Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка веб-сервісу для оптимального вибору косметики засобами Python та PHP

Виконав: студент IV курсу, групи СТ-41
спеціальності 126 Інформаційні системи та технології
(шифр і назва спеціальності)

_____ Ігнат Ю.В.
(підпис) (прізвище та ініціали)

Керівник _____ Млинко Б.Б.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _червня_ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)
за спеціальністю 126 Інформаційні системи та технології
(шифр і назва спеціальності)
Студенту Ігнат Юлія Василівна
(прізвище, ім'я, по батькові)
1. Тема роботи Розробка веб-сервісу для оптимального вибору косметики Python та PHP
Керівник роботи Млинко Богдана Богданівна, кандидат технічних наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом ректора від «*07*» *травня* 2025 року № *4/7-445*
2. Термін подання студентом завершеної роботи 23 червня 2025 р.
3. Вихідні дані до роботи Літературні та інтернет джерела інформації щодо розробки веб-сервісу для оптимального вибору косметики.
4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз предметної області та постановка завдання на розробку веб-сервісу для оптимального вибору косметики. 1.1 Огляд використання цифрових технологій в косметології. 1.2 Огляд та аналіз підходів для створення веб-сервісів для підбору косметики. 1.3 Порівняльний аналіз існуючих веб-сервісів для вибору косметики. 1.4 Аналіз інформаційних методів та алгоритмів для веб-сервісу для оптимального вибору косметики.1.5 Технічна постановка задачі веб-сервісу для оптимального вибору косметики. 1.6 Вимоги до веб-сервісу та етапи його розробки. 1.7 Висновки до першого розділу. 2. Проєктування веб-сервісу для оптимального вибору косметики. 2.1 Вибір технологій та середовища розробки. 2.2 Архітектура та структура веб-сервісу. 2.3 Взаємодія між PHP та Python(Flask). 2.4 Клієнтська частина. 2.5 Серверна частина. 2.6 Структура бази даних. 2.7 Висновки до другого розділу. 3. Практична частина реалізації розробки веб-сервісу для оптимального вибору косметики. 3.1 Реалізація клієнтського інтерфейсу. 3.2 Заповнення бази даних. 3.3 Обробка даних PHP-скриптами. 3.4 Реалізація алгоритму зважених сум у середовищі Python. 3.5 Формування та відображення результатів користувачу. 3.6 Тестування функціоналу веб-сервісу. 3.7 Висновки до третього розділу. 4. Безпека життєдіяльності, основи охорони праці. 4.1 Психологічні фактори впливу на безпеку людини. 4.2 Вимоги електробезпеки при експлуатації персональних комп'ютерів в офісних приміщеннях. 4.3 Висновок до четвертого розділу. Висновки. Перелік джерел. Додатки.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Вступ. 2. Мета та завдання. 3. Порівняльний аналіз існуючих рішень. 4. Архітектура веб-сервісу. 5. Обрані інструменти для розробки. 6. Методи обробки та аналізу даних. 7. Структура бази даних. 8. Механізм обміну даними. 9. Користувацький інтерфейс. 10. Тестування веб-сервісу. 11. Практичне значення отриманих результатів. 12. Висновки. 13. Кінцевий слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Сенчишин В.С., доцент кафедри МТ		

7. Дата видачі завдання 07 травня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	07.05.2025	
2.	Підбір та опрацювання літературних джерел щодо	08.05.2025-15.05.2025	
	розробки веб-сервісів для вибору косметики		
3.	Розробка структури та визначення функціоналу веб-	16.05.2025-22.05.2025	
	сервісу для оптимального вибору косметики		
4.	Оформлення розділу «Аналіз предметної області та	23.05.2025-01.06.2025	
	постановка завдання на розробку веб-сервісу для»		
	оптимального вибору косметики»		
5.	Оформлення розділу «Проєктування веб-сервісу для	23.05.2025-01.06.2025	
	Оптимального вибору косметики»		
6.	Оформлення розділу «Практична частина реалізації	23.05.2025-01.06.2025	
	розробки веб-сервісу для оптимального вибору		
	косметики»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	02.06.2025-05.06.2025	
8.	Виконання завдання до підрозділу «Основи охорони	02.06.2025-05.06.2025	
	праці»		
9.	Оформлення кваліфікаційної роботи	06.06.2025-10.06.2025	
10.	Нормоконтроль	11.06.2025-13.06.2025	
11.	Перевірка на плагіат	16.06.2025	
12.	Попередній захист кваліфікаційної роботи	18.06.2025	

Студент _____ Ігнат Ю.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Млинко Б.Б.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка веб-сервісу для оптимального вибору косметики засобами Python та PHP // Кваліфікаційна робота освітнього рівня «Бакалавр» // Ігнат Юлія Василівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СТ-41 // Тернопіль, 2025 // С.77, рис. – 12, табл. – 15, додат. – 11, бібліогр. – 45.

Ключові слова: веб-сервіс, клієнт-серверна архітектура, Python, PHP, ранжування, обробка запитів.

Кваліфікаційна робота присвячена дослідженню та розробці веб-сервісу для оптимального вибору косметики. В першому розділі кваліфікаційної роботи описано сучасні тенденції у сфері веб-розробки, засобів реалізації клієнт-серверної архітектури та підходи до побудови експертних сервісів. Висвітлено обґрунтування вибору технологій, зокрема Python та PHP. Розглянуто принципи інтеграції між модулями сервісу, а також типові задачі фільтрації та ранжування.

В другому розділі кваліфікаційної роботи досліджено структуру та архітектуру веб-сервісу, а також його взаємодію між його основними компонентами. Досліджено алгоритм багатокритеріального аналізу – метод сум. Подано детальний опис логіки обробки даних.

В третьому розділі кваліфікаційної роботи описано практичну реалізацію веб-сервісу, від створення логіки обробки даних до виводу результатів. Проаналізовано результати функціонування веб-сервісу, проведено тестування та забезпечено узгоджену взаємодію між усіма модулями. Об'єкт дослідження: процес автоматизації оптимального вибору косметики. Предмет дослідження: методи обробки, фільтрації та ранжування даних засобами Python, PHP і MySQL.

ANNOTATION

Development of a Web Service for Optimal Cosmetic Selection Using Python and PHP // Qualification work of the educational level «Bachelor» // Ihnat Yuliia // Ternopil Ivan Pulyu National Technical University, Computer and Information Systems and Software Engineering Faculty, Computer Sciences Department, group ST-41 // Ternopil, 2025 // P. 77, fig. – 12, tabl. – 15, annexes. – 11, references – 45.

Keywords: web service, client-server architecture, Python, PHP, ranking, query processing.

The qualification work is dedicated to research and development of a web service for the optimal choice of cosmetics. The first section of the qualification work describes current trends in the field of web development, tools for implementing client-server architecture and approaches to building expert services. The rationale for the choice of technologies, in particular Python and PHP, is highlighted. The principles of integration between the modules of the service are considered, as well as typical tasks of filtering and ranking.

The second section of the qualification work explores the structure and architecture of the web service, as well as its interaction between its main components. The algorithm of multi-criteria analysis is investigated – the method of weighted coefficients. A detailed description of the data processing logic is provided.

The third section of the qualification work describes the practical implementation of the web service, from creating data processing logic to displaying results. The results of the functioning of the web service were analyzed, testing was carried out and coordinated interaction between all modules was ensured. Object of research: the process of automating the optimal choice of cosmetics. Subject of research: methods of processing, filtering and ranking data by means of Python, PHP and MySQL.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ І ТЕРМІНІВ

API (Application Programming Interface) – інтерфейс прикладного програмування; набір готових функцій для взаємодії між програмами.

БД – база даних; організована сукупність даних, що зберігається і обробляється за допомогою СУБД.

CSS (Cascading Style Sheets) – каскадні таблиці стилів, що визначають вигляд HTML-елементів.

Flask – легкий веб-фреймворк на мові Python, який використовується для створення веб-додатків.

HTML (HyperText Markup Language) – мова розмітки гіпертексту, основа веб-сторінок.

HTTP (HyperText Transfer Protocol) – протокол передавання гіпертексту.

JSON (JavaScript Object Notation) – формат обміну даними між клієнтом і сервером.

MySQL – система управління базами даних, яка використовує мову SQL.

PHP – мова сценаріїв на стороні сервера, що використовується для динамічного створення веб-сторінок.

Python – мова програмування високого рівня, що використовується зокрема для аналітики, автоматизації й бекенду.

REST – архітектурний стиль побудови веб-сервісів на основі протоколу HTTP.

SQL (Structured Query Language) – мова структурованих запитів до баз даних.

ЗМІСТ

ВСТУП

Актуальність теми. Внаслідок глобалізації та активного впровадження інтернет-засобів у повсякденне життя зростає потреба у персоналізованому підході для вибору товарів та послуг [1]. Створення таких веб-сервісів, дозволяє покращити ефективність взаємодії між користувачами та продуктами. Тому створення подібних IT-рішень є актуальним напрямком сучасних досліджень в галузі веб-технологій та експертних систем.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи є розробка веб-сервісу, який надає індивідуальний підбір порад та косметичних компонентів, на основі аналізу особистих параметрів користувача.

Для досягнення поставленої мети потрібно виконати ряд завдань, зокрема:

– проаналізувати стан існуючих веб-сервісів схожого призначення;

– розробити структуру веб-сервісу з урахуванням особливостей інтеграції між Python та PHP;

– розробити модель для збору і попередньої обробки даних користувача;

– реалізувати алгоритм для оптимального вибору косметики;

– виконати тестування та оцінку ефективності роботи веб-сервісу;

– оцінити можливості масштабування та подальшого розвитку системи;

Практичне значення одержаних результатів. Практичне значення результатів даної роботи полягає у створені веб-сервісу з інструментами для персоналізованого вибору косметичних продуктів, який може бути інтегрований у платформи інтернет-магазинів, дерматологічних сервісів або мобільних додатків. Запропонована система демонструє ефективне використання інформаційних технологій для побудови персоналізованих рекомендаційних систем

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ НА РОЗРОБКУ ВЕБ-СЕРВІСУ ДЛЯ ОПТИМАЛЬНОГО ВИБОРУ КОСМЕТИКИ

1.1 Огляд використання цифрових технологій в косметології

Сучасні компанії активно застосовують інформаційні технології – веб-сайти, мобільні додатки, онлайн-чати, рекомендаційні системи та інші веб-застосунки для покращення користувацького досвіду [2]. Одними з таких, є веб-сервіси для оптимального вибору косметики, які аналізують індивідуальні дані користувача та надають рекомендації на основі різноманітних алгоритмів для прийняття рішень. Такі веб-сервіси можуть працювати із моделями та методами багатокритеріального аналізу.

За даними аналітичного порталу Statista [3], ринок продукції з догляду за шкірою демонструє демонструє стабільне зростання, зумовлене підвищеним інтерес споживачів до здорового способу життя, профілактики проблем шкіри та індивідуального підходу до вибору косметичних засобів.

На глобальному рівні подібні рішення стали стандартом для електронної комерції – від віртуальних помічників на сайтах до складних рекомендаційних модулів у мобільних застосунках. Сучасні платформи електронної комерції активно інтегрують такі інструменти у свої веб-сайти для підвищення лояльності клієнтів. На українському ринку таких рішень поки що небагато, або вони мають обмежену функціональність, орієнтовану на певний бренд або лінійку продукції. Це створює передумови для розробки універсального незалежного веб-сервісу, який надає поради базуючись на індивідуальних особливостях користувача.

Таким чином, застосування сучасних веб-технологій, аналітичних алгоритмів та принципів персоналізації дозволяє створювати ефективні інформаційні системи, що сприяють розвитку цифрових рішень.

1.2 Огляд та аналіз підходів для створення веб-сервісів для підбору косметики

Користувачі можуть стикатися з труднощами при виборі продукції, що відповідає їхнім потребам. В умовах активного впровадження інформаційних технологій в повсякденне життя, це створює підвищений попит на веб-сервіси здібні автоматично аналізувати дані та формувати рекомендації [4].

У сучасній практиці можна виділити кілька підходів до реалізації таких веб-сервісів.

Анкето-опитувальний підхід, коли користувач проходить онлайн-опитувальник, у якому вибирає відповіді відповідно до своїх характеристик та побажань. На основі отриманих відповідей веб-сервіси підбирають відповідні продукти та засоби. Перевагою такого підходу є проста реалізація та доступність. Ефективність такого підходу залежить від якості питань та логіки аналізу відповідей.

Фото аналіз є одним із методів, який використовують деякі веб-сервіси. Він надає можливість виявити складніші параметри, завдяки чому забезпечує більше точність.

Веб-сервіси на базі штучного інтелекту, які базуються на алгоритмах машинного навчання. Вони здатні навчатися на великих обсягах даних, що дає змогу строювати та покращувати моделі аналізу та рекомендацій.

Експертні сервіси базуються на заздалегідь сформованих правилах та відповідно станів, які імітують логіку мислення фахівця. Це дозволяє системі надавати логічно обґрунтовані рекомендації без потреба в глибоких обчисленнях.

Попри різноманітність методів, більшість рішень мають певні обмеження:

– обмежений набір параметрів, які враховуються в аналізі;
– залежність від конкретного бренду або виробника;
– відсутність прозорої логіки формування рекомендацій;

Окрім того, багато веб-сервісів акцентують більшу увагу на маркетинговій складовій, такій як, просування продукції, а не на реальних потребах користувача [5]. Це приводить до зниження ефективності підбору та результативності рекомендацій.

1.3 Порівняльний аналіз існуючих веб-сервісів для вибору косметики

Для формування вимог до веб-сервісу для оптимального вибору косметики, було здійснено порівняльний аналіз існуючих веб-сервісів, що стосуються догляду за шкірою та підбору косметичних продуктів. Результати аналізу наведено у таблиці 1.1.

Таблиця 1.1 – Порівняльна таблиця веб-сервісів для вибору косметики

	Назва веб-сервісу			
Критерій	**SafetyMakeup**	**eCosmeticProf**	**EczemaWise**	**Troveskin**
Цілі використання	Аналіз інгредієнтів	Онлайн консультації продаж засобів	Відстеження симптомів екземи	Визначення типу шкіри та поради
Індивідуальний підхід	Відсутній	Повноцінний	Частковий	Повноцінний
Рекомендації продуктів	Відсутні	Присутні	Відсутні	Присутні
Додаткові функції	Поєднання інгредієнтів	Подальша підтримка користувача	Відстеження дієти, мапа шкіри	Збереження фото
Обмеження	Лише склад, без типу шкіри	Орієнтація тільки на один бренд	Тільки поради стосовно екземи	Платний функціонал, мало параметрів

Жоден з розглянутих сервісів не поєднує одразу кількох важливих аспектів, таких як:

– аналіз індивідуальних параметрів користувача (вік, тип шкіри, проблеми, бюджет, алергени);

– підбір доглядових засобів різних брендів;

– врахування взаємодії кількох продуктів;

– гнучка та автоматизована логіка підбору.

Таким чином, постає завдання розробити веб-сервіс, який поєднує інтуїтивно зрозумілий та легкий інтерфейс, та гнучку логіку аналізу, яка не залежить від конкретних брендів та забезпечує прозорість та обґрунтованість рекомендацій. Саме такий підхід дозволяє підвищити якість обслуговування користувачів у сфері веб-технологій.

1.4 Аналіз інформаційних методів та алгоритмів для веб-сервісу для оптимального вибору косметики

Для побудови ефективного веб-сервісу для оптимального вибору косметики важливо застосувати відповідні інформаційні методи та алгоритми, що забезпечують логічний та адаптивний вибір продуктів, відповідно до індивідуальних характеристик та даних користувача [6].

Для веб-сервісу можливо використовувати комбінації методів, що базуються на багакритеріальному аналізі.

На першому етапі часто використовується метод початкової фільтрації [7]. Цей метод дозволяє швидко звузити діапазон пошуку до ревалентних варіантів, виключаючи ті, що однозначно не підходять користувачу. Початкова фільтрація мінімізує обсяг даних для подальшого аналізу, підвищує швидкодію та продуктивність веб-сервісу.

Наступним етапом застосовується метод із більш глибоким аналізом даних, який шукає найоптимальніший варіанти. Одним з таких є метод вагових сум. Він дозволяє ранжувати вже відфільтровані варіанти, щоб вибрати

найкращі. Цей метод є простим та зрозумілим, забезпечує гнучкість, а також дає кількісний результат – оцінку [8].

Така комбінація методів має низку переваг. Головною перевагою є оптимальна швидкість та продуктивність, завдяки початковій фільтрації зменшується обсяг даних для подальшої обробки складнішим методом багатокритеріального аналізу. Також, гнучкість персоналізації, оскільки уподобання та пріоритетність користувача адаптується під нього. Завдяки цьому веб-сервіс дає максимально ревалентні та персоналізовані рекомендації, що підвищують лояльність користувачів. Не менш важливою перевагою є простота реалізації і підтримки, бо поєднання цих двох методів дозволяє реалізувати ефективний та зрозумілий алгоритм, без надмірної складності.

Таким чином, інформаційні методи та алгоритми забезпечують збалансованість між швидкодією, точністю та гнучкістю веб-сервісу, що робить їх оптимальними для реалізації.

