КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет автоматизації і інформаційних технологій

Кафедра інформаційних технологій

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТР

на тему:

Розробка сайту для будівельного інтернет-магазину

Тулупов Гліб Михайлович
(прізвище, ім’я та по батькові здобувача повністю)

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет автоматизації і інформаційних технологій

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТ
_____
„____”_____2024 року

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТР

Розробка сайту для будівельного інтернет-магазину

Виконав: Тулупов Гліб Михайлович
(прізвище, ім’я та по батькові повністю)
122 «Комп`ютерні науки»
(спеціальність)
«Комп`ютерні науки»
(освітня програма)
Групи: КНм-23
Керівник: Поплавський О.А.
(прізвище та ініціали)
Кандидат технічних наук доцент
(вчене звання, науковий ступінь)

Ідентичність підтверджую

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: Автоматизації і інформаційних технологій

Кафедра: інформаційних технологій

Освітній рівень: «магістр» за ОП «Комп'ютерні науки»

Спеціальність: 122 «Комп'ютерні науки»

Освітня програма: «Комп'ютерні науки»

Спеціалізація: Інформаційні управляючі системи і технології

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТ
Гончаренко Т.А.
"___"_____2024 року

З А В Д А Н Н Я
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тулупов Гліб Михайлович

1. Тема роботи: Розробка сайту для будівельного інтернет-магазину, затверджена наказом ректора КНУБА № 2213/2 від «8» жовтня 2024р.
2. Керівник роботи: к.т.н. Поплавський О.А.
3. Строк подання здобувачем роботи до захисту: «__» листопада 2024 р.
4. Зміст пояснювальної записки за розділами:
   - Р. 1. Літературний огляд
   - Р. 2. Теоретична частина
   - Р. 3. Спосіб вирішення задачі
   - Р. 4. Реалізація задачі
   - Р. 5. Економічний аналіз
5. Інформаційні слайди
   - С.1. Титульний слайд
   - С.2. Вступ
   - С.3. Мета роботи
   - С.4. – С.8. Слайди які демонструють готовий сайт
   - С.9. – С.10. Слайди демонструючі як працює серверна частина

сайту

С.11. Структура сайту

С.12. Розрахунок економічного ефекту

С.13. Результат розрахунку економічного ефекту

С.14. Загальні висновки

С.15. Вихідний слайд

6. Календарний план виконання кваліфікаційної випускної роботи

Види робіт та їх зміст	Дата виконання
Р. 1. Літературний огляд	Вересень 2024 р.
Р. 2 Теоретична частина	Вересень 2024 р.
Р. 3. Спосіб вирішення задачі	Жовтень 2024 р.
Р.4. Реалізація задачі	Жовтень 2024 р.
Р.5. Економічний аналіз	Листопад 2024 р.
Остаточне оформлення роботи	Листопад 2024 р.
Направлення роботи на рецензування	Грудень 2024 р.
Попередній захист роботи на кафедрі	Грудень 2024 р.

7. Дата видачі завдання: «__»_____2024 р.

Заф.кафедри Гончаренко Т.А.

(підпис) (прізвище та ініціали)

Керівник Поплавський О.А.

(підпис) (прізвище та ініціали)

Студент Тулупов Г.М.

(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота за темою «Розробка сайту для будівельного інтернет-магазину» виконана студентом кафедри інформаційних технологій ФАІТ *Тулуповим Глібом Михайловичем зі спеціальності 122 «Комп'ютерні науки» за освітньою програмою «Комп'ютерні науки» та* складається зі: вступу; 5 розділів; (*літературний огляд, теоретична частина, спосіб вирішення задачі, реалізація задачі, економічний аналіз)*, висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 32 рисунки, 6 таблиць, 29 джерела, та 4 додатки. Загальний обсяг роботи 106 сторінок.

Актуальність теми. Оскільки сучасний світ висуває вимоги до підприємств, включаючи будівельну галузь, мати присутність в Інтернеті. Очікується, що сайт буде корисним інструментом для будівельних компаній. Особливо актуальним та корисним він може бути для нових компаній які тільки з'вилися на ринку і ще не мають великої різноманітності продукції та великої клієнтської бази.

Мета і завдання роботи.

Метою роботи є створення сайту будівельної компанії.

Її досягнення передбачає вирішення наступних *завдань*:

Аналіз вітчизняних та зарубіжних джерел.Збір та підготовка даних. Розробка бази даних. Створення сайту.

Використані методи. Для створення сайту було використану мову програмування JavaScript, клієнтської частини – фреймворк React. Для серверної частини фреймворки Node.js та Express. База даних була реалізована з технологією MySQL.

Отримані результати. Створено сайт, який надає змогу будівельній компанії розказати про себе та показати свою продукцію. Є можливість додавання нової продукції та регістрація користувачів. Дані про продукцію та користувачів зберігаються в базі даних.

ABSTRACT

The thesis on the topic "Development of a website for a construction company" was completed by a student of the department of information technologies of FAIT Tulupov Gleb Mikhailovich from the specialty 122 "Computer Science" according to the educational program "Computer Science" and consists of: introduction; 5 sections; (literature review, theoretical part, problem solving method, problem implementation, economic analysis), conclusions to each of these sections; general conclusions; a list of sources used, which includes 32 images, 6 tables, 29 sources, and appendices. The total volume of the work is 106 pages.

Relevance of the topic. Since the modern world places requirements on enterprises, including the construction industry, to have a presence on the Internet. It is expected that the site will be a useful tool for construction companies. It can be especially relevant and useful for new companies that have just emerged on the market and do not yet have a wide variety of products and a large client base.

The purpose and objectives of the work.

The purpose of the work is to create a website for a pharmaceutical company.

Its achievement involves solving the following tasks:

Analysis of domestic and foreign sources. Data collection and preparation. Database development. Website creation.

Methods used. The JavaScript programming language was used to create the website, the React framework for the client part. Node.js and Express frameworks for the server part. The database was implemented with MySQL technology.

Results obtained. A website was created that allows a pharmaceutical company to talk about itself and show its products. It is possible to add new products and register users. Data about products and users is stored in the database.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Рис. – рисунок.

БД – база даних.

СУБД – система управління базами даних.

Front-end (передній кінець) – описує те, що відбувається на боку клієнта, тобто у браузері користувача.

Back-end (задній кінець) – описує те, що відбувається на сервері, який обробляє запити від клієнта.

IT (Інформаційні технології) – це галузь знань і практик, пов'язаних з обробкою, передачею, зберіганням, захистом та використанням інформації за допомогою комп'ютерних систем і технологій.

CSS (Cascading Style Sheets) – це мова стилів, яка використовується для опису зовнішнього вигляду веб-сторінок

ВСТУП

Я вирішив обрати тему «Розробка сайту для будівельного інтернет-магазину» для магістерської роботи, опираючись на декілька факторів:

1. Зростання популярності онлайн-торгівлі у сфері будівництва. Виходячи з нинішньої ситуації в країні в майбутньому, попит на будівельні матеріали та інструменти буде зростати. Фірми які не матимуть своєї онлайн-присутності, будуть втрачати своїх клієнтів.
2. Можливість розширення аудиторії. Сайт будівельного магазину дозволить йому залучати нових клієнтів та розширювати свою аудиторію за рахунок зручного та доступного інтернет-магазину та інших онлайн-сервісів.
3. Підвищення рівня обслуговування. Сайт, розроблений спеціально для будівельного магазину, може надати його клієнтам додаткові можливості, такі як онлайн-каталог продукції, інформацію про знижки та акції, а також можливість зв'язатися зі спеціалістами для консультації та отримання допомоги.
4. Можливість покращення бренд-іміджу магазину. Сучасний та зручний сайт може підвищити рівень довіри клієнтів до будівельного магазину та зміцнити його позиції на ринку.

Таким чином, розробка сайту для будівельного магазину має не лише практичний, а й стратегічний зміст для магазину. Ця тема дозволить мені повною мірою проявити свої знання та навички у галузі веб-розробки.

Мета і завдання роботи.

Метою роботи є створення сайту будівельного магазину.

Її досягнення передбачає вирішення наступних *завдань*:

1. Аналіз вітчизняних та зарубіжних джерел.
2. Збір та підготовка даних.
3. Розробка бази даних.

4. Створення сайту.

Використані методи. Для створення сайту було використану мову програмування JavaScript, клієнтської частини – фреймворк React. Для серверної частини фреймворки Node.js та Express. База даних була реалізована з технологією MySQL.

Отримані результати. Створено сайт, який надає змогу будівельному магазину розказати про себе та показати свою продукцію. Є можливість додавання нової продукції та регістрація користувачів. Дані про продукцію та користувачів зберігаються в базі даних.

Публікації. За результатами виконаної роботи публікації не передбачаються.

РОЗДІЛ 1
ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 Постановка задачі

Завдання дипломної роботи "Розробка сайту для будівельного магазину" можна сформулювати таким чином:

1. Дослідити вимоги та потреби клієнтів будівельного магазину, включаючи інтереси та очікування клієнтів від сайту.
2. Визначити функціональні та нефункціональні вимоги до сайту будівельного магазину.
3. Розробити дизайн сайту з урахуванням потреб та інтересів клієнтів.
4. Створити базу даних для зберігання інформації про продукцію будівельного магазину та інші дані, необхідні для роботи сайту.
5. Розробити функціональність сайту, включаючи онлайн-каталог продукції.
6. Розробити систему управління контентом для оновлення інформації про продукти та послуги мазину на сайті.
7. Протестувати функціональність сайту, переконатися у його працездатності та відповідності вимогам замовника.

Метою магістерської роботи буде розробка повнофункціонального сайту, який задовольнить потреби клієнтів будівельного магазину та допоможе йому ефективно просувати свою продукцію та послуги в Інтернеті.

1.2 Аналіз існуючих у світі способів вирішення задачі

Що таке веб-розробка і які етапи роботи вона передбачає?

Web-розробка – це певна процедура в IT-сфері, яка спрямована на створення сайтів та додатків. До неї включені такі етапи:

- верстка сторінок сервісу;
- програмування на сервері клієнта чи сторонньої версії;
- розробка конфігурації.

Це лише загальні поняття, у тому числі можна назвати більш поглиблену модель роботи. Вона складатиметься з низки етапів, які застосовуються всіма фахівцями з розробки сайтів:

- проектування самого сайту та складання технічного завдання;
- підбір концепції з урахуванням вимог замовника та тематики;
- розробка дизайну;
- створення попереднього макету сторінок;
- виробництво елементів мультимедіа;
- верстка;
- розробка та впровадження програмного забезпечення та інтеграція інструментів у існуючу систему управління;
- розміщення та оптимізація сторінки під її призначення;
- додавання текстового та графічного наповнення;
- проведення тестування та виправлення помилок при необхідності;
- запуск проекту на громадських платформах в Інтернеті;
- проведення подальших робіт з наповнення та підтримки сайту у робочому стані.

Виходячи з цих даних, можна зробити висновки, що всі ці роботи виконуються розробниками не обов'язково. Вони можуть змінюватися, об'єднуватися в одну групу або взагалі не використовуватися. Все залежить від того, яке саме завдання постає перед програмістом. Також усі ці роботи неможливі без використання спеціальних методик та трендів в оформленні сторінки [1].

Розробка сайту і бізнес

Розробка сайту у 2023 році є важливою складовою розвитку бізнесу та інтернет-присутності будь-якої компанії. У світі, де цифрові технології

займають все більше місця, створення професійного та привабливого веб-сайту є одним з найважливіших кроків для підвищення конкурентоспроможності та залучення нових клієнтів [2].

Основні тенденції у розробці сайтів в 2023 році:

1. Мобільна адаптивність

   У світі, де понад 50% всіх відвідувань веб-сайтів відбувається з мобільних пристроїв, мобільна адаптивність є важливою складовою успіху будь-якого сайту. В 2023 році розробники сайтів продовжать ставити на мобільну адаптивність, що дозволить забезпечити користувачам зручний та доступний дизайн незалежно від типу пристрою, який вони використовують [2].

2. Швидкість роботи сайту

   Швидкість роботи сайту відіграє ключову роль у задоволенні потреб користувачів. В 2023 році розробники сайтів будуть ставити на підвищення швидкості роботи сайту та зменшення часу завантаження сторінок. Це можна досягти за допомогою оптимізації зображень, скриптів та інших ресурсів, а також використанням CDN-систем [2].

3. Інтерактивний дизайн

   Інтерактивний дизайн, такий як анімація та мікроінтеракції, стає все популярнішим у світі розробки веб-сайтів. В 2023 році дизайнери та розробники сайтів будуть ставити на збільшення взаємодії користувачів з сайтом, забезпечуючи зручний та цікавий інтерфейс [2].

4. Персоналізований контент

   Персоналізований контент – це зміст, який адаптується до інтересів та потреб конкретного користувача. У 2023 році персоналізація контенту стане все популярнішою у світі розробки веб-сайтів. Це дозволить забезпечити кожному користувачеві

унікальний та цікавий контент, що збільшить час перебування на сайті та підвищить конверсію [2].

5. Використання штучного інтелекту
   Штучний інтелект (AI) вже використовується у світі веб-розробки, проте в 2023 році він стане ще більш популярним. AI може бути використаний для аналізу даних користувачів, персоналізації контенту, покращення SEO-стратегії та багатьох інших завдань [2].

Що являє собою сайт та яку цінність він несе як інструмент продажу.

Веб-сайт компанії є аналогом офісу або штаб-квартири тільки онлайн. При відкритті реального офісу ви берете в оренду приміщення, облаштовуєте його, проводите комунікації, купуєте обладнання та меблі, наймаєте спеціалістів, запускаєте маркетингову кампанію, замовляєте вивіску тощо. У режимі онлайн все дуже схоже.

Оренда приміщення та його ремонт - це домен, хостинг та рішення з розробки або техпідтримки. Обладнання та офісні меблі – це інтерфейс сайту, рекламна кампанія – це seo/контекстна реклама/таргет, а логотип займає місце вивіски.

Якщо у вас стартап або невеликий місцевий бізнес, сайт за $100-500$ може вирішити ваші завдання. Але якщо у вас регіональна компанія, маркетплейс, або міжнародна організація, і ви націлені на серйозне позиціонування себе в просторі діджитал — будьте готові інвестувати у свій сайт і час і кошти.

Технології розробки

Існує три найбільш популярні рішення для створення сайту:

- Програмне забезпечення, яке пропонується як послуга, відоме як SaaS-платформи (software as a service);
- розробка на CMS (англ. content management system - система управління сайтом) (можуть бути коробкові або кастомні);
- розробка на Framework.

1. **SaaS** – це спосіб розповсюдження програмного забезпечення на орендній основі. Такі системи орендуються за передплатою. Достатньо лише зареєструватися на сервісі, який надає відповідне ПЗ та заповнити шаблон відповідною інформацією. Цей варіант розробки підходить для запуску нових нескладних проектів або напрямків вже в бізнесі, що діє, і тестування бізнес-гіпотез (банально: «злетить або не злетить»).

Найпопулярнішими рішеннями SaaS є UMI, WIX, Shopify, uCoz.

Плюси створення сайту на SaaS-платформах:

- На цих платформах можна створити готове рішення всього за кілька днів, завдяки швидкості створення;
- Вартість розробки обходиться набагато дешевше за розробку на CMS або Framework, достатньо вибрати потрібний пакет і оплатити передплату;
- У рішення, наданих постачальником SaaS, вже включено все необхідне для повноцінної роботи сайту, таке як панель управління, хостинг та сервер. Постачальник SaaS бере на себе всю цю роботу.

Мінуси:

- На таких платформах надається лише шаблонний дизайн і не завжди навіть є можливість поміняти місцями блоки
- Налаштування проекту для впровадження унікальних можливостей або внесення доопрацювань у функціонал не є можливим, можна користуватися лише тим, що надано системою;
- Більшість таких систем не оптимізовано для високого навантаження на сайт, у них невисокі показники швидкодії, відмовостійкості та продуктивності.

2. **CMS** - це програмне забезпечення, у вигляді готового каркасу сайту, який можна редагувати та наповнювати контентом без спеціальних знань у програмуванні.

Найбільш популярними умовно безкоштовними CMS є WordPress, Joomla, OpenCart, Drupal. Такі системи управління можна освоїти навіть за мінімальних навичок програмування. Завантажити та встановити таку систему можна за пару хвилин.

Але, розробляючи сайт на таких CMS, потрібно брати до уваги і **головні мінуси:**

- Для безкоштовних CMS найголовніша проблема та ризик – це безпека. Освоїти ці CMS може кожен і майже кожен, хто володіє знаннями в програмуванні, може її зламати, що несе за собою непоправну шкоду компанії.
- Відсутність технічної підтримки та документації, пошук необхідної інформації перетворюється на цілу проблему, що ускладнює роботу та обслуговування сайту на такому движку.
- Доступи до окремих функцій та модулів відбуваються на платній основі.
- Коробочні CMS - це платні "движки" для сайту. Вони так само мають вже вбудований функціонал і набір необхідних модулів, які можна освоїти за короткий час. Але на відміну від безкоштовних CMS, вони більш надійні та безпечні щодо злому. За ними є доступна та безкоштовна технічна підтримка та документація. До коробкових CMS відносяться 1С-Бітрікс, NetCat, UMI.CMS, HostCMS.

Плюси розробки на CMS:

- Основні функції вже передбачені обраною CMS
- Для стандартного сайту легко передбачити результат
- Наявна структура сайту дозволяє зекономити час та бюджет на процесі розробки.

Що стосується мінусів:

- Обмежений функціонал у рамках CMS. Створення додаткових функцій, які не передбачені, тягне за собою додаткові витрати

- Виключення невикористовуваного функціоналу, який може ускладнювати та затяжувати роботу сайту, не є можливим;
- Відсутність унікальності. Кожен може купити такий самий каркас сайту.

Кастомні CMS (custom (пер. з англ.) – зроблений на замовлення; клієнт) – це варіант для масштабування, коли бізнес не підлаштовується під рамки «каркаса движка» і коли необхідно впроваджувати «нетипові» рішення. Кастомні CMS полегшують адміністрування та налаштування сайтів, створених на framework. До них належать October CMS, Asgard CMS, PyroCMS.

Такі CMS дозволяють вибирати набір функціоналу та допилювати його на льоту. Кастомні рішення можуть підійти як складним проектам (великі сервіси, соціальні мережі, проекти з безліччю інтеграцій тощо), так і більш простим, але з нетиповими завданнями (партнерські програми, системи управління дочірніми сайтами)

Найчастіше буває, що складні проекти зароджуються лише на рівні коробочної розробки, але з розвитком виростають із них.

3. **Framework** - це набір бібліотек, що полегшує розробку та об'єднання різних модулів програмного проекту. Такий варіант чудово підійде компаніям, які впроваджують нестандартні торгові пропозиції та реалізовують індивідуальні системи ведення бізнесу. Наприклад, створення власної LMS (learning management system — система управління навчанням) у рамках проекту або розробка дочірніх сайтів, які керуватимуться з однієї системи управління.

Плюси розробки сайтів на framework:

- Відсутність обмежень дозволяє реалізувати та впровадити всі задуми та доопрацювання без будь-яких обмежень;
- Дозволяє додати унікальний функціонал під час розробки
- З використанням фреймворків можна розширювати функціонал та масштабувати будь-який проект з мінімальними ризиками;

- Сайти, розроблені з використанням таких платформ, мають значно вищу швидкість завантаження та оптимізації порівняно з тими, що створені на CMS;
- Є можливість повністю виключити функціонал, що не використовується.

З умовних мінусів розробки сайтів на framework:

- Розробка на цих платформах може зайняти трохи більше часу, ніж розробка аналогічних сайтів на готових CMS. Це обумовлено тим, що технологія передбачає створення клієнтської та адміністративної частин сайту з нуля.
- Розробка сайту на фреймворку є складнішою, тому вимагає залучення висококваліфікованих програмістів. Однак, результат, отриманий з використанням цих фреймворків, значно перевершує попередні версії, що робить витратили на нього зусилля вартими.

Чим динамічніше розвивається ваш бізнес, тим сильніше еволюціонує ваш сайт. Потреба йти в ногу з часом диктує і постійний розвиток технологій: сьогодні ваш сайт зручний та швидкий, але вже завтра може статися новий прорив у IT і знову куди прагнутиме. Спираючись на наш досвід, можемо сказати, що більшість великих проектів щонайменше раз на 3-5 років оновлюють функціонал та дизайн свого сайту, лідери галузей роблять це ще частіше. Не бійтеся розвивати сайт своєї компанії, щоб зберегти конкурентоспроможність сайту. Тоді сайт буде найефективнішим інструментом розвитку вашого бізнесу [3].

Front- та Back-end технології

Front-end (інтерфейсні) технології

Інтерфейсні технології призначені для «клієнтської сторони» вашого веб-сайту або програми. Вони використовуються для розробки інтерактивних компонентів вашого сайту та створення елементів, які бачать користувачі і з якими взаємодіють. Сюди входять кольори та стилі тексту, зображення, кнопки та меню навігації [4].

Back-end (серверні) технології

Внутрішні технології призначені для "серверної частини" вашого сайту або програми. Вони зберігають та впорядковують дані та стежать за тим, щоб на інтерфейсі все працювало. Наприклад, коли користувач надає облікові дані для входу до програми соціальної мережі, використовуються внутрішні технології для перевірки правильності цих облікових даних. Після перевірки облікових даних сервер відправить назад ім'я профілю, зображення та іншу пов'язану інформацію [4].

Back-end технології також використовуються для оптимізації основних бізнес-процесів. У випадках, коли у вас є багато даних, які потрібно обробити, ви можете запустити скрипт у серверній частині, щоб створити змістовний звіт у зовнішній частині [4].

Ви також можете надсилати автоматичні листи до груп користувачів. Електронні листи можуть бути надіслані у певні дати, наприклад, після закінчення терміну дії безкоштовної пробної версії веб-сайту користувача [4].

Frontend

React та мультиплатформні рішення

Були часи, коли Vue, Angular та React йшли «ніздрю ніздрю» за затребуваністю. Популярність Vue, крім всіх його переваг, багато в чому забезпечена підтримкою майже «з коробки» впровадження на Laravel та 1С-Бітрікс. Популярність React зростає випереджаючими темпами за рахунок активної спільноти, можливості використання React Native та React для десктопних та мобільних додатків та зростання популярності javascript загалом [5].

У Kotlin Multiplatform (KMM) у 2022 році вийшла очікувана фахівцями beta версія, в якій велику роботу було зроблено з реалізації нової моделі управління пам'яттю в Kotlin Native, що зробило тепер крос-платформну розробку під iOS простою та надійною, як на Android. Використання одного коду на всіх мобільних платформах є одним із основних варіантів використання Kotlin Multiplatform. За допомогою Kotlin Multiplatform Mobile

ви можете створювати мультиплатформні мобільні програми, що спільно використовують один код і в Android, і в iOS, наприклад бізнес-логіку, можливості підключення і багато іншого [5].

Прогресивні веб-програми (PWA)

Прогресивні веб-програми – це щось між мобільними програмами та мобільними веб-сайтами. PWA поводиться так само як мобільний додаток, даючи користувачам той же інтерфейс, але працюючи через браузер [6].

Це не нова технологія у світі веб-розробки. Але в умовах, коли деякі російські програми стали недоступні в сторах, технологія знову стала актуальною [6].

Крім того, що PWA допомагають компаніям обходити перешкоди, у технології є ще кілька додаткових плюсів:

- Низька вартість розробки та обслуговування.
- Швидка швидкість завантаження та «легка вага».
- Робота в режимі офлайн.
- Можна встановити ярлик на екран, який не відрізняється від іконки програми.

PWA — може стати чудовим інструментом для залучення додаткового трафіку та великої кількості замовлень [6].

Javascript та Python

Веб-розробка неможлива без мови програмування, тому я хочу виділити два головні тренди останніх років, які не залишають топ. Javascript багато хто визнає найкращим за всіма показниками. Це підкріплюється статистикою Github, Pypl-index та Stackoverflow. У спину йому дихає back-end мову програмування – Python [7].

Їхня популярність обумовлена такими перевагами:

- найвища продуктивність;
- легка інтеграція наборів інструментів;
- висока безпека кінцевих продуктів.

Серед недоліків, як правило, виділяють лише складність у вивченні. Хоча для Python властива низька ефективність розробки мобільних додатків [7].

Якщо Ви бажаєте працювати веб-розробником у великій корпорації, дві ці мови обов'язкові для вивчення. Статистика показує, що незважаючи на високу конкуренцію, Javascript і Python займатимуть вершину рейтингу, а отже, відповідні фахівці будуть затребуваними [7].

AMP сторінки

AMP (Accelerated Mobile Pages) – технологія прискорених мобільних сторінок з відкритим вихідним кодом. Технологію було анонсовано компанією Google відносно нещодавно, у жовтні 2015 року. Ця технологія лише набирає свої оберти. Завдяки їй ваш сайт на будь-яких мобільних пристроях відкриватиметься практично миттєво. Google активно просуває цю технологію та заохочує власників сайтів, які впровадили її, вищими позиціями в органічному пошуку [8].

Дана технологія має відкритий вихідний код, безліч мануалів та інструкцій різними мовами світу, а також абсолютно безкоштовна. Використовуйте цю технологію та радуйте своїх користувачів [8].

