КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра теорії та технології програмування

Кваліфікаційна робота

на здобуття ступеня бакалавра

спеціальності 122 «Комп'ютерні науки»

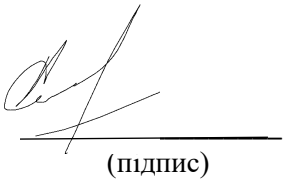
на тему:

РОЗРОБКА ВЕБ ЗАСТОСУНКУ ДЛЯ ПІДТРИМКИ ЗДОРОВОГО СПОСОБУ ЖИТТЯ

Виконала студентка 4-го курсу
Токан Юлія Олегівна

Науковий керівник:
к. ф.-м. н., доцент
Омельчук Людмила Леонідівна


(підпис)


(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент 
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«21» травня 2021 р.,
протокол № 11
Завідувач кафедри
М. С. Нікітченко _____
(підпис)

Київ – 2021

РЕФЕРАТ

Обсяг роботи 51 сторінка, 16 ілюстрацій, 14 таблиць, 16 джерел посилань, 3 додатки.

JAVA, JAVASCRIPT, HTML, CSS, SERVLET, POSTGRESQL, ВЕБЗАСТОСУНОК.

Об’єктом роботи є процес ведення щоденника харчування та оцінка енергетичної цінності раціону, регулярності прийомів їжі, водного режиму та добової активності. Предметом роботи є вебзастосунок для забезпечення можливості підрахунку калорій та поживних речовин, спожитої води та рухів за день.

Метою кваліфікаційної роботи є розробка вебзастосунку «Eat&Fit», який містить прототип щоденника здорового способу життя, що забезпечує можливістю підрахунку калорій та поживних речовин, водного балансу та середньої активності за день.

Методи розроблення: розробка програмного продукту на основі розрахунку добової норми калорій та поживних речовин. Інструменти розроблення: інтегроване середовище розробки IntelliJ IDEA, мова програмування Java.

Результати роботи: досліджено основні етапи розробки вебсайту, вимоги та програмні засоби для його створення. Спроєктовано, розроблено та протестовано вебзастосунок «Eat&Fit» для ведення щоденника здорового способу життя.

Вебсайт «Eat&Fit» може застосовуватися для контролю ваги та раціону в цілому. Також він може використовуватися для оптимізації процесу набору м’язової маси тіла чи схуднення паралельно з рекомендаціями дієтологів чи фітнес-тренерів.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

API – Application Programming Interface;

JavaEE – Java Enterprise Edition;

JavaSE – Java Standard Edition;

JDBC – Java DataBase Connectivity;

SQL – Structured Query Language;

HTTP – HyperText Transfer Protocol;

MVC – Model-View-Controller;

JSP – Java Server Pages;

CSS – Cascading Style Sheets;

HTML – HyperText Markup Language;

JS – JavaScript;

JSON – JavaScript Object Notation;

XML – Extensible Markup Language;

БД – База Даних;

СУБД – Система Управління Базами Даних;

IDE – Інтегроване Середовище Розробки;

БЖВ – білки, жири і вуглеводи;

ЗСЖ – здоровий спосіб життя.

ВСТУП

Оцінка сучасного стану об’єкта розробки. Останнім часом спостерігається позитивна тенденція: все більше молоді відмовляється від згубних звичок і слідує основам здорового способу життя. Це не тільки особливе харчування або заняття спортом, це цілий комплекс заходів, спрямованих на поліпшення здоров'я і профілактику патологічних процесів в організмі. Дотримання правил здорового способу життя дозволяє значно збільшити тривалість життя і поліпшити його якість [1].

Правильне, здорове харчування та активний спосіб життя – основа здоров'я людини. Однак вести здоровий спосіб життя в наш час не так-то просто. Тобто почати звертати увагу не тільки на те, щоб їсти частіше ніж один раз на день (за вечерею), а й скласти раціон, який би не призводив до численних захворювань і раннього старіння. І тоді вона починає шукати інформацію про корисність та шкідливість продуктів, калорійність, вміст в них різних поживних речовин.

Необхідно фіксувати режим прийому їжі: сніданок, обід, вечеря, усі перекуси. Крім того, обов'язково вказуються всі напої, вода, кількість доданих ложок цукру, солі, соуси, мед, джем, кількість масла, використаного при приготуванні страви.

Одне з головних завдань ведення щоденника харчування – це гармонізація білків, жирів і вуглеводів. Їх баланс призводить до нормального самопочуття та функціонування організму. Важливо, щоб і білки, і жири, і вуглеводи обов'язково були присутні в раціоні. Важливо не тільки враховувати отримані і витрачені калорії, але і стежити за співвідношенням між БЖВ. Вченими було встановлено наступне співвідношення БЖВ для здорового

функціонування організму: з білків ми повинні отримувати 20-40% калорій, з жирів 25% -35%, з вуглеводів 30-50% [2].

За щоденником харчування можна оцінити енергетичну цінність раціону, регулярність прийомів їжі, зробити висновки про макро- і мікронутрієнтний склад харчування. Ця інформація стає фундаментом для подальших індивідуальних призначень.

Ідеальне харчування – це перш за все харчування окремої людини відповідно до її віку, конституції, генетичної схильності. Основна ідея, що стосується збалансованого харчування, полягає в тому, щоб найкраще забезпечити прояв всіх можливостей організму і його оптимальне функціонування [2].

На здоровий спосіб життя впливає не тільки кількість і якість з'їденої їжі, а й наш психологічний стан, скільки кроків ми сьогодні зробили за день, де встигли побувати. Щоб проаналізувати все це, необхідно завести щоденник способу життя. Він допоможе дізнатися не тільки, що ви з'їли, а й коли, як, де, а головне – чому це зробили. Ви зрозумієте, який зв'язок існує між фізичною активністю, настроєм і апетитом.

Усім знайомо, що сидячий і малорухливий спосіб життя негативно відбивається не тільки на нашій фігурі, але і на здоров'я, а значить і на настрої, і на працездатності. Регулярні заняття спортом позитивно впливають на людський організм, зміцнюють його фізичне і психологічне здоров'я, підвищують імунітет.

Актуальність роботи та підстави для її виконання. Для того щоб вести здоровий спосіб життя і бути в курсі прогресу, людині потрібно вести щоденник. Деякі дослідження навіть доводять, що підрахунок калорій, спожитої води та активності робить досягнення мети в два рази ефективніше.

Будь-який користувач інтернету в лічені секунди може знайти буквально все про те, що він з'їв або має намір з'їсти. Для цього потрібно мати під рукою комп'ютер, планшет, смартфон або будь-який аналогічний пристрій.

Одним з найголовнішим критерієм їх оцінки по праву може вважатися зручність і функціональність у використанні. Відсутність нав'язливої реклами, яка ускладнює перегляд контенту, гальмує операційну систему, з'їдає трафік і, нарешті, просто дратує – важлива перевага запропонованого в кваліфікаційній роботі застосунку.

Мета й завдання роботи. Метою кваліфікаційної роботи є розробка вебзастосунку «Eat&Fit», який містить систему для підтримки здорового способу життя, що забезпечує можливістю підрахунку калорій та поживних речовин, води та кількості активно проведеного часу. Для досягнення цієї мети поставлено такі завдання:

- дослідити існуючі застосунки для ведення здорового способу життя на ринку;
- дослідити застосування різних технологій для проєктування та реалізації вебсайту;
- розробити технічне завдання до програмного продукту «Eat&Fit»;
- розробити інтерфейс та дизайн вебзастосунку «Eat&Fit»;
- розробити вебзастосунок для підтримки ЗСЖ.

Об'єкт, методи й засоби розроблення. Об’єктом розроблення програми «Eat&Fit» є процес ведення щоденника здорового способу життя та оцінка енергетичної цінності раціону, регулярності прийомів їжі, водного балансу та активно проведеного часу.

Предметом роботи є вебзастосунок для забезпечення можливості ведення здорового способу життя.

Розробці програмного засобу передував аналіз готових рішень щоденника здорового способу життя та способи їх застосування.

В якості інструменту створення програмного засобу було обрано IntelliJ IDEA IDE – інтегроване середовище розробки мовою програмування Java від компанії JetBrains [3].

Можливі сфери застосування. Вебсайт «Eat&Fit» може застосовуватися для контролю ваги користувача, його раціону в цілому, способу життя. Також застосунок може використовуватися для оптимізації процесу набору м’язової маси тіла чи схуднення паралельно з рекомендаціями дієтологів чи фітнес-тренерів.

РОЗДІЛ 1.

ОГЛЯД НАЯВНИХ НА РИНКУ ЗАСТОСУНКІВ

На сьогодні в мережі Інтернет існує значна кількість вебзастосунків для підтримки здорового способу життя. Розглянемо детальніше найпоширеніші з них, визначимо їх переваги і недоліки.

1.1 Програма «My FitnessPal»

Передову позицію в переліку найпопулярніших застосунків для ведення щоденника харчування займає «My FitnessPal» [5]. Розробники стверджують, що система має найбільшу БД (близько 6 млн. найменувань страв та продуктів), що поповнюється кожен день. У застосунку реалізований оптимальний набір функцій: додавання власних продуктів та страв, статистична та звітна інформація про динаміку ваги, статистика основних поживних речовин, включаючи білки, жири, вуглеводи, сканер штрих-кодів.

Ще одним зручним плюсом «My FitnessPal» є синхронізація з вебзастосунком: вести щоденник можна як з комп'ютера, так і з телефону. Система безкоштовна, проте можна придбати платну підписку, що надає доступ до окремих додаткових функцій. З мінусів користувачі відзначають неможливість системи синхронізуватися з окремими фітнес-трекерами.

Приклад користувацького інтерфейсу програми «My FitnessPal» наведено на рис. 1.

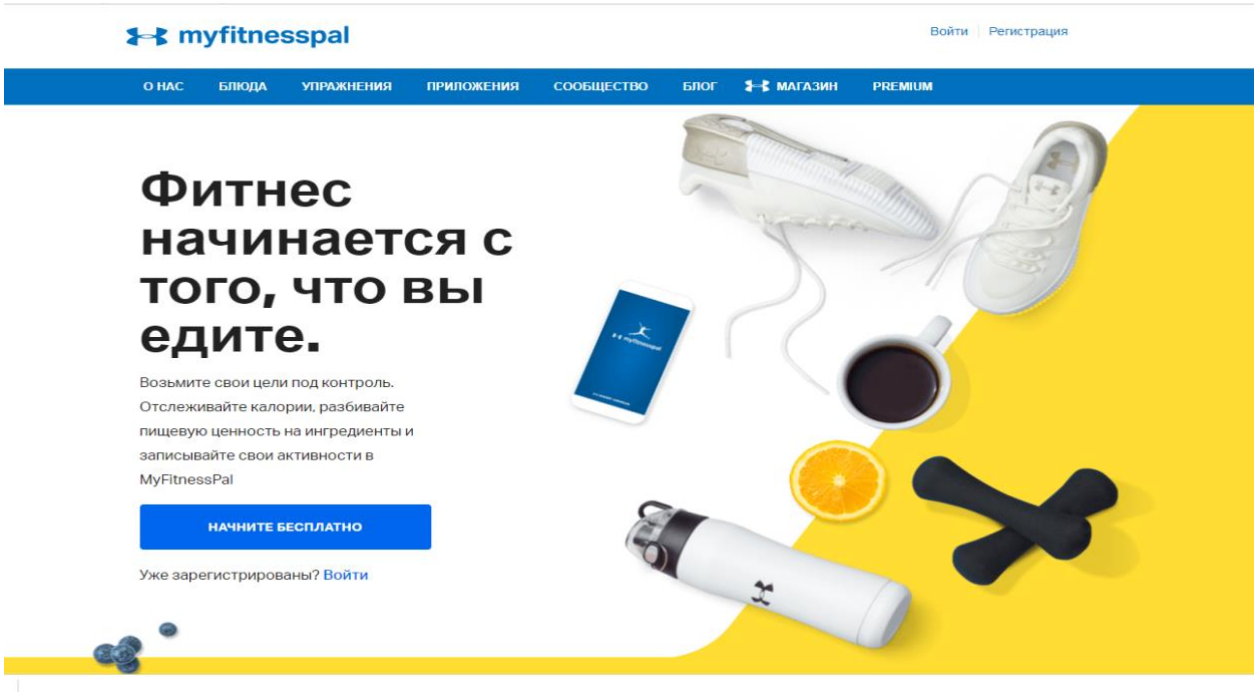


Рисунок 1. Головна сторінка програми «My FitnessPal»

1.2 Програма «Fat Secret»

«Fat Secret» – це безкоштовний вебзастосунок для ведення здорового образу життя без преміум-акаунтів, підписок і реклами [6]. Однією з головних переваг програми є приємний, лаконічний і інформативний інтерфейс (рис. 2). Система «Fat Secret» має базу даних продуктів, яка розділена на різні категорії: продукти, ресторанна їжа, популярні марки, супермаркети. Наявна можливість введення продуктів харчування за штрих-кодом. Крім стандартних функцій користувач інформується щодо кількості цукру, натрію, холестерину, клітковини. Також наявний перегляд щоденних вправ, щоб стежити за спаленими калоріями.

