**Міністерство освіти і науки України**

**A blue and red logo

Description automatically generated**

**ВСЕУКРАЇНСЬКА ЦЕНТРАЛЬНА СПІЛКА СПОЖИВЧИХ ТОВАРИСТВ (УКРКООПСПІЛКА)**

Рівненський кооперативний економіко-правовий фаховий коледж

# Звіт

***З технологічної практики***

***студента Свинобоя Максима Олеговича***

групи ІТ-41

Період практики з «27» січня2025 року до «21» лютого 2025 року

Галузь знань ***12 Інформаційні технології***

Спеціальність ***121 Інженерія програмного забезпечення***

***Місце практики***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Керівник практики від організації***

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(підпис) (прізвище, ім'я, по-батькові)*

***МП***

***Керівник практики від закладу освіти*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Рівне 2025***

# Зміст

Вступ

Календарно-тематичний план

Щоденник практики

Основна частина

Висновки

Список використаних джерел

Додатки

# **Вступ**

Я, Свинобой Макисм Олегович, проходив виробничу практику з 27.01.2025 по 21.02.2025 на базі Приватного підприємства бюро інноваційних технологій, яке займається розробкою та впровадженням програмного забезпечення. Метою практики було не лише закріплення теоретичних знань, отриманих під час навчання в коледжі, а й набуття практичних навичок у реальних умовах виробництва, що є надзвичайно важливим для мого професійного зростання.

На початку практики я мав можливість ознайомитися з організаційною структурою підприємства та його внутрішнім розпорядком. Це дало змогу мені зрозуміти, як побудована система управління, які функції виконують різні відділи, а також як дотримуються вимоги охорони праці та техніки безпеки. Завдяки цьому я зміг швидко адаптуватися до робочого середовища і усвідомити значення кожного етапу виробничого процесу.

Подальший етап практики був присвячений розбору основних напрямів діяльності підприємства, зокрема тих, що пов’язані з розробкою програмних продуктів. Я мав можливість детально вивчити специфіку роботи різних відділів, дізнатися про методи організації праці програмістів і тестувальників, що дозволило мені краще зрозуміти реалії виробничої практики в IT-сфері.

Наступним важливим завданням було долучення до формування технічного завдання. Разом із керівником практики та колегами я брав участь у зборі та аналізі вимог до майбутнього програмного продукту. Цей етап допоміг мені оволодіти методами збору інформації, аналізу специфікацій і формування користувацьких історій, що є важливими навичками для подальшої розробки та впровадження програмних рішень.

Після цього я ознайомився з процесом проєктування архітектури програмного продукту, що включає створення баз даних, розробку інтерфейсу користувача та реалізацію бізнес-логіки. Завдяки цьому досвіду я навчився бачити зв’язок між різними компонентами системи та зрозумів, як важливо правильно інтегрувати їх для досягнення високої якості кінцевого продукту.

На завершальному етапі практики я долучився до розробки, тестування та розгортання програмного продукту або його окремих фрагментів. Це дозволило мені на практиці застосувати набуті знання, провести комплексне тестування та виправити виявлені недоліки. Також я здійснив узагальнення здобутих під час практики знань і підготував письмовий звіт із відповідними додатками

Актуальність теми практики полягає в тому, що сучасний ринок інформаційних технологій вимагає від майбутніх фахівців не лише глибоких теоретичних знань, а й здатності ефективно працювати в команді, аналізувати вимоги замовника, створювати масштабовані та надійні програмні рішення, проводити комплексне тестування та надавати кваліфіковану підтримку програмних продуктів. Отримані знання та практичні навички є безцінними для мого подальшого навчання і кар’єрного зростання в ІТ-галузі, адже вони дозволяють мені впевнено орієнтуватися в сучасних виробничих процесах та адаптуватися до постійно змінного технологічного середовища.

# Календарно-тематичний план

|  |  |  |
| --- | --- | --- |
| **Дата** | **Тема/Завдання** | **Опис виконаної роботи** |
| 27.01.2025 | Ознайомлення з підприємством та вступний інструктаж | Я прибув на підприємство, ознайомився з його основними видами діяльності та структурою управління. Пройшов інструктаж з охорони праці та техніки безпеки, підписав необхідні документи та узгодив графік робіт із керівником. |
| 28.01.2025 | Вивчення організаційної структури та старт формування ТЗ | Я ознайомився з організаційною структурою підприємства, дізнався про компетенції основних відділів (розробки, тестування, підтримки) та вивчив внутрішній статут. Разом із керівником розпочав формування технічного завдання для веб-сервісу. |
| 29.01.2025 | Збір та аналіз вимог | Я проаналізував функції окремих відділів, зібрав необхідну інформацію для визначення вимог до веб-сервісу. Обговорив поточні проєкти з розробниками та уточнив ключові завдання практики, сформував попередній варіант списку вимог. |
| 30.01.2025 | Методи збору вимог та формування користувацьких історій | Я вивчив методи збору та фіксації вимог (інтерв’ю, аналіз специфікацій, опитувальники) та розпочав роботу над уточненням функціональних і нефункціональних вимог. Разом із керівником склав список основних користувацьких історій (use cases). |
| 31.01.2025 | Формування технічного завдання та аналіз ризиків | Я завершив формування технічного завдання, створив діаграми прецедентів для візуалізації функціоналу та провів аналіз можливих ризиків реалізації веб-сервісу. |
| 03.02.2025 | Розробка архітектури: побудова UML-діаграм | Я приступив до опису архітектури веб-сервісу, розглянув базові UML-діаграми (діаграми діяльності, об’єктні моделі) та обговорив з колегами можливі технології для реалізації бази даних. |
| 04.02.2025 | Моделювання даних та уточнення структури | Я створював UML-діаграми об’єктної моделі, уточнював взаємозв’язки між сутностями, звіряв отриману модель з вимогами технічного завдання та вносив необхідні корективи. |
| 05.02.2025 | Проектування графічного інтерфейсу користувача | Я ознайомлювався з принципами адаптивного дизайну та зручності використання, брав участь у макетуванні графічного інтерфейсу (вибір кольорової гами, розміщення елементів) та отримав схвалення керівника для подальшої розробки інтерфейсу веб-сервісу. |
| 06.02.2025 | Проектування бази даних | Я розпочав розробку серверної частини: проектував базу даних, створював ER-діаграми, налаштовував таблиці та зв’язки для подальшого тестування основних функцій веб-сервісу. |
| 07.02.2025 | Початкове наповнення бази даних | Я продовжив роботу над серверною частиною, здійснював початкове наповнення бази даних, організовував рівень доступу до даних через CRUD-операції, використовуючи Express та Mongoose для забезпечення взаємодії з клієнтською частиною. |
| 10.02.2025 | Реалізація бізнес-логіки: початок інтеграції | Я розпочав реалізацію бізнес-логіки веб-сервісу, підключав функції перевірки даних та інтегрував серверний API з клієнтським інтерфейсом. Також впровадив базову автентифікацію користувачів за допомогою JWT. |
| 11.02.2025 | Інтеграція компонентів та тестування зв’язку | Я продовжив інтеграцію клієнтської та серверної частин, тестував взаємодію між компонентами, перевіряв узгодженість роботи від введення даних до їх запису в базі та відображення результатів на інтерфейсі. |
| 12.02.2025 | Тестування: ручне та автоматизоване тестування | Я опановував методи ручного та автоматизованого тестування, писав модульні тести з використанням jest та інших інструментів, проводив пошук дефектів у коді і документував знайдені помилки для подальшого аналізу. |
| 13.02.2025 | Рефакторинг та вдосконалення коду | Я продовжив тестування веб-сервісу, аналізував виявлені дефекти, вносив корективи та застосовував прийоми рефакторингу для оптимізації функціональності системи. |
| 14.02.2025 | Розгортання та інтеграційне тестування | Я взяв участь у додатковому тестуванні веб-сервісу в інтегрованому середовищі, перевіряв взаємодію з публічним API та іншими сервісами, готував пакет поставки продукту. |
| 15.02.2025 | Інтеграційне тестування | Я продовжив інтеграційне тестування, перевіряв стабільність роботи системи та збір метрик продуктивності, документував результати та отримував зворотній зв’язок від колег щодо ефективності роботи веб-сервісу. |
| 16.02.2025 | Розгортання тестового середовища та налаштування | Я розгорнув тестову версію веб-сервісу на окремому середовищі, проводив налаштування всіх компонентів, забезпечував інтеграцію між серверною та клієнтською частинами та перевіряв коректність роботи системи. |
| 17.02.2025 | Підготовка остаточної версії та пакування продукту | Я збирав остаточну версію веб-сервісу, підготовлював пакет поставки, перевіряв якість коду і документацію, впевнився, що всі функції відповідають вимогам технічного завдання. |
| 18.02.2025 | Розгортання та детальне інтеграційне тестування | Я розгорнув тестову версію у фінальному тестовому середовищі, проводив детальне інтеграційне тестування, перевіряв стабільність роботи під навантаженням та документував результати тестування, включаючи знімки екрана і відгуки користувачів. |
| 19.02.2025 | Узагальнення результатів: аналіз та створення графіків | Я аналізував ключові характеристики веб-сервісу, готував графіки та звіти про досягнуті показники, порівнював результати з поставленими завданнями та формував попередні висновки для підсумкового звіту. |
| 20.02.2025 | Підготовка підсумкового звіту та оформлення висновків | Я продовжував роботу над підготовкою підсумкового звіту, оформлював результати роботи, аналізував ефективність функціонування веб-сервісу та записував власні пропозиції щодо можливих поліпшень для подальшого розвитку проекту. |
| 21.02.2025 | Завершення звіту: узагальнення та оформлення рекомендацій | Я завершував складання підсумкового звіту, остаточно узагальнив результати практики, сформував висновки та рекомендації щодо подальшого розвитку веб-сервісу, перевірив відповідність звіту вимогам ЄСКД та підготував його до захисту перед комісією. |

