ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: **«РОЗРОБКА ПЛАТФОРМИ ДЛЯ СТВОРЕННЯ АДАПТИВНИХ САЙТІВ НА БАЗІ ФРЕЙМВОРКУ LARAVEL»**

Виконав: студент 4 курсу, групи ПД-41 спеціальності
121 Інженерії програмного забезпечення
(шифр і назва спеціальності)

Яновський Д.А.
(прізвище та ініціали)
Керівник Негоденко О.В.
(прізвище та ініціали)
Рецензент_____
(прізвище та ініціали)
Нормоконтроль_____
(прізвище та ініціали)

КИЇВ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії Програмного Забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність -121 Інженерія Програмного Забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії Програмного Забезпечення

_____Негоденко О.В.

«____»_____2021 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Яновському Дмитру Андрійовичу

(прізвище, ім‘я, по батькові)

1.Тема роботи: «Розробка платформи для створення адаптивних сайтів на базі фреймворку Laravel»

Керівник роботи Негоденко Олена Василівна, к.т.н. доцент

затверджені наказом вищого навчального закладу від — 12.03 2021 року №65.

2. Строк подання студентом роботи 01.06.2021.

3. Вхідні дані до роботи:

3.1 Науково – технічна документація для розробки застосунків на Laravel

3.2 Розробка адаптивних дизайнів

3.3 Алгоритм прискорення завантаження сторінок

3.4 Тестування UI/UX користувача

3.5 Технічний функціонал Laravel

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

4.1 Аналіз існуючих рішень для створення сайтів.

4.2 Дослідження технічних документацій для створення застосунку на базі фреймворку Laravel.

4.3 Дослідження документації з SEO оптимізації.

4.3 Розробка функціоналу платформи.

4.4 Створення та оптимізація продукту.

5. Перелік графічного матеріалу :

5.1 Титульний слайд

5.2 Об’єкт, мета, предмет дослідження

5.3 Порівняльна характеристика аналогів.

5.4 Опис Laravel

5.5 MySQL

5.6 Макети платформи

5.7 Висновки

6. Дата видачі завдання 19.04.2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково – технічної літератури	19.04.2021 – 22.04.2021	
2	Вимоги для розробки SEO оптимізованого адаптивного сайту	24.04.2021 – 26.04.2021	
3	Аналіз існуючих методів швидко створити сайт	28.04.2021 – 29.04.2021	
4	Дослідження програмних та аналітичних засобів	30.04.2021 – 05.05.2021	
5	Моделювання об’єкту проектування	06.05.2021 – 22.05.2021	
6	Вступ, висновки, реферат	22.05.2021 – 26.05.2021	
7	Розробка презентації платформи	27.05.2021	
8	Здача роботи	01.06.2021	

Студент Яновський Дмитро Андрійович

Керівник роботи Негоденко Олена Василівна

РЕФЕРАТ

РОЗРОБКА ПЛАТФОРМИ ДЛЯ СТВОРЕННЯ АДАПТИВНИХ САЙТІВ НА БАЗІ ФРЕЙМВОРКУ LARAVEL

Об'єкт дослідження – створення та розгортання сайтів з дотриманням усіх вимог SEO.

Предмет дослідження – платформа для створення адаптивних сайтів.

Мета роботи – розробка платформи з можливістю швидкого створення бази для сайту, а також для подальшого маштабування без зайвих фінансових витрат.

Методи дослідження – методи розробки програмного забезпечення, обробка та аналіз інформації, дослідження інформації, методи верифікації та валідації.

Основою розробленої платформи, є фреймворк Laravel, оскільки він зручний для використання, має гарну швидкість обробки даних та великий обсяг документації, яка постійно оновлюється і допомагає при розробці. Це допоможе швидко створити сайт для себе, чи клієнта, отримавши гарний код та головне без використання зайвих скриптів, які сповільнюють завантаження сторінки і, як наслідок, негативно впливають на SEO рейтинг та користувацький відгук.

