МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА САЙТУ ДЛЯ СТВОРЕННЯ ПІКСЕЛЬНИХ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ REACT»**

Виконав: студент 2 курсу, групи 8.1211-1-іпз
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
М.А. Давидов
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, к.ф.-м.н. Мильцев О.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2022 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Давидову Максиму Андрійовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка сайту для створення піксельних зображень з використанням React

керівник роботи Мильцев Олександр Михайлович, к.ф.-м.н.
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 04 » травня 2022 року № 500-с

2. Строк подання студентом роботи 01.12.2022

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік питань до дослідження.
3. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Огляд літератури стосовно фреймворку React.
2. Проєктування вебдодатка.
3. Розробка вебдодатку.
4. Аналіз отриманих результатів.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 04.05.2022

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	20.05.2022	
2.	Обробка теоретичних джерел.	01.06.2022	
3.	Ознайомлення з засобами	08.07.2022	
	автоматизованого тестування.		
4.	Розробка першого розділу.	26.07.2022	
5.	Ознайомлення з обраним вебдодатком.	12.08.2022	
6.	Написання набору тестових випадків.	26.08.2022	
7.	Проєктування інформаційної системи.	17.09.2022	
8.	Розробка другого розділу.	26.09.2022	
9.	Написання коду автотестів.	07.10.2022	
10.	Розробка третього розділу.	15.11.2022	
11.	Оформлення та нормоконтроль	24.11.2022	
	кваліфікаційної роботи магістра.		
12.	Захист кваліфікаційної роботи.	15.12.2022	

Студент _____ М.А. Давидов
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.М. Мильцев
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка сайту для створення піксельних зображень з використанням React»: 51 с., 26 рис., 1 табл., 12 джерел, 2 додатки.

БРАУЗЕР, ВЕБДОДАТОК, ІНФОРМАЦІЙНІ СИСТЕМИ, САЙТ, CSS, JAVASCRIPT HTML, PHP, REACT.

Об’єкт дослідження: розробка сайту з використанням React.

Мета роботи: розробити вебсайт за допомогою React для надання послуг.

Методи дослідження: аналіз предметної області, вивчення та узагальнення, моделювання.

У кваліфікаційній роботі досліджено предметну область, засоби реалізації вебдодатку сервісу-редактора піксельних графічних зображень, обрано найкращий засіб реалізації та розроблено вебдодаток. Розглянуто основні особливості PHP та JavaScript та React, реалізовано базу даних у MySQL.

Клієнтська та серверна частини були розділені з використанням різних технологій, серверна частина реалізована з використанням PHP працює по принципу запит – відповідь, реалізований зручний інтерфейс.

SUMMARY

Master’s qualifying paper «Development of the Website for Creating Pixelated Images using React»: 51 p., 26 figures, 1 table, 12 references, 2 supplements.

BROWSER, CSS, INFORMATION SYSTEMS, HTML JAVASCRIPT, REACT, SITE, PHP, WEB-APPLICATION.

Research object: website development using React.

Job objective: develop a website using React to provide services.

Research methods: analyze subject areas, study and generalization, modeling.

In the qualification work, the subject area, means of implementation of the web application of the pixel graphic image editor service were investigated, the best means of implementation was chosen and the web application was developed. The main features of PHP and JavaScript and React were considered, and the MySQL database was implemented.

The client and server parts were separated using different technologies, the server part is implemented using PHP, works on the request-response principle, and a user-friendly interface is implemented.

ЗМІСТ

ВСТУП

Інженерія програмного забезпечення застосовує сучасні методології та інструментарій проєктування та розробка програмного забезпечення для розробки додатків, що можуть використовуватися в будь-якій сфері людського життя.

Наразі існує багато діджитал агентств, які займаються комерційною розробкою. Вони використовують сучасні методології розробки програмного забезпечення. Програмні інженери цих компаній визначають, які технології доцільно використовувати на поточних проектах, також використовуються вебфреймворки для розробки з використанням архітектурної моделі MVC з використанням REST.

Мета роботи є розробка вебсайту за допомогою React для створення піксельних графічних зображень. Тобто спроектувати, розробити та протестувати роботу вебдодатку.

Робота складається із трьох розділів: огляд сучасного становища проблеми; робота з вебфреймворком React; розробка вебдодатку “Розробка сайту для створення піксельних зображень”.

Непоганим попитом останнім часом користується JavaScript. Особливо це стосується вебутиліт. Ця мова програмування досить проста в освоєнні – з ним зможе впоратися навіть новачок. Має різноманітні інструменти та компоненти, що допомагають швидко писати складне ПЗ. Результат не забариться – програміст отримає на виході якісний софт зі зрозумілим інтерфейсом і потужним функціоналом.

JS має чимало фреймворків та бібліотек. Розробникам, які претендують на звання Джуніорів, слід розібратися з таким компонентом, як React. Ця «утиліта» знадобиться в ПЗ, де потрібно впровадити інтерфейс користувача. Далі технологія буде розглянута докладніше.

JavaScript – це проста об'єктно-орієнтована мова програмування, яка призначається для розробки клієнт-серверних утиліт. Софт, складений з його допомогою, включаються в HTML-документи, після чого поширюється разом з ними. JS виступає як інтерпретований мови програмування [1]. Застосовується як вбудована мова для забезпечення програмного доступу до об'єктів контенту.

До його особливостей відносять:

1) відносно простий синтаксис;
2) високу читальність програмного коду;
3) потужний інструментарій та функціонал;
4) наявність фреймворків та бібліотек майже на всі випадки життя;
5) гідний рівень безпеки.

Звичайний JavaScript є стандартною мовою вебсайтів і вебдодатків. Крім того, що він може працювати в будь-якому браузері, він також може використовуватися для рендерингу на стороні сервера та складної анімації або взаємодії користувача в програмах. Користувач зможе досить швидко створювати браузерні та клієнт-серверні утиліти через JS. Особливо якщо працює React.

React – це елемент програмування. Спеціалізована технологія, що використовується JavaScript для розробки різноманітного контенту.Є бібліотекою для створення інтерфейсів користувача. Має відкритий вихідний код. Вперше React з'явився у 2013 році. Виступає як кросплатформова бібліотека.

React можна назвати:

1) вебфреймворком;
2) бібліотекою функцій;
3) бібліотекою JS.

Зараз цей компонент має підтримку, а також розробляється Facebook та Instagram. Їм допомагає спільнота незалежних розробників та організацій.

1 ТЕХНІЧНЕ ЗАВДАННЯ

1.1 Характеристика платформи

Платформа для малювання пікселями дозволяє створювати растрові зображення, редагувати видаляти та пересилати їх. Піксельні зображення є частиною комп'ютерної графіки, яка має справу зі створенням, обробкою та зберіганням растрових зображень. Растрове зображення є масивом кольорових точок (пікселів). Обробка растрової графіки здійснюється растровими графічними редакторами. Растрові зображення зберігаються у різних графічних форматах.

Такий вебдодаток має містити графічний редактор за допомогою якого буде створюватися зображення. Також інструменти для малювання такі як олівець, ластик функція заливання поверхі кольором, функція відміни останньої дії. Такий вебсайт повинен включати панель управління та клієнтської частину.

1.2 Вимоги до програмного забезпечення

Функціональні вимоги до панелі адміністратора. Повинна бути авторизація. Можливість зайти користувачу, чиї дані є в базі даних також можливість змінити дані адміністратора. Адміністратор може змінити свої особисті дані. Редагування медіа матеріалів. Тобто можливість подивитися усі картинки, додати ще, змінити дані картинки або видалити її [2].

Також додати можливість сортувати зображень по тегу. Потрібна можливість видалити, додати, змінити та переглянути список тегів.

На сайті усі зображення розділені на категорії (Автор, оцінка, Тип). Тому потрібна можливість додання, зміни, видалення та перегляду цих тегів. Можливість додавання, редагування, видалення та перегляду зображень.

Функціональні вимоги до клієнтської частини сайту, а саме робочий редактор зображень, можливість перегляду усіх наявних зображень та сортувати їх за тегом.

При переході на сторінку каталогу, користувач має змогу передивитися усі товари з помітками про категорію та теги цього товару.

Також клієнт може створити обліковий запис, тому потрібно розробити особистий кабінет. В кабінете потрібно зберігати інформацію про всі створені ним зображення.

До нефункціональних можна віднести стандарти інтерфейсу користувача. Зручний та красивий вебсайт приверне більше клієнтів.

Інтеграція з іншими сервісами задля розширення послуг. Це можуть бути будь-які сервіси, що пропонують услуги зв’язані із продажом зображень або навчальні платформи.

1.3 Огляд наявних сервісів

Для зрозуміння необхідних функцій було розглянуто сайт Gravit Designer та DrawSVG. З точки зору функціоналу ці сайти дуже відрізняються. Спільні функції в них є, тому треба їх перелічити:

1) редактор зображень з інструментами;
2) перегляд усіх зображень;
3) фільтрація за категоріями або тегами;
4) реєстрація облікового запису;
5) особистий кабінет зі створеними зображеннями.

Отже треба реалізувати такий самий функціонал. Редактор зображень та каталог зображень – головні сервіси додатку. Клієнт зможе обрати та

купити зображення з наявних або створити нове зображення. Сьогодні особистий кабінет наявний у кожному додатку, ці дані допомагають персоналізувати контент для клієнта, тим самим підвищується залученість клієнтів.

1.4 Аналіз технології

Неспадна популярність мови JavaScript породжує велику кількість нових інструментів для швидшої та зручнішої розробки, що ускладнює процес вибору. На сьогодні не існує єдиного, універсального фреймворку, лише фаворити серед розробників, які мають декілька відмінних один від одного принципів роботи.

Світ JavaScript – це середовище, що включає широкий вибір інструментів для розробки, бібліотек та фреймворків. Проте з великою кількістю варіантів виникає багато плутанини, особливо для новачків. На сьогодні існує близько тридцяти JavaScript-фреймворків, що орієнтовані на розробку вебзастосунків. Вони є важливою частиною сучасної фронтенд-розробки та забезпечують програмістів надійними інструментами для створення масштабованих інтерактивних додатків. Також все більше компаній використовують їх для своїх проектів, а отже знання найпопулярніших фреймворків стає вимогою для багатьох вакансій. Популярність розробників інтерфейсів користувача не спадає, а мова JavaScript вже 8 років знаходиться на першому місці серед запитів на платформі Stackoverflow, що приваблює початківців у сфері IT, а значить питання вибору найзручнішого інструменту виникає постійно та робить дану тему дослідження актуальною.

Сьогодні PHP одна з найпоширеніших мов програмування, що використовуються в web-розробці. Також є багато спорів, що розробка додатку за використанням PHP не дуже вдала думка, тому що проєкт важко

підтримувати та додавати нові функції, а сам код не структурований та дуже складний для розуміння.

