен власник смартфона або комп’ютера може скористатися ними. Це створює нові можливості для аматорів, які прагнуть більш серйозного підходу до тренувань, а також стимулює загальний розвиток культури здорового способу

життя. Проте це водночас і виклик - створити систему, яка буде зручною, зрозумілою та ефективною як для новачків, так і для досвідчених користувачів.

Сучасні глобальні тенденції свідчать про критично низький рівень фізичної активності населення: за даними ВООЗ, понад 80% підлітків і 27% дорослих у світі не відповідають рекомендованому рівню рухової активності. У Європі ситуація подібна: останній євробарометр показав, що 45% європейців взагалі ніколи не займаються спортом, і лише 38% тренуються принаймні раз на тиждень. В Україні дослідження також фіксують високу частку малорухомих громадян: за глобальним опитуванням ВООЗ, 76,7% українських підлітків мають недостатню фізичну активність. Така тенденція загрожує зростанням неінфекційних захворювань (серцево-судинних, цукрового діабету, раку тощо) та збільшенням навантаження на систему охорони здоров’я; експерти підраховують, що адекватна активність могла б запобігти до 5 млн передчасних смертей щорічно.

Одночасно світова урбанізація суттєво впливає на стиль життя: більше 55% населення проживає в містах (з прогнозом зростання до 68% до 2050 року). У щільно забудованих міських районах відсутність пішохідної інфраструктури та умов для активного відпочинку сприяє сидячому способу життя. Інфраструктура міст часто ускладнює активний транспорт (пішохідні та веломаршрути), що призводить до збільшення ризику ожиріння, діабету та інших захворювань. Тому у великих містах питання системного заохочення спорту і створення мотиваційних інструментів виходить на перший план.

Паралельно йде інтенсивна цифровізація суспільства. В Україні 2024 року 80% населення користується Інтернетом щодня (проти 72% торік), а проникнення інтернет-користувачів сягає близько 79% загальної кількості населення. Мобільні та смарт-пристрої майже повністю забезпечують цифрове охоплення: у країні понад 55 млн мобільних підключень (148,7% на душу населення). Це створює сприятливе середовище для впровадження Web- і мобільних рішень у будь-якій сфері, зокрема в галузі здоров’я і фітнесу. Наприклад, 82% користувачів інтернету у віці 18+ в

Україні ведуть активність у соцмережах, що свідчить про високий рівень цифрового спілкування і відкритість до нових онлайн-сервісів.

Водночас сучасний фітнес-ринок вимагає персоналізації: майже половина покоління міленіалів прагне отримувати якнайбільше даних про своє здоров'я, а понад 50% готові купувати «інтелектуальні» пристрої для відслідковування показників. Попит на носимі пристрої (трекери активності, «розумні» годинники) зростає двозначними темпами. Ці пристрої надають реальні показники пульсу, кроків, якості сну тощо в режимі реального часу, що дозволяє формувати індивідуальні рекомендації і відслідковувати динаміку прогресу. У глобальному масштабі спостерігається зростання попиту на віртуальні тренування та гібридні моделі тренувань, які поєднують заняття онлайн і офлайн. Наприклад, в Європі цифровий ринок фітнесу прогнозують досягнути понад 14 млрд євро до 2025 року з річним темпом зростання майже 35%. Нові тренди – використання штучного інтелекту для побудови персональних планів тренувань, інтеграція даних з різних пристроїв та залучення елементів гейміфікації.

З огляду на все вищезазначене, постає очевидна потреба у створенні нового, комплексного Web-ресурсу, який би поєднував функції тренувального планувальника, трекера фізичних показників, аналітичної панелі та консультативного інструменту. Такий ресурс має забезпечувати користувачу повний цикл роботи: від постановки цілей і планування активностей - до моніторингу виконання, аналізу прогресу та адаптації програми. Особливу увагу слід приділити юзабіліті інтерфейсу, кросплатформеності, можливості збереження історії змін, наявності візуалізацій, індивідуальних налаштувань, а також відкритості до подальшого розвитку та інтеграції з іншими сервісами.

Таким чином, розробка подібного ресурсу є не просто актуальною, а й стратегічно важливою в контексті загального розвитку цифрової культури здоров'я. Вона відповідає запитам сучасного користувача, який прагне не лише бути активним, а й робити це свідомо, ефективно та з використанням новітніх

технологій. Розробка комплексного Web-застосунку сприятиме мотивації до регулярних тренувань, підвищенню обізнаності про важливість руху та зниженню ризиків здоров'я на рівні громади і окремої особи.

1.2 Аналіз існуючих Web-ресурсів для планування тренувань

У сучасному цифровому світі існує величезна кількість Web-ресурсів та мобільних застосунків, покликаних допомогти людям у плануванні та проведенні тренувань, фіксації результатів і досягненні фітнес-цілей. Цей ринок активно розвивається, і практично кожна нова платформа намагається запропонувати щось унікальне або принаймні більш зручне для кінцевого користувача. Однак, попри велике різноманіття таких сервісів, користувачі досі стикаються з проблемою фрагментарності функціональності: багато продуктів орієнтуються лише на один аспект - харчування, відеотренування, фіксацію пробіжок або силові вправи - залишаючи поза увагою комплексний підхід до фізичного розвитку.

Одним із найвідоміших і найбільш розповсюджених застосунків є MyFitnessPal. Комплексний трекер харчування та активності, що має велику базу даних продуктів (мільйони одиниць харчування) та дозволяє вести облік калорій, макронутрієнтів і води. Додаток може синхронізуватися з іншими пристроями такими як фітнес-трекерами, додатками. Його ключовою перевагою є зручна та потужна система обліку харчування, зручний інтерфейс, доступний як безкоштовний, так і преміум-план; легкість у введенні даних, підтримка рецептів і домашніх страв. Серед недоліків додатку слід зазначити певну обмеженність у діях в безкоштовній версії додатку, постійні нагадування про перехід на платну підписку, висока вартість преміуму. Користувачі можуть швидко й точно вносити дані про спожиті продукти, контролювати добову норму калорій, баланс білків, жирів та вуглеводів, а також аналізувати загальний вплив харчування на фізичний стан. Однак, коли мова заходить про тренування, цей сервіс має значно обмеженіші

можливості. Хоча є функція реєстрації фізичної активності, вона не дозволяє створювати гнучкі плани, інтегрувати власні вправи або візуально аналізувати динаміку змін у тілі. Таким чином, MyFitnessPal радше виконує роль щоденника харчування, який частково підтримує фітнес-цілі, але не є повноцінним інструментом для ведення тренувального процесу.

Інший популярний застосунок - Freeletics. Фітнес-платформа, відома інтенсивними HIIT-тренуваннями без обтяжень, адаптованими за допомогою штучного інтелекту. Програма аналізує рівень користувача і динамічно змінює вправи під його прогрес. Додаток фокусується на тренуваннях із власною вагою та пропонує персоналізовані комплекси вправ. Платформа використовує базову інформацію про користувача - вік, стать, рівень підготовки - щоб автоматично створювати програму. Також до переваг функціоналу можна віднести якісні відео-демонстрації вправ, індивідуальний «коучинг» для різних цілей (сила, витривалість, спалювання жиру тощо). Водночас ця автоматизація стає обмеженням: користувач не може змінювати запропоновані вправи, поєднувати їх із власними, або коригувати інтенсивність відповідно до змін у самопочутті чи цілі. Також бракує зручних засобів для довготривалого збереження результатів і аналізу прогресу. Програма працює як «чорна скринька»: користувач отримує завдання, виконує його, але не бачить аналітики, яка дозволила б приймати зважені рішення щодо корекції підходу. Також серед недоліків слід зазначити здебільше використання HIIT-формату що може не підходити всім та сильне обмеження та відсутність деякого функціоналу в безкоштовній версії додатку.

Для силових тренувань часто використовуються такі застосунки, як Strong і FitNotes. FitNotes - простий лог для відстеження силових тренувань для операційної системи Android, описується досвідченими користувачами як «як папір, тільки краще». Користувач може вручну вводити вправи, підходи, вагу, кількість повторень і відпочинок. Повністю безкоштовний та працює без необхідності підключення до Інтернету. Strong – додаток-трекер для силового тренування

(гирьовий зал, бодібілдинг). Здійснює роль цифрового журналу тренувань. Зручний для досвідчених атлетів, які самі знають план вправ; інтуїтивне занесення сетів і вага, підтримка безлічі типів вправ. Ці інструменти мають простий і зрозумілий інтерфейс, який дозволяє детально фіксувати дані про вправи - кількість повторень, підходів, вагу, час відпочинку тощо. Вони зручні для щоденного користування та дають змогу зберігати інформацію у вигляді журналу. Проте на цьому їхній функціонал вичерпується: жодної глибокої аналітики, можливостей прогнозування або генерації візуальних звітів тут не передбачено. Дані залишаються пасивними, і користувач повинен самостійно проводити аналіз та робити висновки, що знижує ефективність використання такого інструменту для цілей довготривалого планування чи підвищення мотивації.

Nike Training Club - безкоштовний додаток від Nike з великим каталогом тренувань (йога, силові, кардіо, розтяжка тощо). Програми ведуть сертифіковані тренери і навіть відомі спортсмени. NTC є прикладом платформи, що робить акцент на зручність і візуальну привабливість. Безкоштовна величезна база відеотренувань, поділених за рівнем складності, тривалістю та метою (кардіо, сила, розтяжка тощо), робить цей ресурс дуже популярним серед користувачів, які віддають перевагу заняттям вдома. Але, попри свою привабливість, платформа не пропонує функцій ведення персональних планів тренувань, не дозволяє контролювати фізичні показники або адаптувати програму до індивідуального прогресу. Вона більше схожа на навчальний каталог, а не на інструмент управління фізичною формою. Немає системи зворотного зв’язку, фіксації результатів або глибокої персоналізації, що обмежує користувача у можливості системного підходу до тренувального процесу.

Окрему категорію представляє додаток Strava, яка здобула популярність серед шанувальників бігу, велоспорту та інших кардіоактивностей. Дозволяє відслідковувати маршрути, швидкість, дистанцію тощо за допомогою GPS і приєднуватися до спортивної спільноти. Вона пропонує зручний інтерфейс для

фіксації маршрутів, темпу, пульсу, набору висоти, а також дозволяє взаємодіяти з іншими учасниками через елементи соціальної мережі - лайки, коментарі, таблиці лідерів. Strava добре інтегрується з фітнес-браслетами та годинниками, що забезпечує точність даних. Проте платформа орієнтована на результат кожного окремого тренування, а не на комплексне планування або аналіз фізичних змін у динаміці. Вона не підходить для силових тренувань і не дозволяє будувати гнучкі, довгострокові програми з урахуванням змін у фізичному стані користувача. Бракує також можливостей прогнозування чи рекомендацій на основі зібраних даних.

Підсумовуючи, можна зробити висновок, що ринок фітнес-застосунків наразі представлений великою кількістю ресурсів, кожен з яких має свої сильні сторони, але при цьому жоден не вирішує проблему комплексно. Одні сервіси фокусуються на харчуванні, інші - на кардіо, ще інші - на силовому тренуванні чи просто пропонують відеоуроки без аналітики. Більшість із них не дозволяють вільно створювати індивідуальні тренувальні плани, бракує інструментів візуалізації прогресу, немає алгоритмів прогнозування або рекомендацій, що базуються на динаміці змін. Крім того, у багатьох випадках користувач змушений вручну переносити дані з одного ресурсу в інший або поєднувати кілька застосунків, щоб досягти повного охоплення своїх потреб.

На українському ринку фітнес-додатків також з’являються локальні рішення. Прикладом є BetterMe – популярна світова платформа, заснована українською командою, яка містить тренування, йогу, танці та програми харчування (доступна також українською). Під час війни розробники відкрили українцям безкоштовний доступ до мобільного застосунку BetterMe: Health Coaching. Є і чисто українські проєкти: WOWBODY (автори – відомі українські тренери) пропонує фітнес-програми для жінок і чоловіків (доступна українська/російська версія); колективні рішення на зразок Power Ups (кілька додатків з різними тренуваннями) надаються українськими розробниками (проте здебільшого російськомовні); BeStronger – серія домашніх програм з англійською, українською, польською та іншими мовами (для

українців діє знижка). Усі ці приклади свідчать про зростаючий інтерес українських розробників до фітнес-галузі. Однак на ринку ще бракує комплексних рішень з локалізацією та інтеграцією з офіційними сервісами. Висока інтернет-активність українців (80% щоденно онлайн) і урядова підтримка цифровізації (до 51% населення користується платформою «Дія») створюють перспективне середовище для розробників фітнес-додатків. Перспективи розвитку включають розширення підтримки української мови, створення локального контенту та інтеграцію з національними програмами здоров'я.

