МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет Інститут економіки і бізнес освіти

Кафедра Економіки та цифрового бізнесу

Спеціальність «Комп'ютерні науки»

Форма навчання Денна

Група КН-21-1

КВАЛІФІКАЦІЙНА РОБОТА

Бурей Юлії Степанівни

(прізвище, ім'я, по батькові здобувача)

на тему «Розробка та впровадження WEB-сайту модельного агентства»

(повна назва теми)

за матеріалами

(повна назва бази дослідження)

науковий керівник к.е.н., доцент _____ Соловйова В.В.

(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від _09 червня_____ 2025__р. № 12
Завідувач кафедри

(підпис)

К.е.н., доцент В.М. Радько

Наук. ступінь, вчене звання *Ініціали, прізвище*

Кривий Ріг – 2025

ЗАТВЕРДЖЕНО
Наказ Міністерства освіти і науки, молоді та спорту України
29 березня 2012 року № 384

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ ТА БІЗНЕС-ОСВІТИ
(повне найменування вищого навчального закладу)

Кафедра економіки та цифрового бізнесу
Освітній ступінь бакалавр
Спеціальність «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри _____В.М. Радько

"07" квітня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА ЗДОБУВАЧУ

Бурей Юлії Степанівні

1. Тема роботи «Розробка та впровадження WEB-сайту модельного агентства»

науковий керівник роботи Соловйова В.В.,
затверджені наказом вищого навчального закладу від «04» квітня 2025 р. № 224-ст (д/ф)
№ 151-ст (з/ф)

2. Строк подання здобувачем роботи 31.05.2025р.
3. Зміст кваліфікаційної роботи бакалавра, об'єкт, предмет та мета дослідження:

Розділ 1 ДОСЛІДЖЕННЯ СТРУКТУРИ WEB-САЙТІВ «МОДЕЛЬНОГО АГЕНТСТВА»
Розділ 2 ПРОЄКТУВАННЯ WEB-САЙТУ «МОДЕЛЬНОГО АГЕНТСТВА»
Розділ 3 РЕАЛІЗАЦІЯ WEB-РЕСУРСУ РОЗРОБКИ WEB-САЙТУ «МОДЕЛЬНОГО АГЕНТСТВА»
Об'єкт дослідження – процес розробки та впровадження WEB-сайту модельного агентства
Предмет дослідження - розробка та впровадження WEB-сайту модельного агентства

Мета кваліфікаційної роботи бакалавра – розробка і впровадження сучасного WEB-сайту для модельного агентства

4. Дата видачі завдання 04.04.2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	до 28.04.2025р.	25.04.2025
2	Підготовка розділу 2	до 16.05.2025р.	15.05.2025
3	Підготовка розділу 3	до 30.05.2025р.	29.05.2025
4	Реєстрація завершеної дипломної роботи	до 31.05.2025р.	30.05.2025
5	Отримання відгуку від наукового керівника	03-04.06.2025р.	04.06.2025
6	Отримання зовнішньої рецензії	05-06.06.2025р.	06.06.2025
7	Перевірка кваліфікаційної роботи на плагіат	02-09.06.2025р.	04.06.2025
8	Попередній захист кваліфікаційної роботи на кафедрі	03.06.2025р.	03.06.2025
9	Допуск кафедрою кваліфікаційної роботи до захисту	09.06.2025р.	09.06.2025
10	Підготовка студента до захисту в ЕК	до 17.06.2025р.	17.06.2025

Завдання підготував науковий керівник _____ Соловйова В.В._____
(підпис) (прізвище та ініціали)

Завдання одержав здобувач _____ __Бурей Ю.С._____
(підпис) (прізвище та ініціали)

Примітки:

1. Форму призначено для видачі завдання здобувачу на виконання кваліфікаційної роботи бакалавра і контролю за ходом роботи з боку кафедри.
2. Розробляється керівником кваліфікаційної роботи. Видається кафедрою.
3. Формат бланка А4 (210 × 297 мм), 2 сторінки.

РЕФЕРАТ

Робота містить 55 сторінки, 28 рисунків, 29 джерел, 1 таблицю, 9 додатків.

Об'єкт дослідження: процес розробки та впровадження WEB-сайту модельного агентства.

Предмет дослідження - розробка та впровадження WEB-сайту модельного агентства.

Мета: створити сучасний WEB-сайт для модельної агенції, який дозволить адміністраторам, моделям і фотографам легко взаємодіяти через онлайн-платформу.

У результаті дослідження було проаналізовано особливості створення WEB-сайту для модельного агентства, враховуючи потреби різних ролей користувачів. Було визначено функціональні та нефункціональні вимоги до системи, спроєктовано архітектуру клієнт-серверного типу та макет інтерфейсу користувача. Обґрунтовано використання сучасних технологій, реалізовано повноцінний WEB-інтерфейс з особистими кабінетами, формою подачі заявок, динамічним контентом і системою авторизації. Розроблений ресурс виконує як презентаційні, так і управлінські функції, спрямовані на цифрову підтримку бізнес-процесів модельної агенції.

Сфера застосування: результати цієї роботи можуть бути використані в цифрових платформах модельного бізнесу, кастинг-системах, онлайн-базах анкет моделей і фотографів, а також у WEB-додатках, що передбачають обробку заявок, керування контентом і взаємодію між ролями в межах однієї платформи.

WEB-РОЗРОБКА, МОДЕЛЬНЕ АГЕНТСТВО, HTML, JAVASCRIPT, NODE.JS, REACT, CSS, MYSQL, АВТЕНТИФІКАЦІЯ, WEB-САЙТ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

UI/UX – User Interface / User Experience

SEO – Search Engine Optimization

СУБД – Система Управління Базами Даних

CMS – Content Management System

HTML – HyperText Markup Language

HTTPS – HyperText Transfer Protocol Secure

CSS – Cascading Style Sheets

SASS – Syntactically Awesome Stylesheets

LESS – Leaner Style Sheets

JS – JavaScript

API – Application Programming Interface

REST – Representational State Transfer

JSON – JavaScript Object Notation

XML – eXtensible Markup Language

CI/CD – Continuous Integration / Continuous Deployment

VPS – Virtual Private Server

SSL – Secure Sockets Layer

CAPTCHA – Completely Automated Public Turing test to tell Computers and Humans Apart

SPA – Single Page Application

MVC – Model-View-Controller

JWT – JSON Web Token

SQL – Structured Query Language

ВСТУП

У сучасних умовах цифрової трансформації суспільства розвиток бізнесу сильно залежить від ефективної онлайн-присутності. Це особливо стосується творчих індустрій, зокрема модельного бізнесу, який базується на візуальній комунікації, динамічності та високій конкуренції. У зв'язку з цим розробка та впровадження сучасного WEB-сайту модельного агентства є своєчасною задачею, яка поєднує технічні та дизайнерські аспекти.

Актуальність обраної теми зумовлена висхідною потребою модельних агенцій у цифрових рішеннях, які дозволяють автоматизувати спілкування з клієнтами, покращити ефективність кастингів, демонструвати моделі за допомогою інтерактивного портфоліо та підтримувати зв'язок із фотографами. Сайт агенції – це більше, ніж просто сайт; це платформа, яка дозволяє учасникам індустрії створювати онлайн-кабінети, керувати модельною базою та організовувати фотосесії.

Ступінь вивченості проблеми свідчить про те, що тема розробки WEB-сайтів у сфері модельного бізнесу достатньо активно обговорюється у фаховій літературі. Однак прикладні питання реалізації повнофункціонального ресурсу з підтримкою ролей, базами даних, захистом персональних даних і кастомізованим UI/UX залишаються актуальними для подальшого дослідження. У процесі підготовки було проведено огляд офіційних сайтів IMG Models, ONE MOTHER AGENCY та Ego Models Management. Це дозволило визначити поточні тенденції та сформулювати вимоги до функціональності й дизайну майбутнього проєкту.

Специфіка джерельної бази містить наукові та практичні публікації з тем розробки WEB-додатків, UI/UX-дизайну, використання React, Node.js і MySQL, документацію до технологій, а також методичні рекомендації з дипломного проєктування. Додатково використовувались матеріали онлайн-курсів, офіційні

гіди (наприклад, документація React та Express), досвід фреймворків і приклади коду з відкритих репозиторіїв.

Об'єктом дослідження є процес розробки та впровадження WEB-сайту модельного агентства. Цей процес охоплює технічне проєктування, реалізацію серверної та клієнтської логіки, адміністрування заявок та інтеграцію користувацьких ролей.

Предмет дослідження - розробка та впровадження WEB-сайту модельного агентства.

Метою кваліфікаційної роботи є розробка і впровадження сучасного WEB-сайту для модельного агентства. Цей сайт повинен відповідати вимогам візуальної привабливості, інтерактивності, продуктивності та безпеки, а також забезпечувати зручний доступ до послуг агентства для моделей, фотографів і адміністраторів.

У процесі дослідження вирішувалися наступні завдання:

– провести аналіз сучасних WEB-технологій і ресурсів, які використовуються в модельному бізнесі;

– обґрунтувати вибір технологій для створення сайту;

– спроєктувати архітектуру WEB-додатка з урахуванням ролей користувачів;

– реалізувати фронтенд і бекенд частини ресурсу;

– розробити користувацький інтерфейс із підтримкою мультимедійного контенту;

– забезпечити процеси реєстрації, авторизації та захисту персональних даних;

– протестувати основні функціональні блоки сайту;

– оцінити ефективність реалізованого рішення.

Методика дослідження базується на застосуванні компонентного підходу до розробки користувацьких інтерфейсів (React), побудові RESTful API (Express.js),

використанні реляційної СУБД (MySQL), тестуванні функціональності за допомогою Postman, перевірці безпеки та ефективності WEB-ресурсу. Значна увага приділялася прототипуванню, UI/UX-дизайну та інтерактивній взаємодії з користувачами.

Практична значущість результатів дослідження проявляється в можливості використовувати розроблений ресурс як основу для реальної роботи цифрової платформи модельного агентства. Окрім цього, кваліфікаційна робота може слугувати методичним прикладом для студентів і молодих розробників у сфері створення спеціалізованих WEB-додатків із підтримкою користувацьких сценаріїв, системи автентифікації, обробки мультимедійних даних і SEO-оптимізації.

Отже, обрана тема є актуальною, а реалізований підхід до розробки сайту модельної агенції дозволяє не лише вирішити прикладне завдання, але й показує наявність глибоких знань і навичок у сфері сучасних інформаційних технологій.

РОЗДІЛ 1
ДОСЛІДЖЕННЯ СТРУКТУРИ WEB-САЙТІВ «МОДЕЛЬНОГО АГЕНТСТВА»

1.1 Аналіз сучасних технологій та інструментів для WEB-сайту модельного бізнесу та його потреб у WEB-ресурсах

Сьогодні, в часи цифрових технологій та глобальної мережевої інтеграції наявність функціонального та ефективного WEB-сайту є критично важливою умовою для розвитку бізнесу в будь-якій сфері [1]. Це особливо актуально для представників творчих індустрій, зокрема модельного бізнесу, де візуальна компонента є найважливішою. Як суб'єкт ринку моди, стилю та краси, модельна агенція потребує WEB-ресурсу, який демонструє його діяльність і дозволяє спілкуватися з моделями, фотографами, дизайнерами та іншими учасниками індустрії.

Модельне агентство використовує WEB-сайт для виконання різних завдань, які вимагають відповідних технічних і візуальних рішень. Розглянемо основні з них.

Інформаційна функція. Сайт має містити повну інформацію про агенцію, включаючи історію його створення, напрямки діяльності, команду, успішні кейси, перелік послуг тощо. Блок з анкетами моделей повинен містити портфоліо з фотографіями, відео, параметрами та досвідом роботи. Це дозволяє кастинг-менеджерам швидко знайти потрібну модель, а для моделей – вигідно презентувати себе.

Візуальний компонент. Зважаючи на те, що дизайн є основою модельного бізнесу, WEB-сайт має бути елегантним, візуально привабливим і витриманим у загальному дизайні. Необхідно дотримуватись принципів, які стосуються сучасного UI/UX-дизайну, а саме чисті шрифти, логічна структура, якісні фотографії, анімації та інтерактивність.

Адаптивність. Сайт повинен бути адаптивним, оскільки значна частина користувачів відвідує його мобільних пристроїв. Для цього використовується адаптивна або респонсивна верстка [2].

Інтерактивність. Важливим елементом сайту є зручна форма для зворотного зв'язку або заявки на співпрацю, а також можливості бронювання моделей, запису на кастинг і підписки на новини. Інтерактивні елементи краще залучають користувачів.

Інтеграція з соціальними мережами. У модельному бізнесі Instagram, TikTok, Facebook та інші платформи, де публікується візуальний контент мають велике значення. WEB-сайт повинен тісно взаємодіяти з цими платформами. Наприклад, використання блоків із динамічними стрічками Instagram, кнопок «Поділитися», вбудованих відео YouTube та інших елементів значно збільшить кількість людей, які відвідують сайт.

Можливість адміністрування. Сайт має бути зручним для адміністрації. Це вказує на наявність системи управління контентом, або CMS. Наявність CMS дозволяє без навичок програмування додавати нові портфоліо моделей, змінювати текст, публікувати новини тощо.

Оптимізація SEO. Застосування оптимізації WEB-сайту, включаючи правильну структуру HTML, семантичну розмітку, швидкість завантаження сторінок, метадані, ключові слова та інше, має вирішальне значення для підвищення його видимості в пошукових системах.

Безпека та захист інформації. Оскільки агентства обробляють конфіденційні дані моделей і клієнтів, важливо забезпечити захист особистих даних за допомогою HTTPS, захищених форм і інтеграції систем безпеки, щоб запобігти витоку даних.

Крім вищезазначених вимог, важливо також враховувати доступність (accessibility) сайту для користувачів з особливими потребами. WEB-ресурси модельного бізнесу, як і будь-які сучасні платформи, мають бути інклюзивними,

щоб люди з вадами зору або іншими обмеженнями могли легко отримати доступ до контенту.

Для виконання всіх зазначених завдань використовують сучасні технології, які гарантують високоякісну роботу WEB-сайту.