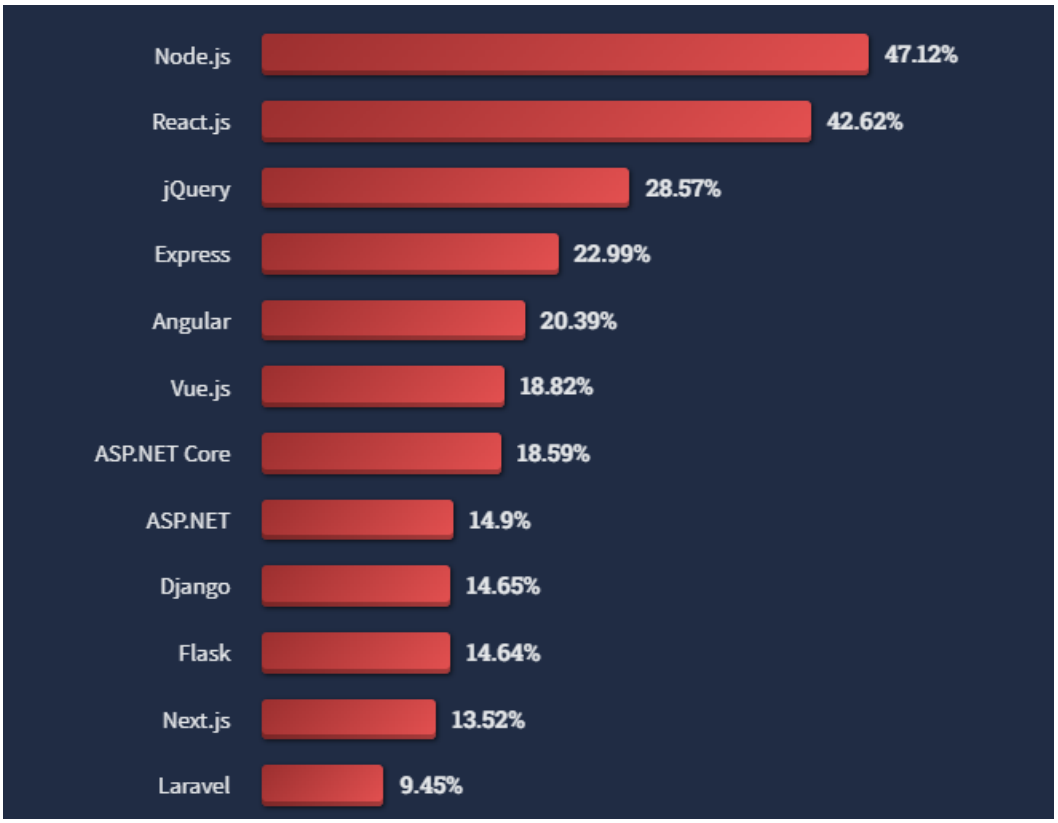

Рисунок 1.1 – Рейтинг 2022 Developer Survey

За статистикою на рисунку 1.1, більш ніж 78 % сайтів розроблено з використанням PHP [12]. Він застосовується для створення сайтів і web-додатків будь-якої складності. від блогів до інтернет-магазинів і браузерних ігор. Його переваги – в широких можливостях і захищеності завдяки закритому вихідного коду [3].

Laravel – це безкоштовний PHP фреймворк загального призначення з відкритим кодом, який з'явився на світ порівняно недавно – в 2011 році, але, завдяки стрімким темпам розвитку і величезної армії шанувальників, сьогодні він є одним з найпопулярніших PHP фреймворків.

Розробка сайту на фреймворке Laravel не сильно відрізняється від інтернет-ресурсів на PHP і при цьому має ряд переваг:

– широкий і різноманітний функціонал;

– можна створювати масштабні інтернет-проєкти, незалежно від складності і спрямованості, у тому числі і багаторівневі вебсайти;
– дозволяє вирішувати самі нестандартні завдання.

Якщо порівнювати з CMS системами, то розробка з використанням WordPress або Drupal, буде значно дешевше та швидше. Задля цього треба найняти одного розробника, який за декілька неділь виконає проєкт. Але, з іншого боку, якщо буде наявність розширювати проєкт, або велика кількість користувачів, то через деякий час потрібно буде розробляти новий додаток з нуля. Отже вигідніше одразу використовувати технології, які в подальшому буде дешевше та легше підтримувати [4].

React має декілька недоліків, що потрібно вказати тут. Він постачається з великою кількістю шаблонного коду, який не потрібен для більшості програм. Іншим недоліком React є те, що він не підтримує IE11 або старіші версії. Це може бути проблематично, якщо ваша програма створена виключно для браузерів IE11 або старіших. Нарешті, є недоліки у використанні React, якщо ви хочете використовувати його на інших платформах, таких як мобільні програми та вебсайти. Хоча ви можете використовувати React на цих платформах, ви не можете повторно використовувати весь той самий код і функції з вебпрограми з React Native або Progressive Web Apps (PWA) на будь-якій платформі, оскільки це абсолютно різні технології самі по собі.

React – це бібліотека JavaScript, яка допомагає в розробці інтерфейсів користувача. Він використовує парадигму функціонального програмування. Першим кроком у створенні програм React є написання розмітки для вашої програми, а потім рендеринг цієї розмітки у віртуальний DOM. Цей віртуальний DOM потім використовується як основа для алгоритму візуалізації React. З іншого боку, простий JavaScript не потребує жодних спеціальних бібліотек чи інструментів. Все, що вам потрібно зробити, це вказати, що ви хочете, і це буде відображено браузером. Ви можете використовувати звичайний JavaScript без жодних бібліотек чи інструментів

взагалі. Однак буде важко досягти певних функцій, які пропонує React, як-от обробка змін стану або використання прослуховувачів подій, де вам доведеться використовувати проксі-сервери, щоб вони працювали належним чином у звичайному JavaScript.

Також розглянемо платформу Angular. Angular – це фреймворк JavaScript із відкритим вихідним кодом, розроблений і керований командою Google Angular. Angular – найпопулярніший клієнтський фреймворк для розробки масштабованих і високопродуктивних мобільних і вебдодатків за допомогою HTML, CSS і TypeScript. Остання версія Angular – це Angular 13, яка пропонує корпоративні рішення для розробки вебдодатків.

Техніка MVC (Model View Controller) використовується Angular, яка розділяє роботу на логічні частини та прискорює початковий час завантаження вебсторінки.

Як показано в таблиці 1.1, AngularJS – це платформа з відкритим кодом, розроблена Google, а ReactJS – бібліотека з відкритим кодом, розроблена Facebook.

Angular JS – це структура вебдодатків на основі TypeScript, тоді як React JS – бібліотека на основі JavaScript. Angular – це фреймворк JS, створений за допомогою TypeScript, тоді як React JS – це бібліотека JS, створена за допомогою JSX.

React.js здебільшого використовується для створення інтерактивних компонентів інтерфейсу користувача з часто змінними даними, тоді як Angular.js використовується для створення складних корпоративних програм, таких як прогресивні вебпрограми та односторінкові програми. JS використовується для створення односторінкових програм за допомогою HTML і TypeScript. React JS зазвичай використовується для створення інтерфейсів користувача для односторінкових програм із ізольованих компонентів.

Таблиця 1.1 – Порівняльна таблиця React та Angular

Параметри	**Angular**	**React**
Розроблений	Google	Facebook
Дата релізу	2009	2013
Мова написання	TypeScript	JavaScript
Тип технології	Повноцінний фреймворк MVC, написаний на JavaScript	Бібліотека JavaScript (перегляд у MVC; для реалізації архітектури потрібен Flux)
Концепт	Переносить JavaScript у HTML. Працює зі справжнім DOM. Візуалізація на стороні клієнта	Вносить HTML у JavaScript. Працює з віртуальним DOM. Візуалізація на стороні сервера
Мова програмування	JavaScript + HTML	JavaScript + JSX
Візуалізація інтерфейсу користувача	Клієнт/сервер	Клієнт/сервер
Найкраще підходить для	Дуже активні та інтерактивні вебпрограми	Більші програми з повторюваними змінними даними
Структура додатку	Складний MVC	Компонентно-орієнтований
Тип DOM	Реальний	Віртуальний

Angular є частиною стеку MEAN і дуже сумісний з багатьма редакторами коду. Також розглядається розробка динамічних вебсайтів і вебдодатків. З іншого боку, React широко використовується для розробки повторно використовуваних елементів HTML для розробки інтерфейсу. І Angular, і React є найпопулярнішими інтерфейсними технологіями в

спільноті розробників. Вони пропонують кілька дивовижних переваг, але React працює краще, ніж Angular. Крім того, величезна спільнота розробників підтримує React JS. Оскільки React складається з Virtual DOM і оптимізує візуалізацію, він перевершує Angular у битві порівняння. Крім того, це дозволяє розробникам без проблем переходити між версіями React. На відміну від Angular, він має дуже легкий і простий процес встановлення. У двох словах, React пропонує багато переваг і надійних рішень для розробників, які збільшують час розробки та усувають помилки.

Отже, враховуючи ці відмінності було обрано React.

2 ПРОЄКТУВАННЯ ВЕБДОДАТКУ

2.1 Дослідження предметної області

Предметною областю даного проєкту є редактор зображень в піксельному стилі. Користувач повинен створити обліковий запис для роботи з додатком.

Функціональні можливості:

1) редактор зображень;
2) створення облікового запису;
3) перегляд наявних зображень;
4) редагування, видалення, приховання зображення.

Функціонал для панелі адміністратора:

1) можливість редагування, видалення та додавання малюнка;
2) можливість редагування, видалення та додавання користувачів;
3) можливість редагування, видалення та додавання тегів;
4) можливість додавання нових адміністраторів.

2.2 Діаграма прецедентів

Діаграма випадків використання – це графічне зображення можливих взаємодій між користувачем та системою. Схема зображує різне використання системи та різні типи користувачів. Варіанти використання представлені колом або еліпсом. Акторів часто зображують у вигляді фігурок.

Хоча сам випадок використання може детально вивчити кожну можливість, діаграма прикладів може допомогти проілюструвати загальний вигляд системи найвищого рівня. Хтось раніше говорив, що «схема

використання – це креслення вашої системи».

Завдяки спрощеному характеру, схема використання може бути хорошим інструментом комунікації для зацікавлених сторін. Ці креслення намагаються імітувати реальний світ і дати зрозуміти зацікавленим сторонам, як буде розвиватися система.

Мета використання діаграм – показати динамічні аспекти системи. Вони забезпечують спрощене графічне представлення того, що повинна робити система, коли вона використовується. Повна функціональність та технічний вигляд системи вимагають подальших схем та документації [5].

