МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
18 грудня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційна технологія проектування сайту для відділу збуту промислового підприємства»
здобувача групи ІН.м-23 Мальцев Владислав Олександрович

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Владислав МАЛЬЦЕВ
(підпис)

Керівник,
Професор кафедри комп'ютерних
наук, д.т.н., професор Анатолій ДОВБИШ _____
(підпис)

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН.м-23 Мальцев Владислав Олександрович

1. Тема роботи: «Інформаційна технологія проектування сайту для відділу збуту промислового підприємства»
затверджую наказом по СумДУ від «06» грудня 2023 р. № 1412-VI
2. Термін здачі здобувачем кваліфікаційної роботи до 18 грудня 2023 року
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз проблеми та формування завдань в області дослідження. 2) Огляд технологій, що використовуються для проектування сайтів. 3) Розробка сайту для промислового підприємства. 4) Аналіз виконаної роботи
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «06» листопада 2023 р.

Завдання прийняв до виконання _____ (підпис) Керівник _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми та формування завдань в області дослідження*		
2	*Огляд технологій, що використовуються для проектування сайтів*		
3	*Розробка сайту для промислового підприємства*		
4	*Аналіз виконаної роботи*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ (підпис) Керівник _____ (підпис)

АНОТАЦІЯ

Записка: 64 стор., 35 рис., 1 додаток, 22 джерел.

Обґрунтування актуальності теми роботи – Актуальність обраної теми обумовлена різким зростанням значення віртуальної присутності для промислових підприємств у сучасному бізнес-середовищі. Завдяки інформаційній технології проектування сайту для відділу збуту промислового підприємства можна вирішити важливі завдання, спрямовані на покращення взаємодії з клієнтами, оптимізацію процесів продажу та підвищення конкурентоспроможності.

Об’єкт дослідження — процес аналізу технічних рішень та стратегій для проектування та розробки вебсайту.

Мета роботи — проектування та розробка сайту для відділу збуту промислового підприємства.

Методи дослідження — методи системного аналізу структури, функціоналу та взаємозв'язків даних відділу збуту промислового підприємства.

Результати — спроектовано та розроблено сайт для відділу збуту промислового підприємства, на якому є можливість переглянути та замовити товар.

ПРОТОТИПУВАННЯ, MONGODB, EXPRESSJS, REACT, NODEJS.

ЗМІСТ

ВСТУП

Актуальність. В світі стрімкого розвитку цифрових технологій, де керівництво промислових підприємств шукає ефективні та інноваційні рішення для оптимізації процесів, інформаційні технології стають визначальним фактором. Одним з ключових напрямків цього формату є розробка веб-сайтів для відділу збуту промислового підприємства, що дозволяє підвищити ефективність, спростити взаємодію з клієнтами та забезпечити більшу прозорість у веденні бізнесу.

Об’єкт дослідження. Процес аналіз технічних рішень та стратегій для проектування та розробки вебсайту.

Предмет дослідження. Високоефективний та надійний інструмент для відділу збуту, який відповідає сучасним вимогам та сприяє подальшому розвитку промислового підприємства.

Гіпотеза. Однією з найважливіших складових успішності цього процесу є інструмент управління інформацією, тобто веб-сайт. Сучасний сайт для відділу збуту повинен відповідати високим вимогам до зручності використання, швидкості завантаження, а також забезпечувати безпеку обміну даними.

Структура. Ця робота спрямована на ретельний аналіз інформаційних технологій проектування сайтів для відділу збуту промислового підприємства. В рамках дослідження буде приділено увагу не лише естетичним та функціональним аспектам, але й питанням безпеки передачі даних та оптимізації роботи інформаційної системи.

1. ІНФОРМАЦІЙНИЙ ОГЛЯД ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ САЙТІВ

1.1 Актуальність тематики роботи

У віці стрімкого технологічного прогресу та динамічного розвитку інтернет-середовища, технології проектування веб-сайтів стають не лише невід'ємною частиною сучасної онлайн-присутності, але й складноструктурованими системами, які вимагають глибокого розуміння різноманітних аспектів програмування, дизайну та оптимізації.

Цей інформаційний огляд зосереджений на технічних аспектах проектування веб-сайтів, включаючи важливі елементи, такі як вибір технологічного стеку, архітектура баз даних, використання фреймворків та бібліотек, а також оптимізація продуктивності та безпеки.

Мета даного дослідження - глибокий аналіз сучасних тенденцій у веб-розробці, ідентифікація оптимальних практик та висвітлення перспектив розвитку технологій проектування веб-сайтів в контексті вимог сучасного інтернет-співтовариства.

Самі веб-технології проектування та розробки сайтів є ключовими в розвитку інформаційних технологій вже велику кількість років. Вперше можливість перегляду веб-сайтів з’явилась 6 серпня 1991 року, коли був створений перший веб-сайт. Цей веб-сайт складався з простих веб-сторінок, що насправді визначали собою початок всесвітньої павутини — World Wide Web. Починаючи з того часу можна було спостерігати еволюцію вебу, появу та смерть новітніх технологій веб-розробки.

Важливим аспектом технічного проектування веб-сайтів є вибір правильного технологічного стеку, що відповідає конкретним вимогам та завданням проекту. Дослідження різноманітних мов програмування, фреймворків та платформ дозволяє визначити оптимальний підхід до розробки, забезпечуючи ефективність, масштабованість та легкість

обслуговування. Окрім того, вивчення архітектури баз даних та їх взаємодії з веб-додатками розкриває проблеми ефективного управління інформацією та забезпечення надійності в системі.

Цей напрямок постійно розвивається, з’являються тренди та ідеї для розробок. Зокрема, на наступний рік визначають такі тенденції [1]: мобільна адаптивність, швидкість роботи сайту, інтерактивний дизайн, персоналізований контент, використання штучного інтелекту. Підприємства, що вдало інтегрують ці тенденції на своїх веб-сайтах, зможуть отримати значну конкурентну перевагу та залучити нових клієнтів.

1.2 Огляд існуючих рішень проектування сайтів

На сьогоднішній день, з ростом конкуренції та розширенням цифрового простору, фахівці та підприємства визнають необхідність систематичного аналізу доступних інструментів та методик, щоб забезпечити ефективність та конкурентоспроможність своїх веб-проектів. У цьому контексті, даний огляд присвячений детальному розгляду різноманітних підходів до проектування веб-сайтів.

Існує багато рішень для дизайну сайтів, більшість з яких є платними. Далі, хотілося б зробити огляд на три найкращих на мою думку: web.com, Bluehost, IONOS.

Web.com пропонує звільнити від турбот щодо створення, підтримки та маркетингу веб-сайту, надаючи можливість зосередитися на розвитку бізнесу. Використовуючи платформу WordPress, вони пропонують гнучке рішення, яке вважається достатньо ефективним для будь-якого початківця чи невеликого підприємства і при цьому не вимагає авансових внесків або додаткових регулярних платежів [2]. Зокрема, ви отримуєте власного менеджера облікового запису як єдину контактну точку та можливість отримати підтримку в режимі реального часу.

Покупці мають можливість доповнювати свій веб-сайт додатковими функціями, такими як галерея, банер, який обертається чи інтернет-магазин, де можна розмістити до 50 продуктів. Клієнти можуть також звертатися до служби підтримки Web.com для отримання відповідей на свої запитання, що є стратегічною частиною їхнього підходу з самого початку.

Bluehost в свою чергу підходить більше для тих, хто хоче покращити рівень свого веб-сайту. Вони очолюють рейтинг кращих хостингів, а також мають великий досвід у цій сфері.

Вони представляють себе як, повноцінна консалтингова компанія у сфері веб-дизайну, яка намагається задовольнити всі можливі потреби, щодо міграцій сайту, PPC-маркетингу, веб-сайт менеджменту, SEO, редизайну та багато чого ще. Структура вартості їхніх послуг не є зовсім очевидною з самого початку [3].

Через це слід зателефонувати для консультації з одним із їхніх експертів, щоб визначити, які саме професійні послуги відповідають вашим потребам. Важливо врахувати, що ці послуги включають 6-місячний термін угоди з комісією за дострокове розірвання, а саме додаткову комісію, що рівна різниці між загальною сумою вже сплачених комісій і 2000 доларів США. Також слід зауважити, що ці послуги доступні лише для англомовних країн, хоча можливість розширення на інші ринки планується.

Останньою в цьому списку є компанія Ionos – вона є передовою в світі веб-хостингових компаній та представляє MyWebsite Design Service, які націлені на сайти з більшими потребами, ніж просто створення односторінкового веб-сайту [4]. Ця послуга веб-дизайну пропонує три рівні з встановленням комісії, 199–399 доларів США, залежно від тарифного плану. Проте слід зазначити, що під час випадкових розпродаж ця вартість може знижуватися. Після консультації щодо дизайну ви можете мати до семи користувальницьких сторінок, проте можливості для придбання додаткових - відсутні.

Мінімальний термін дії контракту MyWebsite становить 12 місяців, після чого ви можете зберігати свій веб-сайт за невелику плату. Найвищий рівень, ймовірно, є найкращим за співвідношенням ціна-якість, оскільки він надає необмежену кількість редагувань веб-сайту і до двох годин "коригування коду HTML/CSS" – функція, яка залишить небайдужими багатьох користувачів.

Слід зауважити позитивний факт, що Ionos включає автоматизовані перевірки безпеки та автоматичний переклад понад 60 мов за замовчуванням. Натомість відсутній більш повний перелік функцій, таких як конструктор форм або галерея.

Загальний висновок з огляду компаній, вказує на те, що кожна з них пропонує унікальні можливості та підходи для користувачів. Bluehost виділяється своєю ефективністю та високоякісним хостингом, Web.com надає комплексні професійні послуги з веб-дизайну, в той час як Ionos пропонує різноманітність тарифів та додаткові функції. Вибір сервісу залежатиме від конкретних потреб користувача, його бюджету та рівня технічної експертизи.

Необхідно також враховувати, що багато функцій та завдань, які пропонують ці компанії, можна виконати самостійно, особливо якщо у користувача є базові технічні навички. Наприклад, вибір та налаштування хостингу, а також виконання базових операцій з безпекою, можливе без залучення професійних служб. Важливо усвідомлювати, що самостійний підхід може забезпечити більший контроль над процесами та більш гнучку адаптацію до конкретних потреб клієнта.

1.3 Постановка задачі

1. Літературний огляд:

- Провести детальний аналіз наукових джерел та ключових публікацій, що стосуються інформаційної технології проектування веб-сайтів для відділу збуту на промислових підприємствах.

- Визначити актуальні тенденції у галузі веб-дизайну та функціональності сайтів для оптимізації збутових процесів.

2. Аналіз методик проектування сайту для відділу збуту:

- Розглянути різні підходи до створення веб-сайтів для відділу збуту на промислових підприємствах.
- Проаналізувати методи взаємодії з клієнтами, інструменти аналітики та можливості інтеграції з існуючими системами управління.

3. Визначення процесу проектування та управління веб-сайтом:

- Розробити детальний план дій для створення та ефективного управління веб-сайтом для відділу збуту.
- Включити етапи визначення вимог, розробки дизайну, тестування та впровадження.

4. Створення веб-сайту з використанням сучасних технологій:

- Розглянути варіанти використання сучасних веб-технологій для розробки сайту відділу збуту.
- Дослідити можливості використання електронних форм замовлень, систем онлайн-консультацій та інших інтерактивних функцій для покращення взаємодії з клієнтами.

5. Висновки та рекомендації:

- Сформулювати висновки на основі результатів дослідження ефективності веб-сайту для відділу збуту.
- Надати рекомендації для подальшого розвитку та вдосконалення інформаційної технології проектування веб-сайтів на промислових підприємствах.

2 ОСНОВНІ ПОЛОЖЕННЯ ТА ЗАГАЛЬНІ ТЕОРИТИЧНІ ВІДОМОСТІ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОЕКТУВАННЯ

2.1 Аналіз процесів проектування та створення сайтів

Для всіх власників власного бізнесу рано чи пізно постає питання масштабування процесів. Саме в таких випадках вони вдаються до створення сайту для власного підприємства.

Розробка веб-сайту є етапом, який має велике вплив на успіх бізнесу. Фахівці з веб-розробки розуміють, що будь-то це створення односторінкового лендінгу або реалізація глобального проекту, всеодно вимагає величезної уваги, витрат часу, фінансових ресурсів та взаємодії з командою кваліфікованих спеціалістів.

На допомогу приходить чимало сервісів, але більшість з них мають чималі недоліки.

- Висока вартість
- Залежність від надавача послуг
- Велика комісія для виходу
- Відсутність багатьох функцій

Для того ж, щоб самостійно розробити необхідний сайт слід розуміти порядок дій, для успішної реалізації проекту. Загалом, цей процес можна розділити на такі етапи:

1. Визначення тематики та мети проекту;
2. Розробка технічного завдання;
3. Прототипування та дизайн;
4. Верстка та програмування;
5. Заповнення контентом;
6. Тестування;
7. Введення в експлуатацію.

Пройдемося по всім етапам більш детально. Спершу слід визначити, з якою метою сайт буде використовуватися. Після чого визначити тип ресурсу, цільову аудиторію та основні вимоги. Чітке усвідомлення фінального результату дозволить структурувати проект та визначити необхідні етапи для досягнення цілей.

Усі ці аспекти слід обговорювати на початковому етапі співпраці, і для цього важливо підтримувати тісний зв'язок з клієнтом. Зазвичай замовники мають труднощі у формулюванні ідеї. І лише за допомогою активної участь у діалозі можна зробити концепцію, сформулювати основні цілі та обрати засоби для їх досягнення. А отже, тільки після взаєморозуміння з клієнтом та визначення основних пріоритетів можна переходити до наступного етапу.

Технічне завдання є офіційним документом та основою для подальших робіт. У ньому визначаються всі деталі, такі як структура сайту, вимоги до дизайну, а також функціональне та візуальне наповнення.

Технічне завдання вимагає відповідності конкретним вимогам:

1. Детальність – кожен аспект і всі етапи виконання детально прописані.
2. Чіткість – відсутність місця для суб'єктивних формулювань.
3. Зрозумілість – всі вимоги прописані мовою з використанням відповідної термінології.

Технічне завдання є практичною інструкцією, яка використовується під час розробки сайту. Перехід до основних етапів роботи можливий лише після повного узгодження всіх аспектів із замовником.

На наступному етапі формується макет, що перетворює концепції в реальний візуальний об'єкт. Це не розробка веб-інструменту, який готовий до функціонування, але є можливість переглянути його та оцінити його переваги. Дизайнери працюють над кількома ескізними варіантами, використовуючи технічне завдання як основу.

Багато людей помиляються, вважаючи, що це застосовується лише до зовнішнього оформлення проекту. Однак більшість уваги приділяється саме

правильному розташуванню елементів, враховуючи юзабіліті принципи та інші технічні особливості. Завдяки урахуванню всіх цих аспектів на етапі створення веб-сайту можна забезпечити його стабільну роботу.

Після обговорення прототипів та макетів із замовником, вносять необхідні зміни, якщо потрібно, допоки проект не буде остаточно затверджено.

Далі слідує технічна реалізація – процес, що включає інтеграцію дизайну з бекендом, що перетворює веб-сайт в інструмент з робочими функціями. Фахівці використовують свої знання з HTML, CSS, JS, а потім з’єднують з CMS. Слід зазначити, що не всі веб-сайти базуються на системі управління контентом.

Подальший етап у створенні веб-сайту включає надання послуг з програмування. Фахівець створює сайт та додає весь необхідний функціонал. У більшості випадків програмування виконується на основі CMS, таких як WordPress, але для реалізації більш складних проектів може знадобитися написання коду з нуля.

Після розробки верстки, отримуємо практично готовий інструмент, проте з порожніми розділами та сторінками. Їх необхідно наповнити відповідним контентом. Цей контент повинен відповідати стандартам оптимізації для успішного просування ресурсу в пошукових системах.

Тестування є завершальним етапом, який включає в себе проведення різноманітних перевірок для виявлення помилок, некоректного функціонування. Виявлені помилки виправляються фахівцями до повного усунення.

Ще одним важливим аспектом є розміщення сайту в інтернеті. Коли ви отримуєте зверстану версію проекту, готову до роботи, важливо усвідомити, що він не почне приносити прибутку миттєво. Для цього сайт повинен бути доступним для користувачів, тому його переносять на постійне місце проживання – хостинг. Крім того, необхідно обрати доменне ім'я для сайту.

Після розміщення сайту в інтернеті, проводиться фінальне тестування для перевірки його працездатності.

Після передачі готового проекту до замовника, виконавці проводять навчання роботі з сайтом. Це допомагає клієнту самостійно оперувати інформацією на ресурсі, збирати всю необхідну аналітику, вносити зміни та діяти самостійно.

За бажанням клієнта, співпраця з розробником може продовжитись, адже всі сервіси потребують подальшого розвитку та підтримки. Насправді тут це все залежить від самого сайту. Наприклад, сайт-візитка після розробки та публікації на хостингу не потребує подальшої уваги чи оновлення інформації. В свою чергу, інтернет-магазин повинен постійно оновлюватися та знаходитись на найкращих позиціях в пошукових системах.

2.2 Технології прототипування сайту

Перш ніж розпочати будівництво корабля чи створення автомобіля, створюють макети та креслення. За допомогою цих інструментів можна візуалізувати кінцевий результат, при цьому не витративши велику суму грошей на його розробку. Прототипи сайту відіграють таку ж важливу роль у контексті дизайну та верстки сторінок. Використання прототипу дозволяє замовнику та виконавцю спільно зрозуміти, вигляд кінцевого результату проекту.