# **Щоденник практики**

Дата: 27.01.2025

1. Ознайомився з підприємством, його основними видами діяльності та структурою управління.
2. Прослухав інструктаж з охорони праці та техніки безпеки, підписав необхідні документи.
3. Домовився з керівником практики щодо плану проходження практики та графіку робіт.

Дата: 28.01.2025

1. Ознайомився з організаційною структурою підприємства та компетенціями основних відділів (розробки, тестування, підтримки).
2. Вивчив розпорядок роботи, внутрішній статут і вимоги до працівників.
3. Разом із керівником практики розпочав формування технічного завдання до майбутнього програмного продукту.

Дата: 29.01.2025

1. Проаналізував функції окремих відділів, зібрав необхідну інформацію для визначення вимог до програмного продукту.
2. Обговорив із фахівцями відділу розробки поточні проєкти та методи організації праці програмістів.
3. Уточнив ключові завдання практики з керівником, сформував попередній варіант списку вимог.

Дата: 30.01.2025

1. Вивчав методи збору та фіксації вимог до програмних продуктів (інтерв’ю, аналіз специфікацій, опитувальники).
2. Розпочав роботу над уточненням функціональних і нефункціональних вимог до програмного продукту.
3. Складав разом із керівником список основних користувацьких історій (use cases).

Дата: 31.01.2025

1. Завершив формування технічного завдання з урахуванням отриманих даних.
2. Почав робити діаграми прецедентів (use-case) для більшої наочності функціоналу.
3. Проаналізував можливі ризики під час реалізації програмного продукту.

Дата: 03.02.2025

1. Приступив до опису архітектури програмного продукту, розглянув базові UML-діаграми (Додаток 2)
2. Ознайомився з прикладами структурного та об’єктно орієнтованого підходу до проєктування.
3. Спільно з колегами обговорив можливі технології для реалізації бази даних.

Дата: 04.02.2025

1. Створював UML-діаграми об’єктної моделі, уточнював взаємозв’язки між основними сутностями. (Додаток 1)
2. Звіряв модель із вимогами, вносив виправлення та коригування структури.

Дата: 05.02.2025

1. Розглянув підходи до проєктування користувацького інтерфейсу (адаптивний дизайн, зручність використання).
2. Брав участь у макетуванні графічного інтерфейсу (підбір кольорової гами, розміщення елементів).
3. Узгодив початкові макети з керівником і отримав схвалення на подальшу розробку.

Дата: 06.02.2025 – 07.02.2025

1. Створював базу даних, налаштовував таблиці та зв’язки згідно з попередніми ER-діаграмами.
2. Здійснював початкове наповнення бази для тестування основних функцій програмного продукту.
3. Організовував рівень доступу до даних, опрацьовував модуль взаємодії з базою (CRUD-операції).

Дата: 10.02.2025 – 11.02.2025

1. Реалізовував бізнес-логіку програмного продукту, підключав функції перевірки даних.
2. Спільно з колегами виконував інтеграцію коду інтерфейсу з логікою та базою даних.
3. Перевіряв узгодженість усього комплексу: від введення даних на формі до запису в базі та відображення результату.

Дата: 12.02.2025 – 13.02.2025

1. Опановував методи ручного та автоматизованого тестування, проводив пошук і усунення помилок.
2. Фіксував знайдені дефекти в журналах для подальшого аналізу.
3. Застосовував основні прийоми рефакторингу (оптимізація коду, розподіл функцій на окремі модулі).

Дата: 14.02.2025 – 18.02.2025

1. Брав участь у додатковому тестуванні на інтегрованому середовищі, перевіряв взаємодію зі сторонніми сервісами.
2. Готував пакет поставки програмного продукту, ознайомився зі способами його розгортання.
3. Розгортав тестову версію на окремому середовищі, документував результати.

Дата: 19.02.2025 – 21.02.2025

1. Узагальнював ключові характеристики програмного продукту, готував графіки та звіти про досягнуті показники.
2. Формував висновки та пропозиції щодо можливих поліпшень.
3. Оформляв підсумковий звіт-щоденник.

# Основна частина звіту

Під час виробничої практики мені було доручено створити веб-сервіс для порівняння характеристик техніки з використанням публічного API. Це завдання вимагало не лише глибокого розуміння технологій, але й уважного аналізу специфіки роботи в IT-сфері, організації командної роботи та дотримання внутрішніх документальних процедур. Проєкт охоплював широкий спектр завдань — від початкового ознайомлення з особливостями корпоративного середовища та документації, до конкретних технічних аспектів розробки, тестування, інтеграції та розгортання готового продукту.

На самому початку я занурився в роботу, вивчаючи корпоративну культуру, внутрішні регламенти та особливості обліку документації. Цей етап був важливим для того, щоб швидко адаптуватися до умов підприємства, зрозуміти ролі колег і визначити, як ефективно співпрацювати у команді. Подібний підхід дозволив мені не лише сформувати чітке розуміння поставленої задачі, а й підготуватися до вирішення технічних викликів, які могли виникнути у подальшому процесі розробки.

Наступним кроком стало формування технічного завдання, яке включало визначення ключових функцій веб-сервісу: пошук моделей товарів за назвою, їх порівняння за різними характеристиками, можливість фільтрації та сортування, а також інтеграцію з публічним API для отримання актуальних даних. Під час цього етапу я активно спілкувався з керівником практики та колегами, обговорюючи потенційні труднощі, можливі методи вирішення проблем і ключові аспекти реалізації проєкту.