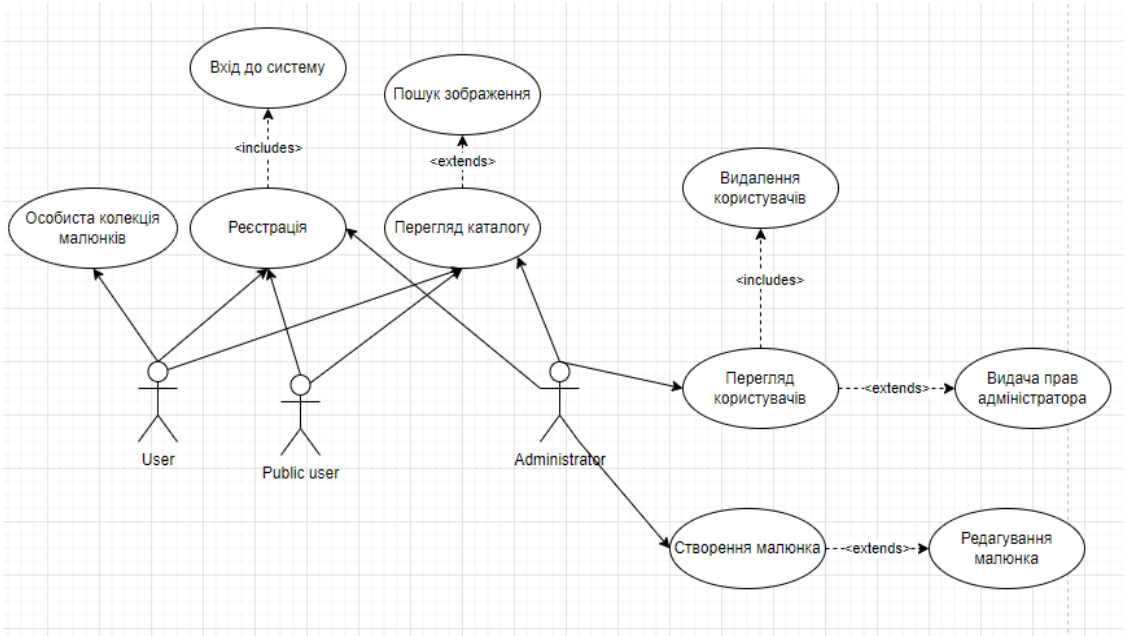

Рисунок 2.1 – Діаграма прецедентів

Для створення діаграми прецедентів було обрано трьох акторів, клієнта та адміністратора. Вони добре зображують дві частини сайту, оскільки клієнт може взаємодіяти тільки з клієнтською частиною сайту, а адміністратор має доступ до панелі керування. Він може створити малюнок, змінювати, видаляти його. Також, адміністратор має змогу додавати нового менеджера або змінювати свої поточні дані. На рисунку 2.1 зображена діаграма прецедентів з трьома акторами.

Прецедент *Реєстрація* – основна функція створення нового облікового

запису користувача в системі. Користувач, що немає облікового запису в системі може створити новий аккаунт для користування послугами системи. Користувач переходить на сторінку створення облікового запису, та надає необхідні данні, а саме:

1) електронна адреса;
2) пароль;
3) логін, для доступу до системи;
4) номер телефону для перевірки або відновлення облікового запису.

Після цього виконується первинна валідація полів, перевіряється відповідність паролів, перевіряється складність пароля, та корректність наданої електронної адреси. Після цього данні відправляються на сервер, де виконується остаточна валідація:

1) наявність електронної адреси;
2) наявність логіну користувача;
3) хешування паролю.

Користувач отримує повідомлення про успішний процес реєстрації або про помилку та інструкції що до її виправлення.

Прецедент *Вхід до системи* – користувач вводить логін та пароль від облікового запису. Додаток відправляє данні на сервер. Сервер виконує пошук за логіном у базі даних, якщо знаходить відповідний запис, то хешує наданий пароль та перевіряє його на ідентичність з наданим. Користувач отримує повідомлення про успішний процес автентифікації або про помилку та інструкції що до її виправлення. Якщо процес був успішно виконаний, система створює два токени доступу з інформацією про час вичерпання і ідентифікатор користувача, хешує їх та відправляє в додаток.

Прецедент *Перегляд каталогу* – завантажує на сторінку усі наявні зображення з бази даних.

Актор публічний користувач, ще не пройшов реєстрацію, або не зайшов до свого облікового запису. Він може зареєструватися або увійди до

системи, переглянути каталог авто або знайти за допомогою фільтрів конкретний малюнок.

Наступний актор, вже пройшов реєстрацію або увійшов до свого облікового запису. Йому стала доступна історія транзакцій та робіт. Тобто він може переглянути, які дії були з цим записом. Також він може подивитися каталог.

Адміністратор системи має усі можливості що і попередні актори. Окрім цього, він має доступ до контенту системи. Адміністратор може створити новий малюнок, додати тег, змінити або видалити їх. Також він може переглянути всіх користувачів, видалити обліковий запис користувача чи підвищити його статус у системі.

2.3 Функціональне проєктування

Перше, на що слід звернути увагу, це функціональне моделювання. Воно представлене методологією IDEF0. Суть IDEF0 полягає у побудові ієрархічної системи діаграм. Іншими словами, один опис для кожної діаграми буде частиною системи. Щоб побудувати контекстну діаграму IDEF0, на початку потрібно підкреслити, що проект має певні ресурси та матеріали для виконання.

Спочатку клієнт обирає створює зображення, та завантажує його в систему. Наступним кроком буде перевірка малюнка на відповідність до умов використання сервісу. Якщо менеджер немає питань, то малюнок може бути опублікований на сайті. Клієнт може заплатити кошти, щоб приховати малюнок, або завантажити його в бажаному форматі.

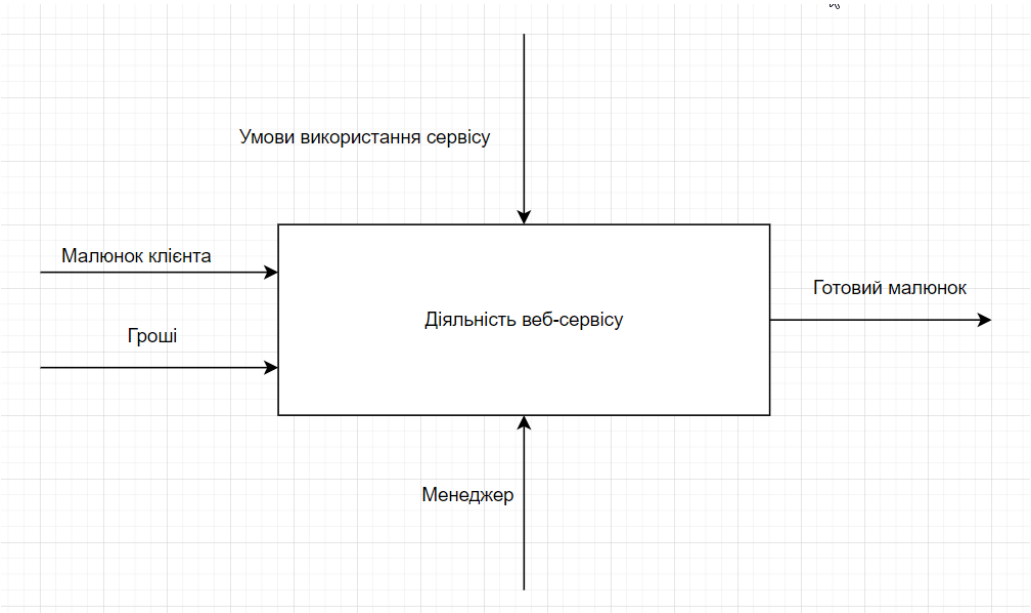

Рисунок 2.2 – Діаграма IDEF0

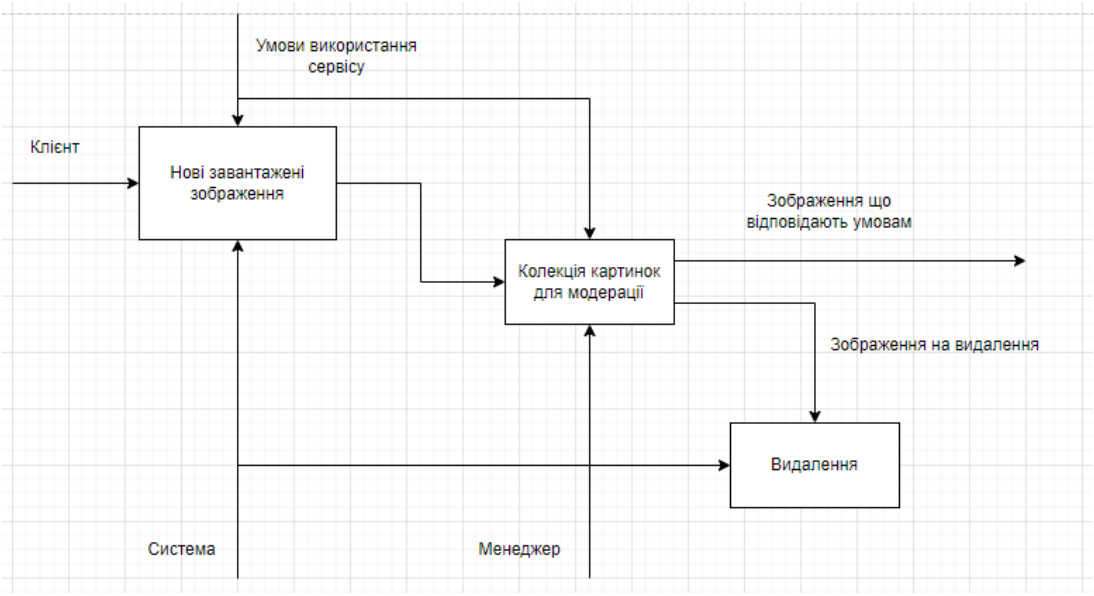

Рисунок 2.3 – Діаграма IDEF0

2.4 Діаграма огляду взаємодії

Діаграма огляду взаємодії подібна до діаграми діяльності, оскільки обидві візуалізують послідовність дій. Різниця полягає в тому, що для перегляду взаємодії кожна окрема діяльність описується як кадр, який може містити вкладену діаграму взаємодії. Це дозволяє використовувати оглядову

діаграму взаємодії для «деконструкції складних сцен, інакше її потрібно пояснити різними способами, хоча б лише як одну діаграму послідовності».

Для створення діаграми послідовності спочатку потрібно визначити об’єкти, які обмінюються інформацією. Встановлено два об’єкти. Перший – це клієнт, він створює малюнки та прибуток сервісу. Другий – адміністратор за допомогою панелі управління, який обробляє роботи від клієнтів.

Послідовність починається, коли користувач відвідує вебсайт, а після презентації каталогу, виконаної сервісом, є можливість відфільтрувати роботи або зробити нову. Сайт формує малюнок і відсилає його менеджеру, менеджер перевіряє його та підтверджує.