1.5 Технічна постановка задачі веб-сервісу для оптимального вибору косметики

Метою даної роботи є створення веб-сервісу для оптимального вибору косметики на основі параметрів користувача [9]. Рішення орієнтоване на покращення персоналізації рекомендації та підбору найкращих продуктів для користувача.

Проєктований веб-сервіс повинний реалізувати наступні функціональні можливості:

– прийом відповідей, вихідних даних, користувача за допомогою веб-форми з анкетою, яка містить необхідна параметри та критерії;

– попереднє фільтрування продуктів відповідно до вказаних параметрів;

– передача відібраного (відфільтрованого) набору продуктів і вагових сум до окремого модуля аналізу через HTTP API;

– обробка даних у модулі аналізу, ранжування засобів за допомогою методу зважених сум, враховуючи користувацькі параметри та пріоритети;

– виведення результатів в інтерфейсі веб-сайту у зрозумілому вигляді.

Вхідними даними є анкета користувача, яка включає параметри косметичних продуктів, побажання та обмеження, потреби користувача.

База косметичних засобів, яка містить необхідні параметри для опису засобу, такі як назва, типу продукту, доступність, бренд, тощо.

Вихідними даними є список рекомендованих продуктів, засоби що відповідають бюджету та обмеженням користувача, та оцінка відповідності до параметрів в балах, на основі вагового аналізу.

Обмеження стосуються індивідуальних вимог до властивостей, характеристик.

Розробка веб-сервісу вимагає чіткого розуміння архітектурних складових [10]. Ці вимоги описано у таблиці 1.2.

Таблиця 1.2 – Технічна постановка завдання

Категорія	**Інструменти**	**Призначення**
Фронтенд	HTML, CSS, JS, PHP	Веб-форма та сторінка з оформленими результатами, початкова фільтрація продуктів, відправка відповідей.
Бекенд	Python (Flask)	Прийом відповідей з анкети через API, аналіз відповідей методом вагових сум, відправка результатів аналізу.
База даних	MySQL	Зберігання інформації про косметичні продукти

Таким чином було описано основні технічні вимоги для роботи веб-сервісу для оптимального вибору косметики.

1.6 Вимоги до веб-сервісу та етапи його розробки

Для досягнення поставленої мети та реалізації задачі для розробки веб-сервісу для оптимального вибору косметики на основі параметрів та критеріїв користувача, сформульовано перелік функціональних та нефункціональних вимог до веб-сервісу [11]. А також, визначено ключові етапи його розробки. Веб-сервіс має бути зручним для кінцевого користувача, ефективним в обробці даних та відкритим для подальшого масштабування.

Функціональні вимоги:

– зручний користувацький інтерфейс для заповнення анкети з параметрами та відображення результатів підбору;

– логіка первинної фільтрації засобів, відповідно до заданих параметрів, обмежень, критеріїв та побажань користувача;

– надсилання, обробка та прийом відповідей між основними архітектурними компонентами – клієнтом, сервером та базою даних, з використанням REST API та форматом JSON;

– ранжування продуктів методом вагових сум за ревалентністю.

Нефункціональні вимоги:

– сумісність із основними веб-браузерами та мобільними пристроями;

– висока швидкість та ефективність обробки запитів;

– простота масштабування бази даних з можливістю додавання нових продуктів без потреби змінювати БД;

– розширюваність із підтримкою для додавання нових фільтрів та логіки оцінювання.

Основні етапи розробки веб-сервісу [12]:

1) Аналіз предметної області, який включає вивчення потрібних параметрів та критеріїв для підбору косметики

2) Проєктування структури веб-сервісу, який включає вибір архітектури, мов програмування та формату обміну даними.

3) Розробка клієнтської частини, створення HTML розмітки, веб-форми, збір даних.

4) Реалізація серверної логіки, побудова обробки відповідей, аналіз методом вагових коефіцієнтів, та ранжування отаманих результатів на Python (Flask).

5) Створення, розробка структури та наповнення бази даних із косметичними продуктами.

6) Тестування веб-сервісу, яке включає налагодження взаємодії клієнта та сервера та перевірку коректності результатів.

7) Аналіз результатів та оптимізація, яке може охоплювати збір зворотного зв’язку, оцінку рекомендацій та коригування правил фільтрації.

Таким чином, розробка веб-сервісу передбачає поєднання веб-програмування, логіки обробки даних та робити із базами даних.

1.7 Висновки до першого розділу

У першому розділі було проведено обґрунтований аналіз предметної області, пов’язаних з використанням інформаційних технологій у сфері косметології. Розглянуто тенденції розвитку персоналізовані онлайн-сервісів.

Було розглянуто технічні підходи до створення веб-рішень, що поєднують алгоритми обробки даних та аналіз бази косметичних засобів. Було проаналізовано ключові інформаційні моделі, які застосовуються для багатокритеріального оцінювання альтернатив, зокрема метод зважених сум.

У результаті проведеного аналізу було сформульовано технічну постановку задачі, вимоги до веб-сервісу, ключові етапи його реалізації. Визначено функціональні та нефункціональні вимоги.

Таким чином, результати першого розділу заклали теоретичну та практичну основу для проєктування та реалізації веб-сервісу для оптимального вибору косметики.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБ-СЕРВІСУ ДЛЯ ОПТИМАЛЬНОГО ВИБОРУ КОСМЕТИКИ

2.1 Вибір технологій та середовища розробки

Проєктування веб-сервісу для оптимального вибору косметики вимагає ретельного підбору технологій, що здатні забезпечити швидку обробку даних, стабільну роботу, зручний користувацький інтерфейс та можливість масштабування в майбутньому. Вибір технологій здійснювався з врахуванням наступним ключових критеріїв вибору:

– простота реалізації архітектури та логіки обробки даних;

– підтримка JSON-формату для обміну даними між клієнтом та сервером;

– можливість працювати з базами даних та зовнішніми API;

– сумісність з поширеними хостингами.

Дані критерії враховують вимоги до функціоналу, зручності реалізації, швидкості обробки даних та можливості масштабування.

2.1.1 Python для обробки і аналізу даних

Для побудови серверної логіки для розрахунку найоптимальніших продуктів методом вагових коефіцієнтів, було обрану програмування Python із мікро фреймворком Flask.

Python – одна з найбільш популярних, універсальних та гнучких мова, що широко використовується в наукових обчисленнях та аналізі даних [13]. Її синтаксис є лаконічним та інтуїтивно зрозумілим, що сприяє швидкій реалізації складних алгоритмів.

Ключовою перевагою є те, що Python має велику екосистему бібліотек для обробки даних. Завдяки цьому, розробка алгоритмів стає більш гнучкою. Python відомий своєю масштабованістю, від невеликих проектів до

високонавантажених систем, він дозволяє ефективно розширювати функціонал проєкту без радикальних змін у серверній частині. Ще однією, перевагою Python є ефективність обробки даних, він добре працює з асинхронними та паралельними обчисленнями.

Flask – легкий мікро фреймворк створений спеціально для швидкої побудови веб-додатків [14]. Його ключовими перевагами є:

– підтримка REST-архітектури, що дозволяє швидко організувати обробку запитів типу POST та GET;

– вбудований сервер для локального тестування;

– простий механізм маршрутизації;

– гнучка робота з JSON-даними, включно з функціями отримання даних та відправки результатів;

– мінімалістична структура, що не вимагає складних конфігурацій.

Такий вибір інструментів для побудови серверної логіки дозволяє реалізувати проєкт максимально оптимально та функціонально.

2.1.2 PHP та фронтенд технології для роботи з користувачем

Для реалізації клієнтської частини було обрано класичний стек інструментів для створення та верстки веб-сторінок – HTML, CSS та JS. Ці технології є базовими та універсальними для фронтенд-розробки, що дозволяє забезпечити сумісність із більшістю браузерів, швидке завантаження сторінок та простоту підтримки інтерфейсу. Завдяки їм легко реалізувати інтуїтивно зрозумілий та адаптивний користувацький інтерфейс.

Обробка форми, надсилання даних та зв’язок із серверною частиною виконуються за допомогою PHP. Вибір PHP як серверного модуля обґрунтований рядом суттєвих переваг:

– розгортання клієнтської частини без додаткових складних налаштувань, оскільки PHP підтримується на більшості хостингів;

– висока швидкість та ефективність обробки HTTP-запитів, швидкий парсинг та підготовка даних для передачі через POST-запити;

– функціональні можливості для ефективної роботи з базою даних MySQL, оскільки PHP має багатий набір вбудованих функцій для безпечного підключення, виконання SQL-запитів і обробки результатів;

PHP виступає надійним «посередником», що обробляє запити від клієнта, контролює валідацію, виконує логіку роботи з БД і передає та приймає дані від серверної частини на Python. Це знижує навантаження на сервер додатку Python, оскільки відокремлюється основна логіка аналізу та обчислень, від клієнтської частини.

2.1.3 MySQL для зберігання даних

У якості сховища даних, для організації зберігання даних про косметичні засоби було обрано реляційну систему керування базами даних MySQL. MySQL є однією з найпопулярніших систем керування базами даних у світі. Цей вибір ґрунтується на критично важливих перевагах, які забезпечують надійність, масштабованість та зручну роботу з даними для веб-сервісу для оптимального вибору косметики. До ключових переваг MySQL [15], що зумовили її вибір, належать:

– широке розповсюдження та стабільна підтримка, оскільки MySQL, що гарантує простоту розгортання;

– тісна інтеграція з php через розширення mysqli та pdo, що дозволяє безпечно виконувати запити та гнучко працювати з даними;

– підтримка складних запитів та оптимізація, що включає використання індексів, фільтрації та сортування враховуючи критерії;

– ізоляція компонентів, що надає можливість відокремити збережені дані в базі даних від логіки оброки на сервері, що робить БД окремим модулем, яким можна оперувати незалежно;

– гнучкість у розширені структури, при необхідності база даних легко розширюється без додавання нових характеристик чи типів даних, що робить систему адаптивною.

MySQL забезпечує надійне та адаптивне зберігання даних, сприяючи стабільній роботі та якості веб-сервісу для оптимального вибору косметики.

2.1.4 Середовище розробки

Для розробки веб-сервісу було обрано зручне та сучасне середовище, яке забезпечує ефективність на всіх етапах створення проєкту. Основним інструментом для програмування є Visual Studio Code. Visual Studio Code є легким, швидким та дуже популярним редактором коду. Його інтуїтивний інтерфейс і велика кількість розширень для різних мов програмування значно полегшують написання, налагодження та підтримку коду. Вбудований термінал дозволяє запускати код безпосередньо в редакторі, що прискорює цикл розробки та тестування. Важливою перевагою є можливість налагодження у VS CODE, що допомагає оперативно знаходити і виправляти помилки, підвищують якість та стабільність програми.

Для хостингу серверної частини було обрано платформу PythonAnywere. PythonAnywere – це хмарний сервіс, спеціально адаптований для запуску Python-додатків у веб-середовищі. Його вибір зумовлений підтримкою Flask-додатків, без потреби у складному налаштуванні серверного середовища. Також серед переваг можна виділити веб-інтерфейс для управління кодом та консоллю, файлами та залежностями, власний домен, вбудовані логи [16].

Для розміщення клієнтської частинини з PHP-обробкою, був використаний безкоштовний веб-хостинг FreeNF, який підтримує виконання PHP-скриптів, надає вбудований сервер для роботи з БД MySQL.

У підсумку, обраний стек технологій дозволяє стабільний та якісний, з налагодженою взаємодією всіх компонентів, веб-сервіс для оптимального вибору косметики, що сприяє досягненню цілей.

2.2 Архітектура та структура веб-сервісу

Архітектура веб-сервісу для оптимального вибору косметики побудована за класичною клієнт-серверною моделлю [17]. Вона передбачає розділення всієї логіки роботи веб-сервісу на три самостійні, але взаємопов’язані рівні: рівень інтерфейсу (користувача), рівень обробки (серверу) та рівень зберігання. Такий підхід дозволяє чітко відокремити відповідальність кожного елемента, забезпечує гнучкість у розвитку веб-сервісу, та полегшує підтримку і масштабування.

На першому, інтерфейсному, рівні відбувається початкова взаємодія користувача з веб-сервісом. Користувач взаємодіє через головну сторінку, форму з анкетою, та сторінку з результатами. Веб-форма містить анкету із варіантами відповідей для різних параметрів та критеріїв що стосуються косметичних продуктів для догляду за шкірою. На даному рівні не виконується складна обробка даних, лише анкета передає дані для подальшої обробки.

Після заповнення анкети, активується другий рівень, рівень обробки. Отримані дані з анкети передаються на окремий сервер, реалізований у середовищі Python, із використанням фреймворку Flask. Цей сервер відповідає за основну логіку функціонування веб-сервісу:

– прийом, обробка та надсилання відповідей у форматі JSON;
– аналіз та обчислення результату методом вагових коефіцієнтів;
– ранжування до найоптимальнішого продукту;

Ці результати передаються на клієнтську сторону, де починає працювати наступний етап – рівень зберігання. Веб-сервіс формує запит до БД, у якій міститься підготовлені таблиці з необхідною інформацією про косметичні продукти. Отримані із БД дані використовуються для формування фінального набору рекомендованих продуктів, найбільш ревалентних для користувача.

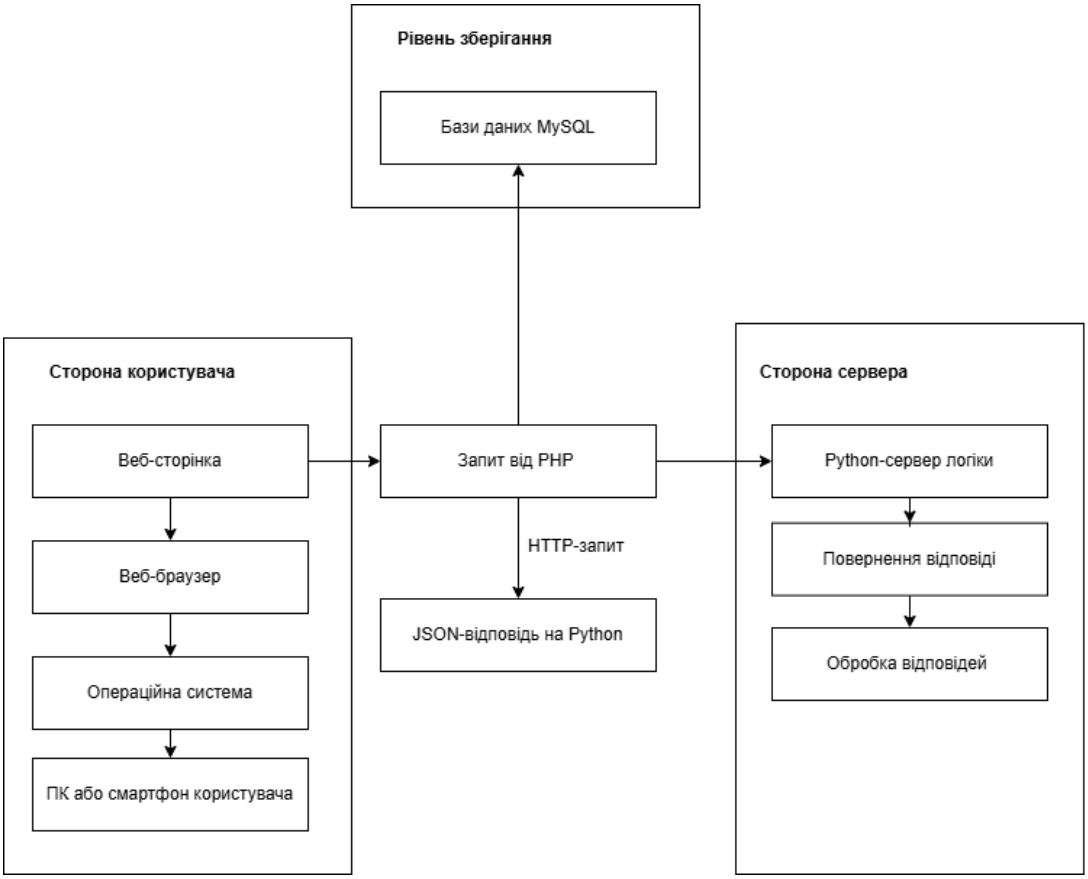


Рисунок 2.1 – Клієнт-серверна архітектура веб-сервісу

Взаємодія між трьома рівнями архітектури веб-сервісу здійснюється послідовно, з чітким розмежуванням обов'язків кожного компоненту. Користувач починає роботу із заповнення форми на клієнтській частині сайту. У формі анкета містить набір питань, що охоплюють індивідуальні параметри шкіри, особисті вподобання, наявність чи бажана відсутність певних компонентів, бажана ціна та пріоритетність для вибору косметики. Після того як дані зібрані, PHP-скрипт виконує первинну фільтрацію на основі зазначених параметрів та критеріїв. Далі сформований список відібраних продуктів, а також параметри пріоритетності критеріїв продукту, передаються на сервер логіки, який реалізований на Python. На ньому проводиться обчислення методом вагових коефіцієнтів та ранжування продуктів. У результаті користувач отримує відповідь у зручному вигляді на сторінці з результатами.