Прототип сайту – це попередній вигляд майбутнього ресурсу в спрощеному варіанті. Прототип сторінки сайту відображає, як будуть розташовуватись основні елементи, та як саме користувач буде взаємодіяти з проектом. По своїй суті він є чернеткою, яка призначена для виправлень недоліків, змін структур та наборів функцій до того, як розробники почнуть розробляти сайт [5].

Офлайн або онлайн-прототип узгоджує питання щодо кількості та розташування блоків, допомагає прорахувати вартість роботи за різного

наповнення сайту. Клієнт, в свою чергу, може додавати або прибирати частини та функціонал в залежності від виділеного бюджету. Коли кількість коштів обмежена, то можна залишити мінімально необхідні функції.

Макет сайту дозволяє знайти рішення, яке задовольнить обидві сторони. На прототипі команда бачить, які складнощі чекатимуть під час розробки, та може запобігти їм на ранніх етапах.

Зазвичай прототипи сайтів поділяють на:

- статичні - сторінка, де видно основні блоки та функції, але з ними не можна взаємодіяти;
- динамічні – повноцінна інтерактивна модель з розміченими елементами.

Зазвичай все починається зі статичного прототипу, коли продумується ідея та створюється структура наповнення сайту. Макет сайту можна створити онлайн або в паперовому вигляді. Слід витрачати час на візуалізацію ідеї в цілому, а не на розгляд окремих деталей.

Після узгодження структури сайту із замовником можна приступати до основних елементів сторінки:

- Розташування кожного об’єкту сторінки;
- Його розміри;
- Взаємодія елементів між собою.

На цьому етапі слід не просто намалювати структуру сайту, а потрібно доповнити розміткою блоків та створити каркас сайту. Так буде простіше оцінити об’єм робіт та витрати на проект [6].

Далі йде деталізація прототипу. На сайті з’являються текст та ілюстрації. Поки що на сайті немає переходів між сторінками, але зрозуміло, яким саме контентом наповнена кожна зі сторінок. Контент повинен бути готовим до моменту, коли почнеться розставляння кольорових акцентів та створення настрою сайту.

Далі вже йде створення динамічних прототипів сайту. За їх рахунок легше визначитися, як буде працювати структура, чи зручно розташовані функціональні елементи, як структура сайту відповідає потребам аудиторії. Інтерактивний прототип сайту потребує більше зусиль і часу, але й результат буде значно кращим.

На цьому етапі ескіз сайту стає діючою інтерактивною моделлю. За допомогою якої можна перевірити, як саме елементи сторінки взаємодіють один з одним, а також як працює анімація та відео. Можна відкрити сторінку в браузері та переглянути, її відображення на робочому столі та телефоні, як саме змінюються ті чи інші блоки та як змінюється їх розташування. Сайт з таким прототипом простіше тестувати і приймати в роботу[7].

Які ж існують програми та сервіси для створення прототипів сайтів? Якщо це питання виникає вперше, то потрібно ознайомитись з інструментами для прототипування. Тут є два варіанти:

1. професійні програми для по типу Adobe Photoshop, InDesign, та Sketch;
2. онлайн-інструменти.

За допомогою професійного програмного забезпеченні можна реалізувати всі ідеї, додати елементи інтерфейсу та виконати переходи. Такий прототип виглядає більш якісно та добротно, відображаючи вашу майстерність. Однак сама програма зазвичай вимагає витрат коштів та часу для навчання. Якщо вам не потрібно створювати прототипи щодня, то мабуть не варто вкладати в це свій час та гроші.

Онлайн-сервіси для прототипування веб-сайтів спрощують процес для неспеціалістів. Їх зрозумілий інтерфейс та базовий функціонал у пробній версії розраховані на користувачів без спеціальних навичок та не потребують великого об’єму пам’яті на комп’ютері. З їх допомогою набагато швидше та простіше створювати прототипи онлайн. Окремо хотілося би розповісти про деякі популярні інструменти для прототипування веб-сайтів.

Wireframe – це сайт який заміняє листок А4. По своїй суті він є сіткою в розмір сторінки, на якій мишкою обирається область і за допомогою піктограм з меню відзначається, що повинно бути в цій області: картинка, текст чи список. Найбільш зрозумілий інтерфейс з іконками сутностей. Їм можна обирати різні забарвлення з відтінками сірого та червоного кольору [8].

Також у Wireframe є відображення колонок з розміткою сторінки, можна змінювати масштаб сітки, а також налаштувати розміри листка. Вся ця функціональність доступна на сайті без реєстрації. Є можливість будувати базову модель сторінки.

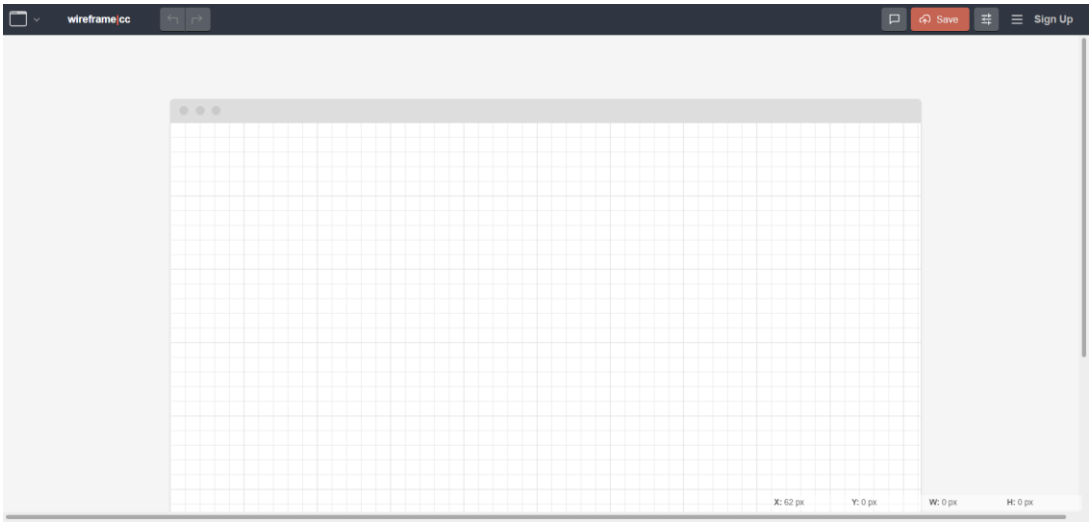

Рисунок 2.1 – Вигляд робочого простору Wireframe

Якщо ж потрібно зберегти ескіз в PDF-форматі або зробити інтерактивний прототип, то слід зареєструватися та обрати тарифний план. Шаблони також з’являються лише після реєстрації. Чим дорожчий пакет, тим більше користувачів можуть працювати над одним проєктом. Також передбачений пробний період на сім днів.

LucidChart — це онлайн-інструмент для прототипування, якому є можливість безкоштовно зареєструватися та використовувати базовий набір функцій. Під час реєстрації оберіть з запропонованих завдань ті, які хочете вирішувати. Враховуючи відповіді сервіс створить шаблони. Хоч інтерфейс

програми і простий, але передбачений підхід до кожного клієнта. Буде запропоновано подивитись коротке відео про те, як оперувати програмою [9].

За допомогою даного сервісу можна створити логічну схему роботи сайту, а також знайти місце для нього в ланцюжку бізнес-процесів. Для виконання цих завдань є такі функції:

- Оргструктури - показують глибину розділів на сайті;
- BPMN-діаграми - знаходять місця для створення сайту в бізнес-процесах замовника, а також візуалізують процеси роботи з макетом;
- Gantt-чарт - відстежує хід розробки сайту від початку до дистрибуції.

Наступний застосунок підійде для активних користувачів Google, GitHub або Dropbox, цей сервіс називається Draw.io. За концепцією він схожий на LucidChart, але він також встановлюється на десктоп чи прив'язується до хмарного сховища.

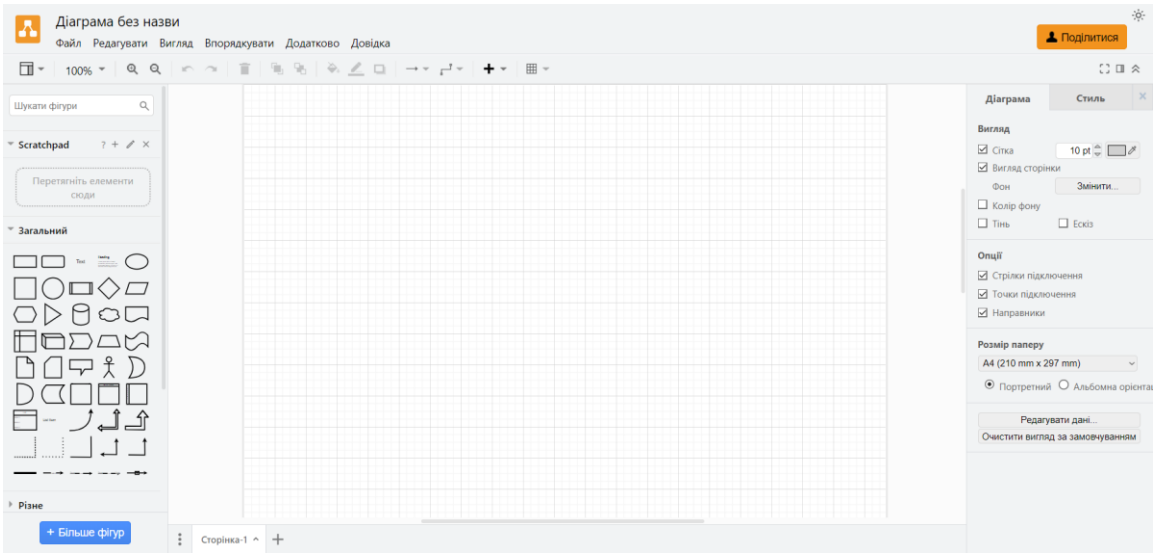

Рисунок 2.2 – Робочий простір Draw.io

В ньому представлені всі ті ж самі можливості та функції що й у LucidChart, але в інших кольорах інтерфейсу. Щоб почати роботу необхідно увійти за допомогою Google-аккаунту. Також є можливість редагувати прототип разом та зберігати результати на гугл-диску [10].

Ще один з варіантів для виконання прототипування про який хотілося б розповісти називається Figma. З його допомогою можна створити цілісний інтерактивний прототип сайту. В застосунку представлені моделі для комп’ютера, планшета та телефону. Також можна пов’язувати сторінки та додавати анімацію.

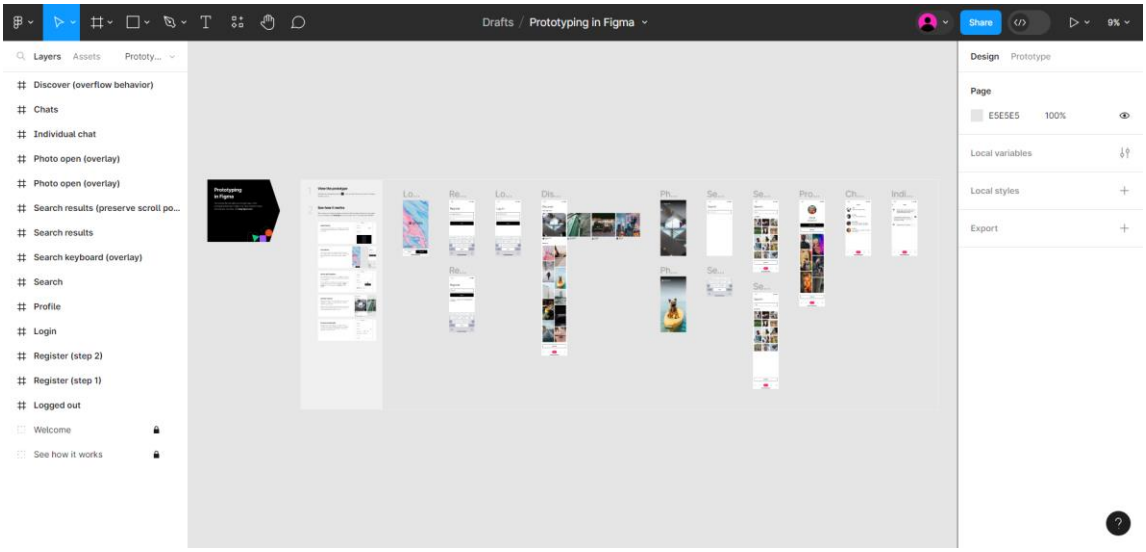

Рисунок 2.3 – Шаблон прототипування мобільного застосунку в Figma

Якщо ви не знаєте, як розробити прототип за допомогою цього способу, то в програмі передбачені готові шаблони та підказки для користувача щодо налаштування. Безкоштовна версія надає можливість зберігання створеного макету в PDF-форматі або як картинку в SVG, PNG чи JPG. Доступ до програми можна отримати онлайн за допомогою сайту або встановивши застосунок на комп'ютер [11].

Підсумовуючи варто зазначити, що Wireframe є зручним та легким у використанні інструментом для створення базових макетів без реєстрації. LucidChart, відзначається можливістю створення логічних схем та діаграм бізнес-процесів, що робить його корисним для визначення ролі сайту в бізнес-ланцюгу. Draw.io, можливість роботи з використанням Google-аккаунту. Figma ж виділяється серед інших завдяки можливості створення інтерактивних прототипів, готовим шаблонам та різноманітним опціям збереження.

Враховуючи їх особливості, кожен інструмент може відповідати конкретним потребам та вимогам користувачів, спрощуючи процес створення та вдосконалення веб-сайтів.

2.3 Стеки технологій для веб-розробки

На етапі розробки необхідно зрозуміти за допомогою якого стеку технологій буде виконуватись проект. На сьогоднішній день їх існує чимало видів і кожен підходить під різні задачі та можливості.

Стеки веб-розробки - це комбінації інструментів і технологій, які використовуються для створення веб-сайту або веб-програми. Їх також називають стеками рішень [12]. Їх створюють за допомогою різних мов програмування, фреймворків, бібліотек, серверів та програмного забезпечення. Хоча веб-розробники можуть вільно створювати стеки на основі власних потреб, деякі технологічні стеки настільки добре поєднуються, що стали стандартами в індустрії веб-розробки.

Стеки веб-розробки зазвичай використовуються неодноразово однією командою, це дозволяє їм працювати ефективніше, швидше усувати помилки та прискорювати процес розробки. З цього випливає чимало переваг використання стеків рішень:

- Підвищується ефективність та продуктивність розробника за рахунок оптимізації процесу розробки та зменшення часу, необхідного для налаштування.
- Розробники можуть приділити більше уваги вдосконаленню свого коду та створенню нових функцій, а не проблемам інтеграції чи винаходу колеса.
- Стандартизований підхід до розробки, який забезпечує послідовність розробки всього проекту та має заготовлені вказівки, щодо найкращих практик для архітектури та розгортання.
- Повторне використання коду та зручність обслуговування.

- За рахунок того, що популярні стеки підтримуються активними спільнотами розробників, до них часто можна знайти обширну документацію та навчальні посібники.
- Масштабованість, оскільки стеки допомагають обробляти більше трафіку, обсяги даних зростають в рази, так само як і взаємодія користувачів, тому можна швидше адаптуватись до змін бізнес-вимог без будь-яких суттєвих змін архітектури.
- Сумісність між різними компонентами різних технологічних стеків, дозволяє більш плавно обмінюватись даними між компонентами.
- Компоненти всередині стеку зроблені для безперебійної взаємодії, дозволяючи розробникам використовувати оптимізацію та найкращі практики.
- Вибравши популярний і добре підтримуваний стек, розробники зменшують ризик виникнення серйозних технічних проблем.
- Наявність безкоштовних компонентів із відкритим вихідним кодом, що значно знижує витрати на ліцензування, дозволяє створювати надійні та багатофункціональні рішення без додаткових витрат.

Для того, щоб обрати необхідний саме вам стек технологій слід відповісти на такі основні питання:

1. Який об’єм проекту?
2. Які досвід в команди та які технології використовуються для розробки?
3. Які є вимоги до масштабованості?
4. Які вимоги, щодо безпеки?
5. За який час необхідно вийти на ринок?
6. Який бюджет на розробку проекту?

Після ознайомлення з цими всіма процесами роботи зі стеками слід зробити огляд по деяким з них, щоб розуміти в чому їх різниця, які в кожного переваги та функціональність.

І розпочати хотілося б зі стеку MERN. Він є одним із найбільш популярних стеків у галузі веб-розробки. Складається з чотирьох ключових технологіїй:

- MongoDB – це система управління баз даних, що використовує концепцію NoSQL. Її особливість полягає в тому, що вона дозволяє ефективно зберігати та обробляти великі обсяги структурованих та неструктурованих даних, що робить її ідеальним вибором для розробки сучасних веб-додатків.
- Express.js – це веб-фреймворк для Node.js, призначений для швидкої розробки серверних додатків. Завдяки своїй мінімалістичній структурі та ряду вбудованих функцій, Express дозволяє розробникам швидко створювати потужні та ефективні веб-сервери і API.
- React – це бібліотека розробки інтерфейсів користувача. React дозволяє розробникам будувати легко підтримувані інтерфейсні елементи, що спрощує процес створення високоякісних веб-додатків.
- Node.js – це середовище виконання JavaScript на сервері, яке дозволяє розробникам використовувати JavaScript для написання серверного коду. Забезпечуючи ефективну обробку багатьох одночасних з'єднань, Node.js дозволяє створювати швидкі та масштабовані серверні додатки.

Існують схожі стеки, такі як MEAN і MEVN. Однак саме MERN відзначається стрімким зростанням популярності в останні роки, завдяки використанню бібліотеки React.

