ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА
(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій
(повне найменування факультету)

Кафедра інформаційних технологій
(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Розробка веб-застосунку для архітектурно-будівельного конструкторського бюро Project UA»
(тема кваліфікаційної роботи українською мовою)

«Development of a Web Application for the Architectural and Construction Design Bureau Project UA»
(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання
спеціальності 122 Комп'ютерні науки
(код, назва спеціальності)
Освітня програма Комп'ютерні науки
(назва)
Твердовський Даніель Леонідович
(прізвище, ім'я, по-батькові здобувача)

Керівник асистент Гадяцький І.А.
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент к.т.н., доцент Перелигін Б.В.
(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .
№ 1 від 09 червня 2024 р.

Завідувачка кафедри
(підпис) КАЗАКОВА Надія .
(прізвище,ім'я)

Захищено на засіданні ЕК № 13 .
протокол № 26 від 21 червня 2024 р.

Оцінка відмінно / A / 90 .
(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК
(підпис) КОПИЧЕНКО Іван .
(прізвище,ім'я)

Одеса 2024

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

UML (Unified Modeling Language) – уніфікована мова моделювання, що використовується для візуалізації, специфікації, конструювання та документування компонентів програмних систем. UML забезпечує стандартний спосіб відображення системи, включаючи її структуру та поведінку.

БД (База даних) – це організований набір даних, які зберігаються та керуються таким чином, щоб можна було легко доступатися, управляти, і оновлювати ці дані. Бази даних використовуються для зберігання великої кількості інформації в структурованому форматі, що дозволяє здійснювати ефективний пошук, сортування, фільтрацію та інші операції з даними.

PHP (Personal Home Page) – скриптова мова програмування, що використовуються у сфері веб-розробок.

HTML (HyperText Markup Language) – це стандартна мова розмітки для створення веб-сторінок та веб-додатків.

CSS (Cascading Style Sheets) – це мова стилів, що використовується для опису зовнішнього вигляду та форматування HTML-документів.

PhpMyAdmin – це веб-застосунок з відкритим кодом на мові PHP із графічним веб-інтерфейсом для адміністрування СКБД MySQL.

СКБД (система керування базами даних) – це програмне забезпечення, яке дозволяє створювати, організовувати, змінювати та керувати БД.

ВСТУП

На сьогоднішній день в сучасному світі існує значний попит на оригінальний творчий авторський стиль та автентичний дизайн. Процес розробки дизайну вже неможливо уявити без використання інформаційних технологій. Кожна компанія прагне мати свій унікальний стиль, який би впізнавали клієнти. Кожен виробник бажає привернути максимальну увагу до свого товару та продати його покупцю, упакувавши в стильну та інформативну упаковку.

Цими завданнями займається графічний дизайнер компанії чи конструктор. Він створює логотип та фірмовий знак компанії, рекомендує унікальні шрифти та фірмові кольори, розробляє впізнавані графічні елементи та конструкторські креслення. Після цього він формує набір елементів фірмового стилю, що включає презентаційні комплекти та різноманітну поліграфічну продукцію. Крім того, дизайнер обов'язково створює посібник з використання фірмового стилю для всіх, хто працюватиме над рекламою компанії, її товарів або послуг.

Основне завдання дизайнера чи конструктора не лише створювати логотипи та флаєри для замовника, але й розробляти спеціальну документацію та проектувати конструкторські проєкти, індивідуальні рисунки, креслення та фотографії.

Метою виконання цієї дипломної роботи є розробка вебзастосунку для власника архітектурно-будівельного конструкторського бюро Project UA.

Диплома робота містить в собі 58 сторінок, 23 зображень, 23 таблиць, 11 джерел посилання.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз вебзастосунку у стилі блог

Веб-сторінка – це документ, який можна переглядати в Інтернеті за допомогою веб-браузера. Веб-сторінка може містити текст, зображення, відео, аудіо та інші мультимедійні елементи, а також інтерактивні компоненти, такі як форми, посилання та скрипти. Використовується для посилення ефективності реклами, збільшення аудиторії. Цільова сторінка зазвичай містить інформацію про товар або послугу. Перехід на цільові сторінки часто здійснюється з соціальних медіа, email-розсилок і рекламних кампаній в пошукових системах.

Головним завданням таких сторінок є конвертація відвідувача в покупця або клієнта компанії, спонукання до цільового дії. Аналіз дій користувачів на цільовій сторінці дозволяє маркетологам визначити успішність реклами.

Будь-яка організація, який веде активну робочу діяльність, має потребу в автоматизації та спрощення протікання інформаційних процесів.

Положення в сфері реклами в наш час високих технологій змінюється з кожним днем. На зміну телереклами і реклами в газетах прийшов новий вид реклами – через Інтернет [1].

За допомогою блогу власник вирішує такі завдання, як отримання нових клієнтів, обробка замовлень, залучення аудиторії, надання клієнту готових робіт для прегеляду, надання відгуків для користувача. Вебзастосунок конструкторського бюро – це обличчя, можливість справити перше враження на потенційного клієнта, донести інформацію про діяльність компанії, і послуги.

На сьогоднішній день немає практично жодної успішної людини у сфері дизайну, яка не мала свого представництва в глобальній мережі. Пояснюється це дуже просто: електронний ресурс являє собою своєрідне обличчя, до якого клієнт може звернутися в будь-який момент. У вебзастосунку можна

розмістити будь-яку необхідну інформацію, матеріали, розцінки цін на певні послуги, з його допомогою легко налагодити зворотний зв'язок. Всі ці переваги беззаперечні і, безумовно, говорять на користь створення власного вебзастосунку.

Інтернет-ресурс для компанії є безпрецедентною можливістю надати кожному бажаючому найбільш повну адресну і оперативну інформацію про себе і свої послуги. Замість того, щоб постійно відповідати на одні і ті ж питання клієнтів, роз'яснювати цінову і маркетингову політику, пересилати факсом прайси, стандартні договори, і, нарешті, просто роз'яснювати, як проїхати в офіс, можна перенаправити клієнта на сайт компанії. Є можливість миттєво змінювати інформацію про компанію, оновлювати прайс-листи, публікувати новини, проводити он-лайн зустрічі та спілкуватися з клієнтами.

Молодіжні організації відіграють ключову роль у розвитку ініціатив та проєктів, спрямованих на вирішення актуальних проблем молоді. Вони стають інкубаторами ідей, де молодь може реалізовувати свої творчі та інноваційні концепції [2].

1.2 Завдання і функції застосунку

Рекламні сторінки мають просту структуру, що запобігає заплутаності користувачів. Їхня вартість значно нижча порівняно з багато сторінковими сайтами. Легкість сайту забезпечує швидке завантаження, що утримує потенційних покупців від переходу до конкурентів. Основна увага рекламних сторінок зосереджена на конкретній пропозиції компанії, поступово і впевнено спрямовуючи користувача до оформлення замовлення або виконання іншої дії. Взаємини підприємства з іншими підприємствами, організаціями, установами, органами державного та муніципального управління, а також з громадянами регулюються законодавством. Діяльність власника спрямована на надання різних послуг:

– створення фірмового стилю компанії і його головного елемента –

логотипу;

– розробка корпоративного шрифту та іншої атрибутики;

– підготовка дизайну рекламної поліграфічної продукції (брошури, листівки, буклети, візитки, бланки, конверти, календарі та інше);

– виготовлення сувенірної продукції;

– розробка упаковки (обкладинки, пакети, етикетки, коробки та інше);

– графічний дизайн банерів, мультимедіа та Web продуктів;

– створення дизайну для соціальної та інформаційної поліграфічної продукції (ілюстрації книг, книжкові макети, газети, журнали);

– візуальний стиль відеоматеріалів (телевізійні передачі, музичні та рекламні кліпи) та інше.

Призначення працівника організації може бути сформульована таким чином: розробка і створення об'єктів графіки і графічного дизайну.

Кожна компанія прагне мати власний стиль, який буде легко впізнаваним для клієнтів і покупців. Виробники хочуть привернути максимальну увагу до свого продукту і продати його, створивши стильну і інформативну упаковку. Цим займається графічний дизайнер. Він розробляє логотип і символ компанії, рекомендує певні шрифти та фірмові кольори, створює впізнавані графічні елементи, а також формує весь набір фірмового стилю, включаючи презентаційні набори та різноманітну поліграфічну продукцію. За створення конструкторських креслень і розрахунків відповідає конструктор. Вони також створюють керівництво з використання створеного стилю для всіх, хто працюватиме над рекламою компанії, її продуктів або послуг.

1.3 Вибір типу сайту для майбутньої розробки

На сьогодні в Інтернеті величезна кількість типів і видів сайтів, які

досить різноманітні. Щоб орієнтуватися у величезній різноманітності сайтів їх можна розділити на такі типи.

Залежно від завдань, їх розділяють на кілька типів:

- інтернет-портал;
- інформаційні ресурси;
- інтернет-представництва власників бізнесу;
- веб-сервіс;
- комбіновані веб-сервіси (соціальні мережі).

Інтернет-портал – це складна структура, яка складається з різних компонентів (порталів), що мають свою функціональну самодостатність. Вони можуть бути сайтами окремих організацій або підрозділів в корпоративній структурі.

Інформаційні ресурси можуть включати:

Тематичний сайт, який надає конкретну вузькотематичну інформацію. А також тематичний портал, який є великим веб-ресурсом і містить повну інформацію з певної тематики, а також інструменти для взаємодії з користувачами, такі як форуми та чати [3].

Інтернет-представництва бізнесу включають:

- сайт-візитка, який містить загальні дані про власника сайту, його вид діяльності, історію та контактні дані;
- представницький сайт, який розширює функціонал сайту-візитки, додавши детальний опис послуг, портфоліо та форму зворотного зв'язку;
- корпоративний сайт, що містить повну інформацію про компанію-власника, його послуги та події, а також функціональні інструменти для роботи з контентом;
- каталог продукції, де можна знайти докладний опис товарів або послуг;
- інтернет-магазин, що має каталог продукції і дозволяє клієнтам замовляти товари за допомогою різних систем розрахунків;

– промо-сайт, який присвячений конкретній торговій марці або продукту, та містить вичерпну інформацію про бренд та рекламні акції;

– сайт-квест, на якому проводяться змагання з розгадування логічних загадок.

Лендінг (або посадкова сторінка) – це односторінковий веб-сайт, призначений для збору контактної інформації від відвідувачів або продажу товару, і який дозволяє ефективно привертати увагу користувачів.Власника конструкторського бюро Project UA необхідно надати забезпечення інформаційної підтримки в Інтернеті, а саме подання інформації про діяльність, представлення готових робіт для оцінення клієнтом досвідченості дизайнера та креативності робіт, креслення, конструкторські роботи, наданих послуг, з метою залучення нових клієнтів. У зв'язку з вище викладеним, було прийнято вибрати представницький тип сайту в стилі Блог.

1.4 Аналіз готових рішень

Перед розробкою власного вебзастосунку були досліджені вебзастосункии інших компаній та сайти-портфоліо.

«Varlamova Photo» вебзастосунок візуального графічного дизайну зображено на рисунку 1.1.

Рисунок 1.1. – Блог дизайнера

Дизайн сайту продуманий, візуально зрозумілий, зручний в використанні, є система навігації. Основними перевагами є наявність повної інформації про людину, є прайс-лист, є форма зворотнього зв'язку, є можливість залишати відгуки та коментарі. Сайт вдало побудований та продуманий.

«Later» вебзастосунок маркетингової платформи у соціальній мережі Instagram дизайнерських і конструкторських портфоліо зображено на рисунку 1.2.

Рисунок 1.2 – Вебзастосунок маркетингової платформи

Примітивний дизайн сайту, є можливість ознайомитися з переліком послуг які надаються, можна переглянути контактні дані, можна скористатися формою зворотнього зв’язку, є сторінка відгуків, перебір рекламних креслень і рисунків, мета яких перенаправити увагу клієнта на іншу сторінку.

1.5 Постановка задачі

На основі аналізу предметної області, необхідно розробити вебзастосунок для власника архітектурно-будівельного конструкторського бюро Project UA.

Функції вебзастосунку для користувача:

- перегляд контенту;
- можливість перегляду портфоліо конструктора-дизайнера (майстрів);
- можливість перегляду конструкторсько-дизайнерських робіт (документи);
- авторизація/реєстрація;
- можливість замовити документацію;
- можливісь користувачам залишати відгуки;
- можливість отримати зворотній зв’язок.