Весь процес розробки веб-сервісу можна розділити на декілька логічних етапів, кожен з яких відігравав свою роль у створенні повноцінного продукту. Спочатку я займався аналізом ринку та вивченням існуючих рішень, що дозволило мені з’ясувати, які підходи використовують конкуренти та як їх можна адаптувати у власному проєкті. Далі була розробка архітектури системи, де я створював UML-діаграми та моделював базу даних для зберігання інформації про товари, що дало змогу чітко визначити структуру взаємодії між різними компонентами.

На наступному етапі я зосередився на реалізації серверної логіки, використовуючи Node.js, Express та Mongoose. Ці технології забезпечили ефективну обробку запитів, надійну роботу з базою даних і можливість масштабування сервісу за потребою. Паралельно або після цього я переходив до розробки клієнтської частини, де використання React, Material UI та інших бібліотек дозволило створити зручний, інтуїтивно зрозумілий інтерфейс для кінцевих користувачів.

Особливу увагу було приділено процесу тестування, який включав як ручне, так і автоматизоване тестування з використанням Jest і supertest. Це дозволило виявити і усунути потенційні недоліки, підвищити стабільність роботи веб-сервісу і переконатися, що всі функції працюють належним чином. Нарешті, після завершення розробки та тестування, сервіс було розгорнуто у тестовому середовищі, де проводилися інтеграційні тести для перевірки коректності взаємодії клієнтської і серверної частин, а також роботи з публічним API.

Отже, створення цього веб-сервісу стало комплексним проєктом, який дозволив мені поєднати теоретичні знання з практичним досвідом, опанувати нові технології та удосконалити навички роботи в команді. Завдяки цьому завданню я отримав можливість докладно вивчити всі етапи розробки програмного забезпечення — від аналізу вимог і проектування архітектури до реалізації, тестування і розгортання готового продукту, що є безцінним досвідом для мого подальшого професійного зростання в IT-сфері.

## 1. Початкове занурення в проєкт

Перша стадія роботи була пов’язана з підготовкою: потрібно було ретельно ознайомитися з організаційною структурою підприємства, визначити відповідальних осіб і зрозуміти внутрішні робочі процеси. На цьому етапі я приділив чимало уваги правилам внутрішнього розпорядку, адже від їх дотримання залежить, наскільки гармонійно мені вдасться включитися у вже сформований колектив. Одночасно я прослухав інструктаж з охорони праці та техніки безпеки, перевірив усю необхідну документацію, домовився з керівником практики щодо основних орієнтирів проєкту та уточнив графік робіт.

Також важливим завданням стало вивчення цілей проєкту з боку підприємства. Я дізнався, що результат моєї роботи має задовольнити потреби користувачів, які шукають надійний і швидкий спосіб порівняти різну техніку з декількох джерел. Зрозумівши загальну ідею, я перейшов до конкретики — збору й формалізації вимог.

## 2. Формування технічних вимог та аналіз середовища

Перш ніж приступати безпосередньо до написання коду, слід було визначити, яким чином розподілити функціонал, як забезпечити інтеграцію з публічними API та які інструменти для цього найліпше підходять. На цьому етапі я переглянув існуючі сервіси, що надають можливості порівняння характеристик: ознайомився з їхнім інтерфейсом, архітектурою та функціями пошуку. Завдяки такому аналізу мені вдалося з’ясувати сильні та слабкі сторони рішень, присутніх на ринку, і зрозуміти, які моменти слід удосконалити у власному веб-сервісі.

Крім того, довелося вивчити декілька публічних API для отримання даних про товари. Я звертав увагу на критерії надійності, актуальності та гнучкості, аби обрати найпридатніший для інтеграції. У цей же період я багато спілкувався із фахівцями відділу розробки та тестування, що допомогло мені визначити початковий список функціональних і нефункціональних вимог.

### Перехід до технічного завдання

Отримані дані склав у вигляді попереднього технічного завдання, яке потім разом із керівником практики доповнював і уточнював упродовж проєкту. Зрештою, це технічне завдання відобразило структуру майбутнього веб-сервісу (модулі пошуку, порівняння, сортування та фільтрації, формування аналітичних звітів тощо), а також описувало способи отримання даних із зовнішніх джерел.

## 3. Проєктування архітектури: логічні та структурні діаграми

Розробка архітектури є основою для того, щоби згодом усі компоненти функціонували узгоджено й ефективно. Тому після формування вимог я створив UML-діаграми, які ілюстрували поведінку системи на різних рівнях. Зокрема, діаграми діяльності (Activity diagrams) допомогли відстежити послідовність дій користувача або системи, тоді як об’єктні моделі та ER-діаграми (для бази даних) дали змогу чітко уявити ключові сутності та взаємозв’язки між ними.

### Суть ER-діаграм

Оскільки я працював із даними про техніку, їх слід було належним чином структурувати. До прикладу, у базі даних могли бути сутності “Product” (товар), “Brand” (виробник), “Category” (категорія товару) тощо. Для кожної сутності я визначав поля (наприклад, назва товару, ціна, характеристики енергоефективності), а також зв’язки (1:1, 1:N або M:N). Така ретельна підготовка дозволила упорядкувати логіку взаємодії з базою й полегшила інтеграцію зовнішньої інформації.

## 4. Вибір технологій для серверної частини

Коли загальна схема системи була зрозумілою, я перейшов до конкретних інструментів, завдяки яким можна створити надійний сервер. Основні компоненти складалися з Node.js як середовища виконання, Express як каркасу для створення REST API та Mongoose для доступу до бази даних MongoDB. Щоб описати цей процес докладніше:

### Node.js

Обрав Node.js через його асинхронну природу, що забезпечує більш швидку обробку HTTP-запитів та кращу масштабованість. Також у мене вже був досвід використання npm (Node Package Manager) — це дуже зручно для керування залежностями та пакунками.

### Express

Саме Express робить створення серверної логіки більш структурованим: зокрема, я налаштовував маршрути (routes) на /products, /compare, /auth та інші ендпойнти, які потім приймали та передавали дані на клієнт. Такий підхід допоміг розмежувати функціонал: кожен маршрут відповідає за свій напрямок.

### Mongoose

Оскільки обсяги й види даних можуть бути різноманітними (різноманітні характеристики), я вирішив застосувати MongoDB. Mongoose, у свою чергу, полегшує роботу з MongoDB. Використовуючи схеми (Schemes) і моделі (Models), я міг визначити структуру документів та виконувати CRUD-операції (створення, читання, оновлення, видалення) у мінімум рядків коду.

### helmet, cors, compression

Щоби підвищити безпеку, я впровадив helmet: ця бібліотека автоматично встановлює низку заголовків HTTP, що захищають від поширених вразливостей на кшталт XSS або клікджекінгу. Пакет cors дав змогу конфігурувати, які саме домени та методи можуть надсилати запити. А compression стискав відповіді сервера перед їх відправленням, прискорюючи завантаження на кінцевих пристроях.

### jsonwebtoken (JWT)

Коли постало питання автентифікації користувача, я обрав JSON Web Tokens. За допомогою JWT зручно зберігати дані про користувача в зашифрованому токені, який передається при кожному запиті до сервера. Така схема дозволяє уникнути складної логіки зі створенням сесій та підвищує масштабованість рішення.

## 5. Реалізація клієнтської частини

Паралельно або відразу після перших успіхів на боці сервера я почав розробляти інтерфейс користувача. Цей інтерфейс мав забезпечувати зручне й наочне відображення продукції, а також надавати інструменти для пошуку, порівняння та сортування. Ця частина роботи стала великою мірою творчою, оскільки від UX/UI залежить, наскільки користувачі оцінять застосунок.