React, як найбільш популярна бібліотека у світі, завдяки здатності спрощувати процес створення та управління інтерфейсом користувача, що призводить до значного покращення продуктивності веб-сайтів. Підтримка від Facebook і захоплена спільнота користувачів також сприяли її великій популярності.

Поєднуючи React із MongoDB, Express.js і Node.js, формується технологічний стек JavaScript, який відзначається легкістю налаштування, зручністю для розробників і економічною ефективністю. Стек MERN

особливо вдалий для реалізації платформ соціальних мереж і організації робочих процесів [13]. До його користувачів належать відомі компанії, такі як Uber та Netflix.

Наступний стек MEAN схожий на стек MERN, за винятком того, що замість React в ньому використовується Angular.

Angular – популярний інтерфейсний фреймворк, який використовується для спрощення процесу розробки та тестування. Створений програмістами з Google, Angular вважається універсальною структурою, оскільки з його допомогою усувають різноманітні проблеми, з якими стикаються розробники під час створення односторінкових програм [14].

Стек MEAN вважався вибором номер один серед веб-розробників протягом багатьох років. Його успіх значною мірою можна пояснити тим, що ці чотири технології надзвичайно добре працюють разом. Як і стек MERN, MEAN включає Node.js, який дозволяє використовувати JavaScript у внутрішній частині. Це означає, що розробникам не потрібно знати Python, Ruby або PHP, достатньо знань з JavaScript, щоб використовувати стек MEAN. Усі базові технології в стеку MEAN мають відкритий вихідний код, що значно спрощує розробку програм.

Наступний стек MEVN, містить усі елементи двох попередніх стеків, але з одним винятком - він використовує Vue.js для інтерфейсу замість того, щоб покладатися на React або Angular.

Vue.js, створений програмістом із Google як легка альтернатива Angular, є інтерфейсним фреймворком, який цінується за його простоту та адаптивність. Vue часто називають «прогресивним» фреймворком, оскільки розробники можуть легко додавати компоненти та адаптувати фреймворк відповідно до своїх потреб [15].

Хоча між стеками MEVN, MERN і MEAN є багато схожого, є кілька причин, чому розробники обирають Vue.js замість інших варіантів. Зокрема, тому що, Vue.js має адаптований дизайн, який дозволяє легко налаштовувати та інтегруватись з різними фреймворками та бібліотеками; такий рівень

налаштування та гнучкості є однією з причин, чому Vue.js став популярним вибором серед веб-розробників. Також причинами обрати саме його є те, що він потужний, легкий та простий у вивченні.

LAMP – це стек, який складається з Linux, Apache, MySQL і PHP. Він вважається одним із найкращих способів створення веб-додатків, а також є популярним серед розробників вже більше 20 років. Цей стек був одним із перших з відкритим вихідним кодом, коли-небудь доступних у всьому світі і він залишався популярним протягом десятиліть завдяки своїй репутації, простоті та стабільності.

Стек LAMP давно вважався важливим інструментом веб-розробки та використовувався на відомих платформах, таких як Shopify. Стек є фаворитом серед веб-розробників, оскільки він економічно ефективний та легко налаштовуваний. З його допомогою можна створювати високопродуктивні веб-сайти та програми. Ефективно працює в будь-якій операційній системі та легко обробляє динамічні сторінки, що дозволяє додавати новий вміст. Загалом, це доволі гнучке рішення, перевірене часом [16].

PERN – це стек такий же, як і MERN, але з однією великою відмінністю: в одному використовується база даних PostgreSQL, а в іншому MongoDB.

Postgre – це надійна система баз даних корпоративного рівня на основі SQL, яка наголошує на розширюваності та сумісності. Вперше була створена в 1996 році в Каліфорнійському університеті в Берклі і з тих пір користується величезною популярністю. Вона є другою за популярністю базою даних згідно з опитуванням від Stack Overflow у 2021 році.

Обирати PostgreSQL замість MongoDB є декілька причин. PostgreSQL має суворі правила щодо цілісності даних і сумісний з ACID, що гарантує достовірність даних. Він також дотримується монолітної архітектури, тому компоненти об’єднані та систематично працюють разом. Пропонує численні запобіжні засоби, які роблять зберігання більш надійним. Саме такі властивості роблять PostgreSQL улюбленим вибором серед медичних і банківських компаній. На відміну від MongoDB, Postgre використовує

традиційний синтаксис і формат SQL, зберігаючи дані в табличному форматі в рядках і стовпцях, а не як документи, що дозволяє робити більш складні запити [17]. Також його легше масштабувати, ніж бази даних NoSQL, тому він добре підходить для великих підприємств.

Ruby on Rails – це стек на популярному фреймворку на основі Ruby. Він дозволяє розробникам створювати цілу веб-програму, поєднуючи з HTML, CSS і JavaScript. Популярність йому приніс синтаксис, простота та швидкість розробки. Існує багато бібліотек і інструментів, які можна використовувати разом з Ruby on Rails, що дозволяє розробникам легко налаштовувати програми.

Присутні додаткові функції, які роблять стек Ruby on Rails привабливим варіантом. Безпека цього стеку вважається однією з найкращих. Він також надає докладний журнал помилок. Так як він заснований на Ruby, мові програмування високого рівня, то є зручним для розробників і простим у використанні [18]. Загалом це хорошим вибір для створення легких програм, але не найкраще рішення для більш складних проектів.

2.4 Стек MERN

Для виконання практичної частини кваліфікаційної роботи було обрано MERN. MERN - це стек інформаційних технологій, які використовуються в процесі розробки програми на основі JavaScript. Сьогодні розробники в усьому світі працюють над покращенням інтерфейсу користувача програми та покращенням процесу розробки, створення додатків для реалізації проектів і розробки вимог до проекту у встановлений термін. Оскільки MERN базується на JavaScript, розробникам потрібно знати лише одну мову програмування, що робить його в рази легше.

Фронтенд реалізовується за допомогою React.js. React використовує JSX, синтаксичне розширення JavaScript, яке надає спосіб структурувати відтворення компонентів за допомогою синтаксису, знайомого багатьом

розробникам. React використовує Virtual DOM (Document Object Model), що дозволяє легко вносити зміни.

Оскільки React — це бібліотека, а не фреймворк, розробникам, можливо, доведеться покладатися на сторонні служби для створення бажаних функцій.Дивлячись на продуктивність React, важливо пам’ятати, що, це все сценарії JavaScript.

Компонентна архітектура є однією з ключових переваг React, що сприяє багаторазовому використанню коду та створенню модульних програм.

Розбиття інтерфейсу користувача на невеликі компоненти відіграє важливу роль у структуруванні та покращенні читабельності коду. Такий підхід спрощує процес обслуговування коду та додавання нових функцій, оскільки кожен компонент має чітко визначене завдання.

Однією з головних переваг компонентної архітектури є можливість повторного використання компонентів у різних частинах програми. Це економить час розробки та зменшує ймовірність помилок, оскільки перевірені компоненти можна легко інтегрувати в різні частини проекту.

У сфері React ви можете поєднувати компоненти, щоб створювати складніші інтерфейси з простих будівельних блоків. Такий підхід дозволяє створювати гнучкі та масштабовані інтерфейси. Ви можете змішувати та поєднувати компоненти, складати їх один на одного та створювати складні структури, не турбуючись про складність взаємодії.

React розроблявся та підтримується командою інженерів з Facebook, що забезпечує переваги стабільності та надійності. Команда активно інвестує в розвиток React, забезпечуючи постійне вдосконалення фреймворку та виправлення помилок. Підтримка Facebook гарантує, що React дотримуватиметься сучасних стандартів і практик розробки, забезпечуючи його довгострокову життєздатність.

React пропонує гнучкість в інтеграції з різними технологіями та платформами. React Native – це фреймворк для створення мобільних застосунків, який дозволяє створювати міжплатформенні додатки за

допомогою JavaScript і React [19]. Розроблений Facebook, він дозволяє розробникам створювати мобільні програми, які можуть працювати як на платформах iOS, так і на Android, використовуючи спільну кодову базу.

React Native дозволяє використовувати один підхід для проектування мобільних додатків на різних платформах, обмінюючись кодом між веб-додатками та мобільними додатками. React можна використовувати для візуалізації на стороні сервера, покращуючи оптимізацію SEO та продуктивність. Фреймворк також можна інтегрувати з іншими бібліотеками та фреймворками, такими як Redux для управління станом.

Бекенд в MERN реалізовується за допомогою Node JS, Express JS та Mongo DB. Почнемо з визначення технології Node.js.

Node.js — це кросплатформенне середовище виконання JavaScript із відкритим кодом. Node.js є однією з найбільш використовуваних технологій у веб-розробці, головним чином завдяки таким ключовим перевагам, як висока продуктивність, гнучкий синтаксис, ефективний обмін кодом і регулярні оновлення бібліотеки [20]. Якщо його більш детально розглядати, то слід відзанчити такі переваги в характеристиках:

- Легкий запуск із Node.js. Це популярний варіант для новачків у веб-розробці, бо почати відносно просто, оскільки є багато інструкцій і велика спільнота.
- Масштабованість. Він забезпечує хорошу масштабованість програми. Завдяки своїй однопотоковій природі Node.js може ефективно обробляти велику кількість підключень одночасно.
- Надійний бекенд. Node.js швидкий і надає потужні можливості такі , як підтримка мереж.
- Продуктивність. Node.js створено на основі двигуна Chrome V8 JavaScript Runtime, який пришвидшує виконання коду. Завдяки двигуну V8 впровадження коду стало більш спрощеним і швидким.

- Кросплатформенність. Node.js є не тільки кросплатформенним, але також може бути написаний таким чином, щоб його можна було запакувати як виконуваний файл, що містить усі його залежності.

- Ремонтопридатний. Оскільки екосистема JavaScript може керувати як фронт-ендом, так і бек-ендом.

- Швидка потокова передача даних. Node.js відносно швидко обробляє дані. Одночасна обробка та завантаження файлу економить багато часу. Як наслідок, Node прискорює потокове передавання даних в цілому.

Node JS - це серверне середовище виконання, яке дає свободу вирішувати, чи використовувати його для зовнішніх чи внутрішніх функцій. Навіть незважаючи на те, що середовище повністю побудовано на механізмі V8 JavaScript, воно має можливості, які спрощують запуск файлів JavaScript, які навіть не запускаються у веб-браузері, що робить його придатним для внутрішнього середовища розробки.

Далі на черзі, Express.js, який іноді також називають «Express», - це мінімалістичний, швидкий серверний фреймворк Node.js, який надає надійні функції та інструменти для розробки масштабованих серверних програм. Він включає систему маршрутизації, а також спрощені функції, що сприяють розширенню інфраструктури шляхом розробки більш потужних компонентів та частин в залежності від потреб проекту.

Фреймворк уміщує в собі набір інструментів для веб-додатків, обробки HTTP-запитів і відповідей, налаштування маршрутів та використання проміжного програмного забезпечення для створення та розгортання масштабованих корпоративних додатків. Крім того, він включає інструмент інтерфейсу командного рядка, відомий як Node Package Manager, де розробники можуть завантажувати готові пакети. Це також сприяє дотриманню принципу «Не повторюйся» під час розробки [21].

Express.js використовується для багатьох речей в екосистемі JavaScript/Node.js – ви можете розробляти програми, кінцеві точки API,

системи маршрутизації та фреймворки. Далі приведений перелік для чого ще його можна використовувати:

- Односторінкові програми
- Інструменти для співпраці в реальному часі
- Потокові програми
- Fintech програми

Є кілька причин, чому слід використовувати Express.js для проектів, давайте детальніше розглянемо основні з них:

- Простота та гнучкість. Він швидший за будь-який інший фреймворк Node.js.
- Мінімалізм. Фреймворк пропонує швидку розробку додатків і полегшує напругу, пов’язану з опануванням багатьох різних частин великого фреймворку. Він також надає багато функцій, такі як чудова система маршрутизації, проміжне програмне забезпечення та узгодження вмісту прямо з коробки.
- Швидше введення-виведення. Для швидших запитів і відповідей до його однопотокової системи та асинхронних процесів.
- Використовує структуру MVC для спрощення маніпулювання даними та систем маршрутизації.
- Масштабованість. Він ефективно обробляє запити та відповіді користувачів і практично не вимагає додаткової конфігурації під час розробки великомасштабної веб-програми. Має чудові модулі, пакети та додаткові ресурси, що допомагає розробникам створювати надійні та масштабовані веб-додатки.
- Підтримка спільноти. Підтримка від Google також широка, що робить фреймворк популярним вибором серед розробників Node.js. Його природа з відкритим вихідним кодом дає розробникам можливість створювати

розширювані пакети та ресурси, щоб полегшити розробку не лише для себе, але й для всіх, хто пише код за допомогою Express.js.

- Маршрутизація. Допомагаючи програмі відповідати на клієнтські запити через конкретну кінцеву точку, Express.js використовує систему маршрутизації, яка дозволяє розділити об'ємну систему маршрутів за допомогою екземпляра маршрутизатора у фреймворку. Ця система експрес-маршрутизації полегшує управління структурою програми, групуючи різні маршрути в одній папці або каталозі. Розробники можуть створювати більш обслуговувані коди, використовуючи маршрутизатор для групування функцій і уникнення дублювання коду.

- Безперебійний процес розробки. Проміжне програмне забезпечення - це коди, які виконуються до того, як запит HTTP досягне обробника маршруту або до того, як клієнт отримає відповідь, що дає структурі можливість запускати типовий сценарій до або після запиту клієнта. Використовуючи middleware, розробники можуть впроваджувати сценарії, які перехоплюють потік програми, наприклад, перевірка успішного входу чи виходу користувача.

І останній в цій четвірці технологій Mongo DB. MongoDB - це система керування базами даних, розроблена для зберігання даних у форматі JSON та виконання операцій читання та запису для сортування даних за допомогою різних типів полів. Зокрема, функціональні можливості MongoDB включають текстовий/географічний пошук, шардування, реплікацію, візуалізацію даних, а також можливості співпраці та управління резервним копіюванням [22].

Розробники можуть здійснювати доступ та аналіз даних за допомогою спеціальних запитів, індексування та агрегації в режимі реального часу, а також використовувати MongoDB Atlas для управління та відновлення розгортань на різних хмарних платформах, таких як AWS та Azure. MongoDB надає інтеграцію з Apache Spark, Apache Kafka та Kubernetes, доступний за

місячною підпискою та надається підтримкою по телефону, у чаті та документації.

Зазвичай MongoDB використовують в таких випадках:

- Зберігання. MongoDB може зберігати великі обсяги структурованих і неструктурованих даних. Масштабується вертикально і горизонтально. Індекси використовуються для покращення ефективності пошуку. Пошук виконується за запитами полів, діапазонів і виразів.
- Інтеграція даних. Інтегрує дані для програм, у тому числі для гібридних і багатохмарних програм.
- Опис складних структур даних. Бази даних документів дозволяють вбудовувати документи для опису вкладених структур і можуть допускати варіації даних.
- Балансування навантаження. MongoDB можна використовувати для роботи на кількох серверах.

MongoDB має декілька переваг над іншими:

- Без схеми. Так само, як і інші системи NoSQL, MongoDB не вимагає створення попередньо визначених схем даних і може зберігати різноманітні типи інформації. Це дозволяє користувачам створювати необмежену кількість полів у документі, сприяючи легкості масштабування бази даних MongoDB у порівнянні з реляційними системами.
- Документоорієнтований. Використання документів має перевагу в тому, що ці об'єкти можуть відображатися на власних типах даних у різних мовах програмування. Наявність вбудованих документів також сприяє зменшенню необхідності об'єднання баз даних, що може призвести до економії витрат.
- Масштабованість. Основною особливістю MongoDB є її здатність до горизонтального масштабування, що робить її важливим інструментом для компаній, які мають справу з великими обсягами даних. Крім того,

шардування дозволяє базі даних розподіляти дані між кластером серверів. MongoDB також підтримує створення зон даних на основі сегментного ключа.

- Підтримка сторонніх розробників. MongoDB підтримує різноманітні механізми зберігання даних і забезпечує API для підключення механізмів зберігання сторонніх розробників, що дозволяє їм створювати власні рішення для зберігання в MongoDB.

- Агрегація. Система управління базами даних також включає в себе вбудовані засоби агрегації, що дозволяють користувачам виконувати код MapReduce безпосередньо в базі даних. Крім того, MongoDB обладнана власною файловою системою під назвою GridFS.

3. ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Розробка клієнта

Стек технологій MERN (MongoDB, Express, React, Node.js) дозволяє клієнту та серверу взаємодіяти на веб-сайті. Клієнт, що може бути веб-браузером або мобільним додатком, відправляє HTTP запити на сервер, використовуючи інтерфейс, створений за допомогою React. Сервер, який працює на Node.js з використанням Express, отримує ці запити, обробляє їх та взаємодіє з базою даних MongoDB для забезпечення необхідної інформації. Після обробки запиту сервер надсилає відповідь клієнту, яка може містити дані з бази даних чи відображати результати дій користувача. Цей обмін даними дозволяє відображати актуальну інформацію на веб-сторінках, реагувати на користувацькі дії та створювати багатофункціональні додатки з динамічним змістом.

Для початку роботи потрібно створити стандартний каркас для створення сайту. Це відбувається за допомогою декількох команд.