Також однією з цікавих функцій є розпізнавання зображень: користувач може сфотографувати продукти харчування, страви і вести щоденник в фотографіях.

Серед недоліків - недостатня кількість прийомів їжі, а також незручне додавання рецептів без зазначення порцій. Є розділ контролю за вагою, а ось контролю за об’ємом, на жаль, немає. Приклад користувацького інтерфейсу програми «Fat Secret» наведено на рис. 2.

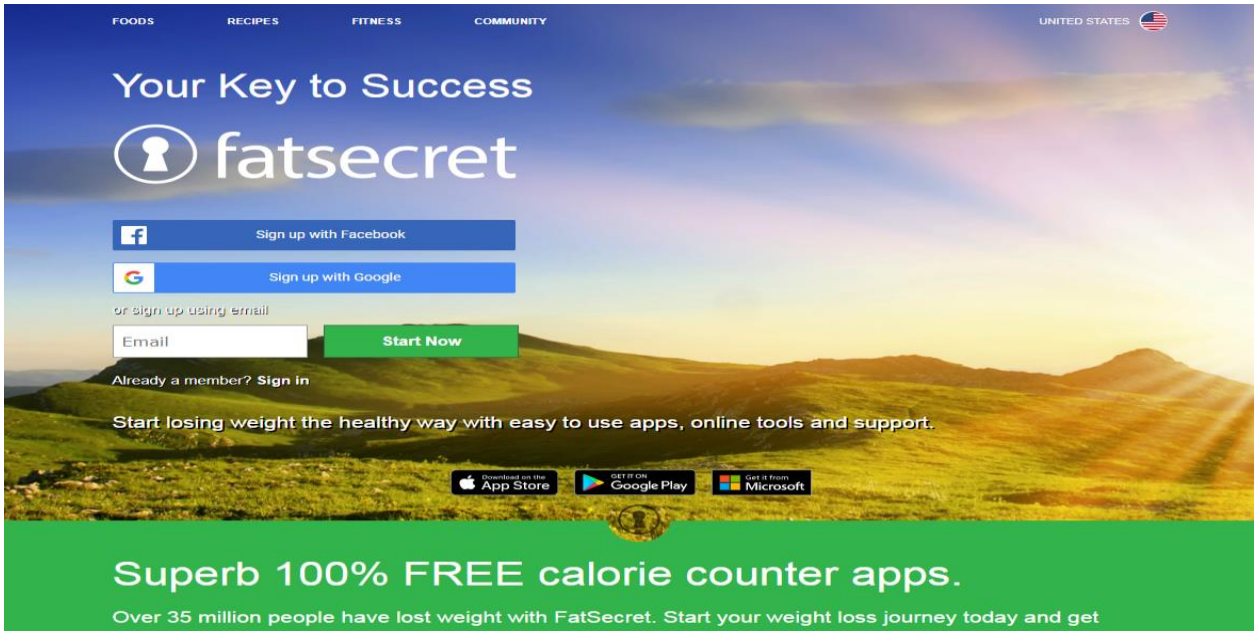


Рисунок 2. Головна сторінка програми «Fat Secret»

1.3 Сайт «Бережи фігуру»

Сайт «Бережи фігуру» [7] дозволяє розрахувати всі показники, що стосується фігури, ваги і способу життя: калькулятор калорій, параметри фігури, ідеальної ваги, відсотки жиру, календар схуднення, фітнес-калькулятори. Щоденники харчування і вправ, статистика харчування, збереження вимірів та інші інструменти дозволяють будувати графіки всіх параметрів вимірів і розрахованих показників.

Простий і мінімалістичний застосунок, що включає всі найнеобхідніші функції для ведення щоденника здорового способу життя.

Приклад користувацького інтерфейсу програми «Бережи фігуру» наведено на рис. 3

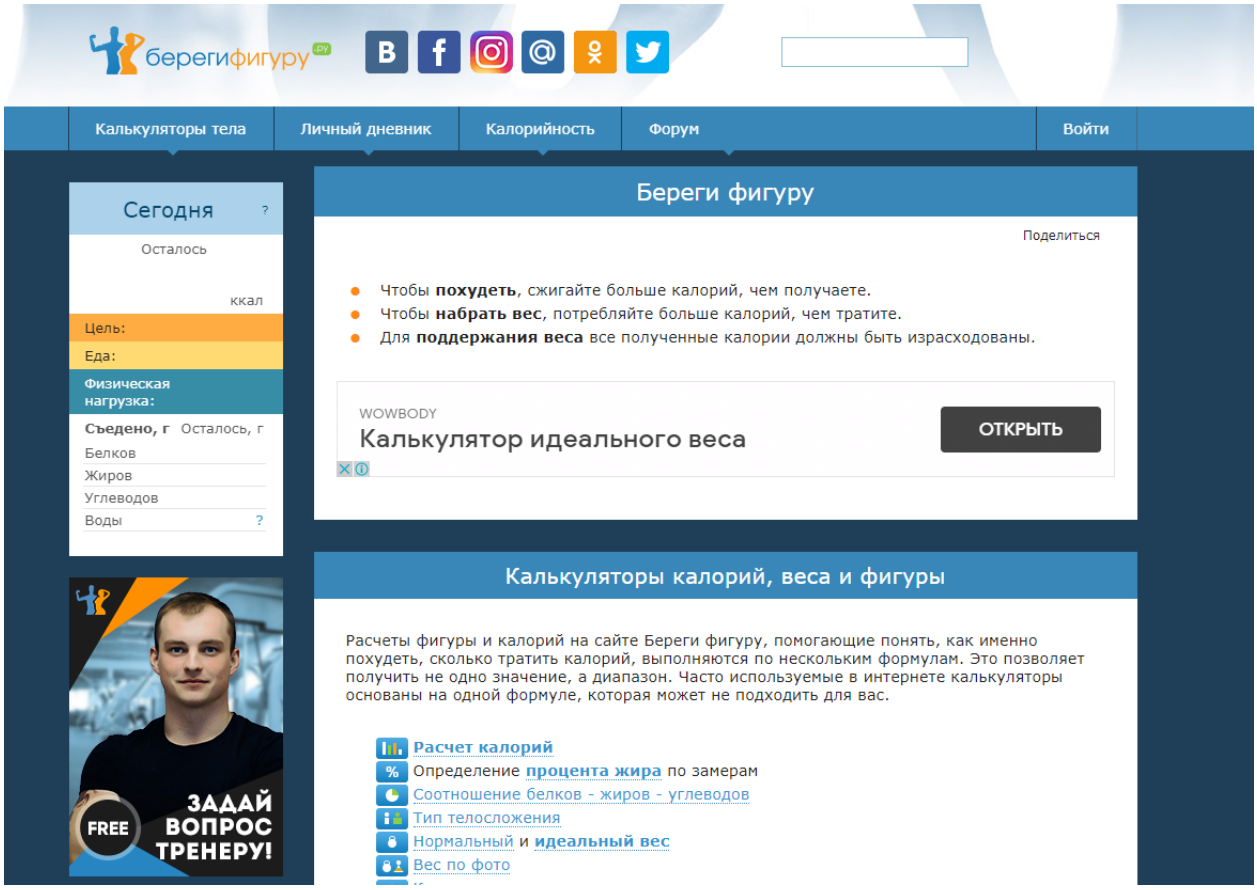


Рисунок 3. Головна сторінка сайту «Бережи фігуру»

1.4 Порівняння розглянутих рішень

Функціонал та інтерфейс розглянутих застосунків відрізняються, хоча вони спроєктовані і розроблені для практично однакових цілей. Оскільки інформація про архітектуру даних застосунків невідома, можна порівняти їх на основі таких характеристик як зручність інтерфейсу, підтримка основного

функціоналу, підтримка додаткового функціоналу, простота використання, можливість інтеграції з іншими платформами, безкоштовне користування (див. табл. 1).

Таблиця 1. Порівняння сервісів

Назва	**My FitnessPal**	**Fat Secret**	**Бережи фігуру**
Зручність інтерфейсу	+	+/-	+/-
Основний функціонал	+	+	+
Додатковий функціонал	+/-	-	+
Простота використання	+	+/-	+/-
Синхронізація з іншими платформами	+	-	-
Безкоштовне користування	+/-	+	+

Відповідно до порівняльної характеристики, наведеної в табл. 1 можна стверджувати, що при оцінюванні за заданими характеристиками найкраще виглядає застосунок «My FitnessPal», навіть незважаючи на деякі проблеми. Тому, при розробці власної реалізації буде зроблений акцент на подібній варіант, проте будуть враховані всі недоліки.

На основі аналізу сайтів ведення здорового способу життя, що були розглянуті вище, можна стверджувати, що доцільною є розробка сайту, котрий

повинен максимально можливо поєднати переваги кожного з розглянутих варіантів та виправити їхні недоліки.

РОЗДІЛ 2.

ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

Розглянемо інструменти, технології та інші рішення, які дозволяють оптимальним чином задовольнити описані раніше вимоги. В силу того, що проєкт обмежений невеликим терміном реалізації, було прийнято рішення розглядати і аналізувати найбільш популярні засоби в своїй категорії.

2.1 Вибір мови програмування серверної частини

Найважливішою частиною у розробці будь-якої інформаційної системи є проєктування архітектури, адже правильно спроєктована гнучка архітектура спрощує майбутню підтримку, розширення та удосконалення системи. Помилки, здійснені при проєктуванні та виявлені на подальших етапах, потребують великих затрат для виправлення.

Саме тому вибору інструментів, мов програмування, технологій та бібліотек потрібно приділити особливу увагу. Необхідно обирати технології для розробки, які максимально задовольняють вимоги функціоналу системи, адже широкі стандартні можливості бібліотек та технологій дозволяють розробнику уникнути написання частини програмного коду через використання вже готових рішень. Широкі можливості обраних бібліотек та технологій також полегшують модернізацію та внесення змін до системи.

Вибір технологій для реалізації проєкту потрібно здійснювати з урахуванням наступних вимог:

· застосунок повинен мати клієнт-серверну архітектуру, тому обрана мова програмування повинна максимально спростити обмін запитами між сервером та клієнтом;

· застосунок працює з БД, що містить великий об’єм даних, тому потрібно зробити механізм взаємодії з БД максимально гнучким та прозорим, що дозволить зменшити кількість програмного коду та полегшить підтримку застосунку у майбутньому.

Мова програмування Java з самого початку розроблялась в першу чергу для написання вебзастосунків, тому її технології надають розробнику широкий спектр можливостей для розробки вебсервісів, включаючи бібліотеки сторонніх розробників, які використовуються при роботі з Java EE. Для взаємодії з клієнтською частиною додатку необхідно організувати обмін за принципом запит-відповідь, при цьому запити клієнта і відповіді сервера повинні бути у форматі HTTP-запитів. Такий функціонал реалізований у Java Servlet API. Клас HttpServlet призначений для обробки HTTP-запитів (методи Get, Post, Put та Delete) [8].

Важливою особливістю роботи з HTTP-запитами у Java Servlet API є те, що завантаження файлів на сервер або з серверу не вимагає великої кількості коду, а здійснюється через вхідний/вихідний потоки запитів і відповідей. Тобто, можна завантажити файл на сервер прямо з вихідного потоку запиту або відправити файл клієнту, записавши його у вхідний потік відповіді сервера [9].

2.2 Вибір технології взаємодії з базою даних

Так як функціонал застосунку щільно пов’язаний з БД, необхідно спроєктувати архітектуру серверної частини таким чином, щоб максимально спростити роботу з БД, а також залишити можливість для подальшого вдосконалення та розширення проєкту.