Таке логічне розділення веб-сервісу на рівні, забезпечує низку важливих переваг, що є критичними для підтримки проєкту [18].

По-перше, кожен рівень може розгортатися окремо, що важливо для збільшення веб-сервісу, перенесення системи тощо. Це дозволяє швидко реагувати на навантаження та додавати ресурси.

По-друге, гнучкість при зміні алгоритмів аналізу або логіки обробки, не потрібно вносити змінити у клієнтський інтерфейс. Це спрощує оновлення та дозволяє мінімізувати кількість помилок.

По-третє, ізольованість логіки. Якщо виникнуть технічні збої або проблеми на стороні інтефейсу, сервер обробки залишиться стабільними і не залежать від клієнтської частини.

По-четверте, завдяки розподіленій структурі будь-який компонент системи може бути змінений, оновлений або доповнений без негативного впливу на інші частини проєкту. Це відкриває широкі можливості для поступного покращення проєкту.

2.3 Взаємодія між PHP та Python(Flask)

Взаємодія між клієнтською частиною веб-сервісу, реалізованою на PHP, та серверною частиною, реалізованою через фреймворк Flask на Python – є центральним елементом архітектури системи. Такий підхід довзоляє чітко розділити обов’язками між двома модулями. PHP відповідає за інтерфейс, первинну фільтрацію, збір даних та вивід результатів. Flask на Python зосереджений виключно на обробці даних та аналізі методом вагових коефіцієнтів.

Це розподілення базується на принципах REST-архітектури, де між компонентами встановлюється взаємодія через чітко визначення HTTP-запити [19]. Для передачі даних використовуються формат JSON, що забезпечує просту та ефективну передачі структурованої інформації маж двома середовищами з різними мовами програмування. У таблиці 2.1. показано основні етапи підготовки та обробки відповідей.

Таблиця 2.1 – Основні етапи обробки відповідей

Крок	Опис дії	Технічна деталь
1	Прийом даних із HTML-форми	Зчитування масиву відповідей через метод $_POST
2	Групування та попередня обробка	Формування масиву із ключами та критеріями продуктів
3	Початкова фільтрація	Запит до бази даних MySQL для початкової фільтрації
4	Підготовка до передачі	Створення масиву відібраних продуктів та масиву із вагами критеріїв
5	Серіалізація в формат JSON	Перетворення масиву у формат JSON
6	Відправка запиту до Python-серверу	Передача JSON-списку через HTTP POST-запитом за допомогою curl
7	Розрахунок результату	Обробка відповіді, обчислення та отримання результату із ID та балами
8	Запит до БД для виводу інформації	Додаткове звернення до БД для отримання повної інформації про рекомендовані засоби
9	Вивід результатів	Динамічне формування веб-сторінки з результатами, із переліком засобів та їх оцінок

Дана таблиця описує послідовність дій у процесі взаємодії між модулями – від отримання запиту, його передання на серверну частину, первинної фільтрації та аналітичної обробки, до то повернення структурованого результату клієнту.

2.3.1 Обмін даними між модулями

Веб-сервіс реалізований як інтегрована система, що містить два основних компоненти – клієнтську та серверну частину. Взаємодія між якими,

організована через стандартизований механізм обміну даними за допомогою HTTP POST-запитів із форматом передачі JSON [20].

Після того, як користувач заходить на головну сторінку веб-сервісу та заповнює анкету з параметрами та критеріями, PHP-скрипт отримує введені дані. На цьому етапі виконується первинна фільтрація – із загального набору відсіюються ті, що не відповідають базовим вимогам користувача.

Відфільтрований перелік продуктів, разом із вагами параметрів які задають ступінь важливості окремих характеристик, упаковуються в структурований JSON-формат. Цей формат універсальний та забезпечує сумісність між клієнтом та сервером. PHP виконує роль клієнта, надсилаючи сформований JSON-список у POST-запит на Python-сервер, який відповідає за глибокий аналіз та ранжування.

Обробка запитів відбувається асинхронно, для забезпечення високої продуктивності веб-сервісу і можливості обслуговування великої кількості користувачів.

2.3.2 Обробка запиту на Python-сервері

Після отримання JSON-списку з ваговими коефіцієнтами та попередньо відібраними даними, проводиться детальний аналіз на Python.

Алгоритм виконує багатокритеріальну оцінку, у результаті якої визначаються найоптимальніші варіанти та формуються рекомендації із ранжуванням за відповідністю. Всі обчислення та логічні операції реалізовані у вигляді окремих методів.

Після завершення обробки знову формується JSON-об’єкт із відповідями, що містить ключові результати – ID продукту та його оцінка. Така мінімалістична та структурована відповідь дозволяє серверу на PHP легко та швидко її обробити і використовувати для подальшої роботи.

2.3.3 Вивід результатів та інтеграція

Отримавши JSON-відповідь від Python-сервера, PHP-скрипт розпаковує її для подальшої обробки. Виходячи з отриманих результатів, а саме оцінки та ID продукту, PHP формує SQL-запит до бази даних MySQL, де зберігається повна інформація про косметичні продукти. Ці дані динамічно виводяться на сторінці результатів у вигляді структурованого набору різноманітних продуктів, із зазначеними балами.

Такий розподіл ролей між PHP з клієнтським інтерфейсом, роботою з базою даних та початковою фільтрацією; та Python що відповідає за складку аналітичну логіку і ранжування – забезпечує високу надійність, масштабованість і зручність підтримки системи. Ця координація між компонентами робить веб-сервіс максимально адаптивним і зручним для користувача, а також дає зможу без проблем розширювати функціонал без суттєвих змін у вже існуючій архітектурі.

На рисунку 2.2 представлена діаграма послідовності, що ілюструє описану логіку обміну між модулями PHP та Python.

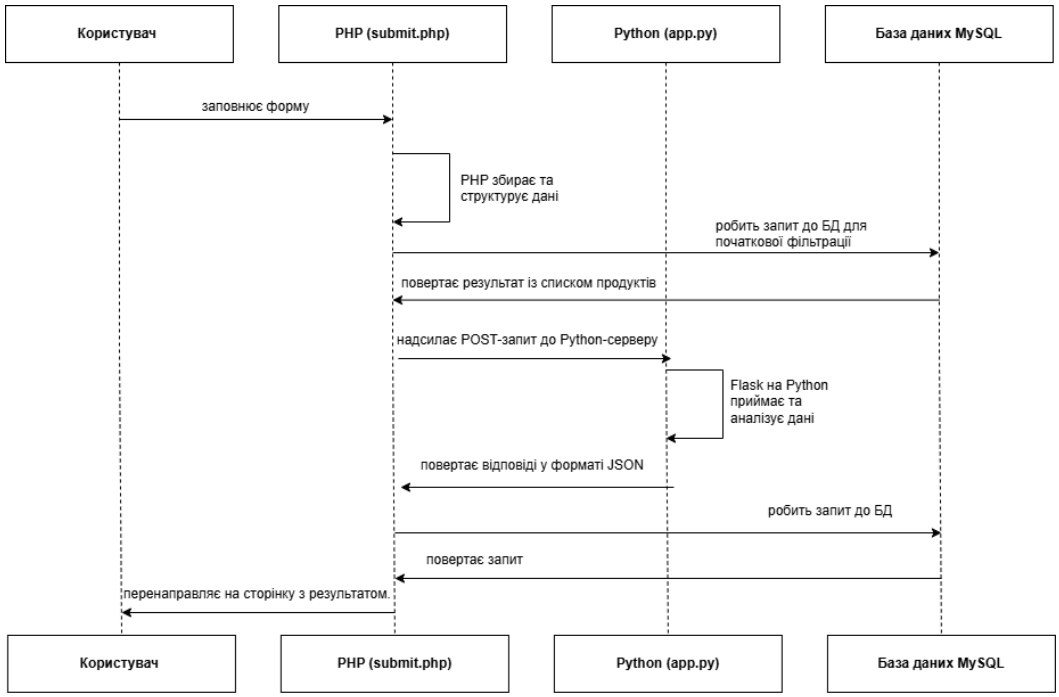


Рисунок 2.2 – Діаграма послідовності обміну запитів між модулями

Діаграма ілюструє чітку логіку обміну даними між модулями та демонструє інтеграцію Python та PHP у єдину систему.

2.4 Клієнтська частина

Клієнтська частина веб-сервісу для оптимального вибору косметики виконує ключову роль у взаємодії з користувачем. Вона відповідає за відображення інтерфейсу, збір даних та ініціювання запиту до серверної логіки та бази даних. Реалізація клієнтської частини здійснена засобами HTML, CSS, JS та PHP, де кожен з компонентів виконує окрему функцію в рамках загальної взаємодії.

Клієнтська частина веб-сервісу реалізована у вигляді кількох взаємопов’язаних сторінок, які відповідають за взаємодію з користувачем, починаючи від первинного збору вхідних параметрів до відображення персоналізованих результатів [22]. Основна логіка представлена через форму-анкету, скрипт обробки даних та сторінки з результатами.

Сторінка із формою містить чітко структуровані запитання, які охоплюють ключові фактори, що враховуються при виборі косметики. Такі як тип шкіри, наявні проблеми, бюджет, персональні вподобання тощо. Запитання згруповано в логічні блоки, що підвищує зручність та зрозумілість взаємодії. Використання відповідних елементів управління забезпечує інтуїтивно зрозумілий інтерфейс.

Після заповнення форми відповіді надсилаються методом POST на другу сторінку з PHP-скриптом, що обробляє дані та починає первину обробку даних. Здійснюється SQL-запит до бази даних для вибірки продуктів, що потенційно відповідають запиту користуча. На цьому етапі відбувається первинна фільтрація, яка виключає продукти, що не відповідають параметрам та критеріям користувача. Цей підхід дозволяє зменшити обсяг подальшої обробки та підвищити її точність. Користувач цього процесу не бачить, дана взаємодія з безшовною.

Після цього, наступним кроком, PHP-логіка формує структурований масив відфільтрованих даних, разом із ваговими критеріями, у формат JSON. Ці дані передаються на Python-сервер, де відбувається складніший аналіз.

Отримані результати у форматі JSON, які надходять із серверної частини, обробляються PHP-логікою на сторінці з результатами. PHP-скрипт знову робить запит до бази даних для пошуку та відображення продуктів. Там відбувається динамічне формування HTML-контенту з результатами підбору косметичних засобів. Відображаються назви засобів, зображення, підсумкові бали відповідності, короткий опис.

У цілому, клієнтська частина системи забезпечує повноцінний та логічна завершений цикл взаємодії з користувачем – від збору інформації до відображення рекомендацій. Такий підхід сприяє зручності, доступності та точності взаємодії.

2.5 Серверна частина

Серверна частина веб-сервісу на Python, з використанням фреймворку Flask, виконує функцію аналітичного ядра [23]. На відміну від клієнтської частини, що забезпечує інтерфейс та базову фільтрацію параметрів, Python-сервер обробляє дані глибше. Він приймає відібраний масив косметичних засобів разом із ваговими критеріями, обчислює найоптимальніші варіанти та ранжує їх за заданими параметрами за допомогою багатокритеріального методу, і повертає персоналізовані результати з балами.

2.5.1 Алгоритм методу зважених сум

У рамках серверної частини працює алгоритм багатокритеріального аналізу, який дозволяє визначити найкращу вибірку косметичних засобів відповідно до індивідуальних потреб користувача. Для цього використовується

метод зважених сум. Він є одним з найпростіших, проте надійних і широко застосовуваних методів у задачах багатокритеріальних рішень.

Суть методу полягає в обчисленні інтегральної оцінки для кожного косметичного продукту на основі суми значень його оцінок за окремими критеріями, помножених на відповідні ваги [24]. Таким чином, кожен продукт оцінюється комплексно, з урахуванням важливості кожного критерію саме для конкретного користувача.

Вхідними даними є список продуктів та ваги критеріїв. Список продуктів представлений у вигляді набору словників, де кожен елемент містить унікальний ідентифікатор продукту, його назву та числові оцінки за ревалентними критеріями.

Ваги критеріїв, також представлені словником, де включені назви критеріїв та числові коефіцієнти, які відображають ступінь важливості кожного критерію для користувача. Вказані ваги можуть бути отримані з відповідей користувача в анкеті або задані системною автоматично.

Принцип роботи алгоритму базується на кількох послідовних етапах. На початковому етапі ініціалізується порожній список, у який надалі буде збережено результати обчислень для кожного з косметичних продуктів. Далі для кожного продукту проводиться розрахунок інтегрального балу, що обчислюється методом зваженої суми, усі оцінки продукту за критеріями перемножуються на відповідні ваги, після чого результати підсумовуються. Отримана сума відображає рівень відповідності продукту персональним запитам користувача. За потреби виконується нормалізація підсумкового балу, щоб зробити результат більш зручним для сприйняття та розуміння. Після цього всі продукти ранджуються у порядку спадання їх загального балу, що дозволяє визначити найбільш ревалентні косметичні засоби. На завершальному етапі серверна частина формує структуровану відповідь у вигляді списку, який містить ідентифікатори рекомендованих продуктів та їх відповідні бали. Ця відповідь надсилається на клієнтську частину PHP-скрипту для подальшого відображення результатів користувачу.

Для обчислення оптимальності косметичного продукту, використовується формула 2.1.

$$S_i = \sum_{j=1}^{n} w_j \cdot x_{ij} \quad (2.1)$$

Де S_i – загальна оцінка продукту i, w_j – вага критерію j, x_{ij} – оцінка продукту за критерієм j, n – загальна кількість критеріїв.

2.5.2 Особливості реалізації та переваги обраного підходу

Для реалізації веб-сервісу використовується метод зважених сум у поєднанні з попередньою фільтрацією продуктів. Така двоетапна архітектура дозволяє досягти максимальної ефективності як для рекомендацій, так і для обчислювальних ресурсів. Кожен із компонентів цього підходу має важливе функціональне навантаження, а в сукупності вони утворюють збалансовану та продуктивну роботу веб-сервісу.

Ключовою превагою використання двох методів є гнучке розділення ролей. Попередня фільтрація відповідає за виключенням косметичних засобів, як не відповідають базовим вимогам користувача [25]. Це дозволяє суттєво зменшити обсяг даних, що передаються до серверної частини на Python.

Оскільки до Python-серверної логіки надходить уже попередньо відібраний набір продуктів, обчислення методом зважених сум проводиться лише для ревалентної підмножини продуктів. Це суттєво скорочує час обробки, адже кількість обчислень істотно зменшується.

При збільшені кількості продуктів у базі даних, система залиється стабільною та ефективною, вона не вимагає значної перебудови навіть при суттєвому зростанні обсягу даних. Така архітектура дозволяє просто додавати нові критерії чи логіку фільтрації, не змінюючи загальний принцип роботи. Усі налаштування правил, критеріїв, результатів та принципів зберігаються у

структурованих форматах. Це дає змогу швидко вносити зміни без глибокого втручання в код.

Також, зменшується навантаження на сервер, оскільки попередня фільтрація дозволяє уникати надсилання великих обсягів зайвої інформації.

Поєднання фільтрації на клієнтській стороні з гнучким багатокритеріальним аналізом на серверній стороні, дозволяє створити надійний і зрозумілий веб-сервіс. Такий підхід задовольняє сучасні вимоги до інтерактивних сервісів та відкриває широкі можливості для інтеграції з іншими модулями.

2.6 Структура бази даних

На рівні сховища даних, для веб-сервісу, використовується реляційна база даних MySQL яка забезпечує структуроване збереження та швидкий доступ до інформації та швидкий доступ до інформації, необхідної для результатів та рекомендацій. База складається з кількох таблиць, серед яких ключовою є таблиця «products». Вона містить дані про косметичні засоби.

Вся логіка взаємодії з базовою реалізована через SQL-запити, які інтегровані в серверну частину додатка. Завдяки цьому забезпечується надійна фіксація результатів та ревалентна видача інформації [26].

Для реалізації веб-сервісу для оптимального вибору косметики було підібрано базу даних у MySQL із використанням механізму InnoDB. База даних складається з кількох взаємопов’язаних таблиць, які зберігають інформацію про характеристики косметики та її властивостей.

Нижче наведене опис основних таблиць бази даних «Cosmetic Selector» з деталізованим описом їх полів, типу даних та функціонального призначення.

Структура таблиці «skin_types», яка зберігає стандартизовані типи шкіри, які використовуюся для класифікації косметичних засобів і потреб користувачів, приведена у таблиці 2.2.

Таблиця 2.2 – Структура таблиці «skin_types»

Ім’я поля	Тип даних та розмір	Функціональне призначення
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(50)	Назва типу шкіри

Структура таблиці «ages», яка відображає вікові групи, приведена у таблиці 2.3.

Таблиця 2.3 – Структура таблиці «ages»

Ім’я поля	Тип даних та розмір	Функціональне призначення
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(50)	Назва вікової категорії

Структура таблиці «usage_times», яка зберігає стандартизовані типи шкіри, які використовуюся для класифікації косметики, приведена у таблиці 2.4.