Безсерверна архітектура (Serverless Architecture (SA))

Триває пошук технології, яка допомагає зменшити перевантаження системи, втрати даних і витрати на розробку. Введіть: безсерверна технологія.

Безсерверна архітектура працює на основі хмарної технології, яка дозволяє користувачам запускати код практично для будь-якого типу додатків або серверних служб без адміністрування. Не потрібно надавати, керувати чи оновлювати сервери. Amazon, Google і Microsoft є провідними прикладами компаній, які використовують і пропонують безсерверну архітектуру [9].

Станом на 2022 рік ринок безсерверної архітектури оцінюється в 36,84 мільярда доларів. Це означає, що ми можемо очікувати, що більше компаній вийдуть на цей ринок у 2023 році та пізніше [9].

Ми також побачимо більше продуктів, зокрема чат-ботів, програми IoT та API, які використовують безсерверні функції для виконання таких завдань, як завантаження резервних копій файлів, доставка сповіщень та експорт об’єктів [9].

Персоналізація контенту за допомогою машинного навчання

Машинний інтелект, включаючи технології машинного навчання, має великий вплив на нашу повсякденну діяльність в Інтернеті, навіть без нашої помітної уваги. Це головна суть ML — надання покращеного досвіду нативно.

Машинне навчання – це здатність програмного забезпечення покращувати свою продуктивність без прямого втручання розробників.. По суті, програмне забезпечення аналізує вхідні дані, виявляє закономірності, приймає рішення та покращує свою роботу.

Airbnb, наприклад, використовував машинне навчання, щоб налаштувати результати пошуку для гостей, щоб підвищити шанси господаря прийняти їхній запит. Алгоритм машинного навчання аналізує рішення щодо прийняття запиту кожного хоста. Залежно від цього результати пошуку в списках, які з більшою ймовірністю будуть прийняті, мають вищий рейтинг. A/B-тестування показало збільшення конверсії на 3,75%. У результаті тепер усі користувачі Airbnb обробляються відповідно до цього алгоритму, що покращує задоволеність клієнтів і збільшує дохід.

Інженери Netflix пішли навіть далі. Щоб застосувати інтелектуальний підхід до персоналізації вмісту, вони використовують більш просунуті алгоритми на основі ML, щоб краще задовольняти потреби користувачів. На відміну від орієнтації на цілий сегмент користувачів, кожен із користувачів ідентифікується окремо. Алгоритми надають вміст і результати пошуку на основі намірів користувачів замість попередніх запитів.

Чудові приклади, але є набагато більше! Увімкнення природної мови та розпізнавання зображень може покращити взаємодію з користувачем. Машинне сприйняття дозволяє комп’ютеру інтерпретувати дані та приймати обґрунтовані рішення. Машинне навчання використовується у веб-додатках у

різних галузях, таких як охорона здоров'я, фінанси, освіта, сільське господарство тощо. Ця технологія пропонує значні покращення, яких було б важко досягти без ШІ [10].

Інтернет речей (Internet of Things (IoT))

Інтернет змінив освіту, менеджмент, охорону здоров'я і навіть те, як ми спілкуємося з друзями.

Посилення впливу Інтернету на наше життя дозволило започаткувати індустрію під назвою Інтернет речей (IoT). Що це означає? Насправді це означає підключення до Інтернету домашніх пристроїв, побутової техніки чи носіїв. IoT робить багато ваших пристроїв доступними через ваш телефон [11].

Чому IoT зарекомендував себе як один із найпотужніших трендів веб-розробки? Справа в тому, що підключені до IoT пристрої створюють постійну передачу даних. Це дозволяє компаніям пропонувати свої послуги для взаємодії з користувачами якомога швидше, створюючи персоналізований досвід [11].

Одним із найвідоміших прикладів пристроїв IoT є розумні колонки Google Nest. Це інтелектуальний пристрій IoT, який дозволяє користувачам насолоджуватися такими функціями, як медіа, будильники, освітлення та інші функції лише голосом [11].

Чи має це сенс для домену веб-розробки? Звичайно, IoT створить розширений зв'язок між макетами веб-сайтів і робочими моделями. Тенденція розробки веб-сайтів має загальну інтенсивність таких речей, як датчики, камери та сигнальне обладнання. Це дозволить більш ефективно вирішувати запити клієнтів і пропонувати найбільш релевантні відповіді [11].

Якщо ви хочете переконатися, що IoT є однією з найбільш швидко розвиваються технологій у веб-розробці, погляньте на діаграму Statista. Він показує, що до 2030 року кількість пристроїв, підключених до Інтернету речей, досягне понад 29 мільярдів [11].

Мобільний оптимізований веб-сайт

Згідно зі статистикою мобільного маркетингу Statista, на сьогоднішній день у світі налічується близько 6 мільярдів активних абонентів смартфонів. Підприємствам потрібен веб-сайт, зручний для мобільних пристроїв, якщо вони хочуть залучити мобільних клієнтів, оскільки Google надає пріоритет веб-сайтам, зручним для мобільних пристроїв, у своєму алгоритмі пошуку.

Ці компанії надають послуги з розробки мобільних додатків, які виходять за рамки очікувань їхніх клієнтів, використовуючи найсучасніші технічні інфраструктури. Ця технологія мобільної веб-розробки тут, щоб залишатися в мобільній екосистемі, що швидко змінюється, і все більше користувачів переходять на невеликі дисплеї для різних цілей.[12]

Гіперавтоматизація та потужність ШІ

Зараз штучний інтелект (AI) і машинне навчання (ML) лежать в основі майже будь-якої цифрової трансформації чи технологічних інновацій, тому 2020 рік також стане роком їх помітного зростання у веб-розробці.

Точність, швидкість і надійність, які забезпечує штучний інтелект, допомагають автоматизувати кожне завдання. Автоматизація процесів є трендом у веб-розробці, як і в усьому світі технологій. Це може полегшити веб-програмування, усуваючи необхідність писати повторюваний код, але також може зробити розробку довшою. З одного боку, алгоритми ШІ можуть допомогти швидше знайти правильні рішення. З іншого боку, є ціла проблема складності. Складність – це те, де вам потрібен досвідчений розробник, щоб зробити ваш сайт надійнішим. Аспект автоматизації також залежить від того, яким є ваш сайт. Якщо ваш сайт переважно текстовий, вам знадобиться автоматичний редактор для створення коду, який вам потрібно буде написати. Якщо ваш сайт трохи динамічніший або у вас є мобільний сайт, то автоматизований редактор ще важливіший. Якщо у вас є кілька мов або ви створюєте мобільний додаток, вам знадобиться окремий редактор для кожної.

Автоматизація — це великий бізнес у світі веб-розробки, тому не думайте про це як про загрозу якості ваших веб-програм або послуг. Це також

не означає, що ви повинні кинути своє ремесло. Автоматизація чудова, тому що вона означає, що ви можете створювати більш складні та надійні веб-додатки та служби без потреби інженера для програмування.

ШІ може автоматизувати не лише процес створення веб-сайту чи веб-додатку, а й увесь бізнес. Підсумовуючи сказане, давайте визначимо, як ШІ може доповнити ваші зусилля з веб-розробки:

- Автоматизований збір даних за допомогою чат-ботів на основі штучного інтелекту
- Покращена взаємодія з користувачем за допомогою інтелектуальних чат-ботів
- Точний аналіз поведінки клієнтів
- Веб-кодування замінено алгоритмами самонавчання
- Розширене тестування та контроль якості програмного забезпечення
- Автоматизація електронного маркетингу
- Налаштування результатів пошуку під користувачів[13]

1.3 Визначення вхідних даних для вирішення поставленої задачі

Потрібно розробити базу даних для супроводу сайту будівельної компанії.

База даних для супроводу даних про актуальні оголошення про продукцію, що виконує функцію швидкого доступу до моніторингу оголошень та цін. Головна ціль інформаційної системи: зручний доступ до актуальної інформації про продукцію та її придбання.

Висновки до розділу 1

Сучасна розробка веб-сайтів вимагає знань і вмінь у використанні новітніх технологій та інструментів, враховуючи найновіші тенденції у дизайні та функціональності. Крім того, на сьогоднішній день, ключовими аспектами у розробці сайтів є аналітика та управління даними, безпека та захист персональних даних, адаптивний дизайн та мобільність.

Враховуючи все вищезазначене, дослідження зазначеної предметної області є необхідним для підготовки компетентного веб-розробника в 2024 році. Вона дозволяє отримати уявлення про новітні технології та інструменти, що використовуються в розробці веб-сайтів, ознайомитися зі сучасними тенденціями дизайну та функціональності, а також вивчити принципи безпеки та захисту персональних даних.

Отже, дослідження предметної області є ключовим етапом, який дозволяє отримати необхідні знання та навички для розробки сучасних, безпечних та функціональних веб-сайтів, відповідно до вимог та очікувань користувачів у 2024 році.

РОЗДІЛ 2
ТЕОРЕТИЧНА ЧАСТИНА

2.1 Розробка бази даних

Список назв прецедентів використання:

1. Реєстрація користувача у системі.
2. Переглядання оголошень про продукцію.
3. Сортування оголошень за вибраними користувачем фільтрами.
4. Створення оголошення ро продукцію.

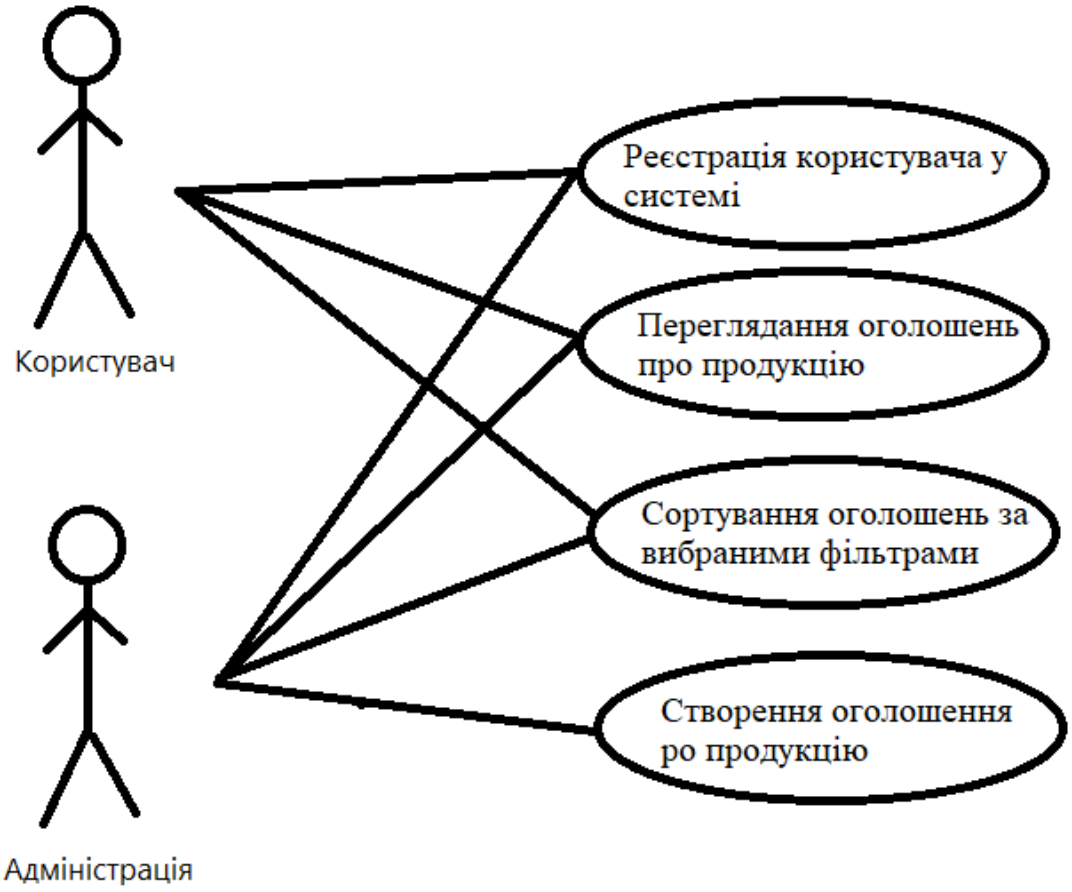


Рис. 2.1 Прецеденти використання

Опис прецедентів використання в стислому форматі:

Розглянемо деякі прецеденти використання зображені на рисунку 2.1

- Прецедент: Реєстрація користувача у системі.

    1. Користувач – людина

    Мета: Створити новий обліковий запис для взаємодії з сайтом.

    Завдання: Запросити необхідні дані, надати їх для перевірки адміністратором; створення особового облікового запису в БД.

    Новий користувач вносить необхідні дані до форми реєстрації та відправляє їх. БД отримує заповнену форму та самостійно перевіряє її, після чого створює новий обліковий запис, або відмовляє в створенні і повідомляє причину.

- Прецедент: Перегляд оголошень про продукцію.

    1. Користувач – людина

    Мета: Переглянути повний перелік продукції.

    Завдання: Відобразити коротку відомість про кожне оголошення про продаж.

    2. Допоміжний користувач – адміністратор

    Мета: Забезпечити коректне та актуальне відображення усіх оголошень.

    Завдання: підтвердити розміщення оголошень про продукцію.

3. Закулісний користувач – магазин

Мета: Отримати прибуток з продажу своєї продукції.

Завдання: Надати детальну, коректну та актуальну інформацію про продукцію.

Користувач відкриває перелік усіх доступних оголошень про продаж, може переглянути короткі відомості про кожне оголошення, або відкрити його і дізнатися більше необхідної інформації. Адміністратор перевіряє, щоб до користувача надходила лише коректна та перевірена інформація. Продавець зацікавлений в надані коректної інформації, щоб його оголошення зацікавило користувача.

- Прецедент: Сортування оголошень за вибраними користувачем фільтрами.

1. Користувач – людина.

Мета: отримати необхідні оголошення шляхом налаштування фільтрів перегляду.

Завдання: обрати параметри, за якими будуть відображатися оголошення.

2. Допоміжний користувач – адміністратор

Мета: забезпечити відображення оголошень.

Завдання: забезпечити відповідність фільтрів відносно продукції в оголошенні.

Користувач відкриває список всіх наявних оголошень та налаштовує параметри для відображення потрібних саме йому.

- Прецедент: Створити оголошення про продукцію.

1. Користувач – магазин

Мета: Опублікувати свою продукцію.

Завдання: Створити нове оголошення з описом товару та ціною.

2. Допоміжний користувач – адміністратор.

Мета: Перевірити коректність створеного оголошення.

Завдання: Підтвердити коректність заповнення всіх даних та схватили розміщення оголошення на БД.

Користувач створює нове оголошення з описом та ціною товару. Адміністратор перевіряє коректність внесених даних та схвалює публікацію оголошення.

Морфологічний аналіз текстів з виділенням кандидатів сутностей, зв’язку та атрибутів

Іменники в тексті та їх аналіз:

Людина – синонім до «користувач», має доступ лише до інтерфейсу ІС.

Обліковий запис - це набір інформації, яка використовується для ідентифікації користувача при підключенні до системи. Він містить дані для авторизації, обліку та атрибути, пов'язані з користувачем.

Адміністратор – синонім до користувача, має набір конкретних прав.

Форма реєстрації – частина інтерфейсу, куди користувач вводить дані для створення облікового запису, не моделюється.

Перелік оголошень про продукцію – список доступних користувачу оголошень, сутність.

Пропозиція – оголошення, які бачить користувач, сутність.

Продукція – будівельні матеріали, інструменти та інші товари компанії, сутність.

Продавець - це особа або організація, яка, отримуючи відповідну винагороду, передає товар чи послугу покупцеві. В широкому розумінні, продавець включає як осіб, що прямо контактують з покупцями та здійснюють продажі товарів, так і тих, хто виконує цю роль у відносинах зі замовниками.

Покупець - це фізична або юридична особа, яка здійснює оплату і є отримувачем товару або послуги. Покупець є центральною сутністю у процесі купівлі-продажу і здійснює оплату за отримання необхідних товарів або послуг.

Призначення, категорія, інструкція – атрибути до сутності «нерухомість».

Дієслова в тексті та їх аналіз:

Користувач переглядає перелік оголошень про продукцію.

Користувач вибирає фільтри відображення оголошень.

Людина заповнює форму реєстрації.

Продавець створює можливості для відображення нових пропозицій.

Перелік оголошень про продукцію містить пропозиції.

Діаграма "сутність-зв'язок" етапу концептуального проектування

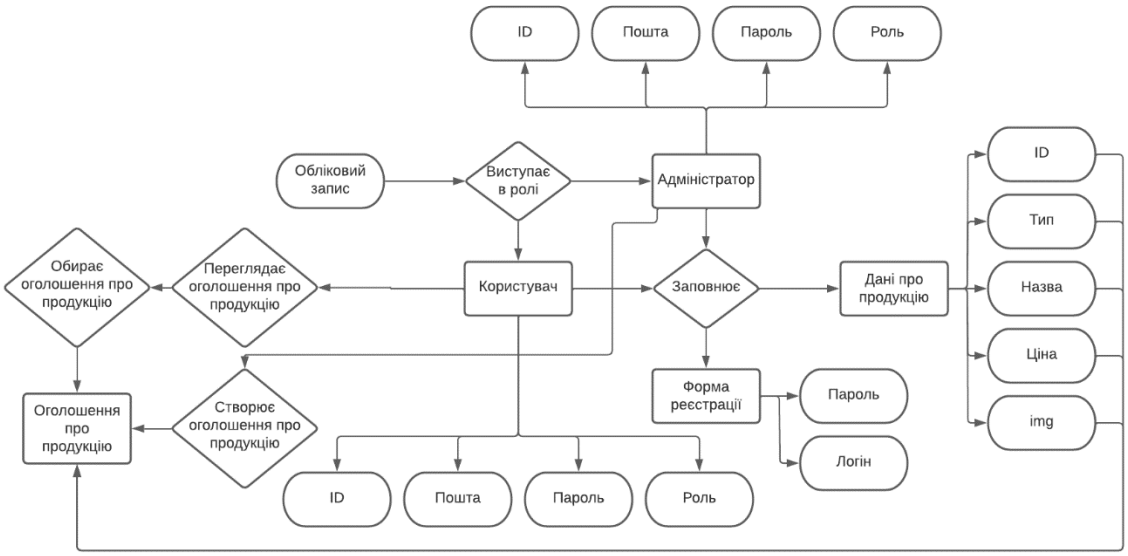


Рис. 2.2 Діаграма "сутність-зв'язок"

Діаграма класів UML етапу логічного проектування

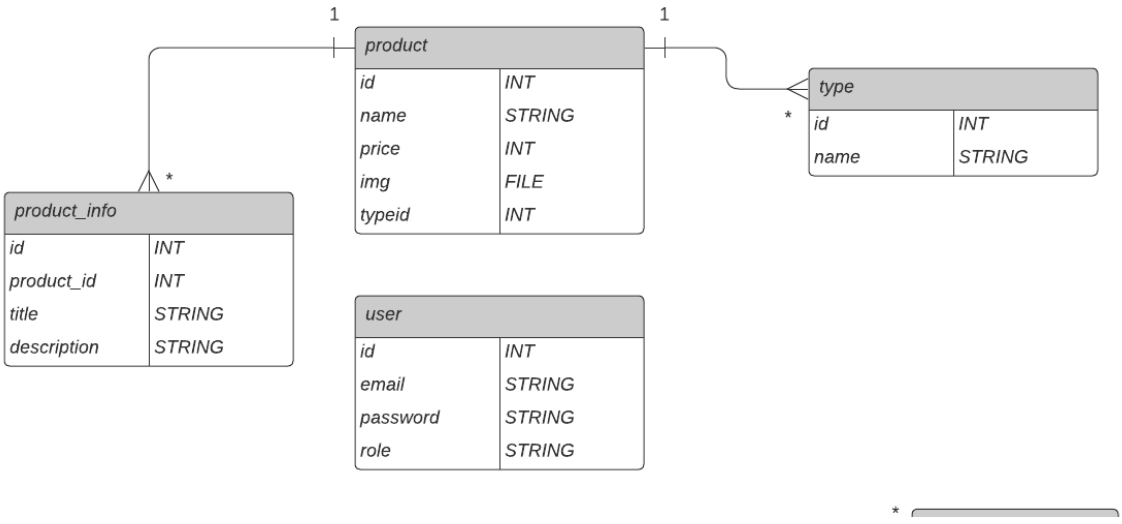


Рис. 2.3 Діаграма класів UML

2.2 Методи та засоби обробки даних

Для обробки даних у базах даних (БД) існує безліч методів та інструментів, включаючи:

1. **SQL (Structured Query Language)** - декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування (C або Pascal), SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних [14].
2. **Мови програмування** - багато мов програмування, такі як Java, Python, C# та інші, мають можливість працювати з базами даних, дозволяючи написати програми для обробки даних.

Це штучна мова, створена для передачі команд машинам, зокрема комп'ютерам. Мови програмування використовуються для створення програм, які контролюють поведінку машин, та для запису алгоритмів [15].

Суворіше визначення: мова програмування — це система позначень для опису алгоритмів і структур даних, певна штучна формальна система, засобами якої можна виражати алгоритми. Мову програмування визначає набір лексичних, синтаксичних і семантичних правил, що задають зовнішній вигляд програми та дії, які виконує виконавець (комп'ютер) під її управлінням [15].

3. **Бізнес-аналіз** – методи та інструменти, такі як OLAP-аналіз, використовуються для аналізу великих обсягів даних у режимі реального часу.

Основою концепції OLAP є ідея віртуально багатовимірного OLAP-куба (гіперкуба). Осями (вимірами) OLAP-кубу є числові або короткі лінгвістичні дані про предметну область роботи. Приклади фрагментів даних із різних сфер діяльності: поштові адреси (країна, місто, район, поштовий індекс, вулиця), регіон планети, географічні координати; системний час виконання операції чи процесу; прізвища продавців, номер картки покупця з його ідентифікаційними даними, назви товарів, код товарів, ціни товарів, кількість товарів; лінгвістичні і числові ідентифікатори лікарів і хворих, назви і коди хвороб та їх груп; назви сільгосппродуктів; назва заявки на обслуговування, атрибути оператора, час прийняття і виконання заявки, атрибути виконавця; прізвища та коди працівників силових структур, прізвища порушників, назви і коди порушень та їх групи; назви зразків озброєння і воєнної техніки, їх груп; ін. Кількість таких числових і лінгвістичних видів даних (вимірів, осей) і їх градацій визначається аналітичними потребами, які можуть потребувати від 10 до 100 і більше даних (вимірів, осей). Загальноприйнята назва "багатовимірний куб" (OLAP-куб) є умовною, адже його осі даних мають різну довжину. Для аналізу утворюють OLAP-гіперкуби, які мають як мінімум кілька осей різної координатної довжини. У великих системах вхідні дані для OLAP можуть бути попередньо узагальненими у сховищі даних (Data Warehouse), адже дані у системах реєстрації транзакцій (OLTP-системах) безперервно змінюються, для прикладу, дані у системах реєстрації продажів товарів, квитків [16].

4. **ETL-інструменти (Extract, Transform, Load)** - процес, який використовується в базах даних та, особливо, у сховищах даних та у засобах Business Intelligence для забезпечення їх роботи для підтримки прийняття рішень. ETL-процес, як концепція, набув поширення у 1970-х роках. Він охоплює наступні етапи обробки даних [17]:
    a. Виймання даних[en] із зовнішніх джерел,
    b. Перетворення даних, для зберігання даних у відповідній структурі або форматі, з метою подальшого аналізу [17].
    c. Завантаження даних у кінцеву базу даних. Більш точно, це може бути вітрина даних або сховище даних [17].

Поняття ETL може стосуватися процесу завантаження будь-якої бази даних. Оскільки виймання даних займає багато часу, то для скорочення загального часу обробки, поширеним є одночасна робота всіх трьох етапів ETL. Поки дані виймаються, процес перетворення отримує інші дані і готує їх для завантаження, щоб уникнути очікування виконання попередніх етапів [17].

Зазвичай ETL системи об'єднують дані з численних застосунків (систем), які створені та підтримуються різними вендорами та розміщені на різному апаратному забезпеченні. Розрізнені системи, які містять первісні дані, нерідко підтримуються та використовуються різними співробітниками. Для прикладу, система обліку витрат може об'єднувати дані по фонду заробітної платні, продажам та придбанням [17].

5. **NoSQL** бази даних - це тип баз даних, які використовуються для зберігання та обробки великих обсягів неструктурованих даних, таких як документи, графи та інші.

Багато NoSQL сховищ нехтують узгодженістю даних (у сенсі теореми CAP) на противагу доступності, толерантності до партиціонування, та, звісно, швидкості. Бар'єрами прийняття

парадигми NoSQL сховищ є використання низькорівневої мови запитів (замість добре розвиненого та стандартизованого SQL), брак стандартизованих інтерфейсів і значні інвестиції уже в існуючі реляційні бази. Більшість NoSQL сховищ не забезпечують використання ACID транзакцій, однак декілька баз, такі як MarkLogic, Aerospike, FairCom c-treeACE, Google Spanner (що технічно теж NewSQL база даних), Symas LMDB, та OrientDB зробили на цьому додатковий акцент [18].

6. **Великі дані** – для обробки великих обсягів даних використовуються інструменти та технології, такі як Hadoop, Spark, Cassandra та ін.