У даній роботі виконаний аналіз конкурентів на ринку. Визначені їх переваги та недоліки. У результаті враховані основні потреби споживача.

Фреймворк Laravel дозволяє виконати всі вище описані вимоги для даної платформи. Розроблено алгоритм передачі даних та інтуїтивність інтерфейсу для зручності користувача. Проект відрізняється у використанні якістю вихідного коду та швидкістю завантаження сторінки.

Сфера використання згадуваної платформи у приватному доступі (без застосування мережі Інтернет). Також, сайт забезпечує підвищення ефективності розробки, оскільки звільняє розробника від великої частини монотонної роботи за рахунок структури його наповнення.

ЗМІСТ

УМОВНІ ПОЗНАЧЕННЯ

ПК – персональний комп’ютер

БД – база даних

СУБД – Система Управління Базами Даних

MVC - (Model – View – Controller)

SEO - Search Engine Optimization

SCSS – Sassy CSS

ООП – Об’єктно-орієнтоване програмування

ВСТУП

Актуальність теми полягає у тому, щоб розробник міг швидко створити сайт витративши на це незначну кількість часу.

На сьогоднішній день сайтами користується майже кожен і попит на послугу створення та підтримку не зникає, а лише збільшується. Оскільки не кожна компанія може дозволити собі розробку власних додатків під всі платформи, та й не завжди це є доцільно, то кожна поважаюча себе організація має свій сайт і в користувача вже не викликає обурення, коли він адаптований під різні розміри екранів та його можна переглянути на будь-якому зручному девайсі з доступом до мережі Інтернет.

Теперішні інструменти для створення сайтів дозволяють IT-спеціалістам швидко реалізовувати нові ідеї в життя, а також зручно керувати даними при необхідності. Кожен має свої переваги та недоліки, вони постійно вдосконалюються та спрощують процес створення сайтів. Ще десять років тому це було складним завданням для якого потрібно знати немалу кількість інформації та вміти нею скористуватись. Сьогодні ж на ринку вже зустрічаються конструктори, де кожен бажаючий може створити власний сайт обравши шаблон і в зручному редакторі налаштувати дизайн на власний смак. Приходить на думку, що актуальність програмістів, на ринку створення сайтів, падає і з часом взагалі зникне. Але не все так просто. Як і в спорті, так і в Інтернеті на ринку сайтів існує конкуренція – кожен намагається стати кращим за інших. Для початку, необхідно відповідати ряду вимог, а вже потім змагатися з іншими. Та чи раціонально буде використовувати конструктори та шаблони? Моя відповідь: ні! Продукти в результаті не мають актуальних технічних характеристик, щоб задовольняти примхи вибагливого користувача та його не менш вибагливого помічника: Google.

Тому було вирішено створити платформу для швидкого створення адаптивних сайтів з дотриманням усіх правил SEO оптимізації, та основними необхідними для її підтримки функціями.

Об'єкт дослідження – можливість швидко створити та розгорнути сайт з дотриманням усіх вимог SEO.

Предмет дослідження – платформа для створення адаптивних сайтів.

Мета роботи – методи розробки програмного забезпечення, обробка та аналіз інформації, дослідження інформації, методи наукового моделювання.

Основні питання дослідження :

1. Проаналізувати необхідні функції на більшості сайтів.
2. Проаналізувати функції, які доступні в аналогах та їх особливості.
3. Проаналізувати інструменти, які будуть використовуватись в процесі реалізації платформи.
4. Вивчити можливість використання фреймворку Laravel для створення платформи з можливістю подальшого редагування.
5. Дослідити використання моделей, які будуть зберігати дані в базі даних.
6. Встановити попит функцій, які має виконувати створена платформа.
7. Опрацювати реалізацію та тестування платформи для створення адаптивних сайтів з дотриманням вимог SEO.

Методи дослідження – методи розробки програмного забезпечення, обробка та аналіз інформації, дослідження інформації, методи наукового моделювання.

Досліджено, що зі всіма поставленими задачами буде доцільно використовувати PHP фреймворк Laravel та CSS.