Отже, спостерігається чітка потреба у створенні цілісного та універсального Web-ресурсу, який би інтегрував функції фіксації, планування, аналізу та прогнозування фізичної активності, забезпечуючи при цьому зручний інтерфейс, гнучке налаштування та адаптацію до індивідуальних потреб користувача. Такий підхід не лише спростить процес ведення тренувань, а й зробить його більш ефективним, аналітично підкріпленим та орієнтованим на досягнення реальних результатів.

1.3 Вибір архітектури та технологічного стеку розробки Web-ресурсу

Розробка Web-додатку для планування фізичних тренувань та аналізу показників здоров'я користувача передбачає ретельний підхід до вибору технологій, що використовуються як у серверній, так і у клієнтській частинах системи. Для дипломного проєкту основними критеріями відбору стали простота впровадження, гнучкість архітектури, швидкість розгортання, підтримка стандартів безпеки, сумісність між компонентами та активна підтримка спільноти розробників. Зокрема, у виборі пріоритет надавався технологіям, які не вимагають складної конфігурації, але дають достатньо можливостей для реалізації функціоналу системи в повному обсязі [1-8].

Як основну платформу для серверної частини обрано Flask - мікрофреймворк на мові програмування Python, що дозволяє створювати легкі й водночас функціонально насичені Web-додатки. Його архітектура передбачає просту маршрутизацію, що дозволяє гнучко керувати запитами до сторінок. Однією з ключових особливостей Flask є його розширюваність за рахунок великої кількості додаткових бібліотек, включаючи підтримку шаблонізатора Jinja2. Це дозволяє будувати логічно структуровані HTML-сторінки з динамічним наповненням, які оновлюються відповідно до введених користувачем даних. Такий підхід дає змогу забезпечити повноцінну взаємодію клієнта з сервером у реальному часі [9-12].

Для зберігання даних у системі використовується SQLite - вбудована реляційна база даних, яка є легкою у використанні та не потребує окремого серверного процесу. У поєднанні з бібліотекою SQLAlchemy, яка реалізує об’єктно-реляційне відображення (ORM), забезпечується простий і безпечний доступ до бази даних через Python-класи. Це дозволяє зосередитися на логіці роботи з даними, не витрачаючи час на написання SQL-запитів вручну. Використання ORM також підвищує надійність коду та дозволяє з легкістю реалізувати міграції, оновлення структури таблиць та обробку помилок [13].

Візуальна частина додатку реалізована через шаблони Jinja2 у зв’язці з HTML, CSS та JavaScript. Це забезпечує гнучкість у створенні адаптивного інтерфейсу, здатного змінюватися відповідно до розміру екрану та типу пристрою. Ресурс розроблений з урахуванням базових принципів юзабіліті: простота навігації, інтуїтивна структура, візуальні підказки. Завдяки використанню шаблонів можна легко оновлювати контент у реальному часі - наприклад, відображати розклад тренувань або змінювати інформацію про прогрес користувача.

Окрему увагу було приділено реалізації безпеки, адже система працює з персональними даними користувачів. Для аутентифікації реалізовано реєстрацію та вхід із хешуванням паролів за допомогою функцій із бібліотеки werkzeug.security, що використовує алгоритм pbkdf2:sha256. Такий підхід гарантує, що паролі

зберігаються в зашифрованому вигляді та не можуть бути отримані навіть у разі компрометації бази даних. Крім того, сесії користувачів контролюються за допомогою вбудованого механізму Flask, який дозволяє обмежити доступ до певного функціоналу залежно від ролі користувача (наприклад, студент або викладач) [14-16].

В архітектурному плані система спроєктована таким чином, щоб окремі її компоненти були модульними - маршрути обробки поділені за функціональним принципом: робота з розкладом, новинами, автентифікацією тощо. Це дозволяє підтримувати код чистим, полегшує налагодження та розширення. У майбутньому це забезпечить легке додавання нових модулів без необхідності переписування вже існуючих частин. Така структура є однією з ключових переваг Flask, оскільки вона дозволяє підтримувати проект у довгостроковій перспективі.

Для забезпечення збереження й роботи із завантажуваними файлами (наприклад, зображеннями або фотографіями тренувального прогресу) передбачено локальне збереження файлів у підкаталозі проєкту. Завдяки модулю werkzeug.utils реалізовано захищене завантаження з автоматичною обробкою імен файлів, що мінімізує ризик конфліктів та полегшує подальшу обробку контенту на стороні серверу.

У процесі розробки також розглядались альтернативні технології. Зокрема, на етапі аналізу технологічного стека були розглянуті такі бекенд-платформи, як Django - повнофункціональний фреймворк на Python, що містить у собі безліч готових інструментів «із коробки» та забезпечує високий рівень безпеки. Проте, у контексті розробки легкого, цільового застосунку, було вирішено надати перевагу Flask, який не нав’язує жорсткої структури проєкту і дозволяє гнучко реалізувати необхідний функціонал без зайвого перевантаження системи. Також розглядалися Node.js/Express - ефективна платформа для JavaScript-орієнтованої розробки, яка надає високу продуктивність при обробці великої кількості одночасних запитів.

Однак для цілей аналізу фізичних даних та використання можливостей Python-бібліотек було доцільніше використати саме стек Flask + SQLAlchemy [17-19].

Загалом, комбінація Flask + SQLAlchemy + SQLite + Jinja2 є оптимальним вибором для дипломного Web-застосунку. Вона дозволяє досягти високого рівня продуктивності, спрощує розробку та обслуговування, а також забезпечує надійну основу для аналізу динаміки фізичних показників і подальшого розвитку функціоналу системи [20-22].

Для подальшого покращення продуктивності та масштабованості, при розробці цього Web-додатку було прийнято рішення використовувати додаткові інструменти для оптимізації роботи з базою даних, кешуванням та обробкою запитів. Одним з таких інструментів є Redis - система кешування, яка дозволяє значно зменшити час відповіді сервера за рахунок зберігання часто використовуваних даних у пам'яті. Використання Redis в комбінації з Flask забезпечує швидкий доступ до найважливіших даних, таких як профілі користувачів або результати тренувань, що значно покращує взаємодію з системою.

Крім того, для інтеграції сторонніх сервісів, таких як платформи для трекінгу фізичної активності (наприклад, Google Fit або Fitbit), було передбачено створення REST API. API дозволяє взаємодіяти з іншими сервісами для отримання даних про фізичну активність користувачів, що дає змогу автоматично оновлювати інформацію про їхні показники здоров'я. Використання стандарту REST забезпечує високу сумісність з різними платформами, що дозволяє з часом додавати нові інтеграції без суттєвих змін у системі [23-27].

Важливим аспектом є також підтримка багатомовності додатку. Для цього використовуються відповідні інтернаціоналізаційні бібліотеки, зокрема Flask-Babel, що дозволяє легко локалізувати інтерфейс для різних мов. Це особливо важливо для забезпечення доступності додатку для користувачів з різних країн. У процесі розробки була інтегрована підтримка кількох мов, включаючи українську,

англійську та російську мови, що дозволяє значно розширити аудиторію користувачів.

Іншим важливим етапом є інтеграція системи збору аналітики щодо використання додатку. Для цього передбачено використання таких інструментів, як Google Analytics або Mixpanel, що дозволяє відстежувати поведінку користувачів та отримувати детальні звіти про активність у додатку. Це допомагає виявляти популярні функції та потреби користувачів, що дозволяє своєчасно коригувати стратегію розвитку додатку [28-30].

Крім того, в рамках забезпечення високої доступності і безперервної роботи системи, було вирішено використовувати Docker для контейнеризації додатку. Docker дозволяє спростити процес розгортання додатку на різних середовищах, зокрема на хмарних платформах, таких як AWS або Google Cloud. Це дозволяє забезпечити стабільну роботу системи, незалежно від умов хостингу, а також значно полегшує процес масштабування при необхідності.

Урахування всіх цих аспектів дозволяє створити не лише функціонально багатий, але й масштабований, надійний Web-додаток, здатний підтримувати різноманітні вимоги користувачів та гарантувати високу продуктивність при роботі з великими обсягами даних.

Висновки до розділу 1

Проведено всебічний аналіз предметної області, вивчено існуючі рішення на ринку Web-ресурсів для планування тренувань, а також обґрунтовано вибір архітектури та технологічного стеку для реалізації дипломного проєкту. Встановлено, що сучасні тенденції цифровізації, активне зростання зацікавленості суспільства у веденні здорового способу життя та поширення носимих пристроїв створюють необхідне підґрунтя для впровадження спеціалізованих програмних рішень у сфері фізичної активності.

Проведений аналіз актуальності розробки засвідчив наявність стійкого попиту на інтегровані платформи, що дозволяють не лише фіксувати фізичну активність, а й виконувати аналітику, візуалізацію результатів, прогнозування прогресу користувача. Статистичні та соціальні дані підтвердили глобальну проблему низького рівня рухової активності, що поглиблює потребу в системах, які можуть підвищувати мотивацію та контроль за тренуваннями. У контексті українського цифрового середовища та високого рівня проникнення інтернету створення подібного Web-ресурсу має додаткову соціальну та освітню цінність.

Під час вивчення існуючих Web-ресурсів встановлено, що жоден із розглянутих застосунків - незалежно від його популярності чи поширеності - не забезпечує комплексного підходу до роботи з фізичними даними користувача. Існуючі рішення є вузькоспеціалізованими, обмеженими функціонально або структурно. Основними недоліками виявилися відсутність персоналізації, обмежені можливості створення гнучких тренувальних програм, нестача систем аналітики або прогнозування, а також труднощі з інтеграцією різних аспектів здоров’я (тренування, харчування, відновлення) в єдиний інтерфейс. Таким чином, обґрунтовано доцільність створення нового комплексного Web-ресурсу, який би забезпечував високий рівень взаємодії, персоналізації та аналітики.

На основі сформульованих вимог до функціоналу системи, було обґрунтовано вибір архітектурних та технологічних рішень. В якості серверного середовища обрано мікрофреймворк Flask, який забезпечує гнучкість, масштабованість і легкість інтеграції з іншими модулями. Для роботи з базою даних - SQLite у зв’язці з ORM-бібліотекою SQLAlchemy. Візуальна частина побудована на HTML-шаблонах з використанням Jinja2, що забезпечує адаптивність та інтерактивність інтерфейсу. Крім того, приділено увагу безпеці, розширюваності та потенціалу масштабування завдяки використанню Redis, контейнеризації Docker, REST API, Flask-Babel і сторонніх аналітичних сервісів.

Таким чином, у першому розділі закладено науково-практичне підґрунтя для подальшої реалізації функціонального, адаптивного та масштабованого Web-застосунку, що здатен задовольнити сучасні вимоги користувачів у галузі цифрового фітнесу та моніторингу фізичної активності.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА WEB-РЕСУРСУ ДЛЯ ПЛАНУВАННЯ ТРЕНУВАНЬ

2.1 Проєктування інформаційної моделі Web-ресурсу

Проєктування інформаційної моделі є базовим етапом у створенні Web-ресурсу для планування тренувань і моніторингу фізичних показників. У моделі чітко окреслено ключові сутності: користувач, тренер, план тренувань, окремі тренування, фізіологічні показники. Користувач взаємодіє із системою через персональний профіль, де зберігаються особисті дані та історія активностей. Тренер має доступ до клієнтів, їхніх планів і прогресу, а план тренувань включає вправи з деталями навантаження. Передбачено також збереження показників здоров'я з прив'язкою до часу, що дає змогу будувати аналітику та візуалізувати динаміку.

Модель реалізована в SQLite через ORM SQLAlchemy, що забезпечує ефективну роботу з даними та збереження логічних зв'язків (один-до-багатьох між користувачами, планами і тренуваннями). У системі також реалізована логіка ролей: користувач, тренер, адміністратор.

Архітектура побудована за клієнт-серверною моделлю з використанням Flask. Серверна частина обробляє логіку, маршрути, автентифікацію, роботу з базою. Клієнтська частина побудована на HTML-шаблонах із Jinja2, що дозволяє динамічно відображати контент. Система розділена на три шари: презентаційний, логічний та даних. Для збереження безпеки реалізовано хешування паролів та механізм сесій.

