Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Бакалаврська дипломна робота

на тему: «Розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React»

Виконав: студент 4 курсу
група 4ПІ-20б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Липовецький О.С.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. ПЗ Катєльніков Д.І.
(прізвище та ініціали)

Рецензент:. к.т.н., доц. каф ОТ Снігур А. В.
(прізвище та ініціали)

Допущено до захисту

Зав. кафедри ПЗ д.т.н., проф. Романюк О.Н.
«10» червня 2024 р.

ВНТУ – 2024

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
О. Н. Романюк
«12» березня 2024 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Липовецькому Олександру Сергійовичу

1. Тема роботи – розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React.

Керівник роботи: Катєльніков Денис Іванович, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від 11 березня 2024 р. № 80.

2. Строк подання студентом роботи 3.червня 2024.

3. Вихідні дані до роботи: тип програми – веб система; модель розробки – ітеративна; засоби організації даних програми – MySQL, Spring Data JPA; вихідні дані: відображення інформації запиту користувача; середовище розробки – IntelliJ Idea; редактор коду – VS Code; мова програмування – Java, JavaScript; використані бібліотеки та фреймворки – React JS, Spring (Boot, OpenAI, Data JPA, Liquibase).

4. Зміст текстової частини: вступ, обгрунтування вибору методу розробки та постановка задач дослідження; розробка програмного забезпечення; тестування програмного забезпечення; висновки; список використаних джерел; додатки.

5. Перелік ілюстративного матеріалу: титульний слайд; актуальність теми; мета, об'єкт та предмет дослідження; задачі дослідження; новизна і практична цінність одержаних результатів; порівняльний аналіз аналогів; структура застосунку; модель системи; загальний алгоритм роботи системи; розробка алгоритму обробки даних користувача та взаємодії з постачальником штучного інтелекту; розробка клієнтської частини; апробація та публікація матеріалів бакалаврської дипломної роботи.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1-4	Катєльніков Д.І. к.т.н., доцент кафедри ПЗ	13.03.24	03.06.24

7. Дата видачі завдання 13 березня 2024р

КАЛЕНДАРНИЙ ПЛАН

Ч.ч.	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз стану питання та постановка задач дослідження	14.03.24 – 17.03.24	Виконано
2	Розробка структури та алгоритмів роботи програмного продукту	17.03.24– 26.03.24	Виконано
3	Розробка загального алгоритму роботи застосунку	26.03.24– 01.04.24	Виконано
4	Аналіз та вибір засобів для реалізації програмної системи	01.04.24– 05.04.24	Виконано
5	Розробка програмного забезпечення	05.04.24– 17.05.24	Виконано
6	Тестування програмного застосунку	17.05.24– 25.05.24	Виконано
7	Оформлення матеріалів до захисту БДР	25.05.24– 30.05.24	Виконано

Студент _____ **О.С. Липовецький**
(підпис) (ініціали та прізвище)

Керівник бакалаврської дипломної роботи _____ **Д. І. Катєльніков**
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

УДК 004.42

Липовецький О.С. Розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React. : бакалаврська дипломна робота зі спеціальності 121 Інженерія програмного забезпечення, освітня програма – інженерія програмного забезпечення. Вінниця : ВНТУ, 2024. 98 с.

На укр. мові. Бібліогр. : 30 назв.; рис. : 21; табл. : 4.

У бакалаврській дипломній роботі було визначено доцільність, практичну цінність та мету розробки, яка полягає у покращені систем для аналізу харчової цінності продуктів, що додатково буде використовувати штучний інтелект для формування персоналізованих рекомендацій.

Запропоновано модифікацію алгоритму для визначення харчової цінності продуктів за рахунок впровадження штучного інтелекту. Розроблено алгоритм формування запитів та їх валідації.

Програмне забезпечення розроблено із використанням середовища розробки Intellij Idea та мови програмування Java, зокрема з використанням фреймворку Spring Boot. У результаті виконання бакалаврської дипломної роботи розроблено програмний веб застосунок, працездатність та коректність виконання якого було перевірено за допомогою впровадження модульного тестування, також було підготовлено інструкцію користувача та визначено системні вимоги.

Отримані в бакалаврській дипломній роботі результати можна використати для ефективного застосування у сфері медицини та фітнесу.

Ключові слова: харчова цінність, штучний інтелект, здорове харчування.

ABSTRACT

Lypovetskyi O.S. The Development of an online intelligent consultant web application for food value analysis using Java, Spring, JavaScript, and React technologies. Vinnytsia : VNTU, 2024. 98 p.

In Ukrainian language. Bibliobrapher: 30 titles; fig. : 21 tabl. : 4.

The bachelor's thesis determined the feasibility, practical value, and goal of the development, which is to improve systems for analyzing the nutritional value of food products, additionally using artificial intelligence to form personalized recommendations.

A modification of the algorithm for determining the nutritional value of food products was proposed by introducing artificial intelligence. An algorithm for forming requests and their validation was developed.

The software was developed using the Intellij Idea development environment and the Java programming language, in particular using the Spring Boot framework. As a result of the bachelor's thesis, a software web application was developed, the functionality and correctness of which were verified by implementing unit testing. A user manual was also prepared and system requirements were defined.

The results obtained in the bachelor's thesis can be used for effective application in the field of medicine and fitness.

Keywords: nutritional value, artificial intelligence, healthy eating.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. В сучасному світі, де зростає усвідомлення важливості здорового способу життя, розробка веб системи з елементами штучного інтелекту для аналізу харчової цінності продуктів набуває ключового значення. Це стає надзвичайно актуальним, оскільки спостерігається зростання захворювань, пов'язаних з неправильним харчуванням, що спонукає людей звертати більше уваги на склад та харчову цінність продуктів, які вони споживають.

Однією з основних причин актуальності таких систем є нестача достовірної інформації про харчову цінність продуктів. Інформація на етикетках може бути заплутаною, а часто навіть неправдивою, що ускладнює вибір здорових продуктів. Зокрема, в медіа часто поширюються необґрунтовані твердження щодо корисності чи шкідливості певних продуктів.

Крім того, існує потреба в персоналізованих рекомендаціях щодо харчування, оскільки кожна людина має унікальні потреби та цілі з точки зору здорового харчування. Така система дозволить кожному користувачеві отримати індивідуальні поради, відповідно до його фізіологічних характеристик та життєвого стилю.

Розвиток штучного інтелекту сприяє появі нових можливостей у цій області. Аналіз великих обсягів даних дозволяє виявляти закономірності та тенденції у споживанні продуктів, що в свою чергу допомагає в розробці більш точних та ефективних рекомендацій щодо харчування. Розробка такої системи має потенціал позитивно вплинути на життя людей. Вона допоможе покращити загальний стан здоров'я, зменшить ризик захворювань, пов'язаних з харчуванням, та зробить здорове харчування більш доступним для всіх, незалежно від їхнього рівня доступу до фахівців у галузі дієтології чи нутриціології.

Також важливо враховувати сучасний стан ринку додатків і веб систем, що займаються аналізом харчової цінності продуктів. Наразі існує багато

застосунків, що пропонують різноманітні функції, від сканування етикеток до створення персоналізованих дієт. Однак, не дивлячись на широкий вибір існуючих рішень, багато з них все ще мають певні обмеження щодо точності, гнучкості та індивідуального підходу.

Актуальність покращення існуючих систем полягає в недостатності точності аналізу харчових даних, що може вводити користувачів в оману. Це стосується як розрахунку калорійності та поживної цінності, так і рекомендацій щодо їжі, які часто базуються на загальних даних, а не на персоналізованих потребах окремої особи. Розвиток штучного інтелекту та машинного навчання може значно покращити ці аспекти, надаючи більш точні та налаштовані під конкретного користувача рекомендації.

Тому реалізація програмного забезпечення застосунку online консультанта з елементами штучного інтелекту для аналізу харчової цінності є актуальною розробкою.

Зв’язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно з планом виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Метою роботи є покращення якості харчування за рахунок застосування вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності продуктів, що додатково буде використовувати штучний інтелект для формування персоналізованих рекомендацій.

Відповідно до поставленої мети потрібно вирішити такі завдання:

- провести аналіз стану питання та методі розв’язання задачі;
- спроєктувати модель даних та структуру веб застосунку;
- розробити загальний алгоритм роботи застосунку;
- розробити алгоритм обробки даних з залученням штучного інтелекту;
- виконати програмну реалізацію головного екрану застосунку;
- запровадити взаємодію між клієнтською та серверною частинами застосунку;

- провести тестування програмного застосунку;

- розробити інструкцію користувача та описати системні вимоги для веб застосунку.

Об’єктом дослідження є процес розробки програмних модулів для реалізації засобів веб системи для аналізу харчової цінності продуктів, що базується на використанні штучного інтелекту.

Предметом дослідження є методи і засоби розробки програмного забезпечення для аналізу харчової цінності продуктів, що базується на використанні штучного інтелекту.

Методи дослідження. У процесі дослідження було використано такі методи: теорія людино-машинної взаємодії при розробці інтерфейсу, теорія баз даних для організації зберігання інформації, теорія розподілених систем для побудови мобільного інтелектуального застосунку.

Новизна отриманих результатів.

Подальшого розвитку отримав метод аналізу харчової цінності продуктів, у якому на відміну від існуючих методів аналізу, застосовується комбінована модель генеративного штучного інтелекту та спеціально розробленого алгоритму валідації запитів. Це дозволяє значно підвищити якість рекомендацій щодо харчування і здорового образу життя.

Подальшого розвитку отримав метод композиції персоналізованих рекомендацій, у якому на відміну від існуючих методів композиції рекомендацій, застосовується комплексний підхід, що дозволяє отримувати поради не тільки стосовно харчування, але й фізичної активності.

Практична цінність отриманих результатів. Практична цінність одержаних результатів полягає в тому, що на основі отриманих в бакалаврській дипломній роботі теоретичних положень запропоновано алгоритми та розроблено програмні засоби для аналізу харчової цінності продуктів.

Особистий внесок здобувача. Усі наукові результати, що викладені у бакалаврській дипломній роботі, отримано автором самостійно. У наукових працях, опублікованих у співавторстві, автору належать такі результати: ---

Апробація матеріалів бакалаврської дипломної роботи. Результати роботи доповідалися на:

- LII Молодь в науці: дослідження, проблеми, перспективи (Вінниця, 2024).

Публікації. За тематикою дослідження опубліковано 1 наукова праця у збірниках матеріалів конференцій.

1. ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДУ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 **Аналіз стану веб систем з використанням штучного інтелекту**

У сучасному світі штучний інтелект (ШІ) стрімко розвивається, проникаючи в різні сфери життя, включаючи веб-розробку. Розробка веб-систем з елементами ШІ набуває все більшої популярності, адже вона відкриває нові можливості для покращення користувацького досвіду, автоматизації процесів та аналізу даних.

Основними перевага ШІ у веб-системах є [1]:

- Покращення користувацького досвіду. ШІ може використовуватися для персоналізації контенту, надання рекомендацій, автоматизації завдань та створення більш інтерактивних інтерфейсів;

- Автоматизація процесів. ШІ може автоматизувати рутинні та повторювані завдання, звільняючи час для більш творчої та стратегічної роботи;

- Аналіз даних. ШІ може аналізувати великі обсяги даних, виявляти закономірності та робити прогнози, які допомагають приймати кращі рішення.

Тенденції розвитку веб-систем з елементами ШІ:

- Зростання популярності машинного навчання. Машинне навчання – це підгалузь ШІ, яка дозволяє комп'ютерам навчатися на даних без явного програмування. Завдяки машинному навчанню вебсистеми з елементами ШІ стають більш точними та ефективними;

- Інтеграція ШІ з іншими технологіями. ШІ все частіше інтегрується з іншими технологіями, такими як голосовий інтерфейс, обробка природної мови та доповнена реальність, що відкриває нові можливості для веб-розробки;

- Зростання доступності ШІ. Рішення ШІ стають все більш доступними для розробників, що робить розробку вебсистем з елементами ШІ доступнішою для ширшого кола компаній.

Отже розробка вебсистем з елементами ШІ – це динамічно зростаюча область, яка відкриває нові можливості для покращення користувацького

досвіду, автоматизації процесів та аналізу даних. Зважаючи на постійний розвиток ШІ, можна очікувати, що в найближчі роки вебсистеми з елементами ШІ стануть ще більш поширеними та досконалими.

1.2 Порівняльний аналіз аналогів

Веб-системи з елементами штучного інтелекту (ШІ) можуть відігравати значну роль у підтримці здорового способу життя та здорового харчування. Такі системи можуть пропонувати користувачам детальну інформацію про харчову цінність продуктів, а також рекомендації щодо їхнього вживання в контексті конкретних потреб і цілей. Головними критеріями виступають:

- Аналіз харчової цінності. Системи можуть аналізувати склад продуктів харчування та надавати користувачам інформацію про вміст калорій, білків, жирів, вуглеводів, вітамінів та мінералів у кожному продукті;

- Персоналізовані рекомендації. На основі інформації про користувача, такої як вік, стать, фізична активність та мета (наприклад, збереження ваги або збільшення м'язової маси), системи можуть надавати персоналізовані рекомендації щодо харчування;

- Моніторинг харчування. Користувачі можуть вести щоденник свого харчування, а система буде вести моніторинг їхнього раціону та надавати звіти про рівень вживання поживних речовин;

- Пошук альтернативних продуктів. Якщо користувач має певні дієтологічні обмеження або вподобання (наприклад, вегетаріанство, безглютенова дієта тощо), система може рекомендувати альтернативні продукти, які відповідають їхнім потребам;

- Навчання користувачів. Системи можуть надавати освітні матеріали щодо здорового харчування, вказувати на корисні властивості певних продуктів та негативний вплив інших.

На ринку існує кілька веб-систем, що використовують штучний інтелект для аналізу поживної цінності продуктів. Ось кілька найвідоміших із них:

1. MyFitnessPal (див. рис. 1.1) – це один із найпопулярніших застосунків для відстеження калорій і фізичної активності. MyFitnessPal дозволяє користувачам вести щоденник харчування та записувати фізичну активність, щоб вони могли відстежувати витрати калорій та спалювання їх під час тренувань. За допомогою функції сканування штрих-кодів користувачі можуть швидко додавати продукти до свого щоденника харчування та отримувати інформацію про їхню поживну цінність. Штучний інтелект використовується для аналізу даних про поживну цінність продуктів та надання користувачам персоналізованих рекомендацій щодо харчування, що допомагає досягти їхніх цілей щодо здоров'я та фітнесу. MyFitnessPal дозволяє користувачам створювати персоналізовані плани харчування на основі їхніх цілей щодо ваги, активності та інших факторів. Користувачі MyFitnessPal можуть приєднуватися до спільноти, де вони можуть ділитися своїм прогресом, отримувати поради та підтримку від інших користувачів [2].

Рисунок 1.1 – Зображення програмного продукту «MyFitnessPal»

2. Lose It! (див. рис. 1.2) – це чудовий приклад веб-системи, яка поєднує в собі функціонал додатка для відстеження калорій з елементами штучного

інтелекту для надання персоналізованих рекомендацій та планів харчування. Сканування штрих-кодів дозволяє користувачам швидко та зручно додавати продукти до свого щоденника харчування, просто скануючи штрих-код продукту. Штучний інтелект використовується для аналізу харчування користувача та надання рекомендацій щодо оптимального раціону на основі введених даних профілю та цільової ваги. На основі цільової ваги та інших особистих даних користувача програма генерує персоналізовані плани харчування, що допомагають досягти цілей щодо ваги та здоров'я. Користувачі можуть відстежувати свій прогрес за допомогою функцій відстеження ваги та активності, які допомагають зберігати мотивацію та досягати поставлених цілей[3].

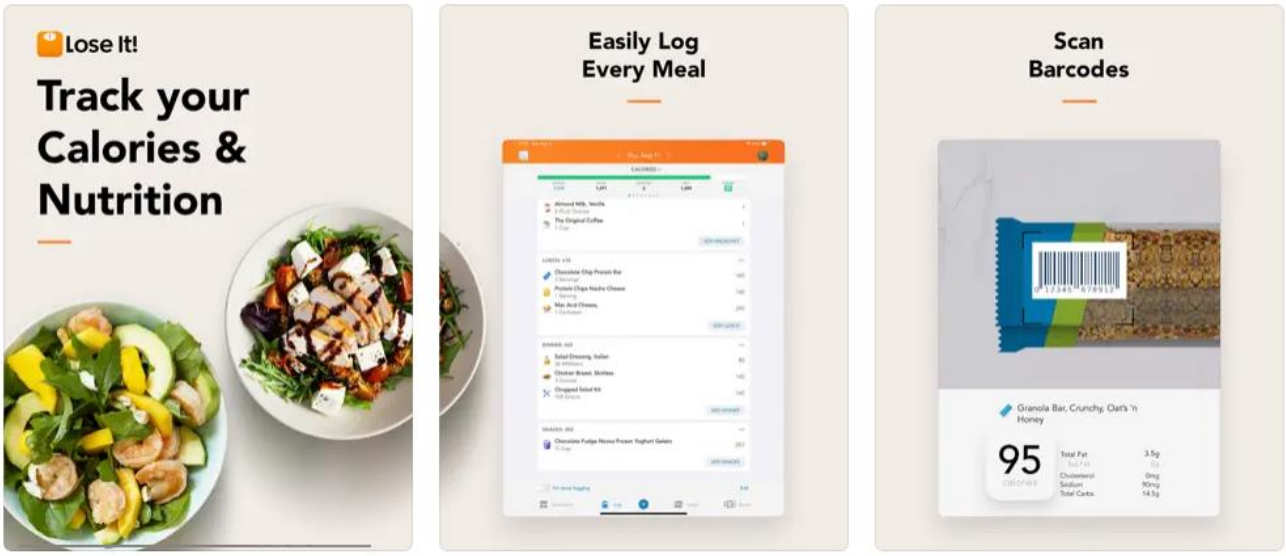


Рисунок 1.2 – Зображення програмного продукту «Lose It!»

3. FatSecret (див. рис. 1.3) – це ще один прекрасний приклад веб-системи та мобільного додатка, які допомагають користувачам вести журнал харчування, відстежувати калорії та отримувати інформацію про харчову цінність продуктів. FatSecret дозволяє користувачам легко вести журнал свого харчування, вносячи інформацію про спожиті продукти та порції. Система дозволяє користувачам відстежувати кількість спожитих калорій і порівнювати це зі своєю цільовою кількістю калорій на день. FatSecret надає користувачам детальну інформацію

про харчову цінність продуктів, включаючи вміст калорій, білків, жирів, вуглеводів та інших поживних речовин. За допомогою елементів штучного інтелекту система аналізує дані про харчування користувача та надає персоналізовані рекомендації щодо харчування, що допомагає досягти здоров'я та фітнесу. FatSecret також має функції спільноти, де користувачі можуть ділитися своїм прогресом, отримувати поради та підтримку від інших користувачів. FatSecret допомагає користувачам стежити за своїм харчуванням, отримувати необхідну інформацію про продукти та отримувати персоналізовані рекомендації для досягнення їхніх здорових цілей [4].

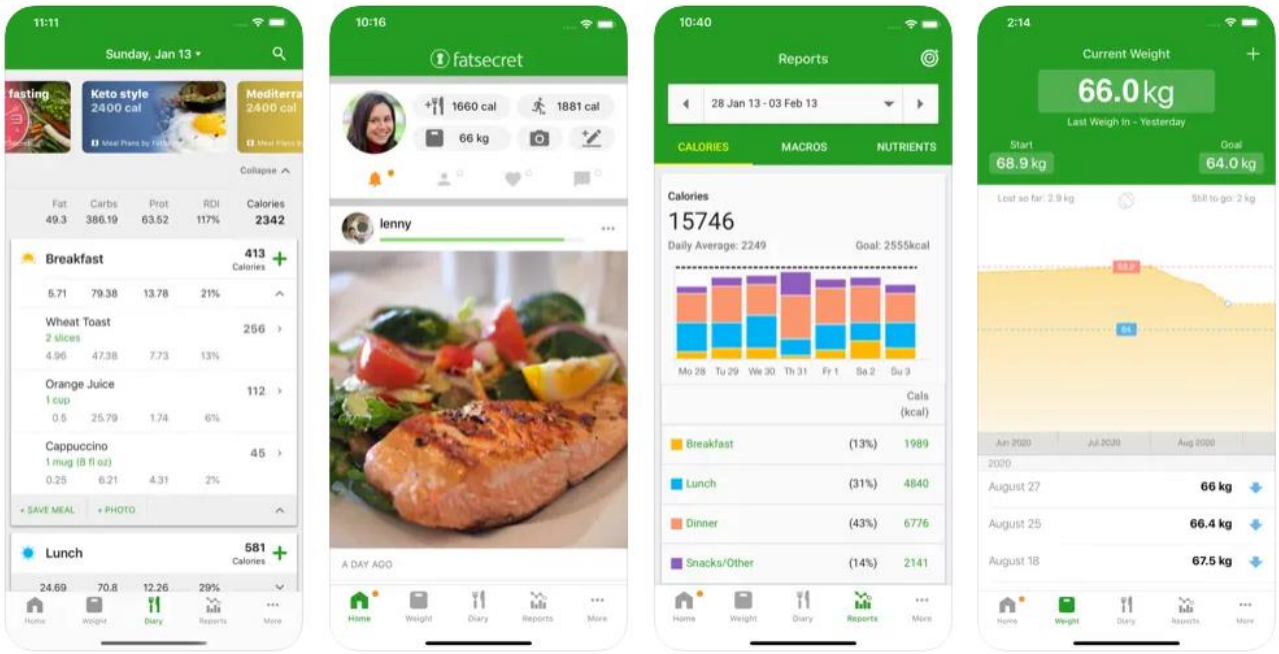


Рисунок 1.3 – Зображення програмного продукту «FatSecret»

4. Yuka (див. рис. 1.4) – це застосунок, який дозволяє користувачам сканувати штрих-коди продуктів харчування та отримувати оцінку їхньої харчової цінності. Користувачі можуть легко сканувати штрих-коди продуктів за допомогою камери свого смартфона, щоб отримати інформацію про їхню харчову цінність. Yuka використовує елементи штучного інтелекту для аналізу складу продуктів та надає користувачам оцінку їхньої харчової цінності на основі рівня поживних речовин, наявності добавок та інших факторів. На основі

аналізу даних про склад продуктів програма може надавати рекомендації щодо більш здорових альтернативних продуктів. Платформа може надавати користувачам освітні матеріали про поживні речовини, добавки та їх вплив на здоров'я, що допомагає зробити більш обізнаний вибір продуктів харчування. Крім того, користувачі можуть ділитися своїм досвідом та відгуками про продукти з іншими користувачами, створюючи активну спільноту здорового харчування [5].