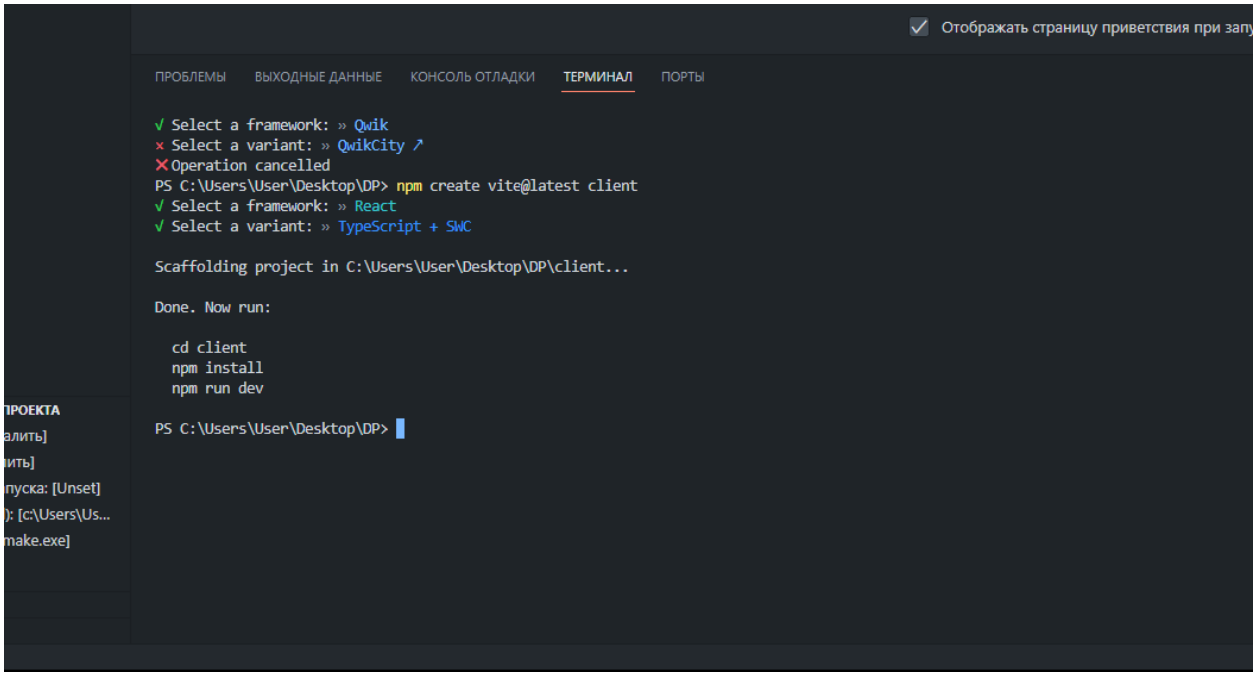

Рисунок 3.1 – Команда для сворення нового проекту

Після створення каркасу потрібно його заповнювати та створорювати особистий контент який потрібний.

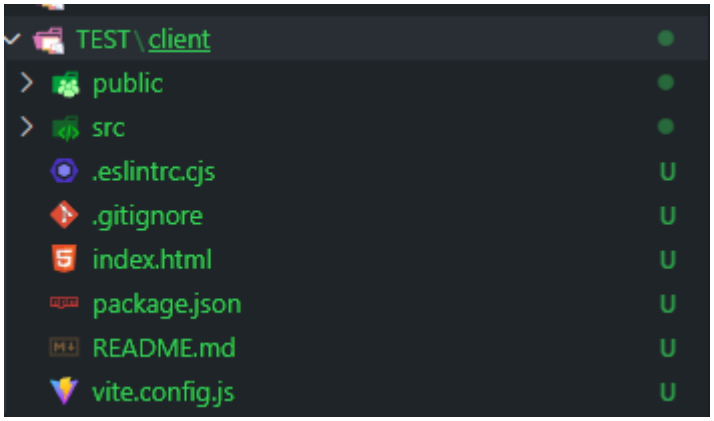

Рисунок 3.2 – Початкова структура

Для подальшої роботи потрібно встановити NPM. (Node Package Manager) - це пакетний менеджер для JavaScript, особливо популярний у світі Node.js. Він дозволяє розробникам легко встановлювати, оновлювати та керувати залежностями в проектах. npm має велику колекцію пакетів бібліотек, фреймворків, утиліт, доступних для використання.

Цей інструмент допомагає спростити роботу розробників, оскільки можна швидше встановлювати пакети, необхідні для роботи програм, та автоматично керувати версіями цих пакетів. npm також дозволяє створювати власні пакети для використання іншими розробниками, сприяючи в розвитку екосистеми JavaScript.

Завдяки npm розробники можуть ефективно використовувати готові рішення, спрощуючи розробку програм, тим самим прискорюючи процес розробки та забезпечуючи стабільність та актуальність коду завдяки оновленню пакетів. Це значно полегшує підтримку та розвиток проектів, сприяючи швидкому зростанню та вдосконаленню програмних продуктів.

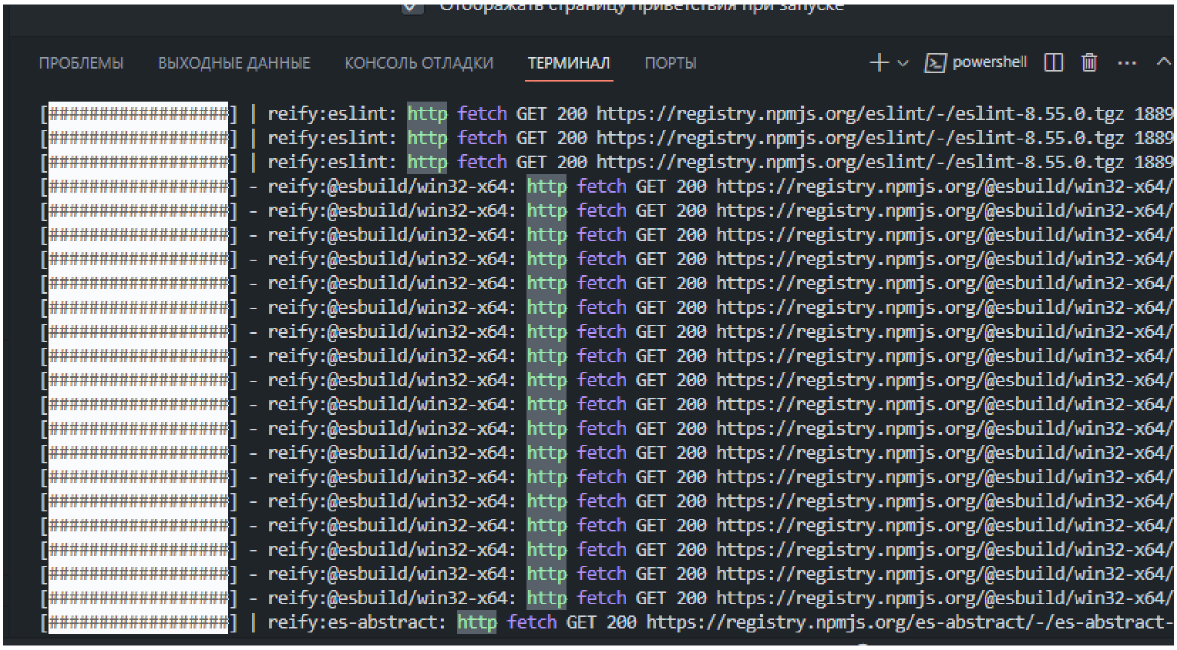

Рисунок 3.3 – Встановлення NPM та потрібних залежністей

Після завантаження всіх залежністей встановлюється Tailwind CSS - це CSS-фреймворк, який пропонує модульний підхід до створення інтерфейсів, базуючись на наборі низькорівневих класів. Замість написання власного CSS, розробники використовують класи, що надають стилізацію прямо в HTML-розмітці.

Tailwind використовує класи, які називаються за функціональністю, наприклад, **text-center**, **p-4** (padding), **bg-blue-500** (фон з коліром). Це дозволяє швидко та просто застосовувати стилі до елементів без необхідності написання власного CSS.

Робота з Tailwind CSS полягає у використанні цих класів у HTML-розмітці, надаючи потрібну стилізацію безпосередньо до елементів. Розробники можуть комбінувати класи, використовуючи їх для створення бажаних ефектів та вигляду веб-сторінок.

Крім того, Tailwind надає можливість налаштувати та розширити набір класів, дозволяючи створювати власні стилізації та компоненти для проекту. Цей фреймворк дозволяє прискорити розробку інтерфейсів, забезпечуючи гнучкість та швидкість у використанні.

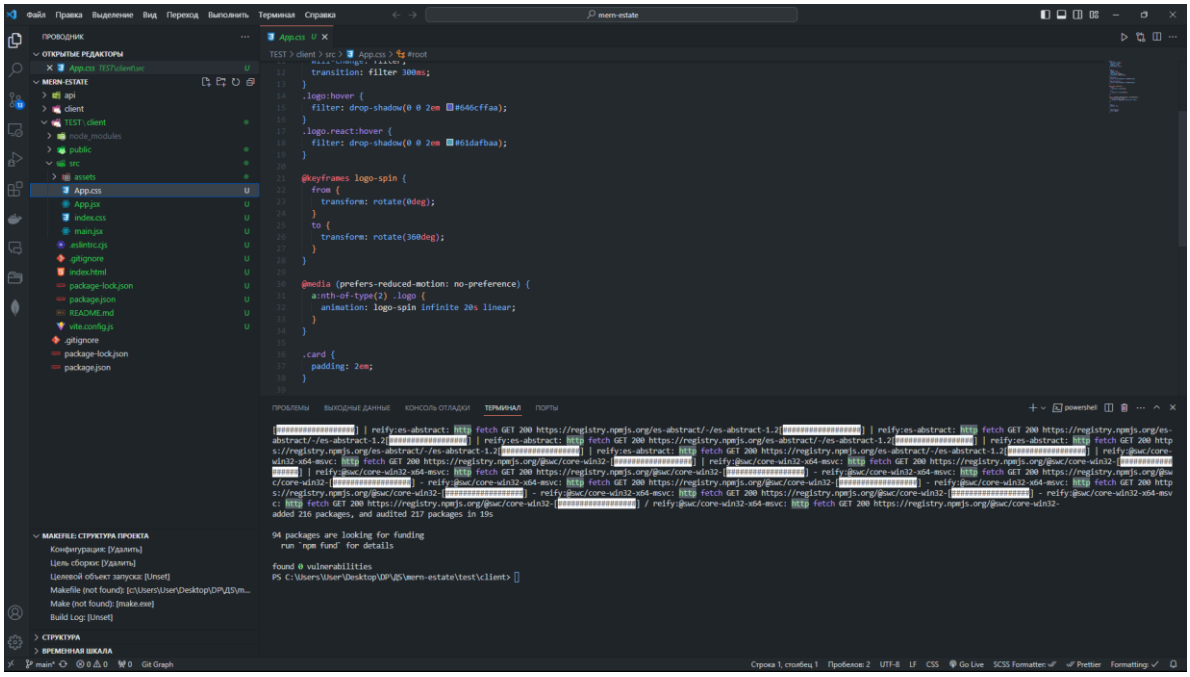

Рисунок 3.4 – Встановлений Tailwind CSS

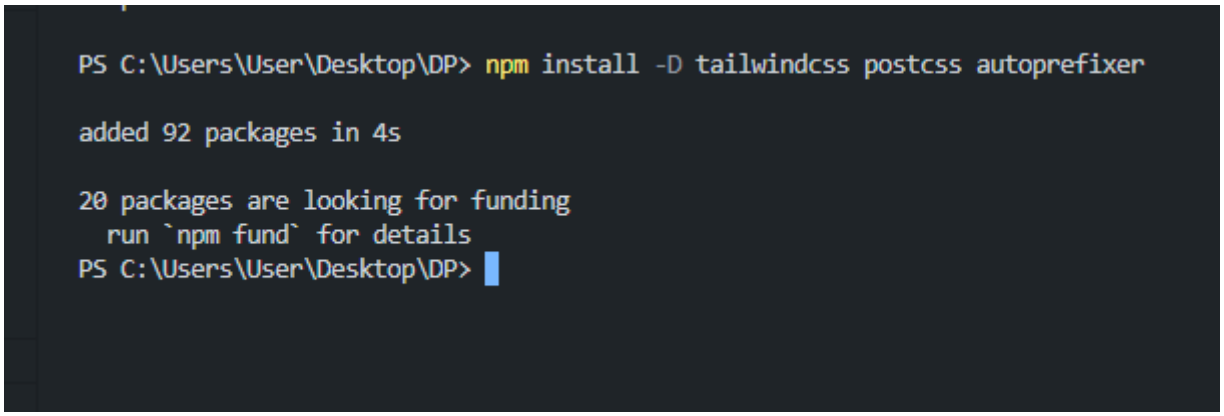

Рисунок 3.5 – Встановлення додаткових компонентів Tailwind

Команда **npm install -D tailwindcss postcss autoprefixer** використовується для встановлення трьох різних пакетів в проекті, використовуючи npm (Node Package Manager):

1) **tailwindcss**: Це пакет, який містить CSS-фреймворк Tailwind. Встановлення цього пакету дозволяє використовувати Tailwind CSS у проекті для швидкого та зручного створення інтерфейсу з використанням набору готових класів;

2) **postcss**: Це інструмент, який дозволяє розширювати функціональність CSS за допомогою плагінів. У випадку з Tailwind, він використовується для обробки і об'єднання стилів, які ви визначаєте у ваших CSS-файлах;

3) **autoprefixer**: Цей плагін для PostCSS автоматично додає вендорні префікси до вашого CSS-коду. Це дозволяє забезпечити сумісність із різними браузерами, не вказуючи вручну префікси для кожної CSS-властивості.

Опція **-D** в команді **npm install** вказує на те, що пакети будуть встановлені як devDependencies, тобто вони необхідні лише для розробки, але не для його функціонування в режимі виробництва.

Разом з цими пакетами та відповідним налаштуванням ви зможете успішно використовувати Tailwind CSS у своєму проекті та автоматично обробляти та додавати вендорні префікси до CSS-коду.

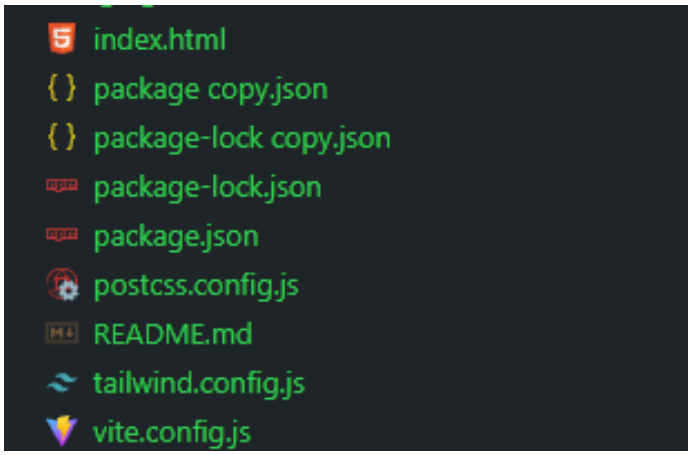

Рисунок 3.6 – Додаткова структура клієнту

Після завантаження Tailwind CSS у проект необхідно об'єднати файли та забезпечити їх взаємодію. Для цього, в головний файл проекту який слугує основним файлом стилів, додаємо імпорт **tailwind.css**. Це сприяє використанню готових стилів Tailwind, які можна доповнити власними стилями та компонентами, створеними в процесі розробки.

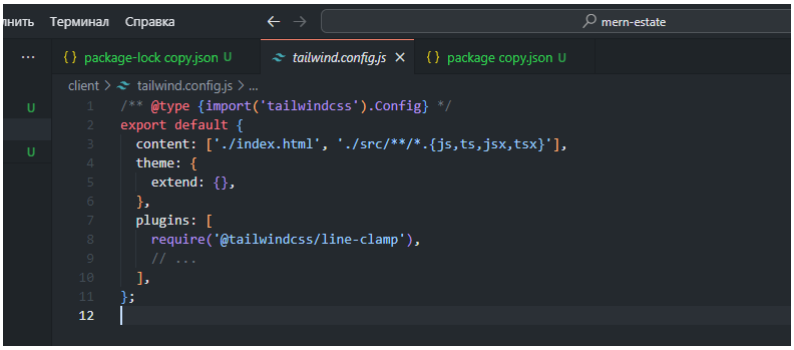

Рисунок 3.7 – Вміст файлу tailwind.config

Коли додані перші стилі, потрібно зробити компоненти та головні сторінки, які будуть імпортовані до головного файлу. У React компоненти та сторінки — це основні будівельні блоки веб-додатку, що допомагають організувати та відображати контент:

- **Компоненти в React:** Компоненти - це малий кубик будь-якого React додатку. Вони є вузлами інтерфейсу, які можна перевикористовувати, групувати та складати разом для створення складних інтерфейсів. Компоненти можуть бути класовими (класові компоненти) або функціональними (функціональні компоненти). Вони можуть містити в собі як логіку, так і представлення, що дозволяє розділити інтерфейс на логічні блоки та робить код більш організованим та перевикористовуваним;

- **Сторінки в React:** Сторінки представляють собою візуальні одиниці, які відображаються користувачеві як окремі URL-адреси в додатку. Кожна сторінка може складатися з одного або кількох компонентів, що утворюють вміст та логіку цієї сторінки. Наприклад, веб-додаток може мати сторінки для реєстрації, входу, профілю користувача, списку товарів тощо. Кожна з цих сторінок в React може бути реалізована як окремий компонент або комбінація компонентів, які утворюють необхідний функціонал.

Використання компонентів і сторінок у React дозволяє створювати структуровані, повторно використовувані та легко змінювані блоки функціоналу, що спрощує розробку, підтримку та розширення веб-додатків.