Завдяки модульній структурі система легко піддається розширенню: можлива інтеграція API, додавання нових функцій або перенесення частини логіки в окремі сервіси. Така архітектура забезпечує масштабованість, стабільність і гнучкість системи, що робить її придатною як для навчальних, так і для реальних проєктів.

2.2 Проєктування бази даних Web-ресурсу

У процесі створення Web-ресурсу для планування тренувань було спроєктовано логічну та фізичну модель бази даних, яка забезпечує зберігання, цілісність і доступність інформації. Основними сутностями є користувач, тренер, план тренувань, окремі тренування та фізіологічні показники. Усі сутності мають атрибути, що описують ключові характеристики: для користувача - ім’я, пошта, фізичні дані; для тренувань - дата, вправи, навантаження тощо. Система дозволяє зберігати історію змін, що дає змогу аналізувати прогрес.

Уся логіка реалізована у вигляді таблиць, між якими встановлено чіткі зв’язки через зовнішні ключі. Один користувач може мати багато планів, кожен план - багато тренувань, які можуть містити кілька вправ. Окремо реалізовано зберігання показників здоров’я, що пов’язуються з користувачем або конкретним тренуванням. Такий підхід дозволяє будувати графіки, діаграми та звіти для візуалізації динаміки змін.

Центральну роль у структурі відіграє користувач, який створює плани, реєструє вправи та слідкує за прогресом. Тренер, у свою чергу, може керувати кількома користувачами, переглядати їхню активність і надавати рекомендації. Це дозволяє реалізувати як індивідуальне використання платформи, так і коучингову взаємодію.

База побудована з урахуванням принципів нормалізації, що дозволяє уникнути дублювання інформації та зберігати її у структурованому вигляді. Така модель забезпечує не лише зручне збереження даних, а й підтримку функцій Web-додатку: створення планів, реєстрація показників, аналітика, зв’язок між користувачем і тренером. База стає не просто сховищем, а активним компонентом аналітичної системи фізичного розвитку. В основі структури лежить ER-діаграма (рис. 2.1), яка демонструє логічні взаємозв’язки між таблицями. Кожен елемент у ній репрезентує окрему таблицю, а лінії з відповідними кардинальностями - типи

зв’язків. Структура реалізована таким чином, щоб забезпечити цілісність даних і мінімізувати їх дублювання.

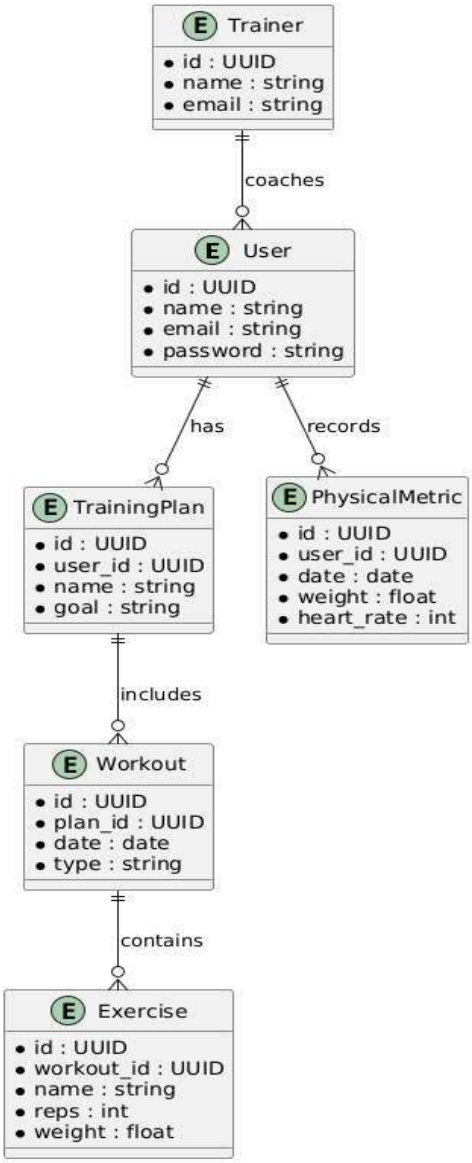

Рис. 2.1. ER-діаграма інформаційної моделі Web-ресурсу

У структурі бази даних система оперує ключовими сутностями: користувач, план тренувань, тренування, фізіологічні показники та тренер. Кожен користувач може мати декілька тренувальних планів, що відображає зв’язок типу «один до багатьох». План, у свою чергу, складається з окремих тренувань, які мають дату,

опис, тривалість та інші характеристики. До кожного тренування прикріплюються фізіологічні показники, що дозволяє відстежувати динаміку змін фізичного стану - наприклад, частоту серцевих скорочень, тривалість чи навантаження. Зв’язки реалізовано за допомогою зовнішніх ключів, що забезпечує логічну цілісність даних і зручність у запитах. Загальна структура дозволяє ефективно масштабувати систему, зберігаючи її гнучкість і адаптивність до змін функціоналу.

Важливим етапом стало визначення функціональності, яку система має надавати різним типам користувачів. Для цього було побудовано діаграму варіантів використання.

Use Case-діаграма ілюструє функціональні можливості системи, показуючи, які дії доступні для кожної ролі. Вона допомагає структуровано представити логіку взаємодії між користувачами та інтерфейсом програми, забезпечуючи прозоре розмежування прав доступу. Так, звичайний користувач має змогу зареєструватися, створити власний план, додати особисті дані, тоді як тренер отримує інструменти для аналізу прогресу, а адміністратор - можливості управління всією системою.

У рамках розробки Web-додатку було визначено основні ролі користувачів та сценарії їхньої взаємодії з системою, що відображено у відповідній діаграмі варіантів використання. Найбільш базовим учасником є звичайний користувач, який має змогу реєструватися в системі, входити до особистого кабінету, створювати тренувальні плани, додавати фізіологічні показники після кожного заняття, а також переглядати особисту статистику і прогрес у вигляді візуалізованих звітів. Розроблена діаграма варіантів використання Web-ресурсу представлена на рис. 2.2.

Роль тренера передбачає ширший доступ до функціоналу: він може переглядати профілі своїх підопічних, аналізувати їхні результати, створювати або коригувати тренувальні програми для конкретного користувача та залишати рекомендації. Така модель реалізує персоніфікований підхід до тренувань і забезпечує зворотний зв’язок, необхідний для досягнення кращих результатів.

Рис. 2.2. Діаграма варіантів використання Web-ресурсу

Окремо функціонує адміністратор, який відповідає за підтримку системи в цілому. Він має змогу керувати обліковими записами, модерувати контент, налаштовувати параметри безпеки та структуру базових довідників (типів вправ, категорій показників тощо). Усі варіанти використання взаємодіють із центральною логікою програми через чітко визначені сценарії, що дозволяє підтримувати стабільність роботи, логічну розмежованість функцій та безпеку даних.

Для побудови внутрішньої логіки програми та взаємодії об’єктів застосовано діаграму класів. Вона дозволяє наочно відобразити атрибути класів, а також методи, які можуть бути реалізовані в рамках системи. Діаграма класів описує структуру об’єктів та способи їх взаємодії. Кожен клас представляє собою логічну одиницю, яка використовується в коді - користувач, тренування, план, показники. Зв’язки між класами забезпечують узгодженість і повторне використання логіки. Розроблена діаграма класів Web-ресурсу представлена на рис. 2.3.

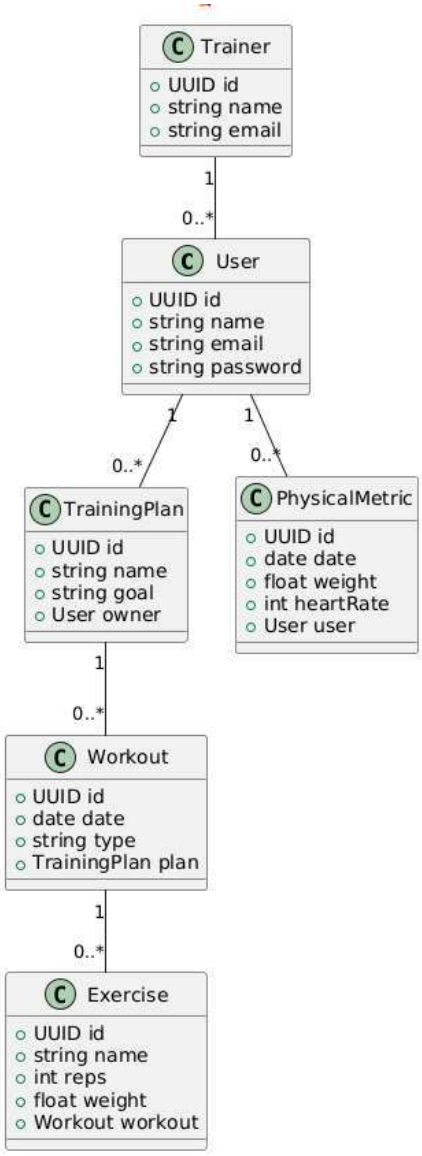

Рис. 2.3. Діаграма класів Web-ресурсу

Система побудована на основі чітко визначених класів, що відображають логіку обробки та зберігання даних. Основним є клас User, який містить персональні дані користувача, його облікові параметри та взаємозв'язки з іншими об'єктами. Зокрема, кожен користувач асоціюється з класом WorkoutPlan, що відповідає за формування індивідуальних тренувальних програм.

Кожна програма включає набір елементів класу Workout, які описують окремі тренування з відповідними характеристиками. Усі тренування можуть мати зв'язок із класом PhysicalMetric, де зберігаються дані про стан користувача: пульс, вага, кількість повторень тощо. Додатково клас Trainer дозволяє реалізувати зв'язок між користувачем і персональним тренером, що супроводжує процес тренувань і може надавати рекомендації. Така структура сприяє впорядкованому доступу до даних та забезпечує цілісність моделі.

На завершення етапу моделювання було створено діаграму активності, яка демонструє послідовність дій користувача під час створення плану тренувань. Цей тип діаграми є важливим для розуміння динаміки системи, особливо в контексті взаємодії з інтерфейсом і базою даних.

Діаграма активності демонструє, як користувач авторизується, вибирає опцію створення нового плану, заповнює деталі, додає вправи й підтверджує збереження. Після завершення система фіксує ці дії у базі та надає можливість переглядати чи редагувати створений план.

У структурі роботи Web-застосунку важливою частиною є моделювання послідовності дій користувача при взаємодії з системою. Діаграма активності демонструє логіку процесу створення персонального плану тренувань. Розроблена діаграма активності представлена на рис. 2.4.

На початковому етапі користувач проходить автентифікацію, після чого потрапляє до особистого кабінету, де може ініціювати створення нового плану. Процес включає послідовне введення ключових параметрів: мету тренувань, бажану тривалість, частоту занять, інтенсивність, а також додаткові обмеження або

побажання. Після формування структури плану користувач додає конкретні вправи до кожного дня, обираючи їх зі списку або вводячи власні.

Рис. 2.4. Діаграма активності

Після завершення заповнення всіх даних користувач підтверджує створення плану. Система зберігає інформацію у базі даних, створює пов’язану структуру сутностей та надає доступ до перегляду, редагування або аналітики тренувального процесу. У разі необхідності користувач має змогу вносити корективи у план, не порушуючи його загальної структури. Такий алгоритм забезпечує логічну, послідовну і зручну для користувача взаємодію з системою.

Таким чином, проєктування бази даних охоплює як статичні, так і динамічні аспекти побудови системи. Усі діаграми разом забезпечують повне уявлення про архітектуру програми, зв’язки між об'єктами та логіку функціонування, що в підсумку дозволяє створити масштабований і зрозумілий Web-додаток для аналізу та планування фізичних навантажень.