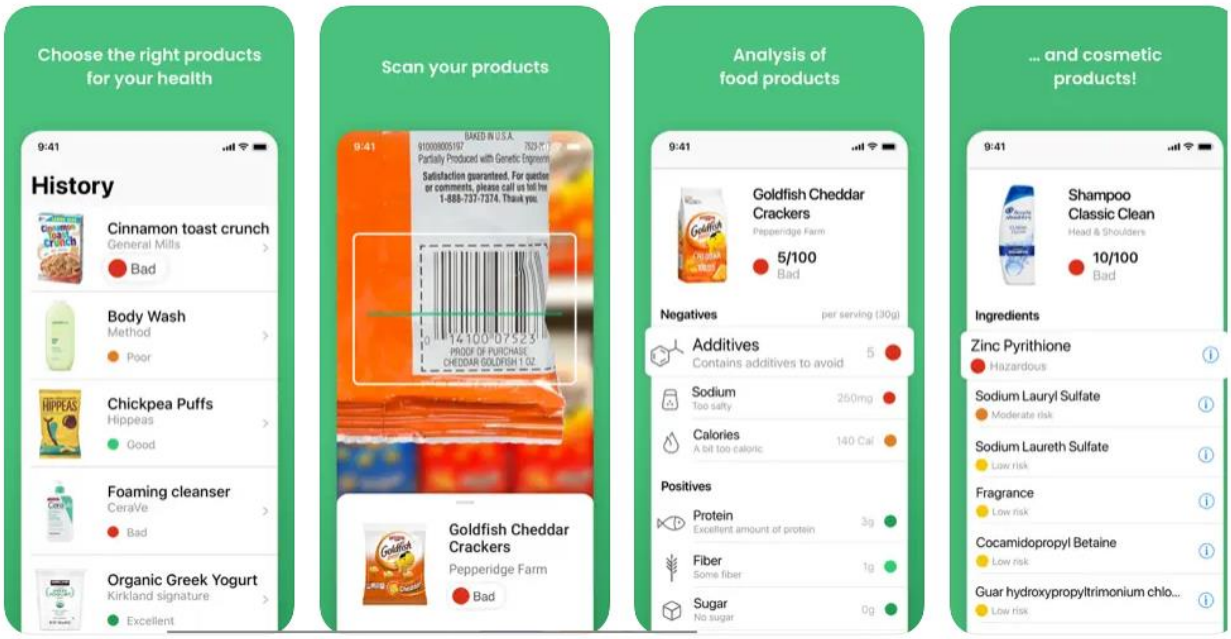


Рисунок 1.4 – Зображення програмного продукту «Yuka»

Основними перевагами розробки власної вебсистеми для аналізу харчової цінності продуктів є:

- Економічна вигода. Користувачам не потрібно буде платити за підписку або за доступ до будь-яких функцій, що робить вебсистему доступною для широкої аудиторії;

- Збільшення кількості користувачів. Безкоштовна вебсистема може залучити більше користувачів, що призведе до зростання популярності та авторитету;

- Персоналізація. Можливість персоналізувати вебсистему відповідно до потреб та вподобань своїх користувачів, що робить її більш корисною та зручною;

- Швидкість вдосконалення. Можливість швидко вносити зміни та вдосконалення до вебсистеми, не очікуючи схвалення від сторонніх розробників;

- Відкритий код. вебсистема з відкритим кодом дозволить іншим розробникам вносити свій вклад та покращувати її.

Для наглядного розуміння порівняльний аналізів усі дані було описано у таблиці 1.1.

Таблиця 1.1 – Порівняльний аналіз вебсистем

Назва	Функції	Переваги	Недоліки
MyFitnessPal	Сканування штрих-кодів, аналіз харчової цінності, персоналізовані рекомендації щодо харчування, відстеження калорій та фізичної активності	Велика база даних продуктів харчування, зручний інтерфейс, інтеграція з фітнес-трекерами	Може бути складним для нових користувачів, деякі функції доступні лише за підпискою
Lose It!	Сканування штрих-кодів, аналіз харчової цінності, персоналізовані плани харчування, відстеження прогресу	Проста у використанні, безкоштовна, інтеграція з фітнес-трекерами	Не така детальна, як інші програми, обмежена база даних продуктів харчування
FatSecret	Ведення журналу харчування, відстеження калорій, аналіз харчової цінності, персоналізовані рекомендації щодо харчування	Велика база даних продуктів харчування, безкоштовна, можливість створювати власні рецепти	Інтерфейс може здатися застарілим, деякі функції доступні лише за підпискою
Yuka	Сканування штрих-кодів, оцінка харчової цінності, рекомендації щодо здорових альтернатив	Швидкий та простий у використанні	Обмежена база даних продуктів харчування
Власна розробка	Аналіз харчової цінності, персоналізовані рекомендації щодо харчування, відстеження калорій та фізичної активності	Економічна вигода, збільшення кількості користувачів, контроль над розробкою, персоналізація, швидкість вдосконалення, відкритий код.	Новий та не протестований на великій кількості користувачів продукт

Згідно з таблицею порівняльних характеристик, бажано розробити унікальну систему з використанням штучного інтелекту. Отриманий продукт може закрити недоліки існуючих рішень та надати нові можливості для користувачів.

Тому розробка веб-інструментів з елементами штучного інтелекту для аналізу харчової цінності продуктів відкриває нові можливості для тих, хто хоче покращити власне здоров'я, створити унікальний користувацький досвід та популяризувати здоровий спосіб життя в суспільстві.

1.3 Аналіз напрямків удосконалення програмного забезпечення для аналізу цінності харчових продуктів

Дивлячись на результати аналізу можливостей існуючих веб-систем для аналізу цінності харчових продуктів, виділимо можливі напрямки їх удосконалення:

Розширення бази даних продуктів: Істотним напрямком удосконалення програмного забезпечення для аналізу цінності харчових продуктів є розширення бази даних продуктів. Це передбачає додавання не лише брендових, але й локальних та екологічно чистих продуктів, а також даних про харчову цінність страв та готових продуктів. Важливим доповненням стане можливість для користувачів самостійно додавати нові продукти та ділитися ними з іншими.

Покращення алгоритму аналізу: Для більш точного та всебічного аналізу цінності продуктів рекомендується вдосконалити алгоритм програми. Це може включати покращення розпізнавання штрих-кодів та назв продуктів, а також додавання можливості аналізувати склад продуктів за фотографією. Крім того, важливо розширити спектр факторів, які враховуються при аналізі, наприклад, етичне походження продуктів, їх вплив на довкілля та відповідність дієтам.

Персоналізація: З метою зробити програму більш корисною та зручною для користувачів, рекомендується розробити систему персональних рекомендацій продуктів. Ці рекомендації мають ґрунтуватися на харчових потребах, вподобаннях та цілях користувача. Додатковими функціями, які

сприятимуть персоналізації, стануть можливість створювати власні списки продуктів та ділитися ними з іншими, а також інтеграція програми з фітнес-трекерами та іншими програмами для здорового способу життя.

Додаткові функції: Для розширення функціоналу програми та підвищення її корисності можна запропонувати такі додаткові функції:

- можливість сканувати чеки та аналізувати витрати на продукти харчування;
- розробка рецептів та планів харчування на основі продуктів, доступних у коморі та холодильнику;
- додавання функції пошуку магазинів, де можна купити ті чи інші продукти.

Інтерфейс та зручність використання: Важливим фактором успіху будь-якого програмного забезпечення є його зручність використання. Для цього рекомендується зробити інтерфейс програми більш інтуїтивно зрозумілим та лаконічним. Додатковою зручністю стане можливість синхронізації даних між різними пристроями, а також розробка мобільного додатку для використання програми на ходу.

Навчання та просвітництво: З метою популяризації здорового харчування та підвищення обізнаності користувачів про продукти харчування, рекомендується додати до програми інформаційні статті та відео про цю тему. Додатково можна розробити систему гейміфікації, яка буде мотивувати користувачів дотримуватися здорових харчових звичок. Важливою складовою цього напрямку стане співпраця з дієтологами та іншими фахівцями з питань харчування для створення якісного та науково обґрунтованого контенту.

Як бачимо, функціонал веб-систем для аналізу цінності харчових продуктів має кілька шляхів для подальшого розвитку та удосконалення.

1.4 Аналіз методів розв’язання поставленої задачі

Розробка веб системи з елементами штучного інтелекту для аналізу поживної цінності продуктів харчування – складний процес, який вимагає

ретельного планування та вибору методів розробки. Існує кілька способів реалізації такої системи, кожен з яких має свої недоліки та переваги.

Одним з варіантів вирішення проблеми розробки веб систем з використанням штучного інтелекту є використання сторонніх AI API, основними перевагами такого підходу є [6]:

Швидкість розробки: за допомогою готових API можна швидко додати AI-функціонал до веб-системи.

- Простота впровадження. Не потрібно наймати команду експертів з машинного навчання або створювати власну AI-інфраструктуру;
- Масштабованість. Веб-систему легко розширити, щоб задовольнити зростаючий попит, використовуючи ресурси постачальника;
- Економічна ефективність. У деяких випадках використання API може бути вигіднішим, ніж розробка AI-рішення власними силами.

Але є й недоліки, серед яких:

- Залежність від постачальника. Залежність від постачальника API обмежує функціональність і гнучкість веб-системи.
- Ризики для конфіденційності. обробка даних відбувається на серверах постачальника, що становить ризик для конфіденційності користувачів.

Іншим підходом до вирішення проблеми розробки веб-системи є розробка власних моделей машинного навчання. Перевагами такого підходу є:

- Повний контроль. Ви повністю контролюєте функціональність, гнучкість і безпеку вашої веб системи;
- Персоналізація. ви можете розробити власні моделі машинного навчання, пристосовані до ваших потреб;
- Конфіденційність. Зберігаючи та обробляючи дані на власних серверах, ви краще контролюєте свою конфіденційність.

Але є й певні недоліки:

- Високі витрати. Розробка власного рішення для штучного інтелекту може коштувати дорого, оскільки потрібно найняти команду експертів з машинного навчання і побудувати власну інфраструктуру;

• Тривалий процес. розробка кастомного рішення зі штучного інтелекту займає багато часу і може затримати запуск веб-системи;

• Складнощі впровадження. Розробка унікального AI-рішення вимагає глибоких знань у галузі машинного навчання.

Оцінивши переваги та недоліки розглянутих методів, було вирішено використати API AI-провайдера для власної розробки. Це дозволяє заощадити власні кошти та використовувати перевірені технології, схвалені IT-гігантами. Також варто відзначити, що такий підхід дозволяє сфокусуватися на конкретних цілях розробки та вимогах користувачів.

1.5 Постановка задач для розробки веб-системи аналізу цінності харчових продуктів

Проаналізувавши переваги та недоліки існуючих веб систем, що використовують штучний інтелект, було визначено наступні завдання, які необхідно виконати для розробки веб-системи аналізу харчової цінності продуктів:

- здійснити аналіз стану питання та методів розв’язання задачі;
- спроектувати модель даних та структуру мобільного застосунку;
- розробити загальний алгоритм роботи програми;
- визначити провайдера послуг штучного інтелекту;
- розробити алгоритм взаємодії користувача із системою;
- розробити зручний інтерфейс користувача;
- описати запити для взаємодії із власним API;
- провести тестування веб-системи згідно поставлених задач.

Технічне завдання на розробку наведено в додатку А.

1.6 Висновки

У розділі було розглянуто аналіз стану веб-систем з використанням штучного інтелекту. Було проведено порівняльний аналіз відомих програмних продуктів у цій сфері, таких як: MyFitnessPal, Lose It!, FatSecret та Yuka. Також

побудовано можливі напрямки удосконалення веб-систем аналізу харчової цінності продуктів.

Було розглянуто способи створення таких систем з використанням штучного інтелекту в цілому. У результаті порівняння було визначено актуальність розробки власної веб-системи та доцільність її вдосконалення. На основі отриманої інформації було сформовано перелік задач, які необхідно виконати для розробки власної веб системи для аналізу харчової цінності продуктів.

2. РОЗРОБКА СТРУКТУРИ ТА АЛГОРИТМІВ РОБОТИ ПРОГРАМНОГО ПРОДУКТУ

2.1 Аналіз вхідних та вихідних даних програмного застосунку

Вхідні та вихідні дані програмного забезпечення — це інформація, що обробляється програмою під час її виконання. Вони є ключовими елементами будь-якого програмного забезпечення і визначають його функціональність та взаємодію з користувачем або іншими системами.

Вхідні дані – це інформація, яку веб застосунок отримує від користувача або з інших джерел для подальшої обробки. У випадку веб застосунку для аналізу харчової цінності продукту, вхідними даними можуть бути: назва продукту, його бренд, тип, вага, а також особисті дані користувача, такі як вік, стать, рівень активності, цілі щодо харчування та харчові уподобання.

Вихідні дані – це результат обробки вхідних даних веб застосунком. Для веб застосунку з аналізу харчової цінності продукту, вихідними даними може бути інформація про калорійність продукту, його макро- та мікронутрієнти, глікемічний індекс, вміст холестерину та інші показники. Також, як вихідні дані програмного застосунку є персоналізовані рекомендації користувача на основі його списку продуктів.

Вхідні дані користувача зберігаються у базі даних для подальшого формування рекомендацій та можливості перегляду історії запитів.

Online консультант може надавати детальну інформацію про продукт, що включатиме такі дані як: кількість білків на 100 грам продукту, кількість жирів на 100 грам продукту, кількість вуглеводів на 100 грам продукту, загальні поради по харчуванню або фізичним навантаженням. Такий підхід у формуванні вихідних даних, що дозволяє отримати усю необхідну інформацію (БЖД) та додаткові персоналізовані поради користувачу може значно покращити досвід використання програмного застосунку та і ефективність роботи консультанту в загальному. Також варто зазначити, що за допомогою використання історії

запитів користувача, програмний застосунок може надавати комплексні поради базуючись на історії.

2.2 Розробка структури програмного застосунку

Розробка структури веб системи аналізу харчової цінності продуктів передбачає планування та організацію всіх функціональних компонентів та їх взаємозв'язків для забезпечення ефективного аналізу та надання користувачеві необхідної інформації. Структура повинна включати всі необхідні функції, які дозволять користувачеві вводити дані про продукти, отримувати інформацію про їх харчову цінність, відстежувати споживання калорій та нутрієнтів, а також налаштовувати параметри системи під свої потреби.

Для візуального представлення структури веб системи та взаємозв'язків між її компонентами було створено діаграму варіантів використання (рисунок 2.1). Вона відображає різні дії, які користувач може виконувати під час роботи, для прикладу це може бути:

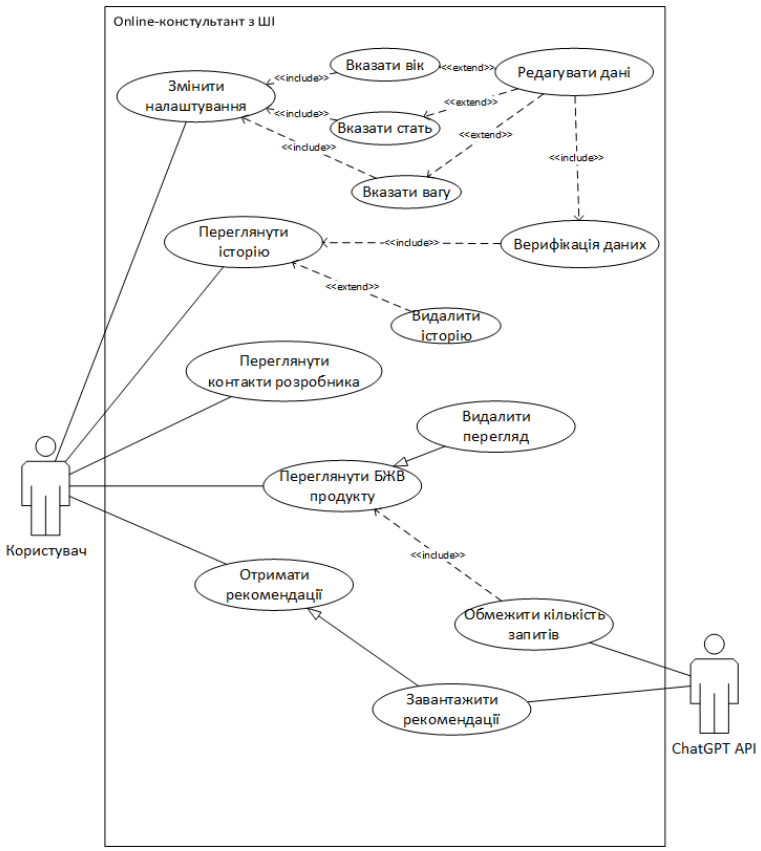


Рисунок 2.1 – Зображення діаграми варіантів використання

2.3 Розробка моделі системи

Модель веб застосунку online консультування для аналізу харчової цінності продуктів складається із двох компонентів: клієнтська та серверна частина. У свою чергу клієнтська частина складається із інтерфейсу користувача, а серверна частина складається із бази даних (MySQL) та API-інтеграції OpenAI програмним продуктом «ChatGpt». Ці компоненти взаємодіють за допомогою HTTP запитів (синхронне з'єднання) та забезпечують безперебійне та ефективне виконання програми.

Клієнтська частина відповідає за шар програмного застосунку, що на пряму взаємодіє із користувачем за рахунок інтерфейсу користувача, забезпечуючи зручний доступ до функціональності веб-застосунку. Інтерфейс користувача дозволяє користувачам вводити дані про продукти, які вони бажають проаналізувати, і отримувати інформацію про їх харчову цінність у зручному форматі.

Серверна частина, яка складається з бази даних MySQL та API-інтеграції з ChatGPT, виконує основні обчислювальні та аналітичні завдання. База даних зберігає інформацію про продукти, їх харчову цінність та інші необхідні дані. API ChatGPT відповідає за обробку запитів користувачів та надання інтелектуальних відповідей, базуючись на введених даних.

Коли користувач вводить дані про продукт в інтерфейсі користувача, клієнтська частина відправляє HTTP-запит до серверної частини. Серверна частина отримує запит, звертається до бази даних для отримання необхідної інформації та використовує ChatGPT для аналізу та формування відповіді. Потім відповідь повертається клієнтській частині, яка відображає результати аналізу користувачу.

Таким чином, клієнтська та серверна частини веб-застосунку взаємодіють у тісній координації, забезпечуючи ефективний аналіз харчової цінності продуктів та надаючи користувачам точну та корисну інформацію у зручному форматі.

Спрощену модель системи розроблюваного застосунку зображено на рисунку 2.2.

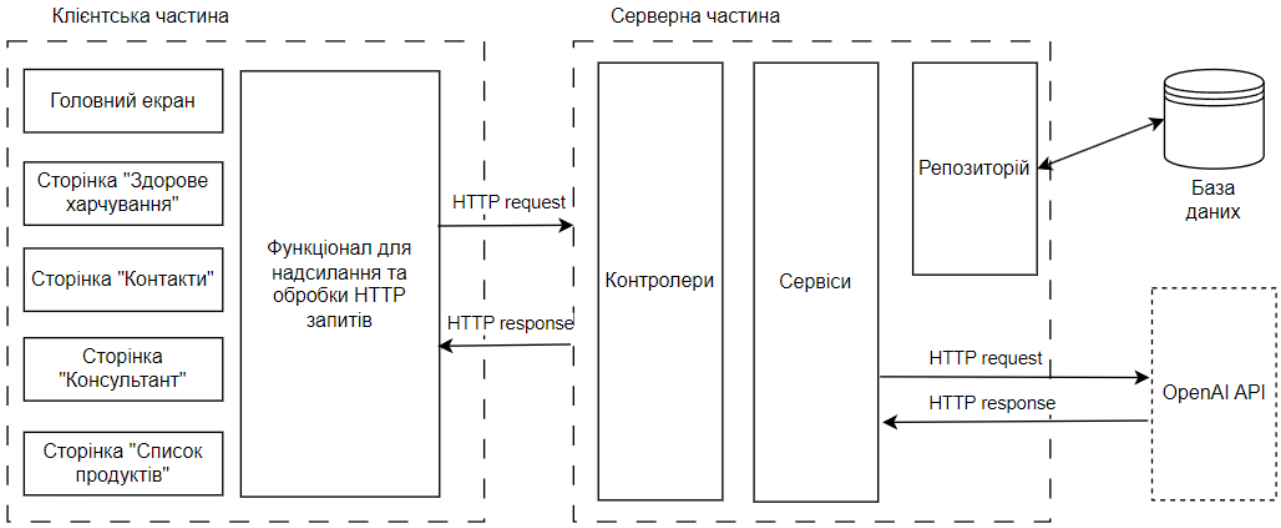


Рисунок 2.2 – Зображення спрощеної моделі системи

Для кращого розуміння моделі системи варто також описати такі терміни: контролер, сервіс та репозиторій. Контролер, сервіс та репозиторій – це ключові компоненти архітектури програмних застосунків, особливо у веб-розробці, що дотримуються принципів поділу відповідальності та шаруватої архітектури. Кожен з них виконує свою специфічну роль, забезпечуючи чітку структуру та логіку роботи застосунку [7].

Контролер (Controller):

• Основна функція: приймає запити від користувача (наприклад, через веб-інтерфейс), обробляє їх та повертає відповідь;

• Роль: виступає "вхідною точкою" застосунку, відповідає за валідацію вхідних даних, виклик відповідних сервісів для виконання бізнес-логіки та формування відповіді у зручному для користувача форматі (наприклад, HTML, JSON, XML).

Сервіс (Service):

• Основна функція: реалізує бізнес-логіку застосунку, виконуючи операції над даними та забезпечуючи взаємодію з репозиторіями;

• Роль: виконує складні операції, що вимагають доступу до даних, обробки даних з різних джерел, застосування бізнес-правил тощо.