Hadoop - проект фонду Apache Software Foundation, що вільно розповсюджується набір утиліт, бібліотек та фреймворк для розробки та виконання розподілених програм, що працюють на кластерах із сотень та тисяч вузлів. Використовується для реалізації пошукових та контекстних механізмів багатьох високонавантажених веб-сайтів [19].

Spark - високопродуктивне рішення для обробки даних, що зберігаються в кластері Hadoop. У порівнянні з наданим у Hadoop механізмом MapReduce, Spark забезпечує у 100 разів більшу продуктивність при обробленні даних в пам'яті й 10 разів при розміщенні даних на дисках [20].

Cassandra – розподілена система управління базами даних, що належить до класу NoSQL-систем і розрахована створення високомасштабируемых і надійних сховищ великих масивів даних, представлених як хеша [21].

7. **Інтернет речей (IoT)** - для обробки даних, одержуваних від пристроїв IoT, використовуються спеціальні інструменти та технології, такі як Apache Kafka, Apache NiFi та ін.

8. **Хмарні БД** - багато хмарних провайдерів, таких як Amazon Web Services (AWS), Microsoft Azure та інші, надають інструменти та послуги для обробки даних у хмарних базах даних.

База даних, що, зазвичай, працює на платформі хмарних обчислень. Є дві поширені моделі розгортання: користувачі можуть запускати бази даних на хмарі незалежно, використовуючи віртуальну машину, або вони можуть отримати доступ до сервісу бази даних, що підтримується провайдером хмарних БД. З баз даних, доступних на хмарі, деякі базуються на SQL-основі, інші використовують NoSQL модель даних [22].

2.3 Обґрунтування вибору створення програмного продукту або використання хмарних технологій

У виборі між створенням програмного продукту та використанням хмарних технологій слід враховувати такі фактори:

1. Розмір бюджету - Створення власного програмного продукту може вимагати значних інвестицій у розробку, тестування та підтримку продукту. У той час як використання хмарних технологій може бути економічно вигіднішим через можливість оплати тільки за використані ресурси.
2. Рівень технічної експертизи - створення програмного продукту може вимагати наявності високої технічної експертизи в різних областях, таких як програмування, тестування, дизайн і т.д. Використання хмарних технологій може зменшити вимоги до експертизи, оскільки багато хмарних провайдерів надають готові сервіси та інструменти для розробки та розгортання програм.
3. Швидкість розробки - Використання готових сервісів та інструментів, що надаються хмарними провайдерами, може значно

прискорити процес розробки, у той час як створення власного продукту може зайняти значний час.

4. Масштабованість - У разі використання хмарних технологій ресурси можуть бути масштабовані в залежності від потреб. У разі створення власного програмного продукту необхідно заздалегідь враховувати можливості масштабування.
5. Надійність та безпека - Хмарні провайдери часто мають великий досвід та ресурси для забезпечення високого рівня надійності та безпеки додатків, включаючи механізми резервного копіювання, моніторинг та захист даних. У разі створення власного продукту необхідно забезпечити високий рівень надійності та безпеки самостійно.

Отже, вибір між створенням програмного продукту та використанням хмарних технологій залежить від низки факторів, і потребує уважного аналізу конкретної ситуації та бізнес-цілей.

Висновки до розділу 2

Одже, враховуючи все вище написане та прочитане мною з теми баз даних. Для свого сайту і його бази даних я вирішив обрати технологію SQL

Оскільки, однією з основних переваг SQL є його універсальність та широке застосування. SQL є стандартною мовою для взаємодії з реляційними базами даних, що дозволяє здійснювати різноманітні операції, такі як створення, модифікація та запити до даних. Використання SQL спрощує розробку та управління базою даних, а також забезпечує високу продуктивність обробки даних.

Іншою важливою перевагою SQL є його надійність та безпека. SQL бази даних можуть бути захищені від несанкціонованого доступу, використовуючи

різні механізми аутентифікації та авторизації. Також, SQL надає можливість забезпечувати цілісність даних за допомогою обмежень (constraints) та транзакцій.

Крім того, SQL має велику спільноту користувачів, що допомагає у вирішенні проблем та обміні досвідом. Існує багато документації, підручників, форумів та онлайн-ресурсів, які стосуються SQL. Це дозволяє розробникам швидко знайти відповіді на свої питання та ефективно вирішувати завдання, пов'язані з базами даних.

Отже, з урахуванням його універсальності, надійності та широкого сприйняття в галузі розробки баз даних, вибір технології SQL є логічним та обґрунтованим. SQL забезпечує потужні інструменти для створення та управління базами даних, що сприяє ефективному зберіганню та обробці даних.

РОЗДІЛ 3
СПОСІБ ВИРІШЕННЯ ЗАДАЧІ

3.1 Архітектура сайту

Розглянемо з чого складається архітектура сайту яка зображена на рисунку 3.1.

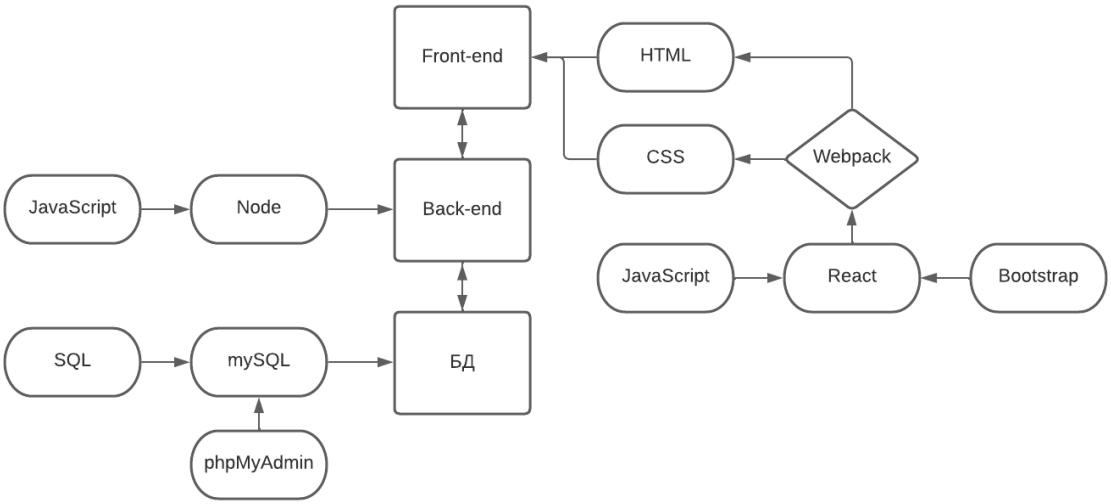


Рис. 3.1 Зображення архітектури сайту

3.1.1 Компоненти

Front-end

Front-end був реалізований на мові JavaScript. Для спрощення коду та зменшення його об’єму був використаний фреймворк React.

Фреймворк React.js — це фреймворк і бібліотека JavaScript із відкритим кодом, розроблена Facebook. Він використовується для швидкого й ефективного створення інтерактивних інтерфейсів користувача та веб-додатків із значно меншою кількістю коду, ніж із ванільним JavaScript [23].

У React ви розробляєте свої програми, створюючи повторно використовувані компоненти, які можна розглядати як незалежні блоки Lego. Ці компоненти є окремими частинами кінцевого інтерфейсу, які, будучи зібраними, утворюють весь інтерфейс користувача програми [23].

Основна роль React у додатку полягає в тому, щоб обробляти рівень перегляду цього додатка так само, як V у шаблоні «model-view-controller» (MVC), забезпечуючи найкраще та найефективніше виконання візуалізації. Замість того, щоб мати справу з усім інтерфейсом користувача як єдиним блоком, React.js заохочує розробників розділяти ці складні інтерфейси користувача на окремі багаторазові компоненти, які утворюють будівельні блоки цілого інтерфейсу користувача. При цьому фреймворк ReactJS поєднує швидкість і ефективність JavaScript з більш ефективним методом маніпулювання DOM для швидшого відтворення веб-сторінок і створення високодинамічних і адаптивних веб-додатків [23].

Webpack — це збірник модулів. Webpack може подбати про об'єднання разом із окремим засобом запуску завдань. Однак межа між комплектувальником і виконанням завдань стала розмитою завдяки плагінам webpack, розробленим спільнотою. Іноді ці плагіни використовуються для виконання завдань, які зазвичай виконуються за межами webpack, наприклад очищення каталогу збірки або розгортання збірки, хоча ви можете відкласти виконання цих завдань поза webpack [24].

Каскадні таблиці стилів (CSS) — це мова таблиць стилів, яка використовується для опису подання документа, написаного на мові розмітки, такій як HTML або XML (включно з діалектами XML, такими як SVG, MathML або XHTML). CSS є наріжною технологією Всесвітньої павутини, поряд з HTML і JavaScript [25].

CSS розроблено для того, щоб уможливити розділення вмісту та презентації, включаючи макет, кольори та шрифти. Це розділення може покращити доступність вмісту; забезпечують більшу гнучкість і контроль у специфікації характеристик презентації; дозволити кільком веб-сторінкам використовувати форматування, вказавши відповідний CSS в окремому файлі .css, що зменшує складність і повторення структурного вмісту [25].

Bootstrap — це безкоштовна інтерфейсна платформа розробки з відкритим кодом для створення веб-сайтів і веб-додатків. Розроблений для

адаптивної розробки мобільних веб-сайтів, Bootstrap надає набір синтаксису для дизайну шаблонів [26].

Як фреймворк Bootstrap містить основи адаптивної веб-розробки, тому розробникам потрібно лише вставити код у попередньо визначену сітку. Інфраструктура Bootstrap побудована на мові гіпертекстової розмітки (HTML), каскадних таблицях стилів (CSS) і JavaScript. Веб-розробники, які використовують Bootstrap, можуть створювати веб-сайти набагато швидше, не витрачаючи час на основні команди та функції [26].

Back-end

Node або Node.js - програмна платформа, заснована на рушії V8 (компілюючому JavaScript в машинний код), що перетворює JavaScript з вузькоспеціалізованої мови на мову загального призначення. Node.js додає можливість JavaScript взаємодіяти з пристроями вводу-виводу через свій API, написаний на C++, підключати інші зовнішні бібліотеки, написані різними мовами, забезпечуючи виклики до них з JavaScript-коду. Node.js застосовується переважно на сервері, виконуючи роль веб-сервера, але є можливість розробляти на Node.js та десктопні віконні програми (за допомогою NW.js, AppJS або Electron для Linux, Windows та macOS) і навіть програмувати мікроконтролери (наприклад, tessel, low.js та Espruino). В основі Node.js лежить подієво-орієнтоване та асинхронне (або реактивне) програмування з неблокуючим введенням/виводом [27].

MySQL є найпопулярнішою у світі базою даних з відкритим кодом. За даними DB-Engines, MySQL займає друге місце за популярністю після Oracle Database. MySQL підтримує багато найбільш доступних програм, включаючи Facebook, Twitter, Netflix, Uber, Airbnb, Shopify і Booking.com [28].

Оскільки MySQL є відкритим кодом, він містить численні функції, розроблені в тісній співпраці з користувачами протягом понад 25 років. Тому дуже ймовірно, що ваша улюблена програма або мова програмування підтримується базою даних MySQL [28].

phpMyAdmin — це безкоштовний програмний інструмент, написаний на PHP, призначений для адміністрування MySQL через Інтернет. phpMyAdmin підтримує широкий спектр операцій з MySQL і MariaDB. Операції, які часто використовуються (керування базами даних, таблицями, стовпцями, відношеннями, індексами, користувачами, дозволами тощо) можна виконувати через інтерфейс користувача, у той час як у вас залишається можливість безпосередньо виконувати будь-який оператор SQL [29].

Висновки до розділу 3

У результаті розгляду архітектури сайту та її компонентів було визначено ефективне рішення для розробки сучасного вебдодатку. Використання фреймворку React.js дозволяє реалізувати динамічний і взаємодіючий інтерфейс користувача з мінімізацією обсягу коду. Для побудови адаптивного дизайну було обрано Bootstrap, що забезпечує швидку розробку вебсторінок без необхідності вручну прописувати складні стилі. Застосування Webpack дозволяє ефективно керувати модулями та оптимізувати процес збірки. На бекенді вибір на користь Node.js дозволяє використовувати асинхронне програмування для досягнення високої продуктивності, а база даних MySQL забезпечує надійність та масштабованість проекту. Таким чином, вибрані технології ефективно вирішують задачу створення стабільного та високопродуктивного вебсайту.

РОЗДІЛ 4
РЕАЛІЗАЦІЯ ЗАДАЧІ

4.1 Структура

Заходячи на сайт ми потрапляємо на головну сторінку, яка зображена на рисунку 4.1.

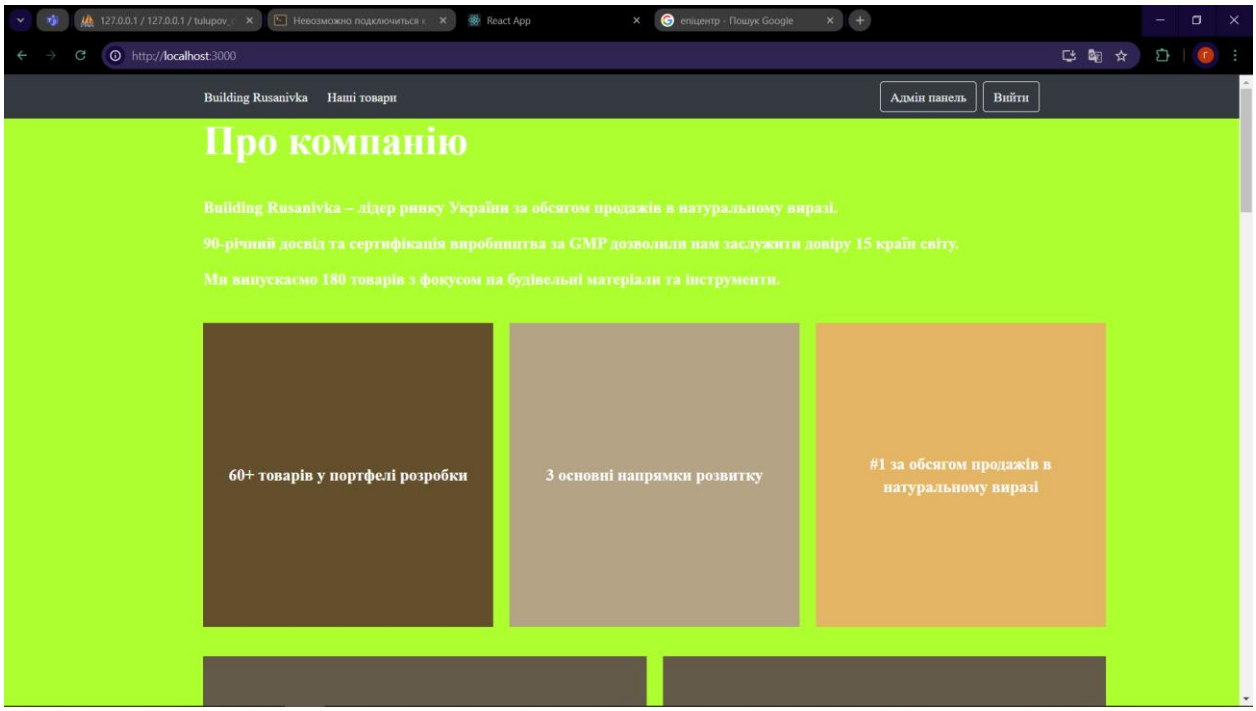


Рис. 4.1. Головна сторінка сайту

На ній надана інформація про компанію, яка може зацікавити користувачів. В хедері сайту є можливість перейти на головну сторінку натиснувши «Building Rusanivka», на сторінку товарів «Наші товари» та на сторінку авторизації або адмін сторінку.

Натиснувши на «Наші товари» ми перейдемо на сторінку товарів які пропонує компанія, сторінка зображена на рисунку 4.2.

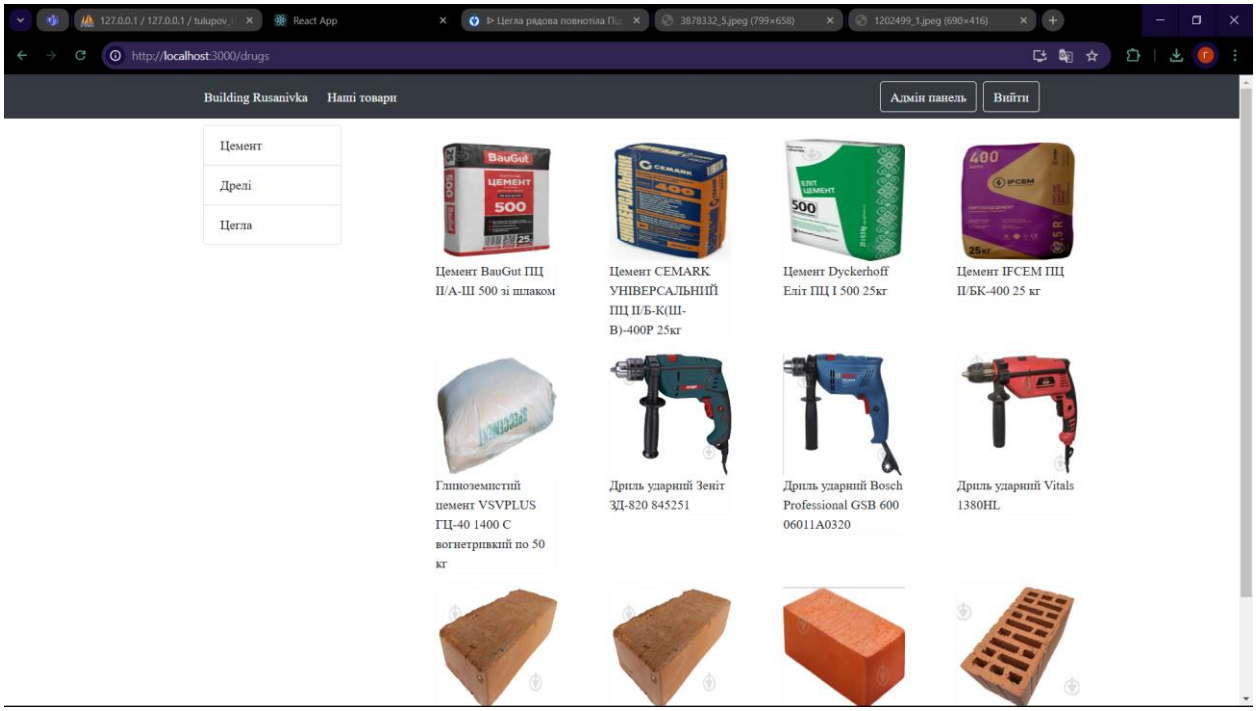


Рис. 4.2. Сторінка товарів

Зліва фільтр за типом товару. Зверху навігаційна панель в неавторизованого користувача там буде кнопка «Авторизація». По центру картки товарів, якщо натиснути на одну з них перейдемо на сторінку товару.

На рисунку 4.3 зображено сторінку товару.

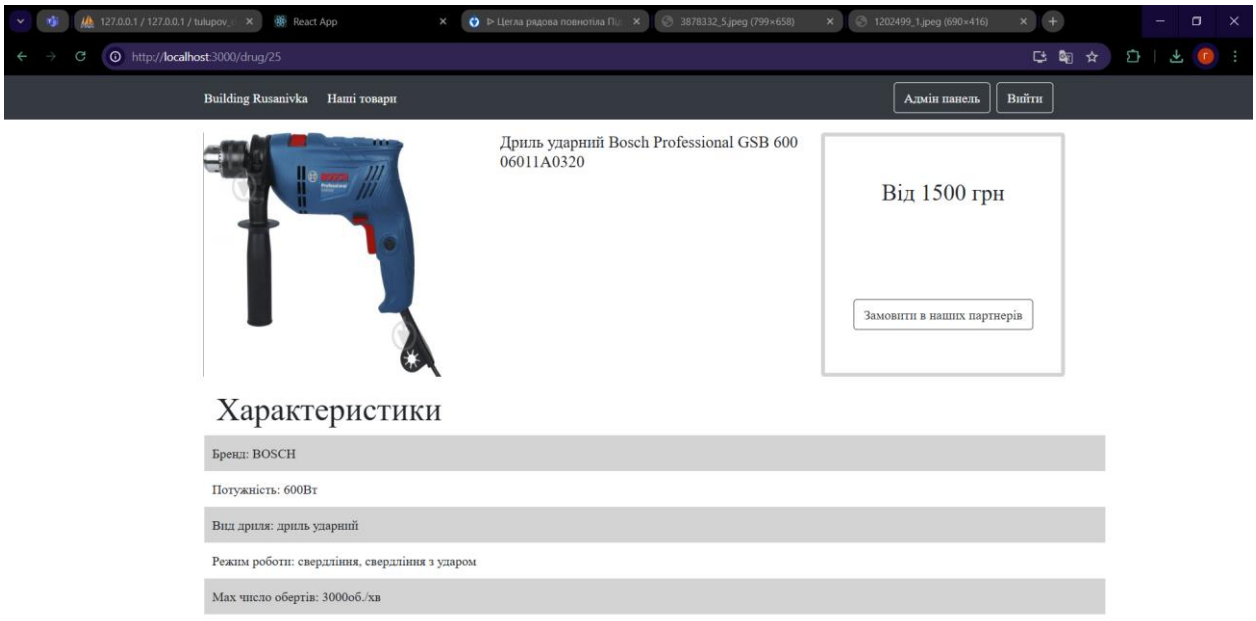


Рис. 4.3. Сторінка товару

Як видно з рисунку 4.3, в верхній половині сторінки розташована картинка товару, назва та рекомендована вартість. В нижній, характеристики товару.

Натискаючи кнопку авторизація ми переходимо на сторінку авторизації. Яка зображена на рисунку 4.4.

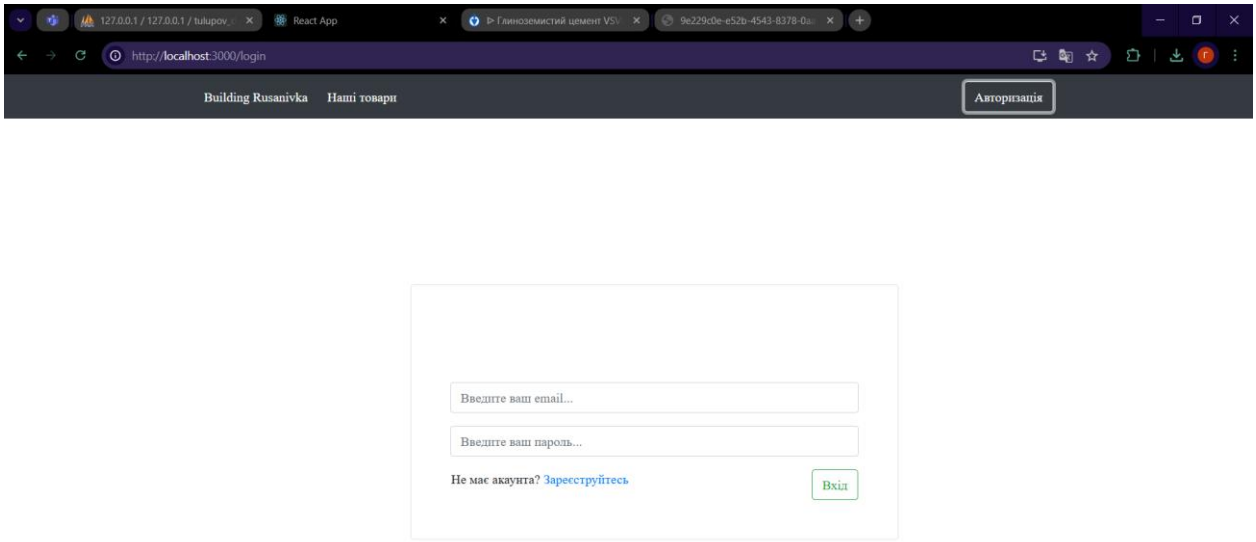


Рис. 4.4. Сторінка авторизації

Якщо натиснути «Зареєструйтесь», ми перейдемо на сторінку реєстрації. Різниця між цими двома сторінками лише в тому, що при натисканні кнопки «Вхід» або «Реєстрація». При вході до БД надсилається запит порівняння введеного логіну та паролю з тими, що вже є в БД. А при реєстрації до БД надсилається запит про створення нового користувача.

Після цього ми повертаємося на головну сторінку, де замість кнопки «Авторизація» з’являються кнопки «Адмін панель» та «Вихід».

Перейдемо на адмін панель. Яка зображена на рисунку 4.5.