2.3 Реалізація серверної та клієнтської частини Web-ресурсу

Процес реалізації Web-додатку відбувався з урахуванням сучасних принципів розробки - модульності, гнучкості, простоти розширення та підтримки. Архітектура системи побудована на основі мікрофреймворку Flask, який забезпечує легкий запуск, високу читабельність коду і чітке розмежування функціональних зон. Кожна логічна частина додатку (автентифікація, робота з тренуваннями, візуалізація, календар тощо) була винесена в окремий модуль, що дало змогу не лише впорядкувати структуру, а й уникнути типових помилок на кшталт циклічного імпорту. Основний серверний код згруповано в кілька основних файлів.

app.py – головний файл ініціалізації додатку, де відбувається підключення конфігурації, бази даних, Blueprint'ів, а також запуск програми. Файл app.py є ключовим компонентом, з якого розпочинається робота всього Web-ресурсу. Він виконує роль основного модуля ініціалізації програми, в якому налаштовуються базові параметри середовища, підключається база даних, створюється додаток Flask і реєструються необхідні розширення. Саме в цьому файлі відбувається оголошення

структури застосунку, зокрема конфігурації з'єднання з базою даних, секретного ключа для сесій, а також налаштування логіки взаємодії з іншими частинами програми. Код з файлу app.py наведений на рис. 2.5.

app.py

```python
from flask import Flask
from extensions import db
from auth import auth_bp
from routes import main_bp

def create_app():
    app = Flask(__name__)
    app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///fitness_app.db'
    app.config['SECRET_KEY'] = 'your_secret_key'
    db.init_app(app)

    app.register_blueprint(auth_bp)
    app.register_blueprint(main_bp)

    with app.app_context():
        db.create_all()

    return app

if __name__ == '__main__':
    app = create_app()
    app.run(debug=True)
```

Рис. 2.5. Код з головного файлу Web-додатку (app.py)

Одразу після створення об'єкта застосунку, Flask конфігурується для роботи з локальною базою даних SQLite, що зберігається у вигляді файлу. Такий підхід є зручним для дипломного проєкту, оскільки не потребує окремого серверного середовища і дозволяє запускати додаток локально з мінімальними ресурсами. Бібліотека SQLAlchemy забезпечує об’єктно-реляційне відображення, що дозволяє працювати з таблицями бази даних через Python-класи, не використовуючи SQL-запити безпосередньо. Це суттєво спрощує взаємодію з даними і підвищує безпеку системи.

У цьому файлі також визначаються основні моделі - об'єкти, які представляють користувачів, плани тренувань та окремі вправи. Структура кожної моделі чітко описує, які саме атрибути зберігаються і як ці об'єкти взаємодіють між собою. Наприклад, кожен користувач може мати кілька планів тренувань, а кожен план, у свою чергу, включає вправи з вказаними параметрами: датою, кількістю повторень, підходів, вагою тощо. Це дозволяє відтворити логіку фітнес-системи у вигляді структурованої бази, де зв’язки між сутностями є чітко окресленими.

Далі йде опис основних маршрутів, що визначають поведінку системи у відповідь на дії користувача. Наприклад, при відкритті головної сторінки система перевіряє, чи є користувач авторизованим, і в залежності від цього відображає або дашборд із планами, або привітальний екран. Реалізовані функції реєстрації, входу і виходу з облікового запису забезпечують базову автентифікацію користувача. Також передбачено створення нових планів і можливість додавання вправ до існуючих тренувальних програм. Дані, що вводяться, обробляються і зберігаються у базу через ORM, а потім відображаються у відповідних шаблонах HTML.

На завершальному етапі роботи програма перевіряє, чи існує база даних. Якщо ні - вона створюється автоматично при першому запуску, що забезпечує зручність у розгортанні системи. Запуск Flask-серверу у режимі налагодження дозволяє виявляти помилки у процесі розробки та оперативно реагувати на них.

Загалом, app.py є об’єднуючою ланкою всього Web-застосунку. Він інтегрує всі функціональні модулі, визначає загальну архітектуру, координує маршрутизацію, обробку даних і запуск додатку. Його структура є зразком мінімалістичного, але водночас повноцінного підходу до побудови серверної логіки в межах невеликого, але функціонального Web-ресурсу.

routes.py - основна логіка маршрутів користувача: авторизація, реєстрація, створення та перегляд планів, робота з калькулятором, сторінка порад і календар. Цей файл реалізує основну логіку взаємодії користувача з Web-застосунком. У ньому описано маршрути для головної сторінки, створення та перегляду планів

тренувань, роботи з календарем, калькулятором калорій, а також перегляду корисних порад. Кожен маршрут - це окрема функція, яка обробляє запити (GET або POST), виконує дії з базою даних через SQLAlchemy та повертає відповідний HTML-шаблон із динамічним вмістом. Наприклад, користувач після входу бачить список своїх планів тренувань - ця функціональність реалізується маршрутом `/dashboard`. Коли він створює новий план - обробляється маршрут `/add_plan`, а перегляд і додавання вправ до плану реалізовано через `/plan/<plan_id>`. Окремі маршрути відображають сторінки з календарем, калькулятором та порадами - усі ці компоненти пов’язані з відповідними шаблонами і дозволяють розширити функціональність системи без перевантаження інтерфейсу. Фрагмент коду з файлу routes.py представлений на рис. 2.6.

```
6   main_bp = Blueprint('main', __name__)
7
8   @main_bp.route('/')
9   def index():
10      if 'user_id' in session:
11          return redirect(url_for('main.dashboard'))
12      return render_template('index.html')
13
14  @main_bp.route('/dashboard')
15  def dashboard():
16      if 'user_id' not in session:
17          return redirect(url_for('auth.login'))
18      plans = Plan.query.filter_by(user_id=session['user_id']).all()
19      return render_template('dashboard.html', plans=plans)
20
21  @main_bp.route('/add_plan', methods=['GET', 'POST'])
22  def add_plan():
23      if 'user_id' not in session:
24          return redirect(url_for('auth.login'))
25      if request.method == 'POST':
26          title = request.form['title']
27          plan = Plan(user_id=session['user_id'], title=title)
28          db.session.add(plan)
29          db.session.commit()
30          return redirect(url_for('main.dashboard'))
31      return render_template('add_plan.html')
32
33  @main_bp.route('/plan/<int:plan_id>', methods=['GET', 'POST'])
```

Рис. 2.6. Фрагмент коду з файлу routes.py

models.py – файл, що відповідає за визначення ORM-моделей для взаємодії з базою даних. В ньому описані сутності User, Plan, Workout, їх зв’язки та атрибути. Файл `models.py` містить опис усіх основних сутностей, які формують основу бази даних Web-застосунку. Його головна мета - визначити структуру таблиць, поля (атрибути) та зв’язки між ними, використовуючи ORM SQLAlchemy. Фрагмент коду з файлу models.py представлений на рис. 2.7.

```
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    password = db.Column(db.String(200), nullable=False)
    role = db.Column(db.String(20), nullable=False, default='user')
    plans = db.relationship('Plan', backref='user', lazy=True)
    notes = db.relationship('Note', backref='user', lazy=True)  # Додаємо зв’язок

class Plan(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    title = db.Column(db.String(120), nullable=False)
    created_at = db.Column(db.DateTime, default=datetime.utcnow)
    workouts = db.relationship('Workout', backref='plan', lazy=True)

class Workout(db.Model):
    __tablename__ = 'workout'
    id = db.Column(db.Integer, primary_key=True)
    plan_id = db.Column(db.Integer, db.ForeignKey('plan.id'))
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    date = db.Column(db.Date, nullable=False)
    exercise = db.Column(db.String(120), nullable=False)
    sets = db.Column(db.Integer)
    reps = db.Column(db.Integer)
    weight = db.Column(db.Float)
```

Рис. 2.7. Фрагмент коду з файлу models.py

У цьому файлі створено кілька моделей, які відповідають ключовим об'єктам системи. Клас `User` зберігає інформацію про зареєстрованих користувачів: логін, хешований пароль і роль (звичайний користувач, тренер або адміністратор). Модель `Plan` представляє тренувальний план, прив’язаний до певного

користувача. У свою чергу, модель `Workout` описує конкретні тренування в межах плану - з датою, назвою вправи, кількістю повторень, підходів і вагою. Також може бути реалізована додаткова модель, наприклад `HealthMetric`, для збереження фізіологічних показників користувача (пульс, вага тощо).

Всі ці моделі пов'язані між собою за допомогою зовнішніх ключів, що забезпечує узгодженість і логічну цілісність даних. Наприклад, кожне тренування (`Workout`) посилається на план (`Plan`), а кожен план - на користувача (`User`). Таким чином, `models.py` виконує критично важливу роль у формуванні та обслуговуванні бази даних, забезпечуючи надійний каркас для всієї системи.

extensions.py – конфігурація та ініціалізація розширень, таких як SQLAlchemy, для спрощення імпорту. Файл extensions.py у Flask-проєкті слугує місцем для централізованого створення та ініціалізації глобальних об'єктів, які потім підключаються до основного застосунку. Це дозволяє уникнути циклічного імпорту та підвищити модульність проєкту. У такому вигляді db є лише об'єктом, що ініціалізується пізніше - вже в app.py (або іншому файлі конфігурації), де Flask-додаток створюється і конфігурується. Завдяки цьому інші частини застосунку (наприклад, моделі у models.py) можуть безпечно імпортувати db з extensions.py і використовувати його для оголошення таблиць.

Цей підхід є стандартом у Flask-проєктах, особливо при розробці застосунків із чіткою структурою: він дозволяє зручно розділити ініціалізацію розширень від основної конфігурації додатку. У майбутньому до extensions.py можуть додаватися й інші розширення - наприклад, Migrate, LoginManager, Babel, Mail тощо.

auth.py – файл, в якому реалізована базова система аутентифікації користувача. Файл auth.py реалізує базову, але надійну систему автентифікації, яка дозволяє захищати особисті дані користувачів і створювати персоналізовані сторінки, доступні лише після входу. Це критично важливо для Web-ресурсу, де фіксується прогрес тренувань, фізіологічні показники та інша приватна інформація. Фрагмент коду з файлу auth.py представлений на рис. 2.8.

```python
@auth_bp.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        password = generate_password_hash(request.form['password'])
        if User.query.filter_by(username=username).first():
            flash('Користувач з таким іменем вже існує')
            return redirect(url_for('auth.register'))
        user = User(username=username, password=password)
        db.session.add(user)
        db.session.commit()
        return redirect(url_for('auth.login'))
    return render_template('register.html')

@auth_bp.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        user = User.query.filter_by(username=username).first()
        if user and check_password_hash(user.password, password):
            session['user_id'] = user.id
            session['username'] = user.username
            session['role'] = user.role
            return redirect(url_for('main.dashboard'))
        flash('Невірний логін або пароль')
    return render_template('login.html')

@auth_bp.route('/logout')
def logout():
    session.clear()
    return redirect(url_for('auth.login'))

```

Рис. 2.8. Фрагмент коду з файлу auth.py

Висновки до розділу 2

На основі аналізу сучасних технологій обґрунтовано вибір архітектурних рішень, інструментів та середовища розробки. Як бекенд-фреймворк було обрано Flask - легкий, гнучкий інструмент із широкими можливостями розширення, що добре підходить для швидкої розробки Web-застосунків. Для збереження та обробки даних використано SQLite у поєднанні з ORM-бібліотекою SQLAlchemy, що забезпечило зручну та безпечну роботу з реляційною моделлю.

Окрема увага приділена реалізації інтерфейсу за допомогою HTML-шаблонів із використанням Jinja2, що забезпечує гнучке відображення динамічних даних. Було реалізовано базові механізми безпеки, включно з хешуванням паролів, управлінням сесіями та обмеженням доступу відповідно до ролей користувачів. Розробка бази даних була здійснена із дотриманням принципів нормалізації та цілісності - сформовано логічну модель із ключовими сутностями, такими як користувачі, тренування, фізіологічні показники, плани занять і ролі.

Для наочного представлення логіки функціонування системи створено низку UML-діаграм: ER-діаграма, яка ілюструє структуру бази даних та зв’язки між таблицями; діаграма варіантів використання, що описує ролі користувачів і доступні їм функції; діаграма класів, яка репрезентує об’єктно-орієнтовану структуру проєкту; а також діаграма активності, що демонструє типовий сценарій роботи користувача із системою.

РОЗДІЛ 3
ФУНКЦІОНАЛ ТА ЕЛЕМЕНТИ ІНТЕРФЕЙСУ СТВОРЕНОГО WEB-РЕСУРСУ

3.1 Огляд реалізованого функціоналу Web-ресурсу

Розроблений Web-ресурс призначений для планування індивідуальних тренувань, ведення обліку фізіологічних показників та аналізу динаміки особистих результатів. Система реалізована за моделлю клієнт–сервер із зручною модульною структурою, що охоплює ключові аспекти взаємодії користувача з ресурсом: реєстрацію та авторизацію, створення планів тренувань, заповнення та перегляд тренувальних сесій, збереження показників і доступ до додаткового функціоналу, зокрема калькулятора калорій і календаря активностей.