Встановлено, що зручний графічний інтерфейс та висока швидкість опрацювання даних буде сприятливим чинником для цільової аудиторії.

Результатом виконаної роботи є платформа, яка орієнтована на швидке створення адаптивних сайтів з високою швидкістю роботи кінцевого проекту. Задача полягає саме у створенні оптимізованої бази з чистим кодом, з можливістю в подальшому зручно редагувати та отримати продукт, який буде мати конкурентоспроможний SpeedRate сторінки та задовільнить функціональні та нефункціональні вимоги як замовника і користувача, так і ряду сервісів Google.

Результати дослідження бакалаврської роботи апробовані:

1. На всеукраїнській науково-технічній конференції: «Застосування програмного забезпечення в інфокомунікаційних технологіях»
2. На всеукраїнскій науково-технічній конференції «Сучасні інфокомунікаційні технології»

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика предметної області

Кожного дня в звичайному житті нам доводиться зустрічатись з великою кількістю питань на які десь потрібно шукати відповіді. Раніше це були лише близькі, знайомі та інформація зображена на папері або іншому предметі. Тобто якість та кількість інформації була дуже обмеженою та в порівнянні з сучасністю - не завжди коректною.

Сьогодні ж ми маємо доступ до межі Інтернет і нам стають доступні безмежні потоки інформації стосовно будь-якої повсякденної теми та відповіді на безліч питань. Звісно хтось повинен створити платформу з якої інші зможуть добувати корисну інформацію та хтось має публікувати свою думку, факти стосовно обраної теми.

До сьогодні сайти розділились за умовними типами: лендінги, магазини, блоги, форуми, соціальні мережі, тощо. А так сталося тому, що вони мають абсолютно різну структуру, наповнення та й сам підхід до створення. Окрім цього - це логічно та й справді зручно, адже нам не потрібно йти в бібліотеку, ми можемо відвідати тематичний блог і навіть декілька, крім цього опублікована книга завжди буде в наявності та якщо хтось інший і буде її читати - вистачить на всіх. В інше місто також непотрібно їхати, щоб обрати меблі додому або колір плитки, щоб підходив під шпалери. Сьогодні Інтернет вирішує велику кількість задач та спрощує життя всім, навіть тим, хто ним не користується.

Тепер повернемося до створення тієї платформи та її власника. Звісно він має отримувати прибуток з неї. Існує декілька легальних джерел доходу власників сайтів: це можуть бути продажі, реклама, підписка на функціонал. Банально реклама може покривати витрати на обслуговування сайту, не говорячи вже про інтернет-магазини чи, наприклад, соціальні мережі, які більше розраховані на обслуговування користувача та можливу винагороду за послуги. Крім цього,

компанії створюють сайти для того, щоб розповісти про себе іншим та показати свої переваги. Звісно куди ж без конкуренції в Інтернеті, якщо можна створити сайт на улюблену тему та отримувати з нього гідний дохід. Що може бути краще з прибутку, який отримуєш зі справи, яка подобається? Ніякі легкі або важкі гроші не зрівняються з тими, які чесно отримуєш, можливо, навіть за своє хобі.

Настав час завести мову про людину, яка буде створювати сайт. Сьогоднішні реалії вимушують нас повірити в те, що сайт може створити абсолютно кожен. Раніше це було можливим з платними застосунками на ПК, але функціонал не виправдовував ціну, оскільки такі проекти не мають відповідної якості та зручності. Їм на зміну прийшли сайти-конструктори, вони є у вільному доступі, мають привабливий зовнішній вигляд, зручні та швидкі в налаштуванні. З таким додатком можна створити сайт лише перетягнувши декілька блоків з панелі віджетів на вікно редактора, налаштувати зображення, текст і готово. Але функціонал в таких конструкторах обмежений і здебільшого платний. Існує безліч обмежень, які часто не підходять навіть маленьким компаніям, оскільки ціна за оплачувані можливості – невиправдана.