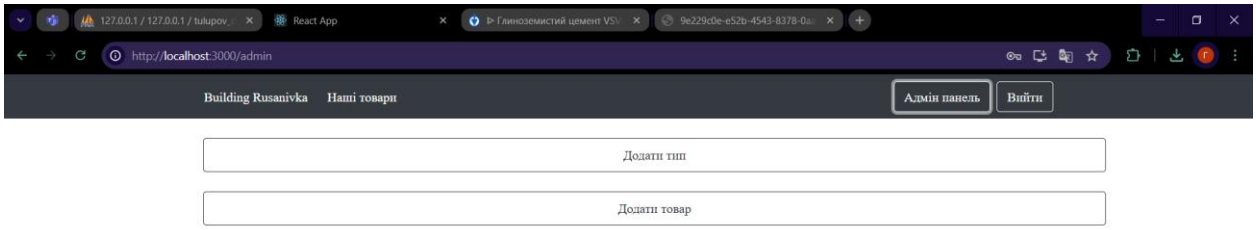


Рис. 4.5 Сторінка адмін панелі

Тут є 2 кнопки які відкривають модальні вікна для створення типу товару та карточки товару, відповідно. Додати дані можуть лише користувачі які в БД в полі «role» вказано ADMIN. За замовчуванням всі користувачі мають роль user. Змінити роль можна лише напряму в БД.

На рисунку 4.6 зображене модальне вікно додавання типу. Спробуємо додати новий тип наприклад «Бетонна суміш».

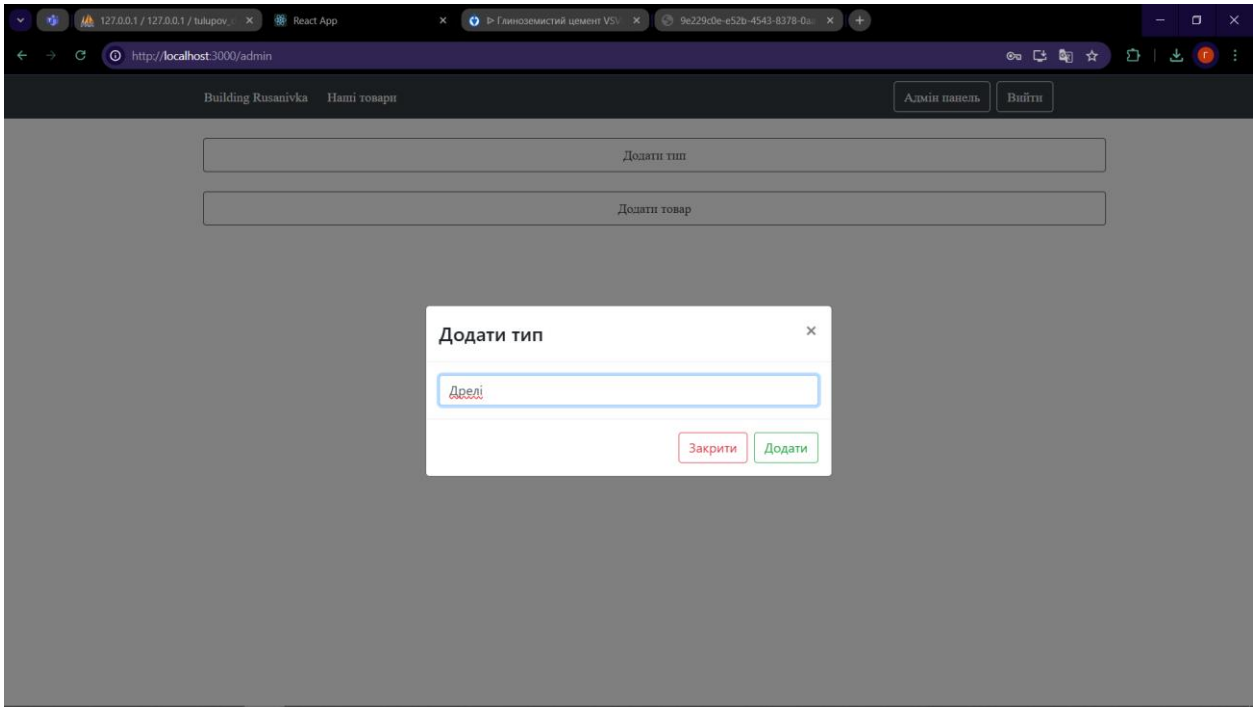


Рис. 4.6. Модальне вікно для створення типу

Натискаємо додати.

На рисунку 4.7 зображене модальне вікно додавання товару. Спробуємо додати новий товар з випадковими даними щоб перевірити чи додається він на головну сторінку.

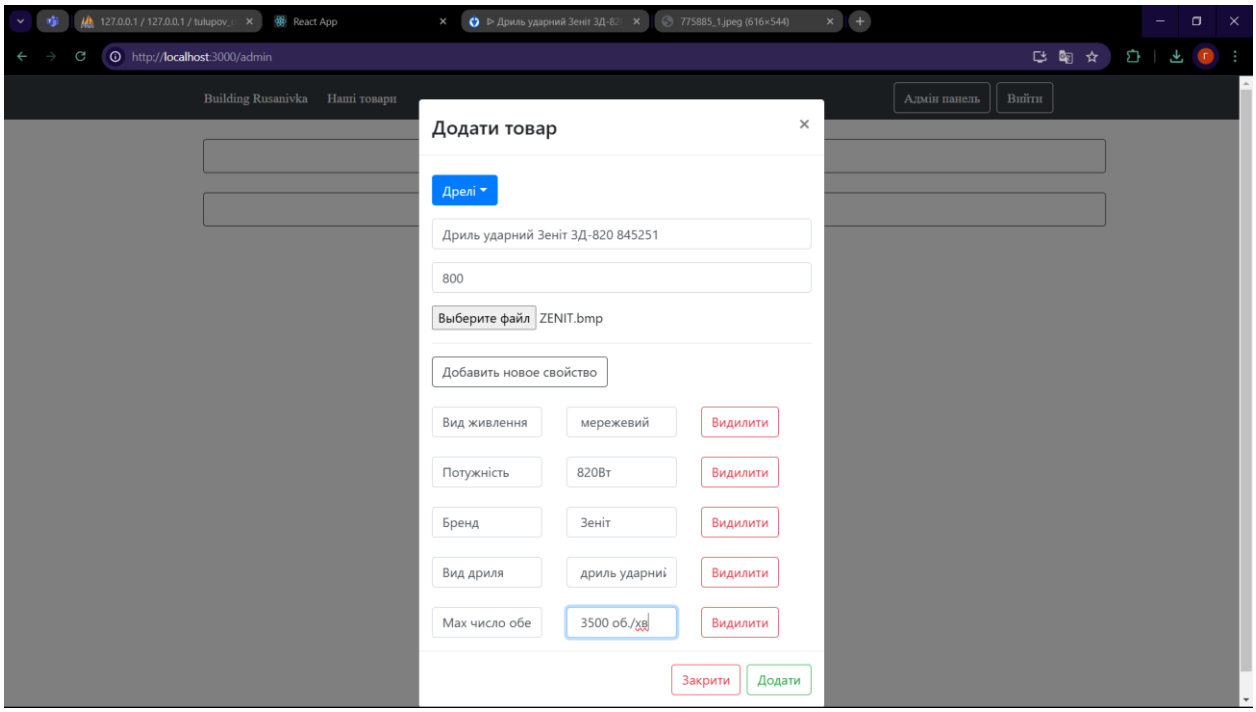


Рис. 4.7. Модальне вікно для створення товару

Натискаємо додати і подивимось чи з’явилися новий тип і товар на головній сторінці. Рисунок 4.8.

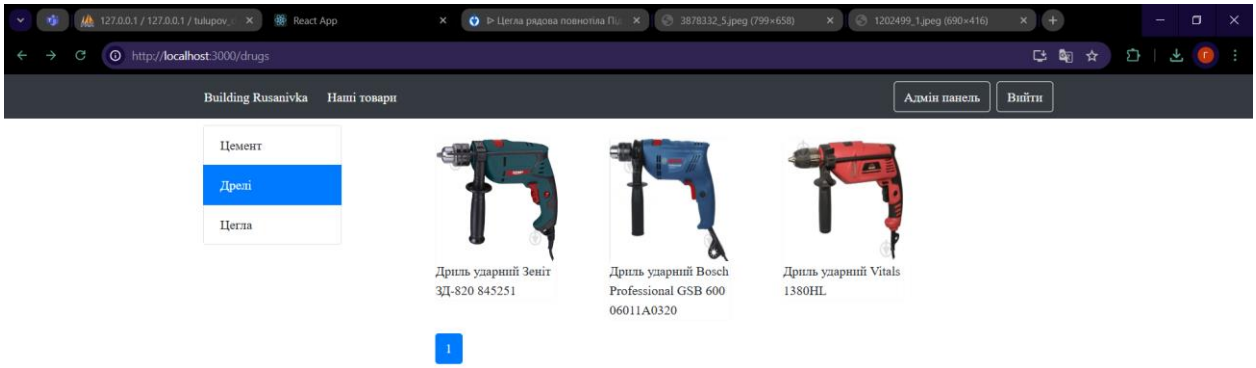


Рис. 4.8. Головна сторінка з доданим типом і товаром

На рисунку можна побачити, що фільтр по типу працює.

4.2 Масштабованість

На разі це не фінальна версія сайту, він ще буде розширюватись. Додадуться нові сторінки завдяки фреймворку React та бібліотеці Bootstrap це можна легко реалізувати.

Висновки до розділу 4

У результаті реалізації задачі була створена структура вебсайту, яка дозволяє ефективно працювати з будівельними матеріалами та інструментами. Головна сторінка сайту надає користувачам можливість ознайомитись з доступними товарами, а фільтр за типом матеріалів та інструментів дозволяє швидко знайти потрібний продукт. Сторінка кожного товару містить детальну інформацію, зокрема фото, назву, вартість та можливість переходу на сайт

постачальника. Система авторизації та реєстрації користувачів гарантує безпечний доступ до персоналізованих функцій. Адмін-панель, доступна тільки користувачам з роллю «ADMIN», дозволяє додавати нові категорії матеріалів та інструментів, а також оновлювати дані товарів. Модальні вікна для додавання нових типів та товарів працюють коректно, і після внесення змін відповідні нові товари та категорії відображаються на головній сторінці. Завдяки використанню фреймворків React і Bootstrap, сайт має високу масштабованість, що дозволяє легко додавати нові сторінки та розширювати функціональність в майбутньому.

РОЗДІЛ 5
ЕКОНОМІЧНИЙ АНАЛІЗ

5.1 Функціонально-вартісний аналіз ПП призначеного для аналізу нелінійних нестаціонарних процесів

5.1.1 Постановка завдання проектування

Проводиться оцінка основних характеристик програмного продукту, призначеного для аналізу нелінійних нестаціонарних процесів. Інтерфейс користувача був розроблений за допомогою мови програмування JavaScript у середовищі розробки Visual Studio Code. Інтерфейс користувача створений за допомогою технології React.js. Програмний продукт призначено для використання сервері та базі даних SQL.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

5.1.2 Обґрунтування функцій програмного продукту

Головна функція F0– розробка програмного продукту, який аналізує процес за вхідними даними та будує його модель для подальшого прогнозування. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F1 – вибір мови програмування;

F2 – розпізнавання вхідних даних;

F3 – інтерфейс користувача.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F1:

а) мова програмування JavaScript;

б) мова програмування HTML;

Функція F2:

а) введення даних вручну (в базу даних);

б) надсилання даних з клієнтської частини сайту до БД.

Функція F3:

а) інтерфейс користувача, створений за технологією Bootstrap;

б) інтерфейс користувача, створений за технологією CSS.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рисунок 5.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 1).

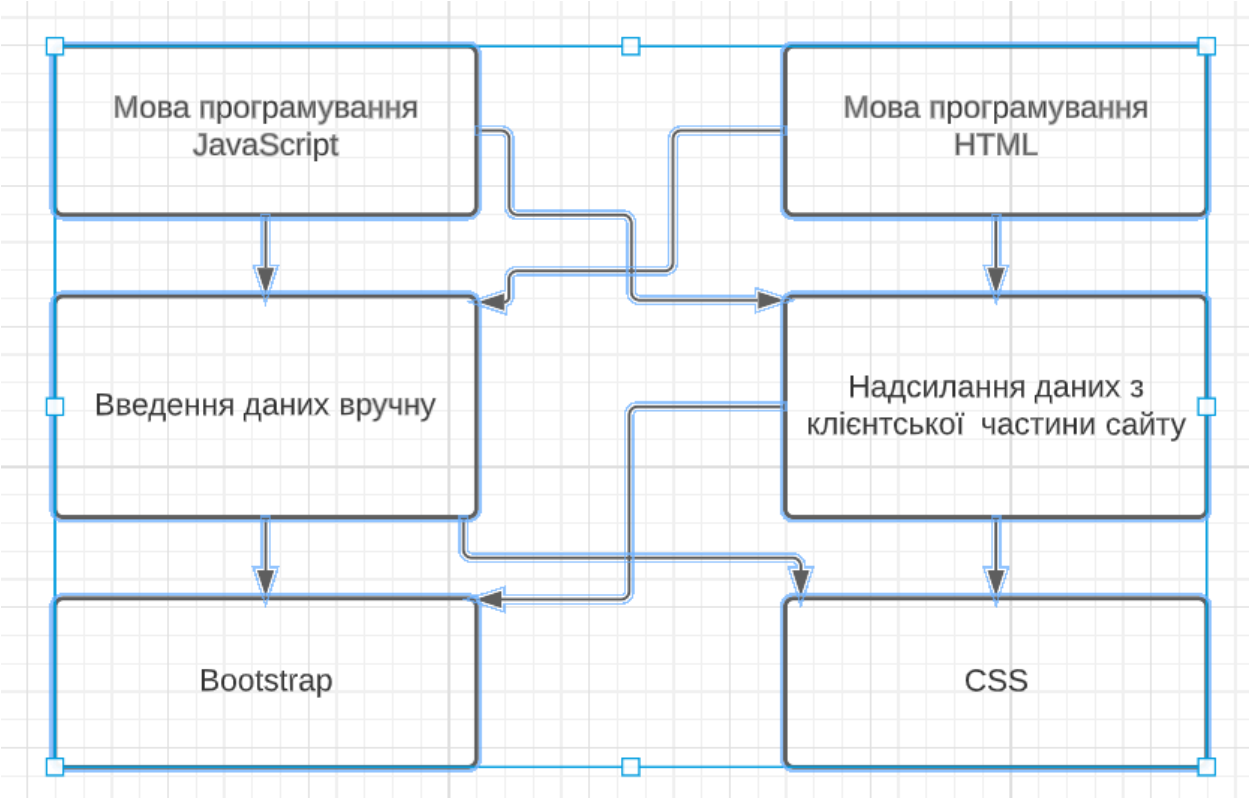


Рис. 5.1 – Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 1

Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	*А*	Високий рівень інтерактивності	Залежність від підтримки браузерів
	Б	Простота використання	Обмежені можливості програмування
F2	А	Точний контроль над введенням даних	Великий обсяг ручної роботи
	Б	Автоматизація процесу	Потенційні проблеми безпеки
F3	*А*	Готові компоненти та сітка	Обмежена унікальність дизайну
	Б	Гнучкість та контроль над стилем	Потреба у великій кількості коду

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція F1:

Оскільки зараз розробка сайтів потребує гарної інтерактивності, то найкраще підходить JavaScript, варіант А.

Функція F2:

Планується що користувачі сайту мали змогу додавати дані, навіть якщо в них не має знань про роботу з БД, тому обираємо варіант Б.

Функція F3:

Інтерфейс користувача не відіграє велику роль у даному програмному продукту, тому вважаємо варіанти а) та б) гідними розгляду.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

F1а – F2б – F3а

F1а – F2б – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

5.1.3 Обґрунтування системи параметрів ПП

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

X1 – швидкодія мови програмування;

X2 – об’єм пам’яті для збереження даних;

X3 – час обробки даних;

X4 – потенційний об’єм програмного коду.

X1: Відображає швидкодію операцій залежно від обраної мови програмування.

X2: Відображає об’єм пам’яті в оперативній пам’яті персонального комп’ютера, необхідний для збереження та обробки даних під час виконання програми.

X3: Відображає час, який витрачається на дії.

X4: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл.2.

Таблиця 2

Основні параметри ПП

Назва параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	Оп/мс	19000	11000	2000
Об'єм пам'яті для збереження даних	X2	Мб	3000	2000	1000
Час обробки даних алгоритмом	X3	мс	400	250	100
Потенційний об'єм програмного коду	X4	кількість строк коду	2000	1500	1000

За даними таблиці 2 будуються графічні характеристики параметрів – рисунок 5.2 – рисунок 5.5.

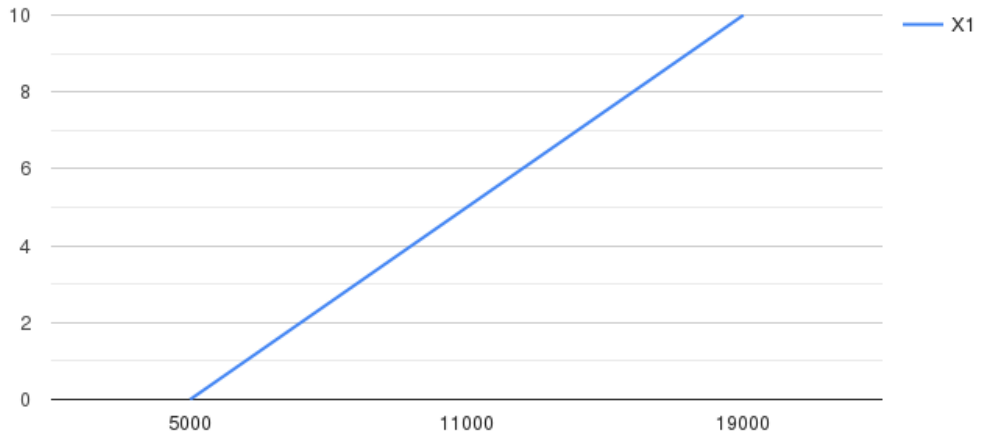


Рис. 5.2 – X1, швидкодія мови програмування

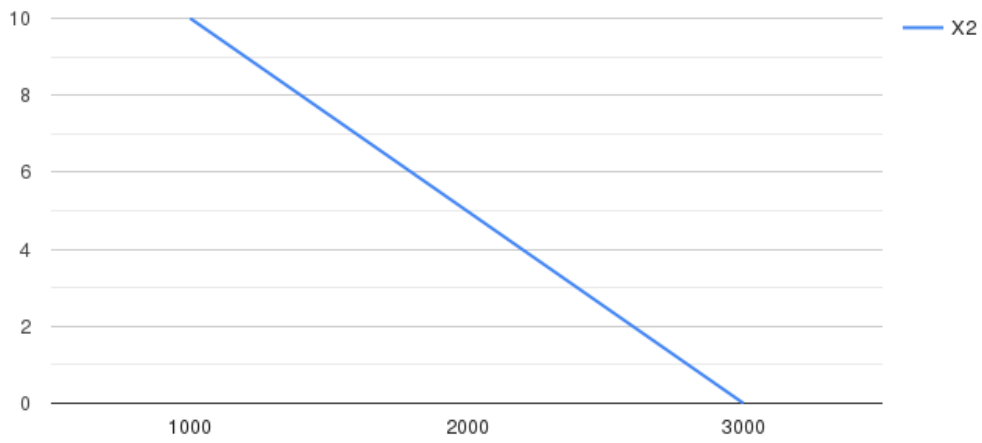


Рис. 5.3 – X2, об’єм пам’яті для збереження даних

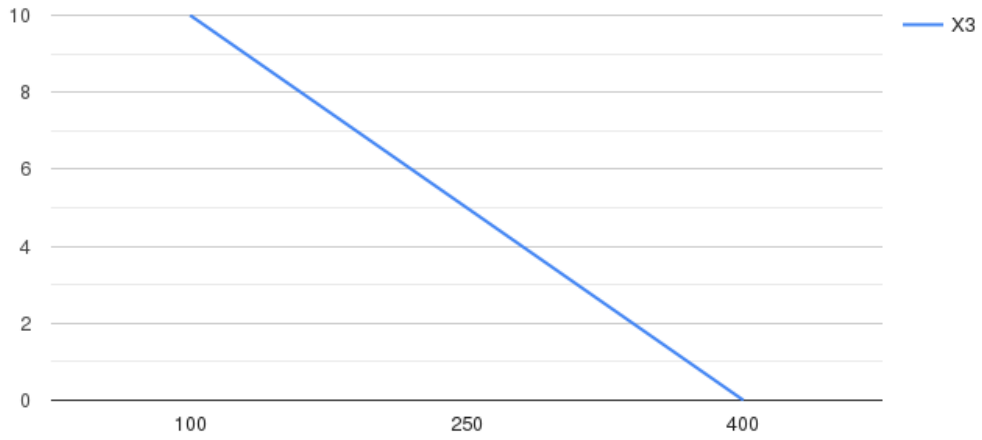


Рис. 5.4 – X3, час обробки даних алгоритмом

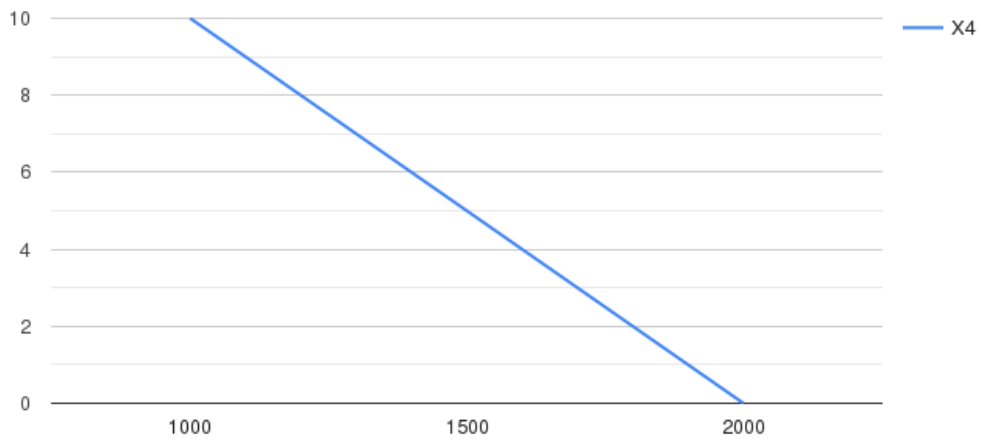


Рис. 5.5 – X4, потенційний об’єм програмного коду

5.1.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який найбільш точно відповідає вимогам клієнтів.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

– визначення рівня значимості параметра шляхом присвоєння різних рангів;

– перевірку придатності експертних оцінок для подальшого використання;

– визначення оцінки попарного пріоритету параметрів;

– обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 3.

Таблиця 3

Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	5	2	3	5	5	5	5	30	3	9
X2	Об’єм пам’яті для збереження даних	Мб	5	1	5	5	4	2	3	25	-2	4
X3	Час обробки даних алгоритмом	Мс	1	3	4	3	4	5	5	25	-2	4
X4	Потенційний об’єм програмного коду	кількість строк коду	3	4	4	5	5	4	3	28	1	1
	Разом		14	10	16	18	18	16	16	108	0	18

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 108,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 27.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 18.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 18}{7^2(5^3 - 5)} = 0{,}04 < W_k = 0{,}67$$

Ранжування не можна вважати достовірним, тому що знайдений коефіцієнт узгодженості не перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.

Таблиця 4

Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	=	>	<	=	>	>	>	>	1,5
X1 і X3	>	<	<	>	>	=	=	>	1,5
X1 і X4	>	<	<	=	=	>	>	>	1,5
X2 і X3	>	<	>	>	=	<	<	<	0,5
X2 і X4	>	<	>	=	>	<	=	>	1,5
X3 і X4	<	<	=	<	<	<	>	<	0,5

Числове значення, що визначає ступінь переваги і–го параметра над j–тим, аіj визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j. \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю А=‖ аіj ‖.

Для кожного параметра зробимо розрахунок вагомості Кві за наступними формулами:

$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}$,де $b_i = \sum_{i=1}^{N} a_{ij}$.

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}$,де $b_i' = \sum_{i=1}^{N} a_{ij} b_j$.

Як видно з таблиці 5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 5

Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер.	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^1	$K_{\text{ві}}^1$
X1	1,0	1,5	1,5	1,5	5,5	0,30	30,25	0,364	166,375	0,425
X2	1,5	1,0	0,5	1,5	4,5	0,25	20,25	0,244	91,125	0,233
X3	1,5	0,5	1,0	0,5	3,5	0,20	12,25	0,148	42,875	0,109
X4	1,5	1,5	0,5	1,0	4,5	0,25	20,25	0,244	91,125	0,233
Всього:					18	1	83	1	391,5	1

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(об'єм пам'яті для збереження даних) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 400мс або варіанту б) 100мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 6):

$$K_K(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j},$$

де n – кількість параметрів;$K_{\text{в}i}$– коефіцієнт вагомості i–го параметра; B_i – оцінка i–го параметра в балах.

Таблиця 6

Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	11000	4,28	0,425	1,819
F2(X2)	Б	2000	3,57	0,233	0,832
F3(X3,X4)	А	400	3,57	0,109	0,389
	Б	100	4	0,233	0,932

За даними з таблиці 6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 1,819 + 0,832 + 0,389 = 3,04

K_{K2} = 1,819 + 0,832 + 0,932 = 3,583

Як видно з розрахунків, кращим є другий варіант, для якого коефіцієнт технічного рівня має найбільше значення.