Таблиця 2.4 – Структура таблиці «usage_times»

Ім’я поля	Тип даних та розмір	Функціональне призначення
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(50)	Назва періоду використання

Структура таблиці «skin_problems», яка містить перелік поширених проблем зі шкірою, які користувач може вибрати для виключення в первинній фільтрації приведена у таблиці 2.5.

Таблиця 2.5 – Структура таблиці «skin_problems»

Ім’я поля	Тип даних та розмір	Функціональне призначення
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(100)	Назва проблеми шкіри

Структура таблиці «skin_types», яка зберігає категорії косметичних продуктів, приведена у таблиці 2.6.

Таблиця 2.6 – Структура таблиці «skin_types»

Ім’я поля	**Тип даних та розмір**	**Функціональне призначення**
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(100)	Назва типу косметики

Структура таблиці «preferences», яка відображає різноманітні уподобання користувача щодо компонентів косметики, приведена у таблиці 2.7.

Таблиця 2.7 – Структура таблиці «preferences»

Ім’я поля	**Тип даних та розмір**	**Функціональне призначення**
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(100)	Назва переваги

Структура таблиці «allergens», яка містить інформацію про алегрени, яких слід уникати при підборі косметики, приведена у таблиці 2.8.

Таблиця 2.8 – Структура таблиці «allergens»

Ім’я поля	**Тип даних та розмір**	**Функціональне призначення**
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(100)	Назва алергену

Таблиця «products» є основною таблицею, що містить інформацію про косметичні засоби. В ній містить уся детальна інформація та унікальний ідентифікатор ID, які є критично важливими для роботи з клієнтом та сервером. Уся початкова обробка, що включає первину фільтрацію та обчислення методом вагах, використовують саме цю таблицю для аналізу. Структура таблиці «products», косметики, приведена у таблиці 2.9.

Таблиця 2.9 – Структура таблиці «products»

Ім’я поля	Тип даних та розмір	Функціональне призначення
id	int(11)	Унікальний ідентифікатор (PK)
name	varchar(255)	Назва продукту
brand	varchar(255)	Назва бренду продукту
price	decimal(10,2)	Ціна продукту
skin_type_id	int(11)	Зовнішній ключ на таблицю «skin_types»
age_id	int(11)	Зовнішній ключ на таблицю «ages»
usage_time_id	int(11)	Зовнішній ключ на таблицю «usage_times»
description	text	Короткий опис продукту
image_url	text	Шлях до зображення

Для відображення залежностей між продуктами та різними параметрами використовуються таблиця-зв’язки. Вони дозволяють ефективно моделювати багатозначні відношення та забезпечують гнучкість при формуванні SQL-запитів. Такий підхід забезпечує нормалізацію структури та спрощує реалізацію механізмів фільтрації та пошуку.

Структура таблиці «product_problems_link», з прив’язкою продуктів до проблем шкіри, приведена у таблиці 2.10.

Таблиця 2.10 – Структура таблиці «product_problems_link»

Ім’я поля	Тип даних та розмір	Функціональне призначення
product_id	int(11)	Ідентифікатор продукту (FK на таблицю «products»)
problem_id	int(11)	Ідентифікатор продукту (FK на таблицю «skin_problems»)

Структура таблиці «product_types_link», з прив’язкою продуктів до типів засобів, приведена у таблиці 2.11.

Таблиця 2.11 – Структура таблиці «product_types_link»

Ім’я поля	Тип даних та розмір	Функціональне призначення
product_id	int(11)	Ідентифікатор продукту (FK на таблицю «products»)
cosmetic_type_id	int(11)	Ідентифікатор продукту (FK на таблицю «cosmetic_types»)

Структура таблиці «product_preferences_link», з прив’язкою продуктів до переваг, приведена у таблиці 2.12.

Таблиця 2.12 – Структура таблиці «product_ preferences _link»

Ім’я поля	Тип даних та розмір	Функціональне призначення
product_id	int(11)	Ідентифікатор продукту (FK на таблицю «products»)
preferences _id	int(11)	Ідентифікатор продукту (FK на таблицю «preferences»)

Структура таблиці «product_preferences_link», з прив’язкою продуктів до алергій, приведена у таблиці 2.13.

Таблиця 2.13 – Структура таблиці «product_ allergens _link»

Ім’я поля	Тип даних та розмір	Функціональне призначення
product_id	int(11)	Ідентифікатор продукту (FK на таблицю «products»)
allergen_id	int(11)	Ідентифікатор продукту (FK на таблицю «allergens»)

Розроблена структура бази даних «Cosmetic Selector» забезпечує зручне і логічно послідовне зберігання всієї необхідної інформації для функціонування

веб-сервісу для оптимального вибору косметики. Використання таблиць-зав’язків дозволяє моделювати складні відношення між продуктами і характеристиками, що підвищує гнучкість і точність рекомендацій [27].

2.7 Висновки до другого розділу

У другому розділі було детально розглянуто ключові компоненти та технологічні рішення, що лягли в основу розробки системи підбору косметичних продуктів. Зокрема, було здійснено обґрунтований вибір технологій. Спроєктовано архітектуру веб-сервісу для процесу обробки даних. Особливу увагу приділено механізму передачі даних між рівнями клієнта та сервера, що гарантує швидкість і надійність виконання запитів.

Реалізація клієнтської частини включає зручний інтерфейс для користувача з адаптивним дизайном та механізми первинної обробки відповідей.

В серверній частині детально описано застосування методу зважених сум для багатокритеріального ранжування косметичних продуктів. Цей метод забезпечує простоту, ефективність та роблять його оптимальним для поставлених задач.

Робота з базою даних побудована на реляційній моделі з чіткою структурою таблиць, що дозволяє швидко та надійно зберігати та обробляти дані про продукти, їх характеристики і зв’язки між іншими сутностями.

Завдяки ретельно організованому процесу виведення результатів користувач отримує зрозумілу, наочну та деталізовану інформацію про найбільш підходящі косметичні засоби, що полегшує прийняття рішення.

РОЗДІЛ 3. ПРАКТИЧНА ЧАСТИНА РЕАЛІЗАЦІЇ РОЗРОБКИ ВЕБ-СЕРВІСУ ДЛЯ ОПТИМАЛЬНОГО ВИБОРУ КОСМЕТИКИ

3.1 Реалізація клієнтського інтерфейсу

Реалізація клієнтського інтерфейсу передбачає три основні сторінки: головну сторінку, сторінку для обробки даних та сторінку з результатами аналізу.

Головна сторінка веб-сервісу index.html реалізована за допомогою стандартної HTML-розмітки та підключеної таблиці стилів styles.css. Вона містить базові структурні елементи, такі як хедер, банер, візуальні блоки з зображеннями, секції переваг та опис методів, та футер.

У верхній частині сторінки розміщено простий хедер із назвою сайту. Після нього йде банер із зображенням та кнопкою, яка веде на сторінку з формою для проходження анкети з критеріями. Кнопка оформлена так, щоб привертати увагу користувача. Вигляд головної сторінки з банером, для веб-сервісу для оптимального вибору косметики зображено на рисунку 3.1.

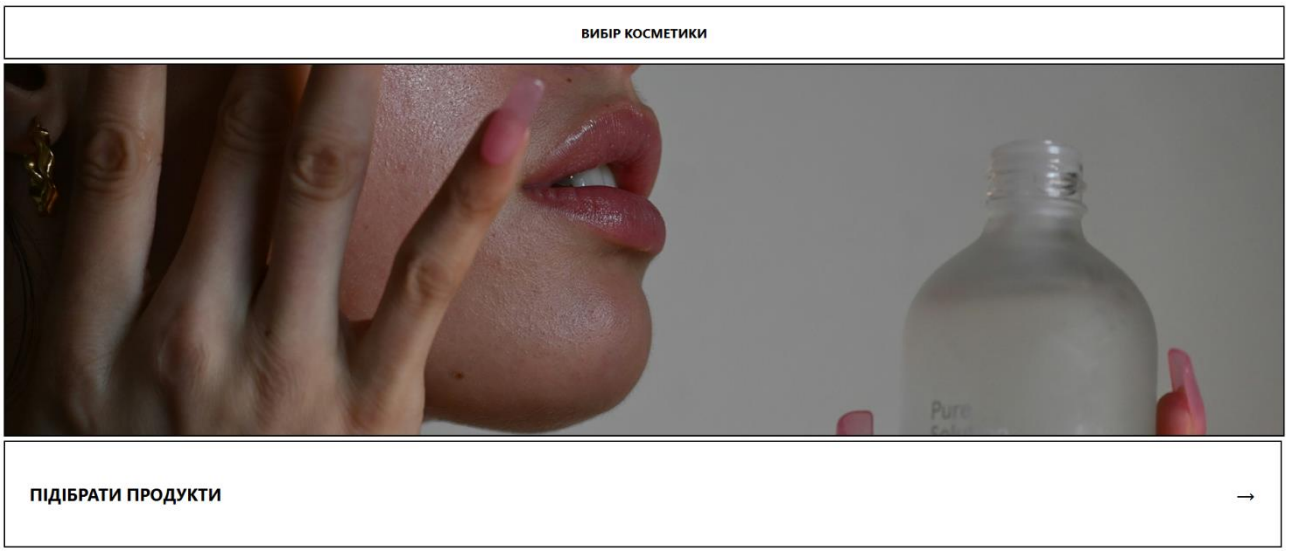


Рисунок 3.1 – Вигляд банера на головній сторінці веб-сервісу

Наступним блоком є переваги веб-сервісу, представлені у вигляді окремих прямокутників з текстом. Частина з них має світле оформлення, частина кольорове. Стиль блоків відповідає айдентиці веб-сервісу та вибраному для нього стилю. Також даний блок містить ефектні та естетичні зображення. Усі елементи рівномірно розташовані по ширині екрану. Дані блоки також є анімованими, для привабливості та ефектності веб-сторінки. Вигляд блоків з перевагами зображено на рисунку 3.2.

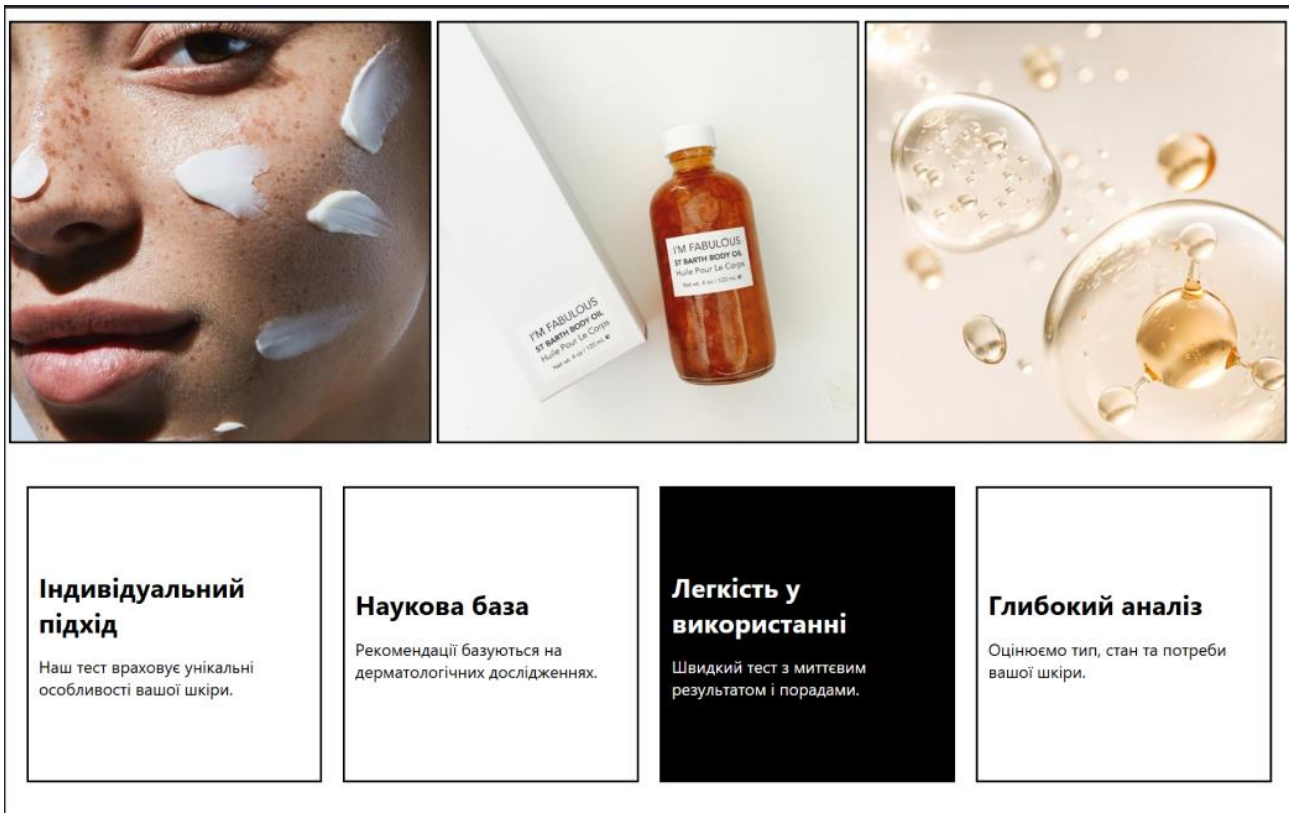


Рисунок 3.2 – Вигляд блоків з перевагами на головній сторінці веб-сервісу

Після блоків з перевагами веб-сервісу, розташований розділ у якому пояснюється як саме працює веб-сервіс. Алгоритм роботи веб-сервісу подано у вигляді трьох послідовних кроків, кожен з яких виділено номером і текстом описом. Цей блок також оформлено згідно айдентики та стилів всього веб-сервісу, щоб візуально відокремити кожен етап. Вигляд блоків із поясненням алгоритму роботи веб-сервісу зображено на рисунку 3.3. В кінці сторінки також розміщено футер з копірайтом.

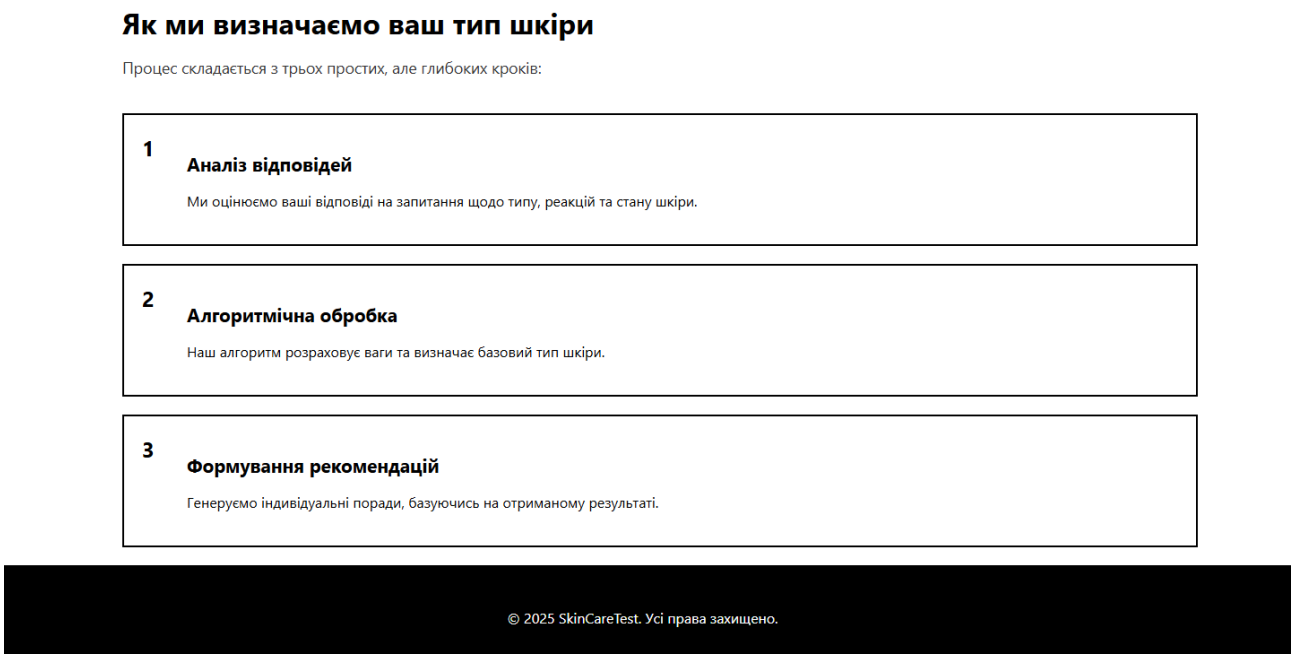


Рисунок 3.3 – Вигляд блоків з поясненням алгоритму роботи веб-сервісу

Головна сторінка повністю реалізована на HTML, зі стилями CSS, що робить її легкою і швидкою в завантаженні. Структура логічна та зручна для користувача.

Після натискання кнопки «підібрати продукти» головна сторінка перенаправляє користувача на сторінку from.php, де знаходиться анкета з параметрами та критеріями для вибору косметичних продуктів.