Інший більш перспективний варіант – використання фреймворку WordPress. Він має безліч платного та безкоштовного контенту такого як: шаблони, віджети, плагіни. Розуміючи, що безкоштовно найкращі рішення в проекті буде інтегрувати не чесно, було вирішено створити свою платформу, яка буде задовольняти стандартні потреби замовника.

Отже, можемо пересвідчитись, що платформа має включати :

1. Адаптивний дизайн.
2. SEO оптимізовані сторінки.
3. Можливість редагувати дані сайту за допомогою адмін панелі.
4. Налаштовані файли sitemap.xml та robots.txt для допомоги пошуковим роботам.
5. Простий та інтуїтивний інтерфейс.

1.2 Аналіз та характеристика аналогів

Сьогодні створено чимало застосунків для створення сайтів. Для того, щоб створити власний проект для швидкого створення сайтів, потрібно вивчити переваги та недоліки аналогів, щоб уникнути ряду помилок. Це також допоможе визначитись як покращити свою платформу та її уніфікувати. Переглянути чого не вистачає в аналогах, що ускладнено та чи можна реалізувати краще за допомогою продуктивніших алгоритмів та засобів. У ході роботи було виявлено, що на даний момент існуючі сайти-конструктори не покривають всіх вимог та забаганок пошукових роботів Google.

У межах даної дипломної роботи досліджено чотири аналоги, які реалізовують схожі рішення даної предметної області. Перший аналог зображений на рис.1.1.

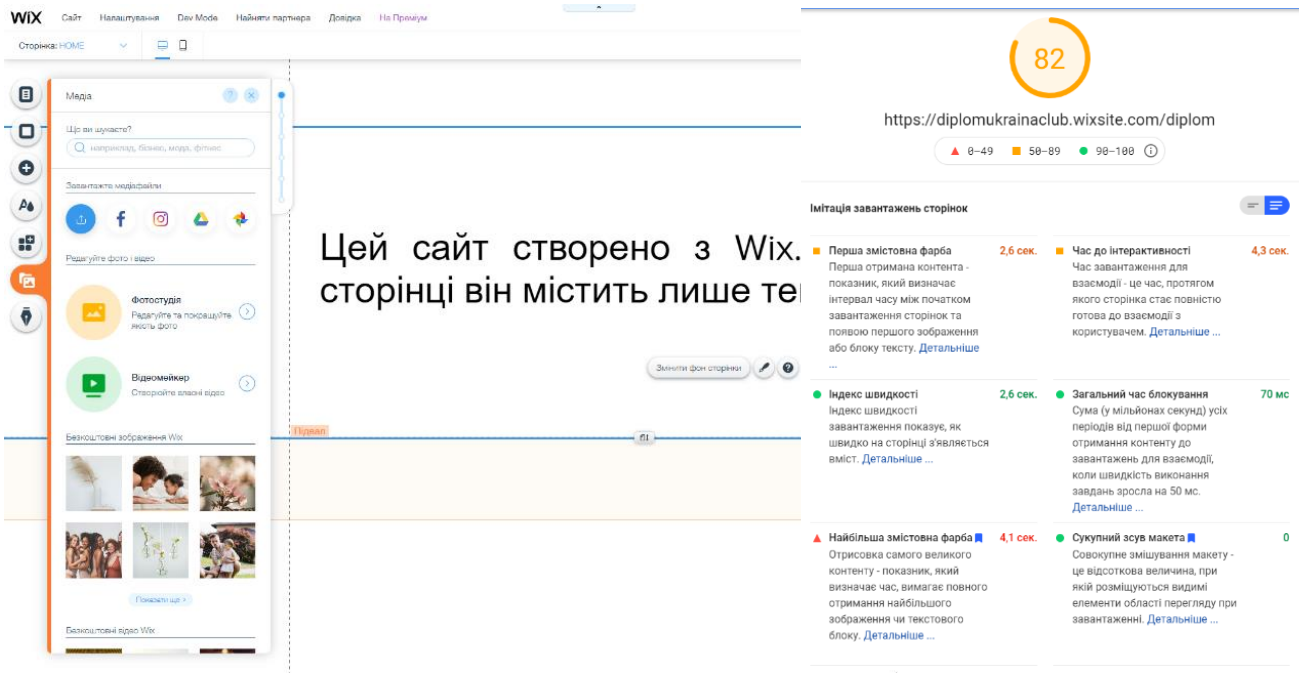


Рисунок 1.1 - Платформа Wix

Wix.com [1] – платформа, що дозволяє швидко створювати сайти різних типів. Логіка додатку полягає в створенні «макету» сайту, методом Drug-and-Drop