Для адміністратора вебзастосунку:

- авторизація;
- наповнення контенту (конструкторсько-дизайнерські роботи)
- керування даними користувачів;
- управління документацію;
- вести портфоліо професійних робіт;
- управління зворотнім зв’язком;
- управління відгуками.

Вхідні данні:

- інформація користувача;
- інформація про майстрів;
- інформація про документи (професійні роботи);
- інформація портфоліо;
- інформація відгуків;
- інформація для зворотнього зв’язку;
- інформація замовлення.

Вихідні дані:

- дані користувача;
- дані майстрів;
- список документів (професійні роботи);
- дані портфоліо;
- дані відгуків;
- дані зворотнього зв’язку;
- дані замовленнь.

Вхід у вебзастосунок.

Щоб розпочати роботу із вебзастосунком, відкрийте будь-який браузер, встановлений на вашому комп'ютері чи мобільному пристрої. У рядку адреси введіть URL-адресу вебзастосунку або знайдіть його через пошукову систему. Натисніть "Enter" для переходу на вебзастосунок.

Головна сторінка.

Після введення адреси ви опинитеся на головній сторінці вебзастосунку. На цій сторінці ви знайдете загальну інформацію про дизайнера, включаючи опис його професійної діяльності, портфоліо робіт, контактні дані, а також інші корисні відомості. Головна сторінка служить вхідною точкою для навігації по всьому вебзастосунку, забезпечуючи доступ до різних розділів та ресурсів [5].

Результат розробки дипломного проекту.

Одним з ключових результатів розробки дипломного проекту є створення вебзастосунку Project UA. Цей вебзастосунок представляє собою завершений продукт, готовий до використання. У вебзастосунку ви можете ознайомитися з детальною інформацією про цей вебзастосунок, дізнатися про його функціональні можливості, цілі і завдання, які він вирішує, а також переглянути приклади його використання. Крім того, вебзастосунок надає можливість завантаження та тестування вебзастосунку.

Додаткові можливості вебзастосунку.

У вебзастосунці реалізовано зручне меню "Процес роботи", яке дозволяє кожному користувачеві переглядати етапи виконаних робіт майстра. Це дає змогу краще зрозуміти методику і стиль роботи дизайнера. Користувачі також можуть ознайомитися з професійними документами та матеріалами конструкторського бюро Project UA, що надає додаткову інформацію про їхню діяльність і досягнення.

Можливість перегляду конкретного проєкту.

У вебзастосунку реалізована функція перегляду окремих проєктів з портфоліо майстрів. Користувачі можуть детально ознайомитися з кожним проєктом, переглядаючи фотографії, описи та технічні характеристики робіт. Це дозволяє отримати повне уявлення про професійний рівень майстрів і якість виконаних ними робіт. Крім того, кожен проєкт супроводжується коментарями майстрів, де вони діляться своїм підходом та пояснюють використані техніки.

Можливість зворотнього зв’язку.

У вебзастосунку також реалізована можливість зворотнього зв’язку. Користувачі можуть легко зв’язатися з майстрами через спеціальну форму зворотнього зв’язку. Це дозволяє відвідувачам вебзастосуноку задавати питання, отримувати консультації або залишати свої пропозиції. Швидкий і зручний спосіб комунікації сприяє налагодженню тісних контактів між майстрами і клієнтами, що підвищує рівень довіри та задоволеності

послугами.

Знайомство з майстрами.

Вебзастосунок надає можливість ближче познайомитися з майстрами конструкторського бюро Project UA. Кожен майстер має власну сторінку, де представлена інформація про його професійний досвід, досягнення, сертифікати та портфоліо робіт. Це дозволяє потенційним клієнтам краще зрозуміти кваліфікацію і стиль кожного майстра, що є важливим при виборі спеціаліста для реалізації власних проектів.

Переваги онлайн-замовлення.

Замовлення послуг через онлайн-форму значно спрощує процес взаємодії з бюро Project UA. Це дозволяє клієнтам зручно та швидко замовляти необхідні послуги в будь-який зручний для них час, без необхідності особистого візиту до офісу. Крім того, наявність електронної форми замовлення забезпечує точність переданих даних та мінімізує можливість виникнення непорозумінь.

Робота з текстовою та графічною інформацією У вебзастосунку.

Для роботи з текстовою та графічною інформацією У вебзастосунку використовується візуальний редактор HTML-коду Visual Studio Code. Цей інструмент забезпечує зручне форматування, редагування тексту, вставку графічних зображень та інші можливості для оптимального створення та управління контентом. Visual Studio Code підтримує також копіювання тексту з текстового процесора MS Word, що значно полегшує процес перенесення текстових даних.

Копіювання тексту з MS Word до Visual Studio Code.

Процес копіювання тексту з MS Word до Visual Studio Code здійснюється за допомогою буфера обміну. Нижче наведені покрокові інструкції для виконання цього завдання:

– відкрити документ у текстовому процесорі Microsoft Word;

– запустіть MS Word і відкрийте документ, з якого потрібно скопіювати текст;

– виділити фрагмент документу (або весь документ), який потрібно скопіювати на сайт;

– клацніть лівою клавішею миші на початку фрагменту та, утримуючи клавішу, перемістіть курсор до кінця фрагменту, який потрібно виділити;

– скопіювати виділений фрагмент до буфера обміну;

– натисніть правою клавішею миші на виділеному фрагменті та виберіть у контекстному меню пункт «Копировать». Текст тепер збережено у буфері обміну;

– відкрити систему адміністрування у вікні браузера;

– запустіть веб-браузер, перейдіть до системи адміністрування вашого вебзастосуноку та виберіть сторінку, на яку потрібно вставити текст;

– вставити скопійований текст з буфера обміну у робочу область FCKeditor;

– у робочій області FCKeditor вставте скопійований текст за допомогою комбінації клавіш CTRL-V.

Вставка тексту в залежності від його формату:

– для тексту без таблиць: Натисніть кнопку «Вставити тільки текст» на панелі інструментів Visual Studio Code;

–для тексту з таблицями: Натисніть кнопку «Вставити з Word». Після натискання відповідної кнопки з’явиться діалогове вікно. Вставте інформацію з буфера обміну за допомогою комбінації клавіш CTRL-V. Зніміть позначки біля опцій «Ігнорувати налаштування шрифтів» і «Видалити налаштування стилів», потім натисніть кнопку «ОК»;

Редагування та форматування тексту у Visual Studio Code.

Використовуйте всі можливості Visual Studio Code для редагування та форматування тексту. Щоб зробити роботу зручнішою, розгорніть панель редактора на весь екран, натиснувши один раз лівою клавішею миші на кнопку

«Розгорнути вікно редактора» [6].

Автоматизоване керування вебзастосунком.

Для автоматизованого керування вебзастосунком конструкторського бюро розроблена Адмін панель з різноманітними модулями. Основні модулі включають:

– MAIN: Центральний модуль для управління основними функціями вебзастосуноку;

– BLOG: Модуль для створення та управління блогами та новинами;

– GALLERY: Модуль для управління галереями зображень та іншими медіафайлами;

– SYSTEM: Модуль для налаштувань системи та управління адміністративними правами.

2 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ВИБІР ЗАСОБІВ РОЗРОБКИ

2.1 Ескізний проект

Ескізний проект (ескізний проєкт) – це початковий етап у процесі розробки будь-якого інженерного або архітектурного об'єкта. Він включає в себе попередні креслення, схеми, ідеї та концепції, які допомагають сформувати загальне уявлення про кінцевий вигляд і функціональність проекту. Ескізний проект зазвичай не є детальним і точним, але він надає достатньо інформації для подальшої розробки.

2.1.1 Вибір засобів моделювання

Для побудови складної інформаційної системи необхідним етапом є прое-ктування. Найбільш відомими візуальними моделями, що використовуються для проектування комп'ютерних систем та їхніх програмних забезпечень, є діаграми мови UML (Unified Modeling Language).

Уніфікована мова моделювання (UML) є стандартним інструментом для створення діаграм та моделей програмного забезпечення. Вона дозволяє візуалізувати, специфікувати, конструювати і документувати артефакти програмних систем. UML може бути використаний для моделювання різноманітних систем, починаючи від корпоративних інформаційних систем і закінчуючи розподіленими веб-додатками та вбудованими системами реального часу [7].

2.1.2 Розробка контекстної моделі системи

Проектування програмного забезпечення починається із розробки контекстної діаграми.

Контекстна діаграма це початкова діаграма моделі, яка характеризує функції системи взагалі і зв’язки системи з навколишнім середовищем. Для подальшого проектування необхідно виділити основних діючих осіб:

– користувач;
– адміністратор.

Контекстну діаграму можна переглянути на рисунку 2.1.

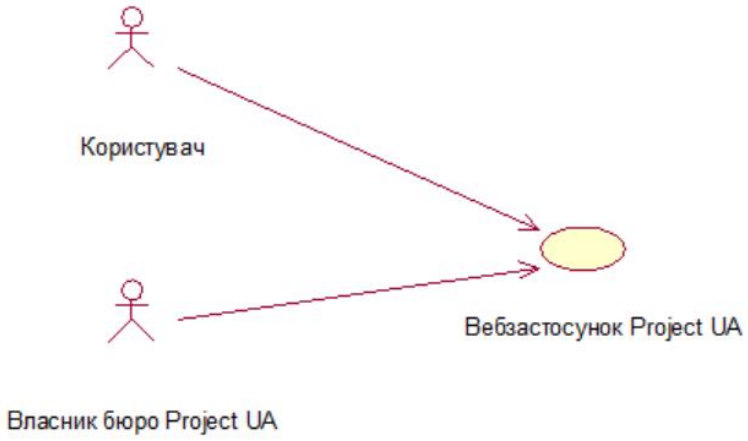


Рисунок 2.1 – Контекстна діаграма вебзастосунку Project UA

2.1.3 Діаграма варіантів використання

Для розуміння роботи системи використовується опис функціональності через варіанти використання (Use Case або прецеденти). Вони представляють собою опис послідовностей дій, які система може виконувати у відповідь на зовнішні взаємодії з користувачами або іншими програмними системами. Ці варіанти використання чітко відображають функціональні можливості системи.

Діаграма варіантів використання вебзастосунку Project UA для користувача і адміністратора представлена на рисунку 2.2.

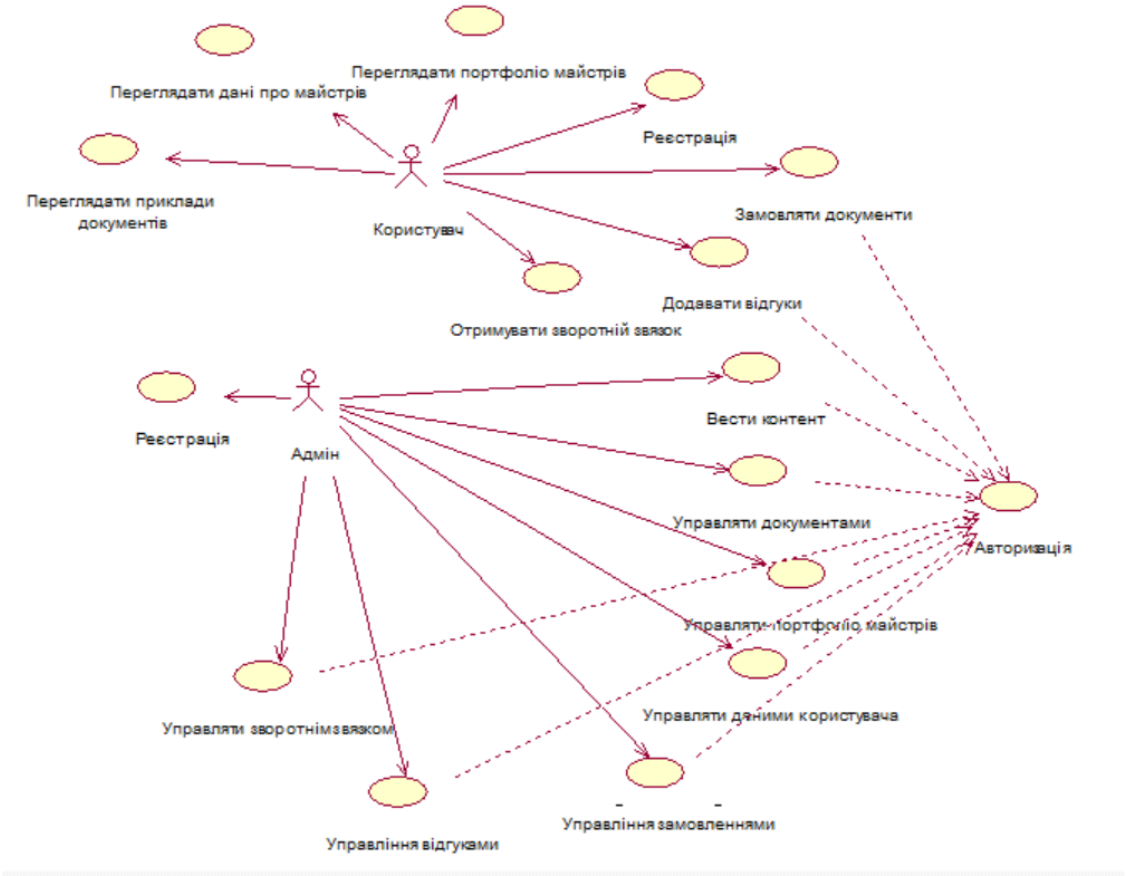


Рисунок 2.2 – Діаграма варіантів використання вебзастосунку Project UA

Специфікації варіантів використання для суб'єкта «Користувач» наведено в таблицях 2.1-2.14.