5.2 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М},$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;$K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =60 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.6. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.7. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 60 \cdot 1.6 \cdot 0.7 = 67,2$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто T_P =20 людино-днів, $K_П$ =0.7,$K_{СК}$ = 1,$K_{СТ}$ =0.6:

$T_2 = 20 \cdot 0.7 \cdot 0.6 = 8,4$ людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$T_I = (67,2 + 8,4 + 4,8 + 8,4) \cdot 8 = 710,4$ людино-годин;

$T_{II} = (67,2 + 8,4 + 6.91 + 8,4) \cdot 8 = 727,28$ людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 20000 грн., один фінансовий аналітик з окладом 40000грн. Визначимо зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} грн.,$$

де М – місячний оклад працівників;T_m – кількість робочих днів тиждень;t – кількість робочих годин в день.

$$С_ч = \frac{20000 + 20000 + 40000}{3 \cdot 21 \cdot 8} = 158{,}73 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д$,

де $С_ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 158,73· 710,4 · 1.3 = 146590,33 грн.

II. $С_{ЗП}$ = 158,73· 727,28 · 1.3 = 150073,50 грн.

Відрахування на єдиний соціальний внесок в залежності від групи професійного ризику (II клас) становить 36,77%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.3677$ = 146590,33 · 0.3677 = 53901 грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.3677$ = 150073,50 · 0.3677 = 55182 грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 20000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З$ = 12 · 20000· 0,2 = 48000 грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З)$ = 48000· (1 + 0.2)=57600 грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.22$ = 57600· 0,22 = 12672 грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 16000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР}$ = 1.15 · 0.25 · 16000 = 4600 грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р$ = 1.15 · 16000 · 0.05 = 920 грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В$ = (365 – 104 – 8 – 16) · 8 · 0.9 = 1706.4 годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН}$ =1706,4 · 0,156 · 0,2436 · 4,87 = 315,80 грн.,

де N_C – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 16000 \cdot 0,67 = 10720$ грн.

Тоді, річні експлуатаційні витрати будуть:

$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$

$С_{ЕКС} = 57600 + 12672 + 4600 + 920 + 315,80 + 10720 = 86827,8$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 86827,8 / 1706,4 = 50,88$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М\text{-}Г} \cdot Т$

I. $С_М = 55,75 \cdot 710,4 = 39604,80$ грн.;

II. $С_М = 55,75 \cdot 727,28 = 40545,86$ грн.;

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

I. $С_Н = 146590,33 \cdot 0,67 = 98215,52$ грн.;

II. $С_Н = 150073,50 \cdot 0,67 = 100549,25$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

I. $С_{ПП} = 146590,33 + 53901 + 39604,80 + 98215,52 = 338311,65$ грн.;

II. $С_{ПП} = 150073,50 + 55182 + 40545,86 + 100549,25 = 364250,61$ грн.;

5.3 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$К_{ТЕРj} = К_{Кj} / С_{Фj}$,

$К_{ТЕР1} = 3,04 / 338311,65 = 8,98 \cdot 10^{-8}$;

$К_{ТЕР2} = 3,583 / 364250,61 = 9,83 \cdot 10^{-8}$;

Як бачимо, найбільш ефективним є другий варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 9,83 \cdot 10^{-8}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 9,83 \cdot 10^{-8}$.

Цей варіант реалізації програмного продукту має такі параметри:

– мова програмування – JavaScript;

– надсилання даних з клієнтської частини сайту з БД;

– інтерфейс користувача, створений за технологією CSS.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, непоганий функціонал і швидкодію.

Висновки до розділу 5

У результаті проведеного економічного аналізу та функціонально-вартісного аналізу програмного продукту для аналізу нелінійних нестаціонарних процесів були оцінені основні характеристики програмного забезпечення, включаючи економічні фактори, продуктивність і сумісність з апаратним забезпеченням. Вибір технологій для розробки, зокрема JavaScript, React.js для інтерфейсу користувача та використання серверних технологій на основі SQL, дозволяє забезпечити оптимальне поєднання вартості і ефективності. Оцінка варіантів реалізації допомогла визначити найбільш ефективний підхід, що відповідає вимогам до продуктивності та масштабованості, забезпечуючи при цьому високу сумісність із різними апаратними платформами. Використання функціонально-вартісного аналізу дозволило вибрати найкраще рішення з точки зору вартості й експлуатаційної ефективності, що в майбутньому забезпечить надійність і доступність продукту для кінцевих користувачів.

ЗАГАЛЬНІ ВИСНОВКИ

В результаті виконаної роботи та розробки сайту для будівельної компанії було досягнуто важливих результатів та отримано цінний практичний досвід. Оцінка даних результатів дозволяє зробити наступні висновки.

По-перше, розроблений сайт виконує свою основну функцію - представляє будівельну компанію в Інтернеті та надає відповідну інформацію. Сайт містить інформацію про продукцію, послуги, контакти та інші важливі аспекти діяльності компанії. Це дозволяє клієнтам та зацікавленим сторонам отримати достовірну та корисну інформацію про компанію та її продукцію.

По-друге, розроблений сайт є зручним та легким у використанні для користувачів. Враховуючи особливості будівельної галузі, було приділено особливу увагу дизайну та навігації, щоб забезпечити зручність використання та ефективну взаємодію з сайтом. Інтерфейс сайту інтуїтивно зрозумілий та легкий у навігації, що дозволяє користувачам швидко знаходити необхідну інформацію та здійснювати необхідні дії.

По-третє, розроблений сайт має потенціал для подальшого розвитку та розширення. Застосування сучасних технологій та розробка гнучкої архітектури сайту дозволяють легко внести зміни та додати новий функціонал у майбутньому. Наприклад, можна розглянути можливість впровадження онлайн-замовлення, збільшення інтерактивності сайту або інтеграцію з іншими системами управління.

Загалом, розробка сайту для будівельної компанії має велике значення для покращення присутності компанії в Інтернеті та забезпечення зручного спілкування з клієнтами. Результати дослідження та розробки свідчать про успішну реалізацію цих цілей і відкривають перспективи для подальшого розвитку та покращення веб-присутності будівельної компанії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ТРЕНДЫ И МЕТОДЫ ВЕБ-РАЗРАБОТКИ В 2022 [Електронний ресурс] – https://merehead.com/ru/blog/dominating-web-development-trends-techniques-in-2022/
2. Особливості розробки сайту в 2023 році [Електронний ресурс] – https://webdevandseo.com.ua/features-of-site-development-in-2023/
3. Які існують технології для розробки сайтів та кому вони підходять [Електронний ресурс] – https://icstudio.online/post/tehnologii-dlya-rozrobky-sajtov
4. ВВЕДЕНИЕ В ТЕХНОЛОГИИ ВЕБ-РАЗРАБОТКИ [Електронний ресурс] – https://webonto.ru/vvedenie-v-tehnologii-veb-razrabotki/
5. Тренды веб-разработки 2023 [Електронний ресурс] – https://vc.ru/u/181937-olga-leonteva/586655-trendy-veb-razrabotki-2023
6. 7 трендов веб-разработки 2023 года [Електронний ресурс] – https://webtense.ru/articles/7-trendov-web-razrabotki-2023-goda/#pwa
7. ТРЕНДЫ ВЕБ РАЗРАБОТКИ 2020 [Електронний ресурс] – https://merehead.com/ru/blog/web-development-trends-in-2020/
8. 11 главных трендов веб-разработки в 2020–2021 [Електронний ресурс] – https://freelancehunt.com/blog/triendy-vieb-razrabotki-2020-2021/
9. 11 Web Development Trends to Expect in 2023, According to Experts & Data [Електронний ресурс] – https://blog.hubspot.com/website/web-development-trends
10. Web Development Trends and the Latest Web Technology Stacks in 2023[Електронний ресурс] – https://clockwise.software/blog/web-development-trends/
11. 15 Burning Web Development Trends to Follow in 2023 [Електронний ресурс] – https://www.codica.com/blog/top-web-development-trends/#4-internet-of-things-io-t

12. 15 latest web development technologies you should know about [Електронний ресурс] – https://www.webdew.com/blog/latest-web-development-technologies#title_12
13. WEB DEVELOPMENT TRENDS: HOW TO BUILD WEB APPS IN 2020 [Електронний ресурс] – https://vilmate.com/blog/web-development-trends-2020/
14. SQL [Електронний ресурс] – https://uk.wikipedia.org/wiki/SQL
15. Мови програмування [Електронний ресурс] – https://uk.wikipedia.org/wiki/Мова_програмування
16. OLAP-аналіз [Електронний ресурс] – https://uk.wikipedia.org/wiki/OLAP
17. ETL-інструменти (Extract, Transform, Load) [Електронний ресурс] – https://uk.wikipedia.org/wiki/ETL
18. NoSQL бази даних [Електронний ресурс] – https://uk.wikipedia.org/wiki/NoSQL
19. Hadoop [Електронний ресурс] – https://ru.wikipedia.org/wiki/Hadoop
20. Spark [Електронний ресурс] – https://uk.wikipedia.org/wiki/Apache_Spark
21. Cassandra [Електронний ресурс] – https://ru.wikipedia.org/wiki/Apache_Cassandra
22. Хмарні БД [Електронний ресурс] – https://uk.wikipedia.org/wiki/Хмарна_база_даних
23. What is React.js? [Електронний ресурс] – https://blog.hubspot.com/website/react-js
24. What is Webpack [Електронний ресурс] – https://survivejs.com/webpack/what-is-webpack
25. CSS [Електронний ресурс] – https://en.wikipedia.org/wiki/CSS
26. What is Bootstrap? [Електронний ресурс] – https://www.techtarget.com/whatis/definition/bootstrap
27. Node.js [Електронний ресурс] – https://ru.wikipedia.org/wiki/Node.js
28. What is MySQL? [Електронний ресурс] – https://www.oracle.com/mysql/what-is-mysql/

29. phpMyAdmin Bringing MySQL to the web [Електронний ресурс] – https://www.phpmyadmin.net/

ДОДАТОК А.
ЛІСТИНГ КОДУ КЛІЄНТСЬКОЇ ЧАСТИНИ

Файл App.js

```
import React, {useContext, useEffect, useState} from 'react';
import {BrowserRouter} from "react-router-dom";
import AppRouter from "./components/AppRouter";
import NavBar from "./components/NavBar";
import {observer} from "mobx-react-lite";
import {Context} from "./index";
import {check} from "./http/userAPI";
import {Spinner} from "react-bootstrap";

const App = observer(() => {
    const {user} = useContext(Context)
    const [loading, setLoading] = useState(true)

    useEffect(() => {
        check().then(data => {
            user.setUser(true)
            user.setIsAuth(true)
        }).finally(() => setLoading(false))
    }, [])

    if (loading) {
        return <Spinner animation={"grow"}/>
    }

    return (
        <BrowserRouter>
            <NavBar />
            <AppRouter />
        </BrowserRouter>
    );
});

export default App;
```

Файл DrugStore.js

```
import {makeAutoObservable} from "mobx";

export default class DrugStore {
    constructor() {
        this._types = []
        this._drugs = []
        this._selectedType = {}
```

```
        this._page = 1
        this._totalCount = 0
        this._limit = 12
        makeAutoObservable(this)
    }

    setTypes(types) {
        this._types = types
    }

    setDrugs(drugs) {
        this._drugs = drugs
    }

    setSelectedType(type) {
        this.setPage(1)
        this._selectedType = type
    }

    setPage(page) {
        this._page = page
    }
    setTotalCount(count) {
        this._totalCount = count
    }

    get types() {
        return this._types
    }

    get drugs() {
        return this._drugs
    }
    get selectedType() {
        return this._selectedType
    }

    get totalCount() {
        return this._totalCount
    }
    get page() {
        return this._page
    }
    get limit() {
        return this._limit
    }
}
```

Файл Admin.js

```
import React, { useState } from "react";
import { Button, Container } from "react-bootstrap";
import CreateDrug from "../components/modals/CreateDrug";
import CreateType from "../components/modals/CreateType";

const Admin = () => {
  const [typeVisible, setTypeVisible] = useState(false);
  const [drugVisible, setDrugVisible] = useState(false);

  return (
    <Container className="d-flex flex-column">
      <Button
        variant={"outline-dark"}
        className="mt-4 p-2"
        onClick={() => setTypeVisible(true)}
      >
        Додати тип
      </Button>{' '}

      <Button
        variant={"outline-dark"}
        className="mt-4 p-2"
        onClick={() => setDrugVisible(true)}
      >
        Додати медикамент
      </Button>{' '}
      <CreateDrug
        show={drugVisible}
        onHide={() => setDrugVisible(false)}
      />
      <CreateType show={typeVisible} onHide={() => setTypeVisible(false)} />
    </Container>
  );
};

export default Admin;
```

Файл Auth.js

```
import React, {useContext, useState} from 'react';
import {Container, Form} from "react-bootstrap";
import Card from "react-bootstrap/Card";
import Button from "react-bootstrap/Button";
import Row from "react-bootstrap/Row";
import {NavLink, useLocation, useHistory} from "react-router-dom";
import {LOGIN_ROUTE, REGISTRATION_ROUTE, SHOP_ROUTE} from "../utils/consts";
import {login, registration} from "../http/userAPI";
import {observer} from "mobx-react-lite";
import {Context} from "../index";
```

```
const Auth = observer(() => {
    const {user} = useContext(Context)
    const location = useLocation()
    const history = useHistory()
    const isLogin = location.pathname === LOGIN_ROUTE
    const [email, setEmail] = useState('')
    const [password, setPassword] = useState('')

    const click = async () => {
        try {
            let data;
            if (isLogin) {
                data = await login(email, password);
            } else {
                data = await registration(email, password);
            }
            user.setUser(user)
            user.setIsAuth(true)
            history.push(SHOP_ROUTE)
        } catch (e) {
            alert(e.response.data.message)
        }

    }

    return (
        <Container
            className="d-flex justify-content-center align-items-center"
            style={{height: window.innerHeight - 54}}
        >
            <Card style={{width: 600}} className="p-5">
                <h2 className="m-auto">{isLogin ? 'Авторизація' :
"Реєстрація"}</h2>
                <Form className="d-flex flex-column">
                    <Form.Control
                        className="mt-3"
                        placeholder="Введите ваш email..."
                        value={email}
                        onChange={e => setEmail(e.target.value)}
                    />
                    <Form.Control
                        className="mt-3"
                        placeholder="Введите ваш пароль..."
                        value={password}
                        onChange={e => setPassword(e.target.value)}
                        type="password"
                    />
                    <Row className="d-flex justify-content-between mt-3 pl-3 pr-
3">
                        {isLogin ?
                            <div>
```

```
                                Не має акаунта? <NavLink
to={REGISTRATION_ROUTE}>Зареєструйтесь</NavLink>
                            </div>
                            :
                            <div>
                                Є акаунт? <NavLink
to={LOGIN_ROUTE}>Увійдіть</NavLink>
                            </div>
                        }
                        <Button
                            variant={"outline-success"}
                            onClick={click}
                        >
                            {isLogin ? 'Вхід' : 'Реєстрація'}
                        </Button>
                    </Row>

                </Form>
            </Card>
        </Container>
    );
});

export default Auth;
```

Файл Shop.js

```
import React, {useContext, useEffect} from 'react';
import {Container} from "react-bootstrap";
import Row from "react-bootstrap/Row";
import Col from "react-bootstrap/Col";
import TypeBar from "../components/TypeBar";
import DrugList from "../components/DrugList";
import {observer} from "mobx-react-lite";
import {Context} from "../index";
import {fetchDrugs, fetchTypes} from "../http/drugAPI";
import Pages from "../components/Pages";

const Shop = observer(() => {
    const {drug} = useContext(Context)

    useEffect(() => {
        fetchTypes().then(data => drug.setTypes(data))
        fetchDrugs(null, null, 1, 2).then(data => {
            drug.setDrugs(data.rows)
            drug.setTotalCount(data.count)
        })
    }, [])

    useEffect(() => {
```

```
        fetchDrugs(drug.selectedType.id, drug.page, drug.limit).then(data => {
            drug.setDrugs(data.rows)
            drug.setTotalCount(data.count)
        })
    }, [drug.page, drug.selectedType,])

    return (
        <Container>
            <Row className="mt-2">
                <Col md={3}>
                    <TypeBar/>
                </Col>
                <Col md={9}>

                    <DrugList/>
                    <Pages/>
                </Col>
            </Row>
        </Container>
    );
});

export default Shop;
```

Файл CreateDrug.js

```
import React, {useContext, useEffect, useState} from 'react';
import Modal from "react-bootstrap/Modal";
import {Button, Dropdown, Form, Row, Col} from "react-bootstrap";
import {Context} from "../../index";
import {createDrug, fetchTypes} from "../../http/drugAPI";
import {observer} from "mobx-react-lite";

const CreateDrug = observer(({show, onHide}) => {
    const {drug} = useContext(Context)
    const [name, setName] = useState('')
    const [price, setPrice] = useState(0)
    const [file, setFile] = useState(null)
    const [info, setInfo] = useState([])

    useEffect(() => {
        fetchTypes().then(data => drug.setTypes(data))
    }, [])

    const addInfo = () => {
        setInfo([...info, {title: '', description: '', number: Date.now()}])
    }
    const removeInfo = (number) => {
```

```
        setInfo(info.filter(i => i.number !== number))
    }
    const changeInfo = (key, value, number) => {
        setInfo(info.map(i => i.number === number ? {...i, [key]: value} : i))
    }

    const selectFile = e => {
        setFile(e.target.files[0])
    }

    const addDrug = () => {
        const formData = new FormData()
        formData.append('name', name)
        formData.append('price', `${price}`)
        formData.append('img', file)
        formData.append('typeId', drug.selectedType.id)
        formData.append('info', JSON.stringify(info))
        createDrug(formData).then(data => onHide())
    }

    return (
        <Modal
            show={show}
            onHide={onHide}
            centered
        >
            <Modal.Header closeButton>
                <Modal.Title id="contained-modal-title-vcenter">
                    Додати товар
                </Modal.Title>
            </Modal.Header>
            <Modal.Body>
                <Form>
                    <Dropdown className="mt-2 mb-2">
                        <Dropdown.Toggle>{drug.selectedType.name || "Виберіть
тип"}</Dropdown.Toggle>
                        <Dropdown.Menu>
                            {drug.types.map(type =>
                                <Dropdown.Item
                                    onClick={() => drug.setSelectedType(type)}
                                    key={type.id}
                                >
                                    {type.name}
                                </Dropdown.Item>
                            )}
                        </Dropdown.Menu>
                    </Dropdown>

                    <Form.Control
                        value={name}
                        onChange={e => setName(e.target.value)}
```

```
                        className="mt-3"
                        placeholder="Введіть назву товару"
                    />
                    <Form.Control
                        value={price}
                        onChange={e => setPrice(Number(e.target.value))}
                        className="mt-3"
                        placeholder="Введіть ціну товару"
                        type="number"
                    />
                    <Form.Control
                        className="mt-3"
                        type="file"
                        onChange={selectFile}
                    />
                    <hr/>
                    <Button
                        variant={"outline-dark"}
                        onClick={addInfo}
                    >
                        Добавить новое свойство
                    </Button>{' '}
                    {info.map(i =>
                        <Row className="mt-4" key={i.number}>
                            <Col md={4}>
                                <Form.Control
                                    value={i.title}
                                    onChange={(e) => changeInfo('title',
e.target.value, i.number)}
                                    placeholder="Введіть назву характеристики"
                                />
                            </Col>
                            <Col md={4}>
                                <Form.Control
                                    value={i.description}
                                    onChange={(e) => changeInfo('description',
e.target.value, i.number)}
                                    placeholder="Введіть опис характеристики"
                                />
                            </Col>
                            <Col md={4}>
                                <Button
                                    onClick={() => removeInfo(i.number)}
                                    variant={"outline-danger"}
                                >
                                    Видилити
                                </Button>{' '}
                            </Col>
                        </Row>
                    )}
                </Form>
```

```
            </Modal.Body>
            <Modal.Footer>
                <Button variant="outline-danger"
onClick={onHide}>Закрити</Button>{' '}
                <Button variant="outline-success"
onClick={addDrug}>Додати</Button>{' '}
            </Modal.Footer>
        </Modal>
    );
});

export default CreateDrug;
```

Файл MainPage.js

```
import React from "react";
import "./Basket.css";

const Basket = () => {
  return (
    <>
      <main class="background content">
        <div class="container">
          <section>
            <h2 class="warm_net_title">Про компанію</h2>
            <p class="warm_net_plot">
              <p>
                Building Rusanivka – лідер ринку України за обсягом продажів в
                натуральному виразі.
              </p>
              <p>
                90-річний досвід та сертифікація виробництва за GMP дозволили
                нам заслужити довіру 15 країн світу.
              </p>
              <p>
                Ми випускаємо 180 товарів з фокусом на будівельні матеріали та
                інструменти.
              </p>
            </p>
            <div class="warm_net">
              <p class="warm_net_item_1">60+ товарів у портфелі розробки</p>
              <p class="warm_net_item_2">3 основні напрямки розвитку</p>
              <p class="warm_net_item_3">
                #1 за обсягом продажів в натуральному виразі
              </p>
              <p class="warm_net_item_4">GMP сертифікація з 2002 року</p>
              <p class="warm_net_item_5">
                30% інвестицій щорічно направляються на нові продукти
              </p>
            </div>
```

```
    </section>
    <section class="building_vehicle_content">
      <h2 class="building_vehicle_title">Наші цінності </h2>
      <div class="building_vehicle">
        <p class="building_vehicle_item_1">
          Амбіційне лідерство
          <br />
          <span>
            Ми найкращі у тому, що ми робимо – кожен і разом. Ми мислимо і
            діємо за межами стереотипів. Ми зухвалі і невгамовні –
            досягнувши високої мети, ми ставимо перед собою ще сміливіші
            цілі.
          </span>
        </p>
        <p class="building_vehicle_item_2">
          Extreme ownership (загострене відчуття власника)
          <br />
          <span>
            Кожен з нас щодня досягає більшого і несе відповідальність за
            кінцевий результат компанії як власник.
          </span>
        </p>
        <p class="building_vehicle_item_3">
          Відповідальність <br />
          <span>
            Ми щоденно свідомо несемо відповідальність перед колегами,
            пацієнтами, медичною спільнотою, партнерами, суспільством і
            планетою.
          </span>
        </p>
        <p class="building_vehicle_item_4">
          Надбання <br />{" "}
          <span>
            Ми пишаємося здобутками, які нас сформували як лідерів і
            скеровують на подальший розвиток. Це підходи, ставлення,
            стандарти, які вибудували міцний фундамент і які ми гордо
            несемо з собою далі: амбітність, якість як наше ДНК,
            піклування про людей, стійкість у складні періоди,
            далекоглядність.
          </span>
        </p>
        <p class="building_vehicle_item_5">Увага до наших партнерів</p>
        <p class="building_vehicle_item_6">Гарантія якості</p>
        <p class="building_vehicle_item_7">Піклування про природу</p>
      </div>
    </section>
  </div>
</main>

<section class="background_2">
  <div class="container client_content">
```

```
<div class="client_header">
  <h2>Наші партнери</h2>
  {/* <a href="#" class="all_projects">
    Все проекты
  </a> */}
</div>
<div class="clients">
  <div class="clients_item_1 clients_item">
    <img src="./img/clients/clients_1.svg" alt="" />
    <div class="clients_address">
      <p class="hover_text">
        ЕПІЦЕНТР
      </p>
    </div>
  </div>
  <div class="clients_item_2 clients_item">
    <div>
      <img src="./img/clients/clients_2_1.svg" alt="" />
    </div>
    <div>
      <img src="./img/clients/clients_2_2.svg" alt="" />
    </div>
    <div class="clients_address">
      <p class="hover_text">
        DNIPRO TM
      </p>
    </div>
  </div>
  <div class="clients_item_3 clients_item">
    <img src="./img/clients/clients_3.svg" alt="" />
    <div class="clients_address">
      <p class="hover_text">
        УКРБУД
      </p>
    </div>
  </div>
  <div class="clients_item_4 clients_item">
    <img src="./img/clients/clients_4.svg" alt="" />
    <div class="clients_address">
    <p class="hover_text">
        КИЇВМІСЬКБУД
      </p>
    </div>
  </div>
  <div class="clients_item_5 clients_item">
    <img src="./img/clients/clients_5.svg" alt="" />
    <div class="clients_address">
    <p class="hover_text">
        АВТОШЛЯХ
      </p>
    </div>
```

```
            </div>
            <div class="clients_item_6 clients_item">
              <img src="./img/clients/clients_6.svg" alt="" />
              <div class="clients_address">
              <p class="hover_text">
                  АТОСТРАДА
                </p>
              </div>
            </div>
          </div>
        </div>
      </section>
    </>
  );
};

export default Basket;
```

ДОДАТОК Б.
ЛІСТИНГ КОДУ СЕРВЕРНОЇ ЧАСТИНИ

Файл index.js

```
require("dotenv").config();
const express = require("express");
const sequelize = require("./db")
const models = require("./models/models");
const cors = require("cors")
const fileUpload = require("express-fileupload")
const router = require("./routes")
const errorHandler = require("./middleware/ErrorHandlingMiddleware")
const path = require('path')

const PORT = process.env.PORT || 8080;

const app = express();
app.use(cors())
app.use(express.json())
app.use(express.static(path.resolve(__dirname, 'static')))
app.use(fileUpload({}))
app.use("/api", router)


app.use(errorHandler)

const start = async () => {
  try {
    await sequelize.authenticate();
    await sequelize.sync();
    // app.listen(PORT, () => console.log(`Server started on port ${PORT}`));
    app.listen(PORT, 'localhost'); // or server.listen(3001, '0.0.0.0'); for all
interfaces
app.on('listening', function() {
    console.log('Express server started on port  at ', app.address().port,
app.address().address);
});
  } catch (e) {
    console.log(e);
  }
};

start();
```