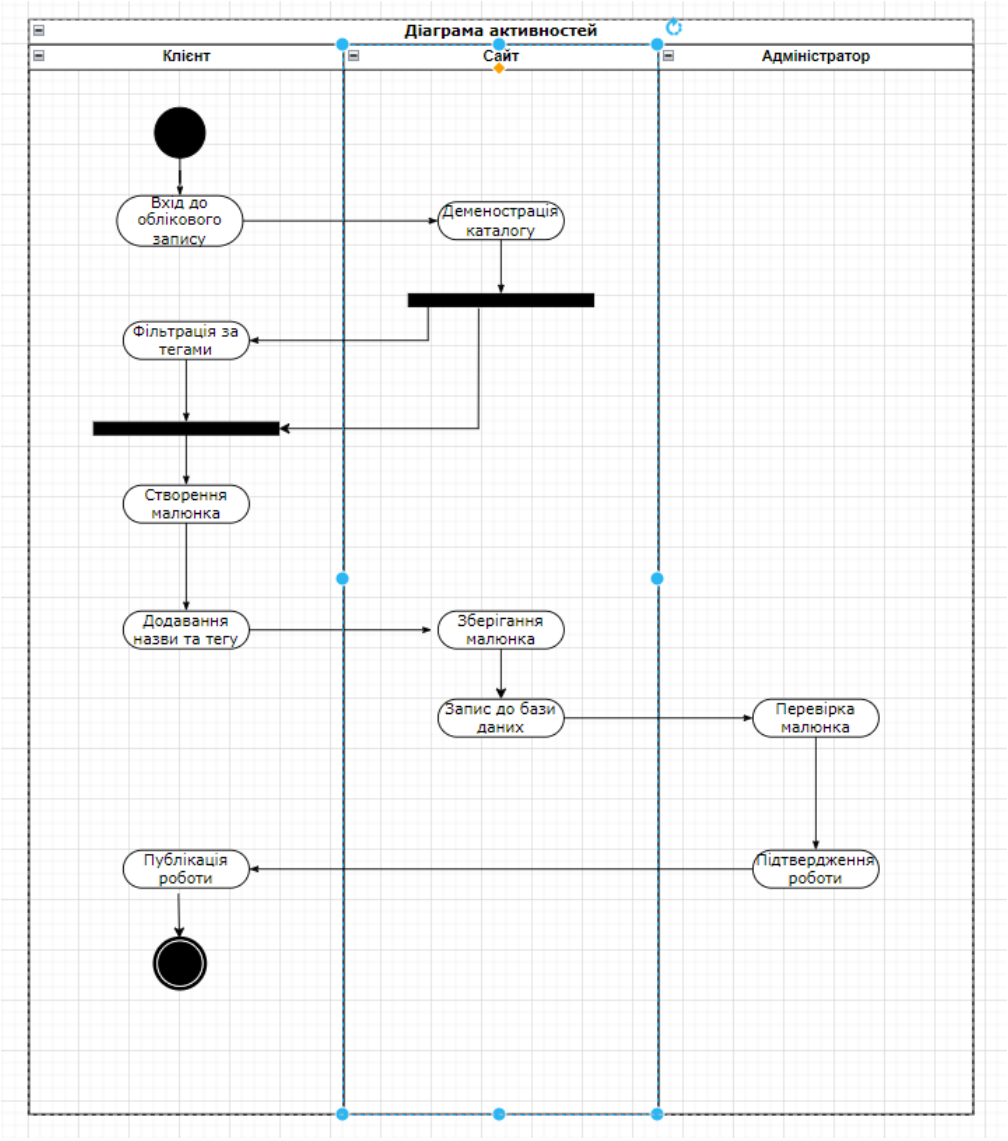

Рисунок 2.4 – Діаграма огляду взаємодії

2.5 Концептуальне проєктування інформаційної моделі

Модель «сутність-взаємозв'язок» або ER-модель описує взаємопов'язані речі, які представляють інтерес для певної галузі знань. Базова модель ER складається з типів сутностей, які класифікують цікаві речі та визначають можливі взаємозв'язки між сутностями.

У програмній інженерії моделі ER зазвичай формуються для представлення речей, про які компанії повинні пам’ятати під час виконання бізнес-процесів. Тому модель ER стає абстрактною моделлю даних, яка визначає структуру даних або інформації, яка може бути реалізована в базі даних. На рисунку 2.5 зображено кінцеву ER-діаграму.

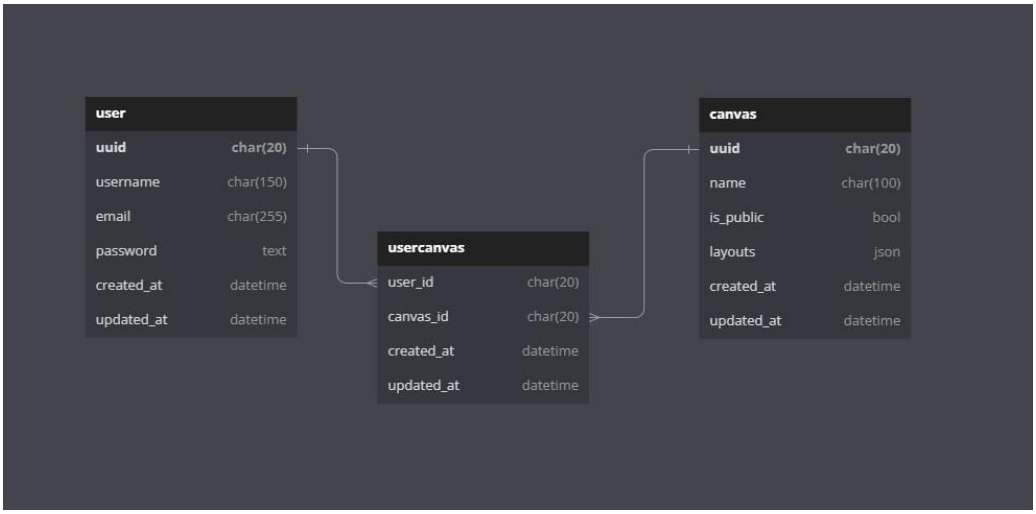

Рисунок 2.5 – ER-діаграма

Для створення ER-діаграми, потрібно виділити основні сутності проєкту. Треба почати з сутності продукту. Ця сутність є фундаментальною для системи. Вона має первинний ключ uuid, ім’я, параметр доступу, шари, час створення, час останнього оновлення та тег.

У сутності Users стандартний набір атрибутів, в саме: uuid, ім’я, електронна пошта та пароль [6].

Usercanvas зберігає відповідності між користувачами та малюнками.

2.6 Діаграма класів

Зараз майже всі поля програмування застосовують об'єктно-орієнтований підхід. Перед початком розробки, щоб уникнути недбалих проблем, необхідно детально спланувати класи, їх поля та методи та реалізувати все відповідно до плану. Реалізацію діаграми класів можна детально розглянути на рисунку 2.6.

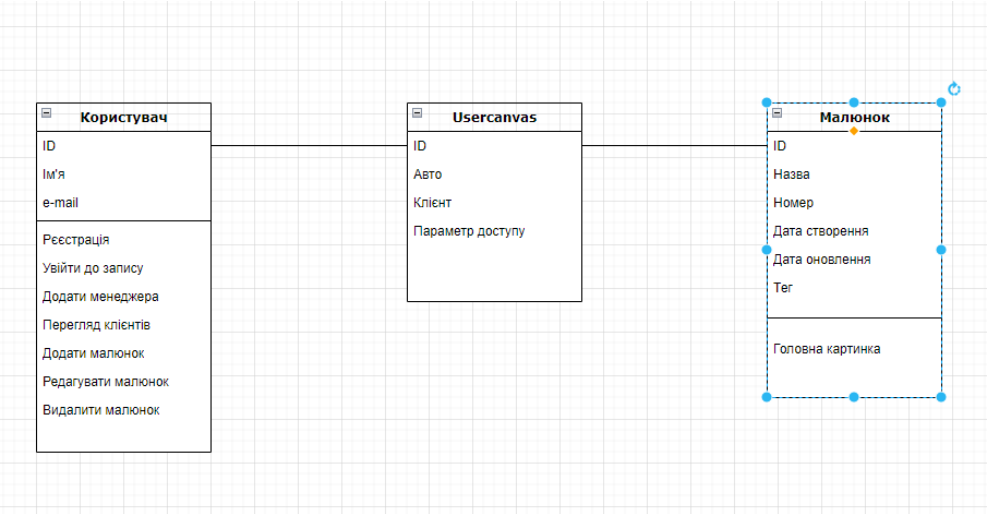

Рисунок 2.6 – Діаграма класів

2.7 Діаграма розгортання

Схема розгортання на уніфікованій мові моделювання імітує фізичне розгортання артефактів на вузлах. Наприклад, для опису вебсайту діаграма розгортання показує, які апаратні компоненти («вузли») існують, а також які програмні компоненти («артефакти») працюють на кожному з вузлів та способи підключення різних частин. Вузли відображаються як вікна, а артефакти, призначені кожному вузлу, відображаються у вікні у вигляді прямокутників [7]. Вузли можуть мати дочірні вузли, що відображаються як вкладені поля. Один вузол у режимі розгортання може концептуально

представляти кілька фізичних вузлів.

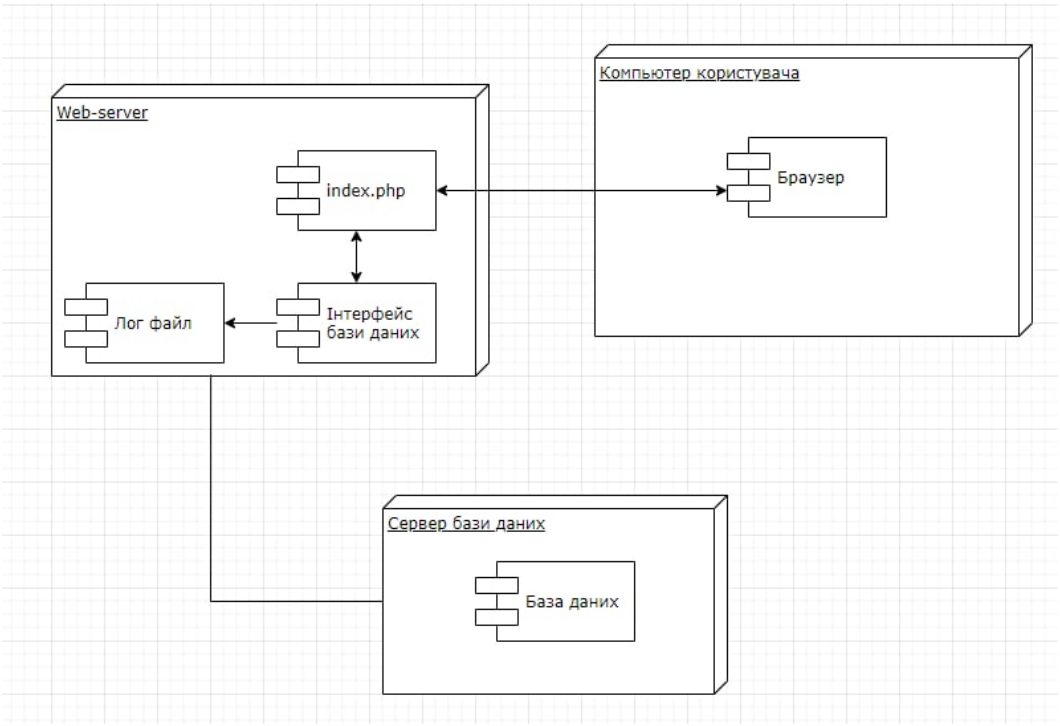

Рисунок 2.7 – Діаграма розгортання

Схема компонентів потрібна, щоб показати склад програмних компонентів. Щодо цього проєкту, схема, подана на малюнку 2.8, дуже проста. Існує сервер, на якому зберігаються дані, передані через захищену лінію зв'язку, а комп'ютер користувача отримує та відображає дані.