### React

Я обрав React, оскільки ця бібліотека дає змогу нарізати складний інтерфейс на окремі компоненти (наприклад, Table, SearchBar, CompareCard), що поліпшує читабельність і повторне використання коду. Техніка збереження стану (state management) та властивостей (props) у React дає змогу легко координувати роботу між різними частинами інтерфейсу.

### Material UI

Щоб мати привабливий вигляд і не витрачати зайвий час на дизайн, я підключив Material UI. Завдяки готовим компонентам (кнопкам, табам, панелям, карткам) я міг швидко скласти наочний та адаптивний інтерфейс, дотримуючись принципів Material Design.

### Emotion

Для більш тонкого контролю над стилями я застосував Emotion, яка дозволяє писати стилі в JavaScript/TypeScript, уникаючи зайвої плутанини у звичайних CSS-файлах. Це особливо корисно, коли треба змінювати стилі динамічно (наприклад, підсвічувати вибрану модель товару).

### axios

Для відправлення HTTP-запитів із фронтенду на сервер мені знадобилася axios — легка й гнучка бібліотека, що спрощує взаємодію з backend’ом. З її допомогою я отримував дані про товари, надсилав оновлення, а також перевіряв авторизацію користувачів.

### React Router

Основні розділи застосунку я організував як маршрути (routes). Наприклад, було створено роутинг /home для головної сторінки, /compare для порівняння моделей, /login для авторизації тощо. Це сприяло кращій структуризації коду й давало мені можливість швидко перемикатися між розділами.

## 6. Організація тестового циклу: ручне та автоматизоване тестування

Коли перші версії веб-сервісу вже працювали, я перейшов до тестування, аби переконатися у відсутності критичних помилок і перевірити зручність використання.

### Jest

Для модульних і деяких інтеграційних тестів я застосував Jest. Зокрема, це стосувалося перевірки, чи правильно функції обробляють API-виклики, чи не виникає винятків (exceptions) за певних умов. Jest має зручну систему генерації звітів і покриття тестами (coverage), що дає змогу вчасно виявляти прогалини.

### supertest

Для тестування серверної логіки на рівні ендпойнтів REST API я звертався до supertest: ця бібліотека імітує реальні запити, а також дає змогу легко стежити за статусами та вмістом відповідей. Такий підхід виявився надзвичайно корисним для підтвердження коректної роботи маршрутів Express і взаємодії з базою даних.

### Ручне тестування інтерфейсу

Зважаючи на важливість користувацького досвіду (UX), я проводив ручне тестування інтерфейсу: перевіряв, як відображаються елементи на різних розмірах екрану, чи коректно працюють спливаючі вікна та кнопки, чи зручно шукати та порівнювати характеристики. Це допомогло виявити дрібні незручності, які інакше могли би залишитися непоміченими в автоматизованих тестах.

### Рефакторинг і оптимізація

За результатами тестування я часто повертався до коду, виправляв неточності, упорядковував імпорти, змінював назви змінних для кращої читабельності тощо. Завдяки такому підходу вдалося забезпечити більш високу надійність і простоту обслуговування на усіх рівнях — від бази даних до компонента інтерфейсу.

## 7. Розгортання та фінальний контроль якості

Коли функціональні можливості веб-сервісу наблизилися до цілковитої готовності, настав час перевірити все “в бойових” умовах — розгорнути застосунок у тестовому середовищі.

### Vite

Для виробничої збірки фронтенду я скористався інструментом Vite, який прискорює процес розробки через гарячу перезавантаженість та надає зручну оптимізацію під час фінальної збірки. Зібрані, стискані файли я розмістив на веб-сервері, підключеному до основного домену або тестового піддомену.

У фінальному середовищі я зазвичай конфігурував змінні оточення: BASE\_URL для API, ключі для JWT, паролі до бази даних, а також інші налаштування безпеки. Це дозволяло швидко перемикатися між середовищами “розробка” (development) та “виробниче” (production), не змінюючи коду.

Наостанок я провів інтеграційне тестування, коли фронтенд, сервер і зовнішній публічний API почали взаємодіяти в умовах, наближених до реальних. Тут важливо було переконатися, що запити при великій кількості одночасних звернень до сервісу обробляються без збоїв, а інформація відображається коректно. Якщо десь виникали затримки, я аналізував логи сервера та вживав заходи щодо оптимізації.

Після цього я підготував і пакет поставки, і технічну документацію. У документації описав ключові ендпойнти, структури даних, приклади запитів і відповідей, варіанти взаємодії з інтерфейсом. Також сформував зведений звіт з усіх проведених тестів і рекомендації для майбутніх розробників (у разі, якщо сервіс потрібно розширювати в перспективі).

## 8. Підсумки реалізованого проєкту

Завдяки чіткому розподілу завдань і послідовному впровадженню кожного компонента мені вдалося створити веб-сервіс, який може ефективно порівнювати моделі техніки, відображати їхні характеристики та будувати наочні діаграми. Серверна логіка, що базується на Node.js, Express, Mongoose і бібліотеках безпеки, гарантує високу продуктивність і безпечну обробку даних, а клієнтська частина на основі React і Material UI забезпечує зручну навігацію й приємний дизайн.

Тестування, як ручне, так і автоматизоване (за допомогою Jest і supertest), підтвердило стабільність роботи, а також показало, що сервіс можна далі масштабувати або адаптувати під нові запити. Особисто для мене найскладнішим моментом було правильно організувати взаємодію з публічним API, адже потрібно було брати до уваги можливі затримки у відповіді та мінливість даних, які постачальник міг змінити без попередження. Тим не менш, застосування бібліотек axios та розумне кешування зробили процес обміну даними надійнішим.

Таким чином, у межах виробничої практики я набув не лише технічні навички, але й удосконалив уміння планувати робочий процес, координувати дії з колегами, складати коректну технічну документацію та вчитися на помилках. Завдяки гнучкій архітектурі та адекватно реалізованому функціоналу веб-сервіс не лише вирішує поставлені завдання, а й може бути легко доповнений чи модифікований у майбутньому. Усе це робить проєкт цінним досвідом для мого подальшого професійного розвитку в ІТ-галузі.

# Висновки

Виробнича практика дала мені можливість отримати неоціненний практичний досвід у розробці сучасних веб-сервісів. Проведений аналіз існуючих рішень дозволив мені обґрунтувати вибір оптимального публічного API, що забезпечує актуальність даних та має розширені можливості інтеграції. Розробка багаторівневої архітектури проекту допомогла мені зрозуміти важливість розділення функціональних компонентів та правильного визначення взаємодії між ними.

Реалізація клієнтської частини із застосуванням React, Material UI та axios дала можливість створити інтерфейс, орієнтований на зручність користувача, що значно спрощує процес взаємодії з системою. Серверна частина, побудована на базі Express і Mongoose, показала свою ефективність у роботі з великим обсягом даних і забезпеченні високого рівня безпеки завдяки використанню сучасних бібліотек, таких як helmet, cors та compression.

Ретельне тестування проекту, як ручне, так і автоматизоване за допомогою Jest і supertest, дозволило мені впевнитися в стабільності та надійності створеного веб-сервісу. Особливо важливим виявилося те, що процес тестування не лише допоміг виявити критичні помилки, але й дав можливість оптимізувати код, покращити структуру застосунку та підвищити продуктивність системи.

Отриманий досвід значно розширив мої професійні компетенції: я навчився аналізувати вимоги, формувати технічне завдання, розробляти комплексну архітектуру програмного продукту, вирішувати проблеми інтеграції зовнішніх API та оптимізувати процеси тестування. Практика також дала мені можливість освоїти методики організації командної роботи, удосконалити свої комунікативні навички, а також навчився ефективно управляти часом та ресурсами під час вирішення складних завдань.