з панелі перетягуються блоки і в самому макеті редагуються. Хочу звернути увагу на кількість контенту, яким наповнена тестова сторінка. Тепер варто глянути на показники від бота Google, який перевірив швидкість завантаження єдиної сторінки сайту. Результати такі: 2,6 секунди до відображення першого контенту та 4,1 сек. до повного завантаження сторінки. Зауважено, що в аналогах використовуються пусті шаблони, без зображень.

Сайт має зручне управління, але перший недолік був помітний одразу. Wix не дає змогу авторизуватись за допомогою сервісів Google, хоча це офіційний сайт платформи, навіть не дочірній домен. До речі, встановити власний домен можна за 4,5$ на місяць і це лише мінімальний тариф з досі включеною рекламою від Wix на сайті. Є налаштування дизайну окремо для смартфону, сторінки помилки 404, що вигідно виділяє Wix серед інших аналогів. В безкоштовній версії доступне налаштування сайту на різних мовах. Цим фактом Wix вже заслуговує тримати лідерство серед сайтів-конструкторів і підтверджує своє заслужене місце в списках рейтингів.

Tilda [2] – простий інтуїтивний інтерфейс рис. 1.2.

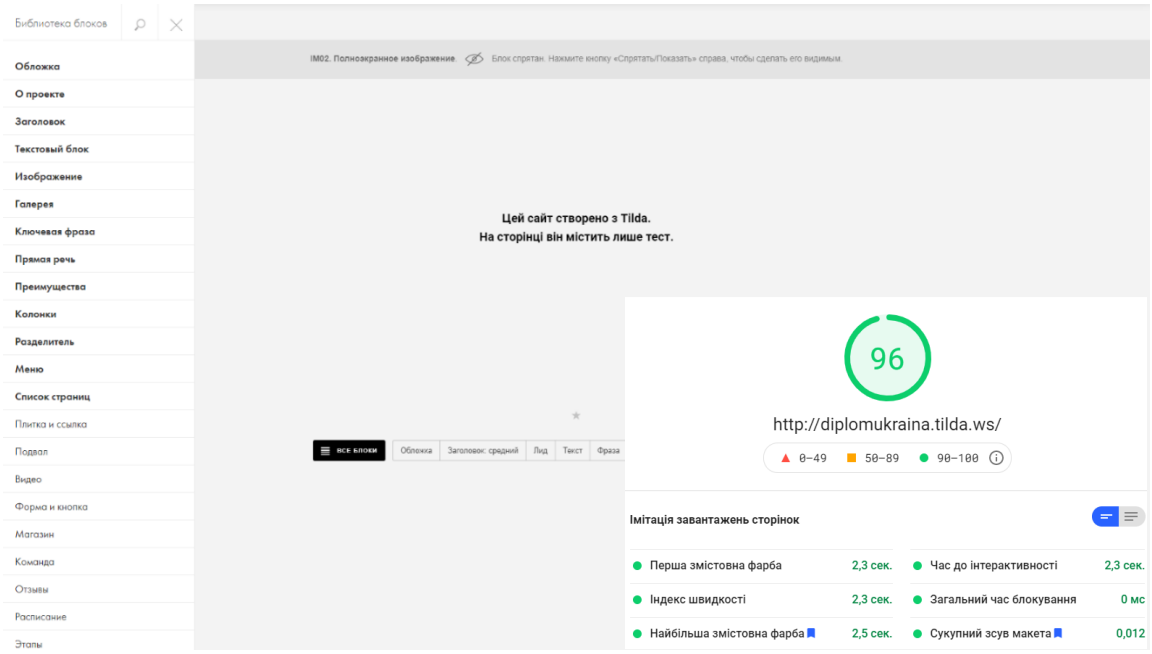