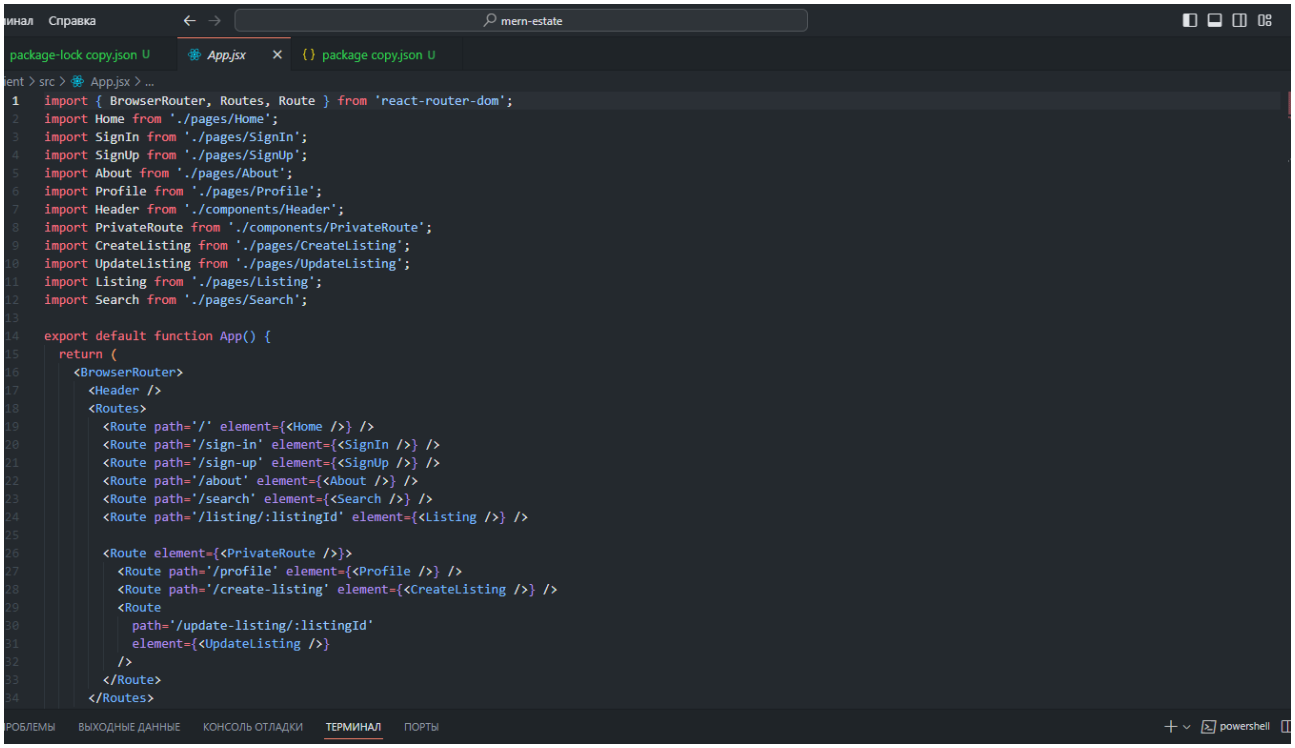

Рисунок 3.8 – Вміст файлу App.js

При запуску сервера всі компоненти наприклад: Header, Contact, OAuth та інші будуть автоматично завантажені на клієнт та коректно відображати їх вміст. В даному випадку працюємо над головною сторінкою, до якої додаємо вкладки.

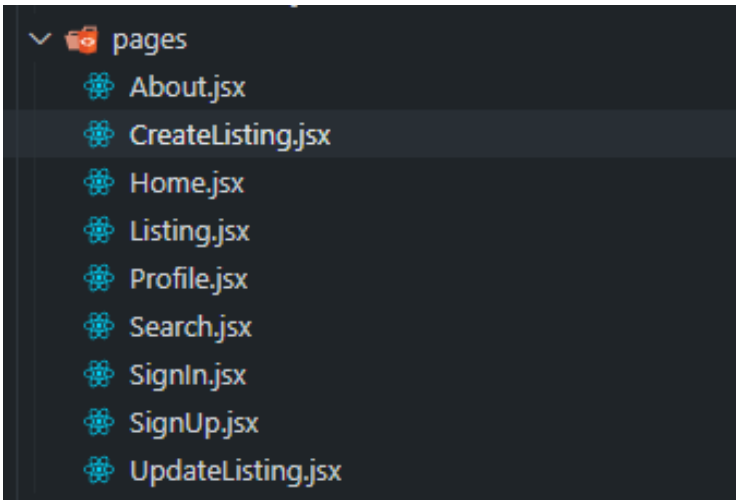

Рисунок 3.9 – Вміст папки з сторінками

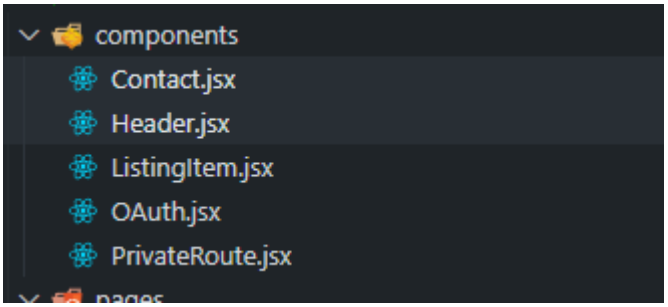

Рисунок 3.10 – Вміст папки з компонентами

Створюємо потрібні компоненти які будуть на сайті. Компонент хедера у React відповідає за створення та відображення верхньої частини веб-сайту. Він містить елементи, такі як логотип, навігаційне меню та інші елементи, що відображаються на кожній сторінці сайту.

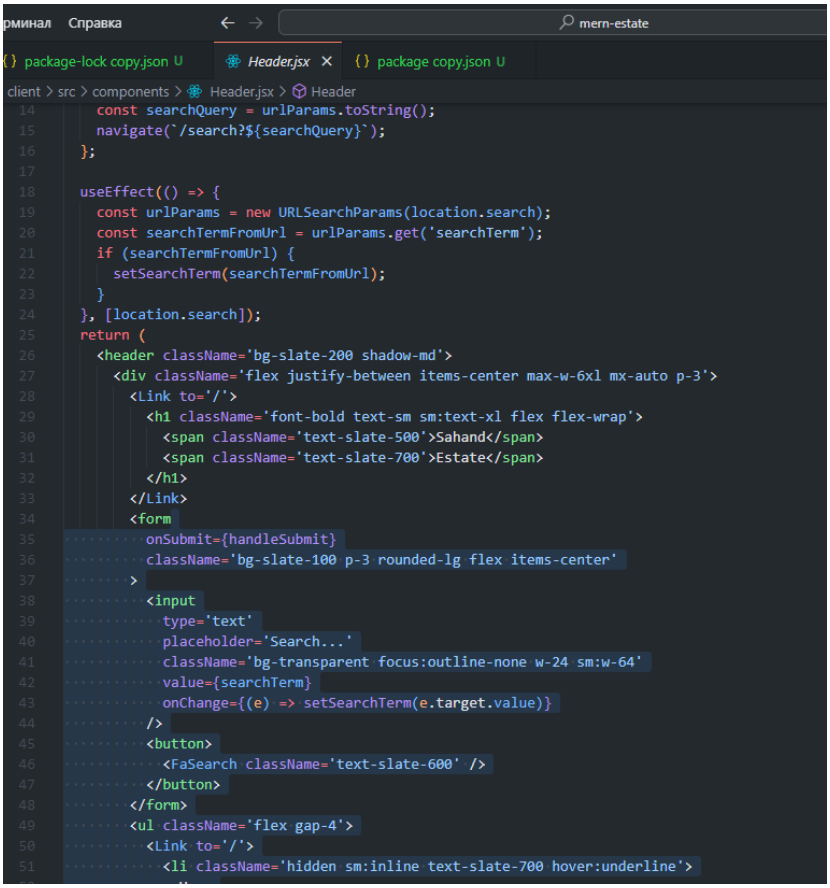

Рисунок 3.11 – Вміст Header

Компонент хедера допомагає створити єдиний візуальний стиль для сайту, спрощує навігацію та забезпечує консистентність дизайну на всіх сторінках.

3.2 Серверна частина

Експрес - це фреймворк для Node.js, який використовується для створення веб-додатків. Він дозволяє швидко створювати сервери, обробляти запити та відправляти відповіді. Експрес відповідає за маршрутизацію запитів, обробку middleware (проміжного програмного забезпечення) та взаємодію з базою даних або зовнішніми API. Він працює за концепцією обробки запиту-відповіді, коли запити від клієнта обробляються і відповіді надсилаються назад через різні маршрути та логіку, яку ви визначаєте. Експрес спрощує розробку веб-додатків, надаючи зручний інтерфейс для роботи з серверним кодом та HTTP-запитами.

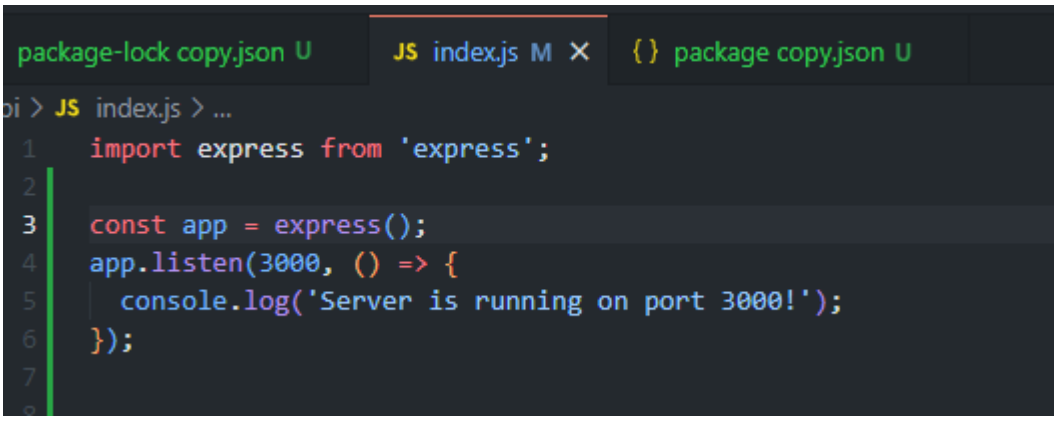

Рисунок 3.12 – Вміст файлу index.js

Коли початковий код для запуску сервера є, потрібно масштабувати та додати потрібні інструменти а саме Pakcage.json та nodemon. ReductionNodemon - це утиліта, яка відстежує будь-які зміни у вашому Node.js-додатку та автоматично перезапускає сервер.

```
npm install -g nodemon
```

Рисунок 3.13 – Команда для завантаження

Цей інструмент особливо корисний у середовищі розробки, оскільки він економить час, усуваючи необхідність вручну перезавантажувати сервер

Node.js після внесення змін до коду. Коли запускається Node.js-додаток за допомогою Nodemon, можна спостерігати за файлами в каталозі, де знаходиться серверний скрипт. Якщо будь-які файли зміняться, Nodemon автоматично перезапустить ваш сервер. Це допомагає оптимізувати процес розробки, роблячи його більш ефективним і продуктивним.

```
20 packages are looking for funding
  run `npm fund` for details
PS C:\Users\User\Desktop\DP> npm start
```

Рисунок 3.14 – Перезапуск серверу для роботи Nodemon

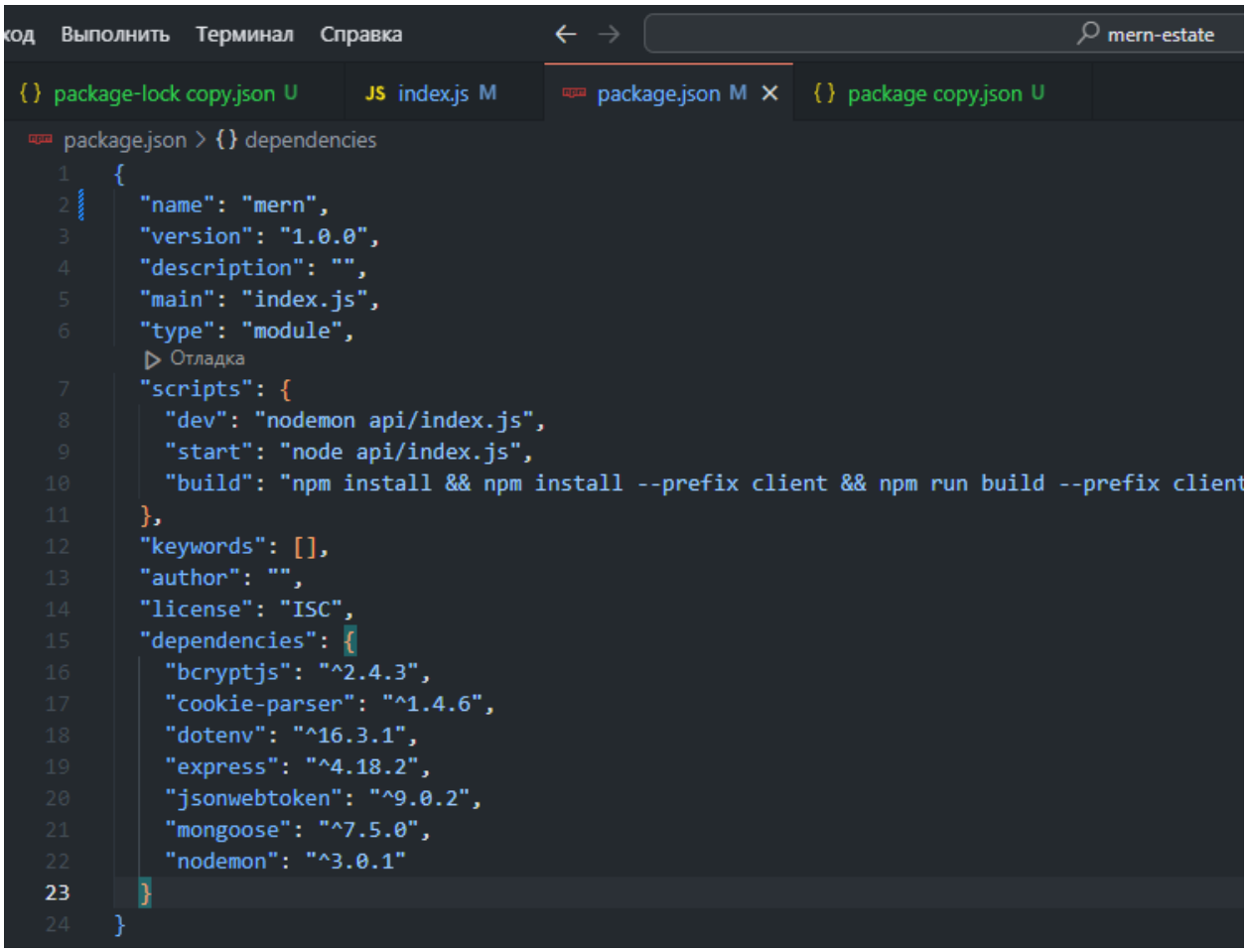

Рисунок 3.15 – Всміст файлу Package

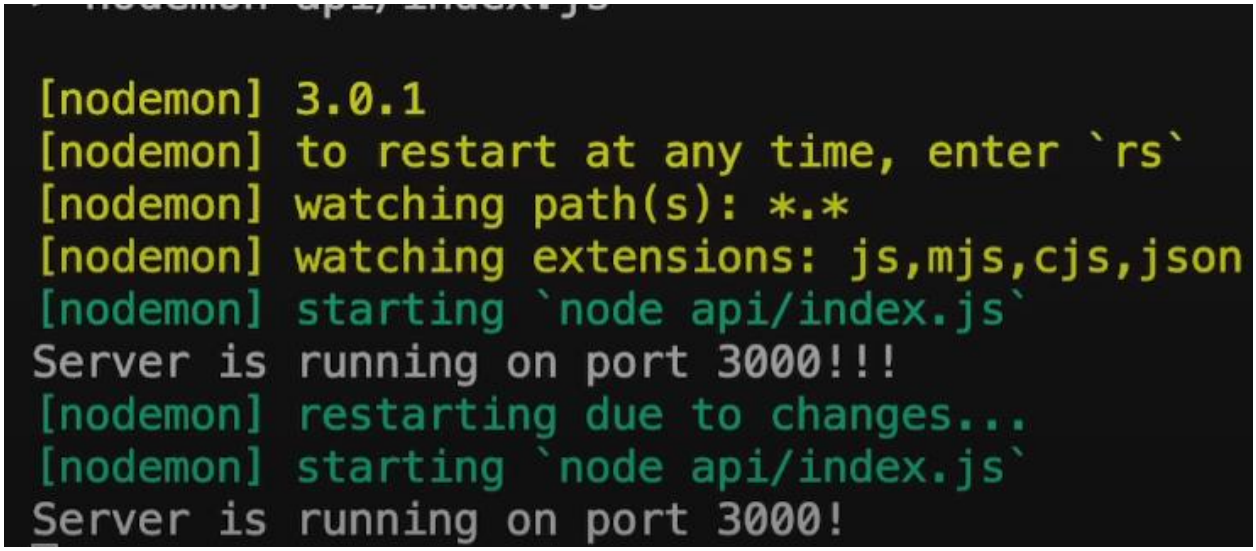

Рисунок 3.16 – Запуск роботи Nodemon

MongoDB та Nodemon можуть працювати разом у середовищі розробки Node.js. MongoDB - це база даних NoSQL, яка зберігає дані у гнучкому JSON-подібному форматі, а Nodemon - це утиліта, яка допомагає автоматично перезапускати сервер Node.js при виявленні змін файлів у каталозі.