В основі будь-якого WEB-додатка лежать Frontend (клієнтська частина) та Backend (серверна частина), кожна з яких має свої характеристики, технологічний стек і функцію в загальній архітектурі WEB-проєкту.

Візуальне відображення контенту та забезпечення взаємодії користувачів із сайтом належать до клієнтської частини. HTML5, CSS3 та JavaScript є основними технологіями для створення Frontend. HTML5 створює основу WEB-додатка, визначаючи структуру сторінки, компоненти контенту та те, як вони розташовуються. Розробники можуть створювати різноманітні мультимедійні елементи, такі як, аудіо, відео та інтерактивні елементи за допомогою HTML5 без додаткових плагінів.

CSS3 дозволяє змінювати дизайн WEB-сторінки, надаючи їй сучасний вигляд інтерфейсу. Робота зі шрифтами, кольорами, тінями, анімаціями та адаптивним дизайном є частиною цього. Крім того, використання CSS-препроцесорів, на зразок SASS або LESS, полегшує роботу зі стилями, надаючи змінні, вкладеність і функціональність, які неможливо досягти за допомогою традиційного CSS [3].

JavaScript є основною мовою програмування, яка використовується для додавання інтерактивності WEB-сайтам. Динамічні елементи, як-от анімації, спливаючі вікна, перевірка даних чи форм і обробку подій користувача можна реалізовувати через JS [4]. Крім того, оновлена версія JavaScript ES6+ пропонує нові функціональні можливості, такі як використання асинхронного коду, класів, модулів і стрілкових функцій.

Розвиток бібліотек і фреймворків значно полегшив процес розробки Frontend. Популярним інструментом є React, який був розроблений компанією Facebook (тепер Meta). React – це бібліотека для розробки користувацьких інтерфейсів, яка

використовує компонентний підхід. Оскільки зміни відображаються лише в потрібних елементах, React забезпечує високу продуктивність і швидкість оновлення інтерфейсу завдяки Virtual DOM.

Vue.js та Angular є іншими популярними інструментами для Frontend-розробки. Vue.js є простим і гнучким фреймворком, який ідеально підходить для проєктів малого та середнього розмірів. Angular, фреймворком, який був розроблений Google, пропонує широкий спектр функцій для створення ефективних програм [5].

Логіку додатка, обробку даних і взаємодію з базами даних виконує серверна частина. Node.js, середовище виконання JavaScript на сервері, є однією з основних технологій Backend [6]. Node.js полегшує розробку та підтримку WEB-додатків, дозволяючи використовувати JavaScript як єдину мову програмування як для клієнтської, так і для серверної частини. Розробники можуть значно скоротити час, необхідний для розробки функціоналу, підключаючи готові бібліотеки та модулі, завдяки пакету npm та модульній архітектурі.

RESTful API використовується для взаємодії між Frontend і Backend. REST надає можливість обміну даними між компонентами сервера та клієнта у форматах JSON або XML. Це гарантує універсальність, швидкість і простоту передачі даних у WEB-додатках. Хоча GraphQL, який надає більшу гнучкість у запитах до сервера, поступово замінює REST як стандарт для створення WEB-сервісів.

Реляційні системи, такі як MySQL і PostgreSQL, а також нереляційні бази даних, серед яких MongoDB, є найпоширенішими базами даних для Backend. Реляційні бази даних використовуються для зберігання структурованої інформації, тоді як MongoDB, база NoSQL, є гнучким варіантом для обробки великої кількості неструктурованих даних.

Без Git, системи контролю версій, сучасна WEB-розробка неможлива. Розробники можуть використовувати Git для відстежування змін у коді, роботи над

проєктами в команді та зберігання резервних копій на платформах, таких як GitHub чи GitLab.

Контейнеризація додатків з використанням Docker є також важливою частиною сучасної WEB-розробки [7]. Docker полегшує доставлення та масштабування, створюючи ізольовані середовища для розробки та розгортання WEB-додатків. WEB-додатки можуть автоматично оновлюватися, що, своєю чергою, мінімізує помилки при розгортанні нових функцій, коли вони використовуються разом із CI/CD-процесами.

Особлива увага приділяється оптимізації продуктивності WEB-додатків. Інструменти, на зразок Webpack і Vite, дозволяють оптимізувати завантаження модулів і ресурсів, що зменшує час завантаження сторінки. Крім того, продуктивність та SEO-оптимізації WEB-додатків аналізуються за допомогою Google Lighthouse [8].

Що стосується безпеки, то хостинг з високою швидкістю роботи, підтримкою SSL-сертифікатів, можливістю автоматичного резервного копіювання та масштабування рекомендується для сайтів модельного агентства [9]. Оптимальним варіантом є віртуальний хостинг або віртуальний приватний сервер (VPS). Необхідно застосовувати SSL-сертифікати, щоб гарантувати захищене передавання даних, особливо коли використовується форма зворотного зв’язку, анкетування чи робота з особистими даними. Також, впровадження компонентів безпеки, таких як CAPTCHA для захисту від ботів і двофакторна автентифікація для захисту облікових записів адміністрації сайту, є життєво важливим.

Було проведено прикладний аналіз офіційних WEB-сайтів провідних міжнародних модельних агентств, щоб отримати краще розуміння технологій і інструментів, які використовуються в сучасній WEB-розробці. З цієї причини було обрано три офіційні WEB-сайти міжнародних модельних агенцій, які відрізняються функціональним наповненням, візуальною привабливістю та високим рівнем

технічної реалізації. У процесі аналізу особлива увага була приділена таким аспектам:

- загальна структура сайту та логіка навігації;
- використані технології на фронтенді та бекенді;
- адаптивність інтерфейсу та мобільна оптимізація;
- швидкість завантаження сторінок;
- наявність інтеграцій із соціальними мережами;
- наявність форм для кастингу або онлайн-анкетування;
- мультимовна підтримка;
- естетика дизайну та зручність взаємодії з сайтом (юзабіліті).

Цей аналіз дозволить краще зрозуміти сучасні вимоги до структури, функціоналу та дизайну WEB-сайтів модельного агентства та визначити найкращі технологічні рішення, які доцільно враховувати під час розробки власного WEB-сайту.

Далі розглянемо результати перевірки вибраних WEB-сайтів.

IMG Models [10].

У WEB-сайта є чітка структура, яка включає головну сторінку, яка зображена на рис. 1.1, з розділами з кастингами та агентами, а також інтерактивний каталог, який може фільтрувати за різними параметрами, такими як місто, вік, стать тощо, що дозволяє швидко знайти потрібну інформацію.

Технології:

- Front-end: HTML5, CSS3, JavaScript;
- JavaScript-бібліотеки: React.js (SPA-компоненти);
- Back-end: Node.js / Express (ймовірно);
- CMS: власна або адаптована система керування контентом.

На сайті застосовано сучасні принципи responsive design, що гарантує комфортну навігацію з будь-якого пристрою. Він інтегрований із соціальними

мережами. Це означає, що він має активні посилання на Instagram, Facebook, YouTube. А у профілях моделей використовуються вбудовані відео та прямі переходи на соцмережі.

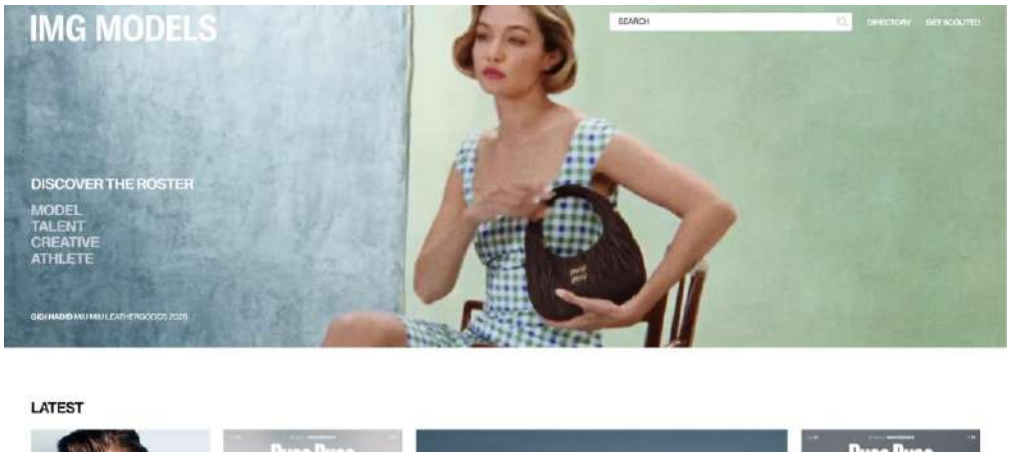


Рис. 1.1. Головна сторінка WEB-сайту IMG Models

Джерело: https://www.imgmodels.com

Для участі в кастингах є зручна онлайн-форма, яка дозволяє завантажити фотографії та відео, а також ввести контактні дані та параметри. Інтерфейс сайту англомовний, але навігація легко зрозуміла навіть для користувачів, які не володіють мовою. Сучасний і мінімалістичний дизайн зосереджується на фото- та відеоконтент, що створює елегантний і стриманий інтерфейс з логічною структурою взаємодії.

ONE MOTHER AGENCY [11].

Сайт має просту, але функціональну структуру, яка включає головну сторінку, представлення агентства, галерею моделей, новини, скриншот зображений на рис. 1.2, та контакти. Кожна модель має окрему сторінку з портфоліо. Водночас можливість приєднатися до агентства або подати заявку на участь у кастингу відсутня.

Технології:

– Front-end: HTML, CSS, JavaScript;
– CMS: WordPress (з кастомною темою);
– можливе використання плагінів типу Elementor або WPBakery.

Рис. 1.2. Сторінка з новинами на WEB-сайті ONE MOTHER AGENCY

Джерело: http://www.1motheragency.com

Інтерфейс сайту чудово працює на мобільних пристроях, але на деяких екранах можуть бути незначні зміщення елементів, пов'язані з окремими темами WordPress. Сайт інтегрований із соціальними мережами: є активні посилання на Instagram і Facebook, а також автоматично оновлювана стрічка Instagram на сторінці. Попри те, що WEB-сайт лише англійською мовою, його структура проста та легко зрозуміла. Дизайн лаконічний і візуально привабливий, з акцентом на великі фотографії, які займають центральне місце в інтерфейсі, а мінімалістичне оформлення полегшує візуальне сприйняття.

Ego Models Management [12].

Традиційна структура WEB-сайту складається з головної сторінки, бази моделей, кастингу, сторінки зворотного зв'язку, блогу та сторінки контактів. Вигляд сайту зображений на рис. 1.3. Окремо представлений розділ із переліком послуг, які пропонує агентство.

Технології:

– Front-end: HTML5, CSS3, JavaScript;

– Back-end: PHP (стандартна конфігурація для CMS);

– CMS: Joomla або WordPress.

Загалом інтерфейс сайту адаптивний, але іноді не оптимізований для мобільних пристроїв, оскільки деякі елементи не повністю масштабуються для невеликих екранів. Сайт має кнопки переходу на соціальні мережі, зокрема Instagram і Facebook, хоча стрічка новин із цих платформ не доступна безпосередньо на сайті.

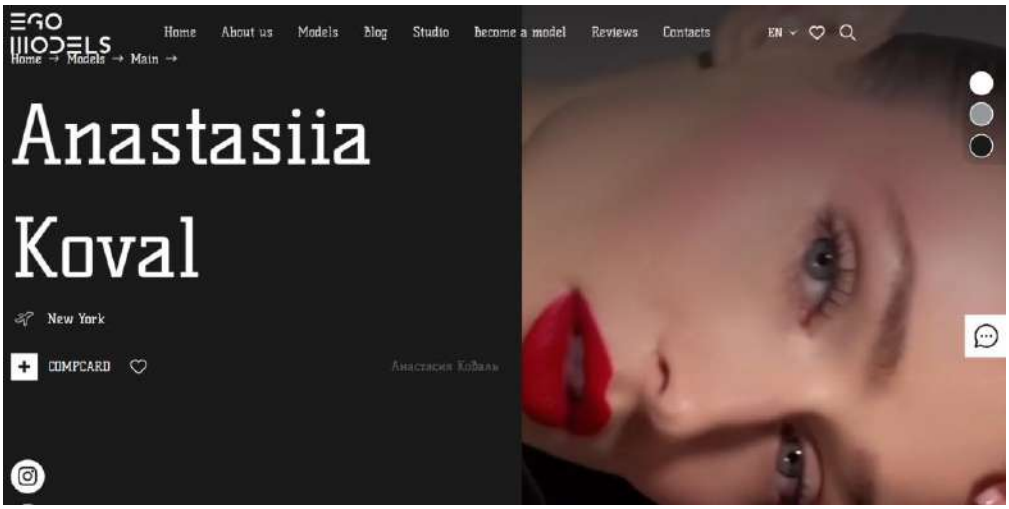


Рис. 1.3. Сторінка моделі на WEB-сайті Ego Models Management

Джерело: https://egomodels.com.ua

Участь у кастингу передбачає заповнення анкети, в якій можна завантажити фотографії. Підтримка двох мов, а саме української та англійської, є перевагою, оскільки це розширює авдиторію та залучає іноземних клієнтів. Деякі візуальні елементи необхідно оновити, щоб відповідати сучасним вимогам UI/UX, хоча дизайн досить простий.

Аналіз трьох сайтів показує, що вони можуть суттєво відрізнятися за рівнем технічної складності, дизайну та функціональності навіть у межах однієї галузі реалізації. Однак існують певні спільні риси:

– візуальна домінанта – великі зображення моделей;
– адаптивність для мобільних пристроїв;
– інтеграція з Instagram – обов'язкова умова;
– використання сучасних технологій, таких як React.js або WordPress, залежно від розміру проєкту.

Отже, ці сайти послужили основою для визначення основних функцій і технологічних рішень, необхідних для розробки власного WEB-сайту модельного агентства в межах кваліфікаційної роботи.