Крім того, я зміг оцінити важливість використання сучасних технологій для забезпечення безпеки, масштабованості й високої продуктивності. Застосування JWT для автентифікації, а також впровадження helmet і cors для захисту серверної частини, стали ключовими аспектами, що забезпечили надійну роботу веб-сервісу навіть у випадку великого навантаження. Важливим аспектом була і оптимізація процесу розгортання, яку я реалізував за допомогою Vite, що дозволило значно прискорити збірку фронтенд-коду та полегшити управління проектом.

Загалом, виробнича практика стала важливим етапом у моєму професійному зростанні. Отримані знання та практичні навички допоможуть мені успішно впроваджувати сучасні технології у майбутніх проєктах, ефективно працювати в команді та вирішувати складні завдання у сфері розробки програмного забезпечення. Цей досвід створив міцну основу для подальшої кар’єри в ІТ-сфері, надихнув мене на подальший розвиток і вдосконалення власних професійних навичок, а також дав зрозуміти, що системний підхід до розробки може забезпечити високий рівень якості та надійності кінцевого продукту.

# Список використаних джерел

1. Програма виробничої практики для студентів спеціальності 121 Інженерія програмного забезпечення Рівненського кооперативного економіко-правового фахового коледжу.
2. Методичні вказівки до виконання та оформлення звіту з виробничої практики.
3. Нормативні документи з охорони праці та техніки безпеки, доступні на підприємстві.
4. Внутрішня документація підприємства (положення про відділи, посадові інструкції).
5. Документація публічних API - https://dummyjson.com/docs
6. Офіційна документація Node.js - https://nodejs.org/en/docs/
7. Офіційна документація Express - https://expressjs.com/
8. Документація Mongoose - https://mongoosejs.com/
9. Офіційна документація React - https://reactjs.org/
10. Документація Material UI - https://mui.com/
11. Документація Emotion - https://emotion.sh/docs/introduction
12. Документація axios - https://axios-http.com/
13. Документація React Router - https://reactrouter.com/
14. Документація JSON Web Tokens (jsonwebtoken) - https://github.com/auth0/node-jsonwebtoken
15. Офіційна документація Jest - https://jestjs.io/
16. Документація supertest - https://github.com/visionmedia/supertest
17. Документація Vite - https://vitejs.dev/

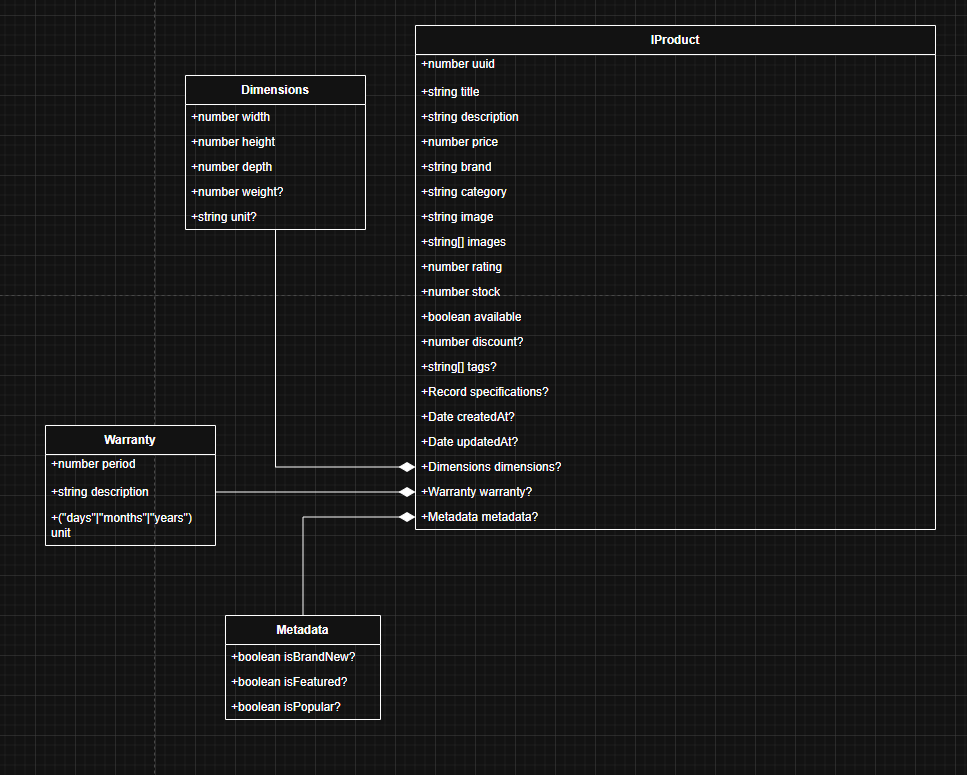
# Перелік додатків

|  |  |  |
| --- | --- | --- |
| **Назва додатка** | **Короткий опис** | **Сторінка** |
| Додаток 1 | Макет інтерфейсу IProduct |  |
| Додаток 2 | Структура усього проєкту |  |
| Додаток 3 | Use-Case діаграма |  |
| Додаток 4 | Додавання або редагування товару в адмінській частині |  |
| Додаток 5 | Макет веб-сайту на етапі розробки |  |
| Додаток 6 | Завантаження та відображення списку товарів на головній сторінці |  |
| Додаток 7 | Логіка оновлення таблиці порівняння |  |
| Додаток 8 | Логіка порівняння товарів (додавання та відображення списку) |  |
| Додаток 9 | Вибір категорії |  |
| Додаток 10 | Вибір товарів для порівняння |  |
| Додаток 11 | Таблиця порівняння |  |
| Додаток 12 | Темна тема, Детальний перегляд зображення |  |