Специфікація варіанту використання «Перегляд інформації портфоліо майстрів» зображена в таблиці 2.1. Вона має 5 характеристик і їх опис.

Таблиця 2.1 - Варіант використання «Перегляд інформації портфоліо майстрів»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу переглянути інформацію про портфоліо майстра
Передумови	Відсутні
Основний потік подій	Користувач ініціює вибір посилання на сторінку портфоліо майстрів. Система відображає інформації задану сторінку.

Продовження таблиці 2.1

Характеристика	Опис
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Перегляд даних про майстра» зображена в таблиці 2.2. Вона має 5 характеристик і їх опис.

Таблиця 2.2 - Варіант використання «Перегляд даних про майстра»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу переглянути особисті дані про майстра
Передумови	Відсутні
Основний потік подій	Користувач ініціює вибір сторінки Даних про майстра. Система відображає каталог задану сторінку.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Перегляд прикладів документів» зображена в таблиці 2.3. Вона має 5 характеристик і їх опис.

Таблиця 2.3 - Варіант використання «Перегляд прикладів документів»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу переглянути документи
Передумови	Відсутні
Основний потік подій	Користувач ініціює вибір сторінки Прикладів документа. Система відображає задану сторінку. Користувач обирає певний зразок і клікає на завантаження. Система завантажує користувачу заданий приклад.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Замовлення документу» зображена в таблиці 2.4. Вона має 5 характеристик і їх опис.

Таблиця 2.4 - Варіант використання «Замовлення документу»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу замовити конструкторські документи
Передумови	Відсутні
Основний потік подій	Користувач ініціює замовлення документу. Система завантажує форму для внесення даних замовлення. Користувач вносе до форми необхідні дані. Система зберігає внесені дані.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Розміщення відгуків» зображена в таблиці 2.5. Вона має 5 характеристик і їх опис.

Таблиця 2.5 - Варіант використання «Розміщення відгуків»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу залишати відгуки на Project UA
Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Користувач ініціює перехід на сторінку відгуків. Система завантажує сторінку відгуків. Користувач додає відгуки. Система зберігає внесені відгуки.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління документами» для суб’єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.6. Вона має 5 характеристик і їх опис.

Таблиця 2.6 - Варіант використання «Управління документами» для суб’єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти документами (додавати послуги, змінювати або видаляти)
Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Адміністратор переходить до модулю Адмін панелі і ініціює завантаження галереї робіт. Система надає можливість додавати, записувати, видаляти та редагувати. Адміністратор обирає необхідну дію та виконує зміни. Система зберігає ці зміни.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління контентом» для суб’єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.7. Вона має 5 характеристик і їх опис.

Таблиця 2.7 - Варіант використання «Управління контентом» для суб’єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти розділом контенту (оновлювати, додавати, змінювати або видаляти)
Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Адміністратор переходить до модулю контенту та ініціює дію. Система надає можливість додавати запис, видаляти та редагувати.

	Адміністратор обирає необхідну можливість та виконує зміни. Система зберігає ці зміни.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління портфоліо майстрів» для суб'єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.8. Вона має 5 характеристик і їх опис.

Таблиця 2.8 - Варіант використання «Управління портфоліо майстрів» для суб'єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти портфоліо робіт майстрів (додавати зображення та опис, змінювати або видаляти)
Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Адміністратор ініціює перехід до модулю портфоліо. Система надає можливість додавати записи та зображення, видаляти та редагувати. Адміністратор обирає необхідну можливість та виконує зміни. Система зберігає ці зміни.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління даними користувача» для суб’єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.9. Вона має 5 характеристик і їх опис.

Таблиця 2.9 - Варіант використання «Управління даними користувача» для суб’єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти даними користувача (додавати записи та зображення, змінювати або видаляти)
Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Адміністратор ініціює перехід до модулю даних користувача. Система надає можливість додавати записи, видаляти та редагувати. Адміністратор обирає необхідну дію та виконує зміни. Система зберігає ці зміни.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління зворотнім зв’язком» для суб’єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.10. Вона має 5 характеристик і їх опис.

Таблиця 2.10 - Варіант використання «Управління зворотнім зв’язком» для суб’єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти зворотнім зв’язком власника з користувачем (додавати записи, змінювати або видаляти)
Передумови	Не визначені
Основний потік подій	Адміністратор ініціює перехід до модулю зворотнього звя’язку. Система надає можливість написати електронний лист. Адміністратор обирає необхідну дію та виконує зміни.

Продовження таблиці 2.10

Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління замовленнями» для суб'єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.11. Вона має 5 характеристик і їх опис.

Таблиця 2.11 - Варіант використання «Управління замовленнями» для суб'єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти замовленнями документів
Передумови	Відсутні
Основний потік подій	Адміністратор ініціює перехід до модулю замовлення документів. Система відображає список замовлень. Адміністратор обирає необхідне замовлення для детального перегляду та за необхідністю надсилає повідомлення користувачу замовлення для уточнення деталей замовлення. Система зберігає дані.
Альтернативний потік подій	Відсутній
Постумови	Не визначені

Специфікація варіанту використання «Управління відгуками» для суб'єкта «Адміністратор вебзастосуноку» зображена в таблиці 2.12. Вона має 5 характеристик і їх опис.

Таблиця 2.12 - Варіант використання «Управління відгуками» для суб'єкта «Адміністратор вебзастосуноку»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє адміністратору управляти відгуками користувачів

Передумови	Виконання варіанта використання «Авторизація»
Основний потік подій	Адміністратор ініціює перехід до модулю відгуків. Система відображає відгуки користувачів. Адміністратор здійснює перехід по відгуках, видаляє, вносить відповіді. Система зберігає зміни.

Специфікація варіанту використання «Авторизація» зображена в таблиці 2.13. Вона має 5 характеристик і їх опис.

Таблиця 2.13 - Варіант використання «Авторизація»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу авторизуватись у системі .
Передумови	Користувач повинен мати спеціальний ідентифікатори (адреса електронної пошти та пароль, які він вводив на етапі реєстрації) для доступу в систему.
Основний потік подій	Користувач ініціює авторизацію. Система надає форму авторизації. Користувач вводить адресу електронної пошти та пароль, які вводив при реєстрації аккаунту в задану форму. Система перевіряє вірність введення даних та допускає до акаунту.
Альтернативний потік подій	Невірний пароль: Користувач ініціює авторизацію. Система надає форму авторизації. Користувач вводить адресу електронної пошти та пароль, які вводив при реєстрації аккаунту, в задану форму. Система перевіряє вірність введення даних: якщо ведений пароль – невірний, то система повідомляє про це, та пропонує повторити введення або завершити варіант використання.
Постумови	Не визначені.

Специфікація варіанту використання «Реєстрація» зображена в таблиці 2.14. Вона має 5 характеристик і їх опис.

Таблиця 2.14 - Варіант використання «Реєстрація»

Характеристика	Опис
Короткий опис	Даний варіант використання дозволяє користувачу зареєструватися у системі.
Передумови	Користувач повинен ввести адресу електронної пошти, яка ще не була зареєстрована.
Основний потік подій	Користувач ініціює реєстрацію. Система надає форму. Користувач заповнює форму реєстрації. Система перевіряє валідність введених даних і створює новий акаунт.
Альтернативний потік подій	Невірна адреса електронної пошти: Користувач ініціює реєстрацію. Система надає форму реєстрації. Користувач заповнює форму реєстрації. Система перевіряє валідність введених даних. Введена адреса вже зареєстрована. Система повідомляє про це, та пропонує можливість повторити введення або завершити варіант використання.
Постумови	Не визначені.

2.1.4 Концептуальна модель

Концептуальне проектування – це побудова семантичної моделі предметної області. Концептуальна модель представлена на рисунку 2.3.

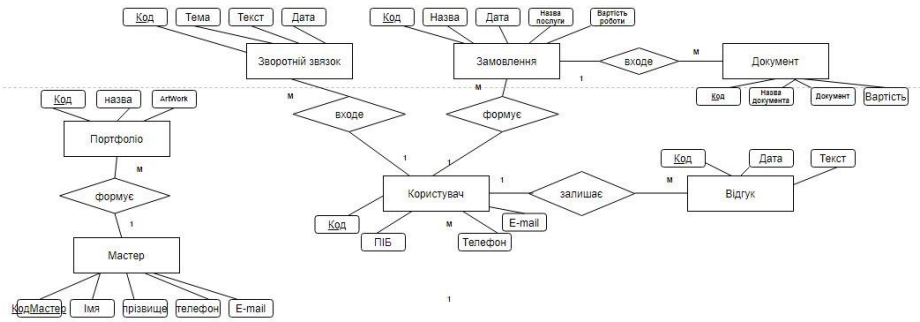


Рисунок 2.3 – Концептуальна модель даних вебзастосунку Project UA

2.1.5 Діаграма станів

Діаграми станів відображають різні стани об'єкта, а також події або повідомлення, що спричиняють зміну стану об'єкта. Вони також показують дії, які відбуваються внаслідок такої зміни стану. Діаграма станів вебзастосуноку графічного дизайнера для суб’єкта «Користувач і адміністартор» представлена на рисунку 2.4.

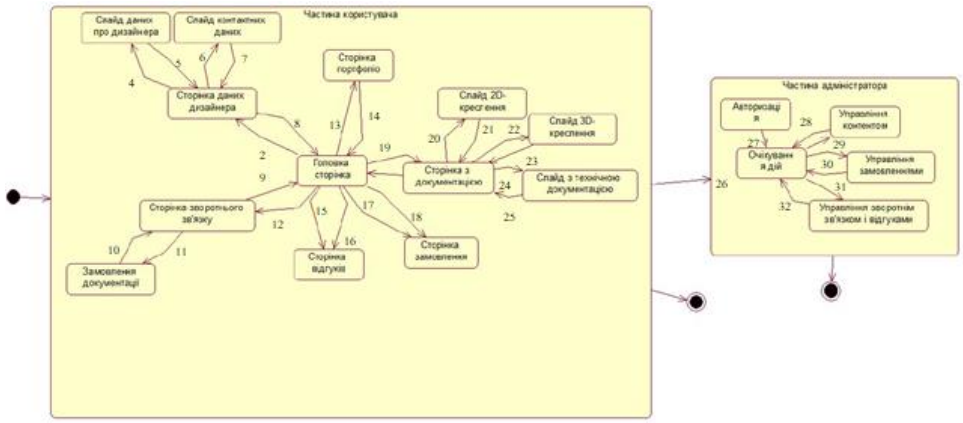


Рисунок 2.4 – Діаграма станів вебзастосунку Project UA для користувача та для дизайнера

Специфікації діаграми переходів станів наведені в таблицях 2.15-2.16.

Специфікація діаграми переходів станів для користувача представлено на таблиці 2.15. Показано номер та дія, яка виконується.