Репозиторій (Repository):

• Основна функція: забезпечує доступ до даних, що зберігаються у базі даних або інших сховищах;

• Роль: абстрагує логіку роботи з базою даних, дозволяючи сервісам працювати з даними, не знаючи деталей їх зберігання. Це спрощує зміну бази даних або способу зберігання даних у майбутньому.

База даних для online консультування є важливо частиною серверної частини програмного застосунку оскільки вона відповідає за зберігання даних про користувачів, що дозволяє налагодити авторизацію та автентифікацію усіх користувацьких записів, також база даних відіграє одну із ключових ролей у оптимізації запитів та формування рекомендацій для користувачів, за рахунок того, що усі дані про продукти зберігаються у відповідну таблицю і використовуються в подальшому для формування комплексної поради стосовно покращення харчування користувача чи підвищення фізичного навантаження.

База даних для online консультування може бути реалізована за допомогою різних технологій та типів баз даних, таких як:

• Реляційні бази даних (RDBMS) – це найпоширеніший тип баз даних, де дані зберігаються у вигляді таблиць з визначеними зв'язками між ними [8]. Приклади: MySQL, PostgreSQL, Oracle, Microsoft SQL Server;

• Нереляційні бази даних (NoSQL) – це більш гнучкий тип баз даних, де дані можуть зберігатися у різних форматах (документи, ключ-значення, графи) [9]. Приклади: MongoDB, Cassandra, Redis;

• Об'єктно-орієнтовані бази даних (OODBMS) – це бази даних, де дані зберігаються у вигляді об'єктів. Приклади: db4o, ObjectStore [10].

У випадку розробки online консультування з аналізу харчової цінності, реляційні бази даних (RDBMS), зокрема MySQL, є найбільш доцільним вибором. Це пояснюється наступними перевагами:

- Структурованість даних. RDBMS забезпечують чітку структуру даних, що полегшує їх організацію, пошук та аналіз. Це особливо важливо при роботі з даними про продукти харчування, які мають багато атрибутів (назва, калорійність, склад тощо);
- Зв'язки між даними. RDBMS дозволяють встановлювати зв'язки між таблицями, що забезпечує цілісність даних та полегшує формування складних запитів. Наприклад, можна зв'язати таблицю продуктів з таблицею категорій, щоб швидко знаходити продукти певного типу;
- Мова SQL. RDBMS використовують мову SQL для роботи з даними, яка є стандартом у галузі та має велику спільноту розробників. Це полегшує пошук інформації та вирішення проблем;
- Зрілість та надійність. RDBMS є зрілою та перевіреною часом технологією, що забезпечує високу надійність та стабільність роботи;
- MySQL. Система управління базами даних MySQL є відкритим програмним забезпеченням, що робить її безкоштовною для використання. Вона також має велику спільноту користувачів та розробників, що забезпечує хорошу підтримку та документацію.

Використання MySQL як бази даних для online консультування дозволить створити надійну, масштабовану та ефективну систему, яка зможе обробляти великі обсяги даних та надавати користувачам точні та актуальні рекомендації щодо харчування.

Для реалізації бази даних, що відповідатиме усім вимогам розроблюваної системи, слід впровадити схему, яка визначатиме таблиці, стовпці та зв’язки між ними. Схема повинна відображати модель даних програмного застосунку і забезпечити логічну структуру для управління даними. На рисунку 2.3 зображено схему бази даних у вигляді UML-діаграми класів.

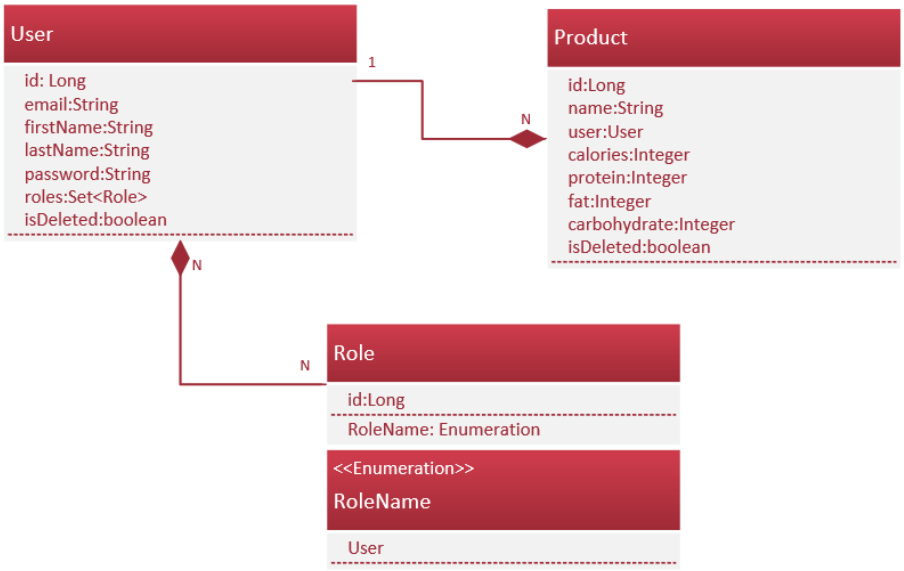


Рисунок 2.3 – Зображення діаграми класів

Класи даних, що зображені на діаграма, відображаються рівень абстракції між реалізацією серверної частини та базою даних, забезпечуючи уніфіковані моделі даних. Змінні, що містяться в цих класів, відповідно репрезентуються на записи БД.

2.4 Розробка загального алгоритму роботи програми

Опис загального алгоритму роботи програми є важливим етапом розробки, що приносить низку переваг. Він забезпечує розуміння логіки роботи програми, допомагаючи виявити потенційні проблеми та оптимізувати її ще на етапі проектування. Чіткий алгоритм полегшує розробку, тестування та налагодження програми, слугуючи "дорожньою картою" для розробників. Це також сприяє ефективній комунікації між членами команди та підвищує якість коду, роблячи його більш структурованим і читабельним. Крім того, опис алгоритму є важливою частиною документації, що допомагає іншим розробникам зрозуміти і використовувати програму. Він також дозволяє оцінити обчислювальну складність програми, що важливо для вибору оптимального рішення. Чітко описані алгоритми можна повторно використовувати для вирішення схожих задач, економлячи час і ресурси.

Для розробки веб системи з елементами штучного інтелекту для аналізу харчової цінності продуктів, необхідно розробити загальний алгоритм, згідно якого буде працювати веб система (див. рис. 2.4 та 2.5).

Згідно даного алгоритму, веб система консультанта налагоджує зв’язок із провайдером штучного інтелекту та отримує доступ до надсилання запитів, користувач у свою чергу за допомогою клієнтської частини (веб сайту) здатний надсилати бажану назву продукту для отримання інформації стосовно кількості білків, жирів, вуглеводів та загальної калорійності на 100 грам продукту. Сама ж серверна частина веб системи відповідальна за обробку клієнтських даних, після отримання запиту від користувача, сервер обробляє дані та формує запит та налагоджує зв’язок із провайдером ШІ, після формування запиту, система надсилає його на ресурс провайдера та очікує відповіді. Провайдер ШІ обробляє дані за вказаним алгоритмом, що також формується при підключенні у застосунку провайдера (у нашому випадку провайдером слугує OpenAI компанія, що займається розробкою та популяризацію штучного інтелекту в світі [11]).

Рисунок 2.4 – Зображення алгоритму роботи веб системи №1

Після обробки провайдером нашого запиту, сервіс отримує відповідь, що протягом подальших кроків проходить етапи фільтрації та формування інформативної відповіді (що використовується на клієнтській частині), після того, як серверна частина відправила відповідь на клієнтський застосунок, клієнтський застосунок (вебсайт) фільтрує та відображає усю необхідну інформацію у зазначеній формі «Детальна інформація» (див. рис. 2.5).

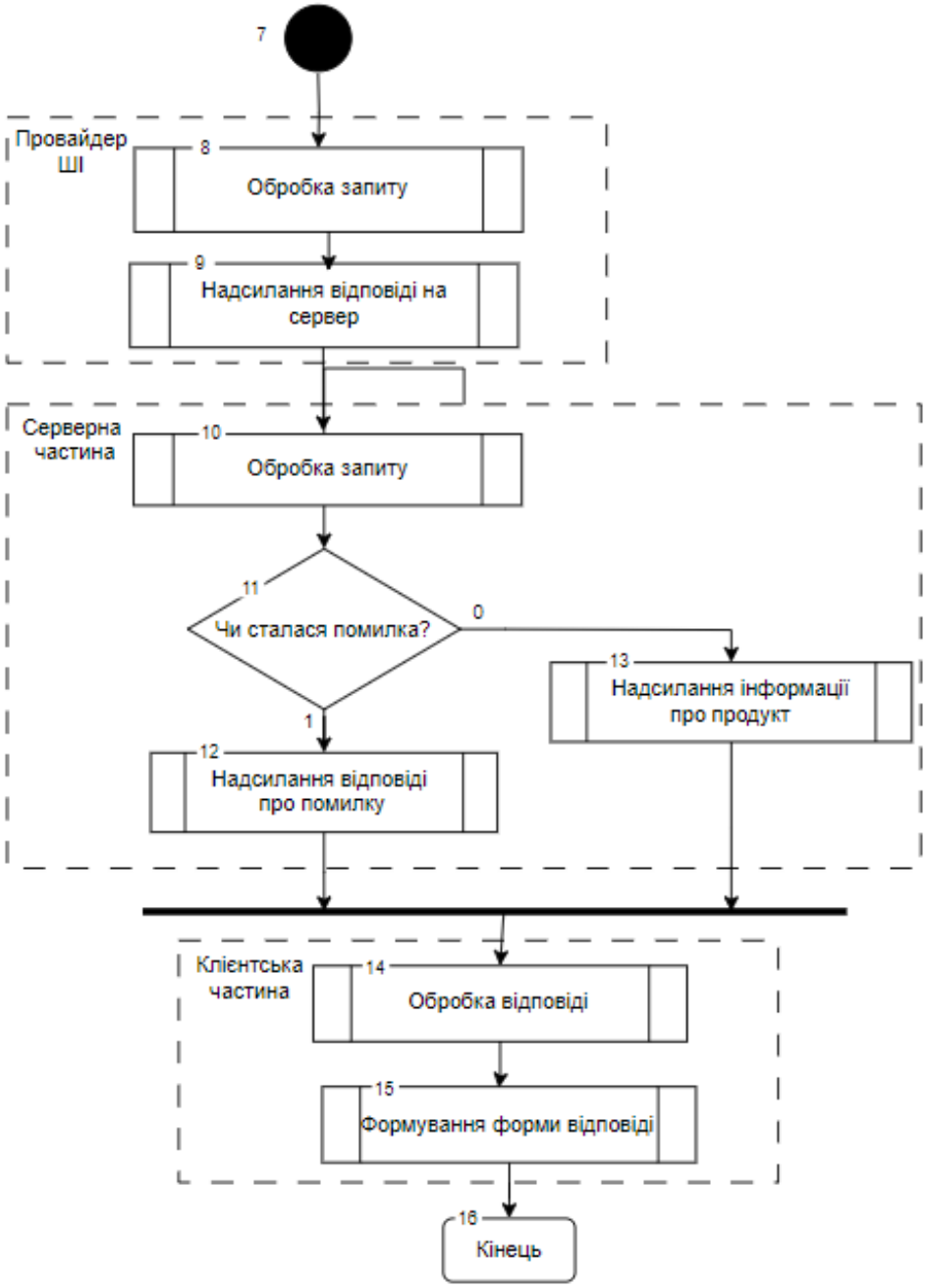


Рисунок 2.5 -Зображення алгоритму роботи веб системи №2

2.5 Розробка методу аналізу харчової цінності продуктів з використанням штучного інтелекту

Для створення ефективної веб системи аналізу харчової цінності продуктів необхідно обрати оптимальний ШІ-провайдер. Проаналізувавши Chat GPT та Gemini, виявилося, що обидва володіють потужними можливостями генерації тексту, перекладу та написання різноманітного контенту, що робить їх цінними інструментами для широкого спектру завдань [11]. Однак, Gemini спеціально розроблений для аналізу тексту та виявлення фактичної інформації, що робить його більш придатним для вилучення та обробки даних про харчову цінність з різноманітних джерел, таких як описи продуктів, етикетки та наукові статті.

Для порівняння було описано всі переваги та недоліки у вигляді таблиці 2.1.

Табл. 2.1. – Порівняння Gemini та ChatGPT

	Переваги	Недоліки
ChatGPT	Широкий спектр можливостей, включаючи генерування тексту, переклад мов та написання контенту. - Безкоштовний план доступний.	Не спеціально розроблений для аналізу харчової цінності продуктів. Може генерувати неточну або оманливу інформацію.
Gemini	Спеціально розроблений для аналізу тексту та виявлення фактичної інформації. Може витягувати інформацію про харчову цінність з текстів. - Доступний через Google Cloud Platform.	Платний план може бути дорогим для деяких користувачів. Не такий широкий спектр можливостей, як ChatGPT.

Незважаючи на переваги Gemini, для розробки даної вебсистеми було вирішено використати ChatGPT, враховуючи його широкий функціонал, гнучкість та наявність безкоштовного плану, що дозволяє оптимізувати витрати на проект. Для реалізації проекту буде застосовано ChatGPT API, що забезпечить інтеграцію потужностей штучного інтелекту у вебсистему, та фреймворк Spring

Boot, що забезпечить швидкість розробки, простоту використання, масштабованість, надійність та продуктивність системи.

Розробка програмних модулів для вебсистеми з використанням штучного інтелекту включає кілька ключових етапів. По-перше, необхідно налаштувати конфігураційні файли для забезпечення коректної взаємодії з провайдером ChatGPT API. По-друге, важливо визначити уніфікований формат запитів та відповідей для ефективної обробки даних системою. По-третє, необхідно розробити загальну структуру запитів, що враховує специфіку аналізу харчової цінності, та каркас веб системи, що забезпечить безперебійну взаємодію між клієнтською та серверною частинами.

Для успішної реалізації проекту необхідні глибокі знання Java, фреймворку Spring та принципів роботи з API. Крім того, важливим є використання ефективних середовищ розробки, таких як IntelliJ Idea та Visual Studio Code, що забезпечують зручний та продуктивний процес розробки.

Отже, створення веб системи для аналізу харчової цінності продуктів з використанням ШІ є комплексним та багатогранним завданням, що включає розробку back-end та front-end компонентів, інтеграцію ChatGPT API, підключення до бази даних, ретельне тестування та впровадження готового продукту. Успішна реалізація такого проекту відкриває широкі можливості для покращення якості харчування та інформування споживачів.

Основною задачею веб системи є подання дійсної та корисної інформації, що відповідає вимогам користувача. Обґрунтування переваг обраного провайдера штучного інтелекту було проведено у попередніх розділах. Процес взаємодії системи із постачальником ресурсів штучного інтелекту вимагає налаштування доступів та генерацію персонального ключа для ідентифікації ресурсу споживача (див. рис. 2.6).

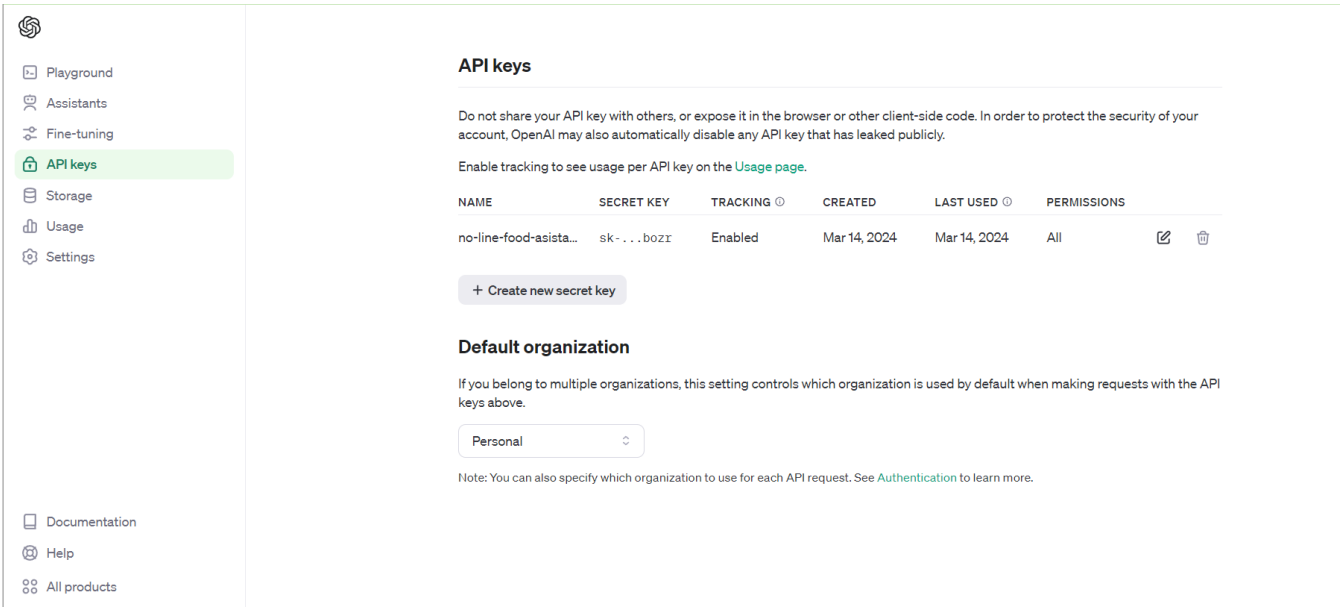


Рисунок 2.6 – Зображення панелі управління персональними ключами

Також, для кращого розуміння «налаштування штучного інтелекту» потрібно розглянути процес fine-tuning моделі ChatGPT.

Для покращення точності та релевантності відповідей для веб застосунку потрібно провести fine-tuning моделі ChatGPT.

Першим кроком було виконано збирання великого набору даних про продукти харчування, включаючи їх назви, калорійність, вміст білків, жирів та вуглеводів. Ці дані були отримані з різних авторитетних джерел, таких як бази даних продуктів, наукові статті та книги з дієтології.

Далі дані було ретельно структуровано у форматі "запит-відповідь", де запит містив назву продукту, а відповідь – детальну інформацію про його харчову цінність та корисну пораду. Цей набір даних був розділений на дві частини: навчальну (80%) та валідаційну (20%).

Наступним етапом було обрано базову модель ChatGPT, gpt-3.5-turbo, та запущено процес fine-tuning з використанням OpenAI API. Під час цього процесу модель навчалася розуміти специфіку запитів про харчову цінність та генерувати відповіді у потрібному форматі.

Завдяки fine-tuning, ChatGPT тепер здатний надавати точні та персоналізовані рекомендації щодо харчування, враховуючи індивідуальні потреби та цілі користувачів веб-застосунку.

Алгоритм взаємодії серверної частини та постачальника штучного інтелекту зображена на рисунку 2.7.

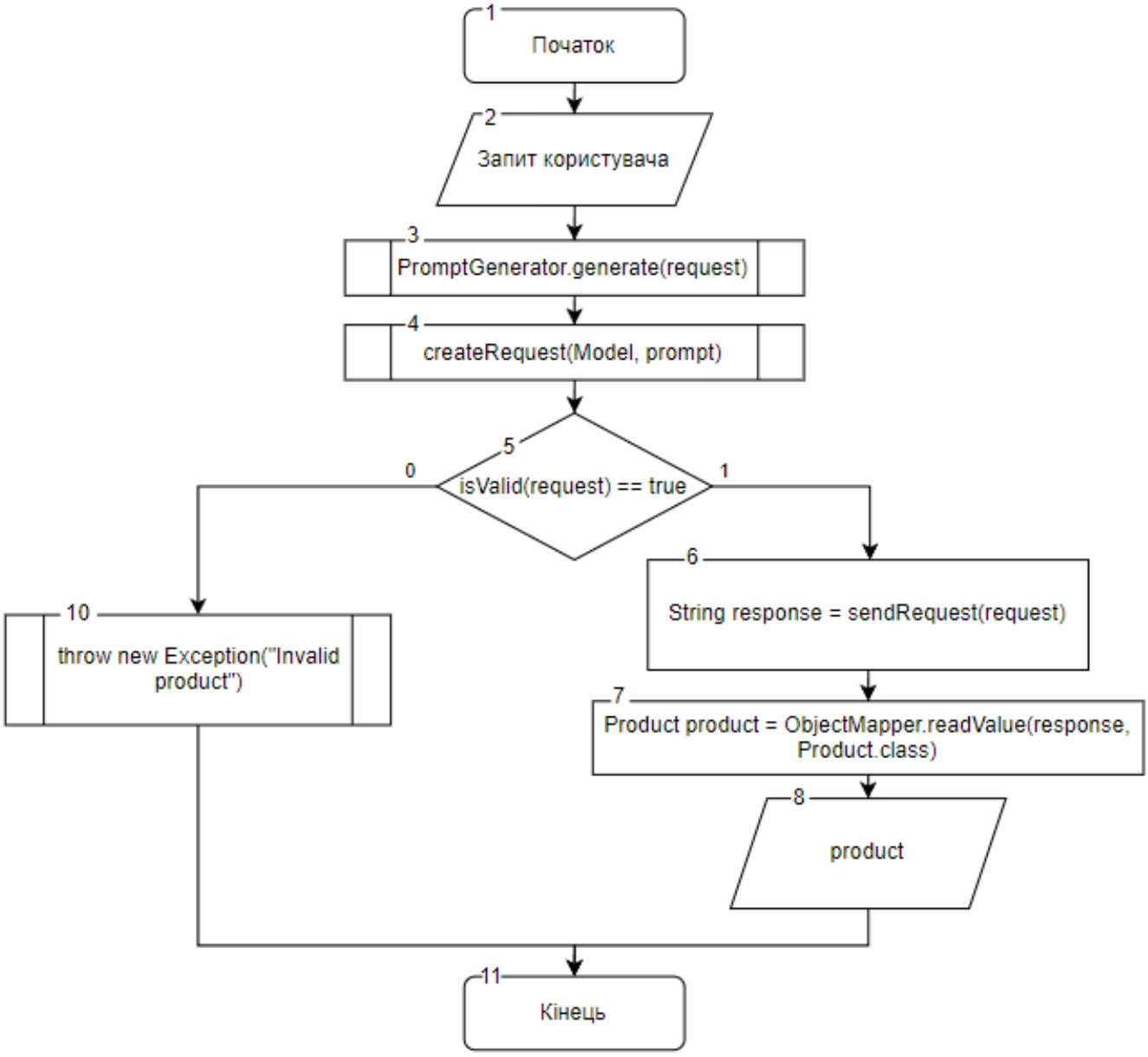


Рисунок 2.7 – Зображення алгоритму формування відповіді користувачу

1. Користувач заповнює форму на клієнтській частині та надсилає кнопку «Відправити».