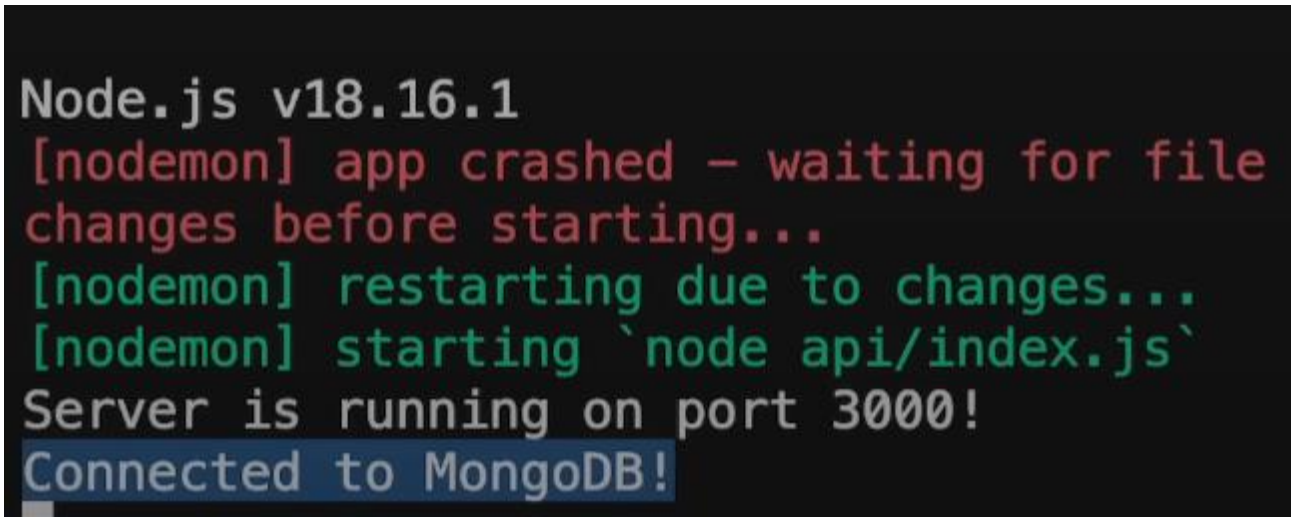

Рисунок 3.17 – Вдале підключення MongoDB

При розробці Node.js додатку, який взаємодіє з базою даних MongoDB, використання Nodemon спрощує процес розробки. Коли вносите зміни в код, Nodemon автоматично перезавантажить сервер, що дозволить негайно протестувати ці зміни без необхідності кожного разу перезавантажувати сервер вручну. Таке налаштування ідеально підходить для швидкої розробки та тестування додатків, які покладаються на взаємодію з базами даних.

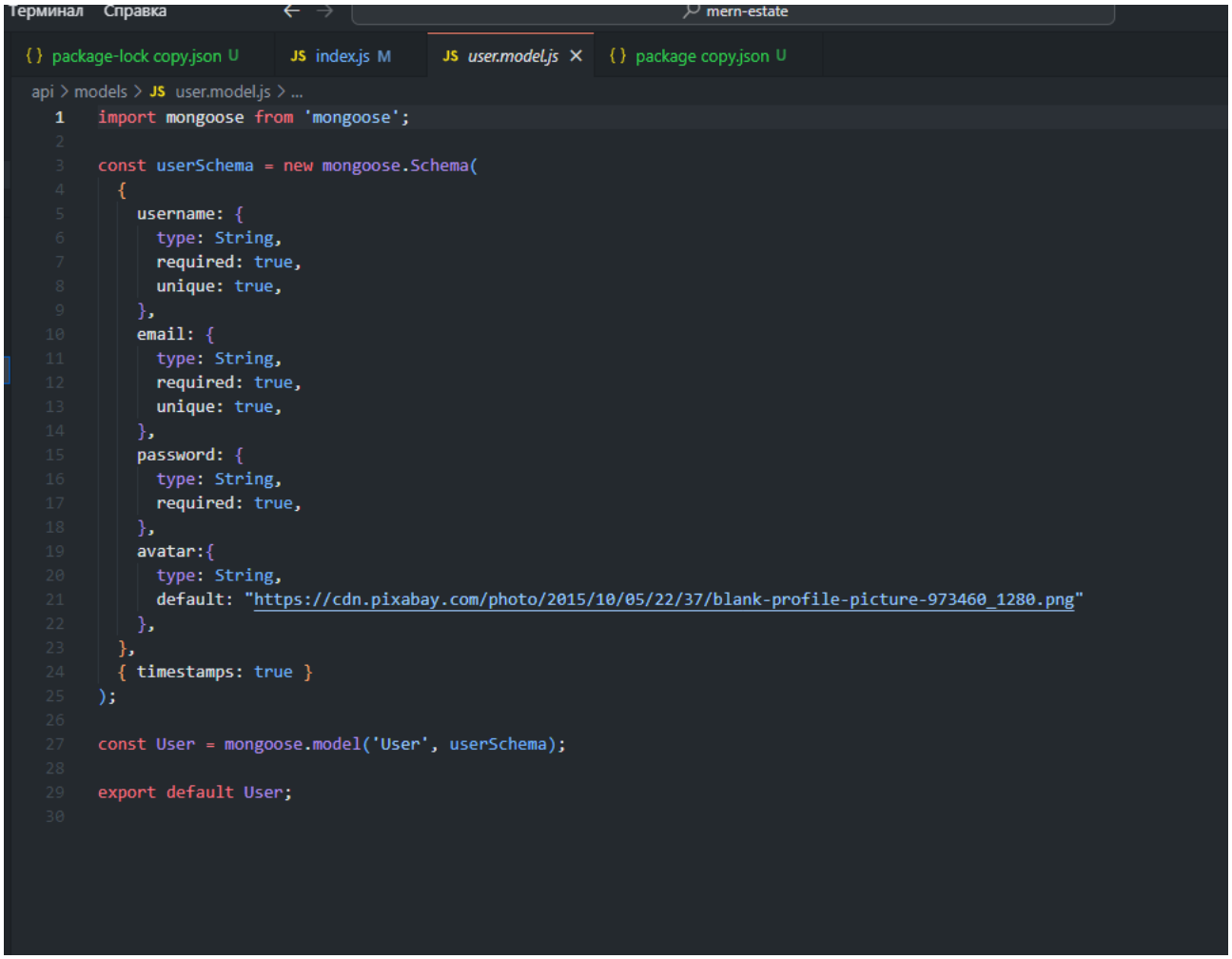

Рисунок 3.18 – Додавання бібліотеки

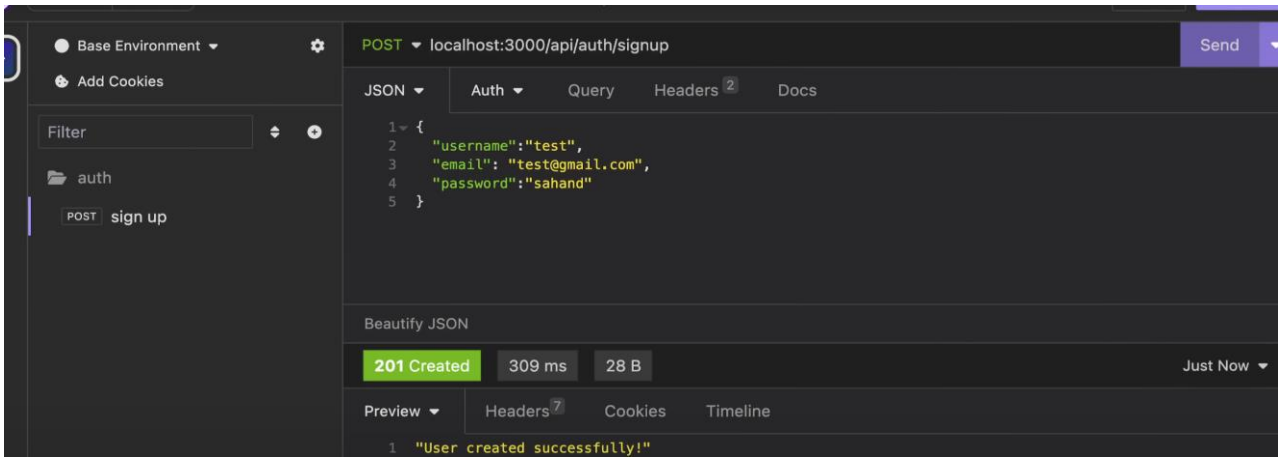

Рисунок 3.19 – Перевірка на правильність роботи запитів

Файли auth.controller.js, listing.controller.js та user.controller.js містять різні контролери (контролери в MVC архітектурі) для обробки запитів до веб-

додатку, який, можливо, працює на базі Node.js або Express.js. Контролери в цих файлах відповідають за обробку різних типів запитів HTTP: GET, POST, PUT, DELETE.

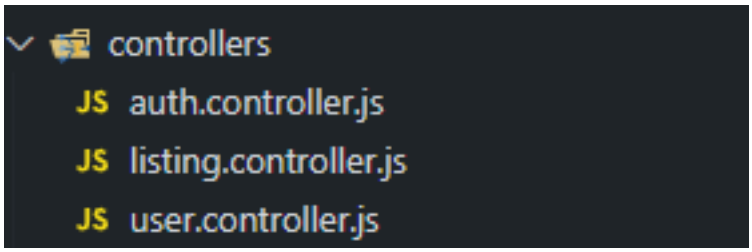

Рисунок 3.20 – Вміст папки контролер

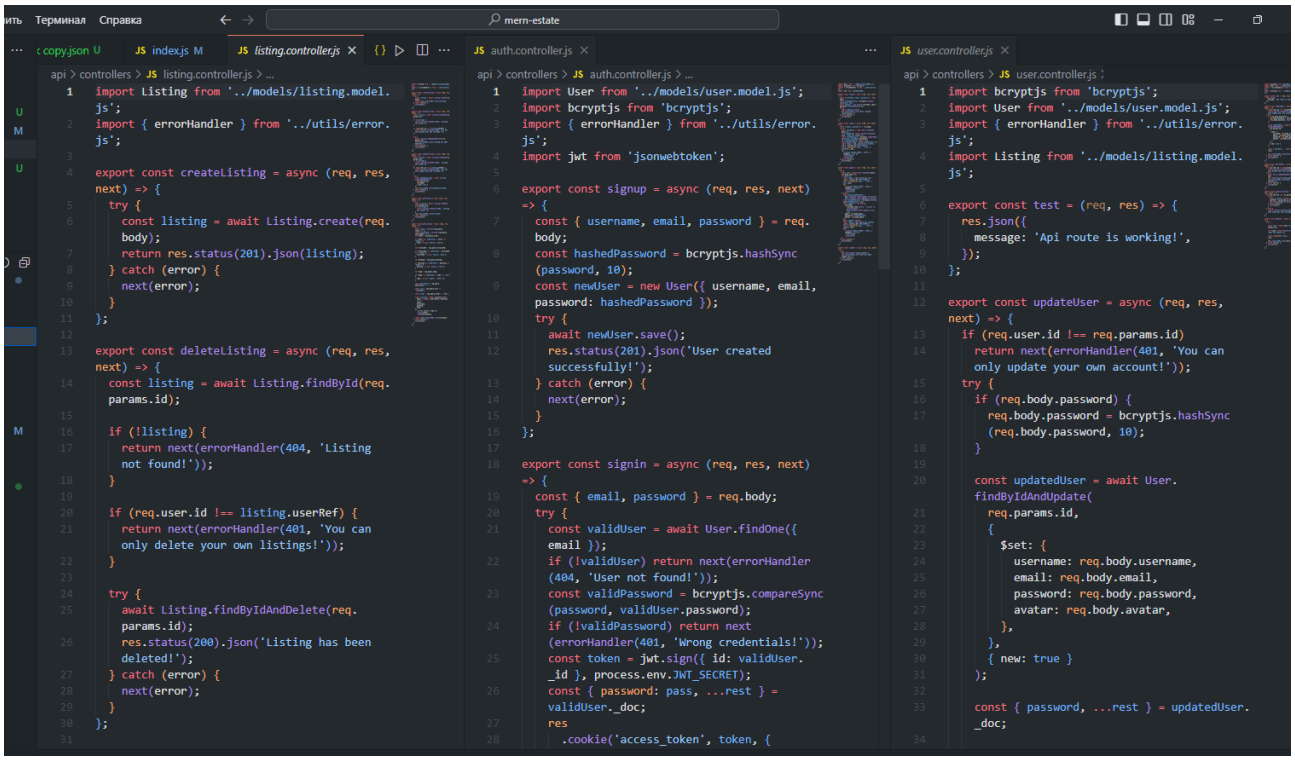

Рисунок 3.21 – Вміст кожного контролеру

Нижче наведено короткий опис кожного контролера з вказаними операціями:

1. **auth.controller.js;**

- **signup**: Обробляє POST-запит для створення нового користувача. Цей контролер отримує дані про користувача (ім'я,

електронну адресу, пароль), хешує пароль та створює новий запис у базі даних;

- **signin**: Обробляє POST-запит для входу користувача. Перевіряє, чи існує користувач з введеними даними (електронна адреса, пароль), генерує токен доступу та повертає дані користувача;
- **google**: Обробляє авторизацію через Google. Перевіряє наявність користувача за електронною адресою. Якщо користувач існує, повертає дані; якщо ні – створює нового користувача;
- **signOut**: Обробляє вихід користувача. Видаляє cookie та повідомляє про вихід користувача;

2. **listing.controller.js;**

- **createListing**: Обробляє POST-запит для створення нового оголошення. Зберігає дані про оголошення в базі даних;
- **deleteListing**: Обробляє DELETE-запит для видалення оголошення. Перевіряє, чи існує оголошення та чи користувач має дозвіл на його видалення;
- **updateListing**: Обробляє PUT-запит для оновлення оголошення. Перевіряє наявність оголошення та дозвіл на його оновлення;
- **getListing**: Обробляє GET-запит для отримання даних про конкретне оголошення;
- **getListings**: Обробляє GET-запит для отримання списку оголошень за різними параметрами (назва, тип, обмеблованість тощо);

3. **user.controller.js;**

- **test**: Обробляє простий GET-запит для перевірки роботи маршруту API;
- **updateUser**: Обробляє PUT-запит для оновлення даних користувача;

- **deleteUser**: Обробляє DELETE-запит для видалення облікового запису користувача;
- **getUserListings**: Обробляє GET-запит для отримання списку оголошень користувача;
- **getUser**: Обробляє GET-запит для отримання даних про користувача за його ідентифікатором.

Ці контролери відповідають за обробку запитів від клієнтів і виконання необхідних операцій у системі на основі отриманих даних.

3.3 Візуалізація сайту

Далі показано дизайн сайту та його можливості та ключові особливості.

JWT Token для реєстрації та входу, функціонал реєстрації та входу на сайт. Використання JWT Token було одним із основних кроків для забезпечення безпеки та ідентифікації користувачів.

Була створена форма, де користувачі могли ввести свої особисті дані, такі як ім'я, електронну адресу та пароль.

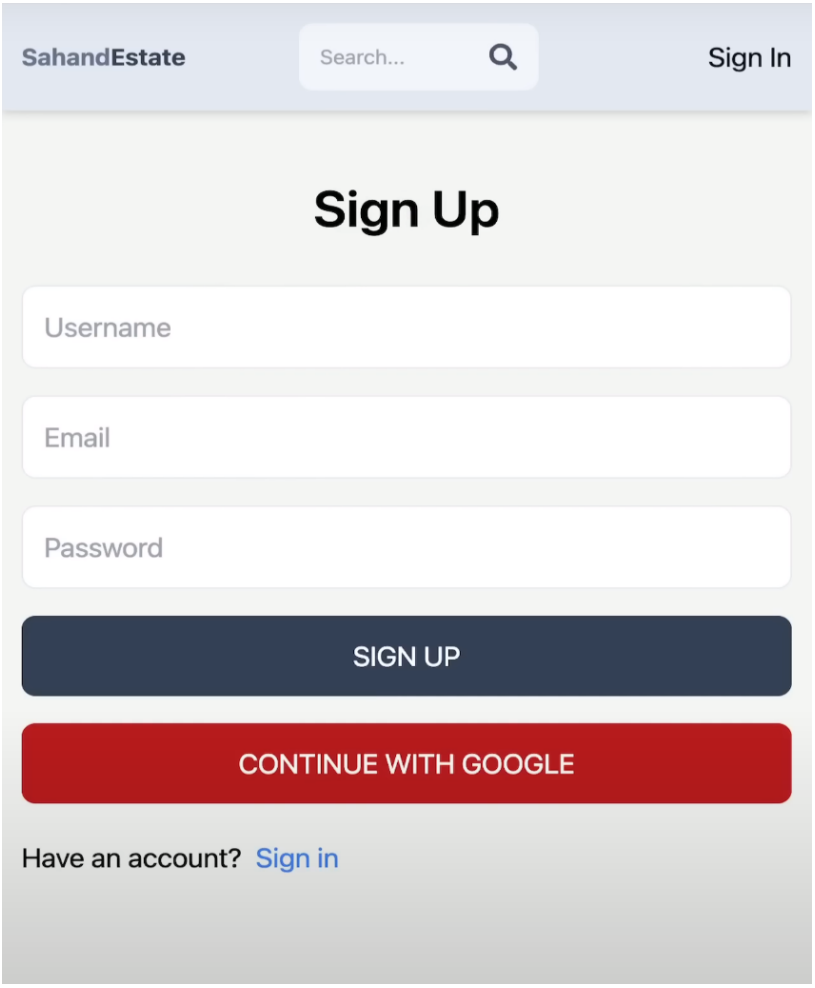

Рисунок 3.22 – Форма реєстарції

Коли користувач вводить свої дані та натискає кнопку "Зареєструватися", дані надсилаються на сервер. Там використовується бібліотека bcryptjs, щоб захешувати пароль та створити новий запис користувача у базі даних разом із захешованим паролем.