Зручність переходу з головної сторінки на сторінку з анкетою полягає в тому, що користувач бачить основну мету сайту – підбір оптимальних косметичних продуктів. Це забезпечує інтуїтивний користувацький досвід, де від першого екрану користувач переходить до форми з анкетою, без потреби пошуку відповідей по сторінці чи в меню.

Анкета із параметрами та критеріями для вибору косметики є основним елементом взаємодії користувача з веб-сервісом та слугує відправною точкою для подальшого логіки аналізу і формування результатів. Саме вона забезпечує збір усіх ключових даних, необхідних для оптимального підбору косметики.

Головною метою при побудові анкети було досягнення балансу між глибиною аналізу та зручністю користування. Поставало питання забезпечити

збирання достатнього обсягу інформації, і в той же час не перевантажити користувача складними або надто деталізованими питаннями. Структура форми розділена на кілька логічних блоків, кожен з яких охоплює певний аспект, що впливає на вибір косметичних продуктів. Такі як, загальні характеристики шкіри, стан шкіри, реакції на зовнішні чинники, проблемні прояви, потреби у догляді, умови використання, бюджет та особисті вподобання.

Для кожного питання передбачено відповідний формат введення radio-кнопки для однієї відповіді, та checkbox для множинного вибору [28]. Такий підхід структурує отримані дані, дозволяє уникати двозначностей та забезпечити уніфікований формат для подальшої обробки. Всі відповіді автоматично зберігаються у вигляді асоціативного масиву на сторінці PHP, де кожному запитанню відповідає унікальний ключ, а вибрана відповідь – значення.

Вигляд форми з анкетою із параметрами та критеріями для вибору косметики зображено на рисунку 3.4.

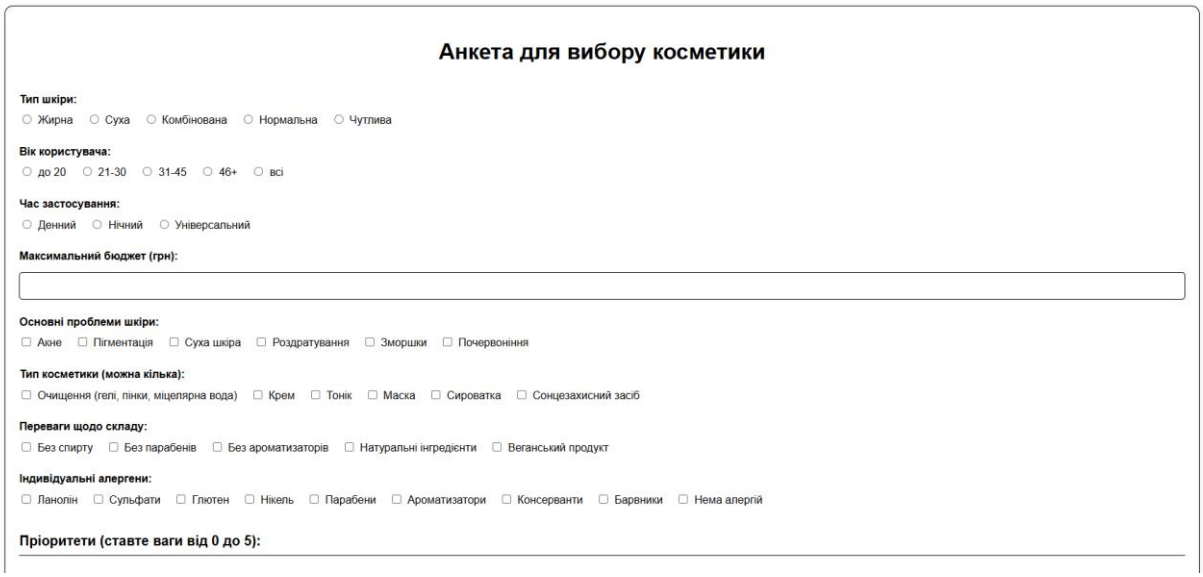


Рисунок 3.3 – Вигляд блоків з поясненням алгоритму роботи веб-сервісу

Після заповнення анкети користувач натискає кнопку «Підібрати», яка здійснює відправку даних методом POST до PHP-скрипту у файл submit.php, де відбувається подальша обробка даних.

3.2 Заповнення бази даних

На етапі реалізації веб-сервісу для оптимального вибору косметики було створено реляційну базу даних для зберігання інформації про косметичні засоби та їх характеристики. Логічна структура бази даних була реалізована за допомогою системи управління базами даних MySQL.

Після підключення до phpMyAdmin було створено таблицю «products» у базі даних проекту [29]. Таблиця призначена для зберігання даних про косметичні засоби з урахуванням різних характеристик, які необхідні для подальшого фільтрування та ранжування. SQL-запит для створення таблиці «Products» подано в лістингу 3.1.

Лістинг 3.1 – SQL-запит для створення таблиці

```
CREATE TABLE products (
        id INT AUTO_INCREMENT PRIMARY KEY,
        name VARCHAR(255),
        type VARCHAR(100),
        skin_type VARCHAR(100),
        age_group VARCHAR(50),
        problems TEXT,
        allergens TEXT,
        budget_level VARCHAR(50),
        usage_time VARCHAR(50),
        rating FLOAT
    );
```

Таким самим чином було створено усі інші таблиці, зокрема «skin_types», «ages», «usage_times», «skin_problems», «cosmetic_types», «preferences», «allergens», а також зв’язкові таблиці «product_problems_link», «product_types_link», «product_preferences_link» та «product_allergens_link». Вони використовуються для зберігання додаткових характеристик продуктів і реалізації зав’язків типу «багато до багатьох» між продуктами та їх властивостями. Це забезпечило можливість ефективно фільтрувати дані для подальшого аналізу.

У phpMyAdmin є можливість переглядати вміст таблиць, редагувати дані вручну та переглядати структуру таблиць Це дозволяє швидко перевіряти збережені продукти та їх значення. На рисунку 3.4 зображено вигляд усіх створених таблиць в базі даних «Products».

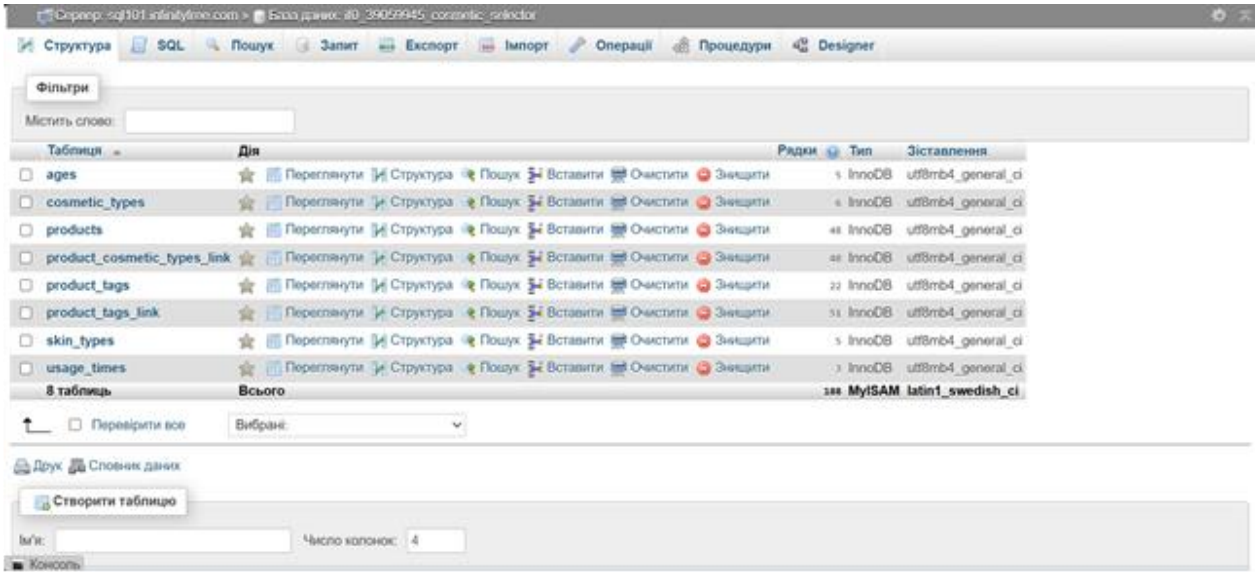


Рисунок 3.4 – Вигляд створених таблиць в phpMyAdmin

У середовищі phpMyAdmin у вкладці Designer можна візуально побачити структуру бази даних, де відображено всі таблиці проєкту та зв’язки між ними [30]. Зокрема, зовнішні ключі, що поєднують таблицю «products» із довідковими таблицями, та таблицями-зв’язками типу «багато до багатьох». На рисунку 3.5. зображено візуальне представлення структури бази даних.

Такий візуальний інструмент значно полегшує розуміння логіки побудови бази даних, та допомагають ефективно аналізувати зв’язки між сутностями. Це особливо корисно під час внесення змін до таблиць. Завдяки інструментам phpMyAdmin можна миттєво виявити помилки серед зав’язків або дублювання інформації.

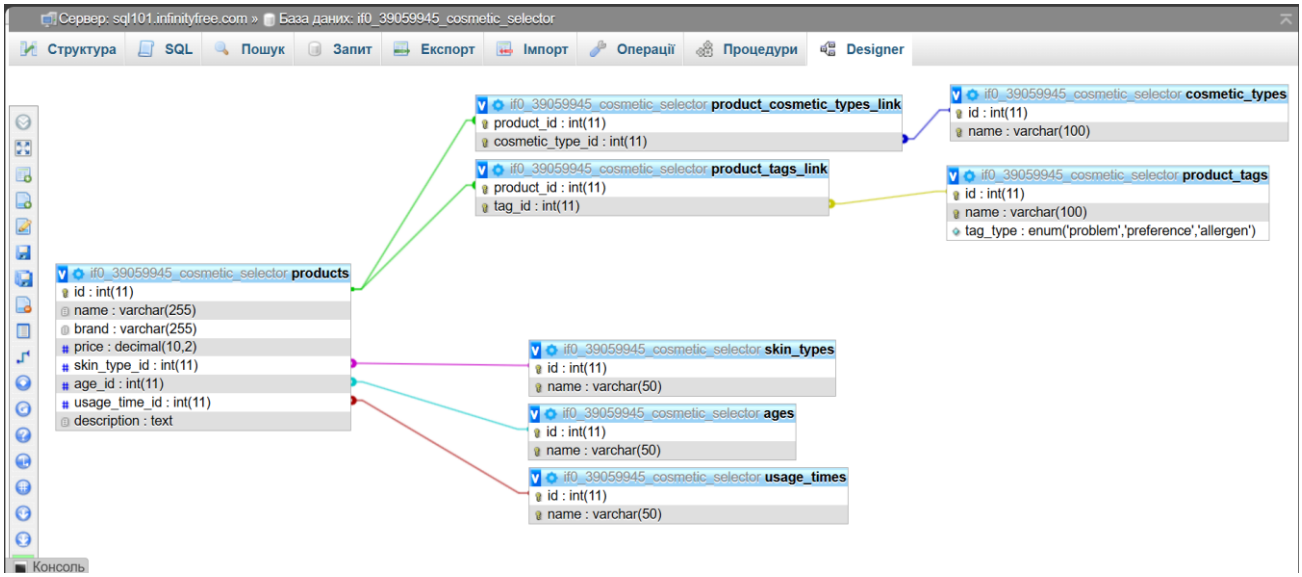
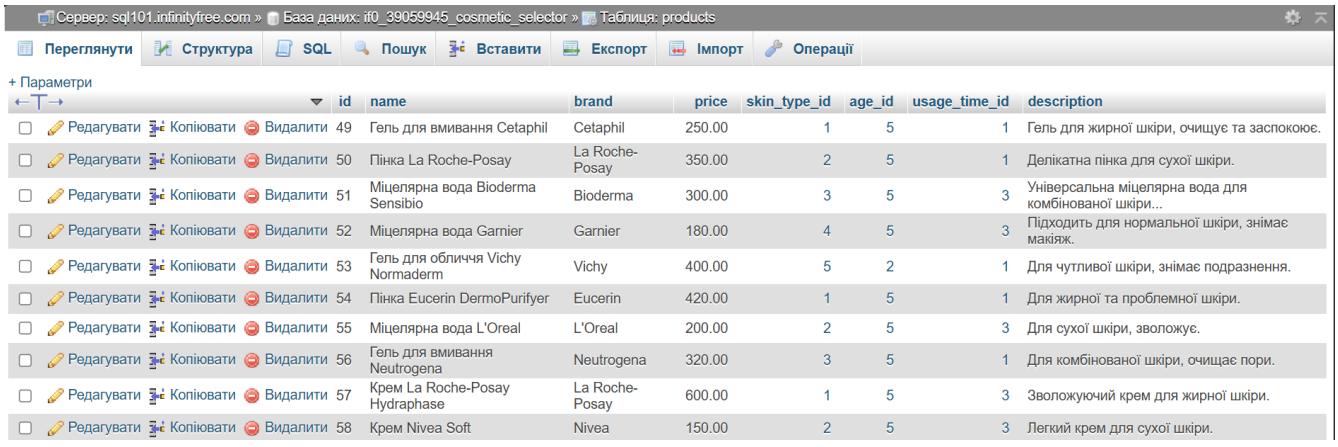


Рисунок 3.4 – Візуальне представлення структури бази даних

Заповнення бази даних «Products» відбувалось вручну, що надавало повний контроль над даними. Цей процес здійснювався за допомогою SQL-запитів INSERT, які є стандартним методом додавання нових рядків до таблиці бази даних. Для виконання цього завдання використовувався інтерфейс phpMyAdmin. У вкладці для роботи з SQL-командами у текстовому полі безпосередньо вводились та виконувались запити [31]. Кожен запит INSERT формувався таким чином, щоб вказати назву таблиці, список стовбців, в які потрібно вставити дані та відповідні значення для кожного стовбця. На рисунку 3.6 показано одну із таблиць «products» із готовити записами.



	id	name	brand	price	skin_type_id	age_id	usage_time_id	description
Редагувати Копіювати Видалити	49	Гель для вмивання Cetaphil	Cetaphil	250.00	1	5	1	Гель для жирної шкіри, очищує та заспокоює.
Редагувати Копіювати Видалити	50	Пінка La Roche-Posay	La Roche-Posay	350.00	2	5	1	Делікатна пінка для сухої шкіри.
Редагувати Копіювати Видалити	51	Міцелярна вода Bioderma Sensibio	Bioderma	300.00	3	5	3	Універсальна міцелярна вода для комбінованої шкіри...
Редагувати Копіювати Видалити	52	Міцелярна вода Garnier	Garnier	180.00	4	5	3	Підходить для нормальної шкіри, знімає макіяж.
Редагувати Копіювати Видалити	53	Гель для обличчя Vichy Normaderm	Vichy	400.00	5	2	1	Для чутливої шкіри, знімає подразнення.
Редагувати Копіювати Видалити	54	Пінка Eucerin DermoPurifyer	Eucerin	420.00	1	5	1	Для жирної та проблемної шкіри.
Редагувати Копіювати Видалити	55	Міцелярна вода L'Oreal	L'Oreal	200.00	2	5	3	Для сухої шкіри, зволожує.
Редагувати Копіювати Видалити	56	Гель для вмивання Neutrogena	Neutrogena	320.00	3	5	1	Для комбінованої шкіри, очищає пори.
Редагувати Копіювати Видалити	57	Крем La Roche-Posay Hydraphase	La Roche-Posay	600.00	1	5	3	Зволожуючий крем для жирної шкіри.
Редагувати Копіювати Видалити	58	Крем Nivea Soft	Nivea	150.00	2	5	3	Легкий крем для сухої шкіри.

Рисунок 3.4 – Візуальне представлення структури бази даних

Таким чином, база даних «Products» була успішно підготовлена для подальшої роботи веб-сервісу та забезпечила функціонал для фільтрації та ранжування косметичних продуктів.

3.3 Обробка даних PHP-скриптами

Файл submit.php відповідає за обробку даних, отриманих від користувача через форму на веб-сервісі, та виконує кілька ключових функцій:

– приймає дані з анкети методом POST;

– фільтрує відповідні продукти з бази даних згідно параметрів користувача;

– передає відібрані дані разом із ваговими коефіцієнтами до Python-серверу для обробки та ранжування;

– зберігає отримані результати в сесії;

– перенаправляє користувача на сторінку з результатами results.php.

На початку submit.php отримує дані з форми, обробляє їх та фільтрує. Для безпеки використовуються функції типу mysqli_real_escape_string та filter_input. Масиви обробляються в циклах для формування SQL-запитів (див. лістинг 3.2).

Лістинг 3.3 – Зчитування параметрів в PHP-скрипті

```
$skin_type = $_POST['skin_type'];
$age = $_POST['age'];
$budget = $_POST['budget'];
$problems = $_POST['problems']; // масив
$allergens = $_POST['allergens']; // масив
$usage_time = $_POST['usage_time'];
```

Після збору даних формується динамічний SQL-запит для вибірки з таблиці «Products», де беруться важливі параметри для первинної фільтрації косметичних продуктів [32].

Таким чином, формується масив із продуктів які підходять під надані користувачем параметри та критерії.