2. Контролер на стороні серверної частини обробляє запит та передає його до сервісів для подальшої роботи над ним.

3. PromptGenerator.generate(request): система валідує отримані дані та перетворює їх відповідно до шаблону.

4. createRequest(Model, prompt): формується об’єкт запиту для подальшого надсилання.

5. isValid(request): перевірка наявності усіх необхідних полів об’єкта.

6. Throw new Exeption(“Invalid product”): формування помилки через некоректно введені дані (у випадку негативного результату пункту 5).

7. sendRequest(request): функція, що відповідає за надсилання запиту до OpenAI та валідації відповіді.

8. ObjectMapper.value(product, Product.class): форматування отриманого JSON об’єкту в необіхдний Product клас.

9. Сервіс передає об’єкт на контролер, який у свою чергу формує відповідь до клієнтської частини.

2.6 Висновки

У розділі було проаналізовано вхідні та вихідні дані програмного застосунку для online консультанта для обрахунку харчової цінності продуктів. Також було розроблено та описано: структуру програмного застосунку, модель системи, загальний алгоритм роботи програми, алгоритм обробки даних користувача та взаємодії з постачальником штучного інтелекту.

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення

Вибір мови програмування є критичним етапом при розробці веб застосунку online консультування з елементами штучного інтелекту для аналізу харчової цінності. На цьому етапі розробники повинні ретельно зважити різні фактори, які включають в себе продуктивність, швидкодію, масштабованість, безпеку та гнучкість.

Розробка веб застосунку онлайн консультування полягає у розробці двох частин, які можна поділити на серверну частину та клієнтську. Серверна частина застосунку відповідає за основний функціонал системи, керування даними та взаємодією із штучним інтелектом. У той час, як клієнтська частина – це частина, що полягає у відображенні графічної оболонки системи та представлення даних.

Для розробки серверної частини було проведено дослідження декількох мов програмування, таких як: Java, JavaScript, Python та Go.

Java вирізняється своєю високою продуктивністю та надійністю. Ця мова відома своєю широкою підтримкою бібліотек для веб-розробки та машинного навчання, а також має велику спільноту розробників. Це дозволяє легко масштабувати проекти та забезпечує значну підтримку у вирішенні різноманітних проблем [12].

JavaScript, будучи однією з основних мов для веб-розробки, має багато фреймворків, таких як Node.js та Express.js, які спрощують розробку серверної частини. Водночас, JavaScript може виявитися менш продуктивним у порівнянні з Java, що може бути критично для високонавантажених додатків [13].

Python займає особливе місце завдяки своїй простоті синтаксису, що робить його відмінним вибором для швидкої розробки та прототипування. Також він підтримує багато бібліотек для машинного навчання. Однак, його продуктивність та масштабованість можуть бути обмеженими для великих застосунків [14].

Go пропонує високу продуктивність і ефективність, а також простоту у паралельному програмуванні, але страждає від відносної нестачі бібліотек та меншої спільноти порівняно з Java або Python [15].

Порівняльний аналіз описаних мов програмування наведено в таблиці 3.1.

Таблиця 3.1. – Порівняння мов програмування

Критерій	Python	Java	JavaScript	Go
Тип	Інтерпретова-на, високорівнева	Компільова-на, високорівнева	Інтерпретова-на, високорівнева	Компільована, низькорівнева (з деякими високорівневими абстракціями)
Призначення	Загальне призначення, веб-розробка (бекенд), наукові обчислення, машинне навчання	Загальне призначення, веб-розробка (бекенд), мобільні додатки, великі корпоративні системи	Веб-розробка (фронтенд та бекенд), мобільні додатки	Системне програмування, мережеві сервіси, хмарні технології, DevOps
Парадигма	Об'єктно-орієнтована, функціональна, імперативна	Об'єктно-орієнтована	Об'єктно-орієнтована (прототипна), функціональна, імперативна	Процедурна, конкурентна, функціональна (частково)
Складність	Легко вивчається, простий синтаксис	Помірно складна, суворий синтаксис	Помірно складна, гнучкий синтаксис	Простий синтаксис, але вимагає розуміння конкурентного програмування
Продуктив-ність	Повільніша, ніж компільовані мови	Висока продуктивність	Повільніша, ніж компільовані мови, але швидка для веб-розробки	Висока продуктив-ність, розроблена для конкурентнос-ті
Популярність	Дуже популярна, велика спільнота	Дуже популярна, велика спільнота	Дуже популярна, незамінна для веб-розробки	Набирає популярність, особливо для хмарних технологій та DevOps

Враховуючи всі ці фактори, Java видається найбільш підходящою мовою для розробки серверної частини веб застосунку. Її здатність поєднувати високу продуктивність, надійність і масштабованість, а також наявність великої кількості бібліотек і спільноти.

Важливість обрання правильного середовища розробки не може бути переоцінена, оскільки воно безпосередньо впливає на продуктивність розробників, підтримку мов та технологій, якість роботи команди та кінцевий результат розробки. Правильно обране середовище розробки допомагає розробникам працювати більш ефективно, надаючи потужні інструменти для автоматизації рутинних задач, легкий доступ до документації та оптимізацію загальних процесів кодування.

Для розробки за допомогою мови Java розглянемо найпопулярніші середовища розробки, такі як: IntelliJ Idea, Eclipse та NetBeans;

- IntelliJ IDEA – це потужне середовище розробки, що надає широкий спектр можливостей для розробки на Java [16]. Його сильні сторони включають в себе глибоку інтеграцію з фреймворками Java, такими як Spring, Hibernate та інші. IntelliJ IDEA ефективно використовує контекстні підказки та рефакторинг коду, що значно спрощує розробку складних програмних рішень. Також IDEA має плагіни для підтримки фронтенд розробки, що є важливим для створення веб-застосунків.

- Eclipse – це одне з найбільш використовуваних середовищ розробки у світі Java, яке вирізняється своєю гнучкістю та масштабованістю [17]. Eclipse підтримує велику кількість плагінів, які можна встановити через Eclipse Marketplace, що дозволяє адаптувати середовище під різноманітні задачі проекту. Eclipse є відкритим продуктом, що дозволяє спільноті безперервно вносити в нього вдосконалення. Його також цінують за високу продуктивність при роботі з великими проектами.

- NetBeans – це середовище розробки також популярне серед розробників Java і має добре інтегровану систему управління проектами та вбудовані інструменти для графічного дизайну інтерфейсів користувача [18]. NetBeans відоме своїм зручним інтерфейсом та простотою використання, яке часто робить його вибором для освітніх проектів та для новачків. Також середовище надає вбудовану підтримку для роботи з серверами додатків і базами даних, що може бути корисним для розробки серверної частини веб-застосунків.

Для розробки серверної частини було обрано IntelliJ IDEA, оскільки це інтегроване середовище розробки має декілька переваг, які роблять його одним з найбільш популярних та ефективних середовищ розробки для Java-проектів:

1. Глибока інтеграція з фреймворками та інструментами. IntelliJ IDEA надає повну підтримку для широкого спектру фреймворків і технологій, що робить його ідеальним для будь-якого Java-проекту. Від Spring і Hibernate до Maven і Gradle, це середовище розробки забезпечує гладку інтеграцію з популярними інструментами розробки.

2. Зручний інтерфейс та продуктивність. IntelliJ IDEA відомий своєю зручністю використання та високою продуктивністю. Він надає широкий спектр інструментів, таких як автоматичне доповнення коду, рефакторинг, швидка навігація, що дозволяє розробникам працювати швидше та ефективніше.

3. Розширюваність і плагіни. IntelliJ IDEA має широкий вибір плагінів, які дозволяють налаштувати середовище розробки під власні потреби. З плагінами ви можете розширювати функціональність IDEA, додавати нові інструменти та функції, що робить його гнучким та адаптивним до будь-яких проектів.

4. Підтримка новітніх технологій. IntelliJ IDEA завжди оновлюється і підтримує новітні технології, що дозволяє розробникам використовувати останні тренди в розробці програмного забезпечення без проблем.

5. Спільнота та підтримка. IntelliJ IDEA має активну спільноту розробників, яка завжди готова допомогти вирішити будь-які проблеми та підтримати вас в процесі розробки.

Також, варто зазначити важливість вибору мови програмування для розробки клієнтської частини, оскільки цей фактор безпосередньо впливатиме на досвід користувача. Для порівняння та вибору технологій для клієнтської частини було досліджено наступні технології: HTML та CSS, React JS, Vue JS, Angular.

- HTML і CSS є фундаментальними технологіями для будь-якого веб-сайту, що забезпечують структуру та стилізацію веб-сторінок. Вони є

невід'ємною частиною розробки веб-інтерфейсів і забезпечують основу для реалізації користувацького інтерфейсу [19].

• React JS є популярною JavaScript-бібліотекою, створеною Facebook, яка дозволяє ефективно управляти станом інтерфейсу користувача в великих застосунках, що потребують частого оновлення даних. React JS забезпечує високу продуктивність завдяки віртуальному DOM, який оптимізує рендеринг компонентів [20].

• Vue JS, ще один популярний фреймворк, є вибором багатьох розробників через його простоту і гнучкість. Vue JS надзвичайно легко інтегрувати в існуючі проекти, що робить його ідеальним для поетапних міграцій або для розробки нових проектів [21].

• Angular, розроблений Google, є комплексним рішенням для розробки веб-додатків. Він надає багато інструментів і рішень "з коробки", включаючи розширену управління станом, модульність та міцні опції тестування, що робить його хорошим вибором для розробки великих масштабованих застосунків [22].

Порівняльний аналіз технологій розробки клієнтської частини наведено в таблиці 3.2.

Таблиця 3.2 – Порівняння технологій розробки клієнтської частини

Критерії	HTML + CSS	React JS	Vue JS	Angular
Простота вивчення	Висока (базові знання)	Середня (вимагає розуміння компонентного підходу)	Середня (легше ніж React, складніше ніж HTML+CSS)	Низька (складний фреймворк)
Швидкість розробки	Низька (ручне оновлення DOM)	Висока (віртуальний DOM)	Висока (віртуальний DOM)	Середня (повний фреймворк)
Продуктивність	Висока (мінімальний розмір)	Висока (віртуальний DOM)	Висока (віртуальний DOM)	Середня (більший розмір)
Масштабованість	Низька (складність підтримки великих проектів)	Висока (компонентна архітектура)	Висока (компонентна архітектура)	Висока (повний фреймворк)

Таблиця 3.3 – Продовження таблиці 3.2

Критерії	HTML + CSS	React JS	Vue JS	Angular
Гнучкість	Низька (базові можливості)	Висока (велика екосистема)	Висока (можливість вибору підходу)	Середня (висока структурованість)
Популярність	Висока (базова технологія)	Дуже висока (велика спільнота)	Висока (швидко зростаюча спільнота)	Висока (підтримка Google)
Сфери застосування	Прості сайти, лендінги	Комплексні веб-додатки, SPA, мобільні додатки (React Native)	Веб-додатки різної складності, SPA	Корпоративні додатки, великі проекти

Отже, React JS пропонує низку переваг, які роблять його привабливим вибором для розробки фронтенду веб-системи аналізу харчової цінності продуктів. Компонентний підхід дозволяє створювати модульний та легко підтримуваний код, що особливо важливо для складних систем з багатьма елементами інтерфейсу. Віртуальний DOM забезпечує високу продуктивність та швидкість роботи, що є критичним для додатків, які обробляють великі обсяги даних та виконують складні обчислення. JSX дозволяє писати декларативний код, що спрощує розуміння структури та логіки компонентів. Односторонній потік даних забезпечує передбачуваність та легкість налагодження додатку.

Велика екосистема React JS надає розробникам широкий вибір готових компонентів, бібліотек та інструментів, що значно прискорює процес розробки та дозволяє зосередитися на реалізації унікальної функціональності системи. Популярність React JS гарантує активну підтримку спільноти, регулярні оновлення та доступ до великої кількості навчальних матеріалів та документації.

Крім того, React Native дозволяє використовувати ті ж самі компоненти та навички для розробки мобільних додатків, що відкриває додаткові можливості для розширення функціональності системи та охоплення більшої аудиторії користувачів.

Також, було проаналізовано ряд редакторів коду для розробки клієнтської частини, таких як: VS Code, Sublime Text та Atom.

VS Code є безперечним лідером серед веб-розробників. Його величезна спільнота забезпечує постійний розвиток та підтримку, а величезна кількість розширень дозволяє налаштувати редактор під будь-які потреби. VS Code відрізняється високою продуктивністю, навіть при роботі з великими проектами, та інтуїтивно зрозумілим інтерфейсом. Він пропонує потужні інструменти для налагодження коду та вбудовану підтримку Git, що робить його чудовим вибором для командної роботи [23].

Sublime Text цінується досвідченими розробниками за його швидкість та гнучкість налаштування. Він дуже легкий та швидкий, що робить його ідеальним для слабших машин [24]. Sublime Text має мінімалістичний інтерфейс та велику кількість плагінів, хоча їх менше, ніж у VS Code .

Atom колись був дуже популярним, але втратив частину аудиторії після виходу VS Code. Він пропонує багато функцій через плагіни, але це може вплинути на його продуктивність. Atom має відкритий вихідний код та схожий на VS Code інтерфейс, але деякі користувачі вважають його менш зручним.

Для розробки клієнтської частини було обрано найпопулярніший редактор коду на ринку програмного забезпечення – VS Code, оскільки цей інструмент пропонує велику гнучкість та ефективність при розробці веб-додатків. VS Code є легким і водночас потужним редактором, який підтримує множину мов програмування, включаючи HTML, CSS, JavaScript та TypeScript, що є основними для розробки клієнтської частини веб застосунків.

Завдяки широкому спектру доступних розширень, VS Code дозволяє розробникам легко додавати функціональність, яка відповідає конкретним потребам проекту, такі як лінтери, інструменти для управління версіями, фреймворки та бібліотеки. Це сприяє підвищенню продуктивності розробки та забезпечує високу якість коду.

Крім того, VS Code має вбудовану підтримку Git, що робить процес управління версіями простішим та інтуїтивнішим. Це особливо корисно в середовищі, де швидкість розробки та часті оновлення є критичними. Інтеграція

з системами автоматичного розгортання та тестування також підсилюється завдяки легкості налаштування VS Code під специфічні потреби проекту [23].

3.2 Розробка модуля композиції персоналізованих рекомендацій

Головний екран веб-сайту, також відомий як домашня сторінка або головна сторінка, є першою сторінкою, яку бачить користувач при відвідуванні сайту. Він відіграє ключову роль у формуванні першого враження про сайт та його зміст, а також у залученні та утриманні відвідувачів. Для веб системи аналізу харчової цінності продуктів, було розроблено власний інтерфейс користувача, який відповідає усім вимогам доступності та інтуїтивно зрозумілий. Основним кольором інтерфейсу було обрано помаранчевий та, як додатковим, сірий.

При відкриті сайту, користувача зустрічає головний екран із логотипом, слоганом та функціональними кнопками (див. рис. 3.1).

Головна сторінка сайту відповідає за перші враження користувача і містить ознайомчу інформацію, що дозволить відразу зрозуміти меті програмного продукту. Головна сторінка сайту реалізована у вигляді композиції React-компонентів, що дозволяє досягти модульності, повторного використання коду та зручності підтримки. Вона має привабливий дизайн, містить необхідну інформацію про сервіс та спонукає користувача до дії за допомогою кнопок з закликами.

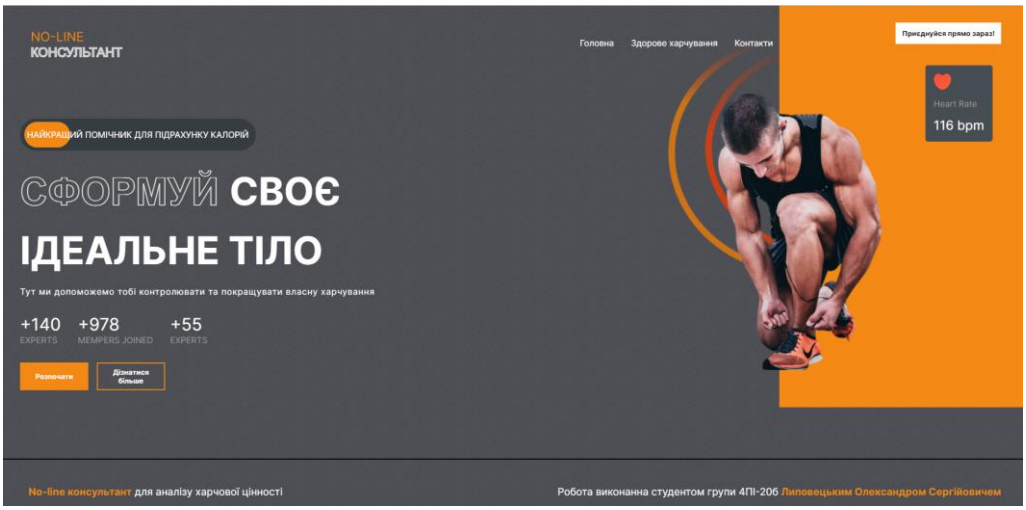


Рисунок 3.1 – Зображення головної сторінки

Головна сторінка сайту відповідає за перші враження користувача і містить ознайомчу інформацію, що дозволить відразу зрозуміти меті програмного продукту. Сторінка містить такі основні елементи, як кнопка «Розпочати», що дозволяє перейти до основного функціоналу продукту (див. рис. 2.2) та містить такі розділи, як «Здорове харчування» та «Контакти».

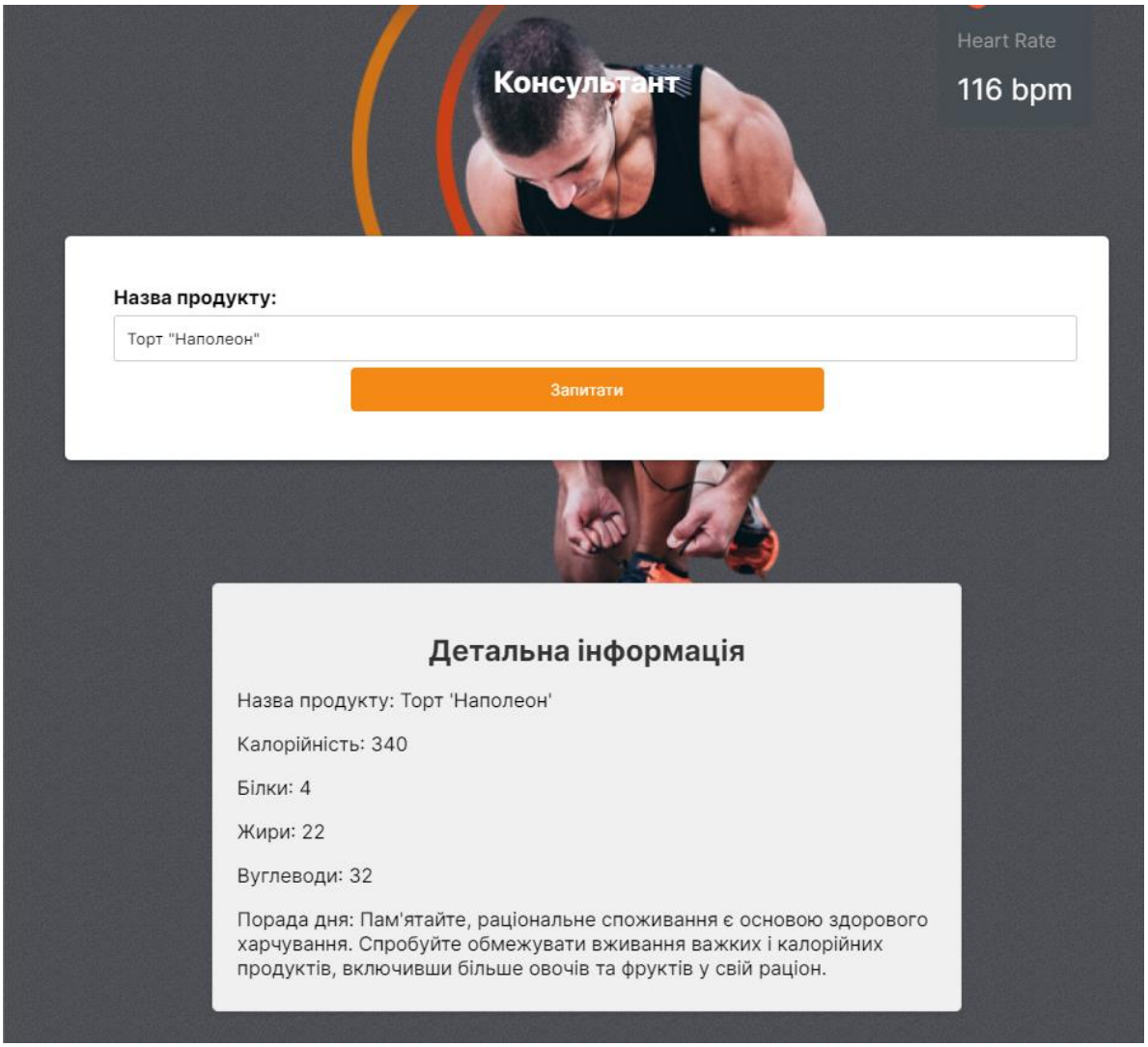


Рисунок 3.2 – Зображення сторінки із основним функціоналом

Ця сторінка слугує основним інтерфейсом для взаємодії користувача з веб-застосунком для аналізу харчової цінності продуктів.

Сторінка «Здорове харчування» дозволяє користувачам отримати корисні поради, щодо здорового харчування, та грає роль лише інформативної частини програмного продукту (див. рис. 3.3).