При першому заході на сайт перед нами постає головна сторінка програми. На головній сторінці можна авторизуватися в системі тим чи іншим способом за допомогою реєстрації або входу до вже існуючого облікового запису. Також на головній сторінці реалізовано функцію розрахунку калорій за допомогою спеціального калькулятора без реєстрації облікового запису. Таким чином, якщо людині необхідний лише калькулятор, то їй не обов'язково реєструвати обліковий запис і ставати користувачем системи. На рис. 3.1 представлена головна сторінка розробленого Web-ресурсу.

Користувач може зареєструвати обліковий запис і здійснювати вхід у систему. Сесійна авторизація дозволяє ідентифікувати користувача, надаючи доступ лише до його особистих даних і планів. З точки зору користувача передбачені сторінки зі формами введення логіна та пароля при реєстрації та вході. Паролі зберігаються у базі даних не у відкритому вигляді – перед записом вони хешуються для безпеки. Форма реєстрації нового користувача розробленого Web-ресурсу представлена на рис. 3.2.

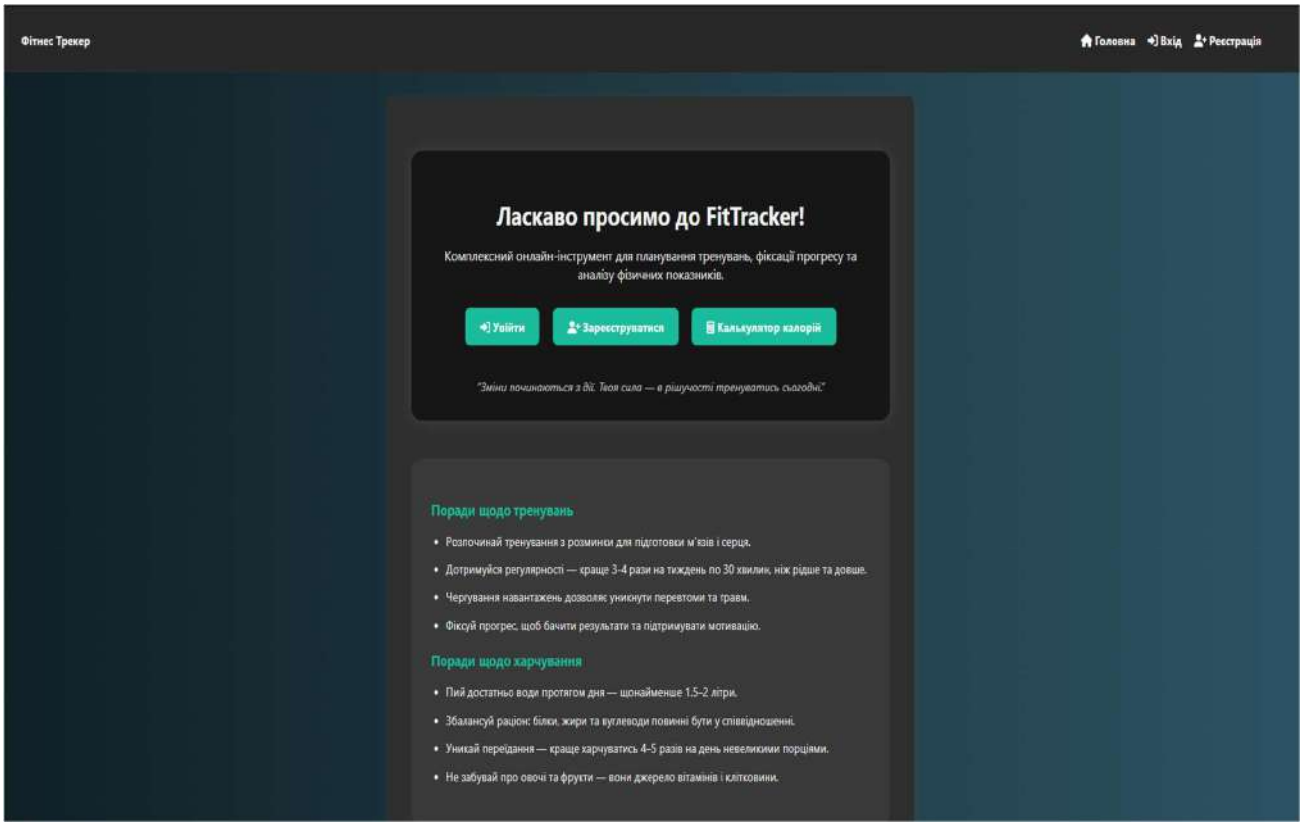

Рис. 3.1. Головна сторінка розробленого Web-ресурсу

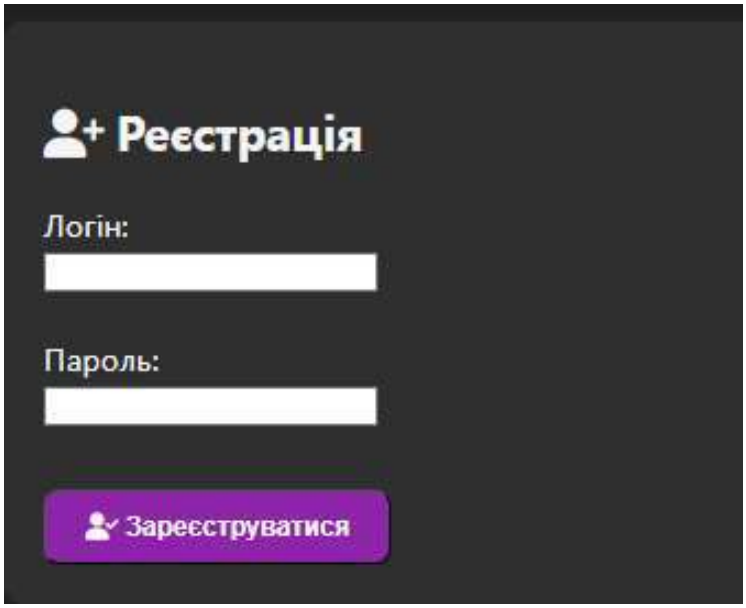

Рис. 3.2. Форма реєстрації нового користувача розробленого Web-ресурсу

Після входу користувач потрапляє на інформаційну панель, яка відображає список створених тренувальних планів. Кожен план може містити набір тренувань із зазначенням дати, типу вправ, кількості повторень, підходів і ваги. Цей функціонал реалізовано у вигляді форм, що дозволяють зручно додавати або редагувати дані безпосередньо через інтерфейс. На рис. 3.3 представлена форма параметрів тренування створеного Web-ресурсу.

Рис 3.3. Форма параметрів тренування створеного Web-ресурсу

Після заповнення форми дані відправляються на сервер через запит до відповідного маршруту. Контролер отримує ці дані, створює новий запис у базі даних (наприклад, у моделі TrainingPlan, пов’язаній з поточним користувачем) та повертає підтвердження успіху. Усі додані вправи зберігаються у таблицях БД і

згодом можуть відображатися у вигляді списку тренувань або інтегруватися до календаря.

Крім основної функціональності, у Web-додаток інтегровано калькулятор калорій, який дає змогу користувачу швидко оцінити добову потребу в енергії на основі базових параметрів, як-от вага, зріст, вік і рівень активності. Інтерфейс цього калькулятора представлений на рис. 3.4.

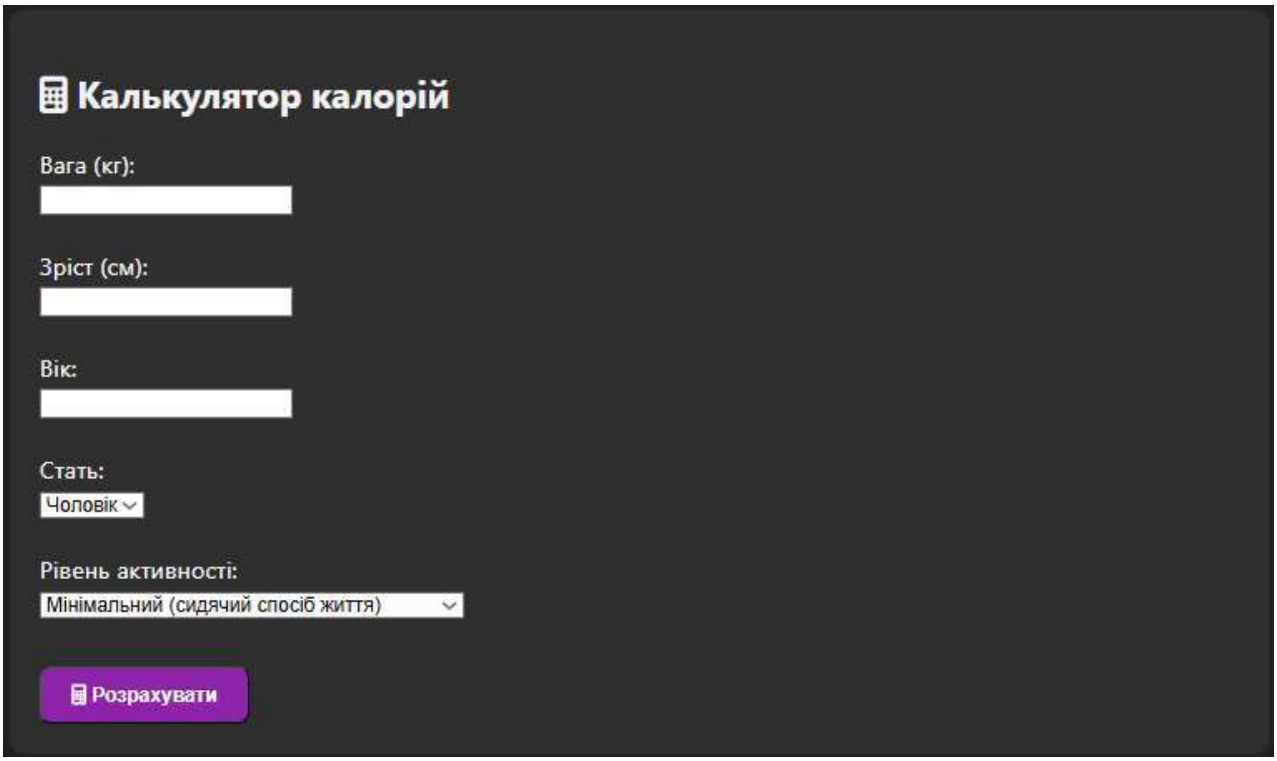

Рис. 3.4. Інтерфейс калькулятору калорій створеного Web-ресурсу

Калькулятор калорій надає можливість оцінити добову потребу в калоріях на основі персональних даних. Користувач вводить параметри (вага, зріст, вік, рівень активності) у спеціальну форму, після чого дані надсилаються на сервер. Контролер обробляє ці дані, виконує розрахунок (наприклад, за формулою Харріса–Бенедикта або іншою) та повертає результат у шаблон. Web-сторінка відображає рекомендовану норму калорій, яку користувач може зберегти у своєму профілі для подальшого аналізу.

Також для проекту був реалізований календар тренувань. Цей календар є окремим функціональним модулем, який виконує роль візуального інструменту для зручного планування, перегляду й орієнтування у тренувальному процесі. Його основна задача - відображати усі тренування, створені користувачем, у формі інтерактивного календаря, що дозволяє швидко оцінити завантаженість на тиждень чи місяць, а також відкривати або додавати нові події безпосередньо з інтерфейсу. Календар тренувань представлений на рис. 3.5.