Таблиця 2.15 - Специфікація діаграми переходів станів для користувача

Номер	Умова/Дія
9,15,8,14,20	Ініціювання переходу до головної сторінки / Повернення до головної сторінки
2	Ініціювання переходу до сторінки даних дизайнера
4	Ініціювання переходу до слайду даних про дизайнера
6	Ініціювання переходу до слайду контактних даних
12	Ініціювання переходу до сторінки зворотнього зв’язку
11	Ініціювання переходу до сторінки замовлення документації

Продовження таблиці 2.15

Номер	Умова/Дія
15	Ініціювання переходу до сторінки відгуків
13	Ініціювання переходу до сторінки портфоліо
17	Ініціювання переходу до сторінки замовлення
19	Ініціювання переходу до сторінки з документацією
20	Ініціювання переходу до слайду 2D креслення
22	Ініціювання переходу до слайду 3D креслення
23	Ініціювання переходу до слайду з технічною документацією

Специфікація діаграми переходів станів для адміністратора представлено на таблиці 2.16. Показано номер та дія, яка виконується.

Таблиця 2.16 - Специфікація діаграми переходів станів для адміністратора

Номер	Умова/Дія
27	/ Авторизація для входу до адміністративної панелі
28	Ініціювання переходу до сторінки управління контентом
30	Ініціювання переходу до сторінки управління замовленнями
31	Ініціювання переходу до сторінки управління зворотнім завяком і відгуками
27, 29, 32	Ініціювання переходу до сторінки очікування дії / Повернення сторінки очікування дії

2.1.6 Проектування інтерфейсу

Web-інтерфейс – це набір інструментів, що дозволяють користувачу взаємодіяти з веб-сайтом або будь-яким іншим додатком через браузер [8].

Користувачі починають своє знайомство з сайтом з головної сторінки. Головна сторінка вебзастосунку представлена на рисунку 2.5.

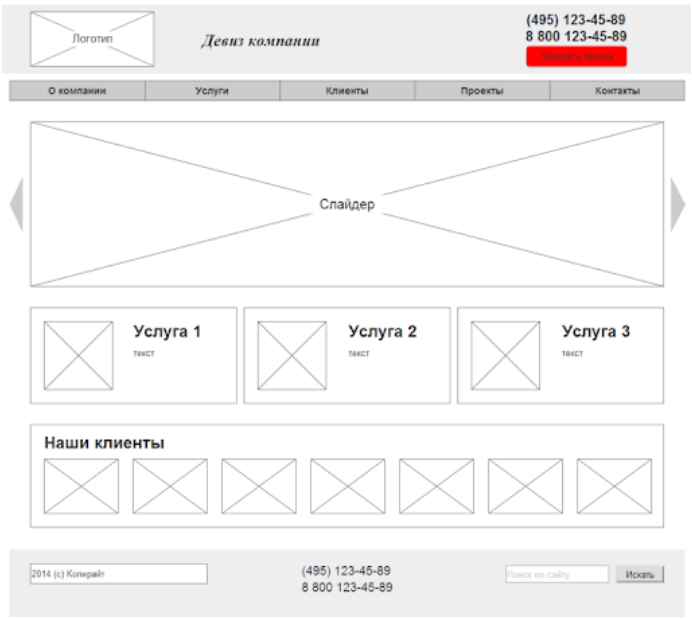


Рисунок 2.5 – Стандартний прототип головної сторінки

2.2 Технічний проект

Технічний проект – це детальний етап розробки інженерних або архітектурних об'єктів, який включає в себе всі необхідні креслення, розрахунки, специфікації та іншу документацію, необхідну для реалізації проекту. Технічний проект є наступним етапом після ескізного проекту і містить всі деталі для будівництва, монтажу та експлуатації об'єкта.

2.2.1 Логічна модель

Логічна модель – це модель даних конкретної предметної області, яка виражається незалежно від конкретного продукту для управління базами даних або технології зберігання (фізичної моделі даних). Вона описується в термінах структур даних, таких як реляційні таблиці та колонки, об'єктно-орієнтовані класи або теги XML. Це відрізняє її від концептуальної моделі даних, яка описує семантику організації без прив'язки до технології. Логічна модель показана на рисунку 2.6.

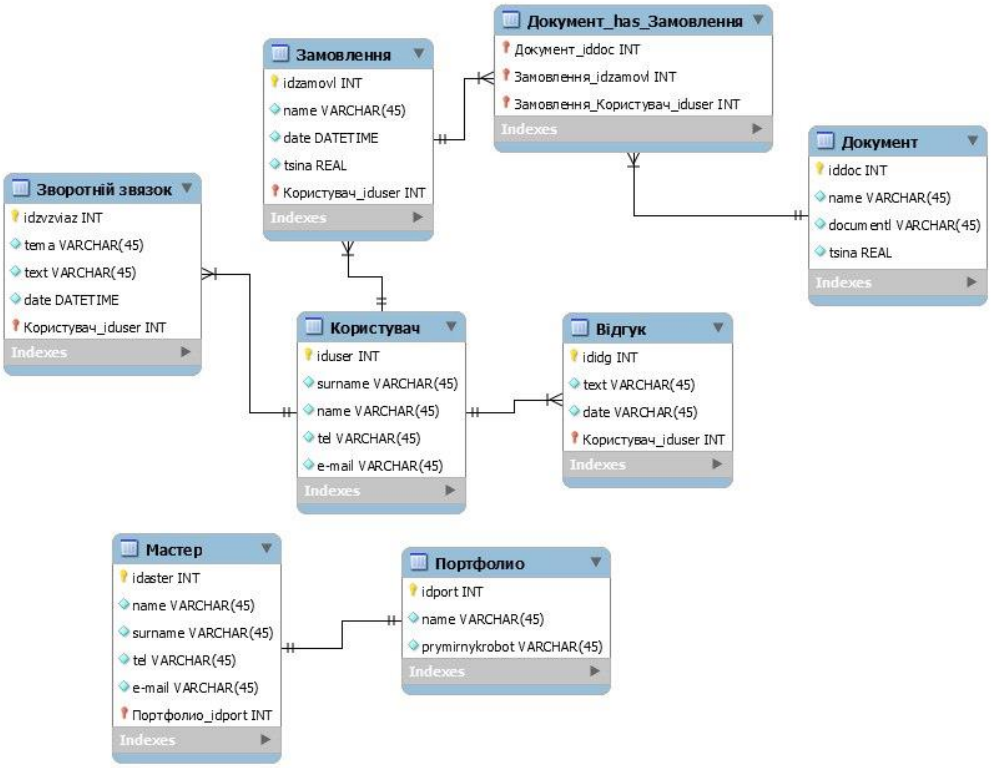


Рисунок 2.6 – Логічна модель даних ПС

2.2.2 Діаграмма взаємодії

Діаграма взаємодії – це одна з моделей опису поведінки груп об'єктів, що взаємодіють в UML. Зазвичай кожна діаграма взаємодії описує поведінку тільки в межах одного варіанта використання. На таких діаграмах відображаються екземпляри об'єктів та повідомлення, якими ці об'єкти обмінюються в рамках цього варіанта використання [9].

На діаграмах послідовностей основна увага приділяється часовому упорядкуванню подій. На них показуються об'єкти та повідомлення, які ці об'єкти приймають і передають. Головна особливість діаграми кооперації полягає в можливості графічно представити не тільки послідовність взаємодії, але і всі структурні відносини між об'єктами, що беруть участь в цій взаємодії в дії.

Діаграма послідовності для суб’єкта «Користувач» при виконанні варіанта використання «Форсування відгуку» представлена на рисунку 2.7.

Діаграма послідовності для суб’єкта «Адміністратор» при виконанні варіанта використання «Управління станом виконання замовлень» представлена на рисунку 2.7.

Діаграма кооперації для суб’єкта «Адміністратор» при виконанні варіанта використання «Управління станом виконання замовлень» представлена на рисунку 2.8.

Рисунок 2.7 – Діаграма послідовності для суб’єкта «Користувач» варіанта використання «Внесення відгуку»

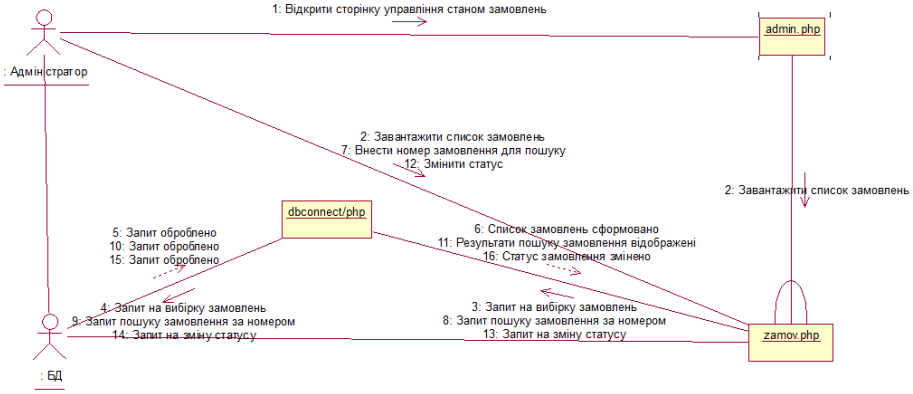


Рисунок 2.8 – Діаграма послідовності для суб’єкта «Адміністратор» варіанта використання «Формування замовлення»

2.2.3 Діаграмма класів

Клас у мові UML використовується для позначення множини об'єктів, що мають однакову структуру, властивості, поведінку і відносини з об'єктами інших класів. Він виступає як шаблон для створення об'єктів. Кожен об'єкт є екземпляром певного класу. Діаграма класів представлена на рисунку 2.9.

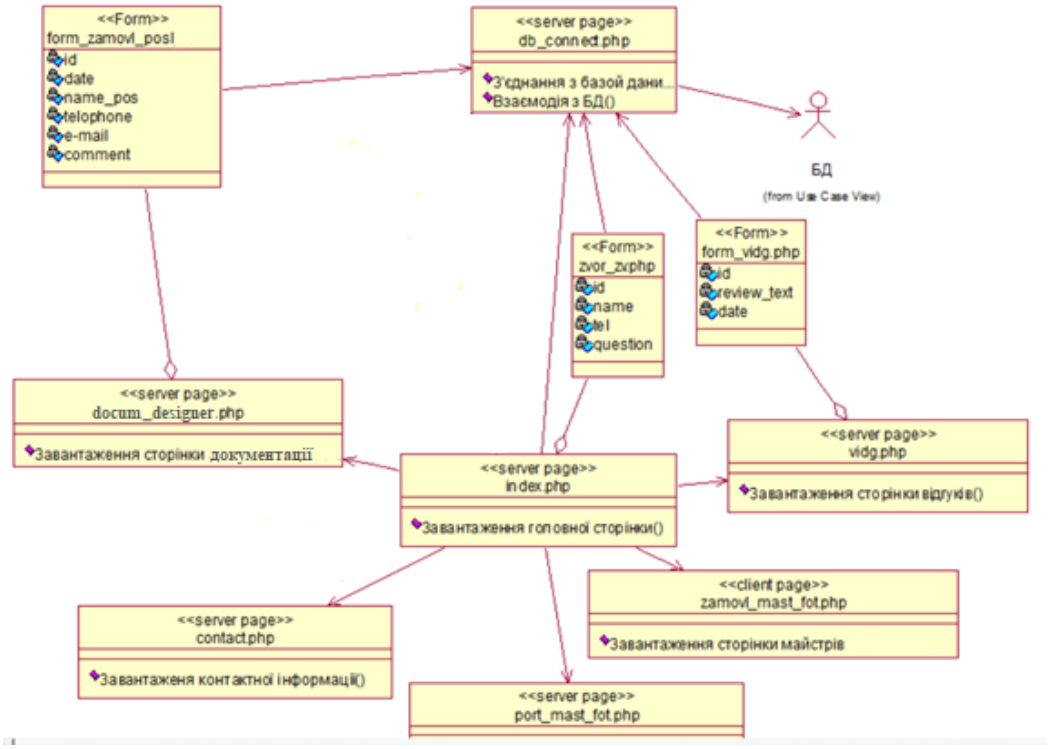


Рисунок 2.9 – Діаграма класів вебзастосунку Project UA

2.2.4 Діаграмма діяльності

На діаграмі діяльності кожен стан відповідає виконанню певної елементарної операції, а перехід до наступного стану відбувається лише після завершення цієї операції в попередньому стані. Графічно діаграма діяльності представлена у вигляді графу діяльності, де вершини відповідають станам дії, а дуги – переходам між цими станами.