# Додатки

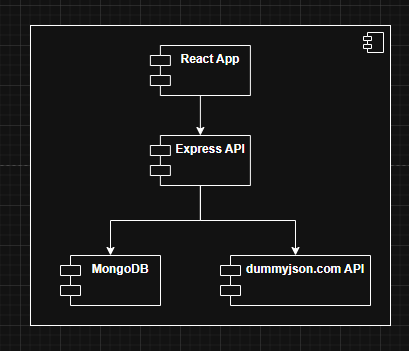
## Додаток 1

Макет інтерфейсу IProduct



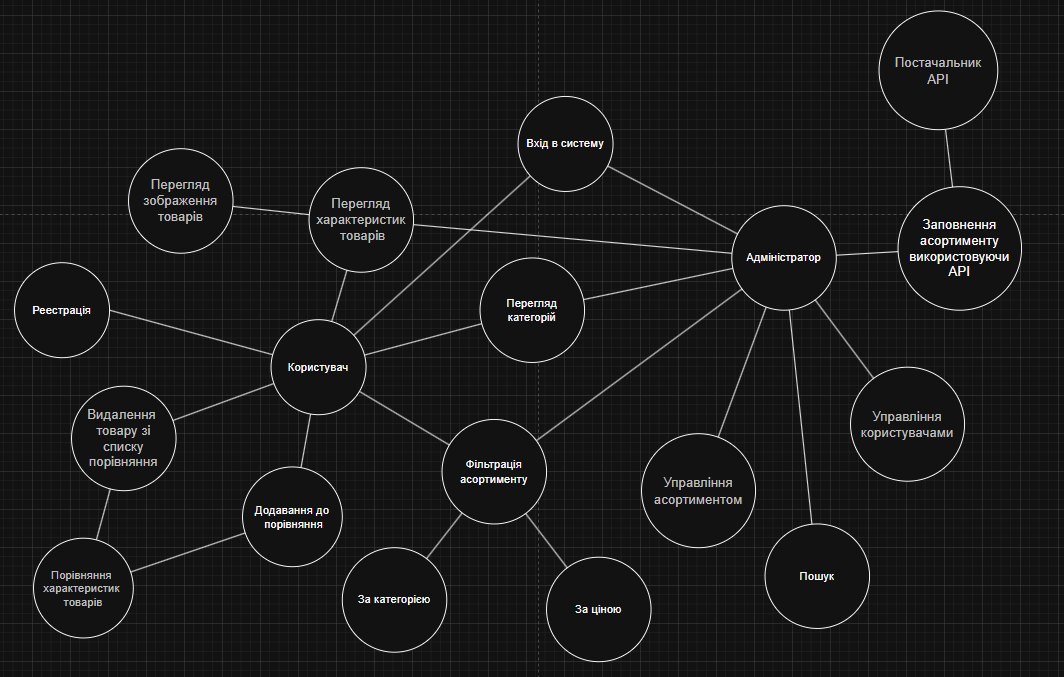
## Додаток 2

Структура усього проєкту



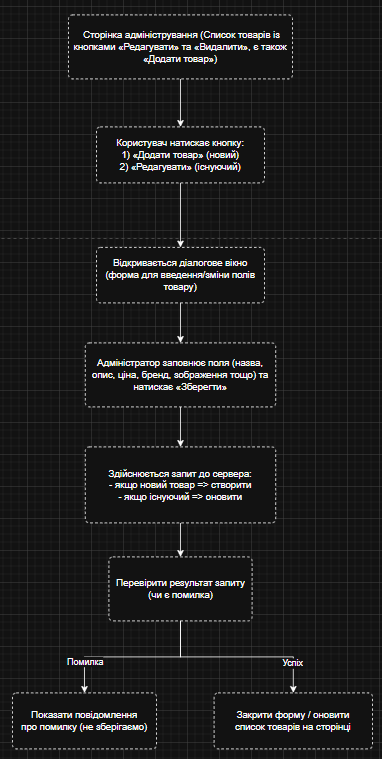
## Додаток 3

Use-Case діаграма



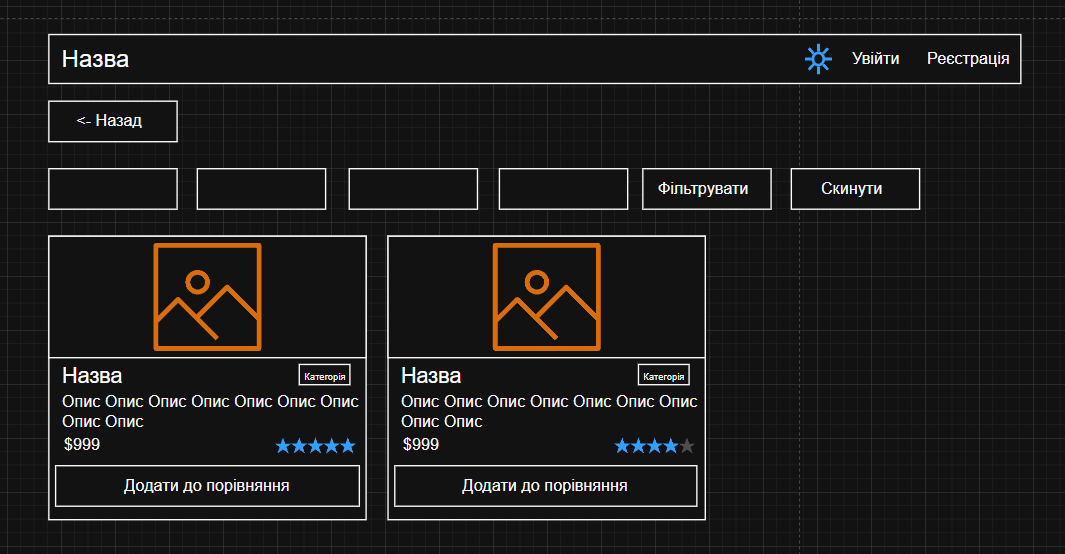
## Додаток 4

Додавання або редагування товару в адмінській частині

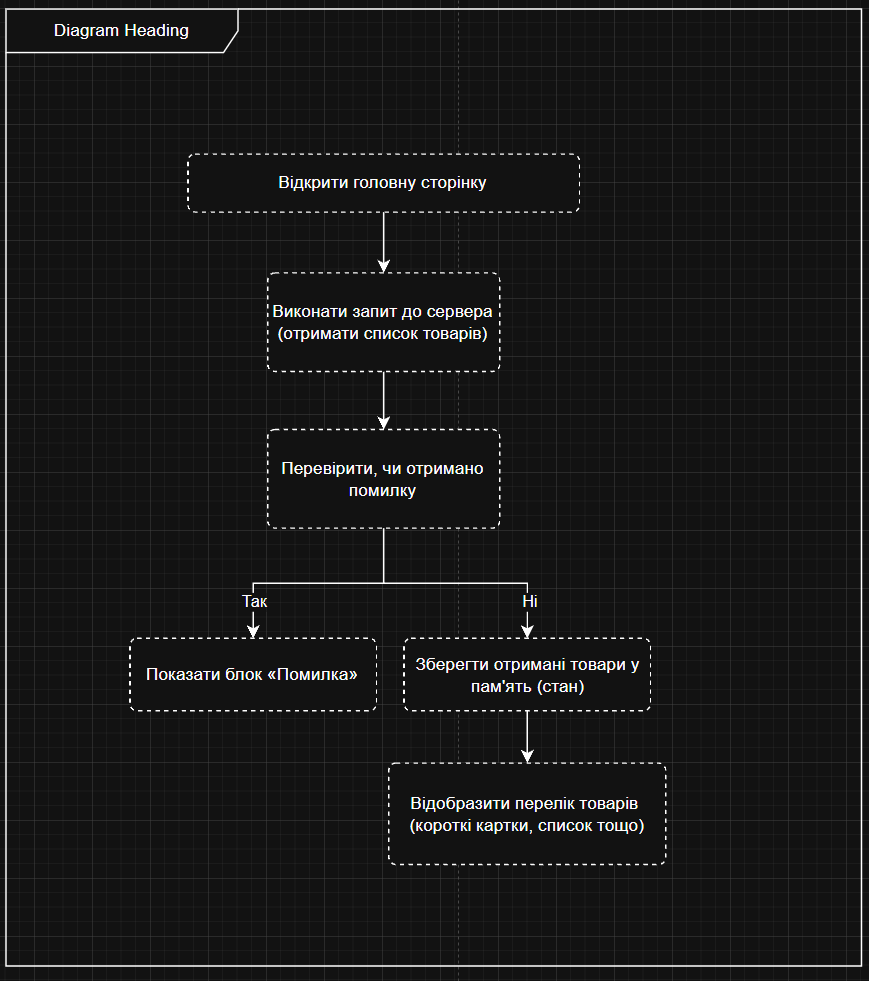