Бібліотека JDBC – прикладний програмний інтерфейс Java, який визначає методи, за допомогою яких Java-додаток здійснює доступ до БД. JDBC – це платформо-незалежний стандарт взаємодії Java-застосунків з різноманітними СУБД, реалізований у вигляді пакета java.sql, що входить до складу Java SE. Бібліотека дозволяє виконувати SQL-запити (вибірку, додавання, оновлення чи видалення даних) та процедури безпосередньо з програмного коду [10].

2.3 Вибір мови програмування клієнтської частини

Клієнтська частина вебзастосунку – це графічний інтерфейс; це то, що користувач бачить на сторінці. Графічний інтерфейс відображається в браузері. Користувач взаємодіє з вебзастосунком саме через браузер, натискаючи на посилання і кнопки.

Основна мова, якою описується графічний інтерфейс вебзастосунку - це HTML. Дана мова описує структуру вебзастосунку, розміщення на ній компонентів.

Оформлення вебсторінок, їх стиль і колірна схема описуються в таблицях стилів – CSS.

Для "пожвавлення" графічного інтерфейсу, надання йому динамічності, використовуються додаткові технології: скрипти JavaScript, а також вбудовані в вебсторінку компоненти, створені на Flash, Java або Silverlight.

2.4 Менеджер сервлетів Apache Tomcat

Контейнер сервлетів – це вебсервер для сервлетів Java і сторінок JSP, що представляє собою програму, яка займається системною підтримкою і забезпечує їх життєвий цикл слідуючи правилам, що зазначені в специфікаціях. Він працює як повноцінний автономний сервер, може надавати сторінки для іншого вебсервера, наприклад Apache, або інтегруватися в Java EE. Забезпечує зв’язок та обмін даними між сервлетом і клієнтами, бере на себе виконання таких функцій, як створення програмного середовища для функціонуючого сервлету, ідентифікацію та авторизацію клієнтів, організацію сесії для кожного з них [11].

Apache Tomcat – це контейнер, який дозволяє використовувати інтернет застосунки такі, як Java сервлети і JSP. Реалізує специфікацію сервлетів і специфікацію JavaServer Pages і JavaServer Faces. Написаний на мові Java. Tomcat використовується в якості самостійного вебсервера, як сервер контенту в поєднанні з вебсервером Apache HTTP Server, а також в якості контейнера сервлетів в серверах додатків JBoss і GlassFish [12].

РОЗДІЛ 3.

ПРИЗНАЧЕННЯ І ЦІЛІ РОЗРОБКИ ЗАСТОСУНКУ

3.1 Призначення застосунку

Задачею даної кваліфікаційної роботи є реалізація системи контролю усіх складових здорового способу життя, ґрунтуючись на перевагах та недоліках уже готових рішень. Розробка повинна задовольняти наступним основним вимогам:

- надавати зручне середовище для постійного користування застосунку;
- надавати доступ до інформації на сервері;
- надавати можливість додавати, зберігати, видаляти прийоми їжі та нові продукти;
- надавати можливість здійснювати контроль програмою кількості спожитих калорій та поживних речовин;
- надавати можливість стежити за кількістю спожитої води;
- надавати можливість відмічати активно проведений час;
- надавати рекомендації щодо найкращого часу для заняття спортом відносно зовнішніх чинників.

Провівши аналіз уже готових рішень та способів їх застосування, можна зробити висновок, що актуальною буде розробка з зручним інтерфейсом, достатньою кількістю функціоналу та максимально простою складністю користування. Це забезпечить зрозумілий та зручний доступ до застосунку для користувача, оскільки такі системи мають великий попит та є корисними.

Отже, очікується отримати багатосторінковий вебсайт з можливістю підрахунку поживних речовин, водного балансу, часу занять спортом та моделювання збалансованого раціону.

3.2 Цілі розробки застосунку

Застосунок «Eat&Fit» розробляється з метою:

- забезпечення розрахунку індивідуальної добової норми калорій;
- організації лічильника калорій та поживних речовин (білків, вуглеводів і жирів);
- надання списку продуктів з усіма макросами;
- стеження за раціоном харчування користувача;
- забезпечення розрахунку водного балансу;
- надання можливості вести облік часу, що був витрачений на заняття спортом;
- надання зручної і наочної графіки, яка допоможе налагодити здоровий спосіб життя.

Застосунок допоможе не тільки рахувати калорії і орієнтуватися у виборі продуктів. Вбудований калькулятор калорій та води покаже, скільки ви вже вжили і скільки ще залишилося до вашої індивідуальної щоденної норми. Денна норма калорій та води розраховується на самому початку виходячи з ваших параметрів і цілей. Крім окремих продуктів, додаток містить дані по калорійності супів, салатів і багатьох інших страв.

Також застосунок допоможе зручно заповнювати час, проведений активним заняттям спортом. Увесь інший функціонал можна подивитись на use-case діаграмі (див. додаток А).

3.3 Вимоги до застосунку

3.3.1 Вимоги до застосунку в цілому

Застосунок «Eat&Fit» повинен відслідковувати підтримання здорового способу життя, а саме: харчові звички та вживання білків, жирів, вуглеводів; водного балансу; заняття спортом. Звітувати користувачу про кількість води, калорій та поживних речовин спожитих протягом певного інтервалу часу. Застосунок повинен надавати деяку інформацію про рецепти страв та фізичні вправи.

В свою чергу кожен користувач повинен пройти етап автентифікації та авторизації перед використанням застосунку для підтвердження особистості та отримання відповідних прав доступу.

Користувач може додавати власні продукти чи страви, вказуючи калорійність та поживну цінність, до вже загальнодоступних продуктів. На основі переліку всіх доступних страв та продуктів власного інтересу користувач має можливість додати інформацію про прийом їжі.

Адміністратор має доступ до статистичних даних (список усіх продуктів та страв).

3.3.2 Вимоги до функцій, які виконуються програмою

Перелік основних функцій адміністратора наведено в табл. 2.

Таблиця 2. Перелік функцій та задач адміністратора що підлягають автоматизації

Функція	Задача
Керувати списком продуктів та страв для користувачів	Додавання нового продукту чи страви (в тому числі калорійність та показники), що будуть відкриті для всіх користувачів.
	Додавання нового продукту чи страви (в тому числі калорійність та показники), що потребують додаткового редагування.
	Редагування інформації про продукт чи страву (в тому числі калорійність та показники).
	Видалення інформації про продукт.

Перелік основних функцій користувача наведено в табл. 3.

Таблиця 3. Перелік функцій та задач користувача що підлягають автоматизації

Функція	Задача
Реєстрація	Введення короткої інформації про себе(логін та пароль).
	Введення повної інформації про себе.
	Редагування повної інформації про себе.
	Вихід з системи.

Робота з списком продуктів та страв	Додавання нового продукту чи страви (в тому числі калорійність та показники), що будуть доступними тільки користувачу.
Керування прийомами їжі	Додавання нового прийому їжі (страва, вага, час).
	Видалення прийомів їжі.
Відслідковування прогресу	Відображення статистики калорій, білків, жирів, вуглеводів за день.
Робота з рецептами	Пошук рецептів.
Відслідковування водного балансу	Додавання випитої склянки води.
	Відображення статистики вжитої води та процентного відношення з добовою нормою.
Керування спортивними досягненнями	Додавання інформації про час, проведений заняттям спортом.
	Відображення статистичної інформації активно проведеного часу.
	Редагування інформації щодо часу та кількості досягнень у спорті.
	Видалення заняття спортом.

3.3.3 Технічні вимоги до застосунку

Необхідно побудувати вебзастосунок, що підтримує таку функціональність:

- На основі сутностей предметної області створити класи. Класи і методи повинні мати відображає їх функціональність назви і повинні бути структуровані по пакетам.
- Інформацію про предметну область зберігати в БД. Для доступу до даних використовувати API JDBC з використанням пулу з'єднань;
- Застосунок має підтримувати роботу з кирилицею, в тому числі і при зберіганні інформації в БД.
- Код повинен бути задокументований.
- Застосунок має бути покрито Unit-тестами.
- Події в системі обробляти за допомогою Log4j.
- У застосунку необхідно реалізувати pagination, transaction.
- Застосунок має коректно реагувати на помилки і виключення різного роду.
- У застосунку повинна бути реалізована система авторизації і автентифікації.
- При розробці бізнес логіки використовувати сесії і фільтри.
- Використовуючи сервлети, JSP, JSTL реалізувати функціональності, запропоновані в постановці конкретної задачі.
- Браузери: Сайт повинен коректно відображатися в інтернет-браузерах MS Internet Explorer 5.5 і вище, Mozilla 1.7 і вище, Opera 7.54 і вище.
- На довільні некоректні дії користувача, пов’язані з введенням невірних даних, не заповненням обов’язкових полів введення в

формах та інші, які можуть бути оброблені програмою, генеруються відповідні повідомлення про помилки українською мовою, в межах загального дизайну сайту.

РОЗДІЛ 4.

РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

4.1 Опис організації інформаційної бази

4.1.1 Логічна структура бази даних

Для використання СУБД в даному застосунку, вона повинна забезпечувати реляційну модель роботи з даними. Сама модель має на увазі певний тип зв'язку між сутностями з різних таблиць. Щоб зберігати і працювати з даними, такий тип СУБД повинен мати певну структуру (таблиці) (див. додаток Б та табл. 4). У таблицях кожен стовпець може містити дані різного типу. Кожен запис складається з безлічі атрибутів (стовпців) і має унікальний ключ, що зберігається в тій же таблиці – всі ці дані взаємопов'язані між собою, як описано в реляційної моделі. Додатковою вимогою є безкоштовне розповсюдження.

Підсумовуючи вищесказане, обрано базу даних PostgreSQL. Використання зовнішніх ключів необхідна умова для зв’язку між об’єктами системи. Також об'єктно-реляційна система керування базами даних PostgreSQL, дозволить повністю зосередитися на розробці програми, уникаючи будь-які проблеми пов’язані зі збереженням даних.

Таблиця 4. Таблиці бази даних «Eat&Fit»

Номер	**Таблиця**	**Опис**
1	users	Таблиця для збереження короткої інформації про користувачів.
2	clients	Таблиця для збереження повної інформації про користувачів.
3	products	Таблиця для збереження інформації про продукти та страви.
4	meals	Таблиця для збереження інформації про вживання їжі.
5	eatPeriods	Таблиця для збереження інформації про час вживання їжі.
6	roles	Таблиця для збереження інформації про ролі користувачів.
7	genders	Таблиця для збереження інформації про стать.
8	nutritionGoals	Таблиця для збереження інформації про типи. мети використання додатку.
9	activities	Таблиця для збереження інформації про види активності.
10	water	Таблиця для збереження інформації про кількість спожитої води.
11	sport	Таблиця для збереження інформації про активно проведений час.

4.1.2 Опис таблиць бази даних програми

Таблиці складають основу бази даних – саме в них зберігаються всі дані. Передусім, повинна бути спроєктована структура кожної таблиці. Вона обумовлюється вмістом тих вихідних форм, запитів та звітів, які повинні бути отримані при роботі з базою даних.

У табл. 5-13 наведено детальну інформацію про структуру таблиць БД (ім’я поля, тип і розмір поля та опис поля).

Таблиця 5. Опис таблиці Users

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
login	varchar	Логін
password	varchar	Пароль
role_id	int	Ідентифікатор клієнта (foreign key)
client_id	int	Ідентифікатор клієнта (foreign key)

Таблиця 6. Опис таблиці Clients

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
name	varchar	Ім’я користувача
gender_id	varchar	Ідентифікатор статі (foreign key)
age	int	Вік користувача
height	double	Ріст користувача
weight	double	Вага користувача
nutritionGoal_id	int	Ідентифікатор мети (foreign key)

activity_id	int	Ідентифікатор активності (foreign key)
calories	int	Добова норма калорій
protein	double	Добова норма білків
fats	double	Добова норма жирів
carbohydrates	double	Добова норма вуглеводів

Таблиця 7. Опис таблиці Products

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
user_id	int	Ідентифікатор користувача (foreign key)
name	varchar	Ім’я продукту чи страви
calories	int	Кількість калорій на 100 г.
protein	double	Кількість білків на 100 г.
fats	double	Кількість жирів на 100 г.
carbohydrates	double	Кількість вуглеводів на 100 г.
common	boolean	Загальнодоступність

Таблиця 8. Опис таблиці Meals

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
user_id	int	Ідентифікатор користувача (foreign key)
product_id	int	Ідентифікатор продукту (foreign key)
weight	int	Вага вживаного продукту

eat_period_id	int	Ідентифікатор часу вживання їжі (foreign key)
date	timestamp	Дата

Таблиця 9. Опис таблиці EatPeriods

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
period	varchar	Тип

Таблиця 10. Опис таблиці Roles

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
role	varchar	Тип

Таблиця 11. Опис таблиці Genders

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
gender	varchar	Тип

Таблиця 12. Опис таблиці NutritionGoals

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
goal	varchar	Тип
coefficient	double	Коефіцієнт

Таблиця 13. Опис таблиці Activities

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
activity	varchar	Тип
coefficient	double	Коефіцієнт

Таблиця 14. Опис таблиці Water

Атрибут	**Тип**	**Опис**
id	int	Ідентифікатор (primary key)
user_id	int	Ідентифікатор користувача (foreign key)
volume	int	Об’єм спожитої води

4.2 Розробка вебзастосунку

Розробка вебзастосунку «Eat&Fit» проходила в декілька етапів. В наступних розділах буде описано кожен з цих етапів.