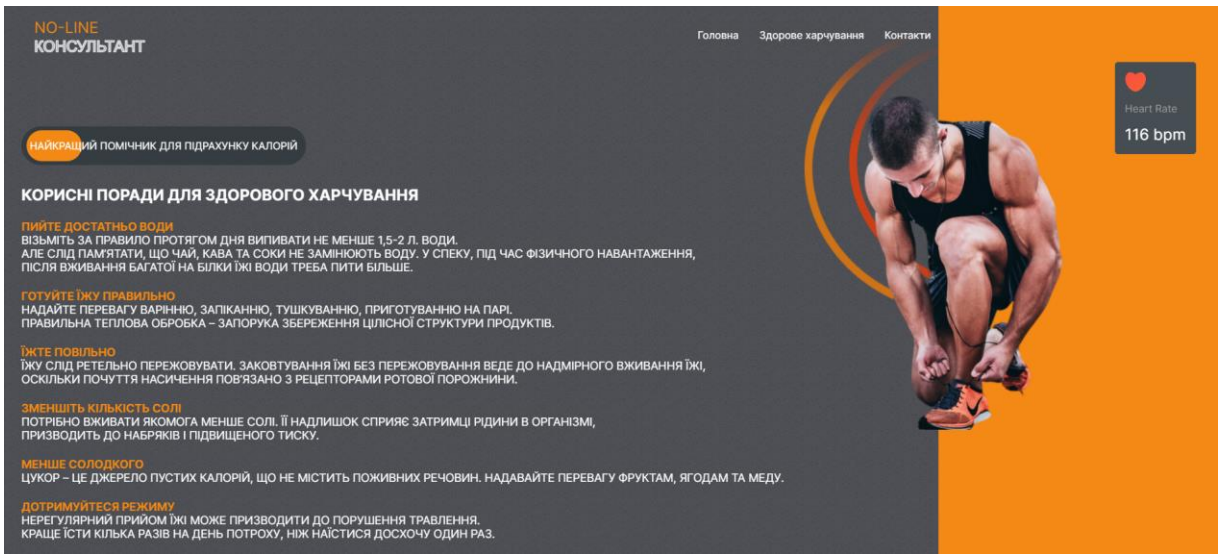


Рисунок 3.3 – Зображення сторінки «Здорове харчування»

3.3 Розробка серверної частини

Розробка серверної частини веб-застосунку "Online консультування з елементами штучного інтелекту для аналізу харчової цінності" за допомогою Java, Spring, та OpenAI ChatGPT пройшла кілька ключових етапів [25]:

Налаштування проекту та інфраструктури:

- Створення проекту Spring Boot за допомогою Spring Initializr або аналогічного інструменту.
- Додавання необхідних залежностей: spring-web, spring-data-jpa, spring-security, lombok, та бібліотеки для роботи з OpenAI API.
- Налаштування підключення до бази даних MySQL.
- Отримання та налаштування API ключа для OpenAI ChatGPT.

Створення моделей даних:

- Визначення моделей даних, таких як Product, User, Role, Choice, Message, Prompt. Було визначено основні моделі для коректної роботи системи: модель "Користувач" зберігає інформацію про користувачів, включаючи їх ідентифікатор, електронну пошту, ім'я, прізвище, пароль, ролі, статус видалення та продукти, які вони додали; Модель "Роль" визначає ролі користувачів, такі як адміністратор або звичайний користувач; Модель "Запит" зберігає інформацію про запити, відправлені до ChatGPT, включаючи текст запиту, відповідь та

користувача, який відправив запит4 Модель "Продукт" містить інформацію про продукти, такі як назва, калорійність, вміст білків, жирів, вуглеводів та статус видалення; Модель "Варіант відповіді" зберігає варіанти відповідей, отриманих від ChatGPT, включаючи індекс та текст відповіді.

Також, між моделями існують зв'язки. Користувач може мати кілька ролей, і одна роль може бути призначена багатьом користувачам. Користувач може додати багато продуктів та відправити багато запитів. Ці моделі разом забезпечують структуроване зберігання даних, необхідних для роботи веб-застосунку, включаючи інформацію про користувачів, їх дії та дані про продукти харчування.

- Використання анотацій Lombok (@Data, @Entity, @Id) для спрощення створення класів сутностей. Lombok спрощує розробку моделей даних, автоматично генеруючи стандартні методи (геттери, сеттери тощо) та надаючи анотації для опису сутностей бази даних та їх зв'язків. Це робить код коротшим, читабельнішим та менш схильним до помилок. Крім того, Lombok дозволяє реалізувати логічне видалення записів, що є корисним при роботі з базами даних [26].

Розробка репозиторіїв:

- Створення інтерфейсів репозиторіїв, що розширюють JpaRepository для кожної сутності.
- Використання методів JpaRepository (save, findById, findAll) для базових операцій з даними.
- Написання власних методів у репозиторіях для складніших запитів (наприклад, пошук продуктів за назвою або категорією).

Створення сервісів:

- Розробка сервісних класів, що містять бізнес-логіку застосунку.
- Використання репозиторіїв для отримання та збереження даних.
- Інтеграція з OpenAI ChatGPT API для аналізу харчової цінності продуктів та формування рекомендацій.

Розробка контролерів:

• Створення контролерів REST, що обробляють HTTP-запити від клієнтської частини.

• Використання сервісів для виконання бізнес-логіки та отримання даних.

• Повернення результатів у форматі JSON.

Налаштування Spring Security:

• Забезпечення безпеки застосунку шляхом налаштування автентифікації та авторизації користувачів.

• Використання Spring Security для захисту API-ендпоінтів та обмеження доступу до певних функцій лише для авторизованих користувачів.

Під час розробки серверної частини було інтегровано ряд нових технологій, таких як Liquibase. Liquibase використовується для управління змінами в базі даних під час розробки програмних застосунків. Це потужний інструмент, який допомагає автоматизувати процеси створення, оновлення та відкату схеми бази даних. Основне призначення Liquibase полягає у спрощенні та упорядкуванні процесу керування змінами бази даних, що є важливим для підтримання консистентності та якості даних.

3.4 Висновки

У розділі було проведено варіантний аналіз і обґрунтування вибору засобів для реалізації веб застосунку online консультування. Визначено, що для серверної частини найкращими засобами для розробки та фреймворк слугуватимуть – IntelliJ Idea та Spring Boot, а для клієнтської частини редактор коду та бібліотека – VS Code та React JS. Також було описано процес розробки клієнтської та серверної частин.

4. ТЕСТУВАННЯ ПРОГРАМНОГО ЗАСТОСУНКУ

4.1 Аналіз методів тестування програмного забезпечення

Тестування програмного забезпечення є невід'ємною та критично важливою частиною життєвого циклу розробки будь-якого програмного продукту. Воно відіграє ключову роль у забезпеченні якості, надійності та функціональності програмного забезпечення, а також у підвищенні задоволеності користувачів та зниженні ризиків, пов'язаних з його використанням.

Основні переваги тестування програмного забезпечення [27]:

- Виявлення та виправлення дефектів. Тестування дозволяє виявити та усунути різноманітні дефекти, помилки та недоліки у програмному коді, що можуть призвести до некоректної роботи програми, збоїв, втрати даних та інших серйозних проблем. Раннє виявлення та виправлення дефектів значно зменшує ризик виникнення проблем у майбутньому та підвищує загальну стабільність системи;
- Покращення якості програмного забезпечення. Тестування допомагає забезпечити відповідність програмного забезпечення всім заявленим функціональним та нефункціональним вимогам. Це включає перевірку правильності роботи всіх функцій, зручність використання інтерфейсу, швидкодію, безпеку, сумісність з різними платформами та інші важливі аспекти. Завдяки тестуванню програмне забезпечення стає більш надійним, ефективним та зручним для користувачів;
- Зниження витрат. Виявлення та виправлення дефектів на ранніх стадіях розробки значно дешевше та простіше, ніж на пізніших етапах або після випуску продукту. Тестування дозволяє виявити потенційні проблеми на ранніх стадіях, що дозволяє уникнути значних витрат на виправлення помилок у майбутньому та скоротити час розробки;
- Збільшення задоволеності користувачів. Якісне, надійне та зручне у використанні програмне забезпечення сприяє підвищенню задоволеності

користувачів. Тестування допомагає забезпечити відповідність продукту очікуванням та потребам користувачів, що робить його більш привабливим та конкурентоспроможним на ринку.

Також варто зазначити, що існує широкий спектр методів тестування програмного забезпечення, кожен з яких має свої особливості, переваги та недоліки. Вибір конкретних методів залежить від специфіки проекту, вимог до якості, бюджету та інших факторів.

Методи тестування [28]:

- Чорний ящик. Цей підхід до тестування не вимагає знання внутрішньої структури програми. Тестувальник взаємодіє з програмним забезпеченням через інтерфейс користувача, перевіряючи його реакцію на різні вхідні дані та дії. Чорний ящик ефективний для виявлення дефектів функціональності, але може пропустити деякі типи помилок, пов'язані з внутрішньою логікою програми;
- Білий ящик. Цей метод тестування вимагає глибокого розуміння внутрішньої структури та коду програми. Тестувальник аналізує код, перевіряє логіку роботи алгоритмів та виявляє потенційні проблеми. Білий ящик дозволяє виявити дефекти, які неможливо знайти за допомогою чорного ящика, але вимагає більше часу та ресурсів;
- Функціональне тестування. Цей вид тестування перевіряє, чи відповідає програмне забезпечення своїм функціональним вимогам, тобто чи виконує воно всі заявлені функції коректно та відповідно до очікувань користувачів;
- Нефункціональне тестування. Цей тип тестування зосереджується на нефункціональних аспектах програмного забезпечення, таких як продуктивність (швидкість роботи, використання ресурсів), безпека (стійкість до атак, захист даних), зручність використання (інтуїтивно зрозумілий інтерфейс, простота навчання), надійність (стійкість до збоїв, відновлення після помилок) та інші;
- Інтеграційне тестування. Цей вид тестування перевіряє, чи коректно взаємодіють різні компоненти та модулі програмного забезпечення між собою. Інтеграційне тестування допомагає виявити проблеми сумісності та забезпечити стабільну роботу системи в цілому;

• Системне тестування. Цей тип тестування перевіряє, чи відповідає програмне забезпечення всім системним вимогам, тобто чи працює воно коректно у цільовому середовищі з урахуванням всіх апаратних та програмних компонентів;

• Приймальне тестування. Цей вид тестування виконується користувачами або їх представниками для перевірки, чи відповідає програмне забезпечення їхнім очікуванням та потребам. Приймальне тестування допомагає виявити проблеми з точки зору користувача та забезпечити його задоволеність продуктом;

• Автоматизоване тестування. Цей підхід до тестування передбачає використання спеціальних інструментів та скриптів для автоматичного виконання тестів. Автоматизація дозволяє значно прискорити процес тестування, зменшити ризик людських помилок та забезпечити більш повне покриття тестових сценаріїв;

• Тестування на проникнення. Цей вид тестування імітує атаку зловмисника на програмне забезпечення з метою виявлення вразливостей у системі безпеки. Тестування на проникнення допомагає виявити та усунути потенційні загрози безпеці до того, як вони будуть використані зловмисниками;

• Регресійне тестування. Цей тип тестування виконується після внесення змін до програмного коду для перевірки, чи не порушили ці зміни існуючу функціональність. Регресійне тестування допомагає забезпечити стабільність та надійність системи після оновлень та виправлень.

Для забезпечення якості роботи онлайн-консультанта було впроваджено підхід функціонального тестування для забезпечення максимальної точності обробки даних та ефективності виконання серверної частини програмного застосунку.

Для впровадження тестування в систему було застосовано такі бібліотеки як: Junit та Mockito.

JUnit – це фреймворк для написання та запуску юніт-тестів в Java. Він надає набір анотацій та методів, які дозволяють створювати тести, перевіряти

результати виконання коду та організовувати тести у групи. JUnit є стандартом де-факто для юніт-тестування в Java і широко використовується в індустрії [29].

Mockito – це бібліотека для створення мок-об'єктів (mock objects) в Java. Мок-об'єкти - це спеціальні об'єкти, які імітують поведінку реальних об'єктів, але дозволяють контролювати їхню взаємодію з іншими частинами системи під час тестування. Mockito дозволяє легко створювати мок-об'єкти, налаштовувати їхню поведінку та перевіряти, чи були вони викликані з очікуваними параметрами [29].

Частини коду автоматизованого тестування, написаного з використанням Junit та Mockito, наведено на рисунку 4.1.

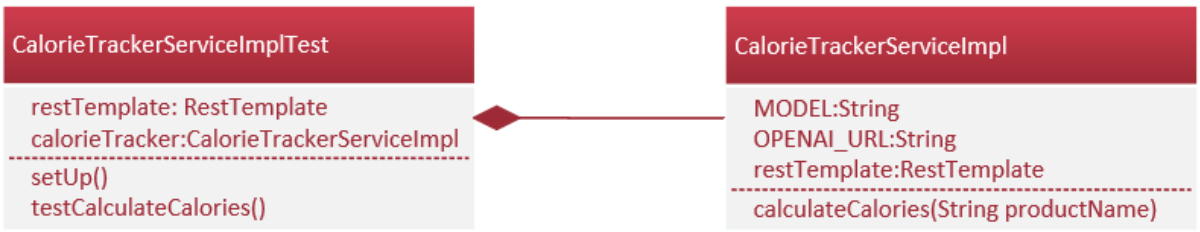


Рисунок 4.1 – Зображення класу, що описує тест

Цей тестовий клас (CalorieTrackerServiceImplTest) призначений для перевірки коректності роботи сервісу CalorieTrackerServiceImpl, який відповідає за отримання та обробку інформації про калорійність продуктів з ChatGPT API.

Основні компоненти тесту:

• Мок RestTemplate. За допомогою Mockito створюється мок-об'єкт restTemplate, який імітує реальну взаємодію з ChatGPT API. Це дозволяє ізолювати тестований сервіс від зовнішніх залежностей і зосередитися на перевірці його логіки;

• Підготовка тестових даних. Визначається тестовий запит (productName - назва продукту "apple") та очікувана відповідь від ChatGPT API у вигляді рядка, що містить інформацію про калорійність та інші характеристики продукту;

• Налаштування моку. За допомогою Mockito вказується, що при виклику методу restTemplate.exchange з будь-якими аргументами (в даному випадку це URL, метод запиту, тіло запиту та очікуваний тип відповіді) має повертатися підготовлена тестова відповідь;

• Виклик методу. Викликається метод calculateCalories тестованого сервісу з тестовим запитом;

• Перевірка результату. З отриманого результату витягуються значення калорійності, білків, жирів та вуглеводів, які порівнюються з очікуваними значеннями;

• Перевірка виклику restTemplate. За допомогою Mockito перевіряється, що метод exchange мок-об'єкта restTemplate був викликаний один раз з очікуваними параметрами. Це гарантує, що сервіс коректно взаємодіє з ChatGPT API.

Тестування онлайн-консультанта дозволяє виявити та виправити потенційні проблеми, забезпечити високу якість обслуговування користувачів та підвищити ефективність роботи системи.

4.2 Розробка інструкції користувача

Інструкція користувача є важливим елементом будь-якого програмного продукту, особливо у випадку веб-застосунку "Online консультування з елементами штучного інтелекту для аналізу харчової цінності". Актуальність інструкції полягає в наступному [31]:

• Полегшення використання. Інструкція допомагає користувачам швидко розібратися з функціоналом застосунку, зрозуміти, як вводити дані, отримувати результати та використовувати різні функції. Це особливо важливо для застосунків, що містять елементи штучного інтелекту, оскільки їх робота може бути не завжди очевидною для користувача;

• Зменшення кількості звернень до підтримки. Чітка та зрозуміла інструкція дозволяє користувачам самостійно вирішувати більшість питань, що виникають під час використання застосунку. Це зменшує навантаження на службу підтримки та дозволяє їй зосередитися на складніших проблемах;

• Підвищення задоволеності користувачів. Коли користувачі можуть легко та ефективно використовувати застосунок, вони отримують позитивний досвід взаємодії з ним. Це підвищує їхню задоволеність та лояльність до продукту;

• Зниження ризику помилок. Інструкція допомагає користувачам уникнути помилок при введенні даних та використанні функцій застосунку. Це особливо важливо у випадку з аналізом харчової цінності, де помилки можуть призвести до неправильних рекомендацій щодо харчування.

Перший крок, перше знайомство із сайтом – «Головна сторінка» (див. рис. 4.2).

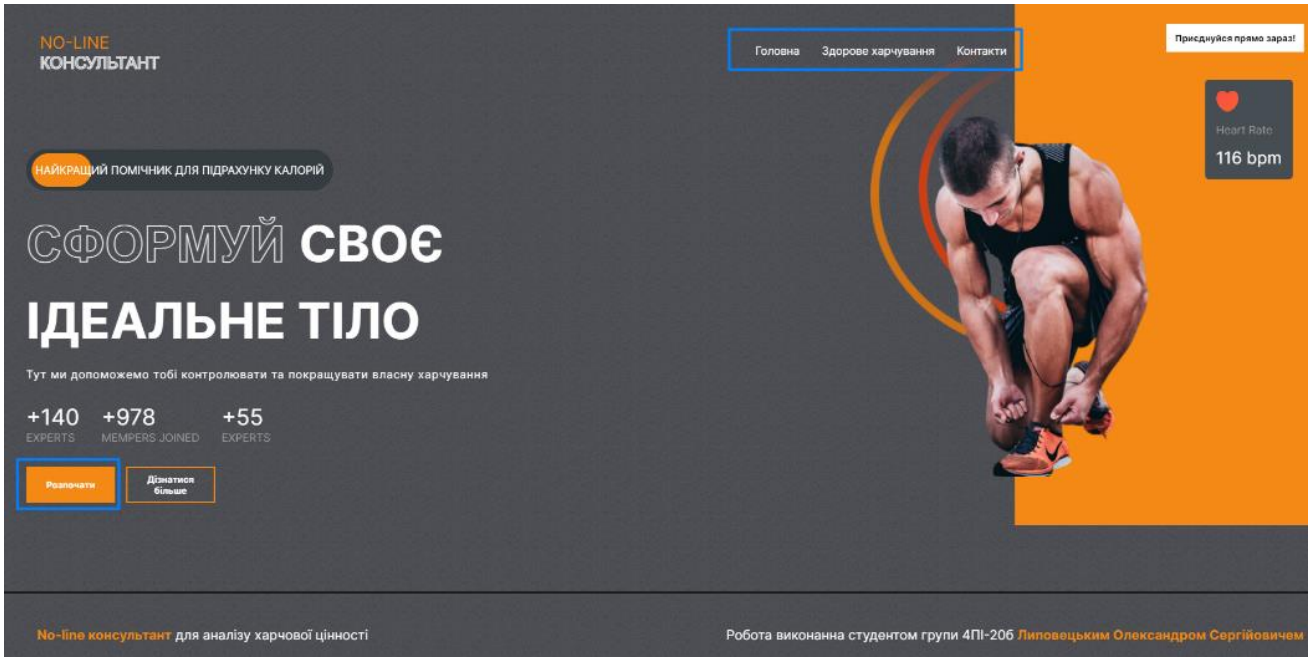


Рисунок 4.2 – Зображення головної сторінки

Для навігації на сайті варто використовувати панель керування, що включає у себе посилання на перехід між сторінками «Головна», «Здорове харчування» та «Контакти» (див. рис. 4.4 та 4.5).

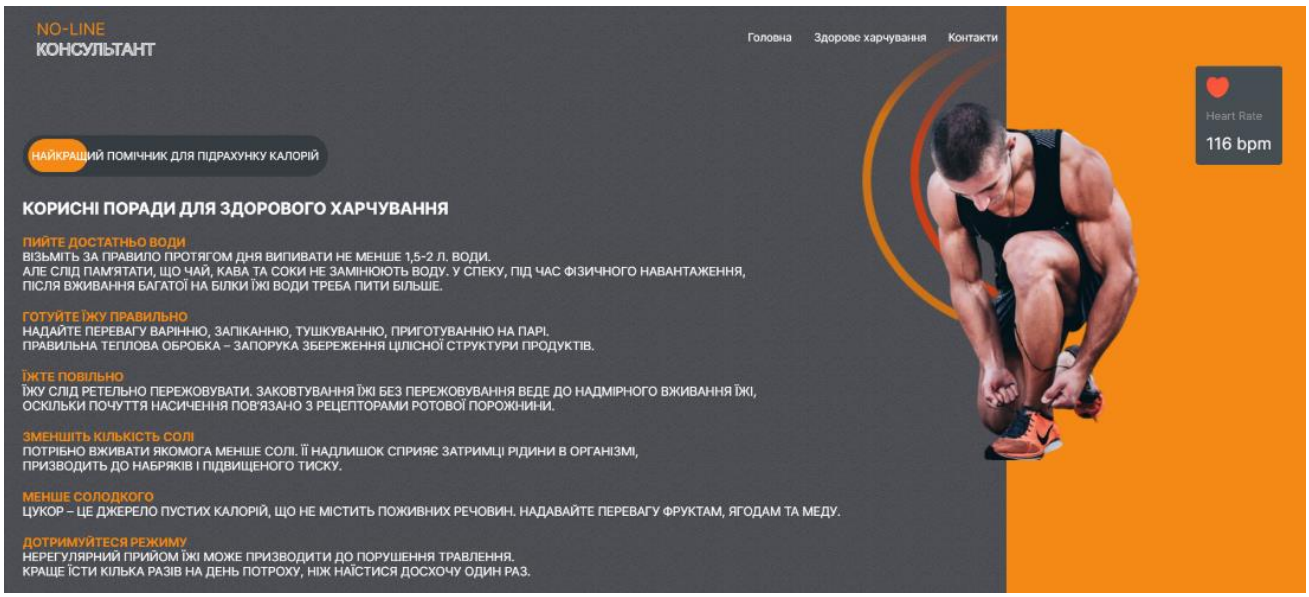


Рисунок 4.4 – Зображення сторінки «Здорове харчування»

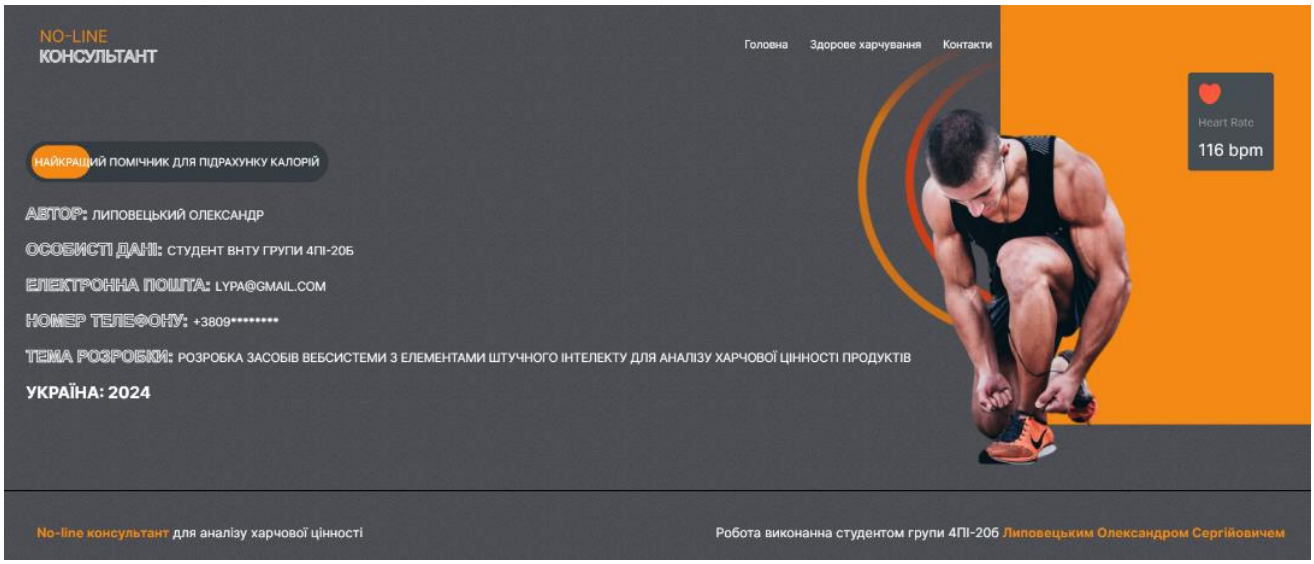


Рисунок 4.5 – Зображення сторінки «Контакти»

Для використання сайту в цілях отримання детальної інформації про калорійність певного продукту, потрібно на головній сторінці сайту натиснути на кнопку «Розпочати» (див. рис. 4.2).