Рисунок 1.2 - Платформа Tilda

Ця платформа, на мою думку, є кращою для певних цілей, оскільки сайт створений на ній, завантажується швидше, а повний контент можна отримати на 1,5 сек. швидше для кожної сторінки. Переглядаючи сайт корситувач зазвичай потрапляє не на одну сторінку, а спочатку проходить ланцюг тест-кейсів, щоб досягнути отримуваного результату. Врешті-решт, може назбиратись немала цифра – час, який користувач витрачає на очікування поки завантажиться сторінка. Tilda також не дозволяє підключити свій домен до сайту на безкоштовному плані. Вартість мінімального тарифу 300 грн. в місяць. Але неодмінною перевагою цієї платформи є функція експорту коду, що може знадобитись деяким веб-спеціалістам. Також вони пропонують послуги веб-дизайнерів, які мають досвід роботи з Tilda та вміють з ним добре працювати. Це великий плюс для гарного досвіду користувачів.

uCoz [3] – платформа, що дає змогу вносити зміни до CSS, рис. 1.3.

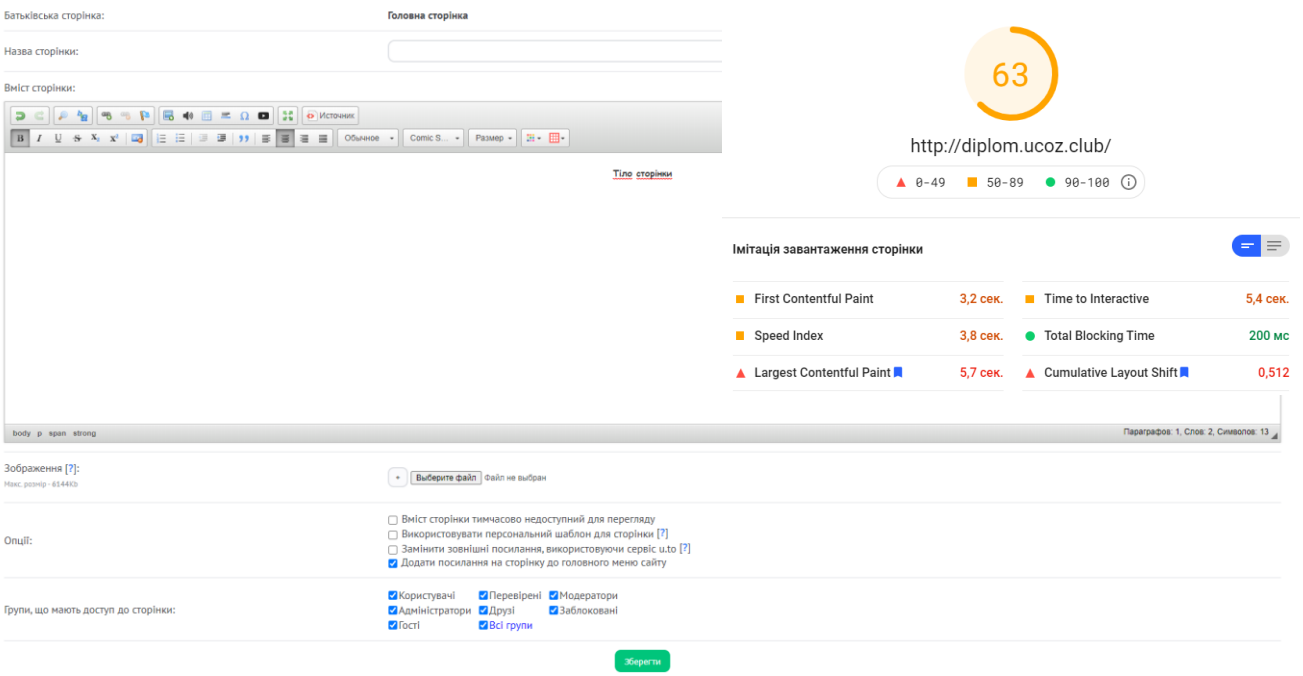


Рисунок 1.3 – платформа uCoz

Найбільш давня та відома серед представлених платформ – uCoz, має ряд переваг такі, як: керування ролями та можливість власноруч вносити зміни до