Провівши аналіз предметної області, було зроблено висновок, що потреба автоматизації щоденника здорового способу життя є досить актуальною в наш час.

База даних повинна містити різноманітну інформацію про продукти харчування, страви та рецепти. Чим більше інформації вдається зберегти, тим детільніше можна буде складати статистику. Сильною стороною реляційної БД PostgreSQL є високопродуктивні і надійні механізми транзакцій,

розширюваність, а це означає, що структурна схема може видозмінюватись з часом [13].

Отже, інформаційна система повинна містити повну інформацію про користувачів, опубліковані продукти харчування та страви, водний баланс та активність, необхідні для ведення статистики, зручну навігацію та пошук.

4.3 Постановка завдання

Попередній аналіз предметної області свідчить про необхідність реалізації наступних завдань:

- авторизація користувачів;
- зручне середовище для редагування та додавання продуктів, або страв;
- можливість користувача додати дані про його раціон;
- збереження та відображення статистики харчування;
- можливість управління доданими продуктами та прийомами харчування;
- збереження та відображення водного балансу;
- можливість користувача додати та переглянути дані про його активний спосіб життя;
- можливість адміністратора переглядати дані всіх продуктів та страв в зручному для нього вигляді;
- можливість адміністратора керувати зареєстрованими в системі продуктами або стравами.

Після підсумування вимог можна зробити висновок, що вебзастосунок буде мати зручний користувацький інтерфейс, доступний з будь якого комп’ютера або мобільного пристрою із доступом до інтернету.

4.4 Структура програми

Програма, що реалізує цільовий вебзастосунок структурно розбита на дві частини: клієнтську і серверну. Кожна з них відповідає за свої функції та навантаження і має різний ступінь доступу.

4.4.1 Серверна частина

Серверна частина відповідно розбита на свої структурні одиниці у відповідності по шаблона проектування MVC, що дозволяє вирішити деякі проблеми масштабованості програм, що мають класичну архітектуру [14]. У додатку В наведено рисунок структури проєкту.

Програмний продукт було реалізовано з використанням технології Java EE – Servlet API, це означає, що він має чітко розшаровану архітектуру, де кожна частина може бути змінена або модернізована, а також згідно Maven фреймворку організація архітектури проекту має структуру директорій.

Програмний код складається з наступних директорій-модулей: src, target, .idea, External Libraries та файлів .gitignore та pom.xml.

Папка src поділяється на main та test, а main у свою чергу на java, та resources. Модуль src містить усю логіку застосунку.

Модуль target містить скомпільований проект, що містить у собі скомпільовані .class файли проекту та вебархів з розширенням файлу .war, що передається на керування менеджеру сервлетів Apache Tomcat, який у свою чергу розгортає та встановлює його.

Файл pom.xml — це основний файл проекту, що використовує Maven, у ньому знаходиться основна інформація про проект, підключаються необхідні бібліотеки, плагіни та визначається порядок компіляції модулів.

У відповідності до вибраної архітектури, реалізовано моделі, які виступають у ролі посередника між програмою і базою даних. Класи моделей розміщені у папці entity.

Директорія .idea — зберігає в собі всі налаштування відкритого проекту при розробці. До таких налаштувань можуть належати комбінації гарячих клавіш, плагіни для розробки у середовищі, налаштування зовнішнього вигляду середовища розробки і т.д [4].

Файл з розширенням .iml створюється автоматично при створенні нового проекту. Він зберігає інформацію про модуль розробки, який може бути компонентом Java, Plugin, Android або Maven; зберігає шляхи модуля, залежності та інші параметри.

External Libraries — це модуль, що містить набір підключених сторонніх бібліотек. У випадку даної реалізації до цього модулю потрапляють файли бібліотек, які були підключені до проекту у файлі pom.xml. Саме фреймворк Maven займається скачуванням цих бібліотек та їх розміщенням у структурі проекту.

Файл .gitignore — це файл плагіну, який додається розробником. Він слугує допоміжним механізмом підтримки системи контролю версій, а саме якщо програміст вважає, що стан деяких файлів проекту впродовж етапу розробки не змінюється, або зміни цих файлів не впливають на кінцеву роботу програми – такі файли додаються у список ігнорованих.

Як вже було сказано вище, модуль src містить модуль main, який поділяється ще на два модулі:

— папка java містить код серверної частини, який написано на мові програмування Java. У відповідності до шаблона проектування MVC програма містить контролери, моделі і представлення, кожні з яких розміщені у відповідній папці. В папці controller розміщені класи, які відповідають за логіку роботи програми для адміністративно і клієнтської частини. Контроллери виступають як проміжковий рівень між моделями і видами [10];

— папка resources містить основні конфігураційні файли проекту – налаштування бази даних, конфігурацію логування, підтримки декількох мов та інші важливі параметри системи.

— папка webapps включає в себе статичні та динамічні файли репрезентаційної частини фінального продукту. Представлення формуються JSP-сторінками, що показують користувачам дані з моделі.

4.4.2 Клієнтська частина

Клієнтська частина відокремлена від основної логіки програми на сервері. Це дає змогу розробнику не витрачати час на додаткові обчислення чи перевірки, а приділити увагу саме зовнішньому вигляду та фізичному інтерфейсу майбутнього користувача. Для того аби кінцевий продукт роботи виглядав не тільки інформативно, але й лаконічно та інтерактивно були обрані графічні бібліотеки мови JavaScript та заздалегідь створені каскадні CSS стилі фреймворку Bootstrap.

4.4.3 Інтеграція зі стороннім сервісом

Зовнішній програмний інтерфейс (іноді інтерфейс прикладного програмування) (англ. Application programming interface, API) - набір готових класів, процедур, функцій, структур і констант, що надаються додатком (бібліотекою, сервісом) для використання в зовнішніх програмних продуктах. API визначає функціональність, яку надає програма (модуль, бібліотека), при цьому API дозволяє абстрагуватися від того, як саме ця функціональність реалізована [15].

Зовнішній програмний інтерфейс – система уніфікованих зв'язків, призначених для обміну даними між компонентами системи. API задає набір необхідних процедур, їх параметрів і способів поводження, а також надає функціональність, яку деякий програмний компонент надає іншим програмним компонентам.

Одна з ключових ідей проекту – надати користувачу інформацію про поточну погоду та на основі цього проаналізувати, якими саме видами спорту найкраще займатися у той момент та надати відповідні результати.

Для цих цілей нам необхідні дані про погоду, а саме API, за допомогою якого програма буде отримувати їх. Існує безліч варіантів, за які потрібно платити. Для наших цілей ми будемо використовувати сервіс OpenWeatherMap, який дозволяє відправляти до 60 запитів в хвилину безкоштовно. Якщо трафік буде більше, то можна або скористатися просунутим платним API, або налаштувати кешування для інформера погоди на сайті.

OpenWeatherMap API надає такий функціонал:

- поточні дані про погоду в будь-якому місці, включаючи понад 200000 міст;
- сервіс збирає та обробляє дані погоди з різних джерел, таких як глобальні та місцеві моделі погоди, супутники, радари та розгалужена мережа метеостанцій;
- формати JSON, XML та HTML;
- погодинний прогноз на 4 дні.

Інтеграція відбувалася в 3 етапи.

1. Отримання ключа OpenWeatherMap API.

Щоб отримати API ключ, нам потрібно зареєструватися в сервісі OpenWeatherMap.

2. Визначення поточних координат користувача.
3. Надсилання запита для отримання прогнозу погоди. У відповідь отримаємо дані в форматі JSON.

Далі ці дані обробляються і відображаються на сторінці користувача.

4.5 Розгортання та перенесення вебсервісу на віддалений сервер

Як віддалений сервер було обрано платформу Heroku. Спочатку було створено базу даних на віддаленому сервері, отримано url доступу.

Для розгортання було використано команду, застосовану до war-файлу:

```
heroku deploy:war --war EatAndFitServletProject_war.war
```

В результаті отримано готовий вебзастосунок за адресою:

https://eatandfit.herokuapp.com

Також код програми був завантажений на GitHub:

https://github.com/JuliaTokan/EatAndFit

4.6 Тестування сайту

Даний розділ описує тестування вебзастосунку після завершення його розробки та аналізу результатів тестування. Це дає можливість переконатися чи все було правильно зроблено, чи все нормально відображається.

Загалом тестування сайтів проводиться в кілька етапів [16]:

- залучається велика кількість тестерів для проведення тестування на зручність;
- перевіряються посилання;
- перевірка можливості сайту працювати при великих навантаженнях.

Крім цього вебзастосунки перевіряються на сумісність з сучасними браузерами. Для цього використовується декілька різних популярних браузерів, потім порівнюються результати відображення і на результатах робляться висновки щодо покращень на певному браузері, якщо той неправильно щось відображає.

В даному випадку тестування вебсайту буде проводитись на двох браузерах:

- Google Chrome;
- Safari.

4.6.1 Модульне тестування

Якість розробленого продукту — характеристика, яку намагаються досягти при розробці програмного забезпечення на максимальний рівень. Задля цього, для пошуку дефектів при розробці даного вебзастосунку якомога раніше, застосовано модульне тестування.

Модульний тест — це розроблений фрагмент коду, який покриває дуже маленьку, спеціалізовану функціональну область тестованого коду. Зазвичай модульний тест виконує деякий конкретний метод в певному контексті.

Оскільки необхідність модульного тестування є очевидною, при розробці системи використано переваги інтегрованої бібліотеки JUnit для виконання таких тестів. Інтегроване середовище тестування JUnit є засобом для модульного тестування клієнтських Java-застосунків.

4.6.2 Інтеграційне тестування

Інтеграційне тестування дозволяє перевірити вже відомі модулі, які були протестовані модульно, інтегруванням один з одним. Таким чином вони перевіряються на наявність несправностей: таке тестування в першу чергу виявляє помилки інтерфейсу.

Для даного типу тестування було використано Selenium WebDriver.

Це бібліотека, яка не має інтерфейсу користувача. Вона надає можливість програмам, що тестуються, взаємодіяти з браузером, керувати його поведінкою, отримувати від браузера певні дані і змушувати браузер виконувати певні дії. Це основний продукт, що розробляється в рамках проекту Selenium.

Вебсайт було протестовано. Зважаючи на результати тестування помітних недоліків відображення вебзастосунку в жодному браузері не знайдено, тому даний вебзастосунок сумісний з сучасними браузерами і працює коректно.

РОЗДІЛ 5.

ІНСТРУКЦІЯ КОРИСТУВАЧА

При відкритті сайту користувач потрапляє на головну сторінку (див. рис. 4). Тут перед ним відкривається широкий вибір активностей, проте ще більше функціоналу можна буде спробувати коли він зареєструється або увійде до системи. Він може ознайомитися з усією наданою інформацією англійською або українською мовою.