У порівнянні з раніше розглянутими діаграмами, компонентні діаграми описують характеристики фізичного представлення системи. Схема компонентів дозволяє визначити архітектуру розробленої системи та встановити взаємозв'язки між програмними компонентами, які можуть виступати як вихідний код, двійковий код та виконуваний код. У багатьох середовищах розробки один модуль або компонент відповідає одному файлу. Стрілками, що з'єднують модулі, вказуються взаємозалежні зв'язки, подібні до тих, що з'являються при компіляції вихідного коду. Основними графічними елементами діаграми компонентів є компоненти, інтерфейси та їх залежності.

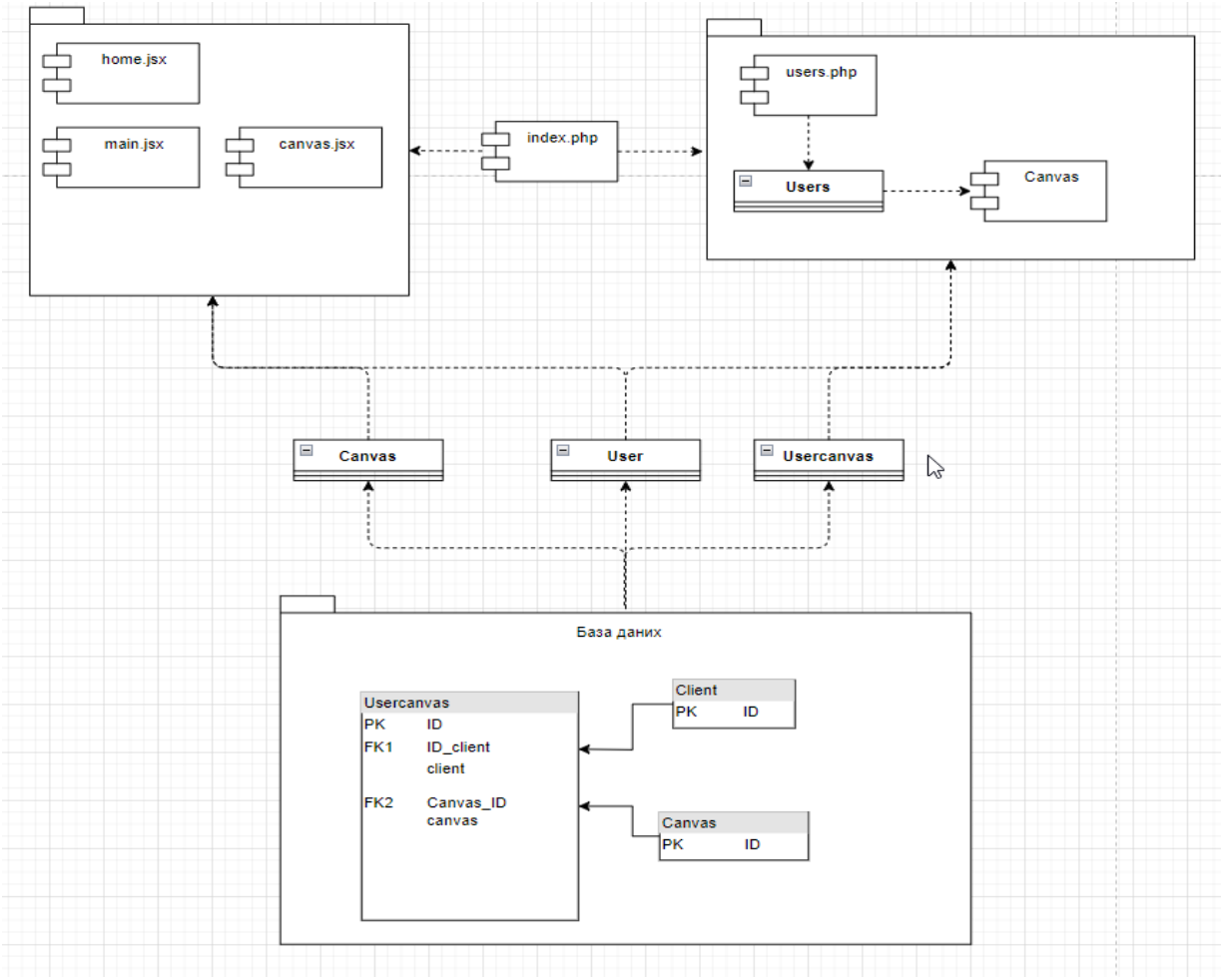

Рисунок 2.8 – Діаграма компонентів

3 РОЗРОБКА ВЕБДОДАТКУ

3.1 Налаштування середовища для розробки проєкту

Задля комфортності розробки було обрано Open Server, як середу для розроблення. Open Server Panel – це програмне середовище, розроблене для створення вебдодатків. Програмний пакет має багатий набір серверного програмного забезпечення, зручний, багатофункціональний та досконалий інтерфейс, а також потужні можливості управління компонентами та можливості конфігурації [8]. Панель Open Server (див. рис. 3.1) широко використовується для розробки, налагодження та тестування вебпроєктів, а також для надання вебслужб у локальній мережі.

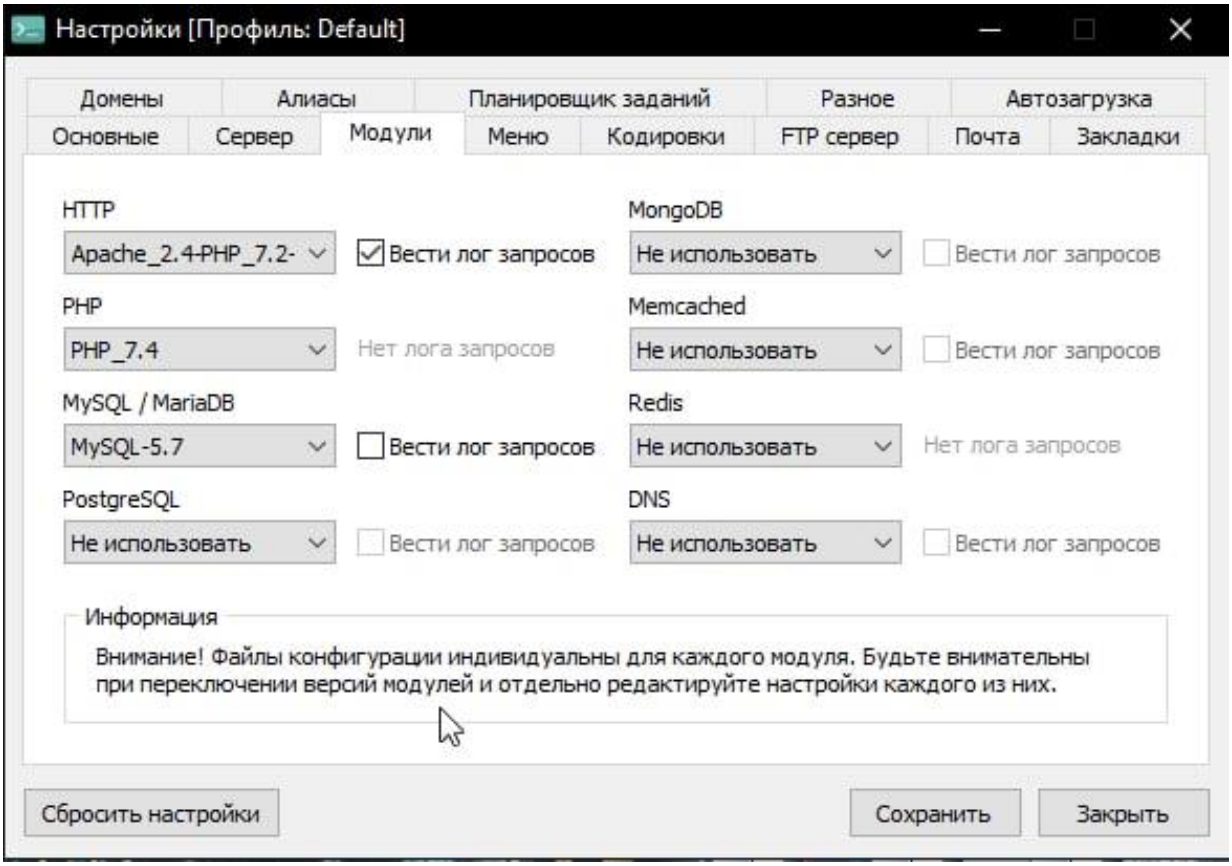

Рисунок 3.1 – Налаштування середи розробки

Як зображено на рисунку 3.1, для цього проєкту було обрано PHP 7.4 та MySQL 5.7.

Наступним кроком є встановлення Composer у наш проєкт, так как ми працюємо з Laravel, то цей пакетний менеджер для PHP забезпечує стандартний формат для управління залежностями у програмному забезпеченні та необхідними бібліотеками.

Щоб встановити Composer, потрібно запустити наступний скрипт з офіційного сайту, який зображено на рисунку 3.2.

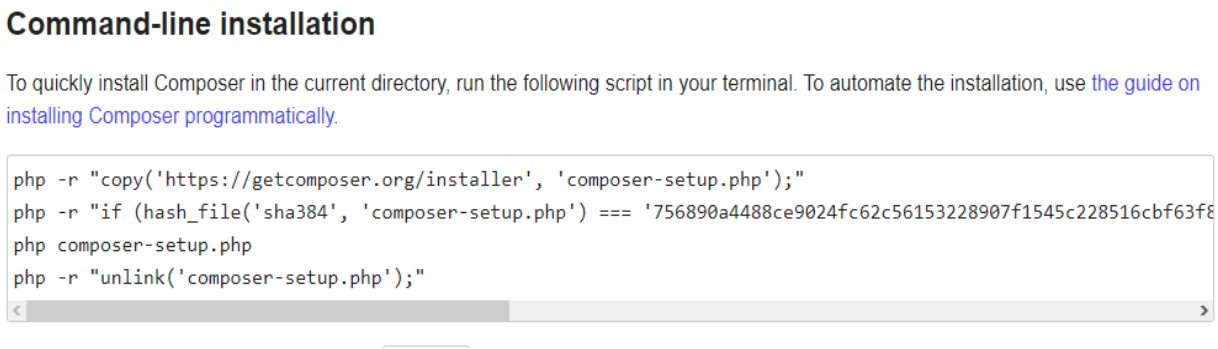

Рисунок 3.2 – Скрипт для встановлення Composer

Тепер потрібно встановити Laravel. За допомогою Composer, можна створити новий проєкт і одразу встановити Laravel. Для цього нам потрібно запустити скрипт як на рисунку 3.3.