CSS, що звісно є дуже корисною функцією, до якої інші представлені аналоги доступу не надають, навіть з найдорожчим тарифом така можливість відсутня. Та при цьому швидкість завантаження сторінок на uCoz бажає бути кращою. Використання цієї платформи однозначно негативно вплине на користувацький відгук та й інтерфейс самої платформи .

Сторінка редагування на платформі Blogger зображена на рис. 1.4.

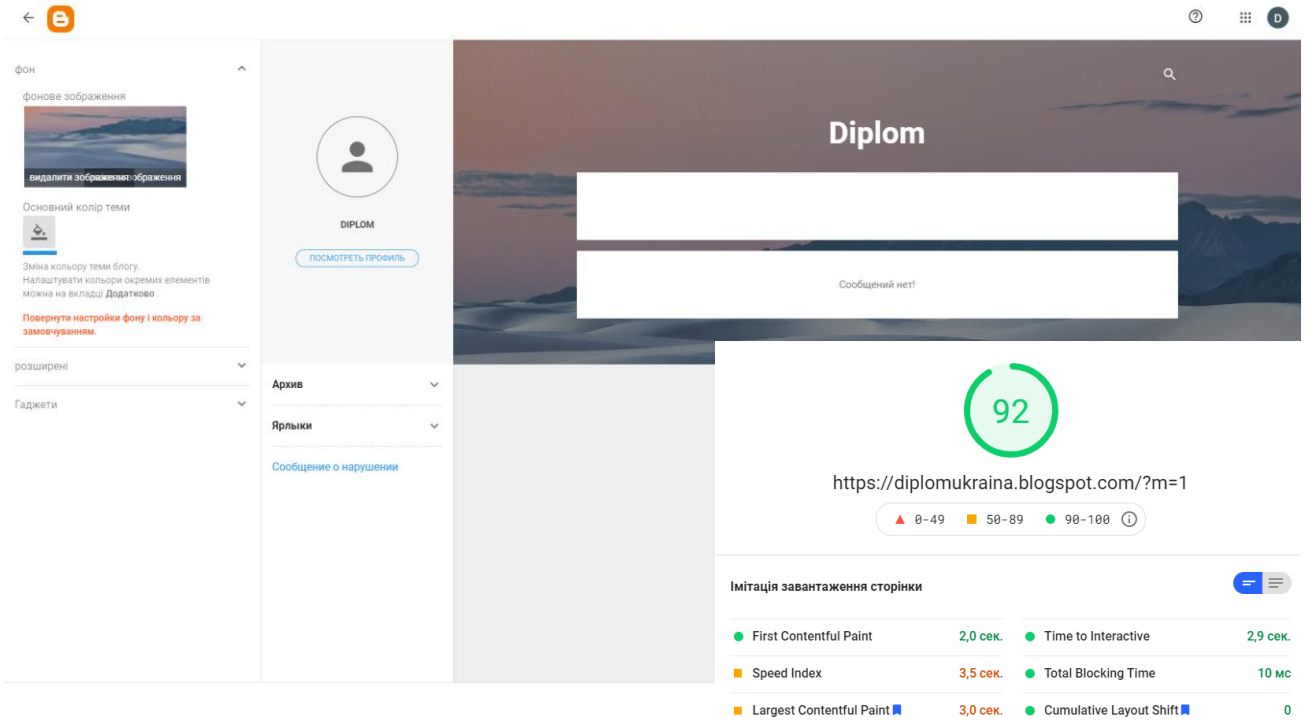


Рисунок 1.4 – Платформа Blogger

Платформа Blogger [4] була створена компанією Google. Фото тесту показує, що перший контент з’являється за дві секунди, на повне завантаження необхідно 3 сек. Можливостей налаштування сайту значно менше у порівнянні з аналогами, налаштування шаблонів примітивні, при цьому шаблони майже однакові та їх мало. Мета Blogger не в палітрі функціональних можливостей, а в простоті та швидкості налаштування. Найголовніше – він не вимагає фінансових