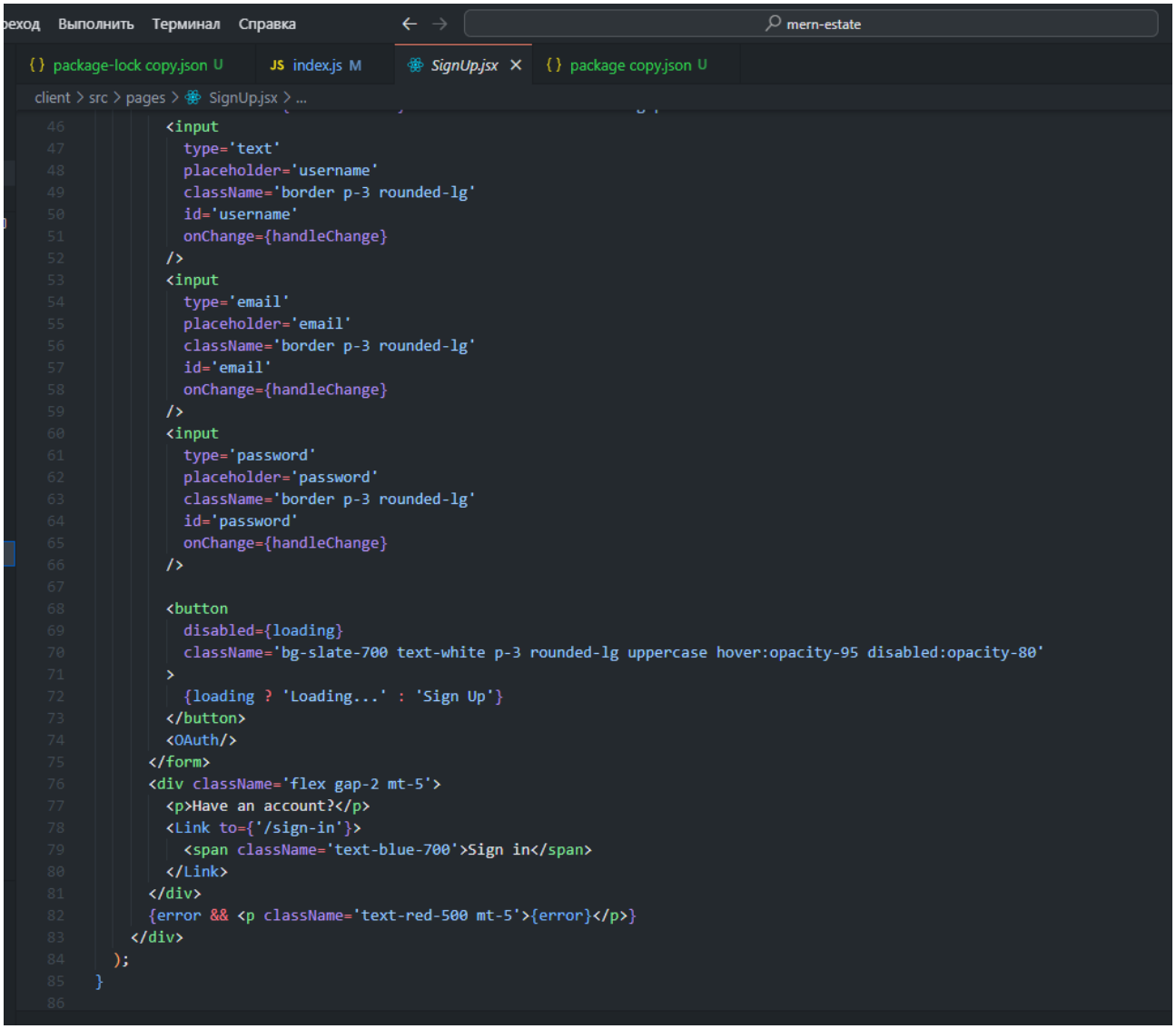

Рисунок 3.23 – Вміст файлу з реєстрацією

Після успішної реєстрації, на сервері генерувався JWT Token, який містив у собі інформацію про користувача. Цей токен був використаний для авторизації користувача на захищених маршрутах.

Вхід за допомогою JWT Token. При вході на сайт, користувач вводив свою електронну адресу та пароль. Після цього, дані перевірялися на сервері, і створювався новий JWT Token для цього сеансу, дозволяючи користувачеві увійти на сайт та мати доступ до його особистого простору.

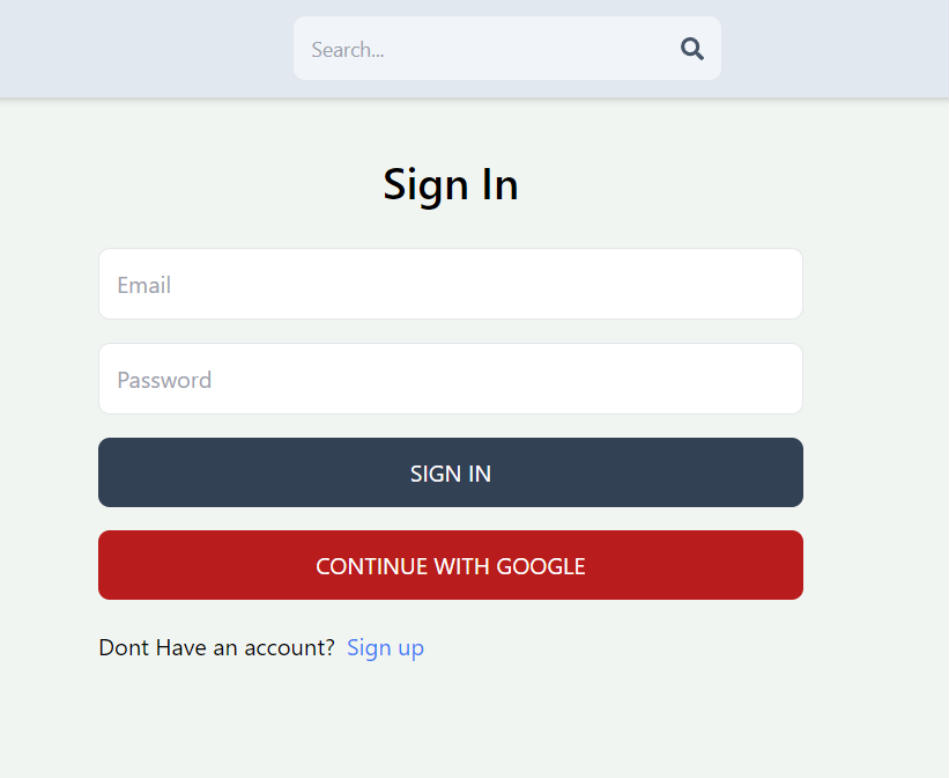

Рисунок 3.24 – Зовнішній вигляд форми входу

Реєстрація через гугл акаунт: Додавання можливості реєстрації через гугл акаунт було ще одним заходом для спрощення процесу входу на сайт.

Для налаштування авторизації через гугл акаунт на веб-сайті було використано інструменти Google API.

Обробка запиту від гугл акаунту на сервері: Коли користувач вибирав опцію "Увійти через Google", дані про аутентифікацію відправлялися на сервер, де вони перевірялись. Якщо користувач був успішно ідентифікований, створювався новий запис користувача на сервері.

Генерація JWT Token для користувача з гугл акаунтом: Після успішної аутентифікації, генерувався JWT Token для цього користувача, щоб він мав доступ до особистих можливостей на сайті.

Додавання цієї можливості спрощує процес реєстрації та входу на сайт, дозволяючи користувачам увійти швидко та безпечно, використовуючи їхні гугл дані.

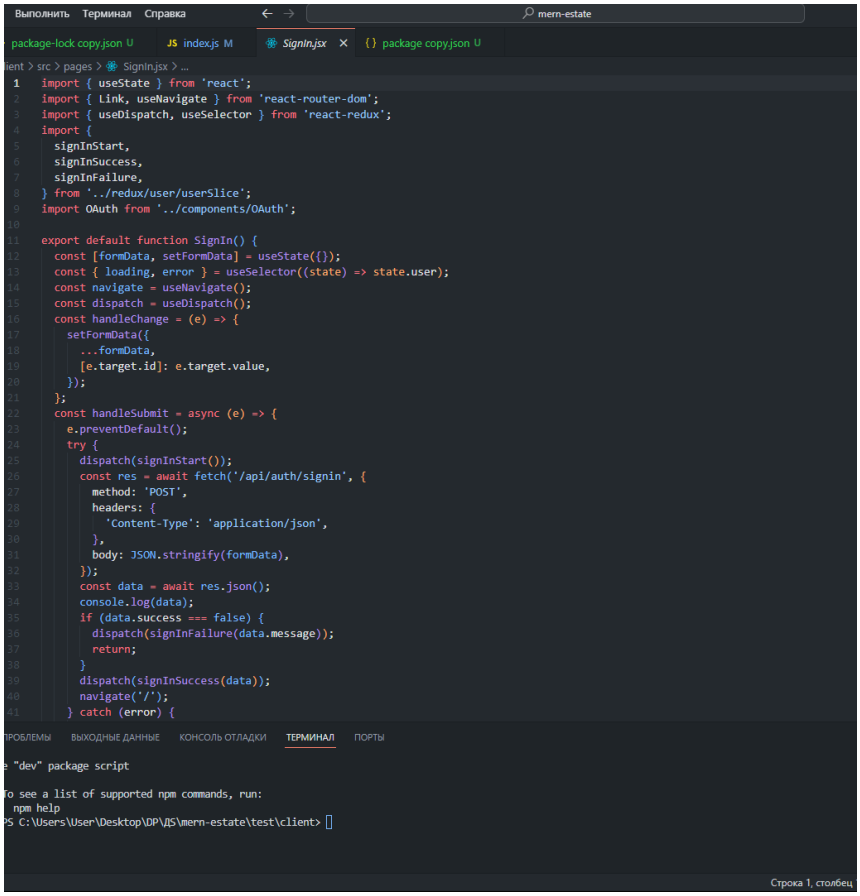

Рисунок 3.25 – Вміст файлу з входом на сайт

Рисунок 3.26 – JWT token

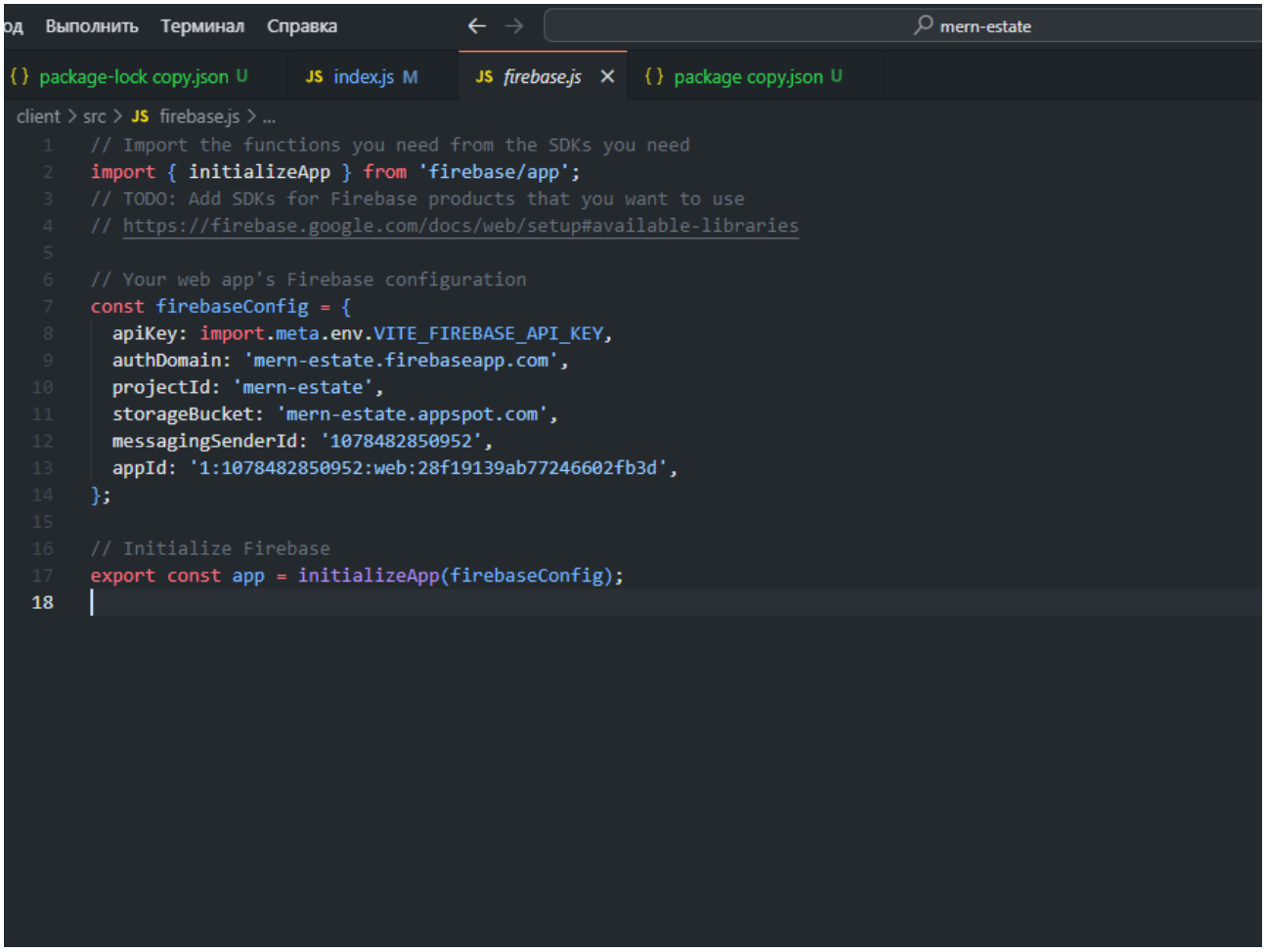

Рисунок 3.27 – Код для входу через Google акаунт

Після написаного коду слід протестувати сайт, спробувати функціонал та перевірити що не має багів та помилок. Для початку потрібно пройти перевірку на реєстрацію.

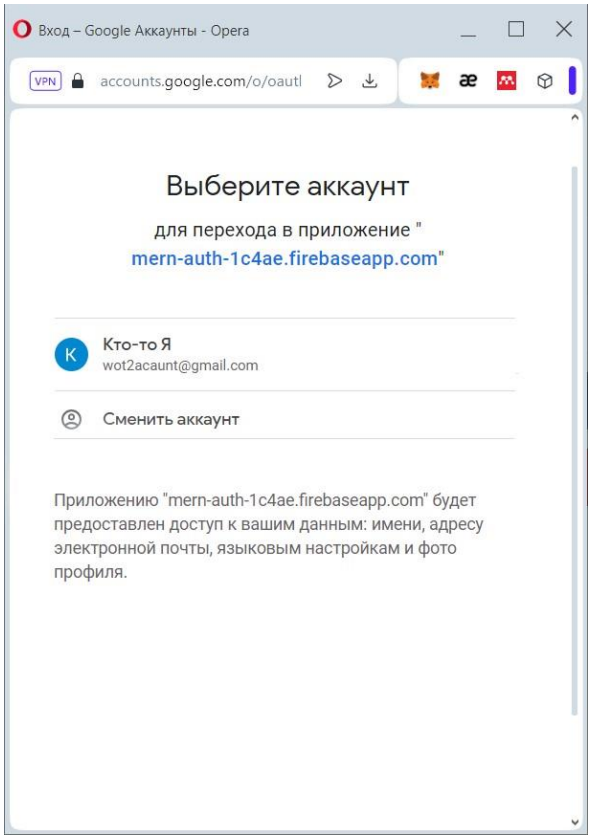

Рисунок 3.28 – Вхід до акаунту через google сервіс

Після вдалого входу бачимо, що акаунт успішно створений. Можна додавати матеріали та змінювати свій профіль.