1.2 Обґрунтування вибору технічних засобів для розробки та впровадження WEB-сайту модельного бізнесу

Для створення функціонального, сучасного та ефективного WEB-сайту модельної агенції було обрано технології React, JS, CSS та HTML. Цей вибір ґрунтується на сучасних вимогах WEB-розробки, необхідності забезпечення гнучкості, продуктивності та привабливого користувацького інтерфейсу. Кожна з цих технологій відповідає поставленим завданням і має свої особливості та переваги.

Стандартною мовою розмітки для створення структури WEB-сторінок є HTML. Основна функція HTML полягає в тому, щоб створити структуру та елементи WEB-інтерфейсу, які відображатимуться на екрані користувача [13]. У рамках роботи HTML використовується для побудови «каркаса» WEB-сайту модельного агентства. Він гарантує, що основні елементи сторінки, такі як заголовки, текстові блоки, форми, зображення та навігаційні меню, будуть

розташовані правильним чином. Наприклад, основні блоки сайту були створені за допомогою елементів , <header> і <section>.

Новіша версія HTML5 відкриває нові можливості для інтеграції мультимедійного контенту та забезпечує семантичну розмітку сторінок [14]. Використання семантичних елементів, таких як <form> і <footer>, дозволяє покращити SEO для сайту, а також полегшує розробникам роботу з підтримкою коду.

CSS є важливою частиною дизайну, щоб створити WEB-сайт. Стилізацію сторінки можна змінювати завдяки CSS, тобто налаштувати шрифти, кольори, розміри елементів, розташування блоків і ефектів анімації.

Для WEB-сайту модельного агентства було обрано CSS3, оскільки він дозволяє створювати новітній і привабливий дизайн. Переваги використання CSS3 в кваліфікаційній роботі включають:

– гнучкість у стилізації елементів завдяки можливості застосування градієнтів, тіней, переходів і анімацій. Наприклад, CSS використовується для додавання ефектів наведення на кнопки чи фотографії моделей;

– зручність налаштування вигляду за допомогою можливостей CSS.

Завдяки CSS3 реалізовано мінімалістичний дизайн, який підкреслює візуальний контент WEB-сайту, що є важливим для демонстрації моделей та їх портфоліо.

JavaScript є основною мовою програмування, яка дозволяє зробити WEB-сторінки динамічними та інтерактивними. У кваліфікаційній роботі розробки WEB-сайту модельного агентства JS використовується для взаємодії з формами та оновлення вмісту сторінки без перезавантаження [15]. Це покращує доступність і швидкодію WEB-сайту. Крім того, перевірка даних у формі «Become a model» була реалізована завдяки JavaScript, що дозволяє автоматично перевіряти введені дані, наприклад, перевіряти правильність формату електронної пошти чи вимоги до

інших полів. А це, своєю чергою, підвищує зручність користування сайтом і гарантує правильність поданих заявок.

У сучасному стандарті ES6+ (ECMAScript 6 і вище) є більше можливостей для спрощення написання коду, зокрема за допомогою стрілкових функцій, модулів і синтаксичної зручності, на зразок const.

React було обрано для виконання частини функціональності WEB-сайту модельної агенції через його гнучкість, швидкодію та взаємодії з компонентами [16].

Переваги використання React у кваліфікаційній роботі включають:

– компонентний підхід. Уся структура WEB-сайту складається з окремих елементів. Це дозволяє організувати код ефективно та зробити його зрозумілим і зручним для підтримки. До прикладу, кожен елемент, як-от форма подачі заявки чи блок з інформацією про модель, можна використовувати як самостійний компонент, що зменшує дублювання коду та спрощує тестування;

– інтерактивність та динамічний контент. React дозволяє забезпечувати динамічне оновлення контенту. Наприклад, React дозволяє швидко відображати зміни без потреби перезавантаження сторінки, подаючи заявки від потенційних моделей або завантажуючи фотографії. Це полегшує користувачам взаємодію з сайтом;

– робота з API. React робить інтеграцію зовнішніх API для обміну даними легкою. Для сайту модельної агенції це є важливим для обробки заявок від потенційних моделей, завантаження їхнього портфоліо або отримання додаткової інформації, такої як статистика агентства.

Використання React у кваліфікаційній роботі спрощує розробку та підтримку функціональності сайту, зберігаючи високу продуктивність.

Реляційна база даних MySQL була обрана для зберігання та обробки даних на WEB-сайті модельного агентства, оскільки вона є однією з найбільш надійних і ефективних СУБД [17]. MySQL є чудовим інструментом для зберігання

структурованих даних, таких як інформація про модель, їхні портфоліо та заявки від потенційних моделей. Продуктивність, надійність і здатність масштабуватися в разі необхідності – це переваги цієї СУБД, зокрема в майбутньому, коли кількість користувачів і даних може збільшитися.

Крім того, MySQL пропонує високий рівень безпеки даних, що є неймовірно важливим для зберігання персональної інформації моделей. Реляційна структура бази даних дозволяє встановлювати зв'язки між різними таблицями та ефективно організовувати дані, що робить роботу з інформацією зручною та логічною. Окрім того, наявність великої кількості підтримки та інструментів для адміністрування, які надається MySQL, значно полегшує роботу з базою даних на етапі розробки та подальшої експлуатації сайту.

Таким чином, вибір технологій для створення та впровадження WEB-сайту модельного агентства був зумовлений бажанням забезпечити зручність користування, високу продуктивність, адаптивність інтерфейсу та можливість масштабування. У поєднанні з використанням HTML5, CSS3, JavaScript (ES6+), React і MySQL можна створити сучасний, функціональний ресурс, який може ефективно підтримувати бізнес-процеси агенції та задовольняти потреби цільової авдиторії.

1.3 Інструменти ефективного адміністрування та їх роль у просуванні WEB-сайту модельного бізнесу

Інструменти ефективного адміністрування WEB-сайту відіграють важливу роль у просуванні бізнесу в Інтернеті, оскільки вони гарантують зручність, продуктивність і надійність сайту, а також допомагають з оптимізацією SEO та залученням клієнтів. Google Analytics є одним із таких інструментів, який дозволяє відстежувати відвідуваність сайту, джерела трафіку та поведінку користувачів [18]. Це допомагає визначити, які сторінки популярні та як маркетингові кампанії

працюють. Інструмент Hotjar допомагає збирає детальнішу інформацію про поведінку відвідувачів, надаючи теплові карти та записи взаємодії з сайтом, що дозволяє виявити слабкі місця дизайну та покращити досвід користувачів.

Інструменти, такі як Google Search Console, є важливими для оптимізації SEO, оскільки вони дозволяють переглядати індексацію сайту та його видимість у пошукових системах [19]. Інші програми, на зразок SEMrush і Ahrefs, дозволяють провести ретельний аналіз конкурентів і розробити стратегію побудови зворотних посилань. Плагін Yoast SEO для WordPress автоматизує створення метатегів і пропонує поради щодо покращення контенту, що також важливо для оптимізації пошукової системи.

Багато бізнесів використовують вищезгаданий WordPress для управління контентом, оскільки ця CMS дозволяє легко створювати та редагувати WEB-сайти та інтегрувати різні плагіни для SEO та маркетингу. Інші складніші системи, такі як Contentful, дозволяють керувати великими та мультимедійними сайтами. Це важливо для компаній, які активно використовують відео та інші медіа [20].

Інструменти автоматизації маркетингу, такі як Mailchimp, дозволяють створювати та перевіряти електронні листи, а також ділити авдиторію на окремі групи для більш таргетованих кампаній. Потужним інструментом для управління взаєминами з клієнтами є платформа HubSpot, яка дозволяє автоматизувати маркетингові операції та відстежувати результати кампаній [21].

Що стосується безпеки та швидкості завантаження сторінок, інструменти, такі як Google PageSpeed Insights, аналізують час завантаження сторінок і пропонують ідеї, як їх покращити. Завдяки швидким серверам та додатковим шарам безпеки сервіс Cloudflare захищає сайт від кібератак і гарантує стабільний функціонал [22].

Для ефективного управління соціальними мережами агенції можуть використовувати інструменти, як Hootsuite та Buffer, які дозволяють планувати пости, публікувати їх та відстежувати їхні результати на різних платформах [23].

Такі інструменти збільшують трафік в соціальних мережах, що є важливим для підвищення видимості бізнесу.

Для підвищення коефіцієнта конверсії, що безпосередньо впливає на ефективність бізнесу, інструменти для тестування, як Optimizely та Google Optimize, допомагають тестувати різні варіанти сторінок [24]. Команда може працювати синхронно та ефективно за допомогою Trello, Asana чи Slack [25].

Усе це є важливими компонентами для ефективного адміністрування WEB-сайту та просування бізнесу. Цей набір інструментів дозволяє забезпечити високий рівень безпеки, швидкості, взаємодії з клієнтами та SEO-оптимізації, що дозволяє компаніям зберігати та покращувати свої позиції на ринку.

Висновки до розділу 1

У першому розділі було проаналізовано структуру, технологічні аспекти й функції сучасних WEB-сайтів модельного агентства. Встановлено, що хороший WEB-ресурс у сфері модельного бізнесу повинен мати наповнення інформацією, зручну навігацію, адаптивний інтерфейс і функції взаємодії з користувачами. Інтеграція з соціальними мережами, наявність зворотного зв’язку, мультимедійний контент, можливість адміністрування та дотримання правил безпеки є важливими. Також необхідно враховувати те, наскільки легко доступним є сайт для більшої кількості людей, особливо для людей із вадами зору або іншими обмеженнями.

На основі аналізу трьох авторитетних WEB-ресурсів було виявлено спільні риси, які відображають поточні тенденції в розробці сайтів модельного бізнесу. Всі сайти, незалежно від складності проєкту, зосереджуються на візуальному контенті, адаптивності, використанні інструментів SEO-оптимізації та підтримці зв’язків із соцмережами.

Дослідження підтвердило, що використання сучасних технологій фронтенд- і бекенд-розробки є доцільними. Також обґрунтовано вибір інструментів, які є

необхідними для створення високоякісних WEB-продуктів: системи контролю версій, хмарні середовища, інструменти оптимізації та моніторингу продуктивності, а також технології безпеки.

Загальні результати розділу сформували методологічну та технологічну основу для подальшої розробки та впровадження власного WEB-сайту модельного агентства. Тому цей WEB-сайт буде відповідати актуальним вимогам сучасного цифрового середовища.

РОЗДІЛ 2
ПРОЄКТУВАННЯ WEB-САЙТУ «МОДЕЛЬНОГО АГЕНТСТВА»

2.1 Визначення вимог до WEB-сайту модельного агентства

Першим кроком у розробці WEB-сайту є визначення вимог до його функціональності та нефункціональних характеристик. Це дозволяє створити чітке бачення очікуваних результатів і гарантувати, що система відповідатиме сучасним стандартам. Нижче наведено детальний опис усіх елементів, необхідних для WEB-сайту модельної агенції.

Основні функції, які WEB-сайт повинен виконувати, щоб досягти своїх цілей, визначені функціональними вимогами. Для модельної агенції вони включають:

– сторінка «Про нас» містить коротку інформацію про історію та переваги агенції;

– презентація моделей включає показ галереї моделей із фотографіями та основною інформацією (ім'я та параметри);

– форма «Become a model» містить поля для введення особистих даних, таких як ім’я, прізвище, номер телефону та посилання на соціальні мережі; поля для введення параметрів моделі (зріст, вага, розміри тощо); поле для віку кандидата; можливість завантаження фотографій; вбудована перевірка правильності введених даних; відправлення заповненої форми адміністратору для подальшої обробки;

– контактна сторінка містить контактні дані агентства, включаючи адресу офісу, номер телефону та email;

– реєстрація та авторизація дають моделям, фотографам і адміністраторам можливість створити обліковий запис;

– бази даних служать для структурованого зберігання заявок моделей з усіма введеними даними.

Нефункціональні вимоги забезпечують якість роботи WEB-сайту, його стабільність, продуктивність, зручність для користувачів і можливість розвитку. Вони встановлюють параметри, які доповнюють функціональні можливості та гарантують відповідність сучасним стандартам.

Щоб сайтом було зручно користуватись, він повинен мати високу продуктивність. Швидкість завантаження сторінок не повинна перевищувати 2-3 секунд для основних сторінок, для зменшення кількості відмов користувачів. Особливу увагу слід приділити формі «Become a Model». Після натискання кнопки «Submit» обробка даних повинна тривати не більше 1 секунди, щоб користувачі могли отримати швидкий зворотний зв'язок.

WEB-сайт відповідає основним стандартам безпеки даних. Усі дані, надіслані через форми, проходять перевірку та валідацію, а це гарантує, що вони правильні й захищені від помилок. Для захисту бази даних були введені механізми захисту від SQL-ін'єкцій [26]. Управління сайтом доступне лише авторизованим користувачам. Це, своєю чергою, запобігає несанкціонованому доступу.

Система має бути розроблена з урахуванням майбутніх розширень. Це означає, що WEB-сайт повинен підтримувати інтеграцію нових модулів або функціоналу. Наприклад, це може включати додавання нових частин, інтеграцію з API сторонніх сервісів або адаптацію до нових вимог бізнесу. Такий метод гарантує довгострокову гнучкість проєкту.

Основні WEB-браузери, такі як Google Chrome, Mozilla Firefox і Microsoft Edge повинні нормально працювати з сайтом. Це дозволить залучити велику кількість людей, незалежно від того, який браузер вони вибирають.

Позитивний досвід користувача залежить від простоти використання. Інтерфейс WEB-сайту повинен бути інтуїтивно зрозумілим, щоб користувачі могли швидко знайти потрібну інформацію без вивчення додаткових інструкцій або структуру. Усі компоненти навігації, включаючи посилання та кнопки, повинні бути розташовані таким чином, щоб були легко видимими та доступними [27]. Це

скоротить час, необхідний для пошуку потрібного функціоналу, а також підвищить якість взаємодії з сайтом.

Дизайн WEB-сайту має відповідати сучасним вимогам. Використання нейтральної колірної гами сприятиме створенню професійного іміджу агентства. Шрифти та розміри тексту повинні бути такими, щоб вони були легкі для читання. Дотримання мінімалізму, інтервали між текстовими блоками та правильний вибір кольорів для контрасту можуть підвищити привабливість сайту та спростити його використання [28].