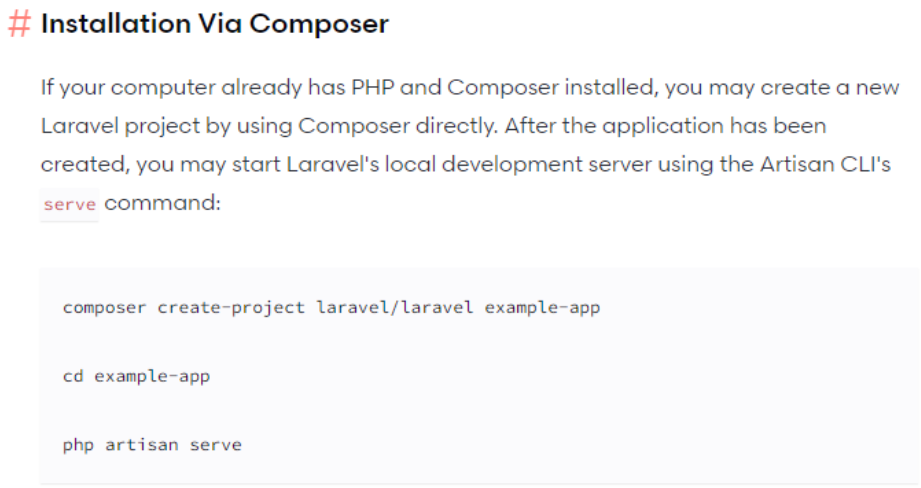

Рисунок 3.3 – Створення проєкту та встановлення Laravel

Для використання React у вебдодатку потрібно скомпілювати файли реакт в декілька основних файлів. До складу яких входять стилі, логіка компонентів, шаблони, в окремій папці публічні файли такі як лого, зображення, фавікон. Ці файли будуть використані для загрузки сторінки.

3.2 Файлова структура проєкту

Оскільки React – це бібліотека, ніхто не диктує вам правила про організацію та структурування проекту. Це дає свободу вибору підходів та подальшої адаптації. Але треба зазначити, що React має свою архітектуру файлів проєкту. Структура React-додатку за замовчуванням – відмінна відправна точка як для великих, так і для маленьких додатків. Отже React накладує обмеженя на те, де буде розміщений який-небудь клас.

Папка src містить компоненти які рендерять сторінку або її частину, також тут знаходиться логіка цих компонентів.

Папка Node modules зберігає усі залежності додаткових бібліотек та їх пакети.

Index.js – точка входу додатку, яка компілює усі файли проекту і монтується у головний html файл де виконується рендер сторінок.

Папка database містить міграції і класи для наповнення початковими даними БД.

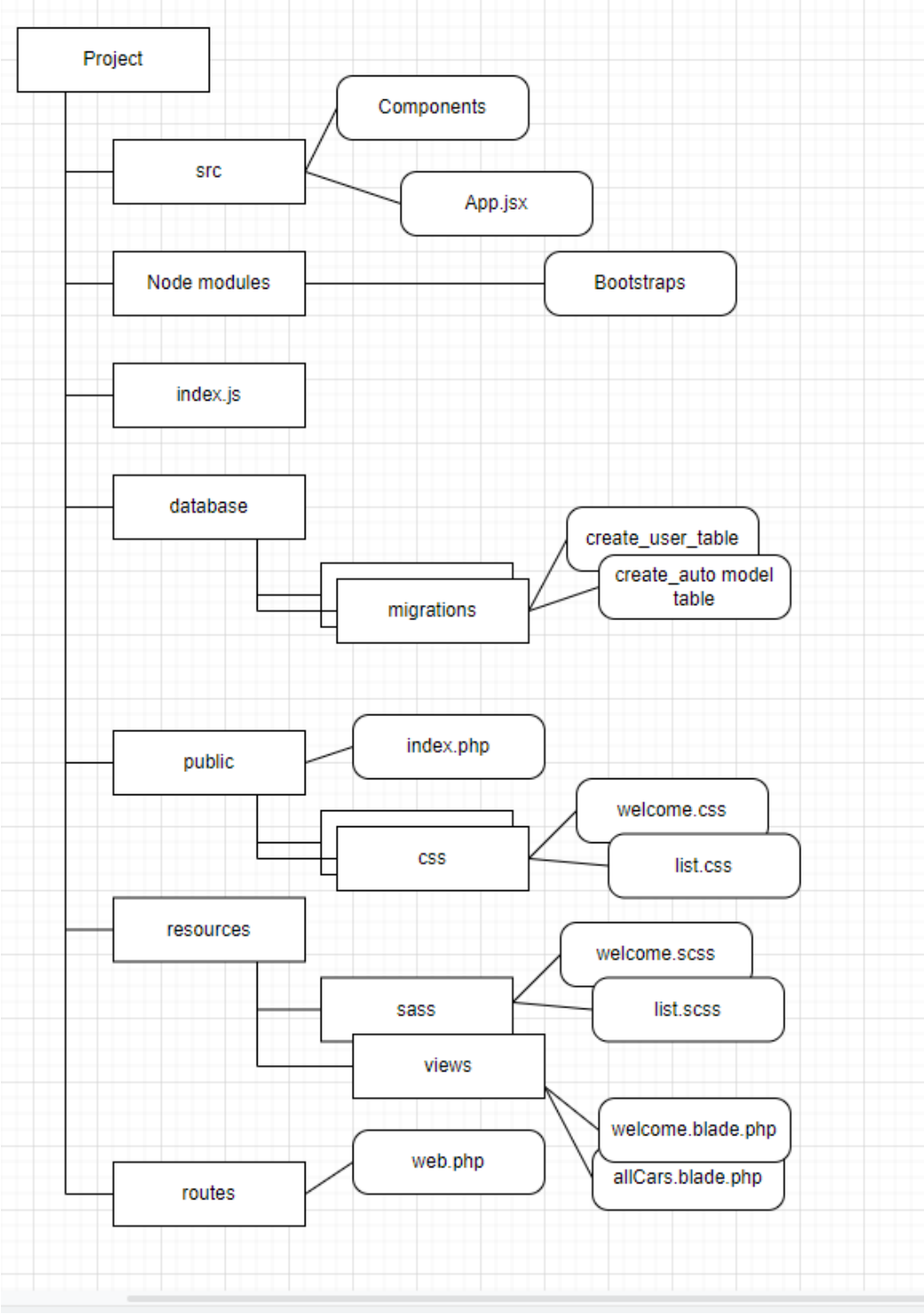

Рисунок 3.4 – Файлова структура додатку

3.3 Розробка дизайну

Проаналізувавши наявні популярні сервіси, було з'ясовано які сторінки потрібні для створення успішного вебсайту. Сайт буде складатися з

декількох сторінок. На сайті буде наявне головне меню, форма зворотного зв'язку, каталог, персональний кабінет. На рисунку 3.5 представлено зовнішній вигляд [9].

Рисунок 3.5 – Карта проєкту

На рисунку 3.6 зображено голову сторінку проєкту. Щоб переглянути наявні роботи потрібно натиснути кнопку "Collection". Щоб перейти до редактору, потрібно натиснути кнопку "Open editor".

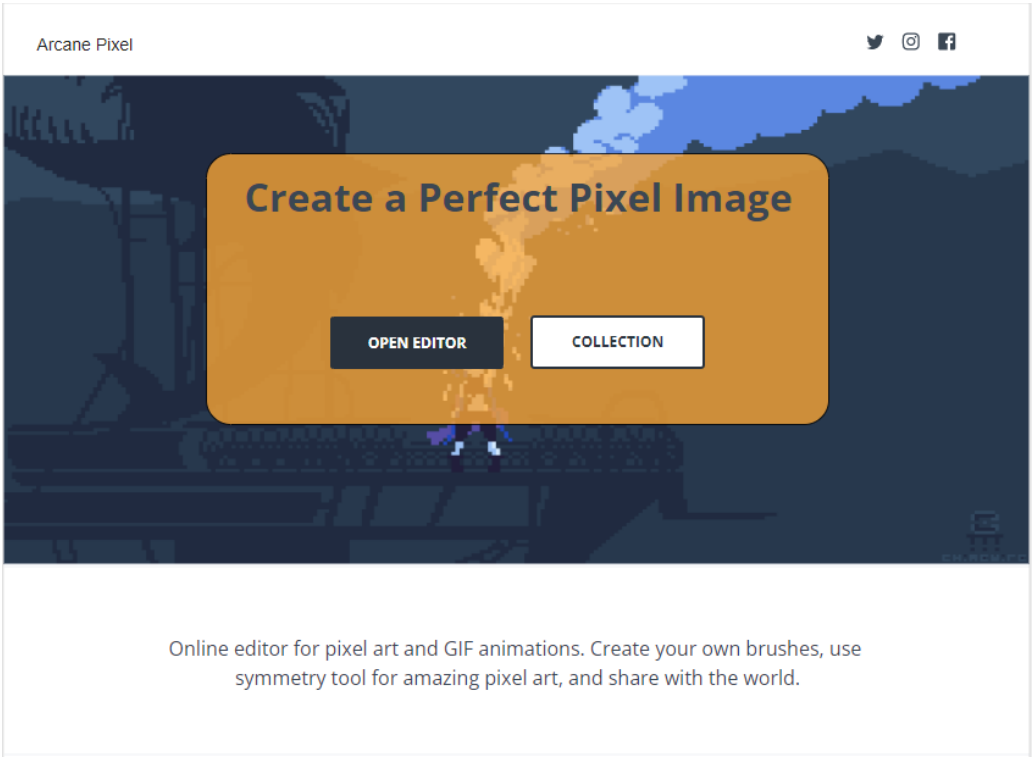

Рисунок 3.6 – Домашня сторінка додатку

На рисунку 3.7 зображено каталог наявних малюнків, які користувач може переглянути. Також на цій сторінці є фільтри для пошуку трьома категоріями, а саме: «Популярність», «Дата розміщення» та «Автор».

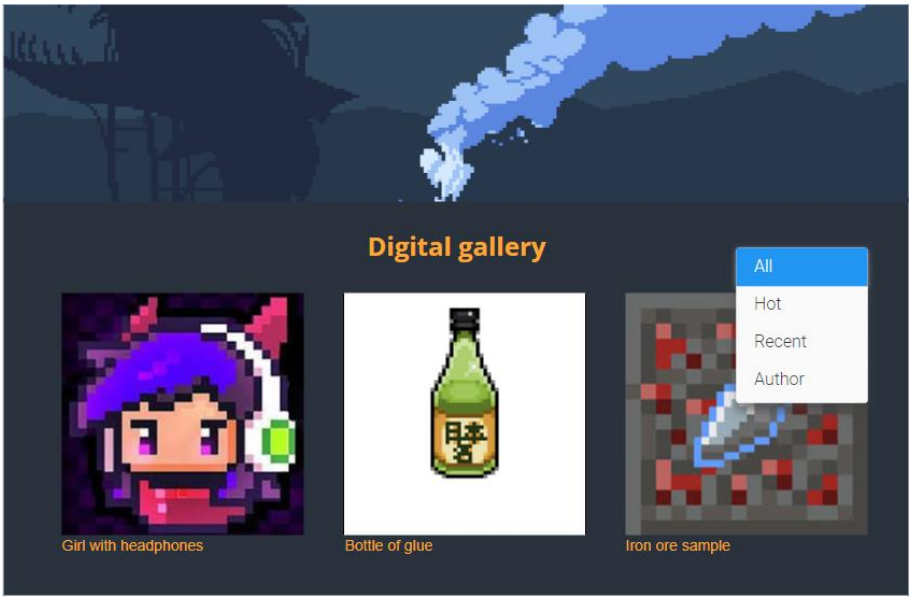

Рисунок 3.7 – Каталог наявних малюнків

На рисунку 3.8 зображено первинні налаштування редактору. Тут можна вибрати розмір полотна для зображення[10]. Якщо натиснути на кнопку “Start drawing”, вікно налаштувань приховується та відображається редактор.