Що у свою чергу перенаправить користувача на сторінку із формою для пошуку продукту та формою із інформацією (див. рис. 4.6).

Рисунок 4.6 – Зображення поля пошуку продукту на сторінці консультанта

Для того, щоб отримати інформацію про бажаний продукт, потрібно заповнити поле «Назва продукту» та натиснути на кнопку «Запитати», після чого буде згенерована відповідь (див. рис. 4.7) Також, можуть бути використані опціональні поля такі як: вік, вага, зріст та рівень акативності користувача для отримання точніших порад.

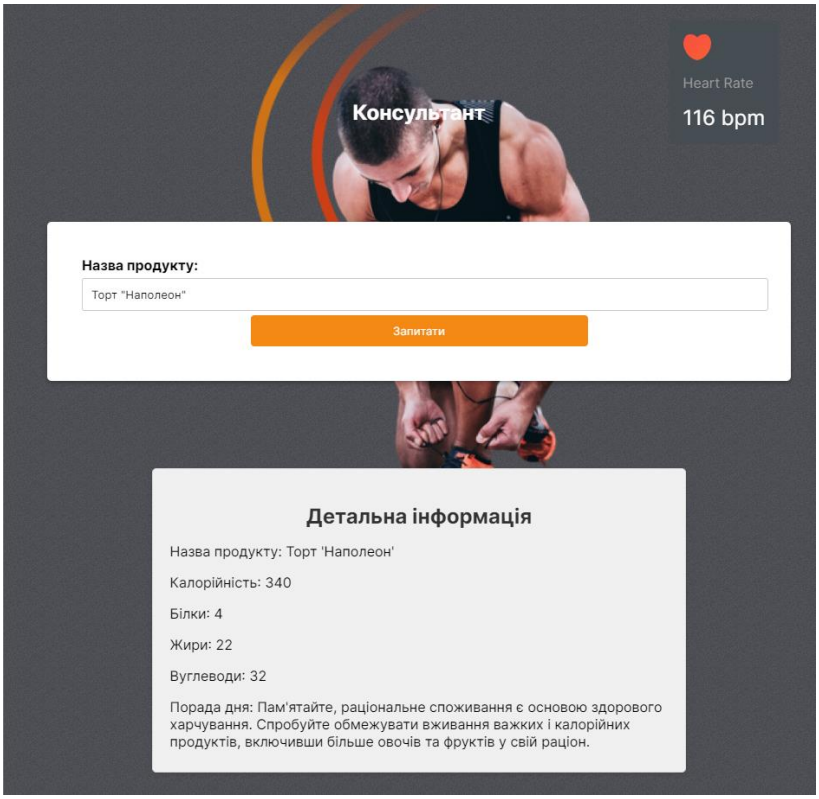


Рисунок 4.7 – Зображення форми із згенерованою відповіддю

Для того, щоб залишити коментар, побажання або скаргу, потрібно перейти на сторінку «Контакти» та надіслати листа на пошту, що вказана на сторінці (див. рис. 4.8).

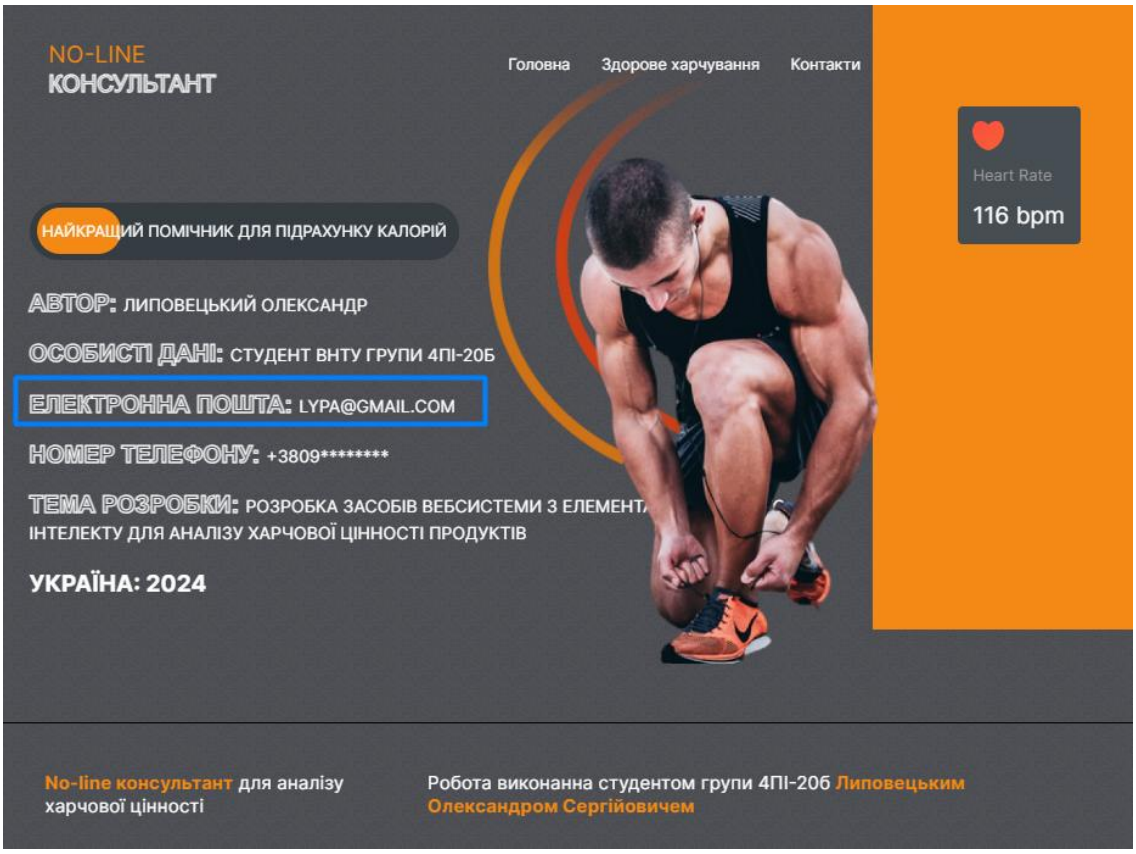


Рисунок 4.8 – Зображення поля із електронною поштою

4.3 Системні вимоги

Системні вимоги до програмного забезпечення – це своєрідний "фундамент" майбутнього продукту. Вони визначають, які технічні та програмні ресурси необхідні для успішної роботи програми, а також які функціональні можливості вона повинна мати.

Мінімальні системні вимоги (для базової функціональності).

Клієнтська частина (користувач):

- Операційна система: Windows 7/8/10/11, macOS 10.13+, Linux (будь-який сучасний дистрибутив);
- Браузер: Google Chrome, Mozilla Firefox, Safari (останні версії).

Апаратна частина:

- Процесор: 2 ядра, 1.8 ГГц+;
- Оперативна пам'ять (RAM): 4 ГБ;
- Вільний простір на диску: 100 МБ;
- Підключення до Інтернету: 5 Мбіт/с.

Серверна частина (хостинг):

- Операційна система: Linux (Ubuntu, CentOS);
- Веб-сервер: Apache, Nginx;
- Платформа: Java 8+;
- База даних: MySQL 5.7+.

Апаратна частина:

- Процесор: 2 ядра, 2.0 ГГц+;
- Оперативна пам'ять (RAM): 8 ГБ;
- Вільний простір на диску: 500 МБ (для бази даних) + 1 ГБ (для застосунку).

Рекомендовані системні вимоги (для оптимальної продуктивності).

Клієнтська частина (користувач):

- Операційна система: Windows 10/11, macOS 11+, Linux (будь-який сучасний дистрибутив);
- Браузер: Google Chrome, Mozilla Firefox (останні версії).

Апаратна частина:

- Процесор: 4 ядра, 2.5 ГГц+;
- Оперативна пам'ять (RAM): 8 ГБ;
- Вільний простір на диску: 200 МБ;
- Підключення до Інтернету: 10 Мбіт/с+.

Серверна частина (хостинг):

- Операційна система: Linux (Ubuntu, CentOS);
- Веб-сервер: Nginx;
- Платформа: Java 11+;
- База даних: MySQL 8+.

Апаратна частина:

- Процесор: 4 ядра, 3.0 ГГц+;
- Оперативна пам'ять (RAM): 16 ГБ+;
- Вільний простір на диску: 1 ГБ (для бази даних) + 2 ГБ (для застосунку);
- SSD накопичувач: рекомендується для швидшого доступу до даних.

Додаткові вимоги:

- API ключ для ChatGPT. Для використання функцій штучного інтелекту необхідно отримати API ключ від OpenAI.

Вимоги, що описані, як мінімальні та рекомендовані базуються на тому, що розроблене програмне забезпечення, що використовує Spring Boot 3, Java 17, React 18.2.0 та MySQL 8.0.34, вимагає певних ресурсів для ефективної роботи.

Для користувачів важливо мати сучасний комп'ютер з Windows, macOS або Linux, достатньо оперативної пам'яті (мінімум 4 ГБ) та швидким інтернет-з'єднанням. Рекомендується використовувати останні версії Google Chrome або Mozilla Firefox для найкращої сумісності.

Серверна частина вимагає потужнішого обладнання. Linux-сервер з 8 ГБ оперативної пам'яті та швидким процесором забезпечить стабільну роботу застосунку та бази даних MySQL. Використання SSD-накопичувача та веб-сервера Nginx допоможе покращити продуктивність.

4.4 Висновки

У четвертому розділі було досліджено ефективність веб системи зі штучним інтелектом шляхом проведення тестування та аналізу покриття коду

тестами. Крім того, було створено посібник користувача для початку роботи з програмним продуктом. Виконано та протестовано системні вимоги.

ВИСНОВКИ

У бакалаврській дипломній роботі було розроблено програмне забезпечення online консультування з елементами штучного інтелекту для аналізу харчової цінності. Розроблене програмне забезпечення має на меті покращення систем для аналізу харчової цінності продуктів, що додатково буде використовувати штучний інтелект для формування персоналізованих рекомендацій.

Було проаналізовано стан веб систем з використанням штучного інтелекту, виконано порівняльний аналіз аналогів та проаналізовано напрямки удосконалення програмного забезпечення для аналізу цінності харчових продуктів. Також було проведено аналіз методів розв’язання задачі та окреслено основні завдання для розробки програмного застосунку.

Було спроєктовано модель даних та розроблено структуру програмного застосунку. Було розроблено загальний алгоритм роботи застосунку, розроблено алгоритм оброки даних користувача та взаємодії з постачальником штучного інтелекту та детально описано його.

Протягом роботи над бакалаврською дипломною роботою було також розглянуто переваги й недоліки популярних мов програмування та середовищ розробки. На основі аналізу було обґрунтовано вибір таких технологій для розробки веб застосунку як: для клієнтської частини – редактор коду VS Code та технологію розробки React JS, для серверної частини – середовище розробки IntelliJ Idea та фреймворк Spring (а особливо модуль Spring Boot). Також було описано деталі розробки клієнтської та серверної частин.

На останок було проведено заходи для аналізу та впровадження методів тестування для забезпечення ефективності та безперебійності у роботі програмного застосунку. Було розроблено інструкцію користувача та визначено мінімальні та рекомендовані системні вимоги.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Липовецький О.С., Катєльніков Д.І.. Актуальність застосування on-line консультування з елементами штучного інтелекту для аналізу харчової цінності. Матеріали LIII науково-технічної конференції підрозділів ВНТУ, Вінниця, 20-22 березня 2024 р. Електрон. текст. дані. 2024. URI: https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/view/21376.

2. MyFitnessPal [Електронний ресурс] – Режим доступу до ресурсу: https://www.myfitnesspal.com/

3. Lose It! [Електронний ресурс] – Режим доступу до ресурсу: https://www.loseit.com/

4. FatSecret [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/FatSecret

5. Yuka [Електронний ресурс] – Режим доступу до ресурсу: https://apps.apple.com/us/app/yuka-food-cosmetic-scanner/id1092799236

6. Advantages and Disadvantages of Artificial Intelligence [AI] [Електронний ресурс] – Режим доступу до ресурсу: https://www.simplilearn.com/advantages-and-disadvantages-of-artificial-intelligence-article

7. Controller-Service-Repository [Електронний ресурс] – Режим доступу до ресурсу: https://tom-collings.medium.com/controller-service-repository

8. RDBMS полегшує взаємодію між користувачами та базою даних [Електронний ресурс] – Режим доступу до ресурсу: https://nofluffjobs.com/uk/log/robota-v-it/rdbms-facilitates-interaction-between-users-and-the-database/

9. What is NoSQL? [Електронний ресурс] – Режим доступу до ресурсу: https://www.mongodb.com/resources/basics/databases/nosql-explained

10. Object-oriented database management system (OODBMS) [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchoracle/definition/object-oriented-database-management-system

11. Google Gemini vs ChatGPT 2024: AI Chatbot Head-to-Head Test [Електронний ресурс] – Режим доступу до ресурсу: https://tech.co/news/google-bard-vs-chatgpt

12. Java Web Development [Електронний ресурс] – Режим доступу до ресурсу: https://www.javatpoint.com/java-web-development

13. Advantages of JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://codeinstitute.net/global/blog/advantages-of-javascript/

14. Python [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Python

15. Time to GO: розбираємо, що таке Golang та радимо, з чого почати [Електронний ресурс] – Режим доступу до ресурсу: https://www.globallogic.com/ua/insights/blogs/time-to-golang/

16. What is Intellij IDEA - A Comprehensive Guide [Електронний ресурс] – Режим доступу до ресурсу: https://www.simplilearn.com/what-is-intellij-idea-article

17. Why should Java developers consider using Eclipse IDE? [Електронний ресурс] – Режим доступу до ресурсу: https://www.linkedin.com/pulse/why-should-java-developers-consider-using-eclipse-ide-vinay-kumar

18. Know More about NetBeans IDE [Електронний ресурс] – Режим доступу до ресурсу: https://fourcornerstone.com/netbeans-ide/

19. Електронний HTML і CSS довідник українською мовою [Електронний ресурс] – Режим доступу до ресурсу: https://html-css.co.ua/

20. React [Електронний ресурс] – Режим доступу до ресурсу: https://uk.legacy.reactjs.org/

21. Vue js [Електронний ресурс] – Режим доступу до ресурсу: https://vuejs.org/

22. Angular [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Angular_(web_framework)

23. Why Visual Studio Code? [Електронний ресурс] – Режим доступу до ресурсу: https://code.visualstudio.com/docs/editor/whyvscode

24. Sublime text [Електронний ресурс] – Режим доступу до ресурсу: https://www.sublimetext.com/

25. Greg L Turnquist, Dave Syer, Mark Heckler: Learning Spring Boot 3.0 - Third Edition: Simplify the development of production-grade applications using Java and Spring 3rd ed. Edition. США . Packt Publishing 2022.

26. Project Lombok [Електронний ресурс] – Режим доступу до ресурсу: https://projectlombok.org/

27. Testing [Електронний ресурс] – Режим доступу до ресурсу: https://www.linkedin.com/pulse/top-9-benefits-software-testing-value-shore-doo

28. Different Types of Testing in Software [Електронний ресурс] – Режим доступу до ресурсу: https://www.perfecto.io/resources/types-of-testing

29. The Difference Between JUnit and Mockito: Detailed comparison Software [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@skillcombo/the-difference-between-junit-and-mockito-detailed-comparison-f5b86e062a25

30. Інструкція користувача: User Manual [Електронний ресурс] – Режим доступу до ресурсу: https://qatestlab.com/resources/knowledge-center/sample-deliverables/user-manuals/

ДОДАТКИ

Додаток А. Технічне завдання

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
д.т.н., професор
_____ О. Н. Романюк
« 12 » березня 2024 р.

Технічне завдання

на бакалаврську дипломну роботу «Розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React» за спеціальністю 121 – Інженерія програмного забезпечення

Керівник бакалаврської дипломної роботи:
_____ д.т.н., доцент Катєльніков Д.І.
« 12 » березня 2024 р.

Виконав:
_____ студент гр. 4ПІ-20б О. С. Липовецький
« 12 » березня 2024 р.

Вінниця – 2024 року

1. Найменування та галузь застосування

Бакалаврська дипломна робота: «Розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React.».

Галузь застосування – веб технології.

2. Підстава для розробки.

Підставою для виконання бакалаврської дипломної роботи (БДР) є індивідуальне завдання на БДР та наказ №80 від 11 березня 2024 року ректора ВНТУ про закріплення тем БДР.

3. Мета та призначення розробки.

Метою бакалаврської дипломної роботи є процес покращення систем для аналізу харчової цінності продуктів, що додатково буде використовувати штучний інтелект для формування персоналізованих рекомендацій.

Призначення роботи – розробка та реалізація програмного забезпечення веб застосунку для аналізу харчової цінності.

4. Вихідні дані для проведення НДР

Перелік основних літературних джерел, на основі яких буде виконуватись БДР:

1. Java Web Development [Електронний ресурс] – Режим доступу до ресурсу: https://www.javatpoint.com/java-web-development

2. React [Електронний ресурс] – Режим доступу до ресурсу: https://uk.legacy.reactjs.org/

3. Greg L Turnquist, Dave Syer, Mark Heckler: Learning Spring Boot 3.0 - Third Edition: Simplify the development of production-grade applications using Java and Spring 3rd ed. Edition. США . Packt Publishing 2022.

4. Testing [Електронний ресурс] – Режим доступу до ресурсу: https://www.linkedin.com/pulse/top-9-benefits-software-testing-value-shore-doo

5. Технічні вимоги

Тип програми – веб система; модель розробки – ітеративна; засоби організації даних програми – MySQL, Spring Data JPA; вихідні дані: відображення інформації запиту користувача; середовище розробки – IntelliJ Idea; редактор коду – VS Code; мова програмування – Java, JavaScript; використані бібліотеки та фреймворки – React JS, Spring (Boot, OpenAI, Data JPA, Liquibase).

6. Конструктивні вимоги

Графічна та текстова документація повинна відповідати діючим стандартам України.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

1. Пояснювальна записка до БДР.
2. Технічне завдання.
3. Лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки:

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи
1	Аналіз стану питання та постановка задач дослідження	14.03.24 –17.03.24
2	Розробка структури та алгоритмів роботи програмного продукту	17.03.24–26.03.24
3	Розробка загального алгоритму роботи застосунку	26.03.24–01.04.24
4	Аналіз та вибір засобів для реалізації програмної системи	01.04.24–05.04.24
5	Розробка програмного забезпечення	05.04.24–17.05.24
6	Тестування програмного застосунку	17.05.24–25.05.24
7	Оформлення матеріалів до захисту БДР	25.05.24–30.05.24

10. Порядок контролю та прийняття

Виконання етапів бакалаврської дипломної роботи контролюється керівником згідно з графіком виконання роботи.

Прийняття бакалаврської дипломної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком.

Додаток Б. Протокол перевірки дипломної роботи на наявність текстових запозичень

ПРОТОКОЛ
ПЕРЕВІРКИ БАКАЛАВРСЬКОЇ ДИПЛОМНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка вебзастосунку on-line інтелектуального консультування для аналізу харчової цінності з використанням технологій Java, Spring, JavaScript та React.

Тип роботи: БДР

Підрозділ: кафедра програмного забезпечення, ФІТКІ

Науковий керівник: Романюк О.Н.

Оригінальність	97.6
Схожість	2.4

Аналіз звіту подібності

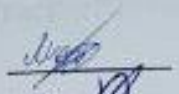
▪ **Запозичення, виявлені у роботі, оформленні коректно і не містять ознак плагіату.**

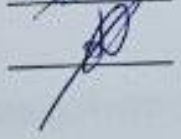
☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цілісності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховання недобросовісних запозичень.

Особа, відповідальна за перевірку  Черноволик Г.О.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Автор роботи  Липовецький О.С.

Керівник роботи  Катєльніков Д. І.