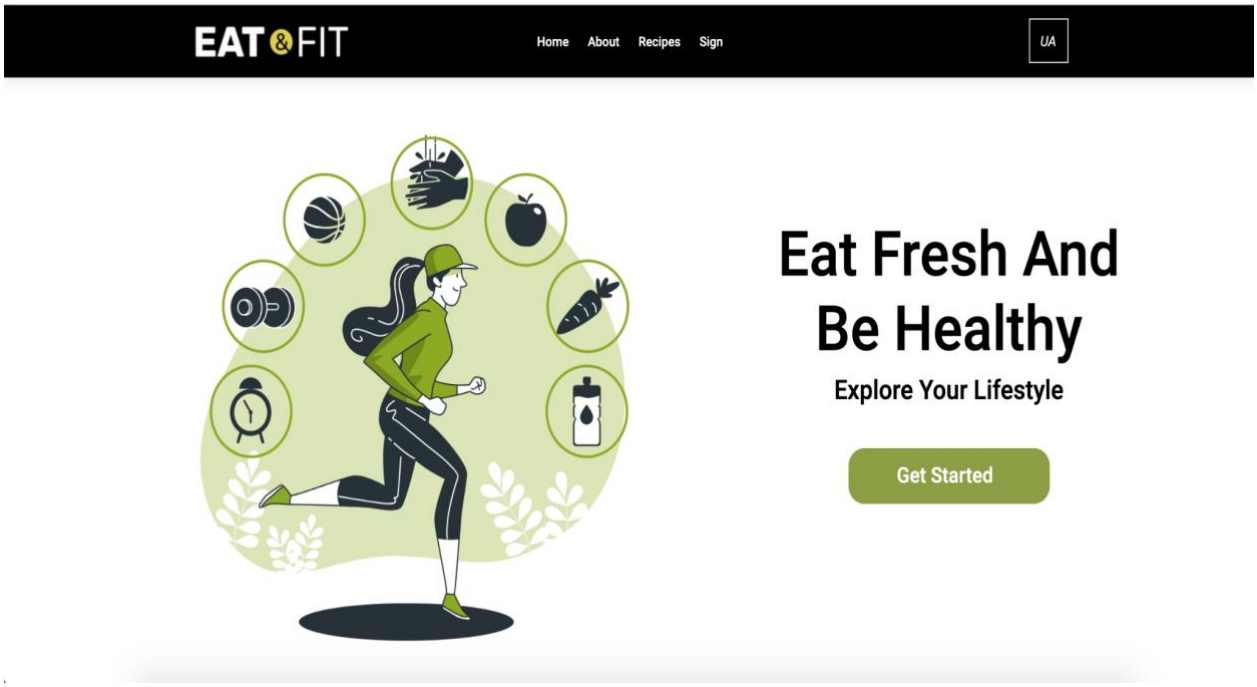


Рисунок 4. Головна сторінка «Eat&Fit»

При реєстрації/авторизації користувача (див. рис. 5) перевіряються усі дані на правильність. Перевірка відбувається як на стороні користувача, так і на стороні сервера. Також наявна перевірка на унікальність електронної адреси

при реєстрації. Якщо дані виявляються невалідними, то користувачу буде повідомлено про помилку.

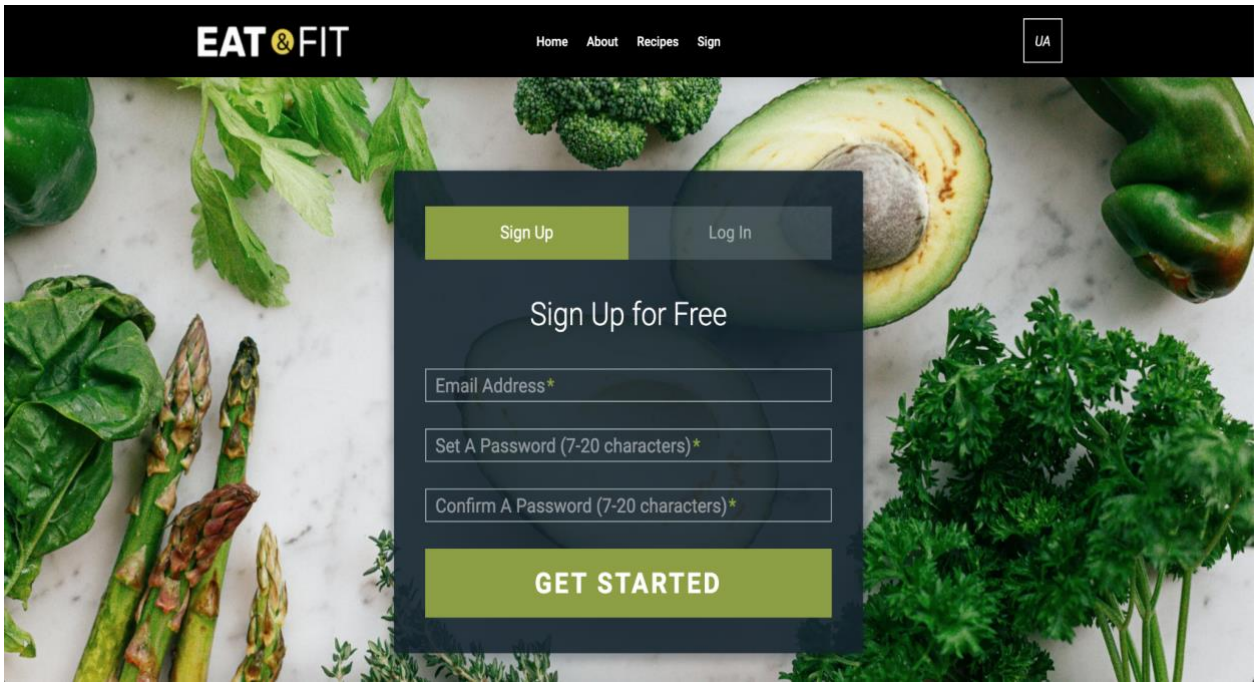


Рисунок 5. Сторінка реєстрації/авторизації

Після успішної реєстрації користувачу буде запропоновано заповнити інформацію про себе (див. рис. 6) для більш продуктивного використання застосунку. Але це не є обов’язковим і він зможе зробити це у будь-який момент потім. За замовчуванням буде встановлено середньостатистична норма. Надалі ця інформація може бути змінена користувачем.

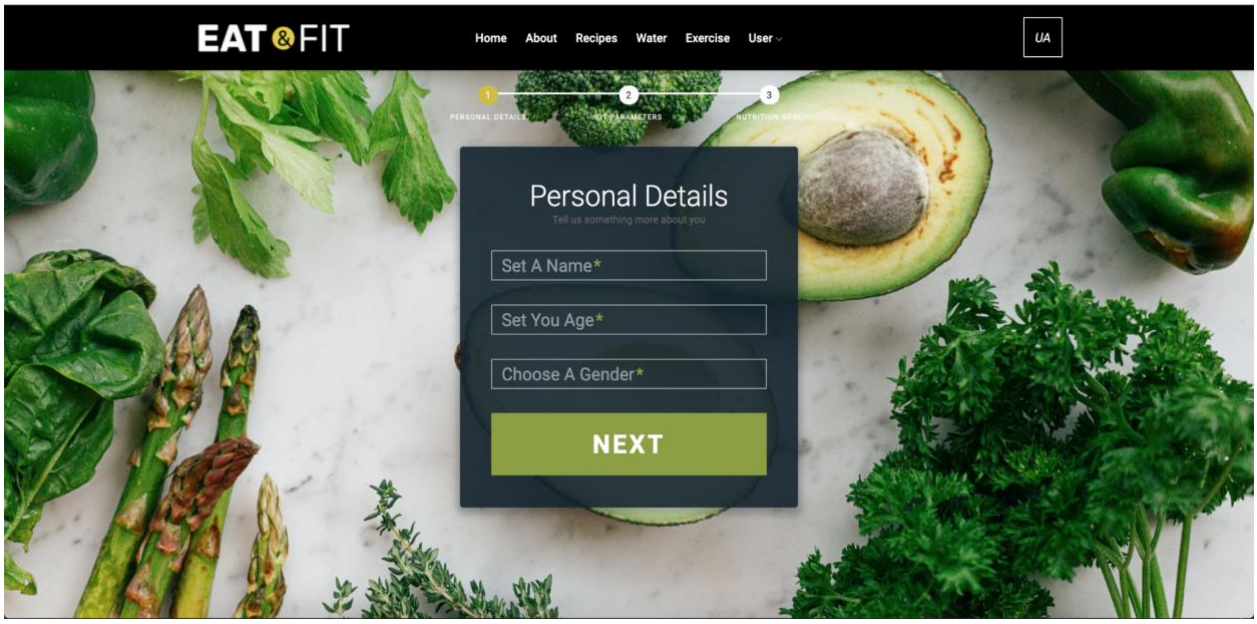


Рисунок 6. Сторінка заповнення інформації клієнта

Після успішної авторизації система запропонує користувачу рекомендації щодо виду спорту, яким можна займатися протягом для, зважаючи на погодні умови (див. рис. 7).

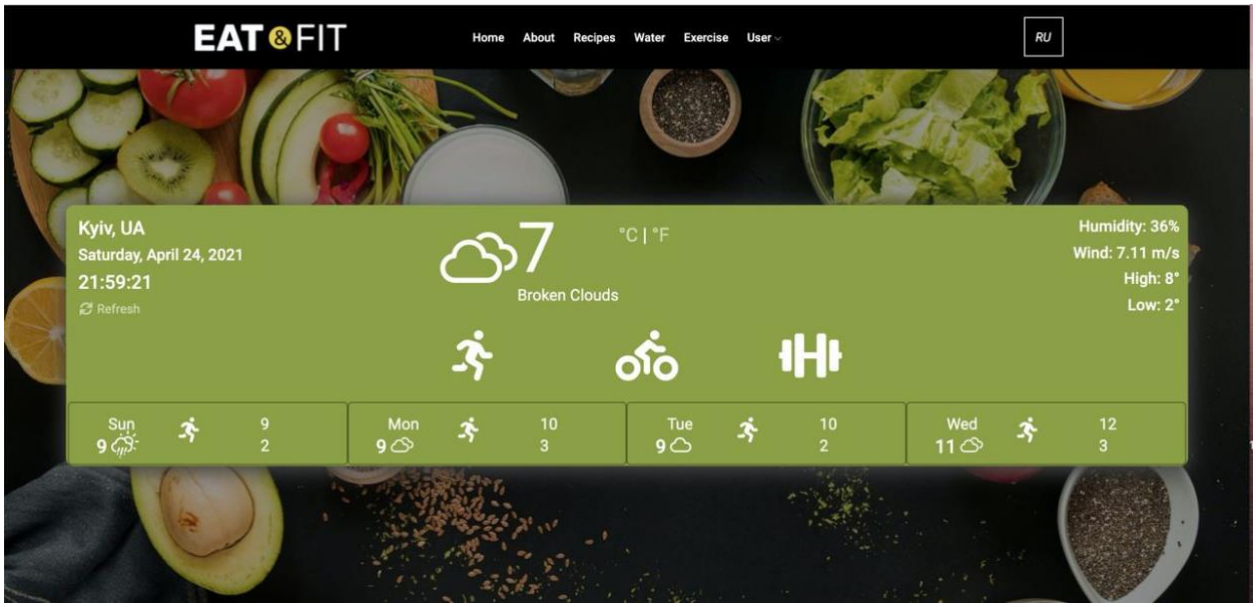


Рисунок 7. Щоденні рекомендації

Основний функціонал програми передбачає щоденне ведення статистики харчування, кількості випитої води, час, проведений займаючись спортом.

Кожний користувач на сторінці Вода може додати кількість склянок води, котрі він споживає за день (див. рис. 8). Там буде відображатися скільки він вже випив та скільки залишилось до щоденної норми, виходячи з його образу життя.

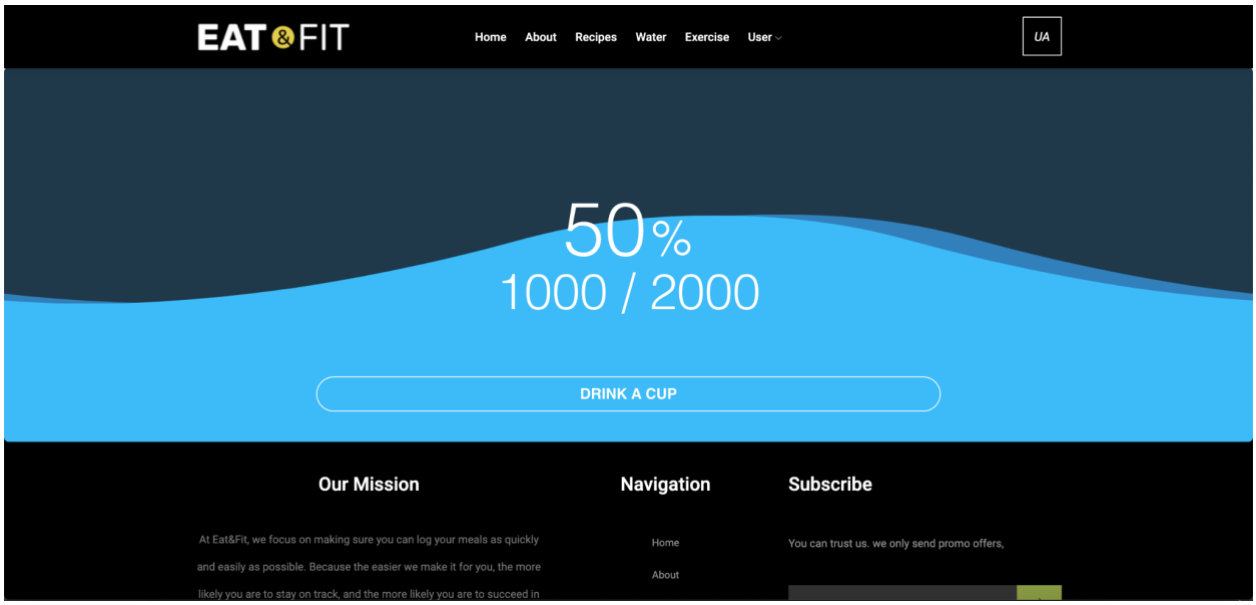


Рисунок 8. Сторінка водного балансу

Також на сторінці Активність користувач може вести облік своїх спортивних досягнень, заповнюючи відповідні результати активностей за день, що відповідають різним видам спорту (див. рис. 9).