Наступним кроком, оброблений масив продуктів готується для передачі до Python-сервера для обчислень та ранжування. Разом із масивом з характерниками продуктів передаються і вагові коефіцієнти для параметрів, що задають пріоритет важливості кожної характеристики. Формується структура даних у форматі JSON (див. лістинг 3.3).

Лістинг 3.3 – Формування структури даних у формат JSON в PHP

```
$data = [
    'products' => $filtered_products,
    'weights' => [
        'skin_type' => 0.3,
        'budget' => 0.2,
        'problems' => 0.3,
        'usage_time' => 0.2
    ]
];
```

Запит надсилається на Python-сервер з фреймворком Flask методом POST. Надається адреса серверу та надсилаються дані із відповідними заголовками. Після отримання відповіді від Flask, PHP-скрипт декодує JSON та зберігає результати в сесії. Це дозволяє передати дані на сторінку з результати results.php, не розміщуючи їх у URL чи формі. Таким чином, користувач одразу перенаправляється на сторінку з результатами, де відображаються рекомендовані засоби з урахуванням персональних потреб та пріоритетів.

3.4 Реалізація алгоритму зважених сум у середовищі Python

Файл app.py, розміщений на PythonAnywhere, реалізує REST API для ранжування продуктів згідно з параметрами користувача.

PHP-сервер надсилає на Flask POST-запит з JSON, який містить два основні блоки. products – список продуктів, які були попередньо відібрані на основі фільтрів у базі даних. Кожен продукт представлений як словник із полями, наприклад, id та price. weights – словник вагових коефіцієнтів, які відображають

важливість кожного параметра у підборі. Наприклад, "skin_type": 0.3 означає, що тип шкіри має 30% впливу на загальний рейтинг.

Основний алгоритм — метод зваженої суми (Weighted Sum Model). Ідея полягає в тому, що кожен критерій (тип шкіри, ціна, час використання, проблеми шкіри) оцінюється окремо, а потім оцінки сумуються з урахуванням ваг.

Основні кроки для розрахунку:

– для кожного продукту визначається оцінка по кожному параметру;

– нормалізується оцінка;

– оцінка базується на сумісності з проблемами користувача;

– кожна оцінка множиться на відповідний ваговий коефіцієнт із отриманих даних;

– всі зважені оцінки сумуються у загальний бал для продукту.

Програмний код для розрахунку методом ваг, реалізує вище поставлені завдання для алгоритму (див. лістинг 3.4).

Лістинг 3.3 – Формування структури даних у формат JSON в PHP

```
def normalize(value, c):
            min_v = min_vals[c]
            max_v = max_vals[c]
            if max_v == min_v:
                return 1
            if c == 'price':
                return (max_v - value) / (max_v - min_v)
            else:
                return (value - min_v) / (max_v - min_v)

        # Розрахунок score
        for product in products:
            score = 0
            for c in criteria:
                try:
                    val = float(product.get(c, 0))
                except (ValueError, TypeError):
                    val = 0
                norm_val = normalize(val, c)
                score += norm_val * weights.get(c, 0)
            product['score'] = score
```

Після розрахунку балів всі продукти збираються в список словників із ключами id і score. Потім цей список сортується за спаданням оцінки – від найбільш релевантного до найменш релевантного [33].

Після сортування список продуктів із їх рейтингами конвертується у JSON і повертається назад PHP-серверу, який потім виводить ці дані користувачу. Таке рішення значно спрощує запуск та підтримку Python API, усуваючи необхідність у складних налаштуваннях інфраструктури. Сервер постійно працює в режимі очікування, готовий приймати та обробляти POST-запити від PHP-сервера.

3.5 Формування та відображення результатів користувачу

Виведення результатів — завершальний та найважливіший етап роботи веб-сервісу, адже саме тут користувач отримує персоналізовану відповідь на основі пройденого анкети з критеріями.

Після виконання алгоритму ранжування сервер отримує від Flask-сервера список продуктів із унікальними ідентифікаторами, балами відповідності та додатковою інформацією.

Для зручності обробки та передачі даних між сторінками використовується механізм PHP-сесій. Після того, як PHP-скрипт submit.php отримує від Flask сервера JSON із результатами ранжування продуктів, він декодує ці дані та зберігає їх у сесії. Це дозволяє уникнути повторних запитів до Flask та зберегти результати для подальшого виводу. PHP-скрипт виконує запити до бази даних для отримання повних характеристик кожного рекомендованого продукту: назви, бренду, ціни, опису, типу шкіри, вікової категорії, часу використання та інших властивостей.

Для обробки та візуального представлення результатів підбору косметичних засобів користувачу на сторінці results.php. Цей файл відповідає за виведення продуктів, відсортованих за релевантністю, із урахуванням оцінок, сформованих алгоритмом на Python.

На основі отриманих даних PHP формує веб-сторінку з результатами підбору. Кожен продукт виводиться у вигляді окремого інформаційного блоку, який містить: назву та бренд косметичного засобу; ціну; короткий опис та ключові характеристики; іконки або текстові позначки, що відображають відповідність типу шкіри, віковій категорії, проблемам і перевагам користувача; рейтинг або бал відповідності, що показує, наскільки продукт підходить конкретному користувачу.

На рисунку 3.5 зображено вигляд виведення косметичного засобу для користувача.

Рисунок 3.5 – Вигляд виведення інформації про продукт після аналізу

Інтерфейс побудований із застосуванням сучасних веб-технологій (HTML5, CSS3, JavaScript), що забезпечує адаптивний дизайн і гарний вигляд на різних пристроях — від смартфонів до десктопів. Це покращує користувацький досвід і сприяє швидкому прийняттю рішення. Для зручності користувача результати відсортовані за балом відповідності. Також передбачено можливість фільтрації та пошуку безпосередньо на сторінці результатів.

Сторінка з результатами results.php є кінцевим етапом логіки підбору косметики. Всі дії, починаючи з обробки анкети на клієнтській стороні, фільтрації у базі даних, передачі даних до Python-сервера та обробки алгоритмом ранжування, завершується виводом у цьому файлі. Цей скрипт гарантує, що користувач отримає зручний і зрозумілий звіт з індивідуально підібраними косметичними засобами, що максимально відповідають його потребам.

3.6 Тестування функціоналу веб-сервісу

Для підтвердження коректності роботи розробленої системи підбору косметичних засобів було проведено комплексне тестування її основних модулів. Тестування включало перевірку клієнтського інтерфейсу, обробки даних на сервері, взаємодії між PHP та Flask, а також коректності виведення результатів.

Тестування форми з анкетою на сторінці index.php полягало у введенні різноманітних варіантів даних. Було перевірено, що всі поля коректно приймають і передають інформацію на сервер, а також що не виникає помилок при відсутності деяких опцій, як зображено на рисунку 3.6.

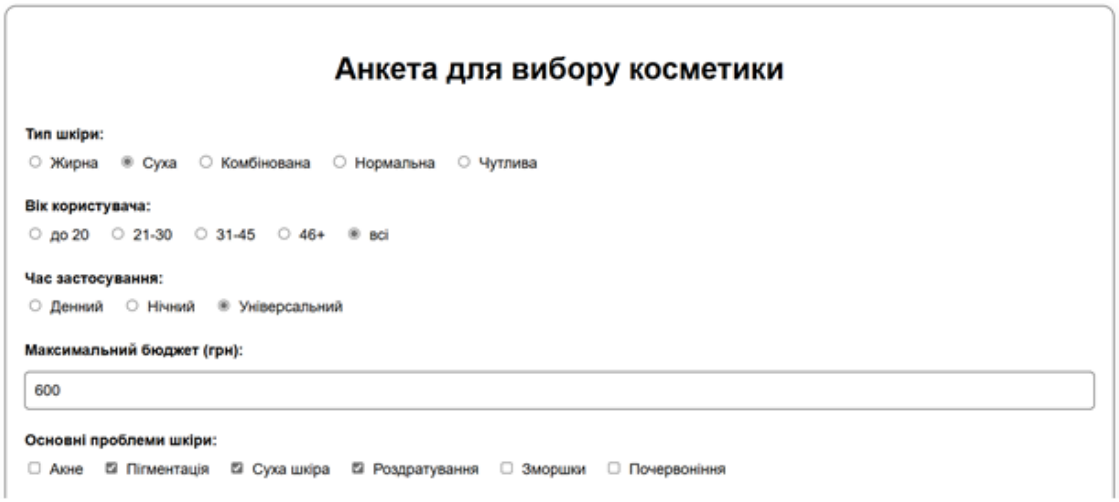


Рисунок 3.6 – Тестування коректності роботи форми із параметрами

Скрипт submit.php проходив тестування на правильність фільтрації даних та формування запиту до Flask-сервера. Для цього використовувалися тестові дані із різними параметрами, щоб упевнитися, що запити формуються коректно, а відповідь від Flask надходить у правильному форматі та обробляється без помилок.

Для перевірки коректності роботи алгоритму підбору в app.py було проведено юніт-тестування основних функцій, зокрема вагового ранжування продуктів [34]. Результати тестів підтвердили, що система правильно ранжує продукти за заданими критеріями та коректно обробляє різні комбінації вхідних даних. Було отримано файл із вмістом який підтверджує правильність роботи Flask сервера.

На сторінці results.php було перевірено коректність відображення списку продуктів, точність виводу даних з бази за id, а також обробку ситуацій, коли підходящих продуктів немає. Результати зображені на рисунку 3.7, підтверджують коректну роботу розробленого механізму.

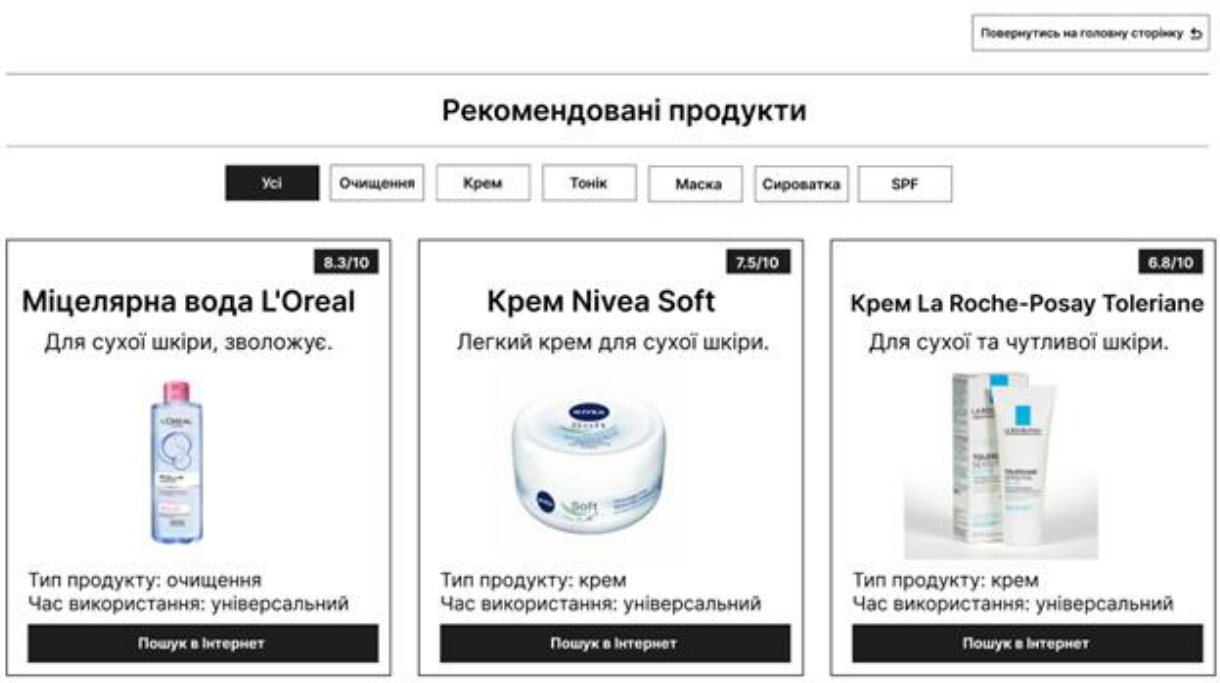


Рисунок 3.7 – Тестування коректності роботи механізму та виводу кінцевого результату

Тестування показало, що розроблена система працює стабільно, коректно обробляє дані на всіх етапах та виводить користувачу точні рекомендації. Виявлені незначні недоліки були оперативно усунуті, що підтвердило надійність і готовність системи до практичного використання.

3.7 Висновки до третього розділу

У результаті реалізації практичної частини було успішно створено повноцінний та функціональний веб-сервіс для оптимального вибору косметики. Даний веб-сервіс є комплексним рішенням, що об'єднує клієнтський інтерфейс, ефективну серверну обробку даних та надійну взаємодію з базою даних. Кожен із модулів системи був розроблений та впроваджений відповідно до поставлених цілей і технічних вимог проєкту.

Клієнтський інтерфейс відіграє ключову роль у забезпеченні зручної та інтуїтивно зрозумілої взаємодії з користувачем, він надає адаптивну форму для збору даних користувача, що є основою для подальшої обробки результатів. Серверна частина системи ефективно обробляє ці вхідні дані, забезпечуючи їх коректну передачу до Python-сервера для аналізу та отримання найбільш релевантних результатів.

Розроблений алгоритм ранжування продукції, що базується на методі зваженої суми, працює коректно, та дозволяє веб-сервісу пропонувати користувачеві найбільш підходящі варіанти на основі комплексної оцінки.

Виведення результатів реалізовано у вигляді інформативної та візуально привабливої сторінки. Вона надає користувачеві чітке та організоване представлення запропонованих варіантів, дозволяючи йому швидко орієнтуватися серед них та приймати обґрунтоване рішення.

Загалом, створена система повністю відповідає поставленим завданням та може бути успішно використана для автоматизованого та персоналізованого підбору косметики, враховуючи індивідуальні параметри користувача.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Психологічні фактори впливу на безпеку людини

Психологічні фактори становлять одну з ключових складових у системі забезпечення безпеки життєдіяльності людини, зокрема в умовах професійної діяльності [40]. Згідно з ДСТУ 2293:2014 «Охорона праці. Терміни та визначення», стан безпеки залежить не лише від технічних чи організаційних чинників, а й від особистісних характеристик працівника, його психоемоційного стану, рівня стресу, втоми та психологічної готовності до виконання трудових обов’язків [42].

Фізіологічна та психічна втома призводить до зниження уваги й ефективності, тому важливо дотримуватися режиму праці та відпочинку з регулярними перервами і фізичною активністю.

Стрес провокує емоційне виснаження та підвищує ризик помилок, доцільно впроваджувати техніки релаксації, тренінги зі стрес-менеджменту та створювати комфортне робоче середовище.

Емоційна нестабільність ускладнює комунікацію та підвищує конфліктність, тому слід забезпечити стабільний психологічний клімат і підтримку з боку колективу та керівництва.

Конфлікти негативно впливають на командну роботу і мотивацію, тому їх необхідно попереджати шляхом розвитку навичок конструктивного спілкування і впровадження етичних норм взаємодії.

Втрата концентрації підвищує ризик допущення помилок, що обумовлює потребу в чергуванні видів діяльності та застосуванні методів тренування уваги.

Уповільнення когнітивних процесів знижує здатність до швидкого прийняття рішень, тому рекомендується уникати перевантаження, забезпечити повноцінний відпочинок і нормоване навантаження.

Низький рівень психологічної підготовленості ускладнює виконання завдань високої інтенсивності, отже варто проводити психологічну підготовку, інструктажі та навчання технікам саморегуляції.

Звикання до потенційної небезпеки притуплює відчуття ризику, що вимагає постійного оновлення знань з техніки безпеки та нагадування про потенційні загрози.

Ігнорування ризиків може призвести до порушення норм безпеки, тому важливо формувати культуру безпечної поведінки та підтримувати обізнаність працівників щодо наслідків недбалості [41].

4.2 Вимоги електробезпеки при експлуатації персональних комп'ютерів в офісних приміщеннях

Офісні приміщення ІТ-компаній містять робоче обладнання (персональні комп’ютери, принтери, маршрутизатори тощо), яке живиться від електромережі. Основним напрямком, що забезпечує необхідний рівень електробезпеки, є застосування нормативних методів захисту в електроустановках [45]. До основних методів захисту від ураження людини електричним струмом, що застосовуються в електроустановках, відносяться:

– використання необхідного типу ізоляції;

– забезпечення недоступності струмоведучих частин електроустановки;

– електричний розподіл електричної мережі;

– використання малої напруги;

– захисне відключення;

– захисне заземлення;

– занулення.

У будь-яких електроустановках важливо правильно підібрати тип ізоляції, щоб гарантувати безпечну роботу обладнання та захистити людей від ураження

струмом. Залежно від умов застосовують різні типи ізоляції струмоведучих частин робочу, додаткову, подвійну та посилену.

Робоча ізоляція — це основний захист, який забезпечує нормальну роботу пристрою і безпеку користувача.

Додаткова ізоляція використовується разом з робочою — вона вступає в дію у випадку, якщо основна пошкоджена.

Подвійна ізоляція поєднує робочу та додаткову, що підвищує загальний рівень захисту.