Надійність WEB-сайту є важливою характеристикою, яка гарантує, що він не буде пошкоджений різними помилками. У ситуаціях, коли на сайт заходять сотні чи тисячі користувачів одночасно, він має бути адаптованим до надмірного навантаження.

Код WEB-сайту має бути структурованим і добре задокументованим. Це передбачає наявність документації для розробників і коментарів у коді, які пояснюють призначення ключових функцій і модулів. Такий метод спростить підтримку проєкту, його подальший розвиток і передачу іншій команді розробників.

Ці вимоги дають основу для створення якісного WEB-сайту, який задовольняє потреби користувачів, забезпечує надійність і зручність у використанні, а також відповідає сучасним стандартам розробки.

2.2 Архітектура WEB-сайту модельного агентства

Архітектура WEB-додатка визначає, як його частини взаємодіють з користувачами та між собою. На продуктивність, масштабованість, безпеку та простоту підтримки системи впливає правильний вибір архітектури. Сучасна WEB-розробка використовує наступні основні архітектури.

Рис. 2.1 показує архітектуру клієнта-сервера. Більшість WEB-додатків побудовані на цій архітектурі. Вона чітко розділяє клієнтську (WEB-браузер) і серверну (обробка даних і бізнес-логіка) частини.

Рис. 2.1. Клієнт-серверна архітектура

Джерело: https://dou.ua/forums/topic/44636

Основні характеристики: частина клієнта відповідає за відображення інтерфейсу користувача (HTML, CSS, JavaScript), а частина сервера обробляє запити клієнтів і надає доступ до бази даних.

До переваг належать підтримка, гнучкість у розробці та простота реалізації. Крім того, це підходить для програм, які мають чітко визначений функціонал.

До недоліків відноситься високе навантаження на сервер при великій кількості одночасних користувачів.

На рис. 2.2 показана модель Model-View-Controller. У WEB-розробці архітектура MVC є однією з найпоширеніших. Вона ділить програму на три основні частини: Модель (Model), Представлення (View) та Контролер (Controller). Це чітко відображує хто й за що відповідає, і полегшує розробку та підтримку WEB-додатка.

Модель керує даними додатка та взаємодію з базами даних, відповідає за бізнес-логіку та правила обробки даних. Цей компонент також дозволяє WEB-додаткам отримувати або змінювати дані.

Представлення дозволяє інтерфейсу користувача представляти дані з моделі. Він може виконувати це за допомогою HTML, CSS, JavaScript і використовує контролер для відображення динамічного контенту на основі даних з моделі.

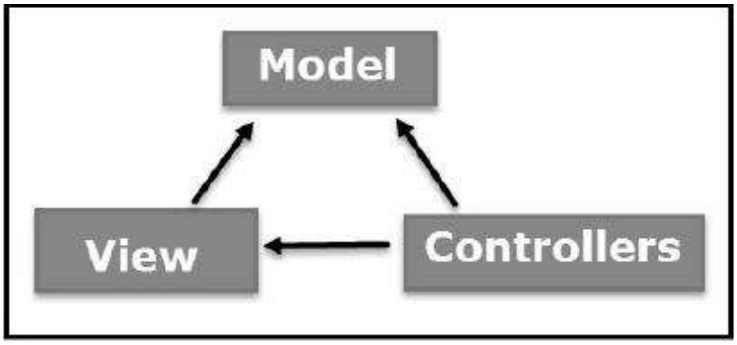


Рис. 2.2. Архітектура MVC

Джерело: https://medium.com/@tarunsinghrawat007/spring-mvc-6e1a9f8eb3b8

Контролер відповідає за обробку запитів користувача та координацію між моделлю та представленням. Він також отримує дані з моделі та передає їх до представлення для відображення. Крім того, обробляє події, такі як запити на подачу заявок або оновлення даних.

Численні переваги включають чітке розподілення відповідальності між компонентами; простоту в тестуванні, модифікації та масштабування; підвищення продуктивності команди під час розробки.

Недоліки: для новачків це може бути важка архітектура.

Архітектура мікросервісів показана на рис. 2.3. Цей метод ділить функції WEB-додатка на невеликі, автономні сервіси, кожен із яких виконує конкретну задачу.

Основні характеристики: кожен мікросервіс виконує певну роботу. Наприклад, він може обробляти заявки моделей чи керувати портфоліо.

До переваг належать легкість масштабування та підтримки, а також можливість використовувати різні мови програмування для різних сервісів.

Недолік – налаштування взаємодії між мікросервісами складне та вимагає складнішої інфраструктури.

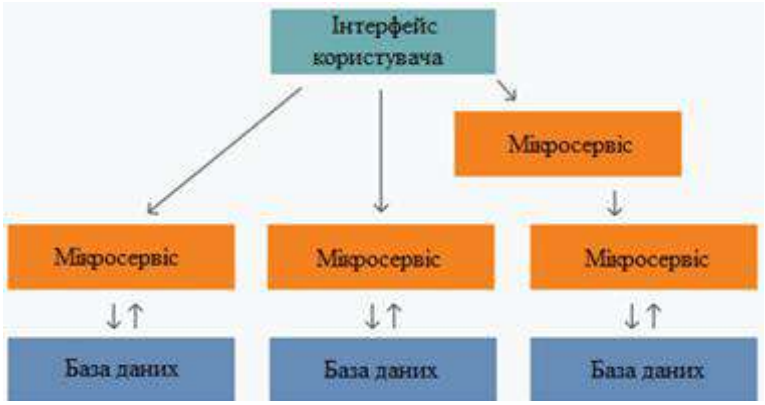


Рис. 2.3. Мікросервісна архітектура

(розроблено автором)

Зважаючи на всі характеристики кожного виду архітектури, клієнт-серверна архітектура буде найбільш відповідною для кваліфікаційної роботи. Оскільки її реалізація полегшує підтримку та розвиток сайту, додавання нових функцій та масштабування системи, коли кількість користувачів збільшується.

2.3 Технології для реалізації WEB-сайту модельного агентства

Для створення WEB-сайту модельної агенції було використано сучасні та надійні технології, які гарантують гнучкість, масштабованість і ефективність роботи ресурсу. Основними компонентами кваліфікаційної роботи є HTML5, CSS3, JavaScript (ES6+), бібліотека React і СУБД MySQL для збереження даних. Додатково використовується Node.js із фреймворком Express.js для реалізації серверної логіки та REST API, а також система контролю версій Git. Далі розглянемо як кожну з технологій було застосовано під час створення WEB-сайту.

HTML5 використовується для створення структури сторінок сайту. Логічна розмітка кожного компонента виконується за допомогою тегів <header>, <main>, <section>, <footer>. Шрифти, кольорова схема, анімація та адаптивність – усе це задається через CSS3 для стилізації інтерфейсу. У межах кваліфікаційної роботи

використано сучасні функції CSS, зокрема методи Flexbox і Grid, які дозволяють компонувати вміст сторінок відповідно до загальної структури сайту. Приклад такої розмітки та стилізації наведено на рис. 2.4.

```jsx
<div className="models-page">
    <Menu />
    <div className="hero">
        <h1 className="models-title">Models</h1>
        <div className="images-container">
            <div className="model-card">
                <Link to="/models/girls">
                    <img
                        src={`${process.env.PUBLIC_URL}/images/dar
                        alt="Girls"
                    />
                </Link>
                <Link to="/models/girls">Girls</Link>
            </div>
            <div className="model-card">
                <Link to="/models/boys">
                    <img
                        src={`${process.env.PUBLIC_URL}/images/art
                        alt="Boys"
                    />
                </Link>
                <Link to="/models/boys">Boys</Link>
            </div>
        </div>
    </div>
    <footer className="footer">
        <p>© 2024 Pollinate Models. All Rights Reserved</p>
    </footer>
```

```css
}

.model-card {
  text-align: center;
}

.model-card img {
  width: 300px;
  height: 400px;
  object-fit: cover;
  border-radius: 10px;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
  cursor: pointer;
  transition: transform 0.3s ease;
}

.model-card img:hover {
  transform: scale(1.05);
}

.model-card a {
  display: block;
  margin-top: 10px;
  font-size: 1.2rem;
  color: #333;
  text-decoration: none;
}
```

Рис. 2.4. Приклад розмітки сторінки сайту та CSS-оформлення

(розроблено автором)

JavaScript відповідає за інтерактивність сайту, включаючи перевірку форм, обробку подій і динамічне оновлення контенту без перезавантаження. В кваліфікаційній роботі використовуються актуальні можливості мови: стрілкові функції, модулі, async/await, деструктуризація тощо, завдяки ECMAScript 6+.

Структура форми «Become a model», скриншот якої зображено на рис. 2.5, є прикладом застосування цих можливостей.

React – основна технологія для побудови користувацького інтерфейсу. Кожен елемент сайту працює самостійно. Це забезпечує високу продуктивність завдяки використанню Virtual DOM, а також дозволяє повторно використовувати елементи та полегшує підтримку коду (рис. 2.6).

```
<div className="become-model">
    <Menu />
    <header className="become-header">
        <h1>Become a Model</h1>
    </header>
    <section className="become-content">
        <h2>Please, fill out the form</h2>
        <form onSubmit={handleSubmit} className="model-form">
            <div className="main-information">
                <label>Name</label>
                <input
                    type="text"
                    name="name"
                    value={formData.name}
                    onChange={handleChange}
                    required
                />
```

Рис. 2.5. Структура форми «Become a model»

(розроблено автором)

```
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about-us" element={<AboutUs />} />
        <Route path="/models" element={<Models />} />
        <Route path="/models/girls" element={<Girls />} />
        <Route path="/models/girls/:id" element={<GirlProfile />} />
        <Route path="/models/boys" element={<Boys />} />
        <Route path="/models/boys/:id" element={<BoyProfile />} />
        <Route path="/become-model" element={<BecomeModel />} />
        <Route path="/login/:role" element={<Login />} />
        <Route path="/contact-us" element={<ContactUs />} />
      </Routes>
    </Router>
```

Рис. 2.6. React-компоненти сайту

(розроблено автором)

Середовище виконання JavaScript, а саме Node.js разом із фреймворком Express.js, використовується на серверній частині. Це дозволяє створити REST API, через який Frontend (React) надсилає запити до Backend (Node.js). Сервер обробляє запити (приклад зображений на рис. 2.7), зберігає та читає дані з MySQL, а також надсилає відповіді у форматі JSON. Повний фрагмент серверного коду наведено в додатку М.

```
app.post("/api/models", upload.array("photos", 4), (req, res) => {
    const {···
    } = req.body;

    const numericFields = { age, height, bust, waist, hips, shoe };
    for (const [key, value] of Object.entries(numericFields)) {
        if (value < 0) {
            return res.status(400).json({ message: `${key} cannot be negat
        }
    }

    const folderName = `${name}_${surname}`;
    const folderPath = `\\uploads\\${folderName}`;

    const query = `
        INSERT INTO models (name, surname, age, city, phone, height, bust,
        VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
    `;

    db.query(
        query,
        [name, surname, age, city, phone, height, bust, waist, hips, shoe,
        (err, result) => {
            if (err) {
                console.error("Error inserting data:", err);
                res.status(500).json({ message: "Error saving data" });
            } else {
                res.status(200).json({ message: "Data saved successfully!"
```

Рис. 2.7. Обробка POST-запиту до API в Express.js

(розроблено автором)