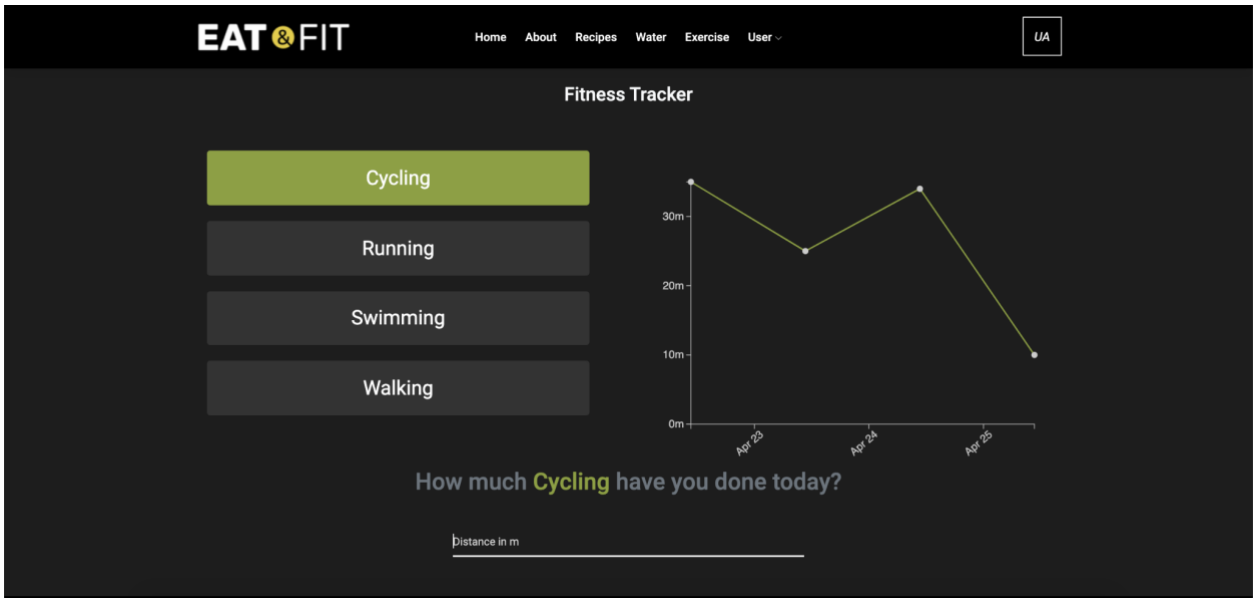


Рисунок 9. Сторінка спортивних досягнень

Кожний користувач на сторінці Денний раціон може заповнити інформацію про з'їдену їжу, вибравши її у списку, або створивши самостійно (див. рис. 10). Для зручного користування введений пошук даних. Продукт, що додається користувачем (не адміністратором) буде видимим тільки йому. При його додаванні, уся надана інформація буде перевірена на коректність і у випадку невалідності, користувач отримає повідомлення.

Щодня, на основі заповнення таблиці, буде створено статистику від початку дня (див. рис. 11). Користувач зможе побачити прогрес: скільки він з’їв калорій, білків, жирів, вуглеводів, а також його максимум калорій та скільки залишилось.

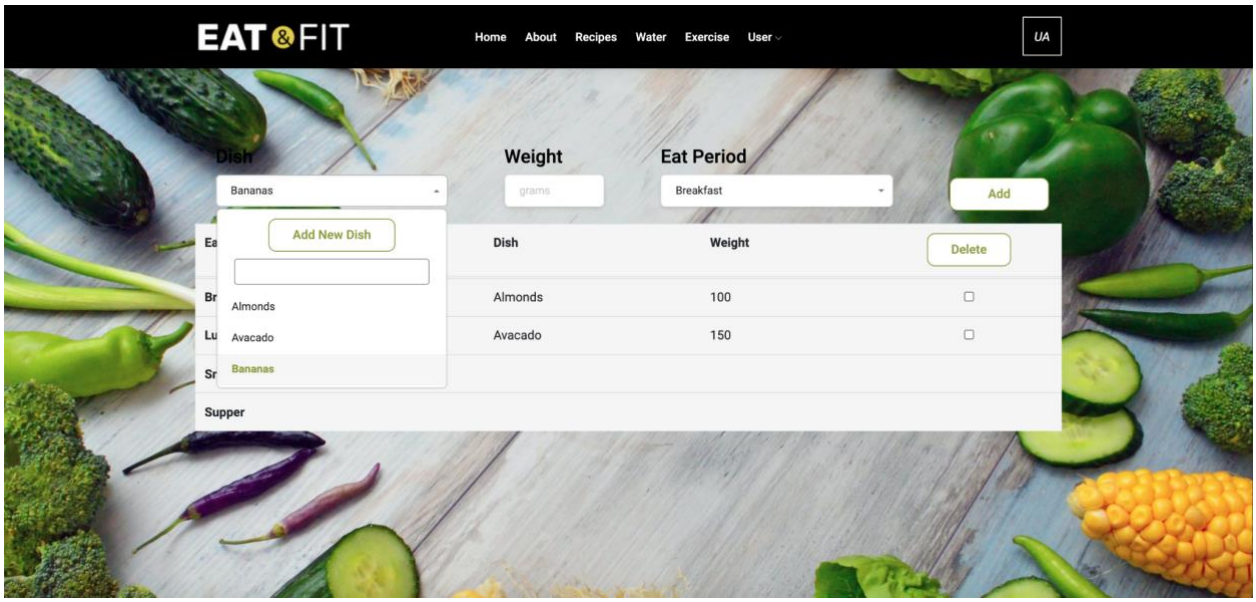


Рисунок 10. Сторінка щоденника харчування

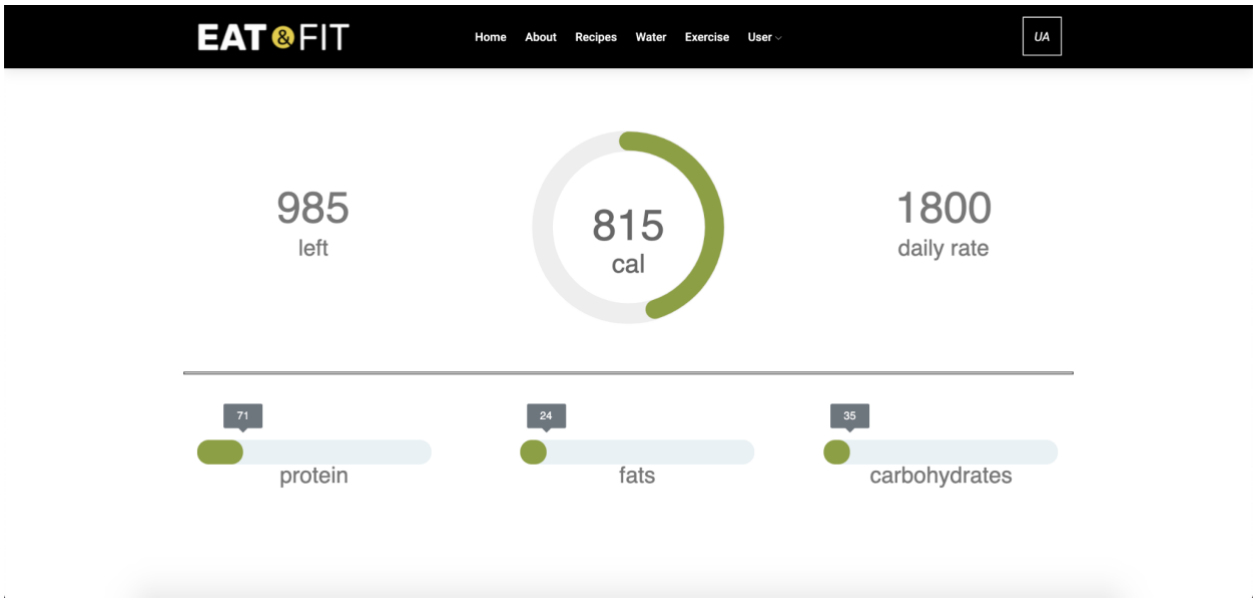


Рисунок 11. Сторінка досягнень користувача

Для будь-якої людини буде відкритий список рецептів будь-якої складності (див. рис. 12).

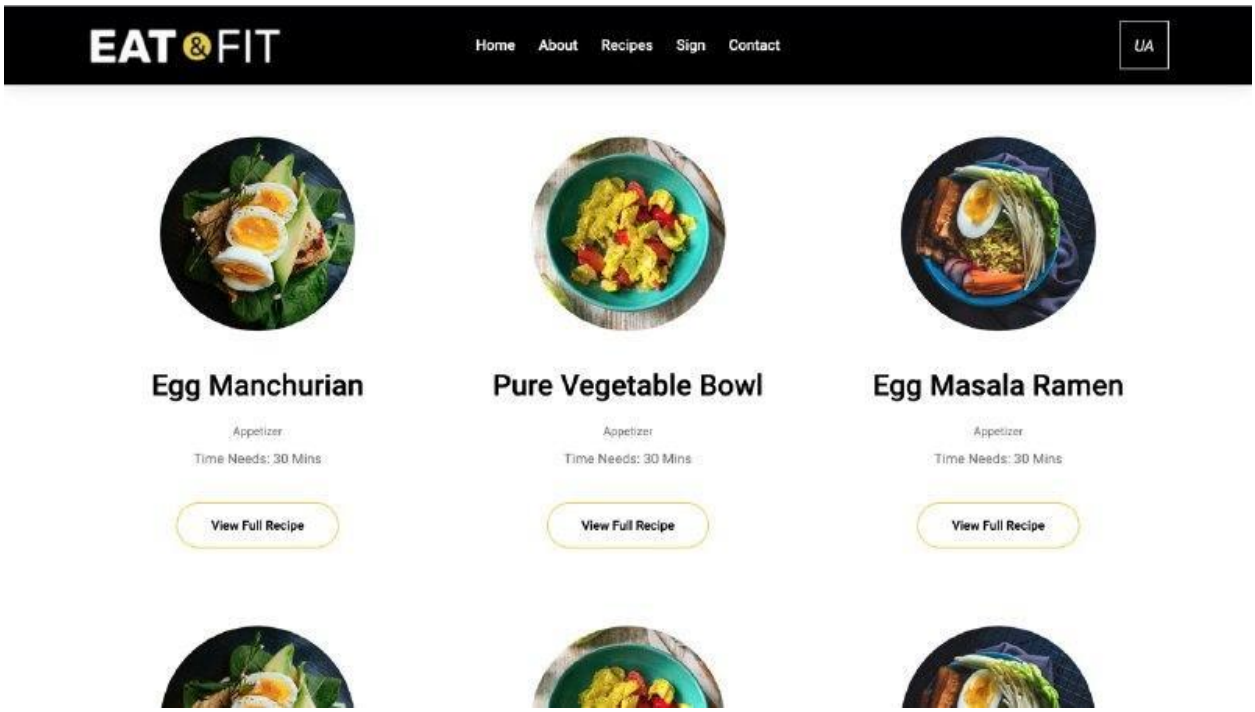


Рисунок 12. Сторінка доступних рецептів

Якщо користувач зареєстрований як адміністратор, то для нього відкриті додаткові можливості. Маючи права доступу він може додавати новий продукт або страву до списку загального користування або відкласти цей процес, відкривши доступ при редагуванні (див. рис. 13).