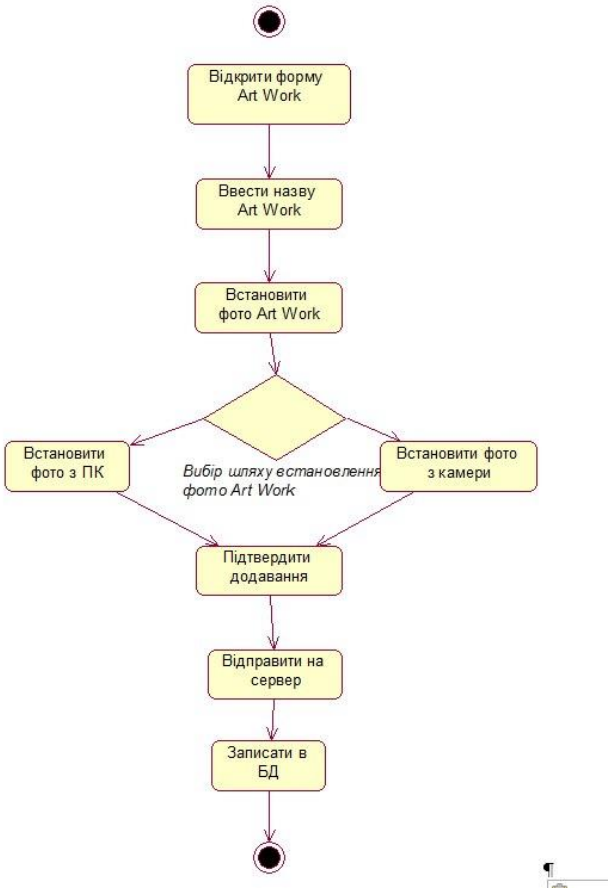


Рисунок 2.10 – Діаграма діяльності для прецеденту «Завантаження професійних робіт»

2.3 Робочий проект

Робочий проект – це завершальний етап проектування, що включає в себе всі необхідні документи, креслення, розрахунки та специфікації, необхідні для будівництва і введення об'єкта в експлуатацію. Робочий проект базується на ескізному та технічному проекті і містить детальну інформацію про всі аспекти будівництва.

2.3.1 Вибір засобів розробки

HTML – це спеціальна мова розмітки, яка використовується для створення веб-документів. Абревіатура HTML означає Hyper Text Markup

Language, що перекладається як "Мова Розмітки Гіпертексту". Гіпертекст – це текст, який містить посилання на інші тексти або веб-документи.

HTML-документ – це веб-документ, створений за допомогою мови розмітки HTML. Він є текстовим файлом, у якому використовуються спеціальні розміткові теги. Вихідний код HTML-документа можна переглянути та відредагувати у будь-якому текстовому редакторі. Програма, яка відображає HTML-документ на основі його розмітки, називається браузером [10].

Фреймворк Bootstrap використовується розробниками по всьому світу, включаючи великі компанії. Він застосовується для створення фронтенд частини вебсайтів та адміністративних інтерфейсів. Bootstrap дозволяє значно прискорити процес розробки в порівнянні з використанням чистого CSS та JavaScript, що робить його популярним серед розробників, зокрема новачків.

Bootstrap складається з набору CSS та JavaScript файлів, які потрібно підключити до вебсторінки. Після підключення можна використовувати такі інструменти, як система колонок (сітка Bootstrap), класи та компоненти для створення різних елементів інтерфейсу.

Каскадні таблиці стилів (CSS) – це мова, яка використовується для визначення зовнішнього вигляду сторінок, створених за допомогою мов розмітки. CSS найчастіше застосовується для візуального оформлення HTML та XHTML сторінок, але також може бути використана для інших типів XML-документів. CSS має різні рівні та профілі, такі як CSS1, CSS2 та CSS3. Профілі являють собою набір правил CSS для певних типів пристроїв або інтерфейсів, наприклад, для принтерів або мобільних пристроїв.

PHP – це популярна мова веб-програмування, створена у 1994 році. PHP використовується для розробки динамічних вебсайтів. Сценарії PHP виконуються на стороні сервера, генеруючи HTML-код, який потім передається на сторону клієнта. Код PHP можна вбудовувати в HTML-сторінки або підключати з зовнішніх файлів. Інтерпретатор мови обробляє код і динамічно формує вебсторінки.

JavaScript (JS) – це динамічна, об'єктно-орієнтована мова

програмування, яка найчастіше використовується для створення інтерактивних елементів на вебсторінках. JavaScript дозволяє коду на стороні клієнта взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером та змінювати структуру і зовнішній вигляд вебсторінки.

phpMyAdmin – це вебзастосунок на мові PHP з відкритим кодом, який надає графічний інтерфейс для адміністрування баз даних MySQL. За допомогою phpMyAdmin можна керувати сервером MySQL через браузер, запускати SQL-запити, переглядати та редагувати вміст таблиць баз даних.

2.3.2 Розробка фізичної моделі

Фізична модель даних (або проектування бази даних) – це подання дизайну даних, яке реалізоване або призначене для реалізації в системі керування базами даних. Завершена фізична модель даних містить всі артефакти бази даних, необхідні для створення зв'язків між таблицями або досягнення мети продуктивності, такі як індекси, визначення обмежень, зв'язані та секціоновані таблиці або кластери [11].

Фізична модель БД представлена в таблицях 2.17-2.23.

Фізична модель таблиці «Користувач» (user) зображена в таблиці 2.17. Вона має 6 ідентифікаторів: номер користувача, ПІБ, телефон, e-mail, логін і пароль користувача.

Таблиця 2.17 - Фізична модель таблиці «Користувач» (user)

Ідентифікатор	Тип	Ключ	Обмеження
id_user (номер користувача)	INTEGER	PK	NOT NULL UNIQUE
PIB (ПІБ)	VARCHAR (50)		NOT NULL
tel (телефон)	VARCHAR (50)		NOT NULL
mail (e-mail)	VARCHAR (50)		NOT NULL
user (логін користувача)	VARCHAR (50)		NOT NULL
pass (пароль користувача)	VARCHAR (50)		NOT NULL

Фізична модель таблиці «Відгук» (Response) зображена в таблиці 2.18. Вона має 4 ідентифікаторів: номер відгуку, текст відгуку, дата, номер користувача.

Таблиця 2.18 - Фізична модель таблиці «Відгук» (Response)

Ідентифікатор	Тип	Ключ	Обмеження
id_res (номер відгуку)	INTEGER	PK	NOT NULL
review_text (текст відгуку)	VARCHAR (50)		NOT NULL
date (дата)	VARCHAR (50)		NOT NULL
id_user (номер користувача)	INTEGER	FK	NOT NULL

Фізична модель таблиці «Замовлення» (Order) зображена в таблиці 2.19. Вона має 6 ідентифікаторів: номер замовлення, дата, кількість, номер користувача, номер стану, номер послуги.

Таблиця 2.19 - Фізична модель таблиці «Замовлення» (Order)

Ідентифікатор	Тип	Ключ	Обмеження
id (номер замовлення)	INTEGER	PK	NOT NULL UNIQUE
date (дата)	VARCHAR (150)		NOT NULL
quantity (кількість)	INTEGER		NOTNULL
id_user (номер користувача)	INTEGER	FK	NOT NULL
id_or_st (номер стану)	INTEGER	FK	NOT NULL
id_serv (номер послуги)	INTEGER	FK	NOT NULL

Фізична модель таблиці «Документ» (Service) зображена в таблиці 2.20. Вона має 3 ідентифікатора: номер послуги, назва послуги, вартість.

Таблиця 2.20 - Фізична модель таблиці «Документ» (Service)

Ідентифікатор	Тип	Ключ	Обмеження
id_serv (номер послуги)	INTEGER	PK	NOT NULL UNIQUE
name_serv (назва послуги)	VARCHAR (50)		NOTNULL
sum (вартість)	INTEGER	FK	NOT NULL

Фізична модель таблиці «Мастер» (Master) зображена в таблиці 2.21. Вона має 3 ідентифікатора: номер дизайнера, ПІБ дизайнера, e-mail.

Таблиця 2.21 - Фізична модель таблиці «Мастер» (Master)

Ідентифікатор	Тип	Ключ	Обмеження
id_mas (номер дизайнера)	INTEGER	PK	NOT NULL UNIQUE
PIB (ПІБ дизайнера)	VARCHAR (50)		NOTNULL
mail (e-mail)	VARCHAR (50)		NOT NULL

Фізична модель таблиці «Зворотній зв’язок» (ZVZviaz) зображена в таблиці 2.22. Вона має 5 ідентифікаторів: код, .тем, текст, дата, id користувача.

Таблиця 2.22 - Фізична модель таблиці «Зворотній зв’язок» (ZVZviaz)

Ідентифікатор	Тип	Ключ	Обмеження
idzvzviaz (код)	INTEGER	PK	NOT NULL UNIQUE
tem	VARCHAR (50)		NOTNULL
taxt	VARCHAR (50)		NOT NULL
date	DATE		NOT NULL
iduser	INTEGER	FK	NOT NULL

Фізична модель таблиці «Документ_has_Замовлення» зображена в таблиці 2.23. Вона має 3 ідентифікатора.

Таблиця 2.23 - Фізична модель таблиці «Документ_has_Замовлення»

Ідентифікатор	Тип	Ключ	Обмеження
id_doc	INTEGER	PK	NOT NULL UNIQUE
id_zamovl	INTEGER	FK	NOTNULL
id_user	INTEGER	FK	NOT NULL

2.3.3 Розробка діаграми розгортання

Діаграма розгортання – це діаграма, на якій показані обчислювальні вузли, компоненти та об'єкти, що виконуються на цих вузлах під час роботи програми. Компоненти представляють робочі екземпляри одиниць коду.

Компоненти, які не мають представлення під час виконання програми, не відображаються на цих діаграмах; замість цього, їх можна показати на діаграмах компонентів. Діаграма розгортання показує робочі екземпляри компонентів, тоді як діаграма компонентів відображає зв'язки між типами компонентів.

Діаграма розгортання представлена на рисунку 2.11.

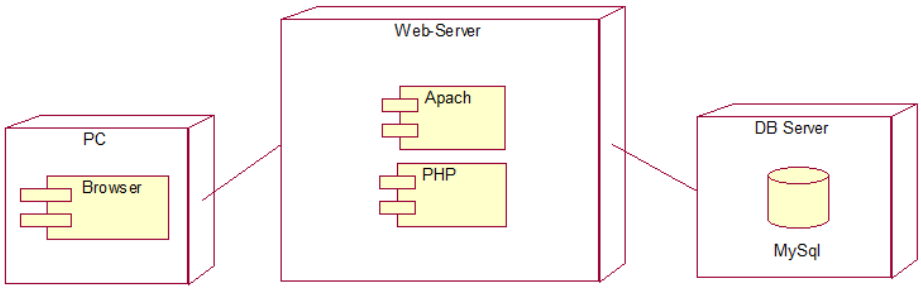


Рисунок 2.11 – Діаграма розгортання

2.4 Вибір та обґрунтування засобів розробки

При розробці програмного забезпечення та веб-застосунків важливо ретельно обрати інструменти та технології, які забезпечать ефективність, надійність і масштабованість кінцевого продукту. У цьому розділі буде розглянуто та обґрунтовано вибір засобів розробки, що використовуються в даному проекті, зокрема UML, бази даних, PHP, HTML, CSS, PhpMyAdmin, системи керування базами даних (СКБД) та Visual Studio Code.

UML. Обґрунтування вибору:

– стандартизація: UML є стандартною мовою моделювання, що широко використовується для візуалізації, специфікації, конструювання та документування програмних систем;

– універсальність: UML підходить для моделювання будь-яких систем, від інформаційних систем масштабу підприємства до розподілених веб-додатків і вбудованих систем реального часу;

– комунікація: Використання UML діаграм (наприклад, діаграм

класів, діаграм послідовностей, діаграм станів) сприяє кращій комунікації між членами команди розробників, а також з замовниками і користувачами.

СКБД. Обґрунтування вибору:

– зберігання даних: База даних необхідна для надійного зберігання, управління та доступу до даних, що використовуються в додатку;

– продуктивність: Використання СКБД дозволяє ефективно керувати великими обсягами даних, забезпечуючи високу швидкість доступу та обробки;

– цілісність та безпека: СКБД забезпечує підтримку транзакцій, цілісність даних та засоби захисту, що критично важливо для надійного функціонування додатку.