MySQL використовується для зберігання структурованих даних, таких як анкети моделей, контактна інформація, заявки. Принцип нормалізації забезпечує ефективність та узгодженість баз даних. Особисті дані, параметри моделі та збережені шляхи до фотографій містяться в таблицях (рис. 2.8).

```
2 • CREATE TABLE models (
3       id INT AUTO_INCREMENT PRIMARY KEY,
4       name VARCHAR(255) NOT NULL,
5       surname VARCHAR(255) NOT NULL,
6       age INT NOT NULL,
7       city VARCHAR(255) NOT NULL,
8       phone VARCHAR(50) NOT NULL,
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

id	name	surname	age	city	phone	height	bust	waist	hips	shoe	eyes
1	Yuliia	Burei	18	Kryvyi Rih	576184395	170	70	50	70	37	green
2	Vlad	Tobias	19	Kryvyi Rih	576184395	180	70	50	70	37	green
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2.8. SQL-структура таблиці models у MySQL

(розроблено автором)

Інші інструменти:

– Git – використовується для керування версіями коду та збереження історії змін;

– Postman – застосовується для тестування роботи API, перевірки запитів і отримання відповідей від сервера;

– Multer – використовується для обробки завантаження фотографій моделей;

– Sequelize – це ORM-бібліотека, яка спрощує взаємодію з MySQL, дозволяючи працювати з базою даних через об’єкти замість SQL-запитів.

Описані технології створюють гнучке та надійне середовище для розробки функціонального WEB-сайту модельної агенції, який легко підтримувати та масштабувати.

2.4 Дизайн та користувацький інтерфейс WEB-сайту модельного агентства

Привабливий, легко зрозумілий і адаптований до потреб користувача інтерфейс є важливою частиною ефективного WEB-ресурсу. У процесі розробки WEB-сайту модельної агенції пріоритетом були візуальна естетика, зручність навігації, логічна структура сторінок і послідовний стиль оформлення.

Інтерфейс базується на компонентах React і використовує кастомізовані CSS-модулі для дизайну. Montserrat був обраний основним шрифтом, щоб підкреслити сучасний вигляд і читабельність контенту. Нейтральні кольори (білий, світло-сірий, графітний і чорний) з акцентом на інтерфейсних елементах складають дизайн сайту. Така гама кольорів створює стриманий і професійний стиль, характерний для fashion-індустрії. Заокруглені кути блоків і тіні додають глибини інтерфейсу. Кнопки оформлені у темному стилі й мають плавний ефект при наведенні, що надає

сайту динамічності. Вирівнювання блоків досягається за допомогою Flexbox, який забезпечує візуальну рівновагу сторінки.

На кожній сторінці використовується фіксований футер у чорному кольорі з білим текстом, який містить авторські права та навігаційні посилання. В hover-стані посилання стають помітнішими, а колір змінюється на акцентний.

Адаптивна верстка не була реалізована в межах поточної версії сайту, однак передбачена як наступний етап розвитку кваліфікаційної роботи. За допомогою технологій CSS, таких як flex, gap, auto margin, max-width, можна легко в майбутньому адаптувати інтерфейс до мобільних пристроїв і планшетів без значних структурних змін.

Крім візуального стилю, велику увагу приділено розміщенню контенту на сайті та його побудові. Схема навігації, яка зображена на рис. 2.9, та користувацький інтерфейс WEB-сайту модельної агенції розроблені таким чином, щоб вони були максимально інтуїтивними та зручними для кінцевого користувача.

Рис. 2.9. Схема навігацій WEB-сайту

(розроблено автором)

Коли користувач відвідує сайт, він автоматично потрапляє на головну сторінку, де відображається відео у повноекранному режимі. Це робить WEB-сайт

динамічним і сучасним. Назва модельної агенції розташована на фоні відео, привертаючи увагу відвідувачів. У лівому верхньому кутку сторінки знаходиться логотип агентства, який служить елементом ідентифікації бренду. Іконка меню, тобто основний навігаційний елемент сайту, розташована у правому верхньому кутку.

При натисканні на меню з'являється список доступних розділів, а саме головна сторінка, про нас, моделі, стати моделлю, вхід/реєстрація та контактна інформація. Ці розділи структуровані логічно й відповідають основним потребам відвідувачів.

Розділ «Про нас» містить інформацію про модельну агенцію, включаючи дату створення, засновників, команду професіоналів та основні принципи роботи. Фотографія колективу доповнює сторінку візуально, додаючи професійну та дружню атмосферу бренду.

Користувач може побачити дві основні категорії моделей: чоловічі та жіночі, коли переходить у розділ «Моделі». Ці категорії представлені великими зображеннями з інтерактивними кнопками. Натиснувши на зображення дівчини, ви отримаєте доступ до сторінки зі списком жіночих моделей, кожна з яких має мініатюрне фото й ім'я. Користувач переходить на окрему сторінку після вибору конкретної моделі. Там він бачить основну фотографію моделі, параметри (зріст, вага, обсяги тощо), а також галерею з чотирма додатковими фотографіями для більш детального ознайомлення. У категорії чоловічих моделей використовується ідентична структура та навігація. Такий підхід забезпечує єдину структуру та узгодженість у навігації.

На сторінці «Стати моделлю» є інтерактивна форма для заповнення, яка має на меті залучити нових людей до модельного агентства. Форма містить поля для введення імені та прізвища, контактних даних, параметрів (зріст, вага, обхвати), а також можливості завантажити фотографії та залишити додаткове повідомлення. Потенційні моделі з легкістю можуть подати заявку онлайн за допомогою цієї

функції. Крім того, інформація автоматично надходить до бази даних агенції для подальшої обробки.

Користувач має доступ до всіх способів зв'язку з агентством у розділі «Контактна інформація». Тут наведено посилання на популярні соціальні мережі, зокрема Facebook, Instagram та TikTok, щоб користувачі могли стежити за діяльністю агенції та переглядати його оновлення. Також для прямого зв’язку вказані номер телефону та електронна пошта. Крім того, на сторінці інтегрована карта Google Maps, яка показує розташування офісу агентства, що полегшує пошук для потенційних клієнтів і партнерів [29].

Додатково в структурі меню є окремий розділ «Вхід/Реєстрація», який дозволяє користувачам різних ролей, а саме адміністратор, модель, фотограф, авторизуватися. Після переходу на сторінку користувач може кнопку – вхід або реєстрацію. У формі реєстрації необхідно заповнити полями для імені, прізвища, номера телефону та вибору ролі. Після натискання на кнопку «Зареєструватись» система проводить перевірку в базі даних агенції. Якщо користувача з вказаними даними не знайдено серед наявних в базі моделей, фотографів або адміністраторів, на екрані з’являється відповідне повідомлення про помилку, що обмежує доступ до особистого кабінету.

Після авторизації користувач потрапляє до особистого кабінету, який має різні функції залежно від обраної ролі:

– адміністратор переглядає заповнені форми «Become a model», додати кандидата до бази моделей або відхилити заявку;

– фотограф створює оголошення про фотосесії та переглядати кандидатури на участь, а потім, відповідно, приймати або відхиляти їх;

– модель бачить список оголошень про фотосесії, на які вона може подати заявку, а також перевіряє чи прийнято її до участі у фотосесії.

Загальна структура WEB-сайту є чіткою, систематичною та спрямованою на те, щоб зробити його зручним для користувачів. Сучасний інтерфейс, зосереджений

на візуальному компоненті, забезпечує позитивний користувацький досвід і відповідає вимогам сучасного WEB-дизайну. Швидкий доступ до всіх необхідних розділів забезпечується грамотно побудованою навігацією.

Висновки до розділу 2

У цьому розділі здійснено комплексне проєктування WEB-сайту модельного агентства. Спочатку було визначено функціональні та нефункціональні вимоги до системи. Це дозволило сформувати чітке бачення майбутнього ресурсу, яке включатиме цілі, розширюваність, безпеку та особливості взаємодії з користувачами.

Аналіз архітектурних підходів підтвердив, що використання моделі клієнт-серверна, яка пропонує гнучкість, простоту підтримки й ефективність масштабування, є доцільним. Обрана архітектура гарантує стабільну взаємодію з базою даних і чітко розділяє логіку між клієнтським і серверним компонентами.

Під час проєктування розглянуто застосування технологій: HTML5, CSS3, JavaScript (ES6+), React, Node.js з Express.js, MySQL, а також допоміжні інструменти Sequelize, Multer, Postman і Git. На прикладі реалізації форми «Become a model», маршрутизації сторінок, API-запитів та обробки даних було продемонстровано практичне використання сучасних WEB-технологій для створення повноцінного функціонального сайту.

Особливу увагу приділено дизайну й інтерфейсу користувача. Привабливість ресурсу та приємний користувацький досвід досягаються за допомогою актуального шрифту, нейтральної колірної палітри, логічної навігації та зручної структури сторінок. Також передбачено створення особистих кабінетів для різних ролей, на зразок адміністратор, модель і фотограф.

Таким чином, було завершено проєктування WEB-сайту модельної агенції, який відповідатиме сучасним стандартам WEB-розробки з огляду на технічні,

функціональні та естетичні елементи. Цей етап дає міцну основу для подальшої роботи з сайтом та його впровадженням.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ WEB-РЕСУРСУ РОЗРОБКИ WEB-САЙТУ «МОДЕЛЬНОГО АГЕНТСТВА»

3.1 Реалізація бекенду WEB-сайту «Модельного агентства»

На серверній частині WEB-застосунку була впроваджена повноцінна логіка обробки запитів користувачів, автентифікації, авторизації, керування базою даних і взаємодії між різними типами користувачів – моделями, фотографами та адміністраторами.

Реєстрація та авторизація користувачів.

Як показано на рис. 3.1, код реєстрації користувача створює форму, яка приймає ім’я, прізвище, номер телефону, пароль і обрану роль, (модель, фотограф або адміністратор) (див. додаток А). Після заповнення форми дані перевіряються. Наприклад, ім’я та прізвище мають містити лише латинські літери; номер телефону має містити лише допустимі цифри та не більше 15 цифр; і пароль має відповідати вимогам складності, використовуючи великі та малі літери, цифри та спеціальні символи. У випадку успішної перевірки надсилається POST-запит до серверного API. Якщо реєстрація проходить успішно, користувач автоматично перенаправляється у відповідний до його ролі кабінет за допомогою збереженого токена авторизації.

Номер телефону та пароль, які також проходять базову валідацію, використовуються у формі входу. Після цього на сервер надсилається запит на перевірку облікових даних. Клієнт отримує токен з інформацією про користувача, включаючи його роль, якщо вхід успішний. Далі ця інформація використовується для автоматичного перенаправлення до відповідного розділу сайту. На рис. 3.1 показано компонент коду (див. також додаток Б).

```
pages > JS Register.js > Register
Register() {
            <label>Role</label>
            <select name="role" value={formData.role} onChange={hanc
                <option value="model">Model</option>
                <option value="photographer">Photographer</option>
                <option value="admin">Admin</option>
            </select>
            <button
                type="submit"
                className="create-account-button"
            >
                Sign up
            </button>
            {message && <p className="message">{message}</p>}
        </form>
    </div>
    <footer className="footer">
        <p>© 2024 Pollinate Models. All Rights Reserved</p>
    </footer>
</div>
```

```
src > pages > JS Login.js > ...
const Login = () => {
    const handleSubmit = async (e) => {
                switch (role) {
                    case 'admin':
                        navigate('/admin-cabinet');
                        break;
                    case 'photographer':
                        navigate('/photographer-cabinet');
                        break;
                    case 'model':
                    default:
                        navigate('/model-cabinet');
                        break;
                }
            } else {
                setErrorMessage(response.data.message || 'Incorrect data.');
            }
        } catch (err) {
            console.error(err);
            setErrorMessage('Login error, please try again.');
        }
```

Рис. 3.1. Фрагменти кодів з файлів Register.js та Login.js

(розроблено автором)

Усі маршрути системи розподілено залежно від ролі користувача. Тому моделі, до прикладу, не можуть створювати фотосесії або переглядати заявки інших користувачів.

Розглянемо реалізацію кожного кабінету по черзі.

Кабінет адміністратора, код якого зображено на рис. 3.2 (див. додаток В), дозволяє адміністратору бачити всі заявки від потенційних моделей, які хочуть приєднатись до агентства. Користувач, який увійшов як адміністратор, отримує свої дані через токен JWT, з якого витягується номер телефону і за ним підвантажується інформація. Ім’я та прізвище кандидатів, а також посилання на їхні профілі в Instagram доступні адміністратору у вигляді карток. При кліку на заявку відкривається профіль кандидата.

У профілі кандидата відображаються всі основні параметри: вік, місто, зріст, об’єми тіла, колір очей і волосся, номер телефону та посилання на соціальні мережі. Крім того, відображається головна фотографія, а також кілька додаткових, які зберігаються на сервері. В цьому ж профілі адміністратор має змогу прийняти або відхилити заявку кандидата, відправивши відповідні запити на сервер. Повернення до списку заявок відбувається після виконання дії. Код показано на рис. 3.3 (див. також додаток Г).

```
function AdminCabinet() {
    <div className="forms-header">
        <h2>
            Forms «Become a model» ({forms.length})
        </h2>
    </div>

    {forms.length === 0 ? (
        <div className="no-session-a">
            <p>There are no new applications at this time.</p>
        </div>
    ) : (
        <div className="forms-container">
            {forms.map((form) => (
                <div
                    key={form.id}
                    className="form-card"
                    onClick={() => handleCardClick(form.id)}
                >
                    <p><strong>{form.name} {form.surname}</strong></p>
                    <p>
                        <strong>Instagram:</strong>{" "}
                        {form.social_link ? (
                            <a
                                href={
                                    form.social_link.startsWith("http")
                                        ? form.social_link
                                        : `https://instagram.com/${form.social
                                }
                                target="_blank"
                                rel="noopener noreferrer"
                            >
                                {form.social_link}
```

Рис. 3.2. Фрагмент коду кабінету адміністратора

(розроблено автором)

```
pages > JS ViewModelForm.js > ViewModelForm > handleReject > headers
orm() {
    <div className="parameters">
        <h2>Parameters</h2>
        <p>Age: {formData.age}</p>
        <p>City: {formData.city}</p>
        <p>Phone number: {formData.phone}</p>
        <p>Height: {formData.height} cm</p>
        <p>Bust: {formData.bust} cm</p>
        <p>Waist: {formData.waist} cm</p>
        <p>Hips: {formData.hips} cm</p>
        <p>Shoe: {formData.shoe} eu</p>
        <p>Eyes: {formData.eyes}</p>
        <p>Hair: {formData.hair}</p>
        <p>
            <a href={formData.social_link} target="_blank" rel="noopener no
                <img
                    src="https://upload.wikimedia.org/wikipedia/commons/a/a
                    alt="Instagram"
                    className="instagram-icon-form"
                />
                {formData.social_link}
            </a>
        </p>
    </div>

    <div className="main-photo">
        {photos.length === 0 ? (
            <p>Photo not found</p>
        ) : (
            <img
                src={`http://localhost:5000${photos[0]}`}
                alt="main-photo"
                className="main-photo-img"
                />
```