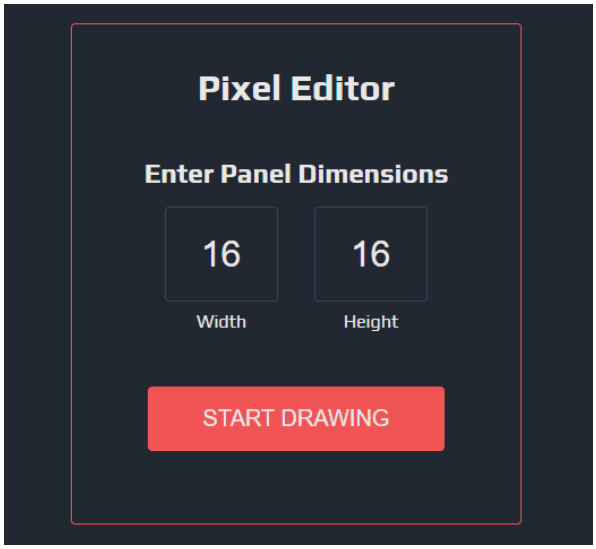

Рисунок 3.8 – Налаштування редактору

На рисунку 3.9 зображено редактор. Невелике полотно може бути тільки квадратної форми за розмірами заданими на попередньому кроці. Кнопка “Reset” відповідає за перезавантаження сторінки, таким чином прибираються пікселі що були змінені.

Для створення кольорової палітри було використано Параметр color propc у CirclePicker використовується для позначення того, який колір наразі вибрано, useChangeComplete - це подія компонента, за допомогою якої ініціюється дія. У цьому випадку після вибору іншого кольору з засобу вибору, стан вибраного кольору буде змінено.

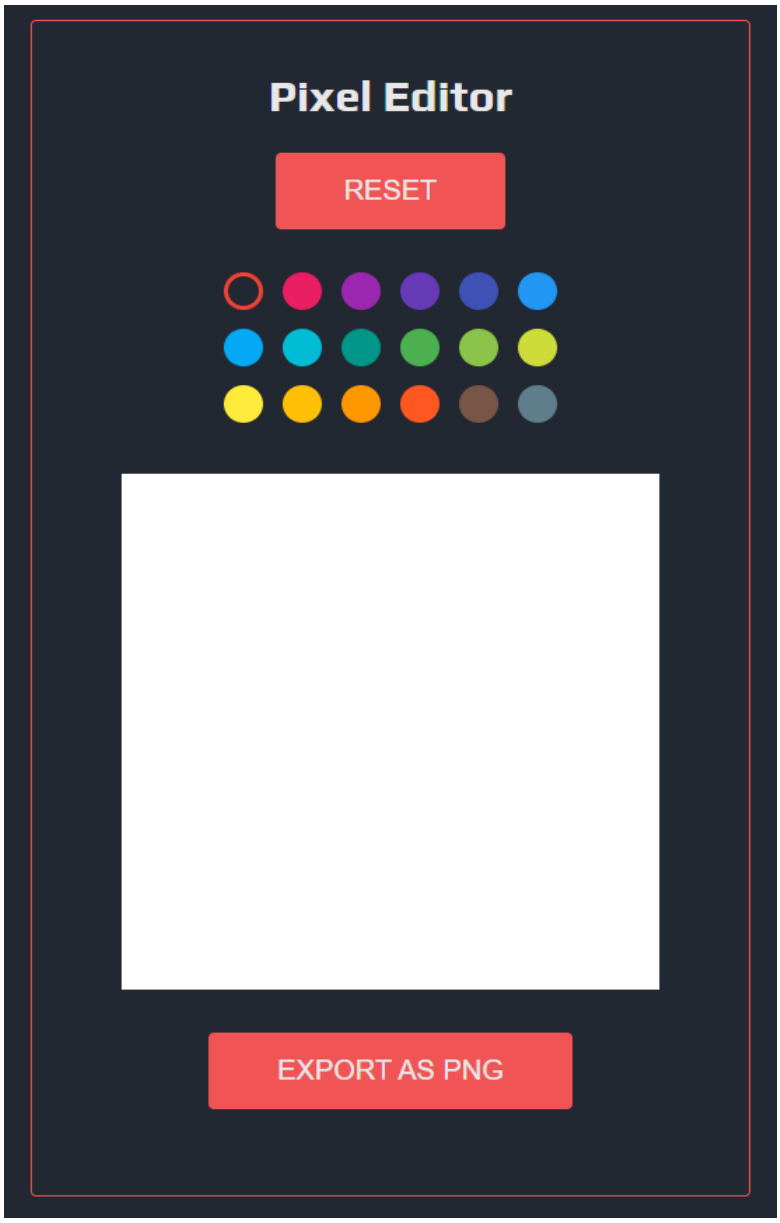

Рисунок 3.9 – Піксельний редактор

```
import React, { useRef } from "react"
import "../styles/drawingPanel.scss"
import Row from "./Row"

export default function DrawingPanel(props) {
  const { width, height, selectedColor } = props

  let rows = []

  for (let i = 0; i < height; i++) {
    rows.push(<Row key={i} width={width} selectedColor={selectedColor} />)
  }

  return (
    <div id="drawingPanel">
      <div id="pixels">{rows}</div>
    </div>
  )
}
```

Рисунок 3.10 – Піксельний редактор

На рисунку 3.10 зображено компонент поверхі для малювання. На основі введеної висоти генерується однакова кількість рядків і надсилає їх у контейнер div, але також потрібно передати ширину кожному компоненту Row, щоб отримати скільки «пікселів» на рядок потрібно згенерувати. Далі потрібно налаштувати компонент Row.

```
import React from "react"
import "../styles/row.scss"
import Pixel from "./Pixel"

export default function Row(props) {
  const { width, selectedColor } = props

  let pixels = []

  for (let i = 0; i < width; i++) {
    pixels.push(<Pixel key={i} selectedColor={selectedColor} />)
  }

  return <div className="row">{pixels}</div>
}
```

Рисунок 3.11 – Компонент Row

На рисунку 3.11 наведено компонент Row. Рядок створено за розмірами що були вказані на першому кроці. Пікселі у рядку генеруються так само, к створюються ряди на поверхі для малювання. Таким чином утворюється рядок з пікселями. В цьому компоненті створюється лише один рядок. В компоненті DrawingPanel створено масив, який викликає компонент Row необхідну кількість разів щоб заповнити поверхню для малювання. Таким чином отримується готова поверхня.

Щоб експортувати зображення використовується плагін. Компонент експорту додано до DrawPanel компоненту. Для його робити використовується useRef хук. Для цього додано невелику функцію, що експортує колекцію усіх рядків.

На рисунку 3.12 зображено форму зворотного зв’язку. Ця форма дозволяє надіслати скаргу, побажання чи пропозицію адміністратору сайту. Поля Ім’я та email, обов’язкові до заповнення, таким чином, адміністратор сайду отримає інформацію про відправника та його адресу для відповіді[11].

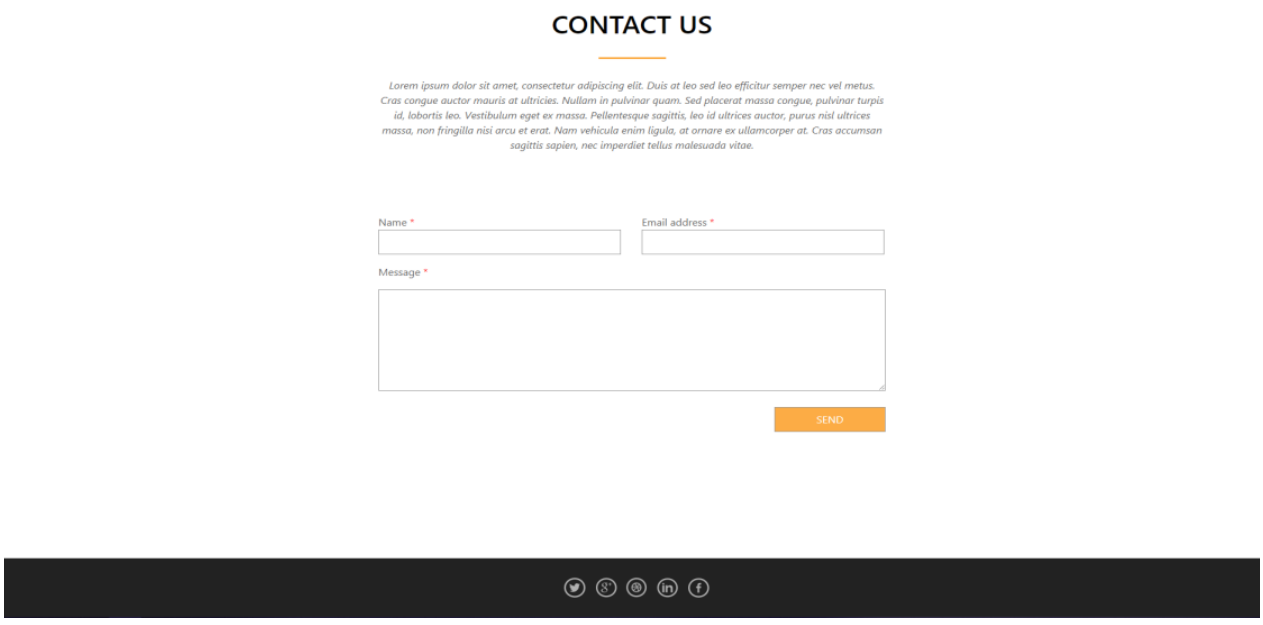

Рисунок 3.12 – Форма зворотного зв’язку

3.4 Розробка вебсайту

У типовій програмі React дані передаються зверху вниз (батьківський до дочірнього) через властивості, але таке використання може бути громіздким для певних типів властивостей (наприклад, налаштування локалі, тема інтерфейсу користувача), які потрібні багатьом компонентам програми. Контекст надає спосіб обмінюватися значеннями, подібними до цих, між компонентами без необхідності явно передавати властивість через кожен рівень дерева. Для створення авторизації на сайті за допомогою React використовують контекст.

```
export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState();
  const [accessToken, setAccessToken] = useState();
  const [refreshToken, setRefreshToken] = useState(window.localStorage.getItem(TOKEN_NAME))
  const [isAuthenticated, setIsAuthenticated] = useState(Boolean(accessToken));

  const apiInstance = axios.create({
    headers: {
      'Authorization':  `Bearer ${accessToken}`,
      'Content-Type': 'multipart/form-data'
    },
  })
```

Рисунок 3.13 – Створення контексту

На рисунку 3.13 зображено створення контексу та створення змінних для збереження токенів доступу.

Кожен об’єкт Context поставляється з компонентом Provider React, який дозволяє споживаючим компонентам підписуватися на зміни контексту. Компонент постачальника приймає властивість значення, яка передається компонентам-споживачам, які є нащадками цього постачальника. Один Провайдер може бути підключений до багатьох споживачів. Постачальники

можуть бути вкладеними, щоб замінити значення глибше в дереві.