Посилена ізоляція — це покращений варіант, який сам по собі забезпечує такий же рівень захисту, як подвійна.

Якість ізоляції визначається її опором до струму. Згідно з нормами, в електроустановках з напругою до 1000 В опір ізоляції має бути не менше ніж 0,5 МОм.

Щоб запобігти випадковому контакту з частинами електроустановок, які знаходяться під напругою, важливо зробити їх фізично недоступними. Це досягається за допомогою правильного розміщення електропроводки. Наприклад, тимчасові електричні мережі прокладають по опорах на висоті не менше 2,5 метра — над робочими місцями, 3,5 метра — над проходами, 6 метрів — над проїздами. Якщо дроти не мають ізоляції, їх обов’язково треба прокладати не нижче ніж 3,5 метра від землі чи підлоги.

У діючих виробничих приміщеннях, у місцях постійного електропостачання використовується: схована електропроводка; огородження струмоведучих частин; блокування та розміщення струмоведучих частин елекроустановки у важкодоступному місці. Огородження можуть бути суцільним або сітчастими з розміром осередку не більше 25 х 25 мм.

Суцільні або сітчасті огородження використовують при напрузі вище: у сухих приміщеннях – 42 В, у сирих – 36 В, а в особливо сирих – 12 В.

Електричний розподіл мереж використовують для підвищення безпеки в електромережах із великою довжиною або складною структурою, особливо там, де нейтраль ізольована і можуть виникати значні струми замикання на землю.

Метою цього методу захисту є зменшення величини ємнісного струму замикання на землю, що збільшує комплексний опір ізоляції фаз відносно землі. Цей метод реалізують шляхом підключення окремих споживачів електричної енергії через розділові трансформатори, що живляться від магістральної мережі.

Застосування малих напруг — це метод зниження напруги живлення електричних установок до значення довгостроково допустимої напруги дотику, при якій навіть двофазний дотик людини є безпечним [44]. Суть методу полягає у використанні напруги живлення електроустановки не вище 42 В з метою зменшення небезпеки ураження людини електричним струмом.

Метод малих напруг реалізують з використанням понижуючих трансформаторів [43]. Застосування автотрансформаторів для одержання малої напруги забороняється. Величину малої напруги вибирають з урахуванням категорії приміщення за ступенем небезпеки ураження людини електричним струмом. У зв'язку з цим у переносних електроустановках, які використовують у виробничих умовах, для забезпечення електробезпеки застосовують малі напруги 12 і 36 В.

Лише низької напруги часто недостатньо, тому додатково використовують подвійну ізоляцію, захист від випадкового дотику та інші технічні заходи.

Захисне заземлення — це навмисне електричне з'єднання з землею або її еквівалентом металевих не струмоведучих частин електроустановки, що можуть виявитися під напругою в аварійних ситуаціях.

Захисне заземлення застосовується у межах до 1000 В з ізольованою нейтраллю, та у мережах до 1000 В з ізольованою нейтраллю. Захисний заземлюючий пристрій складається із сукупності заземлювача і провідників, що заземлюють. Заземлювач являє собою провідник або систему з'єднаних між собою металевих провідників, що знаходяться в безпосередньому контакті з землею. Провідник, що заземлює, – це металевий провідник, що з'єднує частини електричної установки, які заземлюються.

Для заземлення електроустановок використовують природні й штучні заземлювачі. Природними заземлювачами можуть бути металеві конструкції

будинків, трубопроводи й устаткування, що мають надійне з'єднання із землею. Природними можуть бути металеві конструкції будівель, арматура, трубопроводи (крім газових і пальних речовин, або ізольованих труб — їх використовувати не можна). Штучними є металеві труби (Ø 35–50 мм) або кутик 40 мм завдовжки 2,5–3,5 м, з'єднані металевими смугами (≥48 мм²) на глибині не менше 0,5 м.

За розташуванням заземлювачів відносно корпусів електроустановки, що заземлюються, захисні заземлення поділяються на виносні й контурні. У виносного захисного заземлення заземлювачі розташовують на деякому видаленні (не менше 20 м) від устаткування, що заземлюється. У контурного захисного заземлення заземлювачі розташовують у вигляді контуру по площі, на якій розташовані електроустановки, що заземлюються.

Занулення — це навмисне електричне з'єднання металевих не струмоведучих частин електроустановки, які можуть опинитися під напругою в аварійній ситуації, з нульовим захисним провідником. Дія занулення заснована на перетворенні замикання на корпус в однофазне коротке замикання з метою формування великих струмів, здатних забезпечити спрацьовування апаратів захисту. Застосовується у мережі з глухозаземленою нейтраллю напругою до 1000 В. Для занулення мережу перетворюють у трифазну чотирипровідну (рис. 4.9). Провідність нульового захисного проводу має бути не менше 50% від провідності фазного проводу.

Блокування — пристрої, що відключають живлення електроустановки при спробі несанкціонованого доступу, використовуються у місцях із частими роботами на струмоведучих частинах (випробувальні стенди, установки для випробувань ізоляції тощо). Електричні блокування вимикають електроустановку при відкритті дверей або кришки завдяки спеціальним контактам. У разі дистанційного керування вони розривають коло пускача, а повторне включення можливе лише через натискання кнопки. Контакти мають спрацьовувати вже при незначному відкритті (10–15 см).

Механічні блокування не дозволяють включити апарат при відкритій кришці або зняти кожух під напругою. Використовуються в апаратах автоматики, ПІС, радіоустановках. Наприклад, при витягуванні блоку автоматично розмикається штепсельне з'єднання.

Захисне відключення є додатковим захистом, що забезпечує автоматичне відключення електроустановки з появою в ній небезпеки ураження людини електричним струмом (0,2 с).

Основними параметрами пристроїв захисного відключення (ПЗВ) є величина струму, наприклад, в схемі захисного заземлення, на який реагує пристрій, і величина зміни напруги та його швидкодія.

4.3 Висновок до четвертого розділу

У четвертому розділі кваліфікаційної роботи описано комплекс заходів, спрямованих на забезпечення безпечних умов праці персоналу, що працює з комп’ютерною технікою, з урахуванням психофізіологічних особливостей працівників, а також вимог електробезпеки.

Розглянуто основні психофізіологічні фактори, що впливають на ефективність і безпеку роботи — зокрема, рівень зорового навантаження, стомлюваність, робоче положення тіла, вплив монотонності, освітлення, шуму, мікроклімату тощо. Для зменшення впливу шкідливих чинників запропоновано організаційні та технічні заходи: дотримання ергономіки робочого місця, оптимізацію режиму праці та відпочинку, забезпечення достатнього рівня освітлення і вентиляції, використання зручних меблів тощо.

Також розглянуто питання електробезпеки — починаючи від організаційних заходів, завершуючи технічними аспектами безпеки. Особливу увагу приділено відповідності електрообладнання нормативним документам, таким як ПУЕ та Правилам будови електроустановок.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи бакалавра (КВБ) було розроблено веб-сервіс для оптимального вибору косметики засобами Python та PHP. Для реалізації веб-сервісу також було використано інструменти HTML, CSS, JS. Для побудови та розгортання – бази даних MySQL.

У результаті виконання КВБ розв’язано наступні задачі:

– проаналізовано існуючі веб-рішення;

– розроблено структуру веб-сервісу з урахуванням особливостей інтеграції Python та PHP;

– розроблено модель для оптимального вибору косметики;

– виконано тестування та оцінку ефективності роботи веб-сервісу;

– оцінено можливості масштабування та подальшого розвитку веб-сервісу.

Виконання проєкту дозволило успішно реалізувати поставлену мету і підтвердити актуальність теми в умовах сучасного розвитку IT-технологій та електронної комерції.

ПЕРЕЛІК ДЖЕРЕЛ

1 Перегляд ПРОГНОЗУВАННЯ РОЗВИТКУ ПРОДУКТОВОГО IT-РИНКУ В УКРАЇНІ. [Електронний ресурс]. – Режим доступу до ресурсу: https://economyandsociety.in.ua/index.php/journal/article/view/5606/5543 (дата звернення: 08.05.2025).

2 IT і бізнес: як технології впливають на розвиток сучасних підприємств. LemonSchool. [Електронний ресурс]. – Режим доступу до ресурсу: https://lemon.school/blog/it-i-biznes-yak-tehnologiyi-vplyvayut-na-rozvytok-suchasnyh-pidpryyemstv (дата звернення: 08.05.2025).

3 Skin care – worldwide | statista market forecast. Statista. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.statista.com/outlook/cmo/beauty-personal-care/skin-care/worldwide (дата звернення: 09.05.2025).

4 Системи підтримки прийняття рішень для українських підприємств: особливості та переваги – Режим доступу до ресурсу: Softline Defence. [Електронний ресурс]. https://surl.li/mtkhmy (дата звернення: 9.05.2025).

5 Marketing strategy for beauty products: a complete guide. Free QR Code Generator Online with Logo. [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.qrcodechimp.com/marketing-strategy-for-beauty-products/ (дата звернення: 09.05.2025).

6 Build Recommendation System. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.leewayhertz.com/build-recommendation-system/#Types-of-AI-powered-recommendation-systems (дата звернення: 10.05.2025).

7 Data Filtering: Techniques, Benefits, and Best Practices. Astera. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.astera.com/type/blog/data-filtering/ (дата звернення: 10.05.2025).

8 An integrated simple weighted sum product method–wisp. IEEE Xplore. [Електронний ресурс]. – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/9432404 (дата звернення: 10.05.2025).

9 Принципи проектування інформаційних систем. Pidru4niki. [Електронний ресурс]. – Режим доступу до ресурсу: https://pidru4niki.com/74226/informatika/printsipi_proektuvannya_informatsiynih_sistem (дата звернення: 11.05.2025).

10 Banga S. What is Web Application Architecture? Components, Models, and Types. Hackr.io. [Електронний ресурс]. – Режим доступу до ресурсу: https://hackr.io/blog/web-application-architecture-definition-models-types-and-more (дата звернення: 11.05.2025).

11 Функціональні та нефункціональні вимоги. Guru99. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.guru99.com/uk/functional-vs-non-functional-requirements.html (дата звернення: 11.05.2025).

12 Процеси життєвого циклу ІС. Pidru4niki. [Електронний ресурс]. – Режим доступу до ресурсу: https://pidru4niki.com/1003020947736/informatika/protsesi_zhittyevogo_tsiklu (дата звернення: 11.05.2025).

13 Terra J. Python for data science and data analysis. Simplilearn.com. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.simplilearn.com/why-python-is-essential-for-data-analysis-article (дата звернення: 11.05.2025).

14 GeeksforGeeks. Flask Tutorial – GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/python/flask-tutorial/ (дата звернення: 10.05.2025).

15 MySQL overview: key features, benefits, and use cases | openlogic. OpenLogic. [Електронний ресурс]. – Режим доступу до ресурсу: https://surl.lu/lsxuek (дата звернення: 10.05.2025).

16 What is Python Anywhere, and how to use it for your projects? : Collegelib.com. 415 Unsupported Media Type. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.collegelib.com/what-is-pythonanywhere/ (дата звернення: 12.05.2025).

17 Учасники проектів Вікімедіа. Клієнт-серверна архітектура – Вікіпедія. Вікіпедія. [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Клієнт-серверна_архітектура (дата звернення: 12.05.2025).

18 Client-Server Architecture – System Design – GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/system-design/client-server-architecture-system-design/ (дата звернення: 12.05.2025).

19 What is Rest API?: Understanding REST Architecture with Examples | BrowserStack. BrowserStack. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.browserstack.com/guide/rest-api (дата звернення: 14.05.2025).

20 What is REST?. REST API Tutorial. [Електронний ресурс]. – Режим доступу до ресурсу: https://restfulapi.net/ (дата звернення: 14.05.2025).

21 MySQL Queries. Tutorials on Technical and Non Technical Subjects. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.tutorialspoint.com/mysql/mysql-queries.htm (дата звернення: 14.05.2025).

22 Website Structure A to Z (With Examples). Slickplan. [Електронний ресурс]. – Режим доступу до ресурсу: https://slickplan.com/blog/types-of-website-structure (дата звернення: 14.05.2025).

23 Coderivers. Python for Web Development: A Comprehensive Guide. CodeRivers. [Електронний ресурс]. – Режим доступу до ресурсу: https://coderivers.org/blog/python-for-web-development/ (дата звернення: 15.05.2025).

24 Contributors to Wikimedia projects. Weighted sum model – Wikipedia. Wikipedia, the free encyclopedia. [Електронний ресурс]. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Weighted_sum_model (дата звернення: 15.05.2025).

25 What is: Weighted Sum. LEARN STATISTICS EASILY. [Електронний ресурс]. – Режим доступу до ресурсу: https://statisticseasily.com/glossario/what-is-weighted-sum/ (дата звернення: 15.05.2025).

26 GeeksforGeeks. How to Design a Database for Web Applications – GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/dbms/how-to-design-a-database-for-web-applications/ (дата звернення: 14.05.2025).

27 Combining Tables in SQL. Dataquest. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.dataquest.io/tutorial/combining-tables-in-sql-tutorial/ (дата звернення: 16.05.2025).

28 W3Schools.com. W3Schools Online Web Tutorials. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.w3schools.com/html/html_forms.asp (дата звернення: 17.05.2025).

29 SQL Query Examples and Tutorial. DataCamp. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.datacamp.com/tutorial/sql-query-examples-and-tutorial (дата звернення: 17.05.2025).

30 How to use phpMyAdmin to develop a website. HostAdvice. [Електронний ресурс]. – Режим доступу до ресурсу: https://hostadvice.com/how-to/web-hosting/php/how-to-use-phpmyadmin/ (дата звернення: 19.05.2025).

31 Basic SQL Queries: Key Concepts & Examples. Newtum. [Електронний ресурс]. – Режим доступу до ресурсу: https://blog.newtum.com/basic-sql-queries/.

32 W3Schools.com. W3Schools Online Web Tutorials. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.w3schools.com/php/php_mysql_connect.asp (дата звернення: 19.05.2025).

33 5 Best Ways to Calculate Nested List Weighted Sum II in Python – Be on the Right Side of Change. Bot Verification. [Електронний ресурс]. – Режим доступу до ресурсу: https://blog.finxter.com/5-best-ways-to-calculate-nested-list-weighted-sum-ii-in-python/ (дата звернення: 15.05.2025).

34 Unit Testing: Definition, Examples, and Critical Best Practices. Bright Security. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.brightsec.com/blog/unit-testing/ (дата звернення: 19.05.2025).

35 1. Fryz M., Mlynko B. Property Analysis of Conditional Linear Random Process as a Mathematical Model of Cyclostationary Signal. 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP 2022). Ternopil, Ukraine: CEUR Workshop Proceedings, 2022. Vol. 3309. P. 77–82.

36 2. Zaporozhets, Y. Kuts, B. Mlynko, M. Fryz, and L. Scherbak, “EEG Signal Classification Using Linear Process Model-Based Feature Extraction and Supervised Learning,” in Advanced System Development Technologies II. Studies in Systems, Decision and Control, M. Bezuglyi, N. Bouraou, V. Mykytenko, G. Tymchyk, and A. Zaporozhets, Eds., Cham: Springer Nature Switzerland, 2025, pp. 235–257. doi: 10.1007/978-3-031-82035-9_7.

37 3. Fryz M., Scherbak L., Mlynko B., Mykhailovych T. Linear Random Process Model-Based EEG Classification Using Machine Learning Techniques. Proceedings of the 1st International Workshop on Computer Information Technologies in Industry 4.0 (CITI 2023). Ternopil, Ukraine: CEUR Workshop Proceedings, 2023. Vol. 3468. P. 126–132.

38 4. M. Fryz, B. Mlynko, Property analysis of multivariate conditional linear random processes in the problems of mathematical modelling of signals, Technology Audit and Production Reserves, 3/2(65), 2022, pp. 29–32.

39 5. Бабак В. П., Марченко М. Є., Фриз. Б. Г. Теорія ймовірностей, випадкові процеси та математична статистика. К.: Техніка, 2004. 288 с.

40 Про безпеку та здоров’я працівників на роботі : від 13.10.2023, № № 10147. [Електронний ресурс]. – Режим доступу до ресурсу: https://ips.ligazakon.net/document/JI10117A?an=2 (дата звернення: 03.06.2025).

41 Надія Куравська Прояви захищеності як чинника психологічної безпеки студентів в умовах ВНЗ // 28 січня 2016 року. – Збірник наукових праць: психологія №21, 20 серпня 2021. – С. 79-86.

42 Методичні рекомендації щодо запровадження психосоціальної підтримки на робочому місці для роботодавців. Міністерство економіки України. [Електронний ресурс]. – Режим доступу до ресурсу: https://surl.li/wnrkzy (дата звернення: 03.06.2025).

43 Стручок В.С. Безпека в надзвичайних ситуаціях. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання / В.С.Стручок. — Тернопіль: ФОП Паляниця В. А., 2022. — 156 с.