```
src > pages > JS ViewModelForm.js > ...
7   function ViewModelForm() {
32      useEffect(() => {
35          const fetchPhotos = async () => {
36              try {
37                  const response = await axios.get(
38                      `http://localhost:5000/api/model-photos/${formData.name}/${fo
39                      {
40                          headers: {
41                              Authorization: `Bearer ${localStorage.getItem("token"
42                          }
43                      }
44                  );
45                  setPhotos(response.data);
46              } catch (error) {
47                  console.error("Could not upload photo:", error);
48              }
49          };
50
51          fetchPhotos();
52      }, [formData]);
53
54      const handleApprove = async () => {
55          try {
56              await axios.post(`http://localhost:5000/api/approve-model/${id}`,
57                  {},
58                  {
59                      headers: {
60                          Authorization: `Bearer ${localStorage.getItem("token")}`
61                      }
62                  }
63              );
64              navigate("/admin-cabinet");
65          } catch (error) {
66              console.error("Error while approving the application:", error);
```

Рис. 3.3. Фрагменти коду профілю кандидата

(розроблено автором)

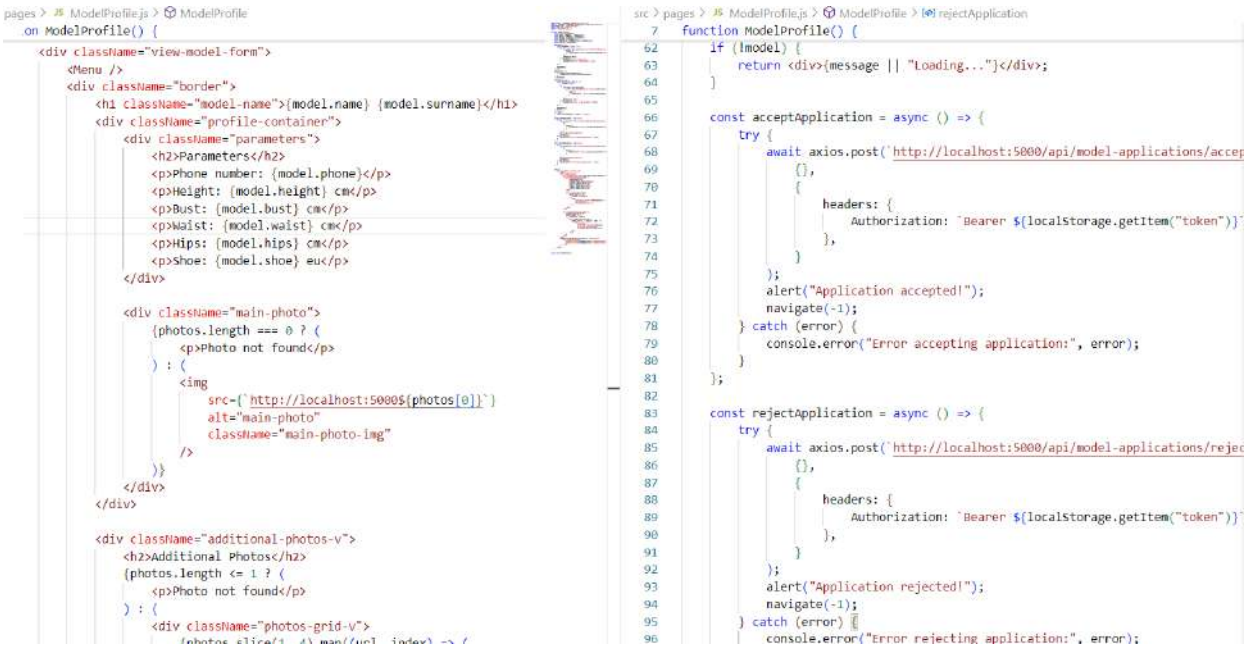
Кабінет фотографа, код якого зображено на рис. 3.4 (див. додаток Д), надає доступ фотографу до інформації про претендентів, які подали заявки на участь у

фотосесіях. Після вибору однієї з цих заявок відкривається профіль моделі, в якому відображаються її анкетні дані, включаючи ім’я, прізвище та номеру телефону. Також відображаються фізичні параметри, такі як зріст, об’єм грудей, талії, стегон та розмір взуття. Крім цього, фотограф бачить основну фотографію моделі й декілька додаткових. Використовуючи запити до API, весь вміст підтягується з серверної частини й автоматично оновлюється при завантаженні сторінки.

Рис. 3.4. Фрагменти коду кабінету фотографа

(розроблено автором)

Фотограф має кнопки для схвалення або відхилення заявка моделі, якщо вона ще не була розглянута. Користувач повертається до попередньої сторінки, коли натискає відповідну кнопку. Оскільки натискання відповідної кнопки призводить до надсилання POST-запит із відповідним статусом на сервер. Це показано на рис. 3.5 (див. також додаток Е).



```
pages > JS ModelProfile.js > ModelProfile
on ModelProfile() {
    <div className="view-model-form">
        <Menu />
        <div className="border">
            <h1 className="model-name">{model.name} {model.surname}</h1>
            <div className="profile-container">
                <div className="parameters">
                    <h2>Parameters</h2>
                    <p>Phone number: {model.phone}</p>
                    <p>Height: {model.height} cm</p>
                    <p>Bust: {model.bust} cm</p>
                    <p>Waist: {model.waist} cm</p>
                    <p>Hips: {model.hips} cm</p>
                    <p>Shoe: {model.shoe} eu</p>
                </div>

                <div className="main-photo">
                    {photos.length === 0 ? (
                        <p>Photo not found</p>
                    ) : (
                        <img
                            src={`http://localhost:5000${photos[0]}`}
                            alt="main-photo"
                            className="main-photo-img"
                        />
                    )}
                </div>
            </div>

            <div className="additional-photos-v">
                <h2>Additional Photos</h2>
                {photos.length <= 1 ? (
                    <p>Photo not found</p>
                ) : (
                    <div className="photos-grid-v">
```

```
src > pages > JS ModelProfile.js > ModelProfile > rejectApplication
7   function ModelProfile() {
62      if (!model) {
63          return <div>{message || "Loading..."}</div>;
64      }
65
66      const acceptApplication = async () => {
67          try {
68              await axios.post(`http://localhost:5000/api/model-applications/accep
69                  {},
70                  {
71                      headers: {
72                          Authorization: `Bearer ${localStorage.getItem("token")}`
73                      },
74                  }
75              );
76              alert("Application accepted!");
77              navigate(-1);
78          } catch (error) {
79              console.error("Error accepting application:", error);
80          }
81      };
82
83      const rejectApplication = async () => {
84          try {
85              await axios.post(`http://localhost:5000/api/model-applications/rejec
86                  {},
87                  {
88                      headers: {
89                          Authorization: `Bearer ${localStorage.getItem("token")}`
90                      },
91                  }
92              );
93              alert("Application rejected!");
94              navigate(-1);
95          } catch (error) {
96              console.error("Error rejecting application:", error);
```

Рис. 3.5. Фрагменти коду профілю моделі

(розроблено автором)

Кабінет моделі, код якого зображено на рис. 3.6 (див. додаток Ж), користувач бачить доступні фотосесії та має змогу подати заявку на участь у них. Система автоматично визначає модель за номером телефону, отриманим з JWT-токена після авторизації, та завантажує особисті дані моделі. Далі відбувається отримання актуального списку фотосесій, які організовані фотографами, і відображення їх у вигляді карток із зазначенням назви, опису, дати та місця проведення. Якщо модель уже надіслала заявку на певну фотосесію, вона бачить статус «Очікує рішення», «Схвалено» або «Відхилено». Якщо заявка ще не подана, кнопка для подачі є доступною.

Функціональність побудована таким чином, що всі заявки прив’язуються до конкретної моделі та фотосесії, а статуси динамічно змінюються для кожної з них. Після подачі заявки модель отримує повідомлення про успішне надсилання. Це дозволяє оновлювати дані без перезавантаження сторінки.

Рис. 3.6. Фрагменти коду кабінету моделі

(розроблено автором)

Форма «Become a model» є особливо важливим функціональним компонентом WEB-сайту модельного агентства, який надає можливість подачі заявок від потенційних моделей. Основним завданням форми є збір інформації про користувача, перевірка коректності введених даних та передача цієї інформації на сервер для обробки та збереження в базі даних.

Як показано на рис. 3.7 (див. також додаток К), файл BecomeModel.js містить код компонента, який використовує React Hooks для керування станом форми.

Для реалізації стану форми використовується React Hook useState. У першу чергу створюється об'єкт formData, який містить усі поля форми, включаючи початкові порожні значення. Під час введення даних користувачем цей об'єкт постійно оновлюється. Це досягається за допомогою функції handleChange, яка отримує подію (e), визначає поле за його іменем та зберігає нове значення у стані.

```
const BecomeModel = () => {
    const handleChange = (e) => {
        const numericFields = ['age', 'height', 'bust', 'waist', 'hips', 'shoe'];
        if (numericFields.includes(name) && value < 0) {
            setError(`${name} cannot be negative.`);
            return;
        }

        setFormData({ ...formData, [name]: value });
        setError('');
    };

    const handlePhotoChange = (e) => {
        const selectedFiles = Array.from(e.target.files);
        if (formData.photos.length + selectedFiles.length <= 4) {
            setFormData((prevData) => ({
                ...prevData,
                photos: [...prevData.photos, ...selectedFiles]
            }));
            // setError('');
        } else {
            setError('You can upload a maximum of 4 photos.');
        }
    };

    const handleRemovePhoto = (fileName) => {
        const updatedPhotos = formData.photos.filter((photo) => photo.name !== fileName);
        setFormData({ ...formData, photos: updatedPhotos });
    };

    const handleSubmit = async (e) => {
        e.preventDefault();

        if (formData.photos.length !== 4) {
            setError('You must upload exactly 4 photos before submitting.');
            return;
        }
```

Рис. 3.7. Фрагмент коду BecomeModel.js

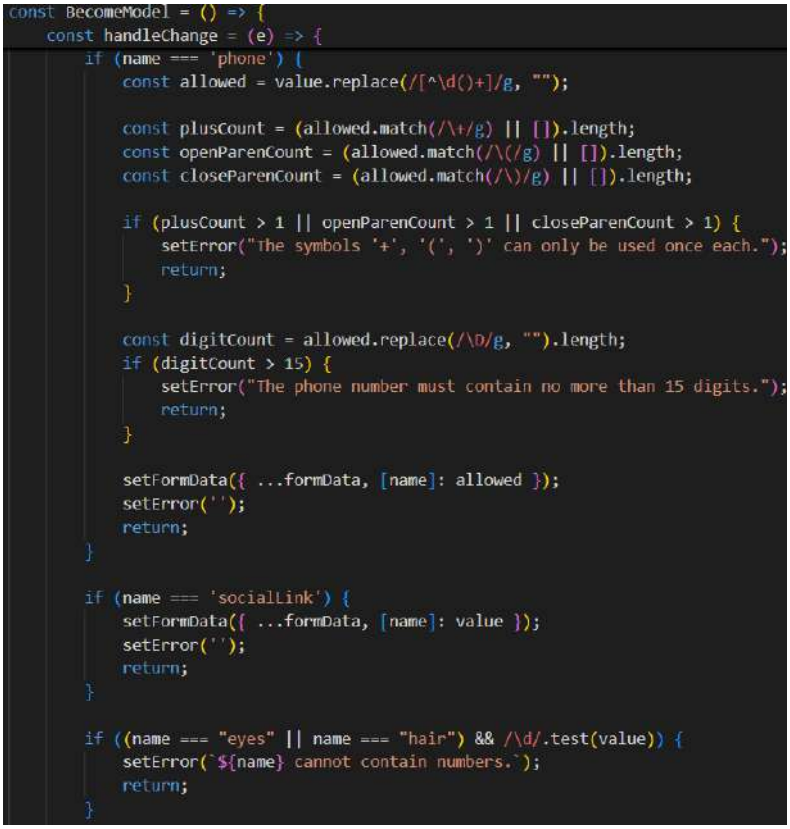
(розроблено автором)

Для забезпечення валідності даних було реалізовано декілька перевірок у функції handleChange. Основні обмеження включають наступне:

– перевірка номера телефону (рис. 3.8): допускаються лише цифри, та символи "+", "-", "()", які можна ввести по одному разу для коректного формату;

– перевірка для полів «Eyes» та «Hair» (рис. 3.8): заборонено вводити цифри в текстових значеннях;

– обмеження для числових значень (рис. 3.7): перевірка, щоб значення не було від'ємним щодо віку, зросту та обхвату тіла.



```
const BecomeModel = () => {
    const handleChange = (e) => {
        if (name === 'phone') {
            const allowed = value.replace(/[^\d()+]/g, "");

            const plusCount = (allowed.match(/\+/g) || []).length;
            const openParenCount = (allowed.match(/\(/g) || []).length;
            const closeParenCount = (allowed.match(/\)/g) || []).length;

            if (plusCount > 1 || openParenCount > 1 || closeParenCount > 1) {
                setError("The symbols '+', '(', ')' can only be used once each.");
                return;
            }

            const digitCount = allowed.replace(/\D/g, "").length;
            if (digitCount > 15) {
                setError("The phone number must contain no more than 15 digits.");
                return;
            }

            setFormData({ ...formData, [name]: allowed });
            setError('');
            return;
        }

        if (name === 'socialLink') {
            setFormData({ ...formData, [name]: value });
            setError('');
            return;
        }

        if ((name === "eyes" || name === "hair") && /\d/.test(value)) {
            setError(`${name} cannot contain numbers.`);
            return;
        }
```

Рис. 3.8. Фрагмент коду BecomeModel.js, який відповідає за перевірки

(розроблено автором)

Крім того, форма, яка представлена на рис. 3.7, дозволяє завантажувати до чотирьох фотографій одночасно. Це досягається шляхом використання поля типу file з атрибутом multiple. Функція handlePhotoChange обробляє завантажені файли та перевіряє кількість файлів. Таким чином, користувач може видаляти завантажені фотографії, натиснувши на значок хрестика поруч із назвою файлу. Функція handleRemovePhoto оновлює список файлів, які було завантажено.

Натискання кнопки «Send» викликає функцію handleSubmit, яка завершує перевірку даних перед надсиланням. Об'єкт FormData збирає дані форми та файли, що дозволяє надіслати інформацію правильно на сервер. fetch API використовується для надсилання запитів на сервер. Ця глобальна функція передає отримані дані методом POST на ендпоінт http://localhost:5000/api/models. У випадку

успішного надсилання форми користувачу надсилається відповідне повідомлення, а стан форми скидається до початкових значень.

3.2 Реалізація фронтенду WEB-сайту «Модельного агентства»

Уся візуальна (фронтенд) частина сайту поділена на окремі функціональні компоненти, які змінюються залежно від ролі користувача та поточного маршруту. Інтерфейс поділений на багаторівневу структуру компонентів, і кожна роль (модель, фотограф, адміністратор) має свій власний набір елементів. Ці елементи включають:

- компоненти загального доступу (Header, Footer, Home, About, Contact);
- компоненти для автентифікації (Login, Register);
- панель управління, яка є унікальною для кожної ролі (ModelCabinet, PhotographerCabinet, AdminCabinet);
- форми (BecomeModelForm, CreateSessionForm);
- компоненти для відображення списків і карток (SessionCard, FormCard, ModelCard, ApplicationStatusBadge тощо).

React Router використовується для реалізації багатосторінкового SPA. Кожна сторінка відображається залежно від маршруту, який динамічно обробляється відповідно до статусу авторизації користувача. На рис. 3.9 наведено файл App.js, який містить усі маршрути (див. додаток Л).

Фронтенд може постійно взаємодіє з бекендом через використання REST API. Це досягається за допомогою HTTP-запитів, які виконуються через вбудований метод fetch або бібліотеки axios. Тобто фронтенд надсилає запити до бекенду за допомогою fetch або бібліотеки axios, а результати відображаються через змінні стану (React useState) і ефекти (useEffect).