Додаток В. Лістинг програми

```
//OpenAIConfig
package ua.vntu.nolinefoodassistant.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.client.RestTemplate;

@Configuration
public class OpenAIConfig {
   private final String openaiKey = "sk ";

   @Bean
   public RestTemplate template() {
      RestTemplate template = new RestTemplate();
      template.getInterceptors().add(
            ((request, body, execution) -> {
               request.getHeaders().add("Authorization", "Bearer " + openaiKey);
               return execution.execute(request, body);
            })
      );
      return template;
   }
}
//CalorieTrackerController
package ua.vntu.nolinefoodassistant.controller;

import lombok.RequiredArgsConstructor;
import org.springframework.web.bind.annotation.*;
import ua.vntu.nolinefoodassistant.dto.ProductDto;
import ua.vntu.nolinefoodassistant.service.CalorieTrackerService;

@RestController
@RequestMapping("/calories")
@RequiredArgsConstructor
public class CalorieTrackerController {
   private final CalorieTrackerService calorieTrackerService;

   @CrossOrigin
   @GetMapping
   public ProductDto chat(@RequestParam String product) {
      return calorieTrackerService.calculateCalories(product);
   }
```

```
}
//HealthController
package ua.vntu.nolinefoodassistant.controller;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.Mapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RequestMapping(path = "/health")
@RestController
public class HealthController {
    @GetMapping
    public String getHealthCheck() {
        return "health";
    }
}
//ChatGptRequest
package ua.vntu.nolinefoodassistant.dto;

import lombok.Data;

import javax.crypto.MacSpi;
import java.util.ArrayList;
import java.util.List;

@Data
public class ChatGptRequest {
    private String model;
    private List<Message> messages;

    public ChatGptRequest(String model, String prompt) {
        this.model = model;
        this.messages = new ArrayList<>();
        this.messages.add(new Message("user", prompt));
    }
}
//ChatGptResponse
package ua.vntu.nolinefoodassistant.dto;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import ua.vntu.nolinefoodassistant.model.Choice;
import java.util.List;
```

```
@Data
@AllArgsConstructor
@NoArgsConstructor
public class ChatGptResponse {
   private List<Choice> choices;
}
//Message
package ua.vntu.nolinefoodassistant.dto;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@AllArgsConstructor
@NoArgsConstructor
public class Message {
   private String role;
   private String content;
}
//Message
package ua.vntu.nolinefoodassistant.dto;

import lombok.Data;

@Data
public class ProductDto {
   private String name;
   private Integer calories;
   private Integer protein;
   private Integer fat;
   private Integer carbohydrate;
   private String advice;
}

package ua.vntu.nolinefoodassistant.model;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import ua.vntu.nolinefoodassistant.dto.Message;

@Data
@AllArgsConstructor
```

```
@NoArgsConstructor
public class Choice {
   private Long index;
   private Message message;
}

//Product
package ua.vntu.nolinefoodassistant.model;

import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;
import org.hibernate.annotations.SQLDelete;
import org.hibernate.annotations.Where;

@Entity
@Table(name = "products")
@SQLDelete(sql = "UPDATE users SET is_deleted = true WHERE id=?")
@Where(clause = "is_deleted=false")
@Getter
@Setter
public class  Product {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   @Column(nullable = false, unique = true)
   private String name;
   @Column(nullable = false)
   private Integer calories;
   @Column(nullable = false)
   private Integer protein;
   @Column(nullable = false)
   private Integer fat;
   @Column(nullable = false)
   private Integer carbohydrate;
   @Column(name = "is_deleted", nullable = false)
   private boolean isDeleted = false;
}
//Prompt
package ua.vntu.nolinefoodassistant.model;

import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;
```

```
@Entity
@Getter
@Setter
@Table(name = "prompts")
public class Prompt {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(nullable = false)
    private String content;
    @Column(nullable = false)
    private String answer;
    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private User user;
}
//User
package ua.vntu.nolinefoodassistant.model;

import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;
import org.hibernate.annotations.SQLDelete;
import org.hibernate.annotations.Where;

import java.util.HashSet;
import java.util.Set;

@Entity
@Setter
@Getter
@SQLDelete(sql = "UPDATE users SET is_deleted = true WHERE id=?")
@Where(clause = "is_deleted=false")
@Table(name = "users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(nullable = false, unique = true)
    private String email;
    @Column(nullable = false)
    private String firstName;
    @Column(nullable = false)
    private String lastName;
    @Column(nullable = false)
```

```
    private String password;
    @Column(nullable = false)
    @ManyToMany
    @JoinTable(name = "users_roles",
            joinColumns = @JoinColumn(name = "user_id"),
            inverseJoinColumns = @JoinColumn(name = "role_id"))
    private Set<Role> roles = new HashSet<>();
    @Column(name = "is_deleted", nullable = false)
    private boolean isDeleted = false;
}
// CalorieTrackerServiceImpl
package ua.vntu.nolinefoodassistant.service;

import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;
import ua.vntu.nolinefoodassistant.dto.ChatGptRequest;
import ua.vntu.nolinefoodassistant.dto.ChatGptResponse;
import ua.vntu.nolinefoodassistant.dto.ProductDto;
import ua.vntu.nolinefoodassistant.model.Choice;

@Service
public class CalorieTrackerServiceImpl implements CalorieTrackerService{
    private static final String MODEL = "gpt-3.5-turbo";
    private static final String OPENAI_URL =
"https://api.openai.com/v1/chat/completions";
    @Autowired
    private RestTemplate restTemplate;

    @Override
    public ProductDto calculateCalories(String productName) {
        ChatGptRequest request = new ChatGptRequest(MODEL,
PromptGenerator.generatePromptText(productName));
        ChatGptResponse chatGptResponse =
restTemplate.postForObject(OPENAI_URL, request, ChatGptResponse.class);
        String response = chatGptResponse.getChoices()
                .stream().findFirst().orElse(new Choice()).getMessage().getContent();
        try {
            return new ObjectMapper().readValue(response, ProductDto.class);
        } catch (JsonProcessingException e) {
            throw new RuntimeException(e);
        }
    }
```

```
import jakarta.validation.ConstraintValidator;
import jakarta.validation.ConstraintValidatorContext;
import org.springframework.beans.BeanWrapperImpl;

public class FieldMatchValidator implements ConstraintValidator<FieldMatch,
Object> {
   private String field;
   private String fieldMatch;

   @Override
   public void initialize(FieldMatch constraintAnnotation) {
      this.field = constraintAnnotation.field();
      this.fieldMatch = constraintAnnotation.fieldMatch();
   }

   @Override
   public boolean isValid(Object obj, ConstraintValidatorContext
constraintValidatorContext) {
      Object fieldValue = new BeanWrapperImpl(obj).getPropertyValue(field);
      Object fieldMatchValue = new
BeanWrapperImpl(obj).getPropertyValue(fieldMatch);

      return (fieldMatch == fieldMatchValue)
            || (fieldValue != null
            && fieldValue.equals(fieldMatchValue));
   }
}

}
package ua.vntu.nolinefoodassistant.service;

public class PromptGenerator {
   private static final String STANDARD_PROMPT_PART1 = "Скільки калорій у
";
   private static final String STANDARD_PROMPT_PART2 = """
         виведи цю інформацію у форматі json:
         {
           name: {назва_продукту}
           calories: {кількість калорій у 100 грамах продукту}
           protein: {кількість білків у 100 грамах продукту}
           fat: {кількість жирів у 100 грамах продукту}
           carbohydrate: {кількіть вуглеводів у 100 грамах продукту}
           advice: {випадкова порада по схудненю}
         }
```

```
            """;


    public static String generatePromptText(String productName) {
        return STANDARD_PROMPT_PART1 + productName +
STANDARD_PROMPT_PART2;
    }
}

//MapperConfig
import org.mapstruct.InjectionStrategy;
import org.mapstruct.NullValueCheckStrategy;

@org.mapstruct.MapperConfig(
        componentModel = "spring",
        injectionStrategy = InjectionStrategy.CONSTRUCTOR,
        nullValueCheckStrategy = NullValueCheckStrategy.ALWAYS,
        implementationPackage = "<PACKAGE_NAME>.impl"
)
public class MapperConfig {
}

//SecurityConfig
import lombok.RequiredArgsConstructor;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.config.Customizer;
import
org.springframework.security.config.annotation.authentication.configuration.Authent
icationConfiguration;
import
org.springframework.security.config.annotation.method.configuration.EnableMethod
Security;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.configurers.AbstractHttpConfig
urer;
import org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;
```

```
import
org.springframework.security.web.authentication.UsernamePasswordAuthenticationF
ilter;

@Configuration
@EnableMethodSecurity
@RequiredArgsConstructor
public class SecurityConfig {
   private final UserDetailsService userDetailsService;
   private final JwtAuthenticationFilter jwtAuthenticationFilter;

   @Bean
   public PasswordEncoder passwordEncoder() {
      return new BCryptPasswordEncoder();
   }

   @Bean
   public SecurityFilterChain securityFilterChain(HttpSecurity httpSecurity) throws
Exception {
      return httpSecurity
            .cors(AbstractHttpConfigurer::disable)
            .csrf(AbstractHttpConfigurer::disable)
            .authorizeHttpRequests(
                  auth -> auth
                        .requestMatchers("/auth/**",,
                              "/error")
                        .permitAll()
                        .anyRequest()
                        .authenticated()
            )
            .httpBasic(Customizer.withDefaults())
            .sessionManagement(
                  session -> session
                        .sessionCreationPolicy(SessionCreationPolicy.STATELESS))
            .addFilterBefore(jwtAuthenticationFilter,
                  UsernamePasswordAuthenticationFilter.class)
            .userDetailsService(userDetailsService)
            .build();
   }

   @Bean
   public AuthenticationManager authenticationManager(
         AuthenticationConfiguration authenticationConfiguration
   ) throws Exception {
      return authenticationConfiguration.getAuthenticationManager();
```

```
    }
}

import io.swagger.v3.oas.annotations.Operation;
import io.swagger.v3.oas.annotations.tags.Tag;
import jakarta.validation.Valid;
import lombok.RequiredArgsConstructor;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@Tag(name = "User's authentication", description = "Endpoints for logging in and
registering users")
@RestController
@RequestMapping("/auth")
@RequiredArgsConstructor
public class AuthenticationController {
    private final UserService userService;
    private final AuthenticationService authenticationService;

    @PostMapping("/login")
    @Operation(summary = "Login an existed user")
    public UserLoginResponseDto login(@RequestBody @Valid
UserLoginRequestDto requestDto) {
        return authenticationService.authenticate(requestDto);
    }

    @PostMapping("/register")
    @Operation(summary = "Register a new user")
    public UserResponseDto register(@RequestBody @Valid
UserRegistrationRequestDto requestDto)
                                            throws RegistrationException {
        return userService.register(requestDto);
    }
}
//AuthRepository
import lombok.RequiredArgsConstructor;
import org.springframework.security.authentication.AuthenticationManager;
import
org.springframework.security.authentication.UsernamePasswordAuthenticationToke
n;
import org.springframework.security.core.Authentication;
import org.springframework.stereotype.Service;
```

```
@Service
@RequiredArgsConstructor
public class AuthenticationService {
    private final JwtUtil jwtUtil;
    private final AuthenticationManager authenticationManager;

    public UserLoginResponseDto authenticate(UserLoginRequestDto requestDto) {
        final Authentication authenticate = authenticationManager
                .authenticate(new
UsernamePasswordAuthenticationToken(requestDto.getEmail(),
                                                    requestDto.getPassword()));

        String token = jwtUtil.generateToken(authenticate.getName());
        return new UserLoginResponseDto(token);
    }
}

//CustomUserDetailsService
import lombok.RequiredArgsConstructor;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Component;

@Component
@RequiredArgsConstructor
public class CustomUserDetailsService implements UserDetailsService {
    private final UserRepository userRepository;

    @Override
    public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
        return userRepository.findByEmail(username)
                .orElseThrow(() -> new EntityNotFoundException("Can't find a user by
username: "
                                                + username));
    }
}

// JwtAuthenticationFilter
import jakarta.servlet.FilterChain;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
```

```
import lombok.RequiredArgsConstructor;
import
org.springframework.security.authentication.UsernamePasswordAuthenticationToke
n;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.stereotype.Component;
import org.springframework.util.StringUtils;
import org.springframework.web.filter.OncePerRequestFilter;

@Component
@RequiredArgsConstructor
public class JwtAuthenticationFilter extends OncePerRequestFilter {
   private static final String AUTH_HEADER = "Authorization";
   private static final String AUTH_TYPE = "Bearer ";
   private final JwtUtil jwtUtil;
   private final UserDetailsService userDetailsService;

   @Override
   protected void doFilterInternal(HttpServletRequest request,
                           HttpServletResponse response,
                           FilterChain filterChain) throws ServletException, IOException
{
      String token = getToken(request);
      if (token != null && jwtUtil.isValidToken(token)) {
         String username = jwtUtil.getUsername(token);
         UserDetails userDetails =
userDetailsService.loadUserByUsername(username);
         Authentication authentication = new
UsernamePasswordAuthenticationToken(userDetails,
               null, userDetails.getAuthorities());
         SecurityContextHolder.getContext().setAuthentication(authentication);
      }
      filterChain.doFilter(request, response);
   }

   private String getToken(HttpServletRequest request) {
      String bearerToken = request.getHeader(AUTH_HEADER);
      if (StringUtils.hasText(bearerToken) &&
bearerToken.startsWith(AUTH_TYPE)) {
         return bearerToken.substring(AUTH_TYPE.length());
      }
      return null;
```

```
    }
}

// JwtUtil
import io.jsonwebtoken.Claims;
import io.jsonwebtoken.Jws;
import io.jsonwebtoken.JwtException;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.security.Keys;
import java.nio.charset.StandardCharsets;
import java.security.Key;
import java.util.Date;
import java.util.function.Function;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class JwtUtil {
    @Value("${jwt.expiration}")
    private Long expiration;
    private final Key secret;

    public JwtUtil(@Value("${jwt.secret}") String secretString) {
        secret =
Keys.hmacShaKeyFor(secretString.getBytes(StandardCharsets.UTF_8));
    }

    public String generateToken(String username) {
        return Jwts.builder()
                .setSubject(username)
                .setIssuedAt(new Date(System.currentTimeMillis()))
                .setExpiration(new Date(System.currentTimeMillis() + expiration))
                .signWith(secret)
                .compact();
    }

    public boolean isValidToken(String token) {
        try {
            Jws<Claims> claimsJws = Jwts.parserBuilder()
                    .setSigningKey(secret)
                    .build()
                    .parseClaimsJws(token);
            return !claimsJws.getBody().getExpiration().before(new Date());
        } catch (JwtException | IllegalArgumentException e) {
            throw new JwtException("Expired or invalid JWT token");
```

```
        }
    }

    public String getUsername(String token) {
        return getClaimFromToken(token, Claims::getSubject);
    }

    private <T> T getClaimFromToken(String token, Function<Claims, T>
claimsResolver) {
        final Claims claims = Jwts.parserBuilder()
                .setSigningKey(secret)
                .build()
                .parseClaimsJws(token)
                .getBody();
        return claimsResolver.apply(claims);
    }
}

// UserServiceImpl
import java.util.Set;
import lombok.RequiredArgsConstructor;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Lazy;
import org.springframework.security.core.Authentication;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

@Service
@RequiredArgsConstructor
public class UserServiceImpl implements UserService {
    private static final RoleName DEFAULT_ROLE = RoleName.ROLE_USER;
    private final UserRepository userRepository;
    private final RoleRepository roleRepository;
    private final PasswordEncoder passwordEncoder;
    private final UserMapper userMapper;

    @Override
    public UserResponseDto register(UserRegistrationRequestDto requestDto)
                                    throws RegistrationException {
        if (userRepository.findByEmail(requestDto.getEmail()).isPresent()) {
            throw new RegistrationException("Unable to complete registration!");
        }
        User user = userMapper.toModel(requestDto);
        user.setPassword(passwordEncoder.encode(user.getPassword()));
        Role defaultRole = roleRepository
```

```
                .findByName(DEFAULT_ROLE)
                .orElseThrow(() -> new EntityNotFoundException("The role: "
                                                + DEFAULT_ROLE + " does not exist"));
        user.setRoles(Set.of(defaultRole));
        User savedUser = userRepository.save(user);

        return userMapper.toDto(savedUser);
    }

    @Override
    public User getAuthenticatedUser(Authentication authentication) {
        return userRepository.findByEmail(authentication.getName())
                .orElseThrow(() -> new EntityNotFoundException("Can't find a user by
email: "
                        + authentication.getName()));
    }
}

//UserLoginRequestDto
import io.swagger.v3.oas.annotations.media.Schema;
import jakarta.validation.constraints.Email;
import jakarta.validation.constraints.NotBlank;
import lombok.Data;
import org.hibernate.validator.constraints.Length;

@Data
public class UserLoginRequestDto {
    @NotBlank
    @Email
    @Schema(example = "new@example.com")
    private String email;

    @NotBlank
    @Length(min = 8)
    private String password;
}

import lombok.AllArgsConstructor;
import lombok.Data;

@Data
@AllArgsConstructor
public class UserLoginResponseDto {
    private String token;
}
```

```
import io.swagger.v3.oas.annotations.media.Schema;
import jakarta.validation.constraints.Email;
import jakarta.validation.constraints.NotBlank;
import lombok.Data;
import org.hibernate.validator.constraints.Length;

@Data
@FieldMatch.List({
        @FieldMatch(
                field = "password",
                fieldMatch = "repeatPassword",
                message = "The password and repeat password aren't the same"
        )
})
public class UserRegistrationRequestDto {
    @NotBlank
    @Email
    @Schema(example = "new@example.com")
    private String email;

    @NotBlank
    @Length(min = 8)
    private String password;

    @NotBlank
    @Length(min = 8)
    private String repeatPassword;

    @NotBlank
    @Schema(example = "John")
    private String firstName;

    @NotBlank
    @Schema(example = "Brown")
    private String lastName;

    private int age;
    private double weight;
    private String activityStatus;
}

import io.swagger.v3.oas.annotations.media.Schema;
import lombok.Data;

@Data
```

```
public class UserResponseDto {
    private Long id;

    @Schema(example = "new@example.com")
    private String email;

    private String firstName;

    private String lastName;
}

//ProductRepository
package ua.vntu.nolinefoodassistant.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import ua.vntu.nolinefoodassistant.model.Product;

public interface ProductRepository extends JpaRepository<Product, Long> {
}

//UserRepository
import java.util.Optional;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

@Repository
public interface UserRepository extends JpaRepository<User, Long> {
    @Query("FROM User u JOIN FETCH u.roles WHERE u.email = :email")
    Optional<User> findByEmail(@Param("email") String email);
}

// create-products-table.yaml
databaseChangeLog:
  - changeSet:
      id: create-products-table
      author: olexander-lypov
      changes:
        - createTable:
            tableName: products
            columns:
              - column:
                  name: id
                  type: bigint
```

```
            autoIncrement: true
            constraints:
              primaryKey: true
              nullable: false
        - column:
            name: name
            type: varchar(255)
            constraints:
              nullable: false
              unique: true
        - column:
            name: calories
            type: bigint
            constraints:
              nullable: false
        - column:
            name: protein
            type: bigint
            constraints:
              nullable: false
        - column:
            name: fat
            type: bigint
            constraints:
              nullable: false
        - column:
            name: carbohydrate
            type: bigint
            constraints:
              nullable: false
        - column:
            name: is_deleted
            type: boolean
            defaultValueBoolean: false
            constraints:
              nullable: false

// create-users-table.yaml
databaseChangeLog:
  - changeSet:
      id: create-users-table
      author: olexander-lypov
      changes:
        - createTable:
            tableName: users
```

```
        columns:
          - column:
              name: id
              type: bigint
              autoIncrement: true
              constraints:
                primaryKey: true
                nullable: false
          - column:
              name: email
              type: varchar(255)
              constraints:
                nullable: false
                unique: true
          - column:
              name: first_name
              type: varchar(255)
              constraints:
                nullable: false
          - column:
              name: last_name
              type: varchar(255)
              constraints:
                nullable: false
          - column:
              name: password
              type: varchar(255)
              constraints:
                nullable: false
          - column:
              name: is_deleted
              type: boolean
              defaultValueBoolean: false
              constraints:
                nullable: false
```

```
//create-roles-table
databaseChangeLog:
  - changeSet:
      id: create-roles-table
      author: olexandr-lypov
      changes:
        - createTable:
```

```
        tableName: roles
        columns:
          - column:
              name: id
              type: bigint
              autoIncrement: true
              constraints:
                primaryKey: true
                nullable: false
          - column:
              name: name
              type: varchar(255)
              constraints:
                nullable: false
                unique: true

//create-users-roles-table
databaseChangeLog:
  - changeSet:
      id: create-users-roles-table
      author: olexandr-lypov
      changes:
        - createTable:
            tableName: roles_users
            columns:
              - column:
                  name: user_id
                  type: bigint
                  constraints:
                    primaryKey: true
                    nullable: false
                    references: users(id)
                    foreignKeyName: fk_users_roles_user_id
              - column:
                  name: role_id
                  type: bigint
                  constraints:
                    primaryKey: true
                    nullable: false
                    references: roles(id)
                    foreignKeyName: fk_users_roles_roles_id


//application.properties
spring.datasource.url=jdbc:mysql://localhost:3306/no_line_food_assistant_clients_db
```

```
spring.datasource.username=root
spring.datasource.password=*****
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.open-in-view=false
server.servlet.context-path=/api

//Liquibase.properties
url=jdbc:mysql://localhost:3306/no_line_food_assistant_clients_db
username=root
password=root
changelog-file=/db.changelog/db.changelog-master.yaml

//pom.xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.2.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>ua.vntu</groupId>
    <artifactId>no-line-food-assistant</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>no-line-food-assistant</name>
    <description>no-line-food-assistant</description>
    <properties>
        <java.version>17</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

```
    <dependency>
      <groupId>org.liquibase</groupId>
      <artifactId>liquibase-core</artifactId>
    </dependency>

    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-devtools</artifactId>
      <scope>runtime</scope>
      <optional>true</optional>
    </dependency>
    <dependency>
      <groupId>com.mysql</groupId>
      <artifactId>mysql-connector-j</artifactId>
      <scope>runtime</scope>
    </dependency>
    <dependency>
      <groupId>org.projectlombok</groupId>
      <artifactId>lombok</artifactId>
      <optional>true</optional>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
        <configuration>
          <excludes>
            <exclude>
              <groupId>org.projectlombok</groupId>
              <artifactId>lombok</artifactId>
            </exclude>
          </excludes>
```