Файл package.js

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon index.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.1.0",
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "express": "^4.18.2",
    "express-fileupload": "^1.4.0",
    "fs": "^0.0.1-security",
    "http": "^0.0.1-security",
    "jsonwebtoken": "^9.0.0",
    "mysql": "^2.18.1",
    "mysql2": "^3.3.1",
    "path": "^0.12.7",
    "sequelize": "^6.31.1",
    "url": "^0.11.0",
    "uuid": "^9.0.0"
  },
  "devDependencies": {
    "nodemon": "^2.0.22"
  }
}
```

Файл package-lock.js

```
{
  "name": "server",
  "version": "1.0.0",
  "lockfileVersion": 3,
  "requires": true,
  "packages": {
    "": {
      "name": "server",
      "version": "1.0.0",
      "license": "ISC",
      "dependencies": {
        "bcrypt": "^5.1.0",
        "cors": "^2.8.5",
        "dotenv": "^16.0.3",
```

```
        "express": "^4.18.2",
        "express-fileupload": "^1.4.0",
        "fs": "^0.0.1-security",
        "http": "^0.0.1-security",
        "jsonwebtoken": "^9.0.0",
        "mysql": "^2.18.1",
        "mysql2": "^3.3.1",
        "path": "^0.12.7",
        "sequelize": "^6.31.1",
        "url": "^0.11.0",
        "uuid": "^9.0.0"
      },
      "devDependencies": {
        "nodemon": "^2.0.22"
      }
    },
    "node_modules/@mapbox/node-pre-gyp": {
      "version": "1.0.10",
      "resolved": "https://registry.npmjs.org/@mapbox/node-pre-gyp/-/node-pre-
gyp-1.0.10.tgz",
      "integrity": "sha512-
4ySo4CjzStuprMwk35H5pPbkymjv1SF3jGLj6rAHp/xT/RF7TL7bd9CTm1xDY49K2qF7jmR/g7k+SkLET
P6opA==",
      "dependencies": {
        "detect-libc": "^2.0.0",
        "https-proxy-agent": "^5.0.0",
        "make-dir": "^3.1.0",
        "node-fetch": "^2.6.7",
        "nopt": "^5.0.0",
        "npmlog": "^5.0.1",
        "rimraf": "^3.0.2",
        "semver": "^7.3.5",
        "tar": "^6.1.11"
      },
      "bin": {
        "node-pre-gyp": "bin/node-pre-gyp"
      }
    },
    "node_modules/@mapbox/node-pre-gyp/node_modules/nopt": {
      "version": "5.0.0",
      "resolved": "https://registry.npmjs.org/nopt/-/nopt-5.0.0.tgz",
      "integrity": "sha512-
Tbj67rffqceeLpcRXrT7vKAN8CwfPeIBgM7E6iBkmKLV7bEMwpGgYLGv0jACUsECaa/vuxP0IjEont6um
dMgtQ==",
      "dependencies": {
        "abbrev": "1"
      },
      "bin": {
        "nopt": "bin/nopt.js"
      },
      "engines": {
```

```
        "node": ">=6"
      }
    },
    "node_modules/@types/debug": {
      "version": "4.1.7",
      "resolved": "https://registry.npmjs.org/@types/debug/-/debug-4.1.7.tgz",
      "integrity": "sha512-
9AonUzyTjXXhEOa0DnqpzZi6VHlqKMswga9EXjpXnnqxwLtdvPPtlO8evrI5D9S6asFRCQ6v+wpiUKbw+
vKqyg==",
      "dependencies": {
        "@types/ms": "*"
      }
    },
    "node_modules/@types/ms": {
      "version": "0.7.31",
      "resolved": "https://registry.npmjs.org/@types/ms/-/ms-0.7.31.tgz",
      "integrity": "sha512-
iiUgKzV9AuaEkZqkOLDIvlQiL6ltuZd9tGcW3gwpnX8JbuiuhFlEGmmFXEXkN50Cvq7Os88IY2v0dkDqX
YWVgA=="
    },
    "node_modules/@types/node": {
      "version": "20.1.7",
      "resolved": "https://registry.npmjs.org/@types/node/-/node-20.1.7.tgz",
      "integrity": "sha512-
WCuw/o4GSwDGMoonES8rcvwsig77dGCMbZDrZr2x4ZZiNW4P/gcoZXe/0twgtobcTkmg9TuKflxYL/Duw
DyJzg=="
    },
    "node_modules/@types/validator": {
      "version": "13.7.17",
      "resolved": "https://registry.npmjs.org/@types/validator/-/validator-
13.7.17.tgz",
      "integrity": "sha512-
aqayTNmeWrZcvnG2MG9eGYI6b7S5fl+yKgPs6bAjOTwPS316R5SxBGKvtSExfyoJU7pIeHJfsHI0Ji41R
VMkvQ=="
    },
    "node_modules/abbrev": {
      "version": "1.1.1",
      "resolved": "https://registry.npmjs.org/abbrev/-/abbrev-1.1.1.tgz",
      "integrity": "sha512-
nne9/IiQ/hzIhY6pdDnbBtz7DjPTKrY00P/zvPSm5pOFkl6xuGrGnXn/VtTNNfNtAfZ9/1RtehkszU9qc
Tii0Q=="
    },
    "node_modules/accepts": {
      "version": "1.3.8",
      "resolved": "https://registry.npmjs.org/accepts/-/accepts-1.3.8.tgz",
      "integrity": "sha512-
PYAthTa2m2VKxuvSD3DPC/Gy+U+sOA1LAuT8mkmRuvw+NACSaeXEQ+NHcVF7rONl6qcaxV3Uuemwawk+7
+SJLw==",
      "dependencies": {
        "mime-types": "~2.1.34",
        "negotiator": "0.6.3"
```

```
      },
      "engines": {
        "node": ">= 0.6"
      }
    },
    "node_modules/agent-base": {
      "version": "6.0.2",
      "resolved": "https://registry.npmjs.org/agent-base/-/agent-base-6.0.2.tgz",
      "integrity": "sha512-
RZNwNclF7+MS/8bDg70amg32dyeZGZxiDuQmZxKLAlQjr3jGyLx+4Kkk58UO7D2QdgFIQCovuSuZESne6
RG6XQ==",
      "dependencies": {
        "debug": "4"
      },
      "engines": {
        "node": ">= 6.0.0"
      }
    },
    "node_modules/agent-base/node_modules/debug": {
      "version": "4.3.4",
      "resolved": "https://registry.npmjs.org/debug/-/debug-4.3.4.tgz",
      "integrity": "sha512-
PRWFHuSU3eDtQJPvnNY7Jcket1j0t5OuOsFzPPzsekD52Zl8qUfFIPEiswXqIvHWGVHOgX+7G/vCNNheh
wxfkQ==",
      "dependencies": {
        "ms": "2.1.2"
      },
      "engines": {
        "node": ">=6.0"
      },
      "peerDependenciesMeta": {
        "supports-color": {
          "optional": true
        }
      }
    },
    "node_modules/agent-base/node_modules/ms": {
      "version": "2.1.2",
      "resolved": "https://registry.npmjs.org/ms/-/ms-2.1.2.tgz",
      "integrity": "sha512-
sGkPx+VjMtmA6MX27oA4FBFELFCZZ4S4XqeGOXCv68tT+jb3vk/RyaKWP0PTKyWtmLSM0b+adUTEvbs1P
EaH2w=="
    },
    "node_modules/ansi-regex": {
      "version": "5.0.1",
      "resolved": "https://registry.npmjs.org/ansi-regex/-/ansi-regex-5.0.1.tgz",
      "integrity": "sha512-
quJQXlTSUGL2LH9SUXo8VwsY4soanhgo6LNSm84E1LBcE8s3O0wpdiRzyR9z/ZZJMlMWv37qOOb9pdJlM
UEKFQ==",
      "engines": {
        "node": ">=8"
```

```
      }
    },
    "node_modules/anymatch": {
      "version": "3.1.3",
      "resolved": "https://registry.npmjs.org/anymatch/-/anymatch-3.1.3.tgz",
      "integrity": "sha512-
KMReFUr0B4t+D+OBkjR3KYqvocp2XaSzO55UcB6mgQMd3KbcE+mWTyvVV7D/zsdEbNnV6acZUutkiHQXv
Tr1Rw==",
      "dev": true,
      "dependencies": {
        "normalize-path": "^3.0.0",
        "picomatch": "^2.0.4"
      },
      "engines": {
        "node": ">= 8"
      }
    },
    "node_modules/aproba": {
      "version": "2.0.0",
      "resolved": "https://registry.npmjs.org/aproba/-/aproba-2.0.0.tgz",
      "integrity": "sha512-
lYe4Gx7QT+MKGbDsA+Z+he/Wtef0BiwDOlK/XkBrdfsh9J/jPPXbX0tE9x9cl27Tmu5gg3QUbUrQYa/y+
KOHPQ=="
    },
    "node_modules/are-we-there-yet": {
      "version": "2.0.0",
      "resolved": "https://registry.npmjs.org/are-we-there-yet/-/are-we-there-
yet-2.0.0.tgz",
      "integrity": "sha512-
Ci/qENmwHnsYo9xKIcUJN5LeDKdJ6R1Z1j9V/J5wyq8nh/mYPEpIKJbBZXtZjG04HiK7zV/p6Vs9952Mr
MeUIw==",
      "dependencies": {
        "delegates": "^1.0.0",
        "readable-stream": "^3.6.0"
      },
      "engines": {
        "node": ">=10"
      }
    },
    "node_modules/are-we-there-yet/node_modules/readable-stream": {
      "version": "3.6.2",
      "resolved": "https://registry.npmjs.org/readable-stream/-/readable-stream-
3.6.2.tgz",
      "integrity": "sha512-
9u/sniCrY3D5WdsERHzHE4G2YCXqoG5FTHUiCC4SIbr6XcLZBY05ya9EKjYek9O5xOAwjGq+1JdGBAS7Q
9ScoA==",
      "dependencies": {
        "inherits": "^2.0.3",
        "string_decoder": "^1.1.1",
        "util-deprecate": "^1.0.1"
      },
```

```
      "engines": {
        "node": ">= 6"
      }
    },
    "node_modules/array-flatten": {
      "version": "1.1.1",
      "resolved": "https://registry.npmjs.org/array-flatten/-/array-flatten-
1.1.1.tgz",
      "integrity": "sha512-
PCVAQswWemu6UdxsDFFX/+gVeYqKAod3D3UVm91jHwynguOwAvYPhx8nNlM++NqRcK6CxxpUafjmhIdKi
Hibqg=="
    },
    "node_modules/balanced-match": {
      "version": "1.0.2",
      "resolved": "https://registry.npmjs.org/balanced-match/-/balanced-match-
1.0.2.tgz",
      "integrity": "sha512-
3oSeUO0TMV67hN1AmbXsK4yaqU7tjiHlbxRDZOpH0KW9+CeX4bRAaX0Anxt0tx2MrpRpWwQaPwIlISEJh
YU5Pw=="
    },
    "node_modules/bcrypt": {
      "version": "5.1.0",
      "resolved": "https://registry.npmjs.org/bcrypt/-/bcrypt-5.1.0.tgz",
      "integrity": "sha512-
RHBS7HI5N5tEnGTmtR/pppX0mmDSBpQ4aCBsj7CEQfYXDcO74A8sIBYcJMuCsis2E81zDxeENYhv66oZw
LiA+Q==",
      "hasInstallScript": true,
      "dependencies": {
        "@mapbox/node-pre-gyp": "^1.0.10",
        "node-addon-api": "^5.0.0"
      },
      "engines": {
        "node": ">= 10.0.0"
      }
    },
    "node_modules/bignumber.js": {
      "version": "9.0.0",
      "resolved": "https://registry.npmjs.org/bignumber.js/-/bignumber.js-
9.0.0.tgz",
      "integrity": "sha512-
t/OYhhJ2SD+YGBQcjY8GzzDHEk9f3nerxjtfa6tlMXfe7frs/WozhvCNoGvpM0P3bNf3Gq5ZRMlGr5f3r
4/N8A==",
      "engines": {
        "node": "*"
      }
    },
    "node_modules/binary-extensions": {
      "version": "2.2.0",
      "resolved": "https://registry.npmjs.org/binary-extensions/-/binary-
extensions-2.2.0.tgz",
```

```
      "integrity": "sha512-
jDctJ/IVQbZoJykoeHbhXpOlNBqGNcwXJKJog42E5HDPUwQTSdjCHdihjj0DlnheQ7blbT6dHOafNAiS8
ooQKA==",
      "dev": true,
      "engines": {
        "node": ">=8"
      }
    },
    "node_modules/body-parser": {
      "version": "1.20.1",
      "resolved": "https://registry.npmjs.org/body-parser/-/body-parser-
1.20.1.tgz",
      "integrity": "sha512-
jWi7abTbYwajOytWCQc37VulmWiRae5RyTpaCyDcS5/lMdtwSz5lOpDE67srw/HYe35f1z3fDQw+3txg7
gNtWw==",
      "dependencies": {
        "bytes": "3.1.2",
        "content-type": "~1.0.4",
        "debug": "2.6.9",
        "depd": "2.0.0",
        "destroy": "1.2.0",
        "http-errors": "2.0.0",
        "iconv-lite": "0.4.24",
        "on-finished": "2.4.1",
        "qs": "6.11.0",
        "raw-body": "2.5.1",
        "type-is": "~1.6.18",
        "unpipe": "1.0.0"
      },
      "engines": {
        "node": ">= 0.8",
        "npm": "1.2.8000 || >= 1.4.16"
      }
    },
    "node_modules/brace-expansion": {
      "version": "1.1.11",
      "resolved": "https://registry.npmjs.org/brace-expansion/-/brace-expansion-
1.1.11.tgz",
      "integrity": "sha512-
iCuPHDFgrHX7H2vEI/5xpz07zSHB00TpugqhmYtVmMO6518mCuRMoOYFldEBl0g187ufozdaHgWKcYFb6
1qGiA==",
      "dependencies": {
        "balanced-match": "^1.0.0",
        "concat-map": "0.0.1"
      }
    },
    "node_modules/braces": {
      "version": "3.0.2",
      "resolved": "https://registry.npmjs.org/braces/-/braces-3.0.2.tgz",
```

```
      "integrity": "sha512-
b8um+L1RzM3WDSzvhm6gIz1yfTbBt6YTlcEKAvsmqCZZFw46z626lVj9j1yEPW33H5H+lBQpZMP1k8l+7
8Ha0A==",
      "dev": true,
      "dependencies": {
        "fill-range": "^7.0.1"
      },
      "engines": {
        "node": ">=8"
      }
    },
    "node_modules/buffer-equal-constant-time": {
      "version": "1.0.1",
      "resolved": "https://registry.npmjs.org/buffer-equal-constant-time/-
/buffer-equal-constant-time-1.0.1.tgz",
      "integrity": "sha512-
zRpUiDwd/xk6ADqPMATG8vc9VPrkck7T07OIx0gnjmJAnHnTVXNQG3vfvWNuiZIkwu9KrKdA1iJKfsfTV
xE6NA=="
    },
    "node_modules/busboy": {
      "version": "1.6.0",
      "resolved": "https://registry.npmjs.org/busboy/-/busboy-1.6.0.tgz",
      "integrity": "sha512-
8SFQbg/0hQ9xy3UNTB0YEnsNBbWfhf7RtnzpL7TkBiTBRfrQ9Fxcnz7VJsleJpyp6rVLvXiuORqjlHi5q
+PYuA==",
      "dependencies": {
        "streamsearch": "^1.1.0"
      },
      "engines": {
        "node": ">=10.16.0"
      }
    },
    "node_modules/bytes": {
      "version": "3.1.2",
      "resolved": "https://registry.npmjs.org/bytes/-/bytes-3.1.2.tgz",
      "integrity": "sha512-
/Nf7TyzTx6S3yRJObOAV7956r8cr2+Oj8AC5dt8wSP3BQAoeX58NoHyCU8P8zGkNXStjTSi6fzO6F0pBd
cYbEg==",
      "engines": {
        "node": ">= 0.8"
      }
    },
    "node_modules/call-bind": {
      "version": "1.0.2",
      "resolved": "https://registry.npmjs.org/call-bind/-/call-bind-1.0.2.tgz",
      "integrity": "sha512-
7O+FbCihrB5WGbFYesctwmTKae6rOiIzmz1icreWJ+0aA7LJfuqhEso2T9ncpcFtzMQtzXf2QGGueWJGT
YsqrA==",
      "dependencies": {
        "function-bind": "^1.1.1",
        "get-intrinsic": "^1.0.2"
```

```
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/chokidar": {
      "version": "3.5.3",
      "resolved": "https://registry.npmjs.org/chokidar/-/chokidar-3.5.3.tgz",
      "integrity": "sha512-
Dr3sfKRP6oTcjf2JmUmFJfeVMvXBdegxB0iVQ5eb2V10uFJUCAS8OByZdVAyVb8xXNz3GjjTgj9kLWsZT
qE6kw==",
      "dev": true,
      "funding": [
        {
          "type": "individual",
          "url": "https://paulmillr.com/funding/"
        }
      ],
      "dependencies": {
        "anymatch": "~3.1.2",
        "braces": "~3.0.2",
        "glob-parent": "~5.1.2",
        "is-binary-path": "~2.1.0",
        "is-glob": "~4.0.1",
        "normalize-path": "~3.0.0",
        "readdirp": "~3.6.0"
      },
      "engines": {
        "node": ">= 8.10.0"
      },
      "optionalDependencies": {
        "fsevents": "~2.3.2"
      }
    },
    "node_modules/chownr": {
      "version": "2.0.0",
      "resolved": "https://registry.npmjs.org/chownr/-/chownr-2.0.0.tgz",
      "integrity": "sha512-
bIomtDF5KGpdogkLd9VspvFzk9KfpyyGlS8YFVZl7TGPBHL5snIOnxeshwVgPteQ9b4Eydl+pVbIyE1Dc
vCWgQ==",
      "engines": {
        "node": ">=10"
      }
    },
    "node_modules/color-support": {
      "version": "1.1.3",
      "resolved": "https://registry.npmjs.org/color-support/-/color-support-
1.1.3.tgz",
      "integrity": "sha512-
qiBjkpbMLO/HL68y+lh4q0/O1MZFj2RX6X/KmMa3+gJD3z+WwI1ZzDHysvqHGS3mP6mznPckpXmw1nI9c
JjyRg==",
```

```
      "bin": {
        "color-support": "bin.js"
      }
    },
    "node_modules/concat-map": {
      "version": "0.0.1",
      "resolved": "https://registry.npmjs.org/concat-map/-/concat-map-0.0.1.tgz",
      "integrity": "sha512-
/Srv4dswyQNBfohGpz9o6Yb3Gz3SrUDqBH5rTuhGR7ahtlbYKnVxw2bCFMRljaA7EXHaXZ8wsHdodFvbk
hKmqg=="
    },
    "node_modules/console-control-strings": {
      "version": "1.1.0",
      "resolved": "https://registry.npmjs.org/console-control-strings/-/console-
control-strings-1.1.0.tgz",
      "integrity": "sha512-
ty/fTekppD2fIwRvnZAVdeOiGd1c7YXEixbgJTNzqcxJWKQnjJ/V1bNEEE6hygpM3WjwHFUVK6HTjWSzV
4a8sQ=="
    },
    "node_modules/content-disposition": {
      "version": "0.5.4",
      "resolved": "https://registry.npmjs.org/content-disposition/-/content-
disposition-0.5.4.tgz",
      "integrity": "sha512-
FveZTNuGw04cxlAiWbzi6zTAL/lhehaWbTtgluJh4/E95DqMwTmha3KZN1aAWA8cFIhHzMZUvLevkw5Rq
k+tSQ==",
      "dependencies": {
        "safe-buffer": "5.2.1"
      },
      "engines": {
        "node": ">= 0.6"
      }
    },
    "node_modules/content-type": {
      "version": "1.0.5",
      "resolved": "https://registry.npmjs.org/content-type/-/content-type-
1.0.5.tgz",
      "integrity": "sha512-
nTjqfcBFEipKdXCv4YDQWCfmcLZKm81ldF0pAopTvyrFGVbcR6P/VAAd5G7N+0tTr8QqiU0tFadD6FK4N
tJwOA==",
      "engines": {
        "node": ">= 0.6"
      }
    },
    "node_modules/cookie": {
      "version": "0.5.0",
      "resolved": "https://registry.npmjs.org/cookie/-/cookie-0.5.0.tgz",
      "integrity": "sha512-
YZ3GUyn/o8gfKJlnlX7g7xq4gyO6OSuhGPKaaGssGB2qgDUS0gPgtTvoyZLTt9Ab6dC4hfc9dV5arkvc/
OCmrw==",
      "engines": {
```

```
        "node": ">= 0.6"
      }
    },
    "node_modules/cookie-signature": {
      "version": "1.0.6",
      "resolved": "https://registry.npmjs.org/cookie-signature/-/cookie-
signature-1.0.6.tgz",
      "integrity": "sha512-
QADzlaHc8icV8I7vbaJXJwod9HWYp8uCqf1xa4OfNu1T7JVxQIrUgOWtHdNDtPiywmFbiS12VjotIXLrK
M3orQ=="
    },
    "node_modules/core-util-is": {
      "version": "1.0.3",
      "resolved": "https://registry.npmjs.org/core-util-is/-/core-util-is-
1.0.3.tgz",
      "integrity": "sha512-
ZQBvi1DcpJ4GDqanjucZ2Hj3wEO5pZDS89BWbkcrvdxksJorwUDDZamX9ldFkp9aw2lmBDLgkObEA4DWN
J9FYQ=="
    },
    "node_modules/cors": {
      "version": "2.8.5",
      "resolved": "https://registry.npmjs.org/cors/-/cors-2.8.5.tgz",
      "integrity": "sha512-
KIHbLJqu73RGr/hnbrO9uBeixNGuvSQjul/jdFvS/KFSIH1hWVd1ng7zOHx+YrEfInLG7q4n6GHQ9cDtx
v/P6g==",
      "dependencies": {
        "object-assign": "^4",
        "vary": "^1"
      },
      "engines": {
        "node": ">= 0.10"
      }
    },
    "node_modules/debug": {
      "version": "2.6.9",
      "resolved": "https://registry.npmjs.org/debug/-/debug-2.6.9.tgz",
      "integrity": "sha512-
bC7ElrdJaJnPbAP+1EotYvqZsb3ecl5wi6Bfi6BJTUcNowp6cvspg0jXznRTKDjm/E7AdgFBVeAPVMNcK
GsHMA==",
      "dependencies": {
        "ms": "2.0.0"
      }
    },
    "node_modules/delegates": {
      "version": "1.0.0",
      "resolved": "https://registry.npmjs.org/delegates/-/delegates-1.0.0.tgz",
      "integrity": "sha512-
bd2L678uiWATM6m5Z1VzNCErI3jiGzt6HGY8OVICs40JQq/HALfbyNJmp0UDakEY4pMMaN0Ly5om/B1VI
/+xfQ=="
    },
    "node_modules/denque": {
```

```
      "version": "2.1.0",
      "resolved": "https://registry.npmjs.org/denque/-/denque-2.1.0.tgz",
      "integrity": "sha512-
HVQE3AAb/pxF8fQAoiqpvg9i3evqug3hoiwakOyZAwJm+6vZehbkYXZ0l4JxS+I3QxM97v5aaRNhj8v5o
Bhekw==",
      "engines": {
        "node": ">=0.10"
      }
    },
    "node_modules/depd": {
      "version": "2.0.0",
      "resolved": "https://registry.npmjs.org/depd/-/depd-2.0.0.tgz",
      "integrity": "sha512-
g7nH6P6dyDioJogAAGprGpCtVImJhpPk/roCzdb3fIh61/s/nPsfR6onyMwkCAR/OlC3yBC0lESvUoQEA
ssIrw==",
      "engines": {
        "node": ">= 0.8"
      }
    },
    "node_modules/destroy": {
      "version": "1.2.0",
      "resolved": "https://registry.npmjs.org/destroy/-/destroy-1.2.0.tgz",
      "integrity": "sha512-
2sJGJTaXIIaR1w4iJSNoN0hnMY7Gpc/n8D4qSCJw8QqFWXf7cuAgnEHxBpweaVcPevC2l3KpjYCx3NypQ
QgaJg==",
      "engines": {
        "node": ">= 0.8",
        "npm": "1.2.8000 || >= 1.4.16"
      }
    },
    "node_modules/detect-libc": {
      "version": "2.0.1",
      "resolved": "https://registry.npmjs.org/detect-libc/-/detect-libc-
2.0.1.tgz",
      "integrity": "sha512-
463v3ZeIrcWtdgIg6vI6XUncguvr2TnGl4SzDXinkt9mSLpBJKXT3mW6xT3VQdDN11+WVs29pgvivTc4L
p8v+w==",
      "engines": {
        "node": ">=8"
      }
    },
    "node_modules/dotenv": {
      "version": "16.0.3",
      "resolved": "https://registry.npmjs.org/dotenv/-/dotenv-16.0.3.tgz",
      "integrity": "sha512-
7GO6HghkA5fYG9TYnNxi14/7K9f5occMlp3zXAuSxn7CKCxt9xbNWG7yF8hTCSUchlfWSe3uLmlPfigev
RItzQ==",
      "engines": {
        "node": ">=12"
      }
    },
```