Також наявна форма зворотного зв’язку. Через цю форму клієнт може залишити коментар, скаргу або побажання щодо наданих послуг. Також клієнт лишає свою електронну адресу, на яку буде надіслано відповідь.

```
@csrf
        <input name="name" placeholder="Ваше имя" type="text" id="name">
        <input name="email" placeholder="Ваше E-mail" type="email" id="email">
        <textarea name="message" id="name" placeholder="Ваш комментарий">
        <button type="submit" class="btn">Отправить</button>
```

Рисунок 3.14 – Форма зворотного зв’язку

На рисунку 3.15 зображено логіку автоматичної авторизації. Вона побудована за принципом, таким що при кожному перезавантаженні сторінки, додаток отримує новий токен доступу і отримує інформацію про користувача.

```
const getNewAccessToken = () => {
  apiInstance.post(REFRESH_ACCESS_TOKEN_URL, {refresh: refreshToken})
  .then(({data: {access}}) => {
      setAccessToken(access);
      setIsAuthenticated(true);
  })
}

const getUserData = () => {
  apiInstance.get(LOGGED_USER_INFO_URL)
  .then(({data}) =>setUser(data));
}

useEffect(() => {
  if (accessToken) getUserData();
  if (!accessToken && refreshToken) getNewAccessToken();
  if (!refreshToken && !refreshToken) setIsAuthenticated(false);
}, [accessToken, refreshToken, isAuthenticated])
```

Рисунок 3.15 – Логіка автоматичної авторизації

Також на сайті є колекція усіх наявних малюнків інших користувачів. На рисунку 3.16, наведено фрагмент коду, що утворює сітку товарів.

```
<div className="card">
  <div className="card-body">
    <div className="d-flex justify-content-between">
      <h5 className="card-title">{eventType} events</h5>
      {user?.user_type === 'Admin' && <Link to="/events/new" className='btn btn-sm btn-primary'>New</Link>}
    </div>
    <hr />
    <table className="table table-striped table-responsive">
      <thead>
        <tr>
          <th> Event </th>
          <th> Lead </th>
          <th> Location </th>
          {user.user_type == 'Admin' && (<><th> Budget progress </th>
            <th> Remaining budget </th></>)}
          <th className="text-center"> Actions </th>
        </tr>
      </thead>
```

Рисунок 3.16 – Сітка сторінки колекції

3.5 Створення бази даних

Open server має у собі PhpMyAdmin, за допомогою якого, було створено базу даних. Для того щоб підключитися до бази потрібно налаштувати .env файл, вказавши тип бази даних, назву, логін та пароль для входу.

Для надійності збереження даних та можливості подальшої розширення бази даних, було використано міграції.

```
),
("created_at", models.DateTimeField(auto_now_add=True)),
("updated_at", models.DateTimeField(auto_now=True)),
("tag", models.CharField(max_length=255)),
("author", models.CharField(max_length=500)),
("end_date", models.DateTimeField()),

(
    "author",
```

Рисунок 3.17 – Сітка сторінки колекції

ВИСНОВКИ

Було розроблено вебдодаток на базі фреймворку React. При розробці цього проєкту були отримані навички роботи з базами даних, контролерами. Також були поглиблені навички з проєктування систем та баз даних. Окрім цього, була проведена робота з клієнтською частиною сайту, що дало змогу покращити дизайнерські навички.

На прикладі вебдодатку «Піксельний редактор», було показано ефективність розробки з використанням технологій PHP та React.

Створений додаток може бути використано для подальшого розвитку бізнесу та його діджиталізації. Наведений приклад використання технологій може використатися в будь-якій сфері виробництва, маркетингу, продаж чи у розважальній сфері.

ПЕРЕЛІК ПОСИЛАНЬ

1. Williams B. Professional WordPress: Design and Development: 2nd Edition. Birmingham: Wrox, 2013. 456 p.

2. Stauffer M. Laravel: Up & Running: A Framework for Building Modern PHP Apps: 2nd Edition. Newton: O`relly, 2019. 585 p.

3. Дронов В. О. Швидка розробка вебсайтів на PHP. Санкт-Петербург: БХБ-Пітербург, 2021. 688 с.

4. Pitt C. Pro PHP MVC Expert's Voice in Open Source. New York: Apress, 2012. 525 p.

5. Larman C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. London: Pearson, 2004. 736 p.

6. Томсон Л. Розробка вебдодатків за допомогою PHP и MySQL. Київ: Техніка, 2017. 438 с.

7. Nixon R. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. Newton: O`relly, 2018. 832 p.

8. Russel C. SCO OpenServer: The Windows Network Solution. London: Pearson, 2004. 304 p.

9. Duckett J. HTML and CSS: Design and Build Websites. Hoboken: John Wiley, 2011. 490 p.

10. Frain B. Responsive Web Design with HTML5 and CSS: Develop future-proof responsive websites using the latest HTML5 and CSS techniques. Birmingham: Packt, 2020. 408 p.

11. McFedries P. Web Design Playground: HTML & CSS the Interactive. New York: Manning Publications, 2019. 440 p.

12. Data survey 2022. URL: https://survey.stackoverflow.co/2022/ (дата звернення: 12.11.2022).

ДОДАТОК А

Міграції бази даних

```
class Migration(migrations.Migration):
    initial = True
    dependencies = [
        migrations.swappable_dependency(settings.AUTH_USER_MODEL),
    ]
    operations = [
        migrations.CreateModel(
            name="BudgetEvent",
            fields=[
                (
                    "id",
                    models.UUIDField(
                        default=uuid.uuid4, primary_key=True, serialize=False
                    ),
                ),
                ("created_at", models.DateTimeField(auto_now_add=True)),
                ("updated_at", models.DateTimeField(auto_now=True)),
                ("name", models.CharField(max_length=255)),
```

```
            ("location", models.CharField(max_length=500)),
            ("end_date", models.DateTimeField()),
            ("budget", models.DecimalField(decimal_places=2, max_digits=11)),
            (
                "author",
                models.ForeignKey(
                    on_delete=django.db.models.deletion.CASCADE,
                    to=settings.AUTH_USER_MODEL,
                ),
            ),
        ],
        options={
            "abstract": False,
        },
    ),
    migrations.CreateModel(
        name="Expense",
        fields=[
            (
                "id",
                models.UUIDField(
```

```
            default=uuid.uuid4, primary_key=True, serialize=False
        ),
    ),
    ("created_at", models.DateTimeField(auto_now_add=True)),
    ("updated_at", models.DateTimeField(auto_now=True)),
    ("description", models.CharField(max_length=500)),
    ("spent", models.DecimalField(decimal_places=2, max_digits=11)),
    (
        "status",
        models.CharField(
            choices=[
                ("Pending", "Pending"),
                ("Approved", "Approved"),
                ("Declined", "Declined"),
            ],
            default="Pending",
            max_length=20,
        ),
    ),
    (
        "budget_event",
```

```
            models.ForeignKey(
                on_delete=django.db.models.deletion.CASCADE,
                to="dashboard.budgetevent",
            ),
        ),
    ],
    options={
        "abstract": False,
    },
),
```

ДОДАТОК Б

Компоненти

```
import React from "react";
import "../styles/App.scss";
import Editor from "./Editor";

export default function App() {
  return (
    <div className="App">
      <Editor />
    </div>
  );
}

import React, { useRef } from "react";
import "../styles/drawingPanel.scss";
import Row from "./Row";
import { exportComponentAsPNG } from "react-component-export-image";

export default function DrawingPanel(props) {
  const { width, height, selectedColor } = props;

  const panelRef = useRef();

  let rows = [];

  for (let i = 0; i < height; i++) {
```

```
    rows.push(<Row key={i} width={width} selectedColor={selectedColor} />);
  }

  return (
    <div id="drawingPanel">
      <div id="pixels" ref={panelRef}>
        {rows}
      </div>
      <button onClick={() => exportComponentAsPNG(panelRef)}
className="button">
        Export as PNG
      </button>
    </div>
  );
}

import React, { useState } from "react";
import "../styles/editor.scss";
import { CirclePicker } from "react-color";
import DrawingPanel from "./DrawingPanel";

export default function Editor() {
  const [panelWidth, setPanelWidth] = useState(16);
  const [panelHeight, setPanelHeight] = useState(16);
  const [hideOptions, setHideOptions] = useState(false);
  const [hideDrawingPanel, setHideDrawingPanel] = useState(true);
  const [buttonText, setButtonText] = useState("start drawing");
  const [selectedColor, setColor] = useState("#f44336");

  function initializeDrawingPanel() {
```

```
  setHideOptions(!hideOptions);
  setHideDrawingPanel(!hideDrawingPanel);

  buttonText === "start drawing"
    ? setButtonText("reset")
    : setButtonText("start drawing");
}

function changeColor(color) {
  setColor(color.hex);
}

return (
  <div id="editor">
    <h1>Pixel Editor</h1>
    {hideDrawingPanel && <h2>Enter Panel Dimensions</h2>}
    {hideDrawingPanel && (
      <div id="options">
        <div className="option">
          <input
            type="number"
            className="panelInput"
            defaultValue={panelWidth}
            onChange={(e) => {
              setPanelWidth(e.target.value);
            }}
          />
          <span>Width</span>
        </div>
        <div className="option">
```

```
          <input
            type="number"
            className="panelInput"
            defaultValue={panelHeight}
            onChange={(e) => {
              setPanelHeight(e.target.value);
            }}
          />
          <span>Height</span>
        </div>
      </div>
    )}

    <button onClick={initializeDrawingPanel} className="button">
      {buttonText}
    </button>

    {hideOptions && (
      <CirclePicker color={selectedColor} onChangeComplete={changeColor} />
    )}

    {hideOptions && (
      <DrawingPanel
        width={panelWidth}
        height={panelHeight}
        selectedColor={selectedColor}
      />
    )}
  </div>
);
```

```
}
import React, { useState } from "react";
import "../styles/pixel.scss";

export default function Pixel(props) {
  const { selectedColor } = props;

  const [pixelColor, setPixelColor] = useState("#fff");
  const [oldColor, setOldColor] = useState(pixelColor);
  const [canChangeColor, setCanChangeColor] = useState(true);

  function applyColor() {
    setPixelColor(selectedColor);
    setCanChangeColor(false);
  }

  function changeColorOnHover() {
    setOldColor(pixelColor);
    setPixelColor(selectedColor);
  }

  function resetColor() {
    if (canChangeColor) {
      setPixelColor(oldColor);
    }

    setCanChangeColor(true);
  }

  return (
```

```
    <div
      className="pixel"
      onClick={applyColor}
      onMouseEnter={changeColorOnHover}
      onMouseLeave={resetColor}
      style={{ backgroundColor: pixelColor }}
    ></div>
  );
}

import React from "react";
import "../styles/row.scss";
import Pixel from "./Pixel";

export default function Row(props) {
  const { width, selectedColor } = props;

  let pixels = [];

  for (let i = 0; i < width; i++) {
    pixels.push(<Pixel key={i} selectedColor={selectedColor} />);
  }

  return <div className="row">{pixels}</div>;
}
```