PHP. Обґрунтування вибору:

– серверна мова: PHP є популярною серверною мовою програмування, яка дозволяє створювати динамічні веб-сторінки та взаємодіяти з базами даних;

– легкість використання: PHP має низький поріг входження, простий синтаксис і велику кількість бібліотек та фреймворків, що прискорює розробку;

– широке використання: PHP використовується багатьма веб-додатками та системами управління контентом (наприклад, WordPress), що підтверджує його надійність і ефективність.

HTML та CSS. Обґрунтування вибору:

– стандарти веб-розробки: HTML та CSS є стандартами для створення структурованих і стильних веб-сторінок. HTML використовується для розмітки контенту, а CSS – для його стилізації;

– сумісність: HTML і CSS підтримуються всіма сучасними веб-браузерами, що забезпечує крос-браузерну сумісність веб-застосунку;

– простота та гнучкість: HTML і CSS є відносно простими у

вивченні та використанні, що дозволяє швидко створювати веб-інтерфейси.

PhpMyAdmin. Обґрунтування вибору:

– управління базами даних: PhpMyAdmin є зручним веб-інтерфейсом для управління MySQL базами даних. Він дозволяє виконувати операції з базами даних, таблицями, полями, індексами, користувачами тощо;

– легкість використання: Завдяки інтуїтивно зрозумілому інтерфейсу PhpMyAdmin спрощує процес адміністрування баз даних, що особливо корисно для розробників і адміністраторів баз даних;

–функціональність: PhpMyAdmin підтримує широкий спектр функцій для роботи з базами даних, включаючи імпорт/експорт даних;

Visual Studio Code. Обґрунтування вибору:

– мультиплатформеність: Visual Studio Code є мультиплатформеним редактором, що працює на Windows, macOS та Linux, що забезпечує його використання на різних операційних системах;

– підтримка різних мов програмування: Visual Studio Code підтримує безліч мов програмування через розширення, що дозволяє використовувати його для написання коду на HTML, CSS, JavaScript, PHP та багатьох інших мовах;

– інтеграція з системами контролю версій: Інтеграція з Git та іншими системами контролю версій забезпечує легке управління версіями коду прямо з редактора;

– зручність у використанні: Visual Studio Code має інтуїтивно зрозумілий інтерфейс, підтримку автозавершення коду, налагодження та інші функції, які значно спрощують процес розробки;

– розширюваність: Visual Studio Code має багатий екосистему розширень, які дозволяють додавати нові функції, налаштовувати редактор під потреби конкретного проекту та розробника.

3 РЕЗУЛЬТАТИ РОЗРОБКИ

Результатом розробки дипломного проекту є готовий вебзастосунок Project UA.

На сторінці вебзастосунку є можливість переглянути основну інформацію про вебзастосунок. Головна сторінка вебзастосунку представлена на рисунку 3.1.

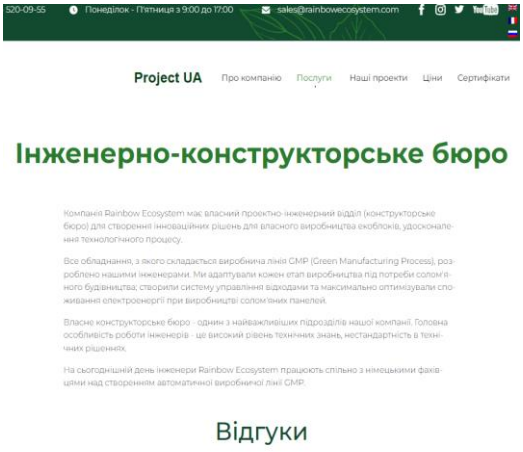


Рисунок 3.1 – Головна сторінка

Користувач може ознайомитись з професійними документами конструкторського бюро Project UA (див. рис. 3.2).

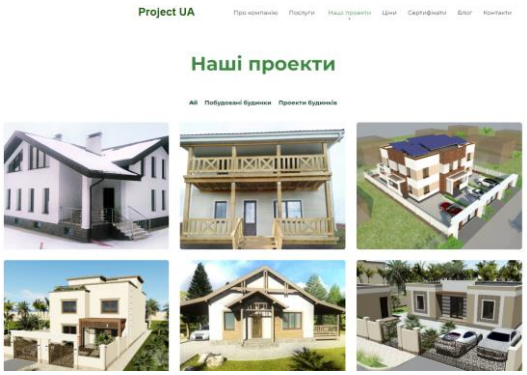


Рисунок 3.2 – Сторінка з представленням професійної документації

Реалізовано можливість перегляду конкретного проєкту з портфоліо майстрів, представлено на рисунку 3.3.

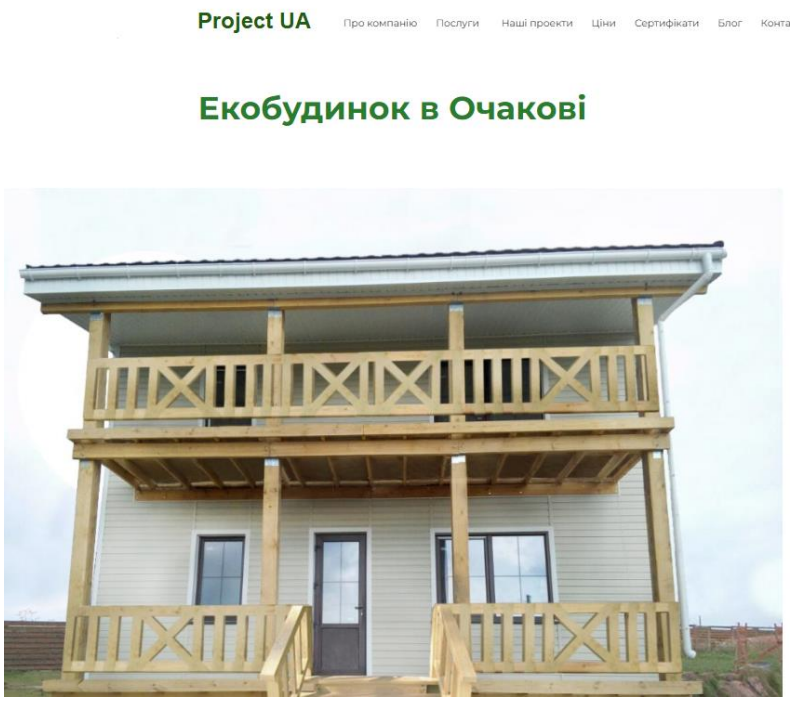


Рисунок 3.3 – Проєкт з Портфоліо майстра

Реалізовано можливість зворотнього зв’язку представлено на рисунку 3.4.

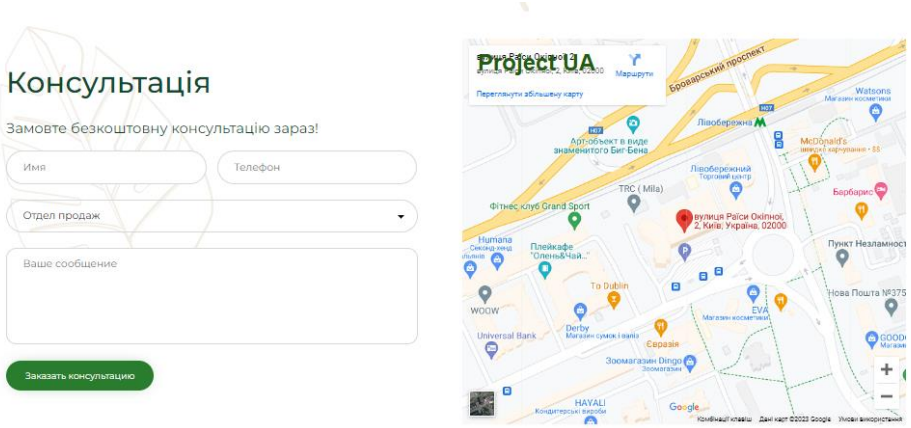


Рисунок 3.4 – Зворотній зв’язок

Реалізовано знайомство з майстром конструкторського бюро Project UA на рисунку 3.5.

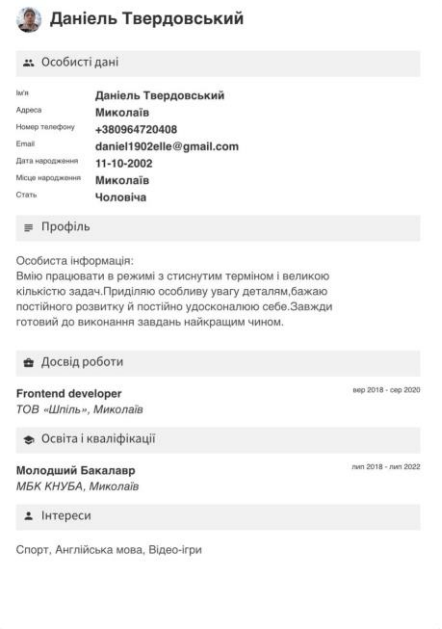


Рисунок 3.5 – Дані майстра конструкторського бюро

3.1. Керівництво вебзастосунку користувач системи

Процес роботи.

У вебзастосунку реалізовано зручне та інформативне меню "Процес роботи". Це меню дозволяє кожному користувачеві детально ознайомитися з етапами та методами виконання робіт майстра. Відвідувачі вебзастосуноку можуть переглядати різні проекти, які були створені дизайнером, що допомагає краще зрозуміти його професійний підхід і техніки.

Ознайомлення з роботами майстра.

Користувачі мають можливість не лише переглядати завершені роботи, а й дізнаватися більше про процес їх створення. Це включає опис етапів роботи, використані матеріали та інструменти, а також пояснення дизайнерських рішень, прийнятих під час роботи над проектами. Такий підхід допомагає потенційним клієнтам оцінити компетенцію та стиль майстра, що

може бути вирішальним фактором при виборі спеціаліста для власних проектів.

Професійні документи.

Крім того, користувачі можуть ознайомитися з професійними документами конструкторського бюро Project UA. У вебзастосунку представлені різноманітні матеріали, включаючи сертифікати, ліцензії, технічну документацію та інші важливі документи, які підтверджують кваліфікацію та досвід бюро. Ця інформація надає додаткову впевненість у професіоналізмі та надійності команди Project UA.

Сторінка відгуків.

У вебзастосунку створена спеціальна сторінка для відгуків, де кожен користувач може залишити свої повідомлення та висловити коментарі стосовно тих чи інших питань. Це може бути корисним для нових відвідувачів, які хочуть дізнатися більше про якість послуг та досвід інших клієнтів. Для того, щоб написати власний відгук, користувач повинен заповнити просту форму, розташовану в нижній частині сторінки. Заповнення форми займає лише кілька хвилин, але дозволяє поділитися своїми враженнями та рекомендаціями з іншими відвідувачами вебзастосуноку.

Заявки для замовлення послуг.

Для того, щоб замовити послуги, користувачам необхідно перейти до спеціальної форми замовлення. Ця форма є важливим інструментом для забезпечення ефективної комунікації між клієнтами та конструкторським бюро Project UA. У формі замовлення користувачі повинні внести всі необхідні дані, що дозволяє бюро точно зрозуміти вимоги та очікування клієнтів.

Процес заповнення форми.

При заповненні форми замовлення клієнт вказує свої контактні дані, включаючи ім’я, прізвище, номер телефону та адресу електронної пошти. Це необхідно для зворотного зв’язку та уточнення деталей замовлення. Також у формі є поля для детального опису бажаної послуги, включаючи специфічні

вимоги та побажання. Це дозволяє майстрам бюро підготуватися до виконання замовлення максимально точно та якісно.

Додаткові опції.

Форма замовлення може містити додаткові опції, такі як вибір дати та часу для консультації або виконання роботи, а також можливість додавання файлів з прикладами або ескізами проєкту. Це сприяє кращому розумінню завдання та забезпечує високу якість виконання робіт.

Підтвердження замовлення.

Після заповнення та відправки форми замовлення, клієнт отримає підтвердження на вказану адресу електронної пошти. У цьому листі буде зазначено, що замовлення прийнято, і найближчим часом з клієнтом зв’яжеться представник бюро для уточнення всіх деталей. Це підтвердження також слугує нагадуванням для клієнта про здійснене замовлення і надає можливість відстежувати його статус.