```
    "node_modules/dottie": {
      "version": "2.0.3",
      "resolved": "https://registry.npmjs.org/dottie/-/dottie-2.0.3.tgz",
      "integrity": "sha512-
4liA0PuRkZWQFQjwBypdxPfZaRWiv5tkhMXY2hzsa2pNf5s7U3m9cwUchfNKe8wZQxdGPQQzO6Rm2uGe0
rvohQ=="
    },
    "node_modules/ecdsa-sig-formatter": {
      "version": "1.0.11",
      "resolved": "https://registry.npmjs.org/ecdsa-sig-formatter/-/ecdsa-sig-
formatter-1.0.11.tgz",
      "integrity": "sha512-
nagl3RYrbNv6kQkeJIpt6NJZy8twLB/2vtz6yN9Z4vRKHN4/QZJIEbqohALSgwKdnksuY3k5Addp5lg8s
VoVcQ==",
      "dependencies": {
        "safe-buffer": "^5.0.1"
      }
    },
    "node_modules/ee-first": {
      "version": "1.1.1",
      "resolved": "https://registry.npmjs.org/ee-first/-/ee-first-1.1.1.tgz",
      "integrity": "sha512-
WMwm9LhRUo+WUaRN+vRuETqG89IgZphVSNkdFgeb6sS/E4OrDIN7t48CAewSHXc6C8lefD8KKfr5vY61b
rQlow=="
    },
    "node_modules/emoji-regex": {
      "version": "8.0.0",
      "resolved": "https://registry.npmjs.org/emoji-regex/-/emoji-regex-
8.0.0.tgz",
      "integrity": "sha512-
MSjYzcWNOA0ewAHpz0MxpYFvwg6yjy1NG3xteoqz644VCo/RPgnr1/GGt+ic3iJTzQ8Eu3TdM14SawnVU
mGE6A=="
    },
    "node_modules/encodeurl": {
      "version": "1.0.2",
      "resolved": "https://registry.npmjs.org/encodeurl/-/encodeurl-1.0.2.tgz",
      "integrity": "sha512-
TPJXq8JqFaVYm2CWmPvnP2Iyo4ZSM7/QKcSmuMLDObfpH5fi7RUGmd/rTDf+rut/saiDiQEeVTNgAmJEd
AOx0w==",
      "engines": {
        "node": ">= 0.8"
      }
    },
    "node_modules/escape-html": {
      "version": "1.0.3",
      "resolved": "https://registry.npmjs.org/escape-html/-/escape-html-
1.0.3.tgz",
      "integrity": "sha512-
NiSupZ4OeuGwr68lGIeym/ksIZMJodUGOSCZ/FSnTxcrekbvqrgdUxlJOMpijaKZVjAJrWrGs/6Jy8OMu
yj9ow=="
    },
```

ДОДАТОК В.
ФОТО БАЗИ ДАННИХ

На рисунку В.1 зображено основна сторінка БД на якій представлені всі наявні таблиці.

Таблиці даної БД представлені на рисунках В.2 – В.5.

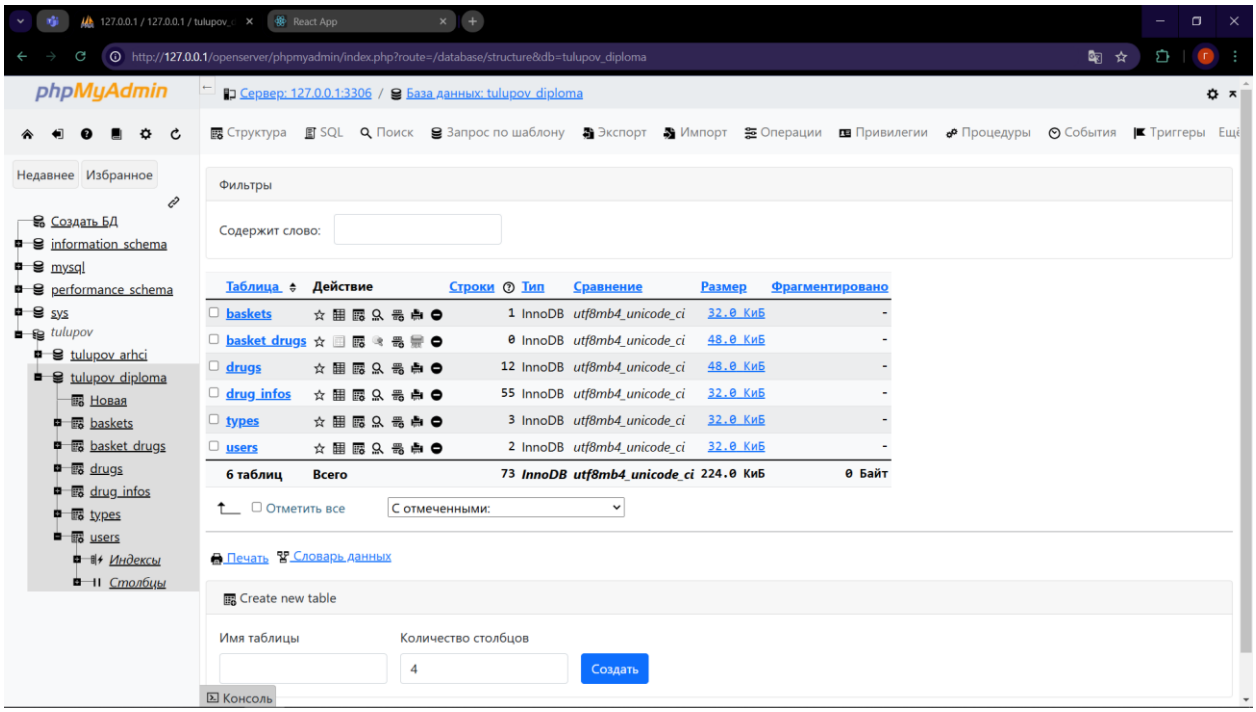