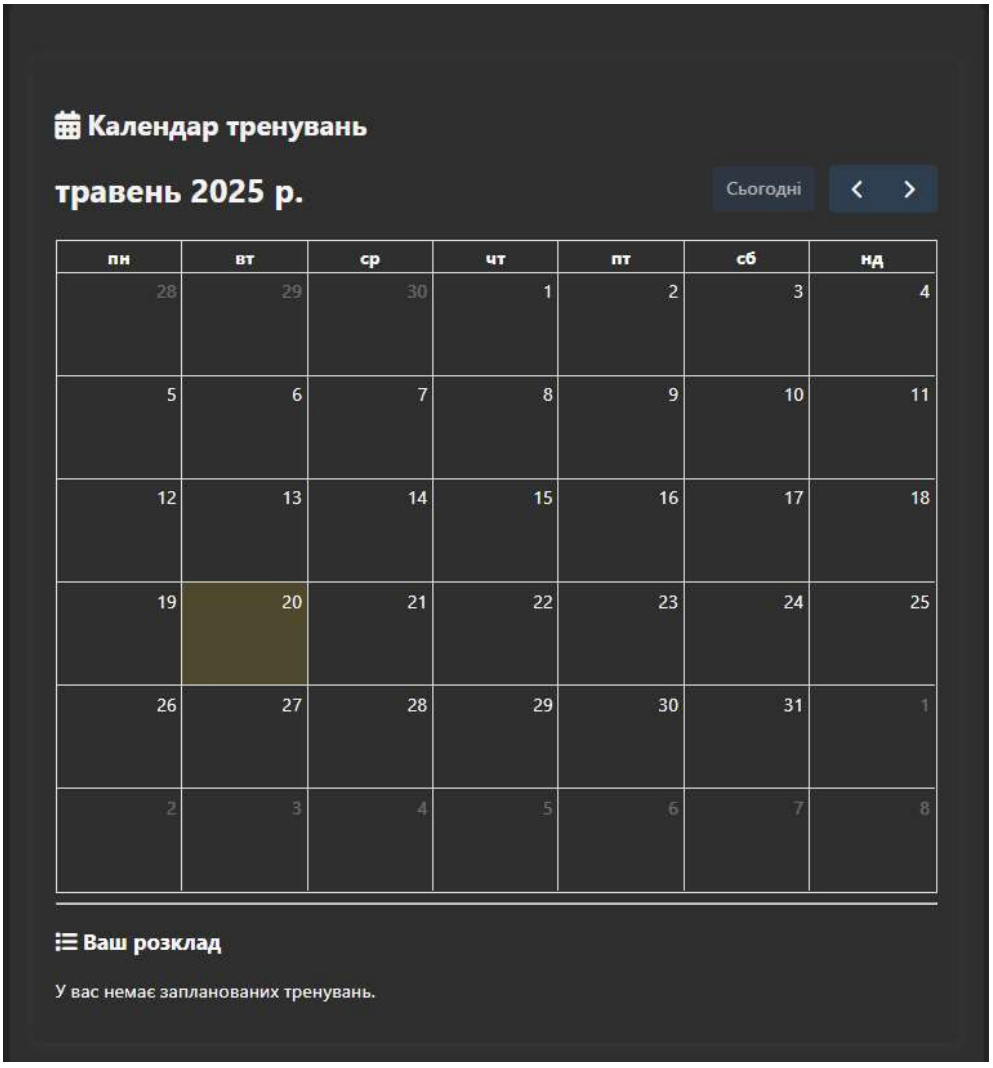

Рис. 3.5. Календар тренувань створеного Web-ресурсу

Серверна частина надає маршрут для сторінки календаря, де контролер отримує з бази даних усі тренування, пов’язані з поточним користувачем. Ці дані передаються у шаблон, де вони можуть бути використані JavaScript-бібліотекою, щоб динамічно відображати тренування у потрібних датах.

На клієнтській стороні, у шаблоні calendar.html, створено HTML-структуру, в якій відображається календар. Для кожного дня календаря можуть бути виведені активні тренування, з коротким описом (назва, час, тип). У разі натискання на день - може відкриватись модальне вікно або форма для додавання нового тренування. Зокрема, користувач може обрати дату, ввести назву вправи, кількість повторень, підходів, вагу тощо. Ці дані надсилаються назад на сервер запитом, і після обробки вони додаються до бази.

Також у рамках розширення функціоналу було додано функцію ведення певних текстових заміток або ж нотаток (рис. 3.6). Функція нотаток у Web-ресурсі виконує роль особистого щоденника або нагадувального інструменту, який дозволяє користувачу зберігати важливі текстові записи, пов’язані з тренуваннями, харчуванням, самопочуттям або іншими аспектами фізичної активності. Вона сприяє персоналізації досвіду, адже кожен користувач може фіксувати власні думки, спостереження чи індивідуальні завдання.

Рис. 3.6. Інтерфейс додавання нотаток

Інтерфейс максимально простий: користувач відкриває сторінку з нотатками, бачить список записів, може додати новий або натиснути «редагувати». При додаванні нотатки здійснюється валідація - наприклад, заборона на порожній заголовок чи надто довгий запис. Після збереження - нотатка з’являється в особистому списку.

Підсумовуючи, реалізований функціонал Web-ресурсу охоплює ключові потреби користувачів, які прагнуть системно підходити до фізичного розвитку та контролю за здоров’ям. Усі компоненти системи - від обліку тренувань і фізіологічних показників до візуальних інструментів планування, нотаток і калькулятора калорій - спрямовані на створення зручного, доступного та ефективного середовища для підтримки активного способу життя. Така структура ресурсу дозволяє не лише зберігати дані, а й гнучко взаємодіяти з ними, що формує якісно новий рівень користувацького досвіду у сфері фітнесу та самопіклування.

3.2 Розроблений Web-інтерфейс та можливі сценарії взаємодії

Реалізований Web-інтерфейс є невід’ємною частиною загальної архітектури системи, адже саме він забезпечує безпосередню взаємодію кінцевого користувача з усіма модулями Web-додатку. Інтерфейс побудований із використанням шаблонізатора Jinja2 у поєднанні з HTML, CSS та JavaScript, що дозволяє створювати динамічні, адаптивні сторінки зі зручною структурою. Дизайн платформи виконано у темній кольоровій палітрі, що відповідає сучасним тенденціям Web-розробки та знижує навантаження на зір користувача при тривалій роботі.

Web-ресурс має логічну та структуровану навігацію: у верхній частині сторінок знаходиться фіксована панель меню, яка забезпечує швидкий доступ до основних розділів сайту: головної сторінки, календаря, планів тренувань, калькулятора калорій, особистого кабінету та сторінки з порадами. Такий підхід

дозволяє уникнути надлишкових кліків і забезпечує максимальну доступність функцій.

Після авторизації користувач потрапляє на персоналізовану інформаційну панель (dashboard), яка відображає його плани тренувань, кожен із яких представлено у вигляді інтерактивного елемента. Система дозволяє переглядати деталі плану, додавати нові тренування або редагувати наявні. Кожне тренування містить дані про тип вправи, кількість підходів, повторень, використану вагу або тривалість. Інтерфейс побудований так, щоб навіть початківець міг інтуїтивно додати тренування або оновити дані про своє фізичне навантаження.

Одним з ключових елементів функціоналу є візуальний календар, який реалізовано як інтерактивну сітку із можливістю додавання, редагування та перегляду тренувальних подій за датою. Користувач може запланувати конкретне тренування на обрану дату, додати до нього короткий опис, а у перспективі - отримати нагадування або синхронізувати ці події з зовнішніми календарями. Така візуалізація дозволяє зручно розподіляти навантаження впродовж тижня або місяця, не втрачаючи контекст і послідовність тренувального процесу. Календар органічно інтегрований у загальну структуру інтерфейсу, а також підтримує динамічне завантаження подій через JavaScript.

Функціонал нотаток доповнює взаємодію з додатком як у тренувальному, так і у мотиваційному плані. Користувачі мають змогу створювати прості текстові записи, в яких можна зафіксувати самопочуття, власні спостереження, мікроцілі або результати. Нотатки відображаються на окремій сторінці в хронологічному порядку, що дозволяє користувачу переглядати свій прогрес не лише через числа, а й через особисті враження, що, в свою чергу, сприяє підтриманню мотивації та кращому самоконтролю.

Калькулятор калорій реалізовано у вигляді Web-форми з полями для введення ваги, росту, віку, статі та рівня активності. На основі формули Харріса-Бенедикта або іншої обраної методики обчислюється базова добова потреба у калоріях, що

відображається одразу після надсилання даних. Такий функціонал дозволяє користувачу більш усвідомлено підходити до побудови режиму харчування, орієнтуючись на індивідуальні параметри.

Додатково передбачено розділ із порадами, в якому надається інформація про ефективне планування тренувань, принципи раціонального харчування, відновлення, гідратацію, профілактику травм тощо. Ці поради згруповані у тематичні блоки та адаптовані для користувачів різного рівня підготовки - від новачків до досвідчених спортсменів. У перспективі можливе оновлення цих матеріалів без перезавантаження серверної частини, оскільки виведення реалізовано через шаблонізатор.

Сценарії взаємодії користувача з системою є логічно послідовними. Наприклад, користувач заходить у систему, переглядає свої поточні плани, додає нове тренування, фіксує результати та створює нотатку щодо самопочуття. Далі, у розділі календаря він планує наступне заняття, а за допомогою калькулятора коригує харчування на основі нових показників. Такий підхід дозволяє користувачеві мати повний контроль над процесом тренувань та власною динамікою.

Інтерфейс адаптований для роботи як на ПК, так і на мобільних пристроях. Завдяки використанню сучасних CSS-фреймворків (наприклад, Tailwind або Bootstrap) сторінки автоматично масштабуються під розміри екранів, забезпечуючи однаково комфортний досвід на різних типах пристроїв. Усі кнопки, поля вводу, списки та інші елементи виконано з дотриманням принципів доступності, що робить ресурс зручним навіть для людей із частковими обмеженнями.

Таким чином, реалізований Web-інтерфейс не лише забезпечує повну функціональність для роботи з тренуваннями, а й підтримує сучасний рівень зручності, персоналізації та ефективності, що є необхідним для створення повноцінного фітнес-рішення. Його логічна структура та інтуїтивність роблять

систему придатною для широкого кола користувачів та сприяють ефективному впровадженню у повсякденну практику.

Реалізований Web-ресурс підтримує низку найбільш поширених сценаріїв використання, які охоплюють повний цикл взаємодії користувача з системою - від реєстрації до щоденного користування аналітичними інструментами. Завдяки модульному підходу й логічній структурі інтерфейсу, користувачі можуть зручно й послідовно виконувати основні дії. Розглянемо типовий сценарій користування системою:

1. Початок роботи користувача. Новий користувач заходить на сайт і натиснувши відповідну кнопку (рис. 3.7) проходить процедуру реєстрації, заповнюючи форму з іменем користувача та паролем (див. рис. 3.2).

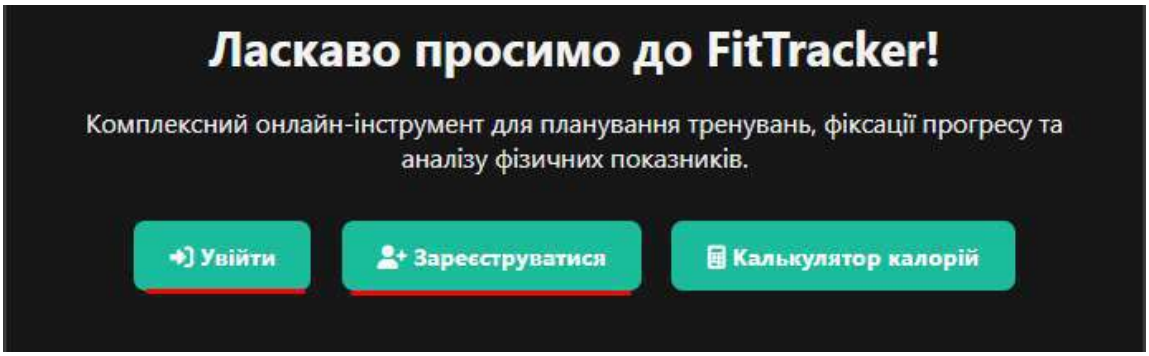

Рис. 3.7. Інтерактивні елементи головної сторінки створеного Web-ресурсу

Після успішної реєстрації він авторизується в системі та потрапляє на персональну панель, де може створити свій перший план тренувань. Це дозволяє одразу зануритися в роботу з додатком, без необхідності в додатковій конфігурації.

2. Створення індивідуального плану. Користувач натискає кнопку «Додати план» (рис. 3.8), вводить його назву (наприклад, «Тестовий план 1») та зберігає. Потім переходить до сторінки детального плану, де поступово додає тренування: дату, тип вправи, кількість підходів, повторень, вагу або тривалість. Усі ці дані миттєво зберігаються в базу даних і доступні для подальшого перегляду.