Будь-який продукт або страву можна змінити або видалити (див. рис. 14).

Також для зручного користування введений пошук даних (див. рис. 15).

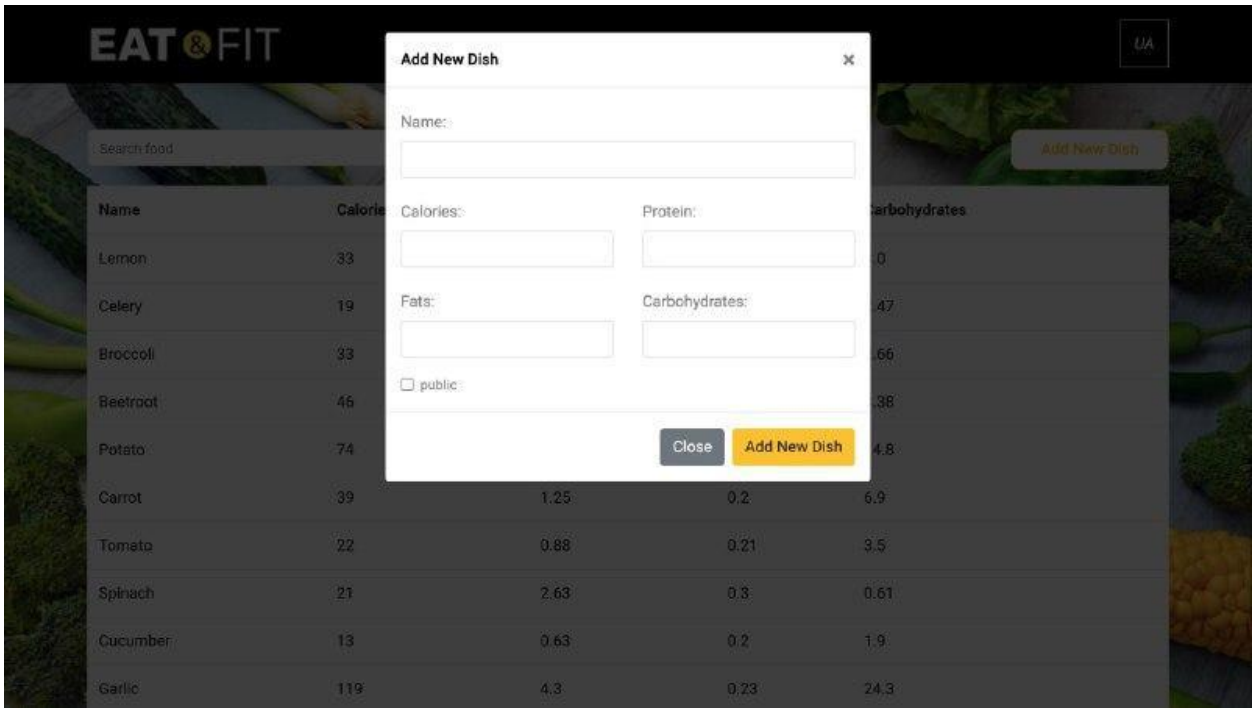


Рисунок 13. Додавання нового продукту/страви

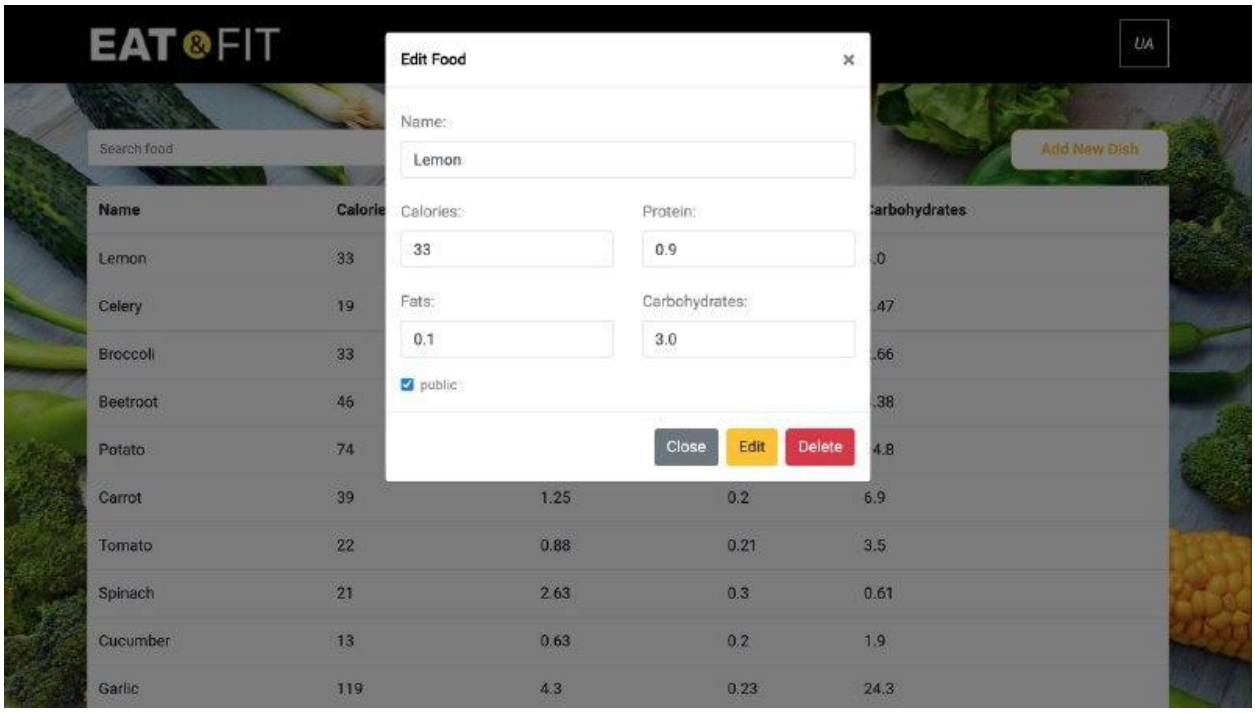


Рисунок 14. Редагування продукту/страви

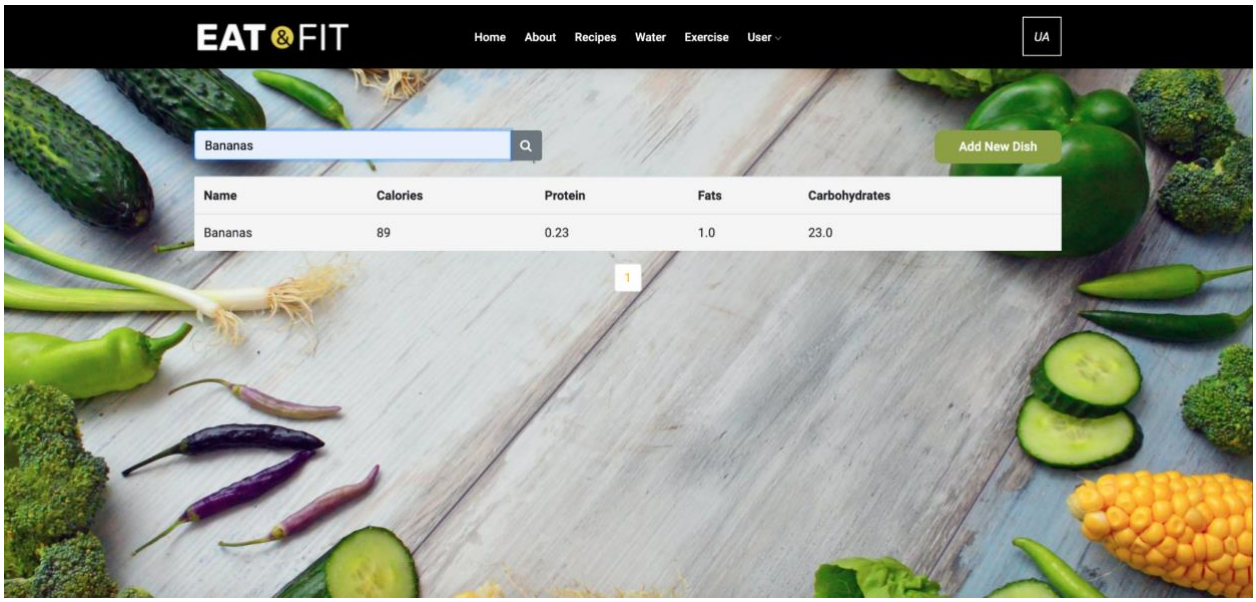


Рисунок 15. Пошук певної страви

Також з будь-яких питань людина може звернутися за всіма контактами на сторінці (див. рис. 16).

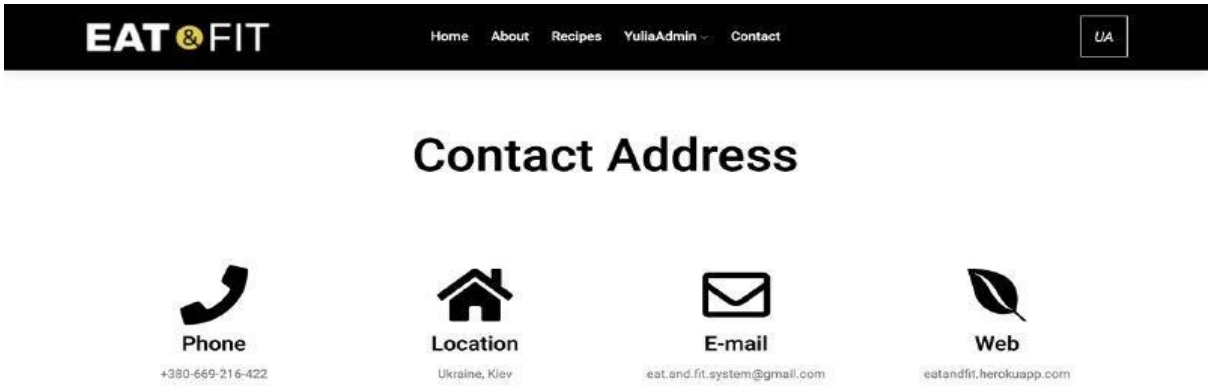


Рисунок 16. Контактна інформація

ВИСНОВКИ

В даній роботі було розглянуто основні принципи та засоби розробки вебзастосунків в цілому. Проаналізовано застосування різних технологій для проєктування та реалізації вебсайту.

В роботі досліджено існуючі системи підтримки здорового способу життя та проведено їх порівняння. В ході виконання дипломної роботи розроблено технічне завдання до продукту.

Під час виконання кваліфікаційної роботи на здобуття ступеня бакалавра було виконано поставлені задачі, а саме: досліджено застосунки, призначені для ведення здорового способу життя, досліджено застосування різних технологій для проєктування та реалізації вебсайтів, розроблено технічне завдання, інтерфейс та дизайн вебсайту «Eat&Fit», розроблено вебпроєкт для підрахунку калорій, поживних речовин, водного балансу та ведення щоденника активного образу життя у вигляді клієнт-серверного вебзастосунку. Для клієнтської частини були обрані JSP – для відображення вмісту сторінок, Java Script для реалізації різних функцій та його бібліотеки для формування сторінок. Для серверної частини була обрана Java, як основна серверна мова. Отримана реалізація була протестована на різних вхідних даних.

В подальшому слід зосередити свою увагу на збільшенні та покращенні функціоналу даного вебзастосунку. Також дана розробка може бути імплементована в певний сервіс для розширення його функціоналу.

На момент написання роботи програма розташовується в GitHub репозиторії і має відкритий код для можливості покращення її сторонніми розробниками, а також розміщена на стандартному хостингу Heroku, який є зручним і безкоштовним, проте з певними обмеженнями в користуванні.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Виноградов П. А. Фізична культура і здоровий образ життя / П. А. Виноградов. – М. : Мысль, 2001. – 287 с.
2. Зачем вести дневник питания. – Режим доступу до ресурсу: https://www.buro247.ru/beauty/health/14-aug-2018-why-keep-a-food-diary.html.
3. JetBrains [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com.
4. Блинов, И.Н. Java. Промышленное программирование : практ. пособие / И.Н. Блинов, В.С. Романчик. –Минск : УниверсалПресс, 2007. –704 с.
5. My Fitness Pal [Електронний ресурс] – Режим доступу до ресурсу: https://www.myfitnesspal.com.
6. Fat Secret [Електронний ресурс] – Режим доступу до ресурсу: https://www.fatsecret.ru.
7. Бережи фігуру [Електронний ресурс] – Режим доступу до ресурсу: https://beregifiguru.ru.
8. Budi Kurniawan . Java for the Web with Servlets, JSP, and EJB: A Developer's Guide to J2EE Solutions / Budi Kurniawan : New Riders Publishing , April 12 2002 – 976 p.
9. Servlet API [Електронний ресурс] – Режим доступу до ресурсу: https://www.javatpoint.com/servlet-api.
10. George Reese. Database Programming with Jdbc & Java (Java (O'Reilly)) Second Edition. — O'Reilly Media, 2000. — 348 с.
11. JEE Servlet API [Електронний ресурс] – Режим доступу до ресурсу: https://javastudy.ru/interview/jee-servlet-api-questions.