Рис. В.1 Головна сторіна БД

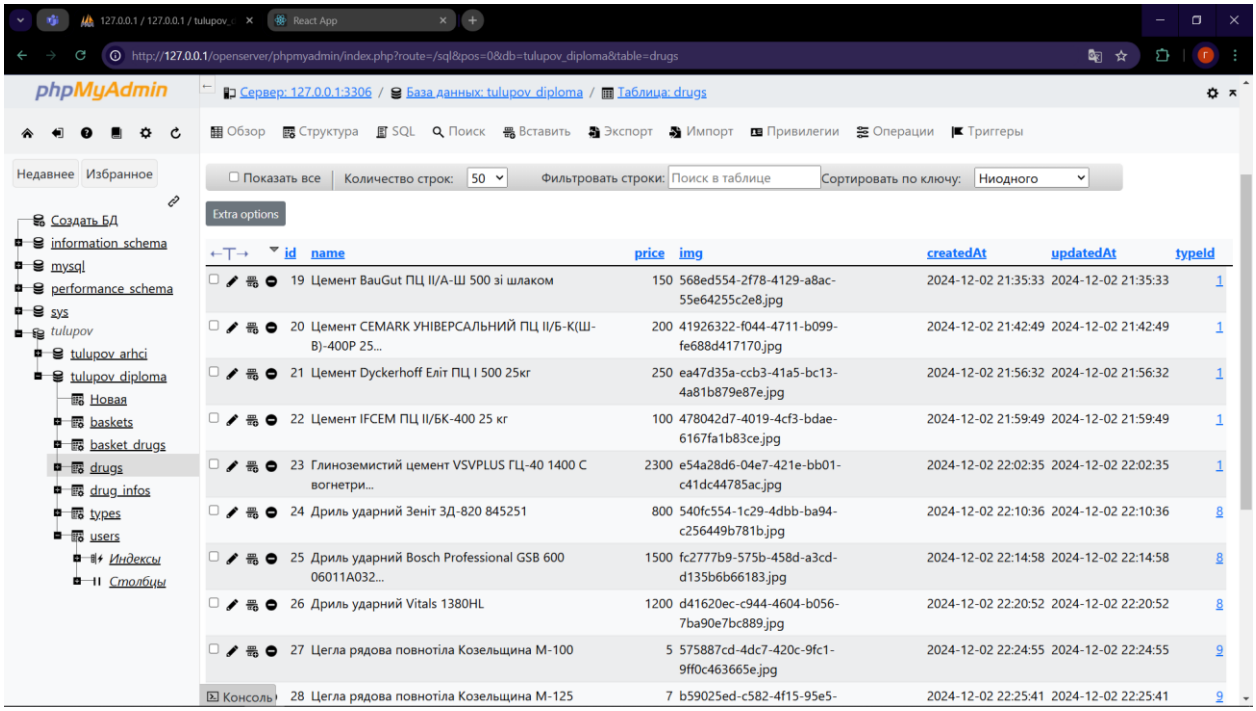


Рис. В.2 Таблиця «drugs»

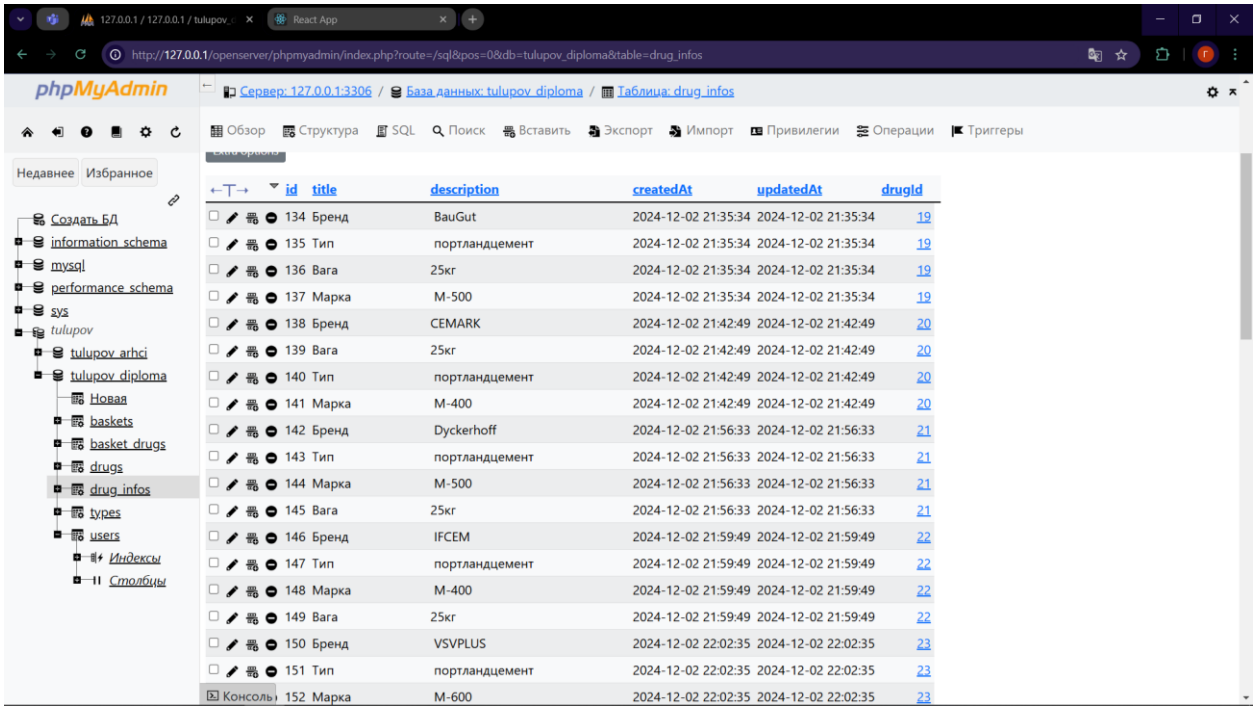


Рис. В.3 Таблиця «drug_infos»

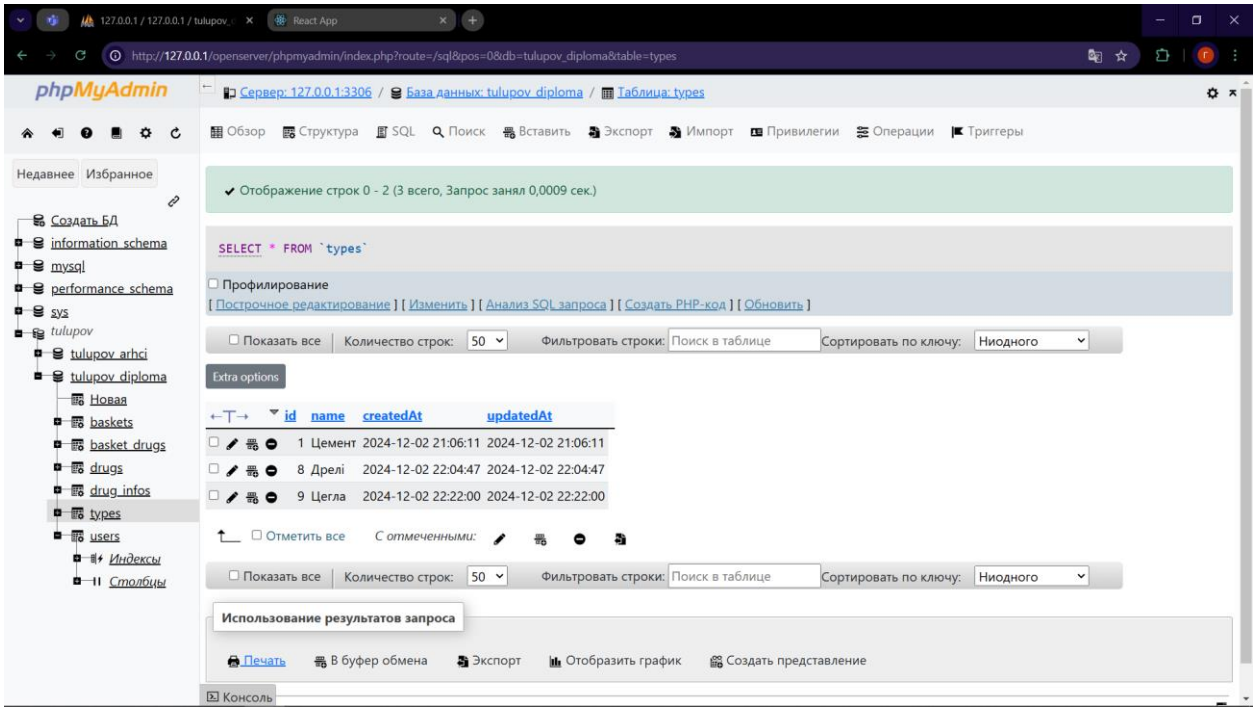


Рис. В.4 Таблиця «types»

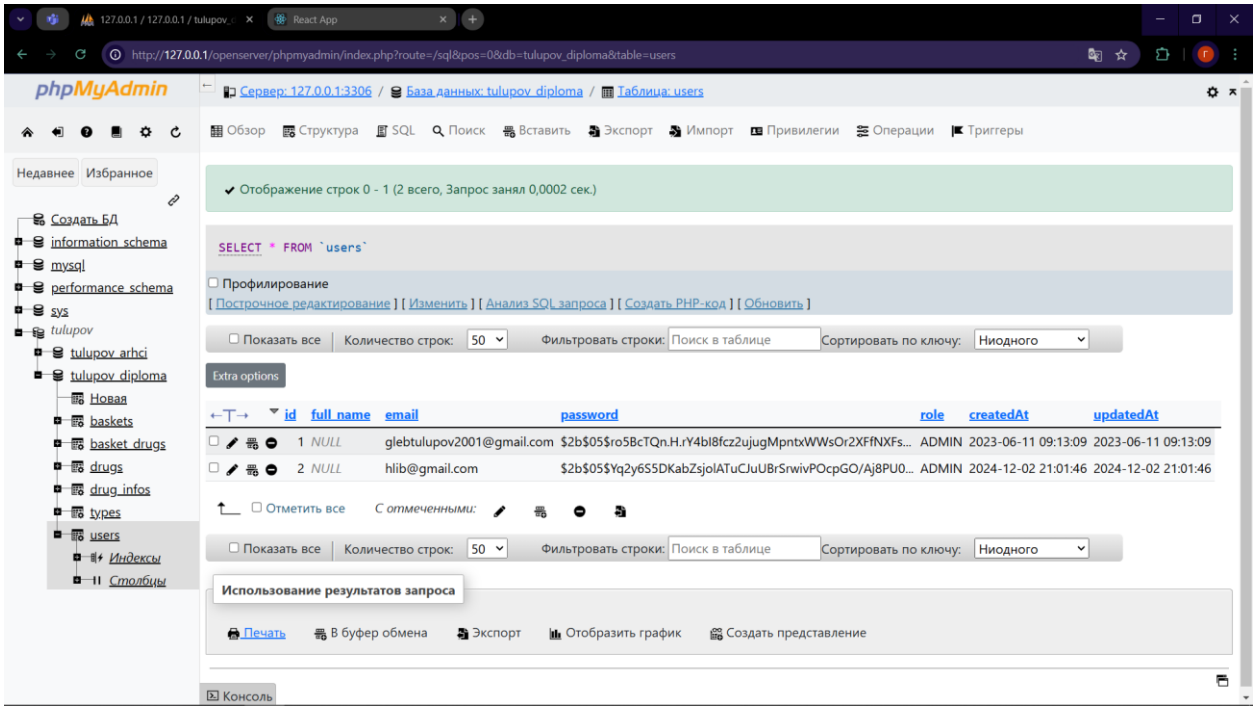


Рис. В.5 Таблиця «users»

ДОДАТОК Г.
СЛАЙДИ ПРЕЗЕНТАЦІЇ

Слайди презенції представлені на рисунках Г.1 – Г.15.

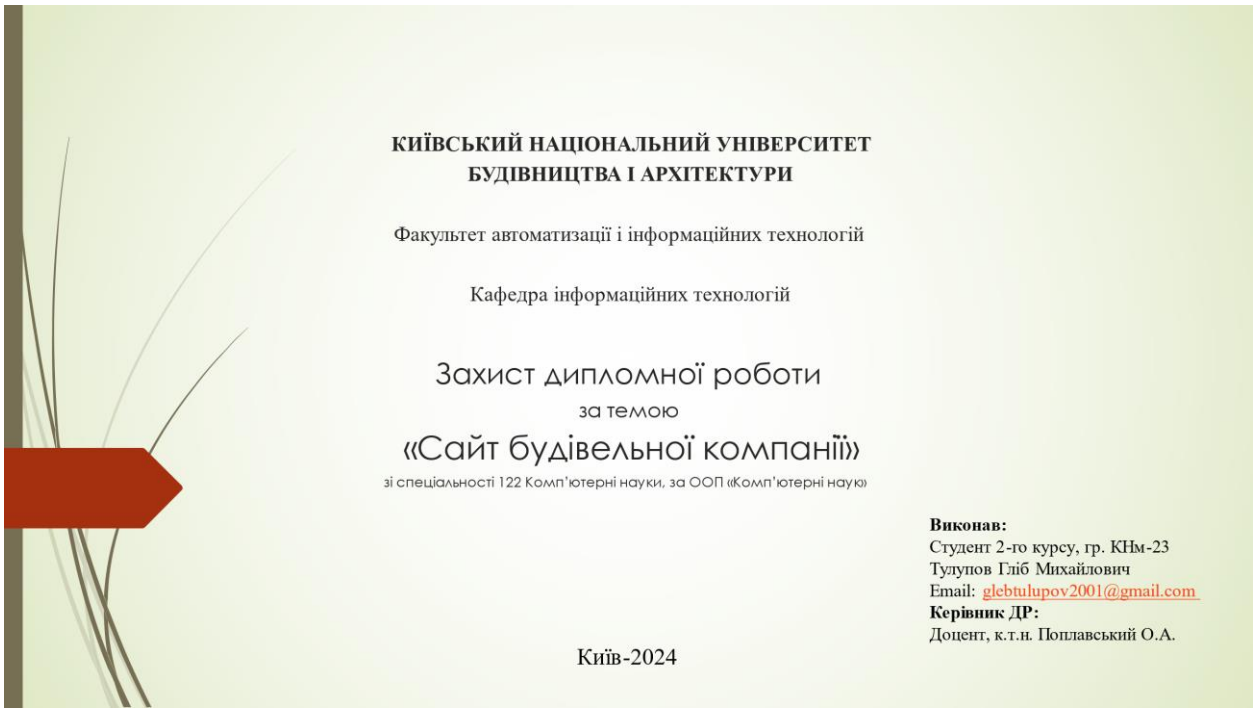


Рис. Г.1 Перший слайд

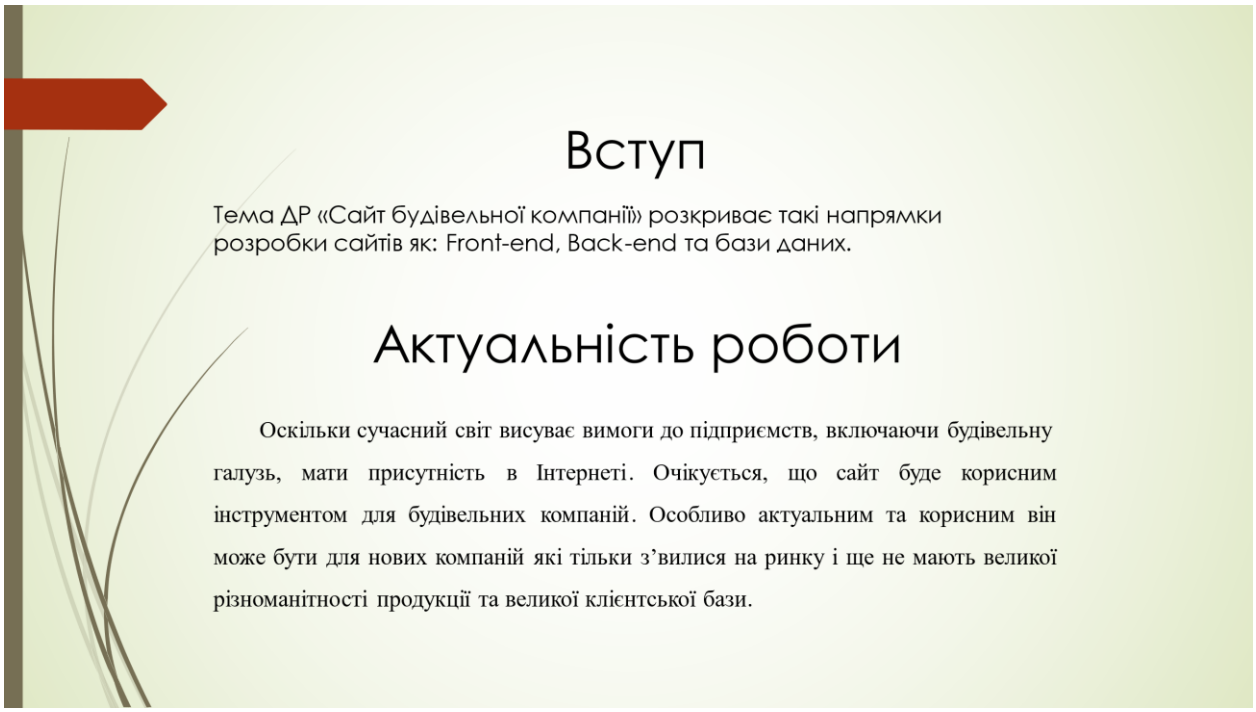


Рис. Г.2 Другий слайд

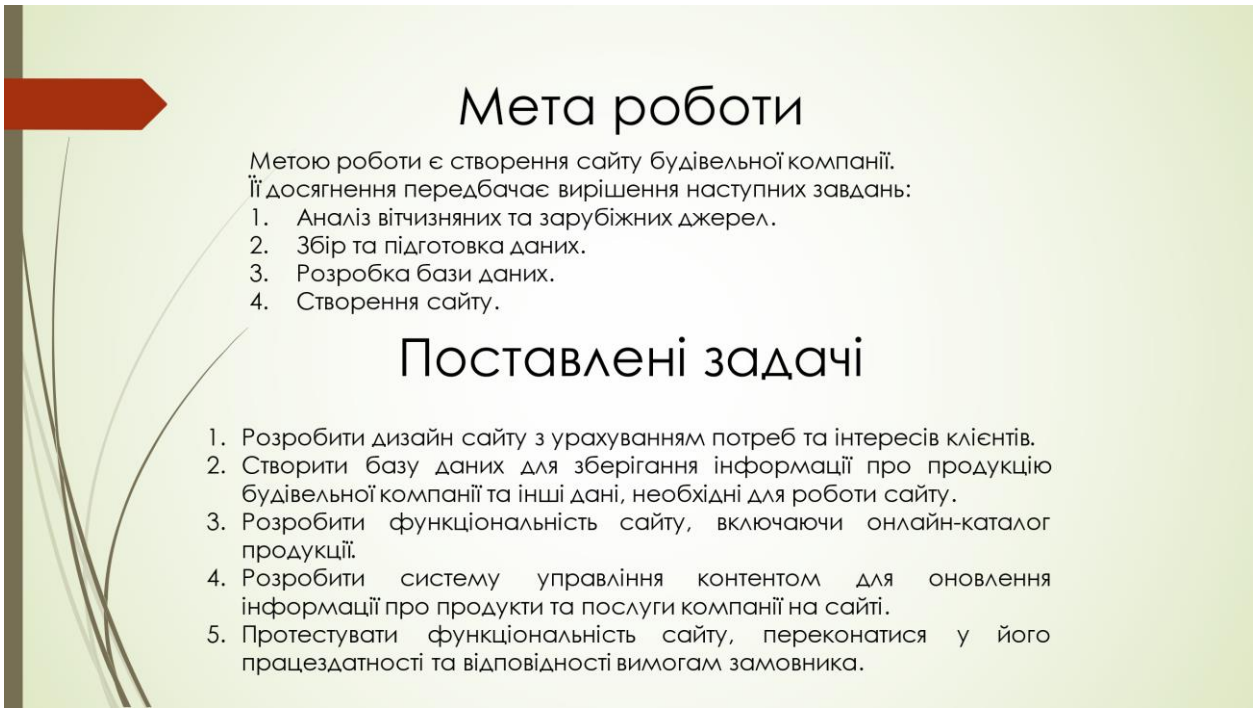


Рис. Г.3 Третій слайд

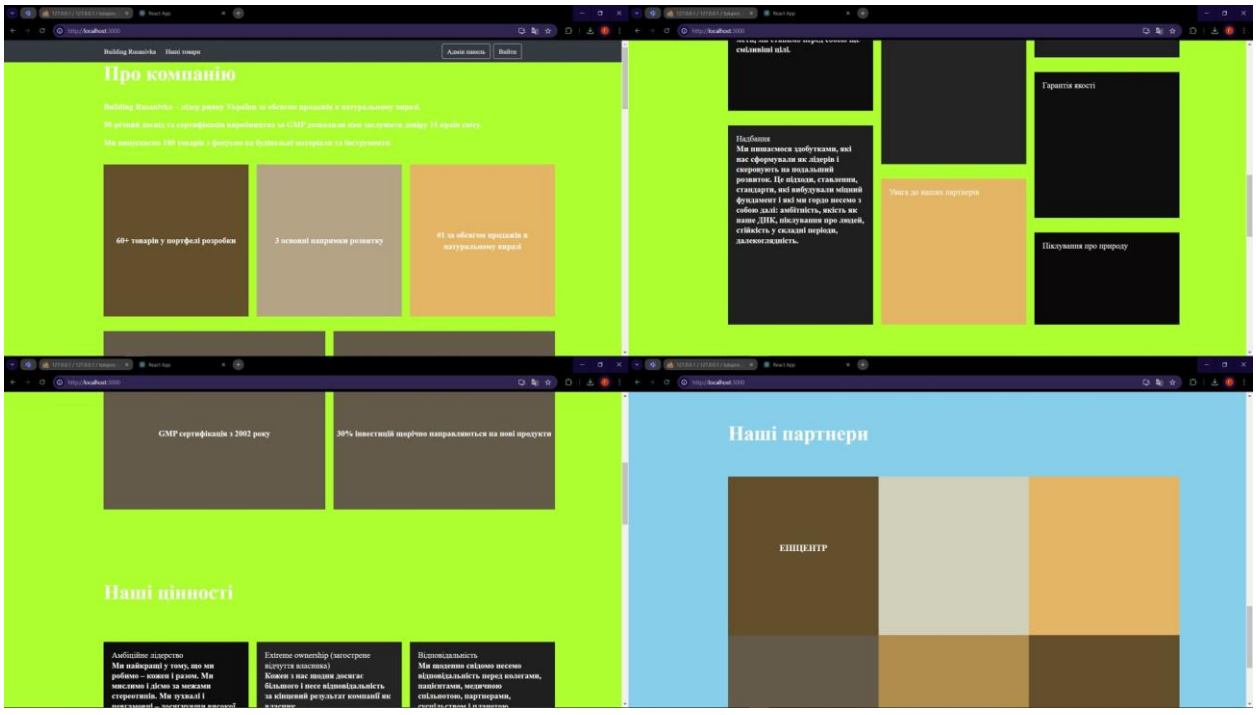


Рис. Г.4 Четвертий слайд, представлення головної сторінки

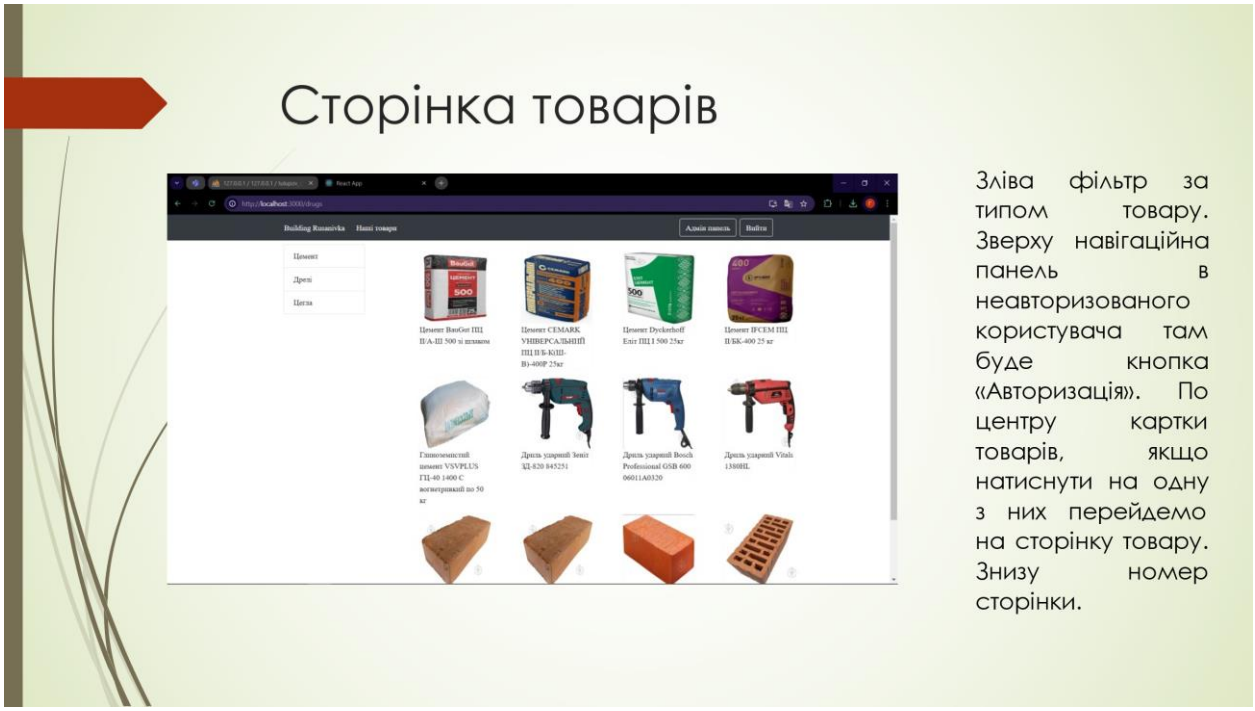


Рис. Г.5 П’ятий слайд

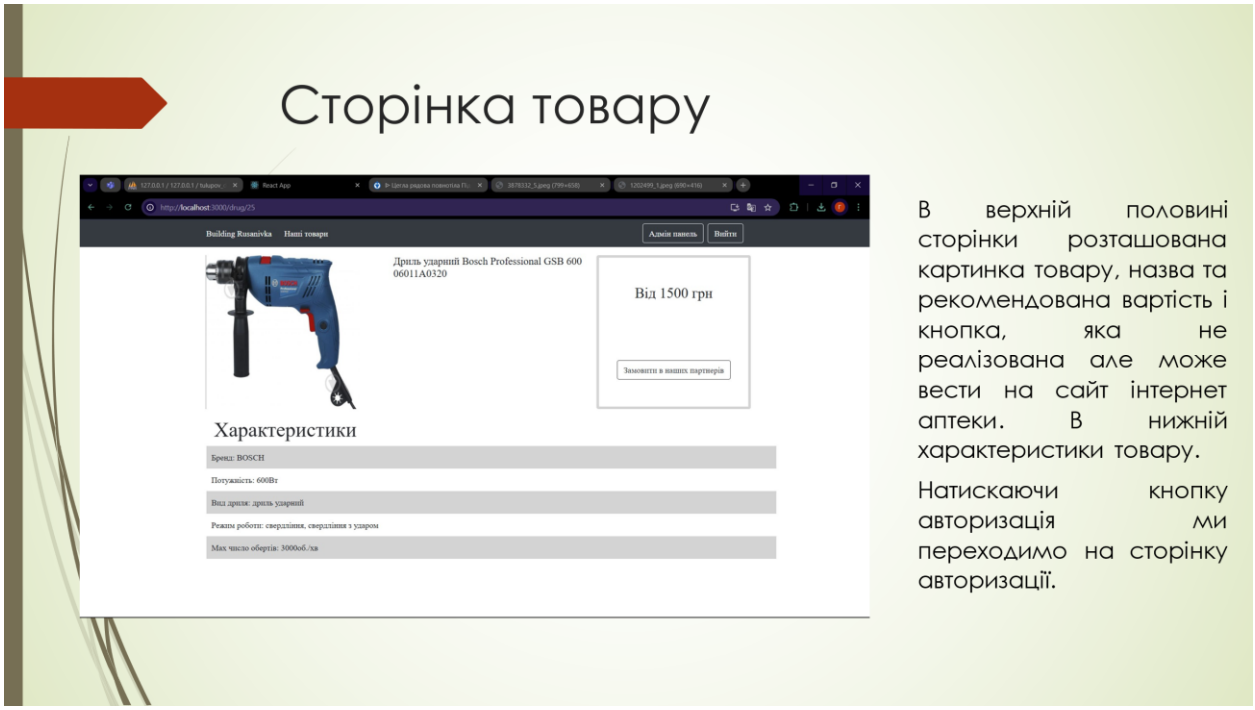


Рис. Г.6 Шостий слайд

На рисунку Г.7 зображено сьомий слайд презентації на якому зображено сторінки авторизації на сайті, адмін панель, модальні вікна додавання типу товару та товару.

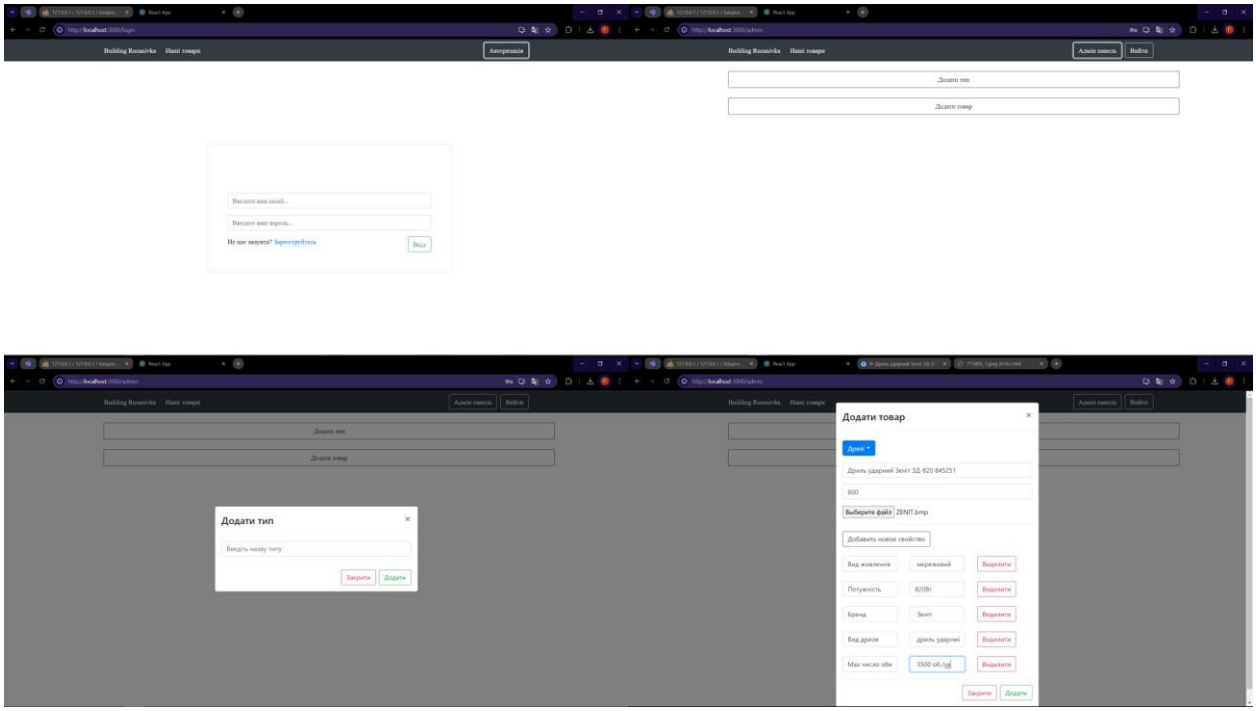


Рис. Г.7 Сьомий слайд

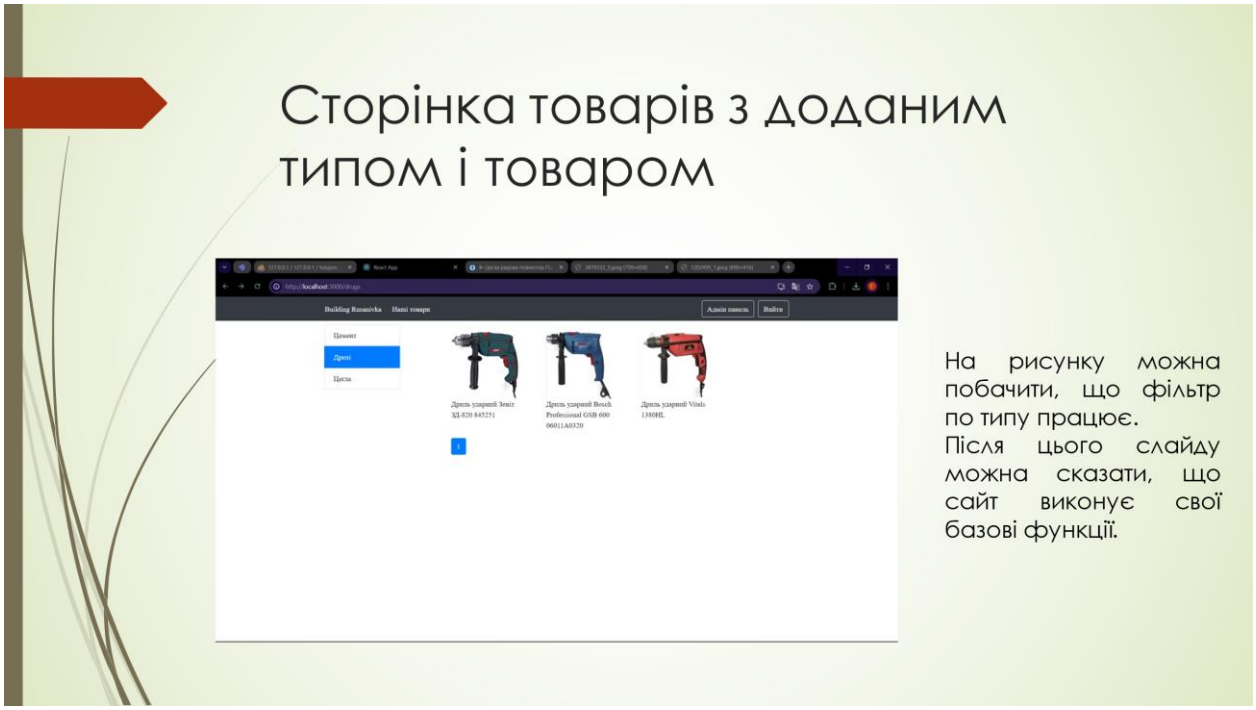
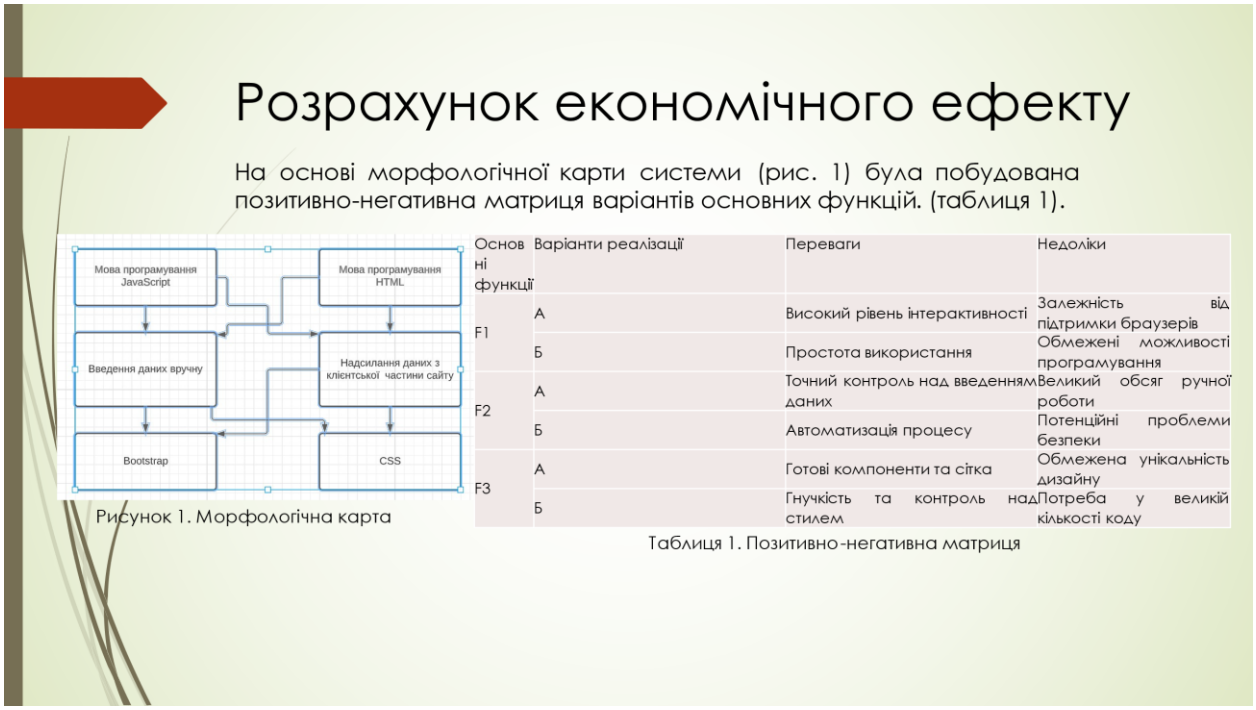


Рис. Г.8 Восьмий слайд

Рис. Г.9 Дев’ятий слайд

Рис. Г.10 Десятий слайд

Рис. Г.11 Одинадцятий слайд



Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	А	Високий рівень інтерактивності	Залежність від підтримки браузерів
	Б	Простота використання	Обмежені можливості програмування
F2	А	Точний контроль над введенням даних	Великий обсяг ручної роботи
	Б	Автоматизація процесу	Потенційні проблеми безпеки
F3	А	Готові компоненти та сітка	Обмежена унікальність дизайну
	Б	Гнучкість та контроль над стилем	Потреба у великій кількості коду

Таблиця 1. Позитивно-негативна матриця

Рис. Г.12 Дванадцятий слайд

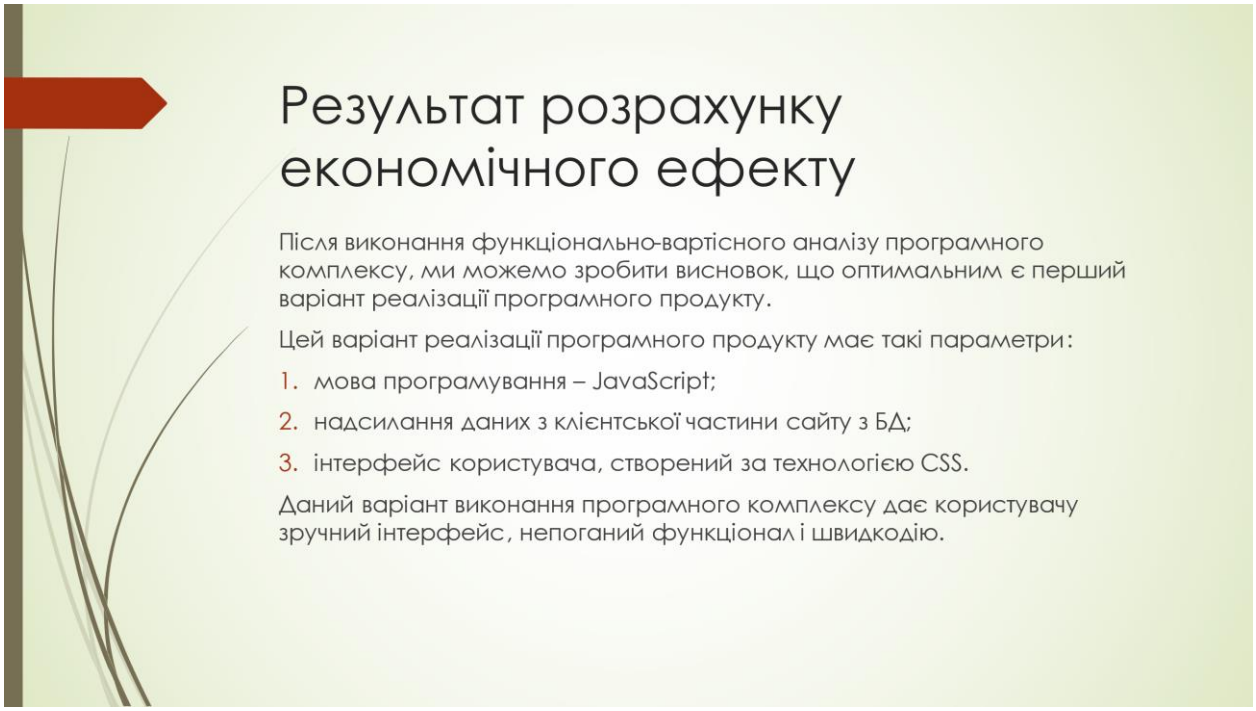


Рис. Г.13 Тринадцятий слайд

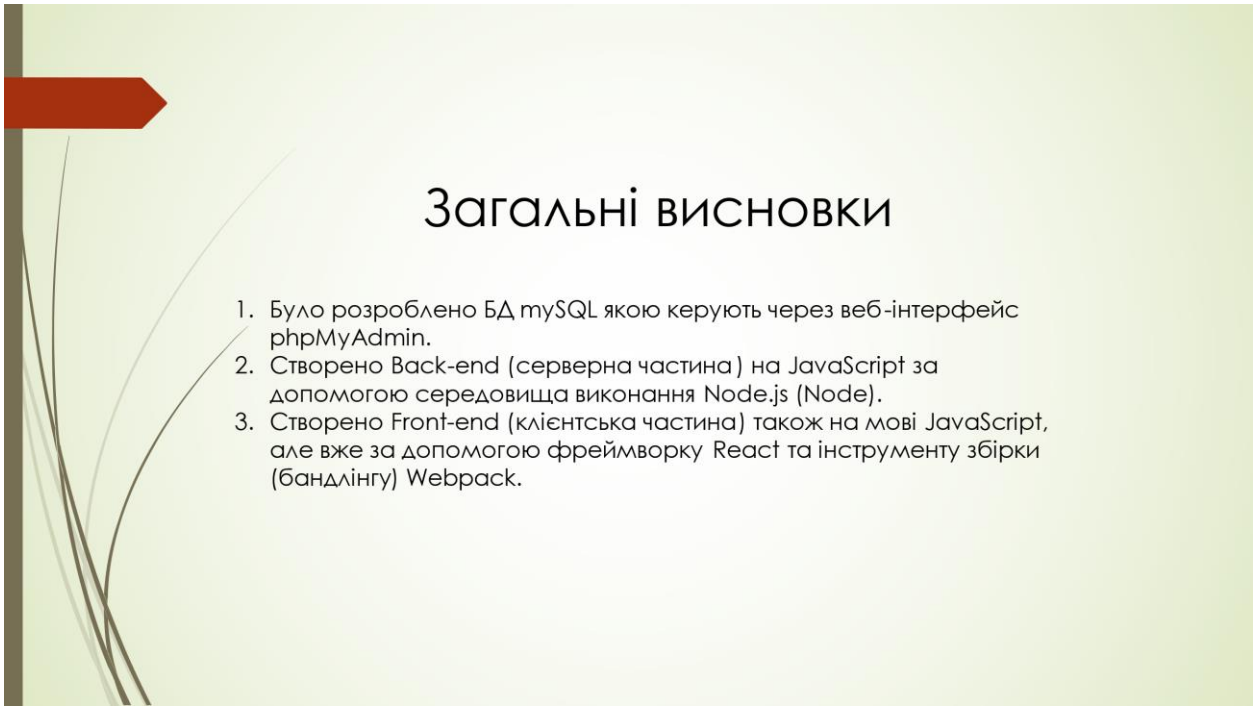


Рис. Г.14 Чотирнадцятий слайд

Рис. Г.15 П’ятнадцятий слайд