3.2. Керівництво вебзастосунку адміністратор системи

Для роботи з текстовою та графічною інформацією У вебзастосунку використовується візуальний редактор HTML-коду Visual Studio Code. Він дозволяє форматувати, редагувати текст, вставляти графічні зображення на сторінку тощо. Також Visual Studio Code підтримує копіювання тексту з текстового процесора MS Word.

Копіювання тексту здійснюється за допомогою буфера обміну. Щоб скопіювати текст з MS Word до Visual Studio Code, потрібно виконати наступні кроки:

- відкрити документ у текстовому процесорі Microsoft Word;
- виділити фрагмент документу (або весь документ), який потрібно скопіювати на сайт. Для цього клацніть лівою клавішею миші на початку фрагменту і, утримуючи клавішу, перемістіть курсор миші до кінця фрагменту, який потрібно виділити;

- скопіювати виділений фрагмент до буфера обміну. Натисніть правою клавішею миші на виділеному фрагменті та виберіть у контекстному меню пункт «Копировать»;
- відкрити систему адміністрування у вікні браузера, вибрати сторінку, на яку потрібно вставити текст;
- вставити скопійований текст з буфера обміну у робочу область FCKeditor:
  - якщо фрагмент тексту з Microsoft Word не містить таблиць, для вставки цього тексту потрібно натиснути кнопку «Вставити тільки текст» на панелі інструментів Visual Studio Code;
  - якщо фрагмент тексту з буфера обміну містить таблиці, натисніть кнопку «Вставити з Word».

Після натискання відповідної кнопки з’явиться діалогове вікно. Вставте інформацію з буфера обміну за допомогою комбінації клавіш CTRL-V. Зніміть позначки біля опцій «Ігнорувати налаштування шрифтів» і «Видалити налаштування стилів», потім натисніть кнопку «ОК».

Відредагуйте та відформатуйте текст за допомогою візуального редактора Visual Studio Code. Для зручності розгорніть панель редактора на весь екран, натиснувши один раз лівою клавішею миші на кнопку «Розгорнути вікно редактора».

Для автоматизованої керування вебзастосунком конструкторського бюро розроблена Адмін панель з модулями MAIN, BLOG, GALLERY, SYSTEM зображена на рисунку. Модуль MAIN зображено на рисунку 3.6.

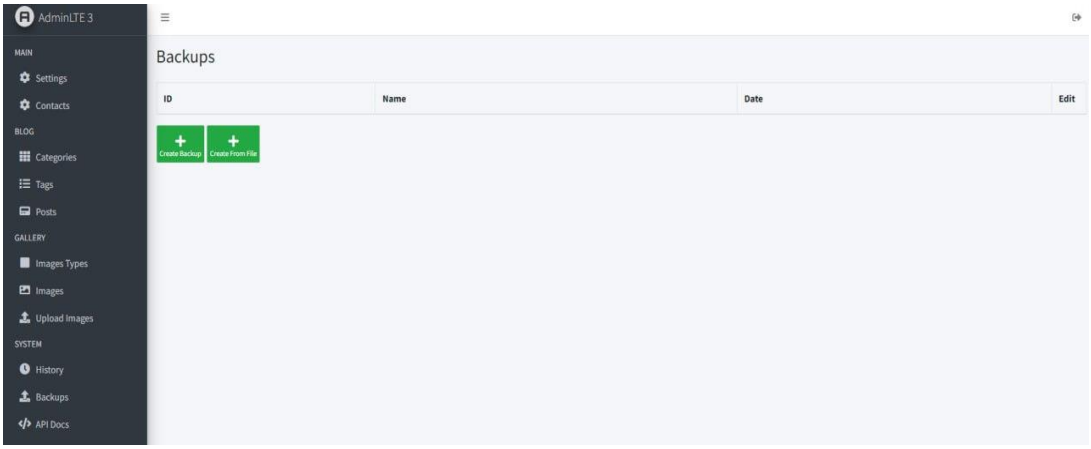


Рисунок 3.6 – Адміністративна панель

Форма Авторизації зображено на рисунку 3.7.

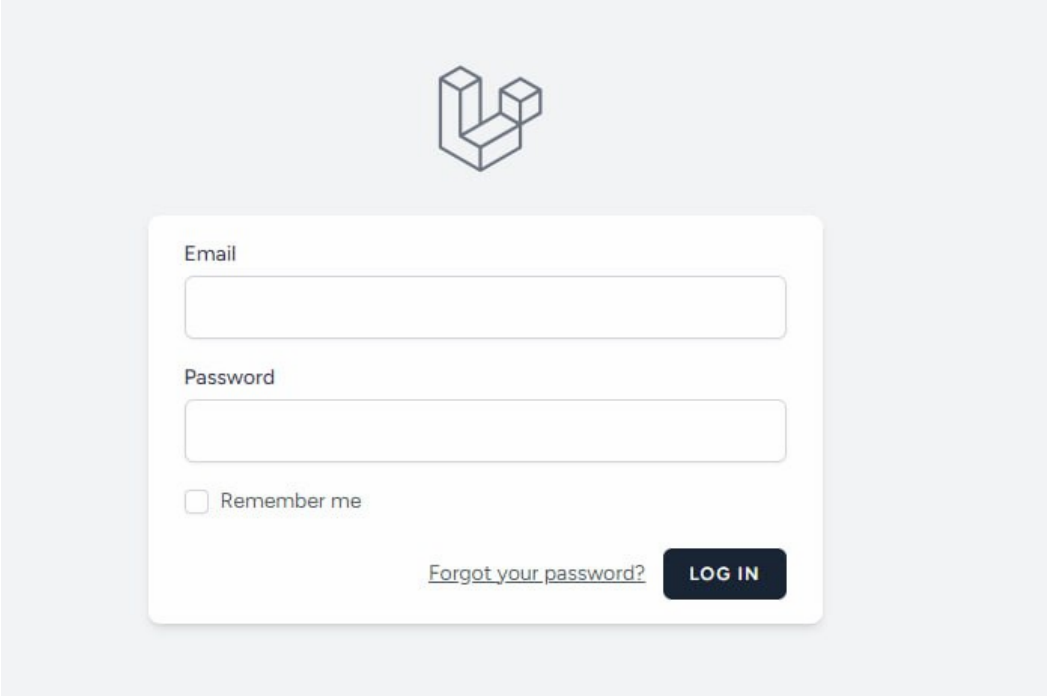


Рисунок 3.7 – Авторизація

Модуль MAIN складається з модулів Налаштування системи та Контакти, зображено на рисунку 3.8.

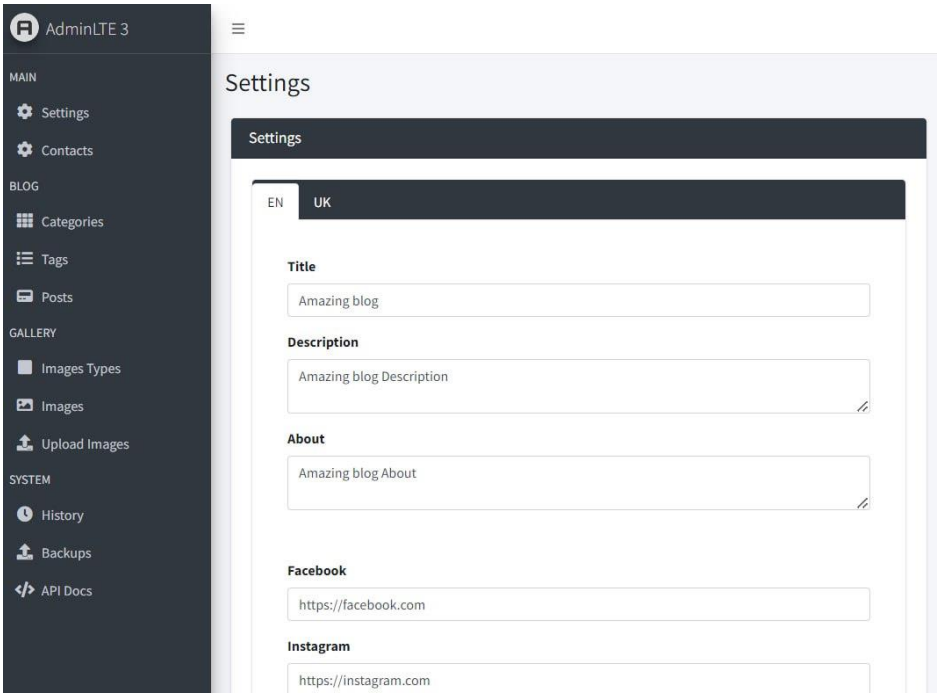


Рисунок 3.8 – Модуль MAIN

Модуль BLOG з панеллю Категорії забражено на рисунку 3.9.

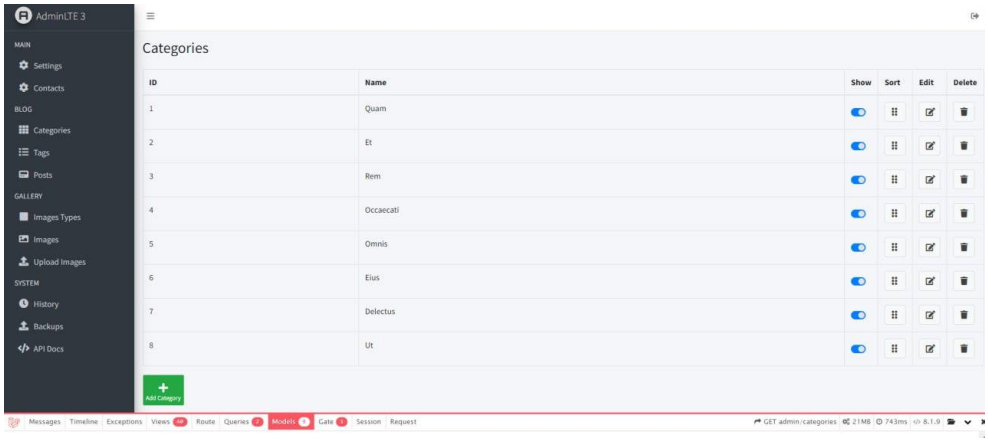
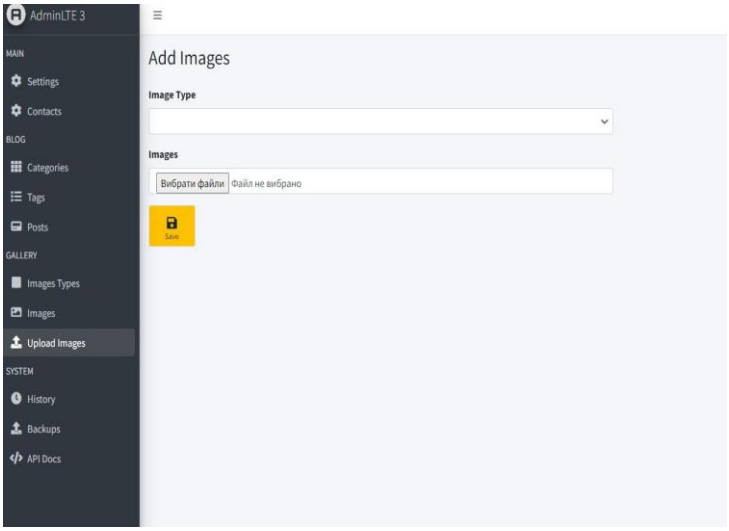


Рисунок 3.9 – Категорії Блогу

Модуль GALLERY відповідає за портфоліо, документи, креслення, дизайнерські роботи, підмодуль Додати рисунки зображено на рисунку 3.10.



3.10 – Модуль додавання рисунків

ВИСНОВКИ

В рамках дипломного проекту було проведено всебічний аналіз завдань та функціоналу вебзастосунку для архітектурно-будівельного конструкторського бюро Project UA. Ґрунтовно вивчено та обґрунтовано вибір типу сайту з урахуванням потреб та цілей замовника.

Проведено огляд існуючих рішень та сучасних інструментів розробки вебсайтів, що дозволило обрати оптимальну технологію для створення вебзастосунку. В результаті виконання дипломного проекту було розроблено вебзастосунок для архітектурно-будівельного конструкторського бюро Project UA, який володіє наступними ключовими функціональними можливостями:

- надання детальної інформації про дизайнера та конструктора, а також перелік їхніх послуг;
- демонстрація портфоліо виконаних робіт з можливістю сортування та фільтрації за різними критеріями;
- зручний інтерфейс для замовлення документів у форматах 2D та 3D, доповнених технічною документацією.