12. Apache Tomcat Apache Tomcat [Електронний ресурс] – Режим доступу до ресурсу: https://ru.qaz.wiki/wiki/Apache_Tomcat.
13. PostgreSQL [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/PostgreSQL.
14. Э. Фримен, Э. Фримен. Изучаем HTML, XHTML и CSS = Head First HTML with CSS & XHTML. — П.: «Питер», 2010. — 656 с.
15. Ellis, B., Stylos, J. and Myers, B. The Factory Pattern in API Design: A Usability Evaluation. International Conference on Software Engineering, 2007.
16. Unit тестирование с JUnit. – Режим доступу до ресурсу: http://devcolibri.com/864.

ДОДАТКИ

Додаток А. Use-Case діаграма для застосунку

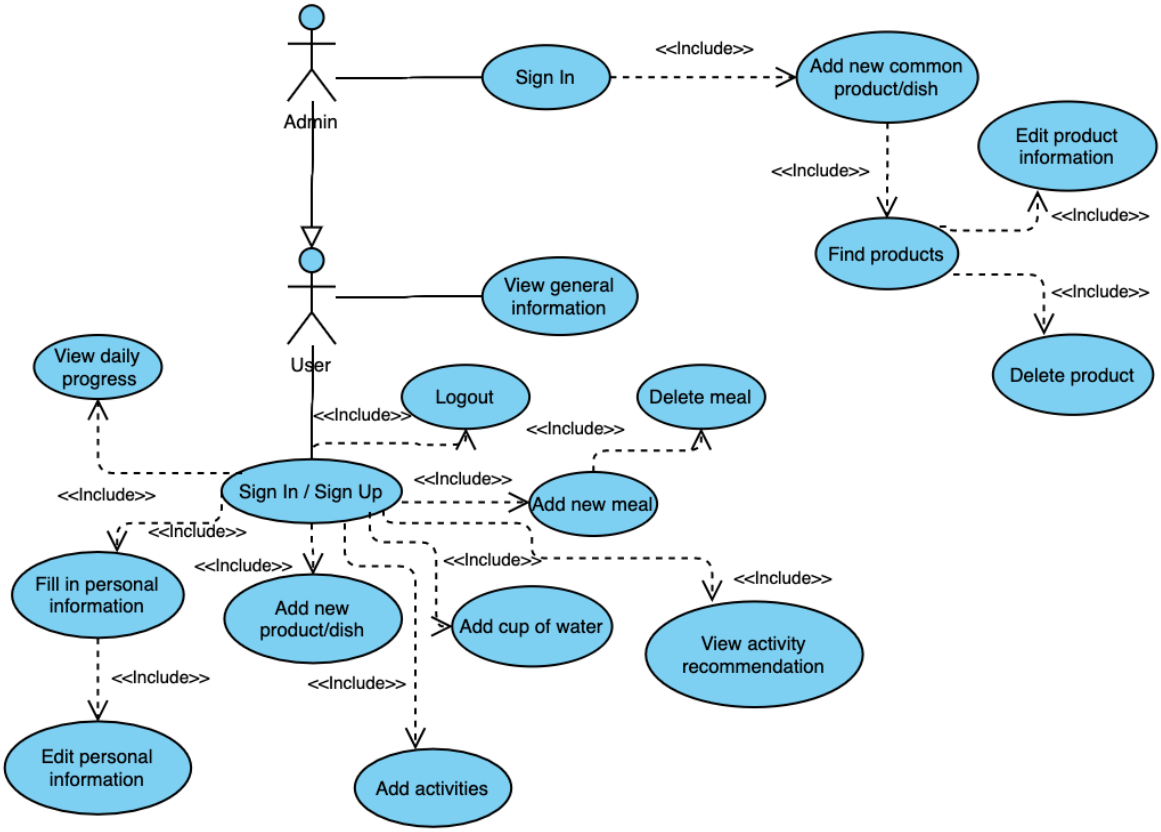


Рисунок А.1. Use-case діаграма програми «Eat&Fit»

Додаток Б. Діаграма бази даних застосунку

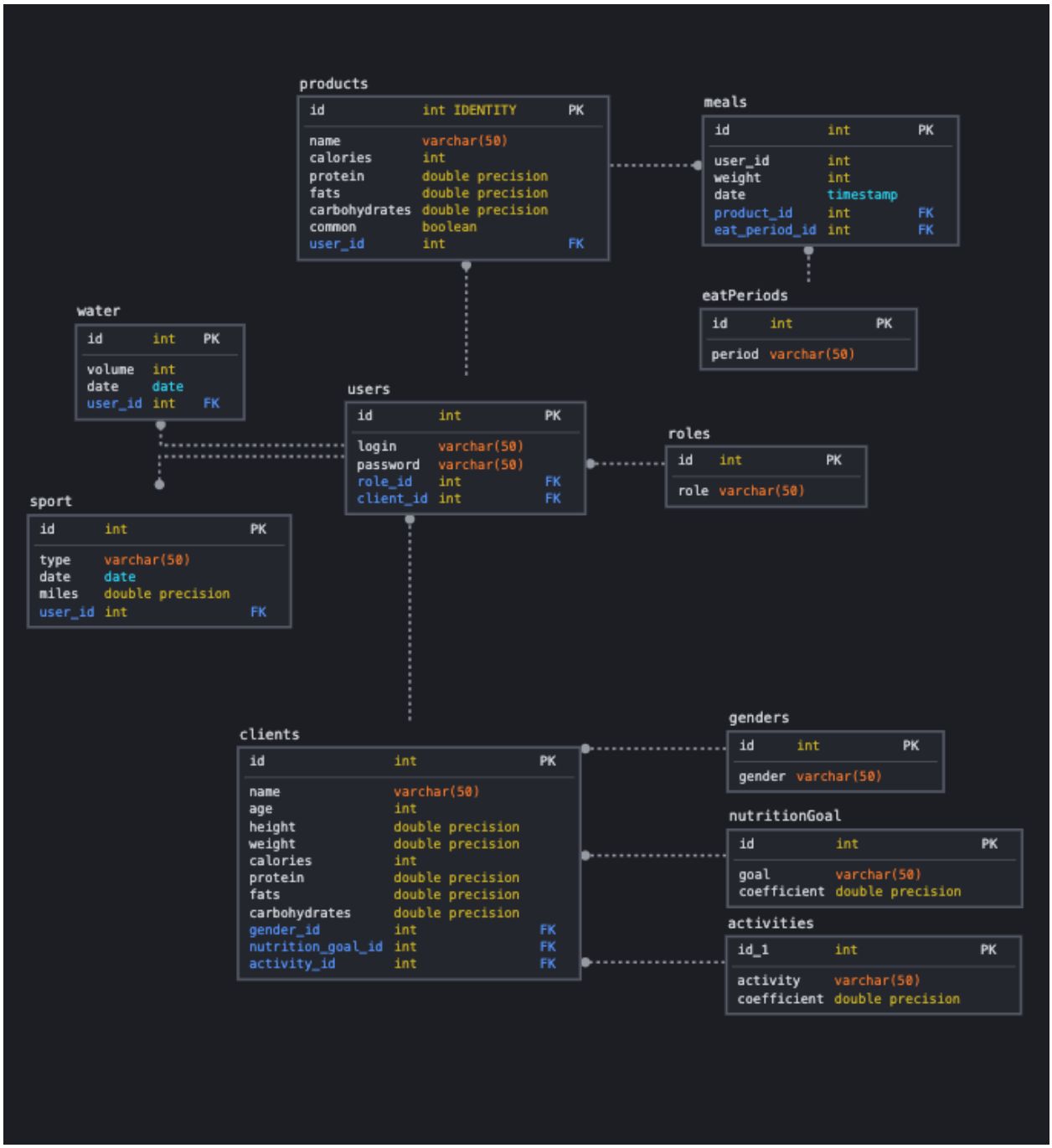


Рисунок Б.1. Діаграма бази даних «Eat&Fit»

Додаток В. Структура програми застосунку

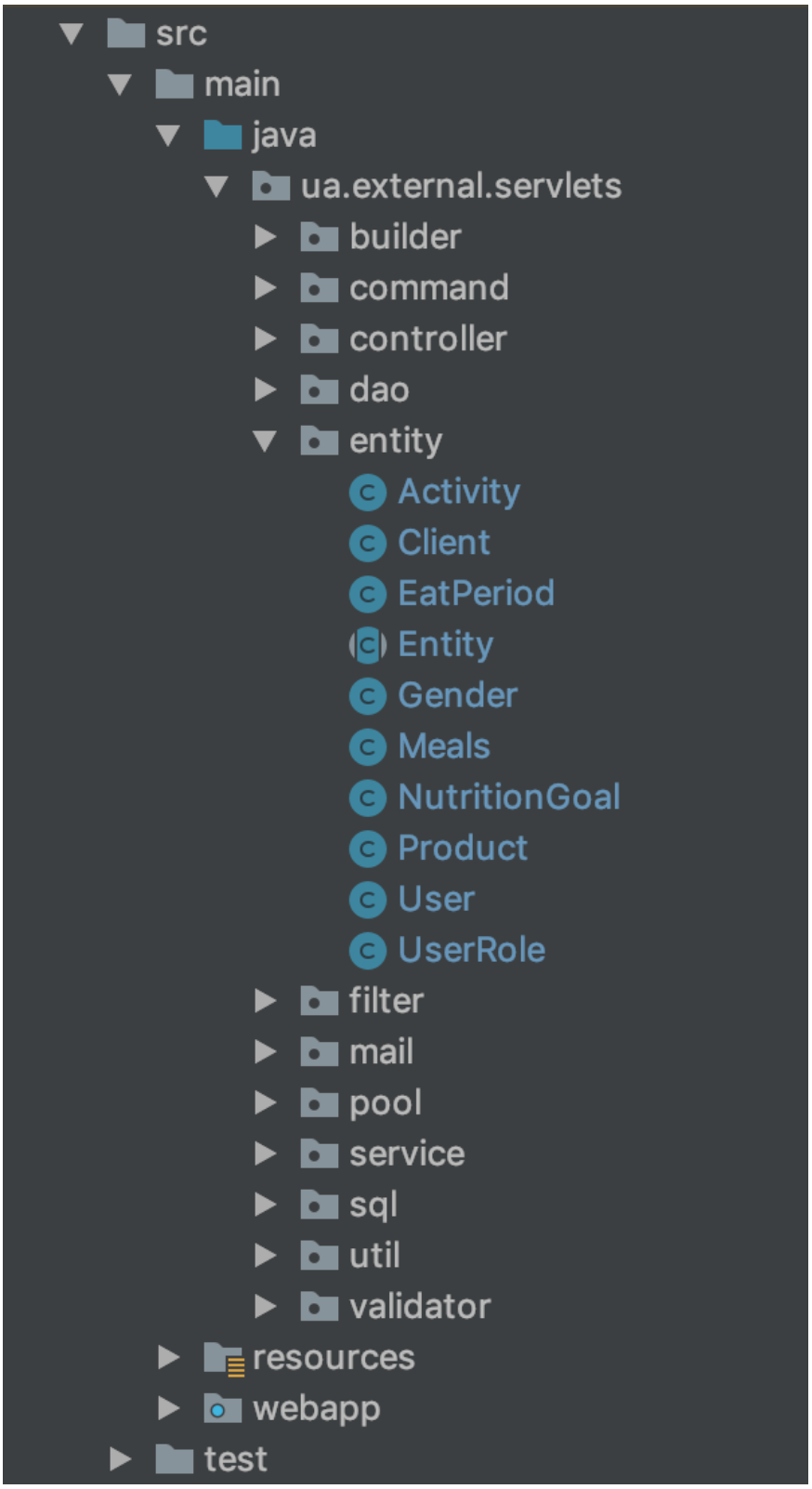


Рисунок В.1. Структура програми