```
            </configuration>
          </plugin>
        </plugins>
    </build>

</project>

//TestClass sample from 4 topic
package ua.vntu.nolinefoodassistant.service;

import com.fasterxml.jackson.core.JsonProcessingException;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import org.mockito.InjectMocks;
import org.mockito.Mock;
import org.mockito.MockitoAnnotations;
import org.springframework.http.HttpEntity;
import org.springframework.http.HttpMethod;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.client.RestTemplate;
import ua.vntu.nolinefoodassistant.dto.ChatGptResponse;
import ua.vntu.nolinefoodassistant.dto.Message;
import ua.vntu.nolinefoodassistant.dto.ProductDto;
import ua.vntu.nolinefoodassistant.model.Choice;

import java.util.List;

import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.mockito.Mockito.*;
public class CalorieTrackerServiceImplTest {
    @Mock
    private RestTemplate restTemplate;
    @InjectMocks
    private CalorieTrackerServiceImpl calorieTrackerService;
    @BeforeEach
    public void setUp() {
        MockitoAnnotations.openMocks(this);
    }
    @Test
    public void testCalculateCalories() throws JsonProcessingException {
        String productName = "apple";
        String expectedResponse = "name: apple\ncalories: 52\nprotein: 0.3\nfat:
0.2\ncarbohydrate: 13.8\nadvice: Drink plenty of water throughout the day to stay
hydrated and help with weight loss.";
```

```
        ChatGptResponse mockResponse = new ChatGptResponse();
        Choice mockChoice = new Choice();
        Message mockMessage = new Message();
        mockMessage.setContent(expectedResponse);
        mockChoice.setMessage(mockMessage);
        mockResponse.setChoices(List.of(mockChoice));
        when(restTemplate.exchange(anyString(), eq(HttpMethod.POST),
any(HttpEntity.class), eq(ChatGptResponse.class)))
                .thenReturn(new ResponseEntity<>(mockResponse, HttpStatus.OK));
        ProductDto result = calorieTrackerService.calculateCalories(productName);
        assertEquals("apple", result.getName());
        assertEquals(52, result.getCalories());
        assertEquals(3, result.getProtein());
        assertEquals(2, result.getFat());
        assertEquals(14, result.getCarbohydrate());
        verify(restTemplate, times(1)).exchange(anyString(), eq(HttpMethod.POST),
any(HttpEntity.class), eq(ChatGptResponse.class));
    }
}
```

Додаток Г. Ілюстративна частина

ІЛЮСТРАТИВНА ЧАСТИНА

РОЗРОБКА ВЕБЗАСТОСУНКУ ON-LINE ІНТЕЛЕКТУАЛЬНОГО КОНСУЛЬТУВАННЯ ДЛЯ АНАЛІЗУ ХАРЧОВОЇ ЦІННОСТІ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVA, SPRING, JAVASCRIPT ТА REACT

Рисунок Г.1 – Титульний слайд

Рисунок Г.2 – Актуальність теми

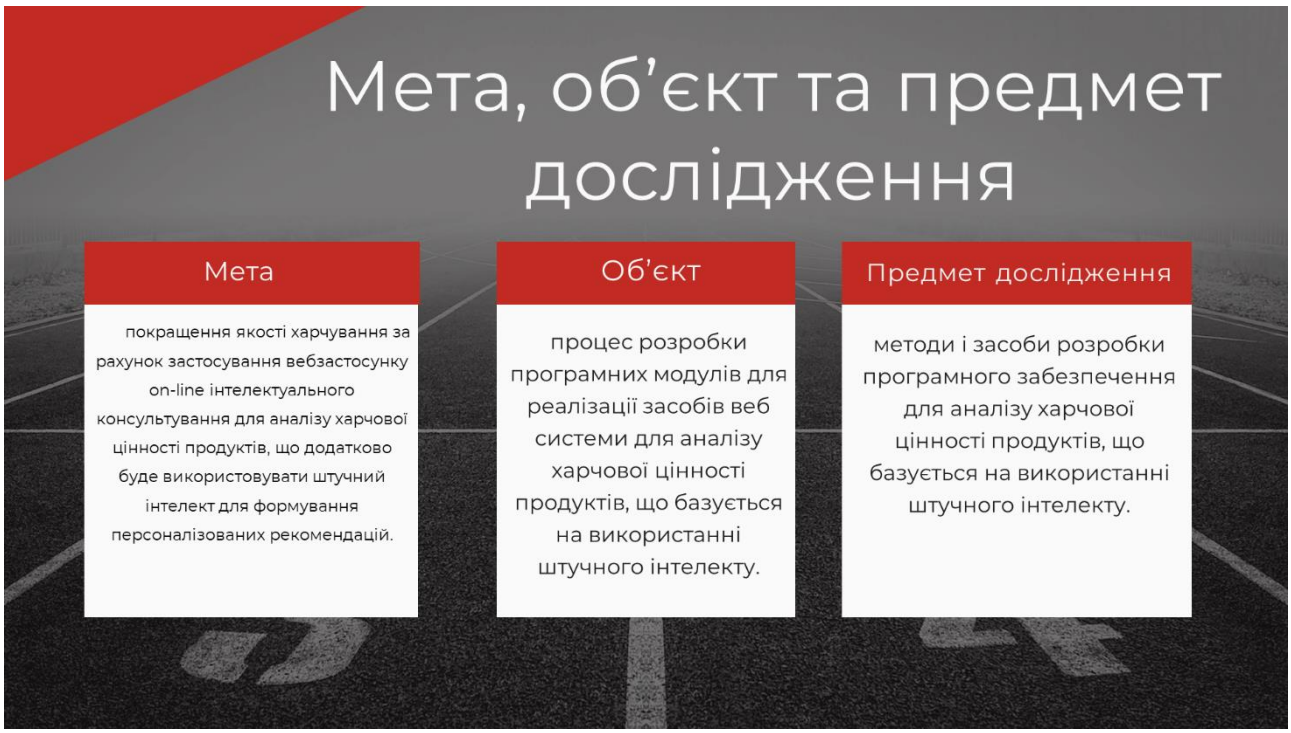


Рисунок Г.3 – Мета, об’єкт та предмет дослідження

Рисунок Г.4 – Задачі дослідження

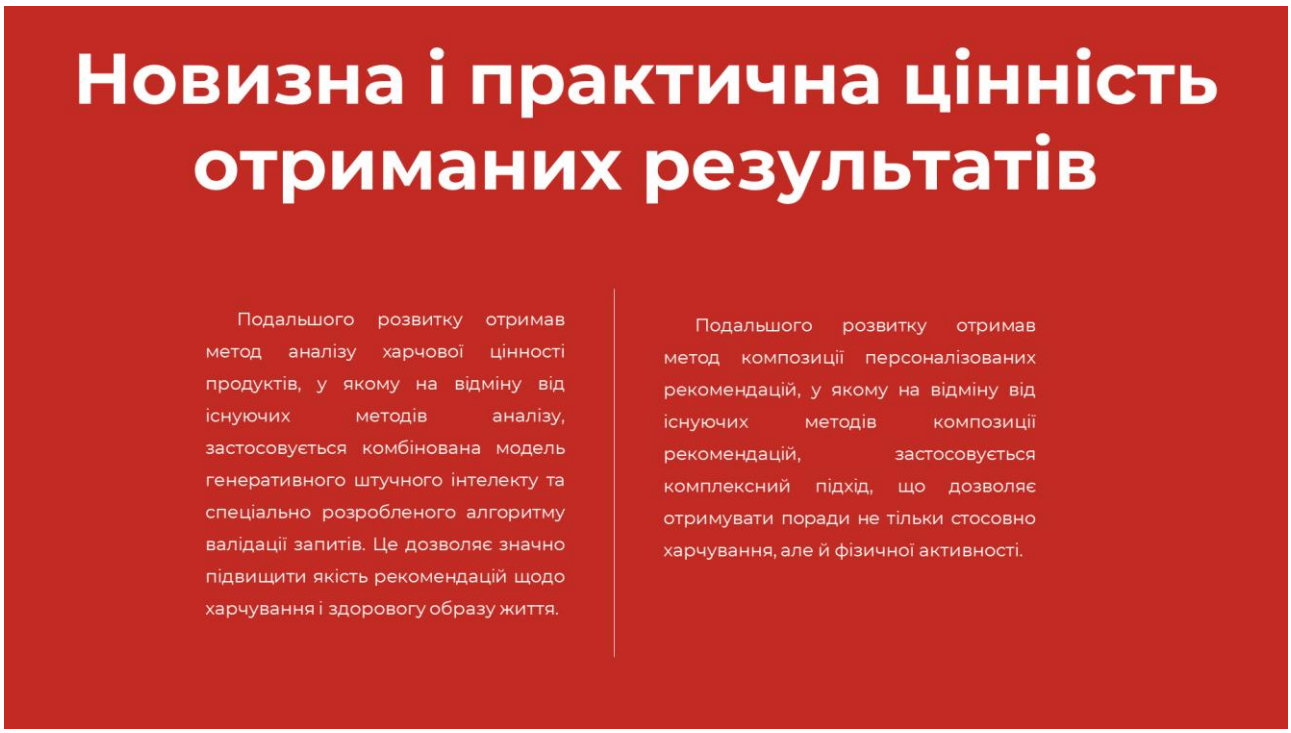


Рисунок Г.5 – Новизна і практична цінність отриманих результатів

Рисунок Г.6 – Порівняльний аналіз аналогів

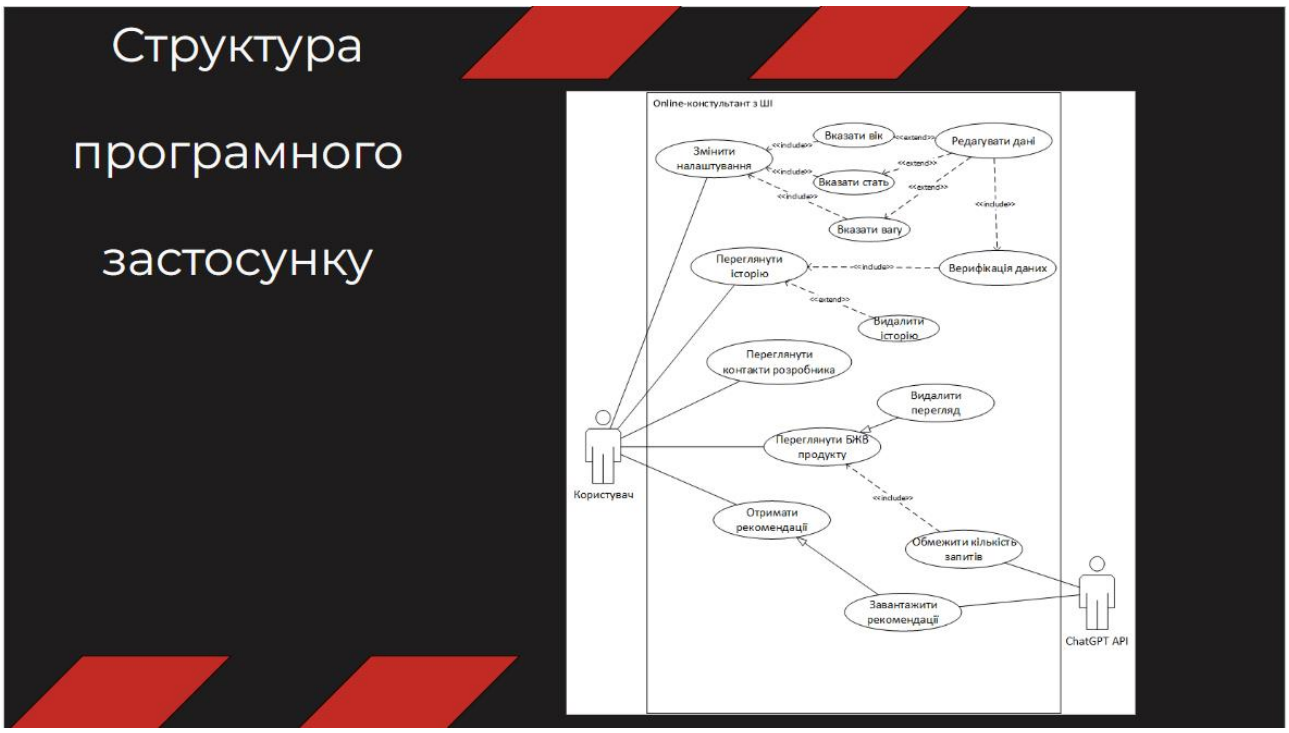


Рисунок Г.7 – Структура програмного застосунку

Рисунок Г.8 – Модель системи

Рисунок Г.9 – Загальний алгоритм роботи системи

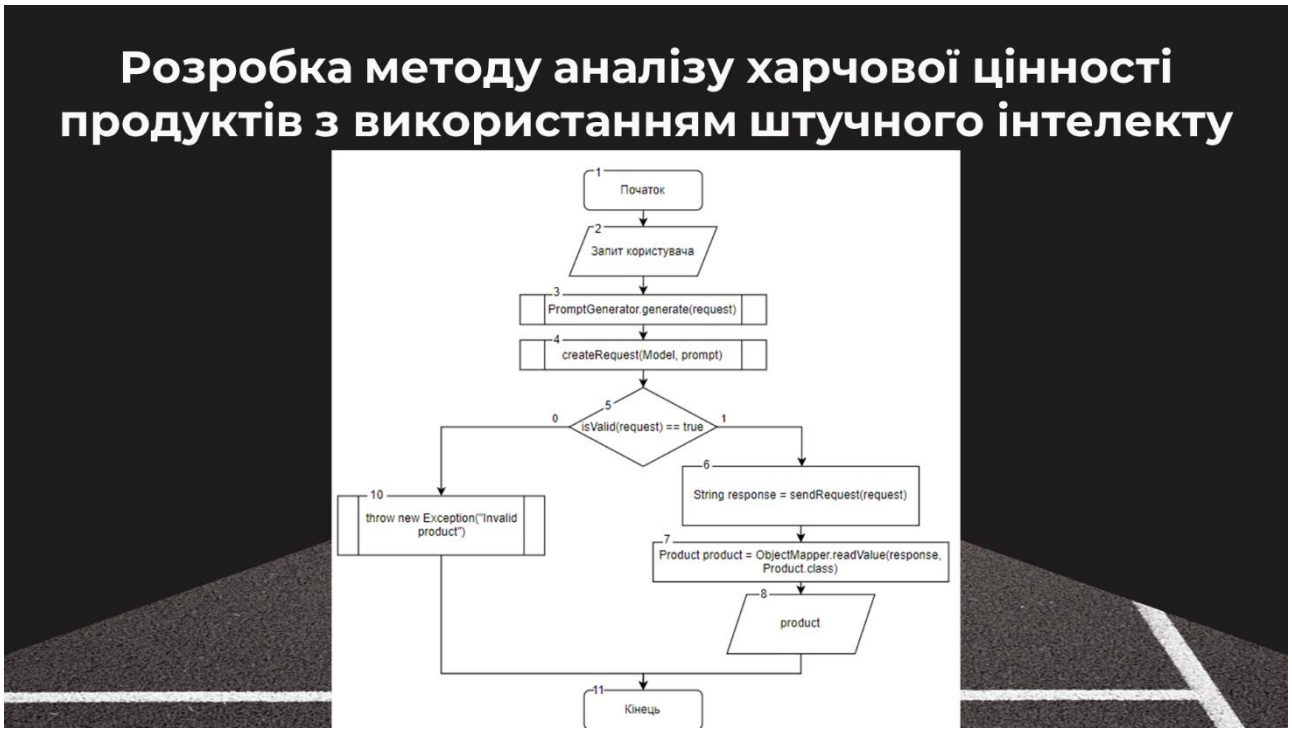


Рисунок Г.10 – Розробка методу аналізу харчової цінності продуктів з використанням штучного інтелекту

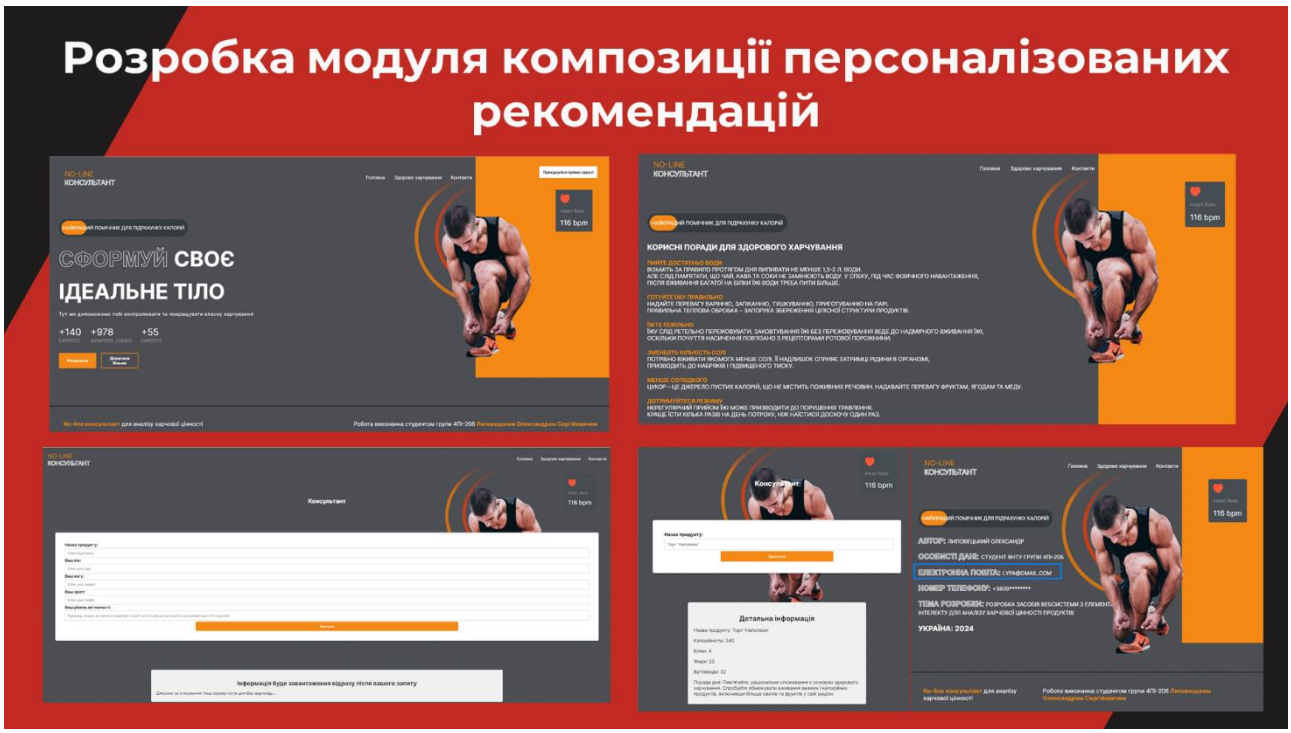


Рисунок Г.11 – Розробка модуля композиції персоналізованих рекомендацій

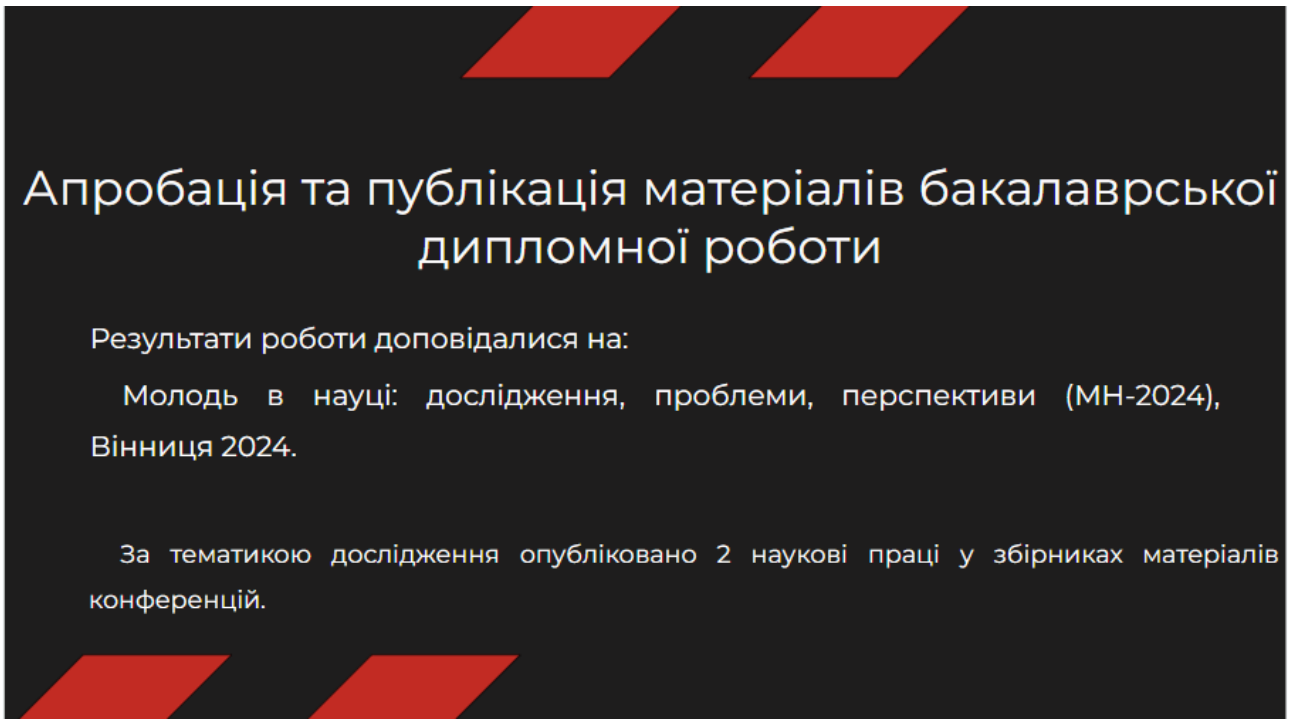


Рисунок Г.12 – Апробація та публікація матеріалів бакалаврської дипломної роботи