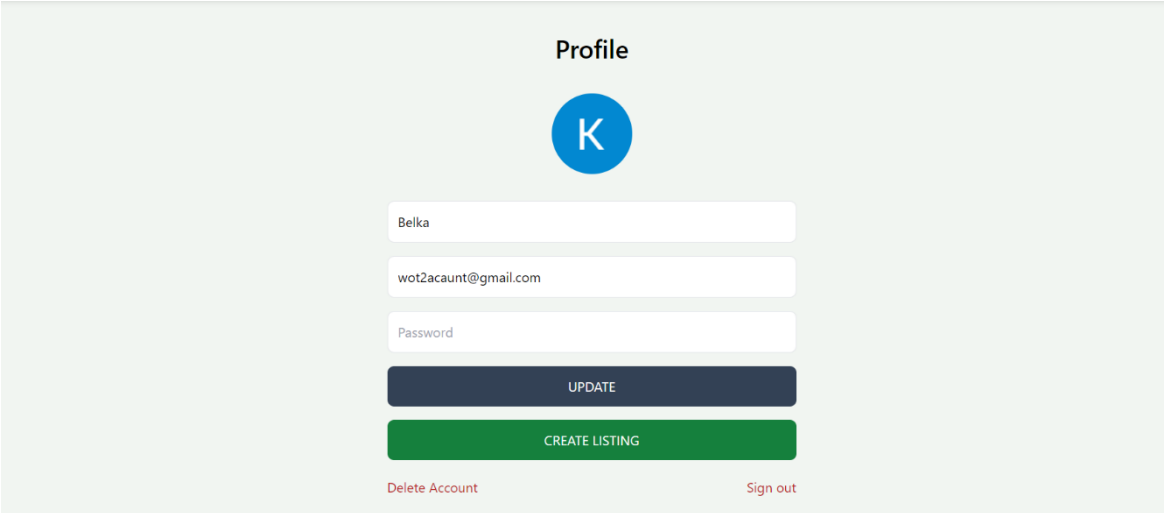

Рисунок 3.29 – Налаштування профілю

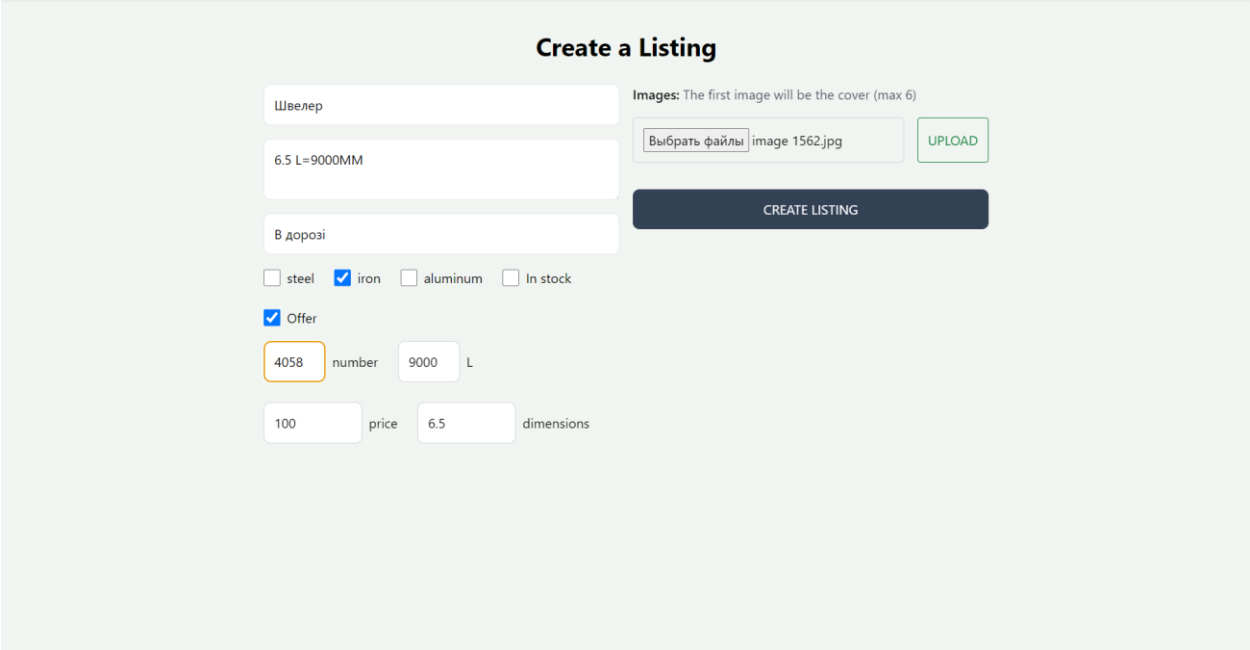

Рисунок 3.30 – Додавання контенту

Загальний вигляд контенту та його змісту з головної сторінки на якому розташовані ключові аспекти сайту.

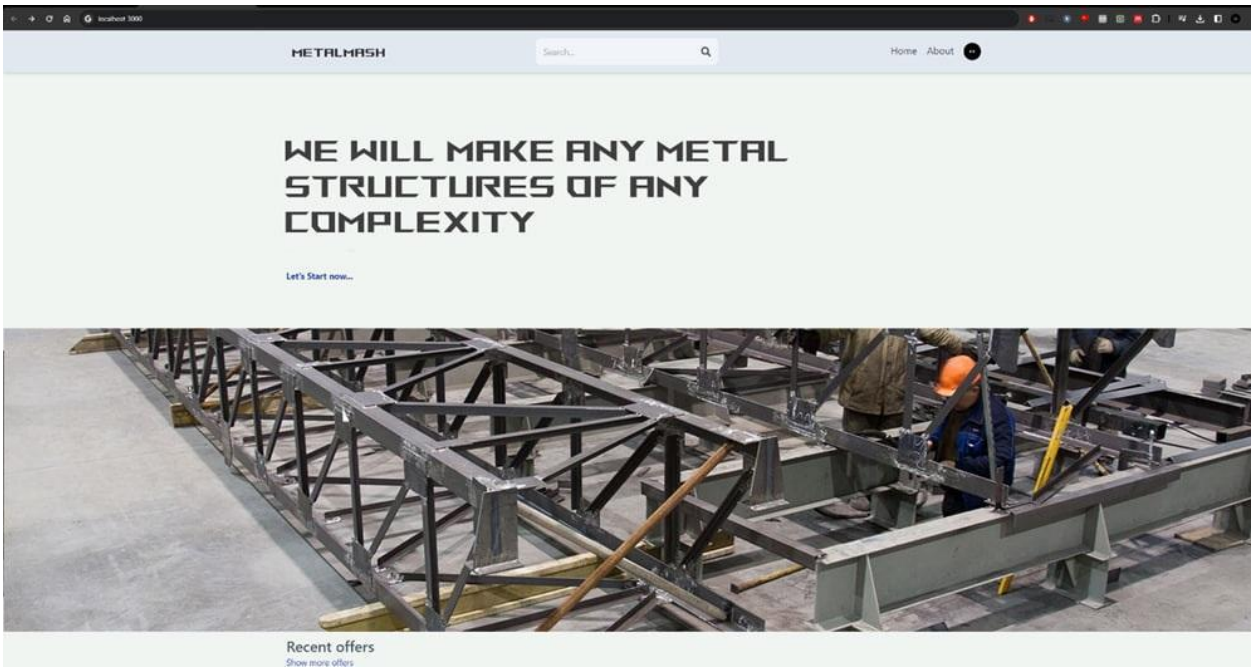

Рисунок 3.31 – Головна сторінка сайту

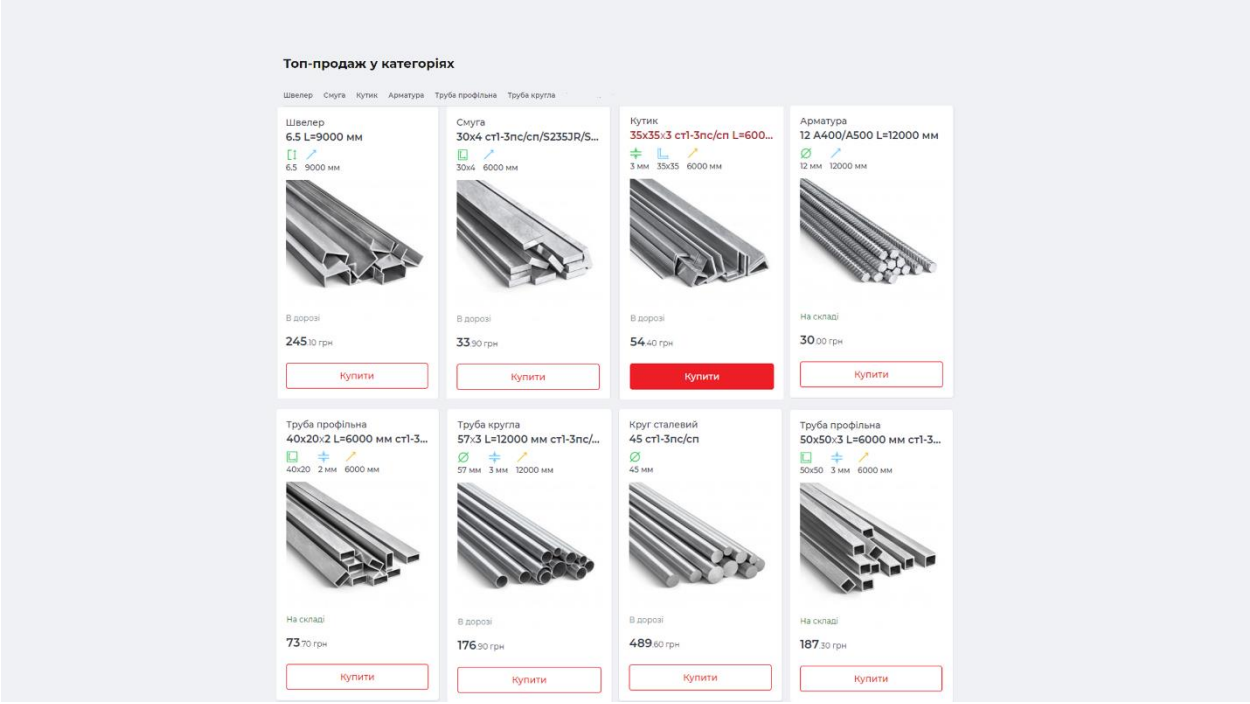

Рисунок 3.32 – Вміст сайту

ВИСНОВКИ

Отже, під час виконання кваліфікаційної роботи було детально розглянуто та розроблено інформаційну технологію для проектування веб-сайту, спрямованого на оптимізацію роботи відділу збуту промислового підприємства. Використовуючи стек технологій MongoDB, Express.js, React, Node.js, було досягнуто високого рівня ефективності та гнучкості у розробці та функціоналі запропонованого веб-інструменту.

Застосування цього стеку дозволило створити продукт, який відповідає сучасним вимогам, відзначається високою продуктивністю та зручністю у використанні. Результати роботи засвідчують необхідність та перспективність використання технології MERN в області розробки веб-рішень для промислових підприємств, зокрема для оптимізації процесів у відділі збуту.

Цей проект надає відділу збуту можливість ефективно використовувати інструменти для керування, аналізу та відстеження даних, підвищуючи тим самим його продуктивність та спрощуючи рутинні завдання. Цей проект також акцентує увагу на важливості використання сучасних технологій та підходів у сфері веб-розробки для досягнення оптимальних результатів, щодо викликів бізнес-процесів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. WEBDEV&SEO. Особливості розробки сайту в 2023 році [Електронний ресурс] – Режим доступу до ресурсу: https://webdevandseo.com/features-of-site-development-in-2023/.
2. Portilho, Thalles Guilherme Bogar. Desenvolvimento de uma aplicação web com linguagens funcionais puras, Universidade Federal de Uberlândia, 2021.
3. TechRadar. Bluehost Web Hosting review. [Електронний ресурс] – Режим доступу до ресурсу: https://www.techradar.com/reviews/bluehost.
4. Tooltester. IONOS MyWebsite Review: The Business Website Builder. [Електронний ресурс] Режим доступу до ресурсу: https://www.tooltester.com/en/reviews/1and1-mywebsite-review/.
5. Marina Buzzi; Barbara Leporini; Clara Meattini. Design Guidelines for Web Interfaces of Home Automation Systems Accessible via Screen Reader, IEEE, 2019.
6. Hasan H Khaleel, Rahmita OK Rahmat, Dimon M Zamrin. Components and implementation of a picture archiving and communication system in a prototype application. Reports in Medical Imaging, 2019.
7. S Boiano, A Borda, G Gaia. Participatory innovation and prototyping in the cultural sector: a case study. Proceedings of EVA London 2019, 2019.
8. Jieshan Chen, Chunyang Chen, Zhenchang Xing. Wireframe-based UI Design Search through Image Autoencoder. ACM Transactions on Software Engineering and Methodology. Volume 29 Issue 3Article No.: 19 pp 1–31, 2020
9. University of Maryland. Overview of Lucidchart. [Електронний ресурс] – Режим доступу до ресурсу: https://itsupport.umd.edu/itsupport?id=kb_article_view&sysparm_article=KB0010488.

10. Medium. User Guide to draw.io. [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@hasaliedirisinghe/user-guide-to-draw-io-6a1a4d7b5d33.
11. Ferdi Fauzan Putra, Hamidillah Ajie, Ika Anwar Safitri. DESIGNING A USER INTERFACE AND USER EXPERIENCE FROM PIRING MAKANKU APPLICATION BY USING FIGMA APPLICATION FOR TEENS. IJISTECH, 2021.
12. Fingent. Top 10 Tech Stacks That Reigh Software Development in 2023. [Електронний ресурс] – Режим доступу до ресурсу: https://www.fingent.com/blog/top-7-tech-stacks-that-reign-software-development/.
13. Monika Mehra, Manish Kumar, Anjali Maurya та інші. MERN Stack Web Development. Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 11756 – 11761, 2021.
14. Nobledesktop. React vs. Angular: How Do They Compare? [Електронний ресурс] – Режим доступу до ресурсу: https://www.nobledesktop.com/classes-near-me/blog/react-vs-angular.
15. T Kumpulainen. Web application development with Vue. Js, 2021.
16. M Nguyen. Full-stack crud application: User management system, 2023.
17. DD PANJALU. Pengembangan Aplikasi Requirement Change Management Berbasis Web dengan PERN Stack, 2022.
18. Railsware. Everything You Need to Know about Ruby on Rails Web Application Framework. [Електронний ресурс] – Режим доступу до ресурсу: https://railsware.com/blog/ruby-on-rails-guide/.
19. DEV. Overview of the React.js framework. [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/mahmoudessam/overview-of-the-reactjs-framework-4ieg.
20. M Casciaro, L Mammino. Node. js Design Patterns: Design and implement production-grade Node. js applications using proven patterns and techniques, 2020.

21. DEV. Introduction to ExpressJs. [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/fredabod/introduction-to-expressjs-2nh4.
22. A Chauhan. A review on various aspects of MongoDb databases. Int. J. Eng. Res. Sci. Technol, 2019.

ДОДАТОК

Home.jsx

```
import { useEffect, useState } from 'react';
import { Link } from 'react-router-dom';
import { Swiper, SwiperSlide } from 'swiper/react';
import { Navigation } from 'swiper/modules';
import SwiperCore from 'swiper';
import 'swiper/css/bundle';
import ListingItem from '../components/ListingItem';

export default function Home() {
  const [offerListings, setOfferListings] = useState([]);
  const [saleListings, setSaleListings] = useState([]);
  const [rentListings, setRentListings] = useState([]);
  SwiperCore.use([Navigation]);
  console.log(offerListings);
  useEffect(() => {
    const fetchOfferListings = async () => {
      try {
        const res = await fetch('/api/listing/get?offer=true&limit=4');
        const data = await res.json();
        setOfferListings(data);
        fetchRentListings();
      } catch (error) {
        console.log(error);
      }
    };
    const fetchRentListings = async () => {
      try {
```

```
      const res = await fetch('/api/listing/get?type=rent&limit=4');
      const data = await res.json();
      setRentListings(data);
      fetchSaleListings();
    } catch (error) {
      console.log(error);
    }
  };

  const fetchSaleListings = async () => {
    try {
      const res = await fetch('/api/listing/get?type=sale&limit=4');
      const data = await res.json();
      setSaleListings(data);
    } catch (error) {
      log(error);
    }
  };
  fetchOfferListings();
}, []);
return (
  <div>
    {/* top */}
    <div className='flex flex-col gap-6 p-28 px-3 max-w-6xl mx-auto'>
      <h1 className='text-slate-700 font-bold text-3xl lg:text-6xl'>
        Find your next <span className='text-slate-500'>perfect</span>
        <br />
        place with ease
      </h1>
      <div className='text-gray-400 text-xs sm:text-sm'>
```

```
            Sahand Estate is the best place to find your next perfect place to
            live.
            <br />
            We have a wide range of properties for you to choose from.
          </div>
          <Link
            to={'/search'}
            className='text-xs sm:text-sm text-blue-800 font-bold
hover:underline'
          >
            Let's get started...
          </Link>
        </div>

        {/* swiper */}
        <Swiper navigation>
          {offerListings &&
            offerListings.length > 0 &&
            offerListings.map((listing) => (
              <SwiperSlide>
                <div
                  style={{
                    background: url(${listing.imageUrls[0]}) center no-repeat,
                    backgroundSize: 'cover',
                  }}
                  className='h-[500px]'
                  key={listing._id}
                ></div>
              </SwiperSlide>
            ))}
```

```
            </Swiper>

            {/* listing results for offer, sale and rent */}

          <div className='max-w-6xl mx-auto p-3 flex flex-col gap-8 my-10'>
              {offerListings && offerListings.length > 0 && (
                <div className=''>
                  <div className='my-3'>
                    <h2 className='text-2xl font-semibold text-slate-600'>Recent offers</h2>
                    <Link className='text-sm text-blue-800 hover:underline' to={'/search?offer=true'}>Show more offers</Link>
                  </div>
                  <div className='flex flex-wrap gap-4'>
                    {offerListings.map((listing) => (
                      <ListingItem listing={listing} key={listing._id} />
                    ))}
                  </div>
                </div>
              )}
              {rentListings && rentListings.length > 0 && (
                <div className=''>
                  <div className='my-3'>
                    <h2 className='text-2xl font-semibold text-slate-600'>Recent places for rent</h2>
                    <Link className='text-sm text-blue-800 hover:underline' to={'/search?type=rent'}>Show more places for rent</Link>
                  </div>
                  <div className='flex flex-wrap gap-4'>
                    {rentListings.map((listing) => (
```

```
                    <ListingItem listing={listing} key={listing._id} />
                  ))}
                </div>
              </div>
            )}
            {saleListings && saleListings.length > 0 && (
              <div className="">
                <div className='my-3'>
                  <h2 className='text-2xl font-semibold text-slate-600'>Recent places for sale</h2>
                  <Link className='text-sm text-blue-800 hover:underline' to={'/search?type=sale'}>Show more places for sale</Link>
                </div>
                <div className='flex flex-wrap gap-4'>
                  {saleListings.map((listing) => (
                    <ListingItem listing={listing} key={listing._id} />
                  ))}
                </div>
              </div>
            )}
          </div>
        </div>
      );
    }
```