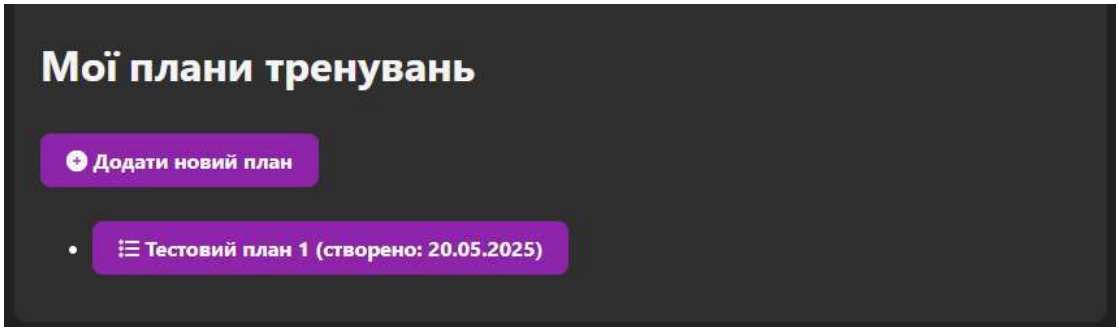

Рис. 3.8 Інтерактивні елементи Web-інтерфейсу для додавання та редагування плану тренувань

Наступний третій пункт - використання календаря. У розділі «Календар» користувач бачить свій місяць з тренувальними подіями (рис. 3.9). Він може натискати на дати, щоб додати або переглянути заплановане тренування.

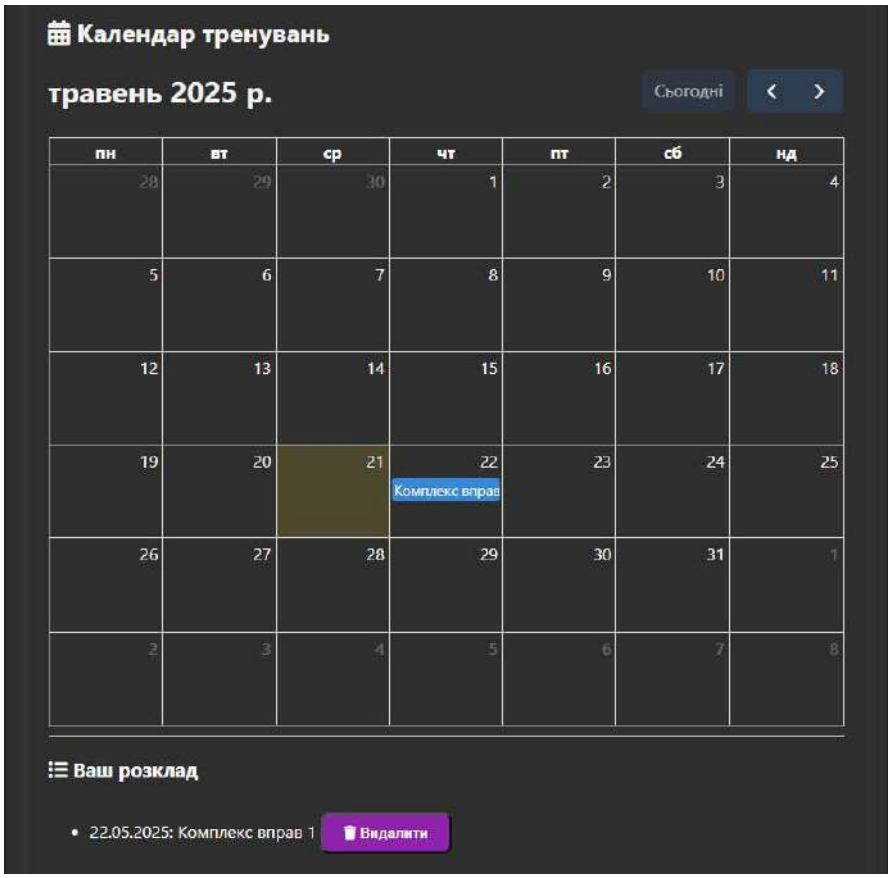

Рис. 3.9. Календар з доданою тренувальною подією

Також якщо користувач ставить своє тренування чи окремий комплекс вправ на наступний день, то програма надішле на пристрій відповідне сповіщення-нагадування що назначена подія наступить завтра (рис. 3.10).

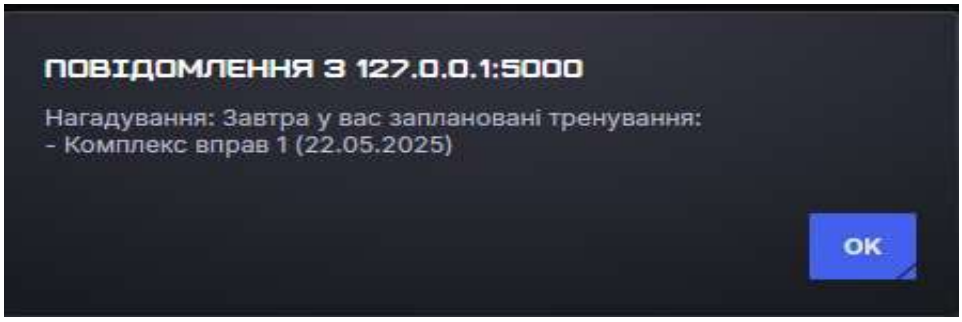

Рис. 3.10. Загальний вигляд нагадування користувачеві Web-ресурсу про наступну тренувальну подію

Такий візуальний контроль дозволяє зручно планувати навантаження на тиждень або місяць уперед, що є особливо корисним при підготовці до спортивних подій.

4. Ведення нотаток. Після завершення тренування користувач переходить до розділу нотаток, де залишає коментар щодо свого самопочуття, рівня енергії чи результатів. Наприклад: «Після інтервального бігу - легка втома, але серцебиття в межах норми» (рис. 3.11). Ці нотатки зберігаються й доступні для подальшого аналізу разом із тренувальними даними.

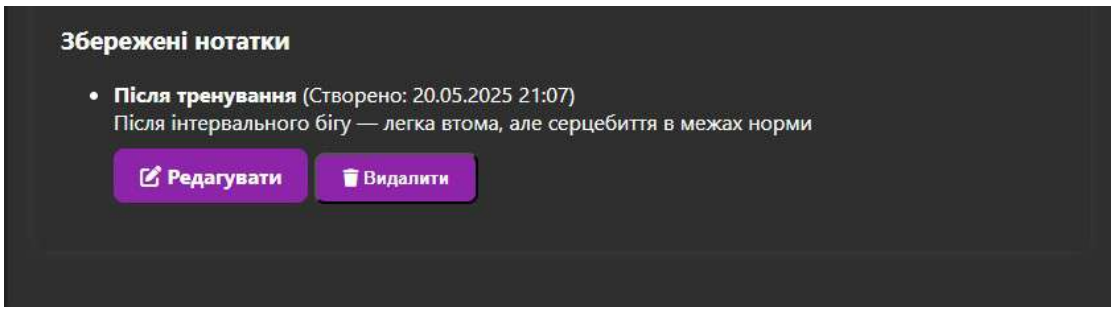

Рис. 3.11. Додана нотатка

5. Використання калькулятора калорій. У розділі «Калькулятор калорій» користувач вводить свої параметри - вік, вага, зріст, рівень фізичної активності - та отримує розрахунок добової потреби в калоріях. На основі цих даних користувач може коригувати свій режим харчування, щоб відповідати тренувальним цілям: набір м’язової маси, зниження жиру або підтримка форми. Приклад використання калькулятору калорій представлений на рис. 3.12.

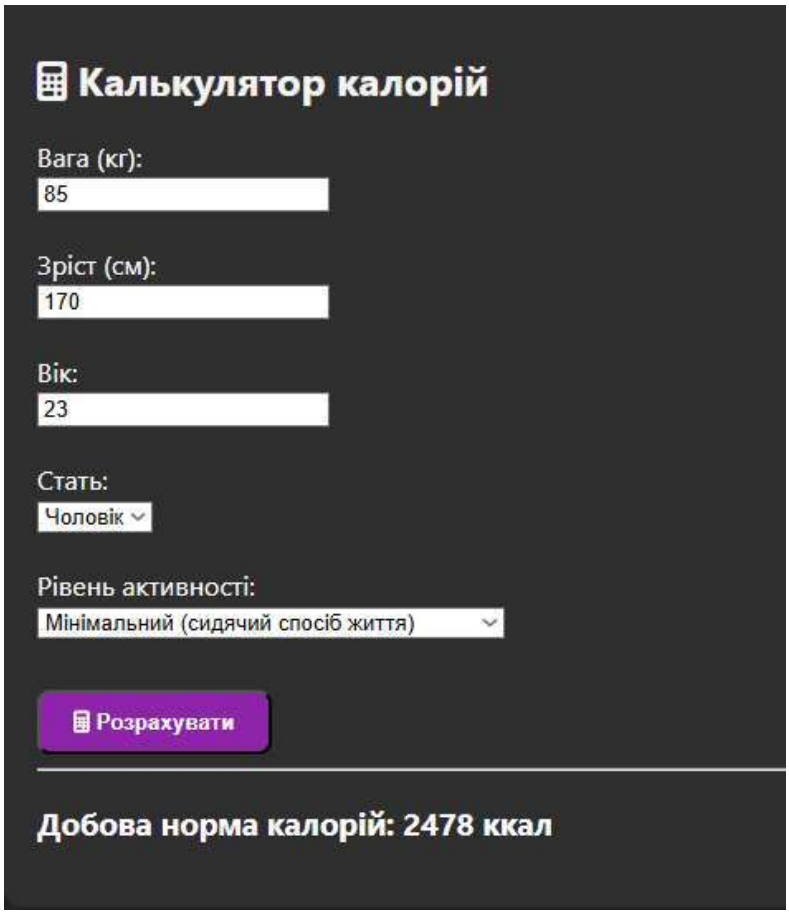

Рис. 3.12. Приклад використання калькулятору калорій

Після вводу та отримання даних про добову норму калорій користувача вони запишуться до бази даних, або якщо користувач розраховує свої дані не будучи користувачем додатку, дані будуть зберігатися до наступного оновлення сторінки відповідно.

6. Повернення до занять після перерви. Після тривалої перерви користувач заходить у систему, переглядає свої попередні плани, порівнює прогрес, оцінює фізіологічні показники, що були зафіксовані раніше, і створює новий план з урахуванням поточного стану. Завдяки збереженій історії тренувань і показників, користувач має змогу приймати зважені рішення щодо інтенсивності та типу майбутніх тренувань.

3.3 Аналіз ефективності використання та обмежень системи

Створений Web-ресурс виконує роль багатофункціонального інструменту для організації та аналізу особистих фізичних тренувань. Його функціональність охоплює основні потреби користувачів у фітнес-сфері - від базового планування активностей до ведення нотаток і візуалізації результатів у вигляді календаря, а також обчислення добової калорійної потреби. Завдяки цьому система позиціонується як ефективний засіб самоконтролю та мотивації.

З технічної точки зору, додаток реалізований на основі перевіреного технологічного стеку: серверна частина написана з використанням мікрофреймворку Flask, який забезпечує гнучке управління маршрутами, просту інтеграцію з базою даних та високу швидкість обробки запитів. Це робить систему легкою для розгортання, підтримки та масштабування. База даних на SQLite у поєднанні з ORM-бібліотекою SQLAlchemy забезпечує швидкий і зручний доступ до сутностей, що описують користувачів, плани тренувань, вправи та фізіологічні параметри.

Клієнтська частина побудована на основі шаблонізатора Jinja2, HTML та CSS, що дозволяє виводити динамічний контент без втрати керованості структури. Усі сторінки мають єдиний базовий шаблон, що забезпечує візуальну цілісність інтерфейсу. Темна тема, адаптивне компонування, логічна навігація - усе це робить взаємодію із застосунком приємною та інтуїтивною, особливо для користувачів без

технічної підготовки. Додаткові функції, як-от калькулятор калорій чи інтерактивний календар, сприяють залученню користувачів і створюють додану цінність.

У ході тестування було підтверджено, що система стабільно працює при стандартному навантаженні, не вимагає значних ресурсів комп'ютера або браузера та здатна обробляти паралельну роботу декількох користувачів. Наявність розмежування ролей у системі дозволяє чітко визначити функціональні межі між користувачами та тренерами, забезпечуючи гнучкість у наданні доступу до персональних даних. Кожен користувач може повністю контролювати свій тренувальний процес, бачити історію власного прогресу, додавати вправи, редагувати плани та аналізувати показники.

Однак, незважаючи на позитивні характеристики, система має низку функціональних і технічних обмежень. Одне з головних - обмежена автоматизація процесів. Вся взаємодія з даними відбувається вручну: користувач самостійно вводить параметри вправ, тренувань і фізіологічних показників. Це може призвести до втрати точності або зниження мотивації в разі потреби частої фіксації інформації. Відсутність повноцінної синхронізації з носимими пристроями (наприклад, фітнес-трекерами або смарт-годинниками) унеможливлює автоматичне оновлення параметрів і зменшує глибину аналітики.