```
App.js > ...
function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about-us" element={<AboutUs />} />
        <Route path="/models" element={<Models />} />
        <Route path="/models/girls" element={<Girls />} />
        <Route path="/models/girls/:id" element={<GirlProfile />} />
        <Route path="/models/boys" element={<Boys />} />
        <Route path="/models/boys/:id" element={<BoyProfile />} />
        <Route path="/become-model" element={<BecomeModel />} />
        <Route path="/signinsignup" element={<Register />} />
        <Route path="/register" element={<Register />} />
        <Route path="/model-cabinet" element={<ModelCabinet />} />
        <Route path="/photographer-cabinet" element={<PhotographerCabinet />} />
        <Route path="/model-profile/:id" element={<ModelProfile />} />
        <Route path="/admin-cabinet" element={<AdminCabinet />} />
        <Route path="/admin-cabinet/view/:id" element={<ViewModelForm />} />
        <Route path="/login" element={<Login />} />
        <Route path="/contact-us" element={<ContactUs />} />
      </Routes>
    </Router>
  );
}
```

Рис. 3.9. Вміст файлу App.js

(розроблено автором)

Інтерфейс кабінету адміністратора (рис. 3.10) містить елементи для перегляду списку заявок, профілю моделі та ухвалення рішень.

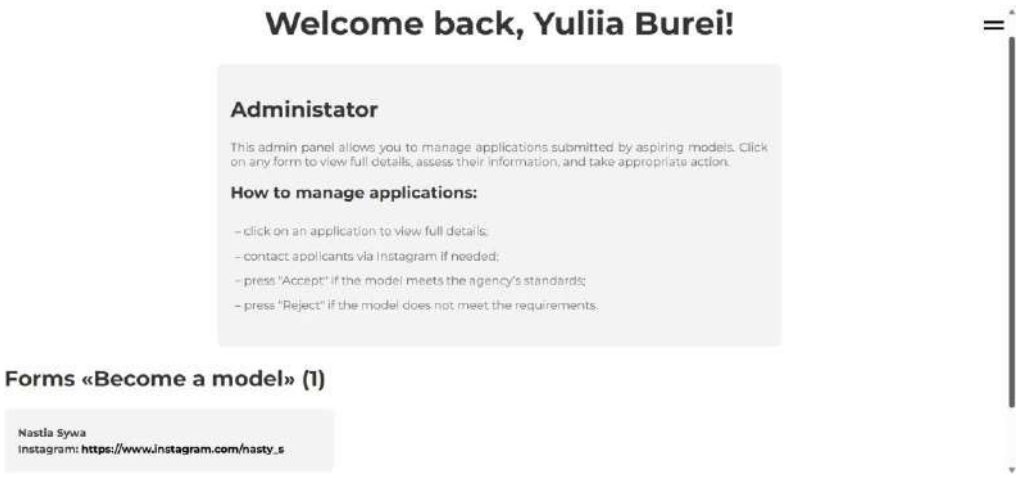


Рис. 3.10. Інтерфейс кабінету адміністратора

(розроблено автором)

У кабінеті фотографа, як зображено на рис 3.11, є список створених фотосесій, а також заявки на участь. Додано компоненти для обробки заявок і

створення нової фотосесії. Крім того, статус заявки підсвічено кольором для кращого сприйняття. Приклад проілюстровано на рис. 3.12.

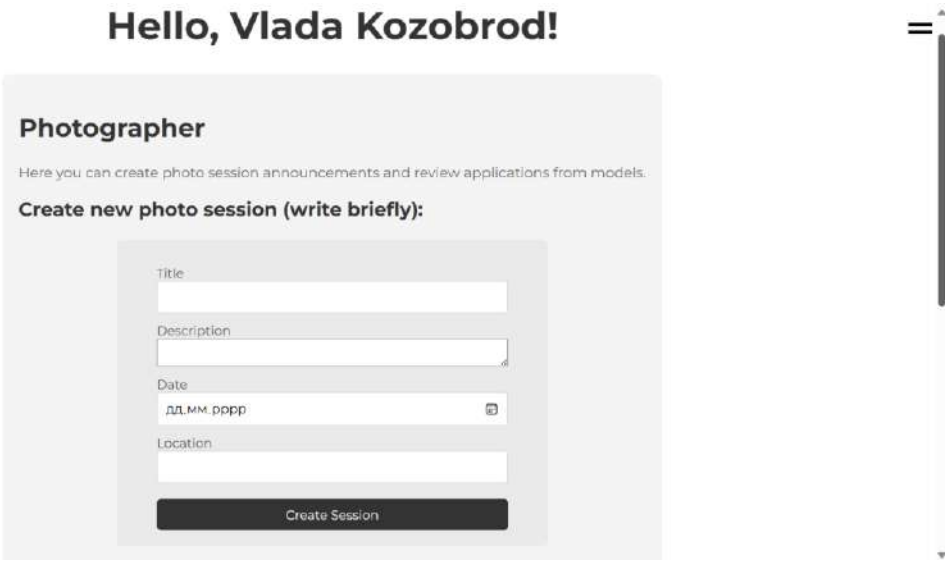


Рис. 3.11. Інтерфейс кабінету фотографа (частина 1)

(розроблено автором)

Рис. 3.12. Інтерфейс кабінету фотографа (частина 2)

(розроблено автором)

Кабінет моделі складається з трьох компонентів: список фотосесій, картки фотосесій, відображення статусу поданих заявок. Кожна заявка містить динамічну мітку статусу з кольоровим індикатором. Приклад проілюстровано на рис. 3.13.

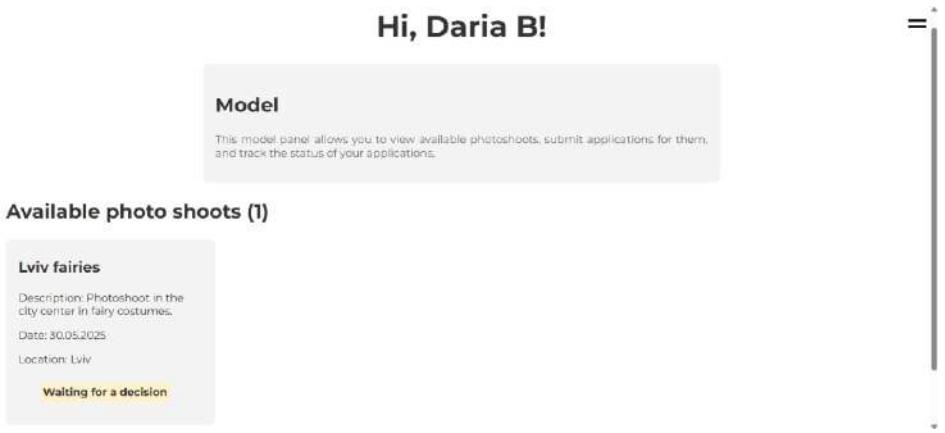


Рис. 3.13. Інтерфейс кабінету моделі

(розроблено автором)

Також, передбачено систему повідомлень про помилки та успішні дії. Ці повідомлення можуть включати, наприклад, підтвердження надсилання заявки, помилку валідації тощо. Щоб покращити користувацький досвід і швидко реагувати на проблеми, повідомлення виводяться на інтерфейсі в простій формі.

3.3 Побудова бази даних для модельного агентства та її інтеграція

База даних, зображена на рис. 3.14, є важливим компонентом функціонування WEB-сайту модельної агенції, оскільки вона дозволяє зберігати інформацію про користувачів, фотосесії, заявки, ролі тощо в одному місці.

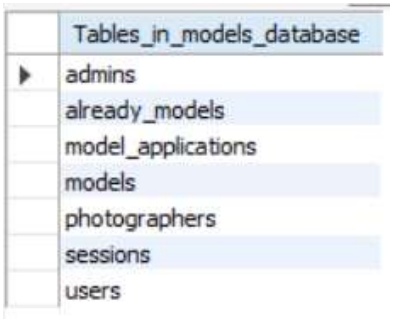


Рис. 3.14. База даних WEB-сайту модельного агентства

(розроблено автором)

Базу даних розроблено з урахуванням основних бізнес-процесів модельного агентства. Основні таблиці:

– users – зберігає дані про всіх зареєстрованих користувачів (моделей, фотографів, адміністраторів);

– sessions – інформація про фотосесії (назва, дата, місце, опис);

– model_applications – заявки моделей на участь у фотосесіях;

– models – зберігає анкети подані через форму «Become a Model».

Зовнішні ключі використовуються для реалізації зв’язків маж таблицями. Наприклад, model_applications має зовнішні ключі до users (модель) та sessions.

По таблицях admins, already_models, photographers виконується перевірка при реєстрації користувача, коли він обирає роль. Тобто користувач з таблиці admins не може зареєструватись під роллю фотографа, оскільки його немає в таблиці photographers. Запити SQL виконуються у відповідь на запити клієнтської частини для інтеграції API з базою даних.

API забезпечує безпеку, контроль доступу відповідно до ролі користувача, валідацію введених даних і обробку помилок під час взаємодії. Крім того, він дозволяє працювати з файлами, наприклад, завантажувати фотографії, і обробляти складні запити, наприклад, отримувати всі заявки, пов’язані з конкретним користувачем, водночас зберігаючи стабільність і масштабованість системи. Основні можливості API під час реалізації WEB-сайту модельної агенції:

– користувачі можуть створювати облікові записи, вказуючи свої ролі. Після успішного входу система видає токен доступу (JWT), який використовується для захисту подальших запитів. Приклад API-запиту зображений на рис. 3.15.

– Фотограф може створювати нові фотосесії. На рис. 3.15 показано приклад API-запиту.

– Модель може подати заявку на участь у фотосесії. Приклад API-запиту зображений на рис. 3.16.

– Адміністратор може схвалювати або відхиляти заявки на вступ до агентства. На рис. 3.16 показано приклад API-запиту.

Рис. 3.15. API-запит для реєстрації користувача та створення фотосесії

(розроблено автором)

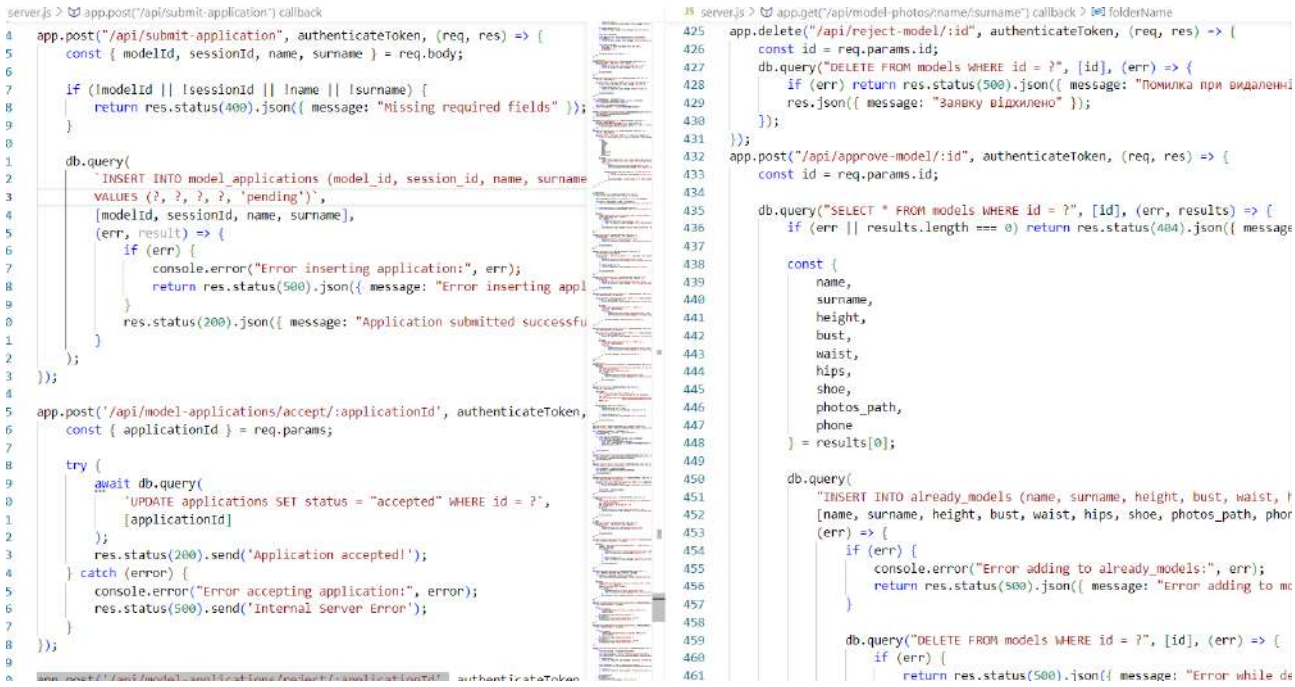


Рис. 3.16. API-запит для подання моделлю заявки на фотосесію та ухвалення рішення стосовно заявки адміністратором

(розроблено автором)

Таким чином, база даних гарантує стабільне зберігання та обробку даних, необхідних для функціонування WEB-сайту модельного агентства. Її інтеграція з RESTful API полегшує обмін даними між клієнтською частиною та сервером, реалізовує бізнес-логіку та забезпечує масштабованість і безпеку системи.

3.4 Тестування WEB-сайту «Модельного агентства»

Тестування має суттєве значення під час процесу розробки програмного забезпечення, оскільки воно дозволяє виявити та виправити помилки до впровадження продукту. Далі розглянуто процес тестування WEB-сайту модельної агенції, описано типи проведених тестів, наведено приклади тест-кейсів, а також проаналізовано результати перевірок.

Метою тестування полягала в тому, щоб переконатися, що основні функціональні модулі сайту працюють належним чином. Зокрема, перевірялася реєстрація та авторизація користувачів із різними ролями (модель, фотограф, адміністратор), а також робота особистих кабінетів відповідно до ролі, можливість створювати та переглядати фотосесії, надсилати заявки на участь у фотосесіях та подальша обробка цих заявок. Крім того, були проведені перевірки відповідності інтерфейсу дизайнерському рішенню та зручності користування сайтом.

У процесі перевірки було використано наступні типи тестування:

– функціональне тестування, яке перевіряє наскільки добре працюють ключові функції сайту;

– тестування інтерфейсу користувача (UI-тестування), яке перевіряє наскільки правильно відображаються елементи інтерфейсу та їх поведінка;

– кросбраузерне тестування – це тестування відображення сайту у різних браузерах, таких як Google Chrome, Mozilla Firefox і Microsoft Edge;

– тестування безпеки – це тестування наявності захисту авторизації, зокрема перевірка дій, які містять недійсний токен або взагалі без JWT-токена;

– тестування зручності користування (usability testing) – це процес визначення зручності навігації та логіки переходів між сторінками.

Для автоматизованого та ручного тестування використовувалися різні інструменти. Зокрема, Postman застосовувався для перевірки API-запитів, таких як реєстрація, авторизація, створення фотосесій тощо. Інструменти розробника в браузері, відомі як Browser DevTools, використовувалися для аналізу верстки та виявлення помилок у консолі. Перевірка стану бази даних здійснювалася за допомогою MySQL Workbench. Крім того, інтерфейс WEB-сайту тестувався вручну в різних браузерах.

Приклади конкретних тест-кейсів можна переглянути в таблиці 3.1.

Таблиця 3.1

Приклади тест-кейсів для тестування роботи WEB-сайту

Назва тесту	Вхідні дані	Очікуваний результат	Статус
Реєстрація моделі	Ім’я, прізвище, номер телефону, пароль, роль: модель	Користувач створений, перенаправлення до кабінету моделі	Успішно
Авторизація з неправильним паролем	Номер телефону існує, пароль - неправильний	Повідомлення про помилку	Успішно
Створення фотосесії (фотограф)	Назва, опис, дата, локація	Запис створено, відображається у списку	Успішно
Подання заявки (модель)	Вибір фотосесії, натиснення кнопки «Подати заявку»	Заявка збережена, відображається у кабінеті моделі	Успішно
Спроба доступу до кабінету адміністратора з роллю моделі	URL-адреса кабінету адміністратора	Повідомлення про помилку доступу	Успішно
Прийом заявки (фотограф)	Кнопка «Прийняти» біля заявки	Статус заявки змінено, кандидат бачить результат	Успішно
Перегляд кандидатів (фотограф)	Обрана фотосесія	Список заявок відображається	Успішно

Джерело: власний аналіз

Тестування також було проведено для обробки неправильних сценаріїв і потенційних помилок користувачів. Перевірено, що система не дозволяє відправити форму з незаповненими обов’язковими полями, виводячи відповідне повідомлення про необхідність заповнення всіх полів. Валідація успішно блокує помилки, пов’язані з введенням некоректних даних, таких як неправильний формат номера телефону, використання нелатинських символів або чисел у полях, де це не дозволено. Окрім цього, підтверджено, що спроба завантажити більше, ніж чотири фотографії у формі «Become a model» викликає відповідне повідомлення про помилку та блокує додавання зайвих файлів.

3.5 План розвитку проєкту WEB-сайту «Модельного агентства»

У процесі розробки WEB-сайту модельної агенції було виявлено багато способів подальшого покращення та розширення функціональності.

Розглянемо найголовніші зміни на майбутнє.

Адаптація під мобільні пристрої. Попри те, що WEB-сайт вже досить адаптивний, планується провести більш детальне тестування та оптимізацію для мобільних пристроїв. Це включатиме покращення інтерфейсу для більш зручного використання на смартфонах і планшетах.

Розширення функціоналу. В майбутньому планується додати нові функції, такі як інтерактивна галерея моделей, можливість фільтрації за різними критеріями, такими як вік і тип моделі, а також система відгуків від користувачів. Це забезпечить кращу взаємодію з відвідувачами сайту.

Покращення процедури валідації. Буде розглянуто впровадження складніших алгоритмів валідації даних, щоб зменшити ймовірність помилок при заповненні форм. До прикладу, використання регулярних виразів для перевірки форматів телефонів, електронних адрес та інших полів.