Слід зазначити, що поставлена на початку дипломного проектування мета була досягнута в повній мірі. Розроблений вебзастосунок Project UA відповідає всім вимогам та очікуванням замовника, ефективно презентує послуги бюро та сприяє залученню нових клієнтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація Bootstrap 5.3 [Електронний ресурс]. – Режим доступу: https://getbootstrap.com/docs/5.3 – дата звернення 14.01.2024.
2. Документація MySQL [Електронний ресурс]. – Режим доступу: https://www.php.net/manual/en/book.mysqli.php – дата звернення 15.01.2024.
3. Документація PHP [Електронний ресурс]. – Режим доступу: https://www.php.net/manual/en/ – дата звернення 10.01.2024.
4. Документація phpMyAdmin [Електронний ресурс]. – Режим доступу: https://www.phpmyadmin.net/docs/ – дата звернення 10.01.2024.
5. Документація XAMPP [Електронний ресурс]. – Режим доступу: https://www.apachefriends.org/docs/ – дата звернення 13.01.2024.
6. Швець О. Занурення в патерни проектування. - 2021. -240 с. ISBN - 978-620-2-92280-5.
7. Приклад вебзастосунку візуального графічного дизайну [Електронний ресурс]. – Режим доступу: https://varlamovadesign.com/ – дата звернення 04.01.2024.
8. Приклад вебзастосунку маркетингової платформи у соціальній мережі Instagram дизайнерських і конструкторських портфоліо [Електронний ресурс]. – Режим доступу: https://later.com/ – дата звернення 04.01.2024.
9. Звернення за пошуком ідей [Електронний ресурс]. – Режим доступу: https://github.com/ – дата звернення 12.12.2023.
10. Роберт Мартин, Чиста архітектура. Мистецтво розробки програмного забезпечення. – 2019. – 142 с. ISBN - 978-617-09-5286-8.
11. Шлях самурая [Електронний ресурс]. – Режим доступу: https://www.youtube.com/watch?v=gb7gMluAeao – дата звернення 05.09.2023.

ДОДАТОК А – Код розробки

Лістинг А.1– Код головної сторінки

```
Post.blade.php
<div class="item post grid-sizer col-md-6">
    <figure class="overlay overlay1 rounded mb-30">
        <a href="{{ route('posts.show', compact(['post'])) }}">
            <img src="{{ asset($post->image) }}" alt="" />
        </a>
        <figcaption>
            <h5 class="from-top mb-0">Read More</h5>
        </figcaption>
    </figure>
    <div class="category">
        @if($post->category)
        <a href="{{ route('posts.index', ['category' => $post->category-
>id]) }}"
           class="badge badge-pill bg-purple">{{ $post->category->title
}}</a>
        @endif
    </div>
    <div class="category">
        @if(!$post->show)
        <span class="badge badge-pill bg-red">UNPUBLISHED</span>
        @endif
    </div>
    <h2 class="post-title">
        <a href="{{ route('posts.show', compact(['post'])) }}">{{ $post-
>title }}</a>
    </h2>
    <div class="post-content">
        <p>{{ $post->description }}</p>
    </div>

    <div class="meta mb-0">
        <span class="comments">
            BY {{ $post->user->name }}
        </span>
        <span class="date">
            <i class="jam jam-heart-f"></i> {{ $post->likes }} Likes
        </span>
        <span class="date">
            <i class="jam jam-clock"></i>{{ $post->created_at->format('d
M Y') }}</span>
        <span class="comments">
            <i class="jam jam-message-alt"></i>
            <a href="#">{{ $post->commentsCount }} Comments</a>
        </span>
    </div>
</div>
```

Лістинг А.2 – Header.blade.php

```
<div class="wrapper image-wrapper bg-image page-title-wrapper inverse-
text"
     data-image-src="{{ asset('style/images/art/bg31.jpg') }}">
```

```
            <div class="container inner text-center">
                <div class="space90"></div>
                <h1 class="page-title">{{ $settings?->title }}</h1>
                <p class="lead">{{ $settings?->description }}</p>
            </div>

        </div>
```

Лістинг А.3 – Users

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Builder;
use Illuminate\Database\Eloquent\Casts\Attribute;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Prunable;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\BelongsToMany;
use Illuminate\Database\Eloquent\Relations\HasMany;
use Illuminate\Database\Eloquent\Relations\MorphMany;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\Storage;

class Post extends Model
{
    use HasFactory, SoftDeletes, Prunable;

    const PER_PAGE = 6;

    protected $fillable = ['title', 'slug', 'description', 'body',
'image', 'category_id', 'user_id', 'show', 'created_at'];

    public function prunable()
    {
        return static::where('created_at', '<=', now()->subWeek());
    }

    /**
     * Get the route key for the model.
     */
    public function getRouteKeyName(): string
    {
        return 'slug';
    }

    public function category(): BelongsTo
    {
        return $this->belongsTo(Category::class);
    }

    public function notes(): MorphMany
    {
        return $this->morphMany(Note::class, 'notable');
    }

    public function reactions(): BelongsToMany
    {
        return $this->belongsToMany(
            User::class,
            'reactions',
```

```
                'post_id',
                'user_id',
            );
        }

        public function tags(): BelongsToMany
        {
            return $this->belongsToMany(Tag::class);
        }
        public function user(): BelongsTo
        {
            return $this->belongsTo(User::class);
        }

        public function comments(): HasMany
        {
            return $this->hasMany(Comment::class)
                ->whereNull('comment_id')
                ->with('comments.user');
        }

    //    public function blocks(): HasMany
    //    {
    //        return $this->hasMany(Block::class);
    //    }

        public function blocks(): MorphMany
        {
            return $this->morphMany(Block::class, 'blockable');
        }

        public function scopeCategory(Builder $query, int|null $categoryId)
:void
        {
            $query->when($categoryId, function ($query) use ($categoryId) {
            $query->where('category_id', $categoryId);
             });
        }

        public function scopeTag(Builder $query, int|null $tagId): void
        {
            $query->when($tagId, function ($query) use ($tagId) {
            $query->whereHas('tags', function ($query) use ($tagId) {
                $query->where('tag_id', $tagId);
            });
        });
        }

        public function scopeSearch(Builder $query, string|null $search):
void
        {
            $query->when($search, function ($query) use ($search) {
            $query->where('title', 'like', '%'.$search.'%')
                   ->orWhere('description', 'like', '%'.$search.'%');
                });
        }

        public function scopeDateFilter(Builder $query, string|null $date)
        {
            $query->when($date, function ($query) use ($date) {
                $dateArray = explode('.', $date);
                $month = $dateArray[0];
                $year = $dateArray[1];
```

```
                $query->whereMonth('created_at',                    $month)-
>whereYear('created_at', $year);

            });
        }

        public function deleteImage(): void
        {
            if (\File::exists(public_path($this->image))) {
                \File::delete(public_path($this->image));
            }
            }
        public static function uploadImage($image) :array
        {
            $fields = [];
            if ($image) {

                $filename    =    rand(1000000000,    9999999999).'.'.$image-
>getClientOriginalExtension();
                Storage::disk('public')->putFileAs('images',         $image,
$filename);
                $fields+=['image' => "storage/images/$filename"];
            }
            return $fields;
        }
```

Лістинг А.4 – Controlers

```
<?php

namespace App\Http\Controllers;

use App\Models\Setting;
use App\Models\User;
use App\Models\Post;
use Illuminate\Http\Request;

class UserController extends Controller
{
    public function feed(User $user)
    {
        $settings = Setting::first();
        $authors     =     auth()->user()->load('authors:id')->authors-
>pluck('id')->toArray();
        $posts       =       Post::whereIn('user_id',       $authors)-
>paginate(Post::PER_PAGE);
        return view('posts.index', compact(['posts', 'settings']));
    }

    public function authors(User $user)
    {
        $authors = auth()->user()->authors;
        return view('authors.index', compact('authors'));
    }

    public function readers()
    {
        $authors = auth()->user()->readers;
        return view('authors.index', compact('authors'));
    }
```

```
        public function posts()
        {
            $settings = Setting::first();
            $posts = Post::where('user_id', auth()->id())->paginate(6);
          return view('posts.index', compact('posts', 'settings'));
       }
    } }
```

Лістинг А.5 – Routes/web.php

```
    <?php

    use App\Http\Controllers as Web;
    use App\Http\Controllers\Admin as Admin;
    use Illuminate\Support\Facades\Route;

    Route::redirect('/', 'posts');

    Route::resources([
        'posts' => Web\PostController::class,
        'comments' => Web\CommentController::class,
    ]);

    Route::post('contacts', [Web\ContactController::class, 'store'])-
>name('contacts.store');


    Route::post('changeLocale', [Web\LocaleController::class, 'changeLo-
cale'])->name('change-locale');

    Route::get('posts/{post}/add-block', [Web\PostController::class, 'add-
Block'])->name('posts.add-block');
    Route::post('posts/{post}/store-block', [Web\PostController::class,
'storeBlock'])->name('posts.store-block');


    Route::post('blocks/store', [Web\BlockController::class, 'store'])-
>name('blocks.store');

    Route::get('authors', [Web\AuthorController::class, 'index'])->name('au-
thors.index');


    Route::get('admin', [Admin\AdminController::class, 'index'])
        ->name('admin.index')->middleware(['admin']);

    Route::group(['middleware' => 'auth'], function () {

        Route::put('sort', [Admin\AdminController::class, 'sort'])-
>name('admin.sort');
        Route::put('on-off', [Admin\AdminController::class, 'onOff'])-
>name('admin.on-off');
        Route::delete('delete-item', [Admin\AdminController::class,
'deleteItem'])->name('admin.deleteItem');

        Route::group(['middleware' => 'admin', 'prefix' => 'admin', 'as' =>
```

```
'admin.'], function () {
              Route::resource('categories', Admin\CategoryController::class);
              Route::resource('tags', Admin\TagController::class);
              Route::resource('posts', Admin\PostController::class);
              Route::resource('imagetypes', Admin\ImageTypeController::class);
              Route::resource('images', Admin\ImageController::class);

              Route::get('contacts',   [Admin\ContactController::class,   'in-
dex'])->name('contacts.index');
              Route::get('contacts/export',   [Admin\ContactController::class,
'export'])->name('contacts.export');
              Route::delete('contacts/{contact}',       [Admin\ContactControl-
ler::class, 'destroy'])
                   ->name('contacts.destroy');

              Route::get('backups/create-from-file',     [Admin\BackupControl-
ler::class, 'createFromFile'])
                   ->name('backups.create-from-file');
              Route::resource('backups', Admin\BackupController::class);
              Route::post('backups/{backup}/recover',    [Admin\BackupControl-
ler::class, 'recover'])
                   ->name('backups.recover');
              Route::post('backups/store-from-file',     [Admin\BackupControl-
ler::class, 'storeFromFile'])
                   ->name('backups.store-from-file');
              Route::get('backups/{backup}/download',    [Admin\BackupControl-
ler::class, 'download'])
                   ->name('backups.download');
              Route::delete('events/truncate',  [Admin\EventController::class,
'truncate'])->name('events.truncate');
              Route::delete('events/delete-day',          [Admin\EventControl-
ler::class, 'deleteDay'])->name('events.delete-day');
              Route::resource('events', Admin\EventController::class);


              Route::resource('settings', Admin\SettingController::class);
          });

          Route::group(['controller' => Web\SubscribeController::class], func-
tion () {
              Route::post('subscribe/{user}', 'subscribe')->name('subscribe');
              Route::delete('subscribe/{user}',   'unsubscribe')->name('unsub-
scribe');
          });

          Route::group(['controller' => Web\UserController::class, 'as' => 'us-
ers.'], function () {
              Route::get('users/feed', 'feed')->name('feed');
              Route::get('users/authors', 'authors')->name('authors');
              Route::get('users/readers', 'readers')->name('readers');
              Route::get('users/posts', 'posts')->name('posts');
          });

          Route::group(['controller' => Web\ReactionController::class], func-
tion () {
              Route::post('reactions/{post}',            'store')->name('reac-
tions.store');
              Route::delete('reactions/{post}',        'destroy')->name('reac-
tions.destroy');
          });
      });
      require __DIR__.'/auth.php';
```