## Додаток 5

Макет веб-сайту на етапі розробки

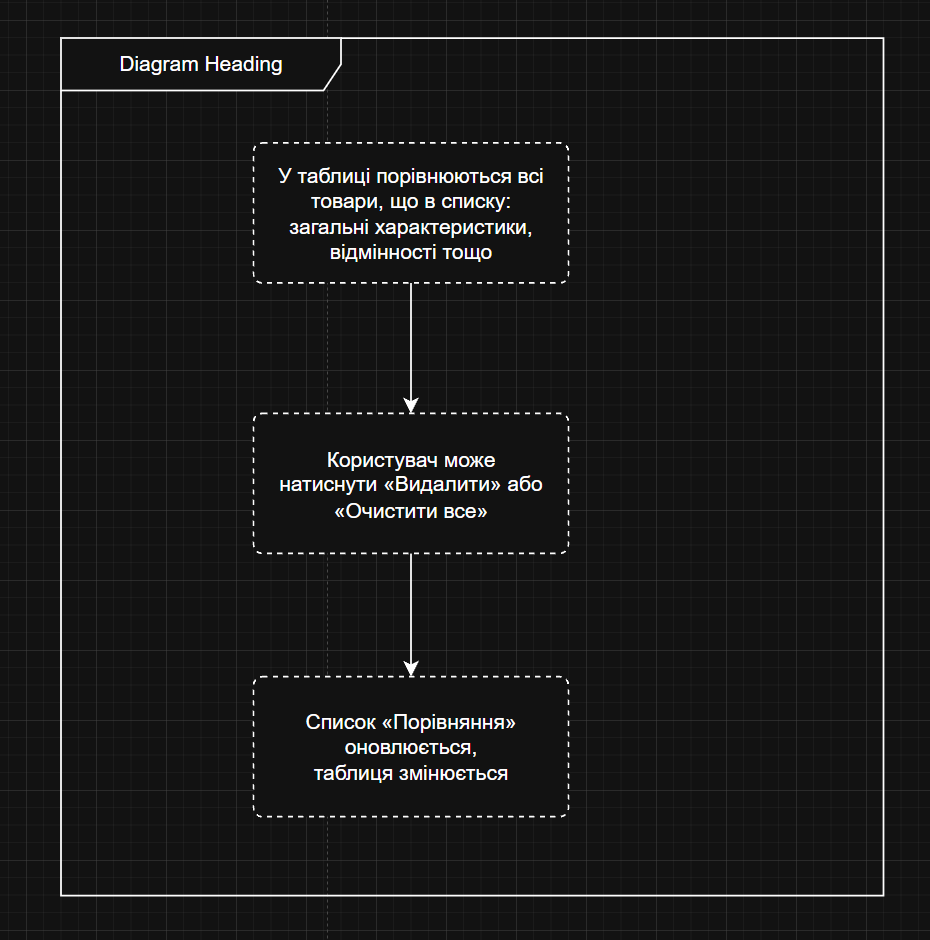


## Додаток 6

Завантаження та відображення списку товарів на головній сторінці

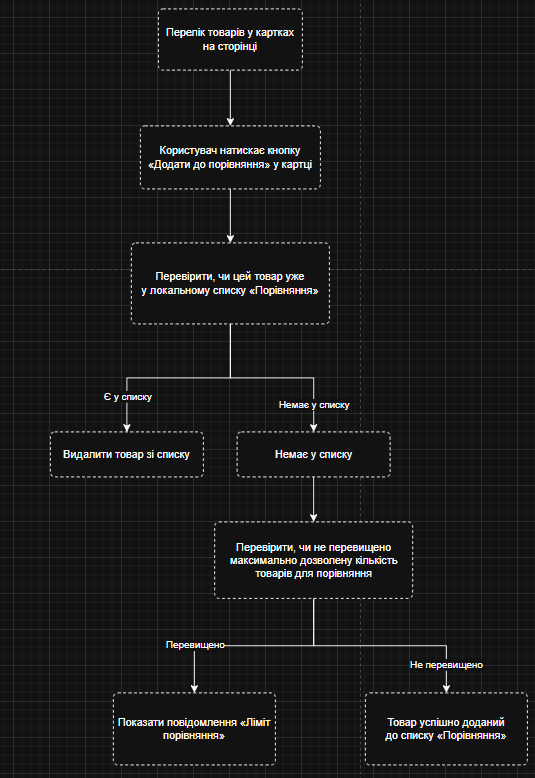
## Додаток 7

Логіка оновлення таблиці порівняння



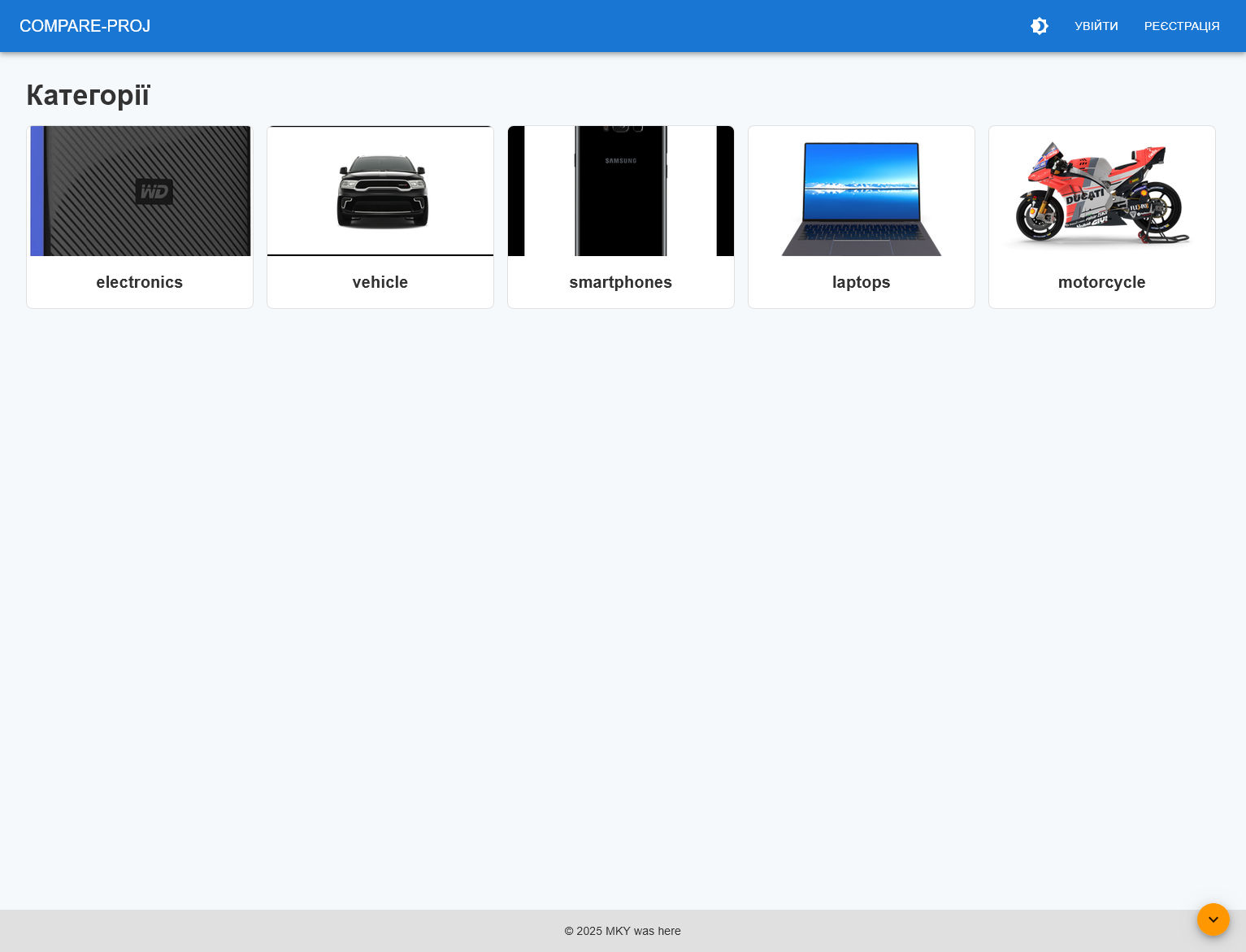
## Додаток 8

Логіка порівняння товарів (додавання та відображення списку)