Аналіз даних та звітність. Використання аналітичної системи для збору даних про відвідувачів сайту, їх поведінку й вподобання. Це допоможе зрозуміти потреби користувачів і адаптувати контент відповідно до їхніх інтересів.

Постійне тестування та вдосконалення. Планується регулярне проведення тестування сайту, щоб виявити потенційні помилки та недоліки функціонування. Щоб гарантувати стабільність і якість продукту, це включатиме як автоматизовані, так і ручні тести.

Підвищення функціональності WEB-сайту, покращення користувацького досвіду та підтримка його конкурентоспроможності на ринку модельних агентств є центром цих майбутніх планів.

Висновки до розділу 3

У цьому розділі представлено процес реалізації WEB-сайту модельної агенції, з акцентом на важливих технічних рішеннях і функціональних модулях. Розподіл ролей і прав доступу забезпечує логічну структуру взаємодії користувачів із системою та підвищує організованість бізнес-процесів.

Для створення інтерфейсу було використано фреймворк React, який надав гнучкість у розробці компонентів і дозволив використовувати React Router для ефективної побудови маршрутизації. Клієнт і серверна частина взаємодіяли через REST API, а дані зберігалися в реляційній базі даних MySQL. Технологія fetch використовується для реалізації запитів на сервер, яка дозволяє передавати дані між фронтендом і бекендом у форматах JSON і FormData.

Особливу увагу приділено реалізації форми подачі заявки для потенційних моделей. Вона має можливість завантаження фотографій, перевірку формату та обов’язкових полів, валідацію даних на клієнтському рівні та механізми динамічного оновлення стану. Обмеження кількості зображень до чотирьох покращує користувацький досвід і оптимізує обсяг даних. Надсилання форми

супроводжується перевірками, що зменшує ймовірність помилок і дозволяє безпосередньо обробляти введену інформацію на сервері.

Тестування проєкту підтвердило, що основні функції сайту працюють стабільно. Перевірено логіку автентифікації, працездатність форм, доступність інтерфейсів для різних ролей і взаємодію з базою даних. Результати UI-тестів та кросбраузерної перевірки засвідчили коректне відображення елементів на різних платформах. Навігація працює без перезавантаження сторінок, що покращує продуктивність і комфорт користувачів.

У додаток, було окреслено перспективи розвитку системи, які включають, адаптацію до мобільних пристроїв, покращення алгоритмів перевірки даних і впровадження нових функцій. Ці кроки мають на меті розширити можливості WEB-сайту та покращити зручність використання.

Таким чином, виконання проєкту підтвердило ефективність обраних технічних рішень і показало, що ресурс готовий до повного використання. Сайт відповідає вимогам і має потенціал для розвитку.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було повністю реалізовано поставлену мету – створити сучасний, функціональний і безпечний WEB-ресурс, який відповідає вимогам модельного бізнесу, враховує потреби різних категорій користувачів і сприяє ефективній цифровій трансформації агентства.

Дослідження призвело до таких основних результатів:

- проаналізовано структуру та функціональність провідних сайтів міжнародних модельних агентств, визначено сучасні тенденції у сфері UI/UX-дизайну, інструментів просування, адміністрування й безпеки;
- встановлено доцільність використання React, Node.js, Express, MySQL та інших актуальних технологій для створення масштабованого та адаптивного сайту;
- спроєктовано архітектуру клієнт-серверного WEB-додатка з чітким розподілом ролей користувачів (модель, фотограф, адміністратор) та реалізовано відповідні особисті кабінети;
- реалізовано реєстрацію, авторизацію, динамічну обробку заявок, подачу та модерацію фотосесій, що дозволяє сайту виконувати не лише презентаційні, а й управлінські функції;
- забезпечено захист персональних даних і обмеження доступу до контенту через JWT-аутентифікацію, яка відповідає сучасним вимогам інформаційної безпеки;
- впроваджено базову SEO-оптимізацію, яка включає правильну структуру HTML, наявність метаданих, семантичну розмітку, швидкість завантаження сторінок та інтеграцію із соціальними мережами.

Кількісними результатами реалізації проєкту є створення повнофункціонального WEB-застосунку з понад десятьма сторінками, трьома типами кабінетів користувачів, функцією завантаження файлів, інтеграцією бази

даних і REST API. Якісні результати виражаються у стабільній роботі сайту, привабливому дизайні, зрозумілому інтерфейсі та зручності адміністрування.

Завдання, які були поставлені на початку дослідження, повністю виконані. Це підтверджується тим, що сайт відповідає технічним, функціональним і візуальним критеріям, описаним у розділах роботи.

Отже, кваліфікаційна робота не лише підтвердила теоретичну обґрунтованість теми, але й продемонструвала успішне практичне втілення результатів дослідження у вигляді готового до розгортання WEB-продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ceccotti F., et al. Traditional Agencies on Bridges: How Is Digital Transformation Changing Business Models // Journal of Management and Governance. 2024. C. 1–39.
2. Bootstrap Framework. "Responsive Frontend Development Framework." Bootstrap. URL: https://getbootstrap.com/ (дата звернення 24.03.2025).
3. Sass Documentation. "CSS Preprocessor." Sass. URL: https://sass-lang.com/documentation/ (дата звернення 24.03.2025).
4. Mozilla Developer Network (MDN). "Web Docs: HTML, CSS, JavaScript, and Web APIs." MDN Web Docs. URL: https://developer.mozilla.org/ (дата звернення 24.03.2025).
5. Vue.js Documentation. "Frontend Framework." Vue.js. https://vuejs.org/ (дата звернення 24.03.2025).
6. Node.js Documentation. "JavaScript Runtime Environment." Node.js. https://nodejs.org/uk (дата звернення 24.03.2025).
7. Miell, Ian, and Aidan Hobson Sayers. Docker in Practice. Manning Publications, 2016.
8. Lighthouse Documentation. "Web Performance Analysis Tool." Google Developers. URL: https://developer.chrome.com/docs/lighthouse/overview/ (дата звернення 30.03.2025).
9. Mănescu, V.-A., Bădescu, M., & Stănescu, D. (2021). Analysis of SSL certificates trends and extended validation SSL usage for e-commerce websites and Internet of Things. UPB Scientific Bulletin, Series C: Electrical Engineering, 83, 12.
10. IMG Models. Офіційний WEB-сайт. URL: https://imgmodels.com/ (дата звернення: 03.04.2025).
11. Mother Agency. Офіційний WEB-сайт. URL: https://www.1motheragency.com/ (дата звернення: 03.04.2025).

12. Ego Models. Офіційний WEB-сайт. URL: https://egomodels.com.ua/ (дата звернення: 03.04.2025).
13. Duckett, Jon. HTML and CSS: Design and Build Websites. John Wiley & Sons, 2011.
14. W3Schools. "HTML, CSS, JavaScript Tutorials and References." W3Schools. URL: https://www.w3schools.com/ (дата звернення 04.04.2025).
15. Freeman, Eric, and Elisabeth Robson. Head First JavaScript Programming: A Brain-Friendly Guide. O'Reilly Media, 2014.
16. Abramov, Dan. "React Official Documentation." React. URL: https://react.dev/ (дата звернення 04.04.2025).
17. MySQL Documentation. "MySQL Reference Manual." MySQL, URL: dev.mysql.com/doc/ (дата звернення 04.04.2025).
18. Deiss, R., Henneberry, R. Digital Marketing for Dummies. – Hoboken, NJ: John Wiley & Sons, 2020. C. 368. For Dummies Series.
19. Clarke, A. SEO 2023: Learn Search Engine Optimization with Smart Internet Marketing Strategies. – 7th ed. – United States: CreateSpace Independent Publishing Platform, 2023. C. 234.
20. Halvorson, K., Rach, M. Content Strategy for the Web. – 2nd ed. – Berkeley, CA: New Riders, 2012. C. 216.
21. HubSpot vs. Mailchimp: Which Should You Use?. URL: https://zapier.com/blog/hubspot-vs-mailchimp/ (дата звернення: 08.04.2025).
22. Cloudflare: The Invisible Shield Protecting 20% of the Web. URL: https://technologymagazine.com/articles/cloudflare-the-invisible-shield-protecting-20-of-the-web?utm_source=chatgpt.com (дата звернення: 08.04.2025).
23. Hootsuite vs. Buffer: Which Social Media Tool Should You Use?. URL: https://zapier.com/blog/hootsuite-vs-buffer/?utm_source=chatgpt.com (дата звернення: 08.04.2025).

24. Optimizely vs. Google Optimize. URL: https://www.optimizely.com/vs-google-optimize/ (дата звернення: 08.04.2025).
25. Єрохіна Д. О., Козинець Я. О. Інструменти цифрових комунікацій для управління командами. Рекомендовано до видання вченою радою Сумського державного університету (протокол № 5 від 12 грудня 2024 року). Суми: Сумський державний університет, 2024. С. 198.
26. Buja G., et al. Detection Model for SQL Injection Attack: An Approach for Preventing a Web Application from the SQL Injection Attack // 2014 IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE). – IEEE, 2014.
27. Krug S. Don't Make Me Think: A Common Sense Approach to Web Usability. – New Riders, 2000. С. 206.
28. Miller B. D. Principles of Web Design. – New York: Simon and Schuster, 2022. С. 192.
29. Google Developers. Integration with Websites // Google Maps API Documentation. URL: https://developers.google.com/maps/documentation (дата звернення: 16.04.2025).

ЗГОДА здобувачки вищої освіти

Державного університету економіки і технологій про перевірку кваліфікаційної роботи на прояви академічного плагіату

та розміщення в Репозитарії Університету

Я, Бурей Юлія Степанівна, підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.
Засвідчую, що кваліфікаційна бакалаврська (магістерська) робота «Розробка та впровадження WEB-сайту модельного агентства» виконана самостійно та не містить академічного плагіату. Я не надавала і не одержувала недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Державного університету економіки і технологій ознайомлений(а). Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення норм академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

Також я поінформований(на), що відповідно до «Положення про Репозитарій (електронну базу даних) Державного університету економіки і технологій» зазначена робота буде розміщена в Електронному архіві Університету (Репозитарії ДУЕТ). З умовами такого розміщення ознайомлений(на).

Бурей Ю.С.

10.06.2025