44 В. Божко, А. Гінайло, О. Гончаров, М. Зеленкевич, В. Мозирський, В. Плакидюк, Д. Розинський Електробезпека в будівлях і спорудах. Вимоги До захисних заходів від ураження електричним струмом // ДСТУ Б В.2.5-82:2016 // 1 липня 2016. – С. 1-110. ДЕРЖСПОЖИВСТАНДАРТ України.

45 О. Бондаренко, О. Євдін, К. Жебровська, М. Кучма, В. Скакун, А. Ющенко Джерела фізичного походження природних надзвичайних ситуацій ДСТУ 4934:2008 // 20 липня 2008 року. – ДЕРЖСПОЖИВСТАНДАРТ України. – С. 1-12.

ДОДАТКИ

Додаток А

Програмний код індексної сторінки веб-сервісу

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8" />
  <meta   name="viewport"   content="width=device-width,   initial-
scale=1.0" />
  <title>Оптимальний догляд за шкірою</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>
<header class="header">
  SKIN TYPE
</header>
<section class="banner-cropped">
  <div class="banner-image-wrapper">
    <img src="/banner.png" alt="skin banner" />
  </div>
  <div class="banner-buttons-wrapper">
    <button                                    class="banner-button"
onclick="window.location.href='/form.php'">
      <span   class="btn-text">ДІЗНАЙСЯ   СВІЙ   ТИП   ШКІРИ   (тиць
сюди*)</span>
      <span class="btn-arrow">→</span>
    </button>
    <button         class="banner-button         banner-button-alt"
onclick="window.location.href='/products.php'">
      <span class="btn-text">Я вже знаю / переглянути продукти (тиць
сюди*)</span>
      <span class="btn-arrow">→</span>
    </button>
  </div>
</section>
<section class="image-grid">
  <img src="/img1.jpg" alt="Image 1">
  <img src="/img2.jpg" alt="Image 2">
  <img src="/img3.jpg" alt="Image 3">
</section>
<section class="advantages">
  <div class="advantage-block light">
    <h2>Індивідуальний підхід</h2>
    <p>Наш тест враховує унікальні особливості вашої шкіри.</p>
  </div>
  <div class="advantage-block colored">
    <h2>Наукова база</h2>
    <p>Рекомендації базуються на дерматологічних дослідженнях.</p>
  </div>
  <div class="advantage-block colored">
```

```
    <h2>Легкість у використанні</h2>
    <p>Швидкий тест з миттєвим результатом і порадами.</p>
  </div>
  <div class="advantage-block light">
    <h2>Глибокий аналіз</h2>
    <p>Оцінюємо тип, стан та потреби вашої шкіри.</p>
  </div>
</section>
<section class="image-grid">
  <img src="/img4.jpg" alt="Image 4">
  <img src="/img5.jpg" alt="Image 5">
  <img src="/img6.jpg" alt="Image 6">
</section>
<section class="method">
  <div class="method-header">
    <h2>Як ми визначаємо ваш тип шкіри</h2>
    <p>Процес складається з трьох простих, але глибоких кроків:</p>
  </div>
  <div class="method-steps">
    <div class="step-block">
      <div class="step-number">1</div>
      <div class="step-content">
        <h3>Аналіз відповідей</h3>
        <p>Ми оцінюємо ваші відповіді на запитання щодо типу,
реакцій та стану шкіри.</p>
      </div>
    </div>
    <div class="step-block">
      <div class="step-number">2</div>
      <div class="step-content">
        <h3>Алгоритмічна обробка</h3>
        <p>Наш алгоритм розраховує ваги та визначає базовий тип
шкіри.</p>
      </div>
    </div>
    <div class="step-block">
      <div class="step-number">3</div>
      <div class="step-content">
        <h3>Формування рекомендацій</h3>
        <p>Генеруємо індивідуальні поради, базуючись на отриманому
результаті.</p>
      </div>
    </div>
  </div>
</section>
<footer class="main-footer">
  <p>&copy; 2025 SkinCareTest. Усі права захищено.</p>
</footer>
</body>
</html>
```

Додаток Б

Програмний код файлу submit.php

```
<?php
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
session_start();

$problems = $_POST['problems'] ?? [];
$preferences = $_POST['preferences'] ?? [];
$allergens = $_POST['allergens'] ?? [];
$cosmetic_types = $_POST['cosmetic_type'] ?? [];

if (!is_array($cosmetic_types)) {
    $cosmetic_types = [$cosmetic_types];
}
if (empty($problems)) {
    $problems = ['нема'];
}
if (empty($preferences)) {
    $preferences = ['нема'];
}
if (empty($allergens)) {
    $allergens = ['нема'];
}

$data = [
    'skin_type' => $_POST['skin_type'] ?? '',
    'age_range' => $_POST['age'] ?? '',
    'budget' => $_POST['budget'] ?? '',
    'problems' => $problems,
    'cosmetic_types' => $cosmetic_types,
    'preferences' => $preferences,
    'allergens' => $allergens,
    'usage_time' => $_POST['usage_time'] ?? '',
    'priority' => [
        'price' => max(0, min(5, intval($_POST['priority_price'] ??
0))),
        'skin' => max(0, min(5, intval($_POST['priority_skin'] ??
0))),
        'natural'            =>            max(0,            min(5,
intval($_POST['priority_natural'] ?? 0))),
        'brand' => max(0, min(5, intval($_POST['priority_brand'] ??
0))),
    ],
];

$host = 'sql101.infinityfree.com';
$db = 'if0_39059945_cosmetic_selector';
```

```
$user = 'if0_39059945';
$pass = 'tkMN5FofA2yv';

try {
    $dsn = "mysql:host=$host;dbname=$db;charset=utf8mb4";
    $pdo = new PDO($dsn, $user, $pass, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
} catch (PDOException $e) {
    echo json_encode(['error' => 'Database connection failed: ' .
$e->getMessage()]);
    exit;
}

$skin_type = $data['skin_type'] ?? null;
$age_range = $data['age_range'] ?? null;
$problems = $data['problems'] ?? [];
$cosmetic_types = $data['cosmetic_types'] ?? [];
$preferences = $data['preferences'] ?? [];
$allergens = $data['allergens'] ?? [];
$usage_time = $data['usage_time'] ?? null;
$weights = $data['priority'] ?? [];
$skin_type_id = getIdByName($pdo, 'skin_types', $skin_type);
$age_range_id = getIdByName($pdo, 'ages', $age_range);
$usage_time_id = getIdByName($pdo, 'usage_times', $usage_time);

file_put_contents('debug_ids.txt',                    "skin_type_id:
$skin_type_id\nage_range_id:          $age_range_id\nusage_time_id:
$usage_time_id\n");

function getIdByName($pdo, $table, $name) {
    if (!$name) return null;
    $stmt = $pdo->prepare("SELECT id FROM $table WHERE name = ?");
    $stmt->execute([$name]);
    return $stmt->fetchColumn();
}

function getIdsByNames($pdo, $table, $names) {
    if (empty($names)) return [];
    $placeholders = implode(',', array_fill(0, count($names), '?'));
    $sql = "SELECT id FROM $table WHERE name IN ($placeholders)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute($names);
    return $stmt->fetchAll(PDO::FETCH_COLUMN);
}

function getTagIdsByNamesAndType($pdo, $names, $tag_type) {
    if (empty($names)) return [];
    $placeholders = implode(',', array_fill(0, count($names), '?'));
    $sql = "SELECT pt.id FROM product_tags pt WHERE pt.name IN
($placeholders) AND pt.tag_type = ?";
    $stmt = $pdo->prepare($sql);
    $params = array_merge($names, [$tag_type]);
```

```
    $stmt->execute($params);
    return $stmt->fetchAll(PDO::FETCH_COLUMN);
}

function productHasAllTags($pdo, $product_id, $required_tag_ids) {
    if (empty($required_tag_ids)) return true;
    $placeholders = implode(',', array_fill(0,
count($required_tag_ids), '?'));
    $params = array_merge([$product_id], $required_tag_ids);
    $sql = "SELECT COUNT(DISTINCT tag_id) FROM product_tags_link
WHERE product_id = ? AND tag_id IN ($placeholders)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);
    $count = $stmt->fetchColumn();
    return $count == count($required_tag_ids);
}

function productHasNoneTags($pdo, $product_id, $excluded_tag_ids) {
    if (empty($excluded_tag_ids)) return true;
    $placeholders = implode(',', array_fill(0,
count($excluded_tag_ids), '?'));
    $params = array_merge([$product_id], $excluded_tag_ids);
    $sql = "SELECT COUNT(*) FROM product_tags_link WHERE product_id
= ? AND tag_id IN ($placeholders)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);
    $count = $stmt->fetchColumn();
    return $count == 0;
}

$skin_type_id = getIdByName($pdo, 'skin_types', $skin_type);
$age_range_id = getIdByName($pdo, 'ages', $age_range);
$all_ages_id = getIdByName($pdo, 'ages', 'Yci');
$usage_time_id = getIdByName($pdo, 'usage_times', $usage_time);
$problem_ids = getTagIdsByNamesAndType($pdo, $problems, 'problem');
$preference_ids = getTagIdsByNamesAndType($pdo, $preferences,
'preference');
$allergen_ids = getTagIdsByNamesAndType($pdo, $allergens,
'allergen');
$cosmetic_type_ids = getIdsByNames($pdo, 'cosmetic_types',
$cosmetic_types);
$stmt = $pdo->query("SELECT * FROM products");
$all_products = $stmt->fetchAll(PDO::FETCH_ASSOC);
$filtered_products = [];

foreach ($all_products as $product) {
    $pid = $product['id'];

    // Вивід для дебагу
    file_put_contents('debug_log.txt', "Product ID:
{$product['id']} Age ID: {$product['age_id']}\n", FILE_APPEND);
```

```
    if ($skin_type_id && $product['skin_type_id'] != $skin_type_id)
continue;

    if ($age_range_id && $age_range_id != $all_ages_id) {
        if     ($product['age_id']     !=     $age_range_id     &&
$product['age_id'] != $all_ages_id) {
            file_put_contents('debug_log.txt',   "Skipped   product
{$product['id']} by age filter\n", FILE_APPEND);
            continue;
        }
    }

    if    ($usage_time_id    &&    $product['usage_time_id']    !=
$usage_time_id) continue;

    if  (!empty($problem_ids)  &&  !productHasAllTags($pdo,  $pid,
$problem_ids)) continue;

    if (!empty($cosmetic_type_ids)) {
        $placeholders      =      implode(',',      array_fill(0,
count($cosmetic_type_ids), '?'));
        $stmt     =     $pdo->prepare("SELECT     COUNT(*)     FROM
product_cosmetic_types_link    WHERE    product_id    =    ?    AND
cosmetic_type_id IN ($placeholders)");
        $params = array_merge([$pid], $cosmetic_type_ids);
        $stmt->execute($params);
        $count = $stmt->fetchColumn();
        if ($count < count($cosmetic_type_ids)) continue;
    }

    if (!empty($preference_ids) && !productHasAllTags($pdo, $pid,
$preference_ids)) continue;

    if  (!empty($allergen_ids)  &&  !productHasNoneTags($pdo,  $pid,
$allergen_ids)) continue;

        if (!empty($fail_reasons)) {
        file_put_contents('filter_debug.txt',   "Product   ID   $pid
failed: " . implode(', ', $fail_reasons) . "\n", FILE_APPEND);
        continue;
    }

    $filtered_products[] = $product;
}

if (empty($filtered_products)) {
    echo   json_encode(['error'   =>   'No   products   matched   the
filters']);
    exit;
}

$payload = [
    'products' => $filtered_products,
```

```
    'weights' => $weights,
];

$flask_url = 'https://yuliiaihnat.pythonanywhere.com/process';

$ch = curl_init($flask_url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch,        CURLOPT_HTTPHEADER,        ['Content-Type:
application/json']);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($payload));

file_put_contents('log_payload.json',         json_encode($payload,
JSON_PRETTY_PRINT));

$response = curl_exec($ch);

if (curl_errno($ch)) {
    echo  json_encode(['error'  =>  'Flask  request  failed:  '  .
curl_error($ch)]);
    curl_close($ch);
    exit;
}
curl_close($ch);

file_put_contents('log_response.txt', $response);
file_put_contents('log_flask_response.json', $response);

$result = json_decode($response, true);

if (json_last_error() !== JSON_ERROR_NONE) {
    echo  json_encode(['error'  =>  'Invalid  JSON  from  Flask:  '  .
json_last_error_msg()]);
    exit;
}

if (!isset($result['ranked_products'])) {
    echo json_encode(['error' => 'Invalid Flask response: missing
ranked_products']);
    exit;
}

$_SESSION['products_ranked'] = $result['ranked_products'];
header('Location: results.php');
exit;
?>
```

Додаток В

Програмний код файлу app.py

```
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/process', methods=['POST'])
def process():
    data = request.get_json()

    products = data.get('products', [])
    weights = data.get('weights', {})

    if not products or not weights:
        return jsonify({'error': 'No products or weights
provided'}), 400

    criteria = list(weights.keys())

    # Підготовка до нормалізації
    norm_values = {c: [] for c in criteria}
    for product in products:
        for c in criteria:
            val = product.get(c, 0)
            try:
                val = float(val)
            except (ValueError, TypeError):
                val = 0
            norm_values[c].append(val)

    min_vals = {c: min(norm_values[c]) if norm_values[c] else 0 for
c in criteria}
    max_vals = {c: max(norm_values[c]) if norm_values[c] else 1 for
c in criteria}

    def normalize(value, c):
        min_v = min_vals[c]
        max_v = max_vals[c]
        if max_v == min_v:
            return 1
        if c == 'price':
            return (max_v - value) / (max_v - min_v)
        else:
            return (value - min_v) / (max_v - min_v)

    # Розрахунок score
    for product in products:
        score = 0
        for c in criteria:
```

```
            try:
                val = float(product.get(c, 0))
            except (ValueError, TypeError):
                val = 0
            norm_val = normalize(val, c)
            score += norm_val * weights.get(c, 0)
        product['score'] = score

    # Сортування та підготовка результату
    products_sorted = sorted(products, key=lambda x: x['score'],
reverse=True)
    ranked_products = [
        {
            'product_id': p['id'],
            'score': round(p['score'] * 100, 2)  # шкала від 0 до
100
        }
        for p in products_sorted
    ]

    return jsonify({'ranked_products': ranked_products})

@app.route('/')
def hello():
    return "Flask API працює!"

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

Додаток Г

Програмний код файлу results.php

```
<?php
session_start();
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
if             (!isset($_SESSION['products_ranked'])             ||
empty($_SESSION['products_ranked'])) {
    echo "<p>No products found. Please go back and submit the form
again.</p>";
    exit;
}
$host = 'sql101.infinityfree.com';
$db = 'if0_39059945_cosmetic_selector';
$user = 'if0_39059945';
$pass = 'tkMN5FofA2yv';

try {
    $pdo  =  new  PDO("mysql:host=$host;dbname=$db;charset=utf8mb4",
$user, $pass, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
} catch (PDOException $e) {
    echo "<p>Database error: " . htmlspecialchars($e->getMessage())
. "</p>";
    exit;
}
$ranked_products = $_SESSION['products_ranked'];
$product_ids = array_column($ranked_products, 'product_id');
$scores = array_column($ranked_products, 'score', 'product_id');

if (empty($product_ids)) {
    echo "<p>No valid product IDs to display.</p>";
    exit;
}
$placeholders  =  implode(',',  array_fill(0,  count($product_ids),
'?'));
$stmt = $pdo->prepare("SELECT id, name, description, usage_time_id
FROM products WHERE id IN ($placeholders)");
$stmt->execute($product_ids);
$products = $stmt->fetchAll(PDO::FETCH_ASSOC);

function   getLinkedNames($pdo,   $table,   $link_table,   $field,
$product_id) {
    $stmt = $pdo->prepare("
        SELECT $table.name
        FROM $link_table
```

```
        JOIN $table ON $link_table.$field = $table.id
        WHERE $link_table.product_id = ?
    ");
    $stmt->execute([$product_id]);
    return $stmt->fetchAll(PDO::FETCH_COLUMN);
}

function getUsageTimeName($pdo, $usage_time_id) {
    $stmt = $pdo->prepare("SELECT name FROM usage_times WHERE id =
?");
    $stmt->execute([$usage_time_id]);
    return $stmt->fetchColumn() ?: 'N/A';
}
?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Product Recommendations</title>
</head>
<body>
<h1>Recommended Products</h1>
<?php foreach ($products as $product): ?>
    <?php
        $id = $product['id'];
        $score = round($scores[$id] ?? 0, 2);
        $types     =     getLinkedNames($pdo,     'cosmetic_types',
'product_cosmetic_types_link', 'cosmetic_type_id', $id);
        $usage_time             =             getUsageTimeName($pdo,
$product['usage_time_id']);
    ?>
    <div class="product">
        <h2><?= htmlspecialchars($product['name']) ?></h2>
        <p><?= htmlspecialchars($product['description']) ?></p>
        <p class="score">Match score: <?= $score ?> / 100</p>
        <div class="tags">
            <strong>Types:</strong> <?= implode(', ', $types) ?><br>
            <strong>Usage:</strong>                                  <?=
htmlspecialchars($usage_time) ?>
        </div>
    </div>
<?php endforeach; ?>

</body>
</html>
```