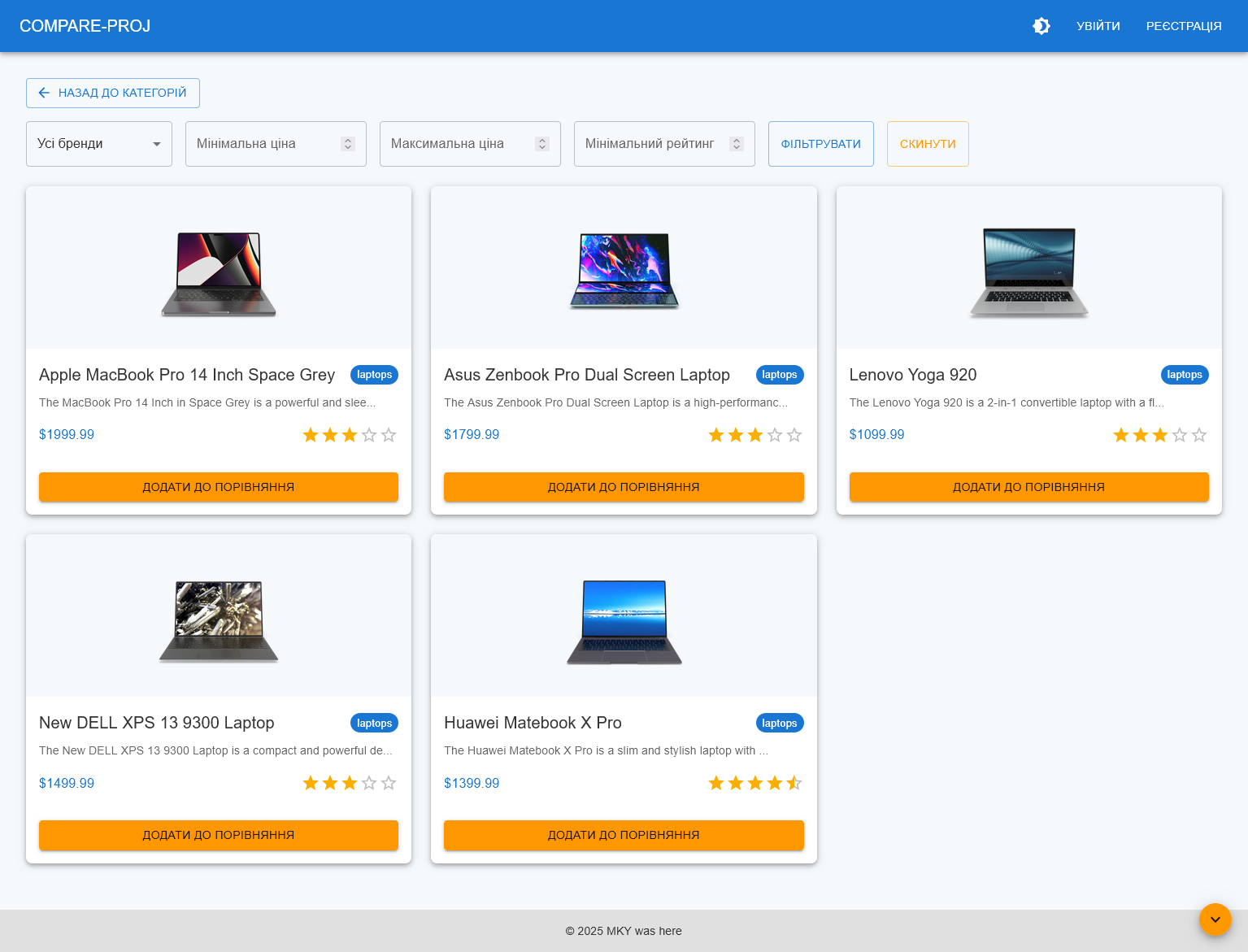
## Додаток 9

Фінальна версія веб-сайту (Вибір категорії)



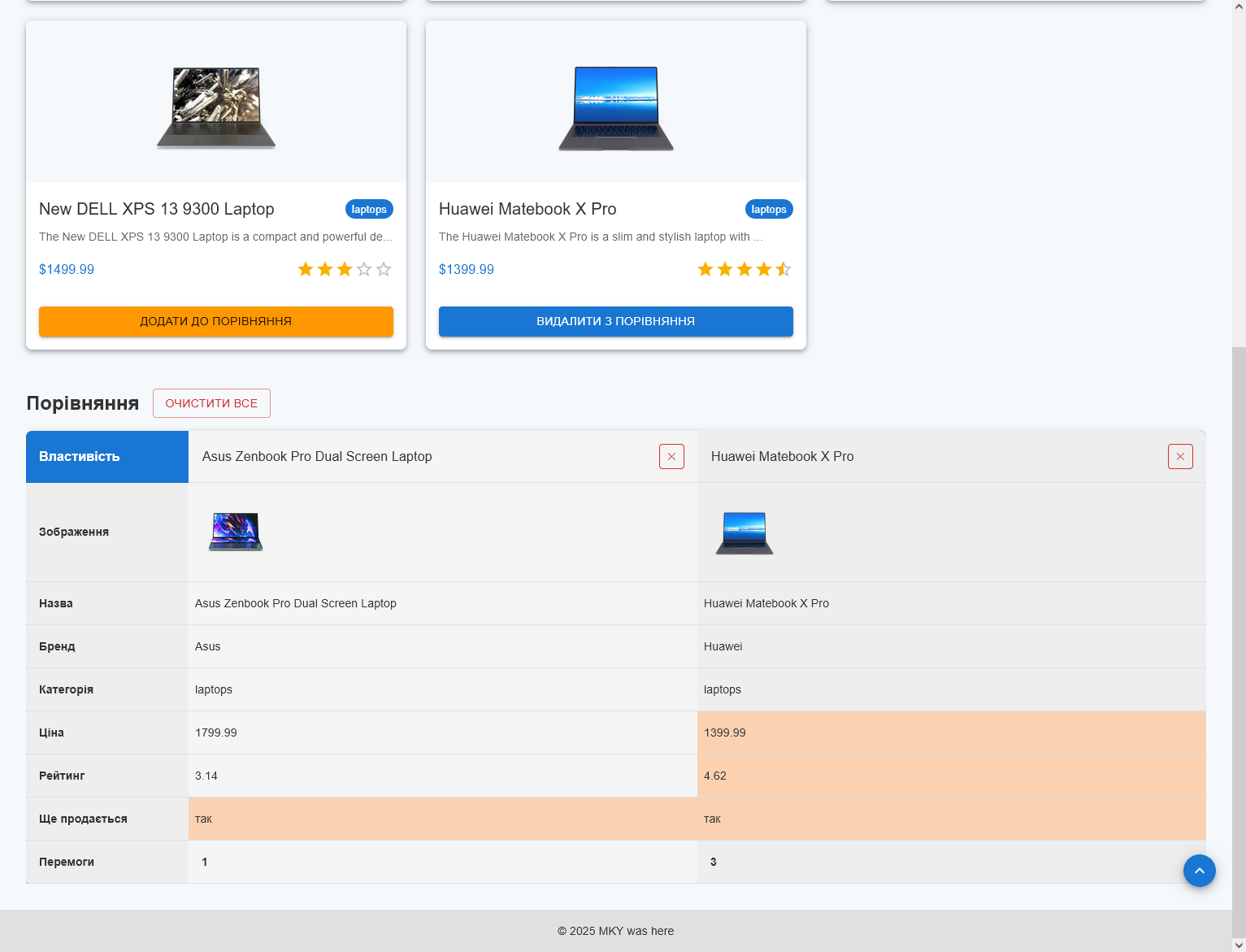
## Додаток 10

Фінальна версія веб-сайту (Вибір товарів для порівняння)



## Додаток 11

Фінальна версія веб-сайту (Таблиця порівняння)



## Додаток 12

Фінальна версія веб-сайту (Темна тема, Детальний перегляд зображення)