Іншим аспектом, який обмежує застосування системи, є використання локальної бази даних SQLite. У межах індивідуального використання це є цілком виправданим рішенням, однак для розширення проєкту до багатокористувацької системи потрібна буде міграція до більш масштабованої СКБД, наприклад PostgreSQL або MySQL. Це дозволить працювати з великими обсягами даних, масштабувати систему та забезпечити багаторівневу безпеку.

З погляду UX-дизайну, система не повністю адаптована до мобільних пристроїв. Незважаючи на базову адаптивність інтерфейсу, деякі елементи відображаються некоректно на екранах з малою роздільною здатністю. Це може

знизити комфортність використання в мобільному середовищі - сфері, яка наразі домінує серед більшості користувачів Web-ресурсів.

Також варто відзначити, що система наразі не містить глибокої аналітики - прогнозування, оцінки ефективності програм тренувань, персоналізованих порад на основі змін показників. Такі функції є логічним продовженням розвитку проєкту та можуть бути реалізовані в майбутньому за допомогою бібліотек машинного навчання або зовнішніх API.

Попри ці обмеження, Web-застосунок цілком відповідає поставленим вимогам і цілям дипломної роботи. Він реалізує повний цикл користувацької взаємодії - від реєстрації до аналізу прогресу - і є готовим фундаментом для подальшого розвитку та впровадження нових функцій. Його структура логічно вибудувана, а технології - актуальні та придатні для розширення. Таким чином, реалізована система демонструє як приклад успішного застосування теоретичних знань на практиці, так і потенціал для використання у реальному середовищі, особливо за умов удосконалення й масштабування.

Висновки до розділу 3

Здійснено комплексний огляд реалізованого Web-ресурсу, призначеного для планування тренувань та аналізу фізичних показників користувачів. Розглянуто як загальну архітектуру застосунку, так і ключові функціональні модулі, зокрема: авторизацію та реєстрацію користувачів, створення індивідуальних планів тренувань, облік вправ і фізіологічних показників, календар для зручного планування активностей, калькулятор калорій, нотатки та інтерактивні елементи інтерфейсу.

Застосунок побудований на основі класичної клієнт-серверної архітектури із застосуванням сучасного технологічного стеку: Flask, SQLite, SQLAlchemy, HTML/CSS та шаблонізатора Jinja2. Таке технічне рішення дозволило реалізувати

систему, що є стабільною, функціональною та легко масштабованою. Весь функціонал згруповано у логічні модулі, що забезпечує зручність розробки, підтримки та розширення. Завдяки грамотній організації коду, реалізовано чіткий поділ між серверною та клієнтською логікою, що підвищує читабельність та зменшує ризики помилок.

Проведено аналіз ефективності застосунку, в якому підкреслено як сильні сторони системи, так і наявні обмеження. Зокрема, Web-ресурс виконує поставлені задачі в межах індивідуального використання, однак потребує вдосконалення у сфері автоматизації, адаптивності інтерфейсу до мобільних пристроїв і аналітичного функціоналу. Визначено можливості подальшої інтеграції зі сторонніми сервісами, підключення пристроїв трекінгу та реалізацію API для мобільної версії додатку.

У результаті проведеної розробки було створено повноцінний Web-застосунок, який відповідає вимогам сучасного користувача та має чітко окреслений потенціал розвитку. Це свідчить про доцільність обраного підходу та підтверджує практичну цінність реалізованого проєкту.

ВИСНОВКИ

При виконанні дипломної роботи був створений повнофункціональний Web-ресурс, призначений для планування фізичних тренувань і аналітичного моніторингу фізіологічних показників користувача. Від самого початку дослідження було поставлено завдання не лише розробити програмний продукт, але й обґрунтувати його необхідність, спроєктувати надійну інформаційну модель та забезпечити зручний, сучасний інтерфейс для взаємодії з кінцевим користувачем.

Першочергово було проаналізовано загальні тенденції в галузі цифрових технологій, що стосуються самоконтролю, ведення активного способу життя та персоналізованого підходу до тренувань. У межах дослідження розглянуто актуальні проблеми, зокрема зростання малорухомого способу життя, вплив пандемії на фізичну активність населення, а також поширення інтересу до засобів цифрового здоров’я. Встановлено, що більшість існуючих фітнес-додатків орієнтовані лише на вузькі аспекти, не забезпечуючи комплексного підходу до поєднання тренувального планування, аналітики, моніторингу фізіологічних показників і рекомендацій. Такий стан визначив необхідність розробки системи, яка б інтегрувала ці компоненти в єдиному Web-застосунку.

Здійснено моделювання інформаційної системи. Визначено основні сутності, які охоплюють користувача, тренера, план тренувань, окремі тренування, фізіологічні показники, нотатки та інші логічні компоненти. Структура системи проєктувалася з дотриманням принципів нормалізації, урахуванням можливостей масштабування та збереження цілісності даних. Усі компоненти пов’язані між собою відповідними зв’язками, що дозволяє реалізувати гнучкий функціонал, включаючи персоналізацію тренувальних планів, динамічний облік змін, зберігання історії результатів, а також можливість взаємодії користувача з тренером у межах системи. Діаграми, створені під час проєктування (ER-діаграма, діаграма

варіантів використання, класів та активності), дозволили узагальнити внутрішню логіку і визначити ключові точки взаємодії з користувачем.

На етапі розробки реалізовано серверну частину на базі мікрофреймворку Flask, що дало змогу гнучко керувати маршрутами, обробляти запити та налаштовувати логіку програми у відповідності до потреб системи. Для збереження даних використано SQLite у поєднанні з SQLAlchemy, що забезпечує ефективну роботу з таблицями без необхідності написання сирих SQL-запитів. Фронтенд частина побудована з використанням шаблонізатора Jinja2, HTML, CSS та JavaScript, що дозволяє динамічно відображати дані, забезпечувати адаптивний інтерфейс та інтерактивну взаємодію з елементами сайту.

Серед ключових функцій системи слід відзначити реєстрацію та автентифікацію користувачів, створення та редагування індивідуальних тренувальних планів, фіксацію виконаних вправ з параметрами, облік фізіологічних змін, аналітичний перегляд результатів, ведення нотаток, а також розміщення порад до тренувань. У рамках розширення функціоналу також реалізовано календар тренувань, що дозволяє візуально планувати активності, і калькулятор калорій, який забезпечує користувачеві швидкий розрахунок енергетичних витрат. Усі ці компоненти поєднуються в єдину систему з простим, зручним і зрозумілим інтерфейсом.

Результати реалізації показали, що запропоноване програмне рішення добре масштабується, дозволяє інтегрувати нові модулі, підтримує можливість розмежування ролей (звичайний користувач, тренер, адміністратор), забезпечує безпечне зберігання персональних даних та ефективну взаємодію з базою даних. Інтерфейс системи відповідає сучасним вимогам зручності та доступності, зокрема завдяки використанню темної теми, фіксованого навігаційного меню, адаптивної верстки та інтуїтивної структури сторінок.

У підсумку можна зробити висновок, що поставлені завдання повністю виконані. Було не лише розроблено та реалізовано програмний продукт, але й

обґрунтовано його доцільність, сформовано інформаційну модель, визначено архітектурну структуру, забезпечено роботу як серверної, так і клієнтської частини. Усе це дозволяє розглядати проєкт як базу для подальшого розвитку - зокрема, можливість інтеграції з носимими пристроями, мобільними застосунками, впровадження інтелектуального аналізу трендів, автоматизованої генерації планів та розширення системи рекомендацій.

Розроблений Web-ресурс демонструє, що за допомогою сучасних технологій можна створити гнучке, зручне та ефективне середовище для підтримки здорового способу життя, що відповідає реальним запитам суспільства.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Brown, A. Advanced Web Applications: A Practical Guide. – Manning Publications, 2021. – 380 p.

2. W3Schools. «Курси та посібники з веб-технологій» [Електронний ресурс]. – Режим доступу: https://www.w3schools.com/.

3. Smith, J. Python and Web Development: The Complete Guide. – Springer, 2019. – 320 p.

4. Коваленко, В. П. Основи веб-технологій. – Київ: Освіта, 2022. – 300 с.

5. MDN Web Docs. «Документація з HTML, CSS і JavaScript» [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/.

6. Graham, P. Hackers and Painters: Big Ideas from the Computer Age. – O'Reilly Media, 2004. – 272 p.

7. McFarland, D. S. CSS: The Missing Manual. – O'Reilly Media, 2018. – 816 p.

8. Google Developers. «Розробка Progressive Web Apps» [Електронний ресурс]. – Режим доступу: https://developers.google.com/web/progressive-web-apps.

9. Duckett, J. HTML & CSS: Design and Build Websites. – Wiley, 2011. – 490 p.

10. Anderson, K. Building Modern Web Applications. – Apress, 2021. – 320 p.

11. Wang, W. Learning Flask Framework. – Packt Publishing, 2019. – 400 p.

12. Bootstrap Documentation. «Документація Bootstrap» [Електронний ресурс]. – Режим доступу: https://getbootstrap.com/.

13. Литвиненко, Т. І. Проєктування баз даних для інформаційних систем. – Харків: Фактор, 2020. – 320 с.

14. Stack Overflow. «Обговорення питань веб-розробки» [Електронний ресурс]. – Режим доступу: https://stackoverflow.com/.

15. Pilgrim, M. Dive Into Python 3. – Apress, 2010. – 250 p.

16. Mitchell, J. Web Performance in Action: Building Fast Web Pages. – Manning Publications, 2018. – 320 p.

17. Grigorik, I. High-Performance Browser Networking. – O'Reilly Media, 2013. – 400 p.

18. Robertson, K. Designing User Interfaces for Web Applications. – Addison-Wesley, 2020. – 290 p.

19. SQLAlchemy Documentation. «Документація для SQLAlchemy» [Електронний ресурс]. – Режим доступу: https://docs.sqlalchemy.org/.

20. Бойчук, І. В. Python у веб-розробці. – Львів: Видавництво ЛНУ, 2021. – 270 с.

21. Novak, A. Modern Web Development with React. – Packt Publishing, 2020. – 450 p.

22. Flask Documentation. «Документація Flask» [Електронний ресурс]. – Режим доступу: https://flask.palletsprojects.com/.

23. Goodrich, M. Data Structures and Algorithms in Python. – Wiley, 2013. – 688 p.

24. Souders, S. High Performance Web Sites: Essential Knowledge for Front-End Engineers. – O'Reilly Media, 2007. – 180 с.

25. Черняк, Л. Ф. Інженерія програмного забезпечення: основи проєктування. – Київ: КНЕУ, 2019. – 400 с.

26. Nielsen, J. Designing Web Usability. – New Riders Publishing, 2000. – 420 с.

27. Глушаков, С. О. Архітектура інформаційних систем. – Львів: Львівська політехніка, 2021. – 270 с.

28. Venkatesan V., Hariharan G. An Implementation Approach Towards The Automation Of Document Formatting Using Python. International Journal of Aquatic Science. 2023. Т. 12. №. 02. С. 394-395.

29. McGuirk M. Performing web analytics with Google Analytics 4: a platform review. Journal of Marketing Analytics. 2023. Т. 11. №. 4. С. 854-868.

30. Porsche L., Zbiejczuk Suchá L., Martinek J. The potential of Google Analytics for tracking the reading behavior in web books. Digital Library Perspectives. 2022. Т. 38. №. 4. С. 532-541.

Додаток Е

ЗГОДА здобувача(чки) вищої освіти

Державного університету економіки і технологій про
перевірку кваліфікаційної роботи на прояви
академічного плагіату
та розміщення в Репозитарії Університету

Я, Будяк Ігор Анатолійович (ПІП),
підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.

Засвідчую, що кваліфікаційна бакалаврська (магістерська) робота
Розробка WEB-ресурсу для планування тренувань та аналізу динаміки фізичних показників.

(назва роботи повністю) виконана самостійно та не містить академічного плагіату. Я не надавав(ла) і не одержував(ла) недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Державного університету економіки і технологій ознайомлений(а). Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення норм академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

Також я поінформований(на), що відповідно до «Положення про Репозитарій (електронну базу даних) Державного університету економіки і технологій» зазначена робота буде розміщена в Електронному архіві Університету (Репозитарії ДУЕТ). З умовами такого розміщення ознайомлений(на).

10.06.2025 Будяк І.А

Дата підпис ініціали, прізвище (власноруч)