витрат. У Blogger влаштовано основні інструменти Google Analytics та Google Ads. Сайт можна створити лише з доменом sitename.blogspot.com .

У даному розділі було розглянуто найкращі сайти-конструктори, які доступні на ринку. За рахунок того, що конструктори постійно покращують свої алгоритми та розширюють функціонал, а вимоги користувачів та пошукових ботів змінюються щоденно, платформа повинна мати всі зручні та необхідні функції для подальшого масштабування проекту. Основним рішенням було розробка та проектування платформи, яка буде відповідати всім вимогам SEO, швидко завантажувати сторінки, та дозволить швидко створити сайт, який буде добре сприйматись користувачами та роботами Google.

Таблиця 1.1 - Недоліки та переваги аналогів

Назва застосунку	Переваги	Недоліки
Wix	1. Підтримка ручного та автоматичного перекладу. 2. Адаптивний дизайн.	1. Швидкість завантаження сторінок. 2. Влаштована реклама навіть з платними тарифами.
Tilda	1. Простота використання. 2. Можливість експорту коду.	1. Немає підтримки багатьох мов. 2. Відсутність функціоналу перевірки знань.
uCoz	1. Можливість вносити корективи в CSS файл.	1. Застарілий дизайн. 2. Немає підтримки багатьох мов.
Blogger	1. Влаштовані Google сервіси. 2. Швидке завантаження сторінки. 3. Безкоштовний.	1. Занадто простий та малофункціональний інтерфейс. 2. Відсутність змоги масштабувати свій сайт за потреби.

Існують також інші платформи, що схожі між собою, виконують подібні функції та задовольняють основні потреби користувачів. А саме – швидко створюють сайти та звільняють від необхідності в програмуванні.

Але основним мінусом багатьох платформ є те, що продукт в результаті повільно завантажує сторінки та одразу ж має негативні результати. Крім цього в безкоштовній версії вони вимагають використовувати власний домен третього рівня, який має негативний вплив на SEO оцінку. Важливою функцією на сайті є підтримка декількох мов, оскільки кожен потенційний споживач хоче отримати змогу переглядати контент зручною для себе мовою, а також деякі бізнеси, які хочуть просувати свої продукти за межами своєї країни, не мають іншого виходу та використовують різні методи для реалізації багатомовності. Але не всі ці методи завжди відповідають SEO ідеалам. А для конкуренції необхідно використовувати всіх можливі методи аби покращити власний результат та стати найкращим.

Враховуючи всі переваги та недоліки аналогів, що описані в таблиці 1.1 вирішено розробити платформу з урахуванням усіх основних потреб споживача. Назва платформи – «D», що призначений для створення адаптивних сайтів з дотриманням усіх вимог SEO.

Основна властивість платформи «D» - це те, що використовуючи його, розробник звільняється від монотонного написання та обробки програмного коду та уникає можливих недоліків через людський фактор. Немає потреби кожного разу з початку створювати проект, адаптивно його налаштовувати, робити поправки та виправляти баги.

Основною метою платформи являється те, щоб створити SEO оптимізовану, швидку систему, яка буде лояльно сприйматись пошуковими роботами.

На рис.1.5 зображена статистика використання пристроїв для web-серфінгу у світі за 2020 рік. Близько 50 – 55 % людей використовують мобільну платформу.

Завдяки даним, вказаним на ресурсі [5] , було вирішено розробляти платформу для створення адаптивних сайтів з високою швидкістю завантаження сторінок.

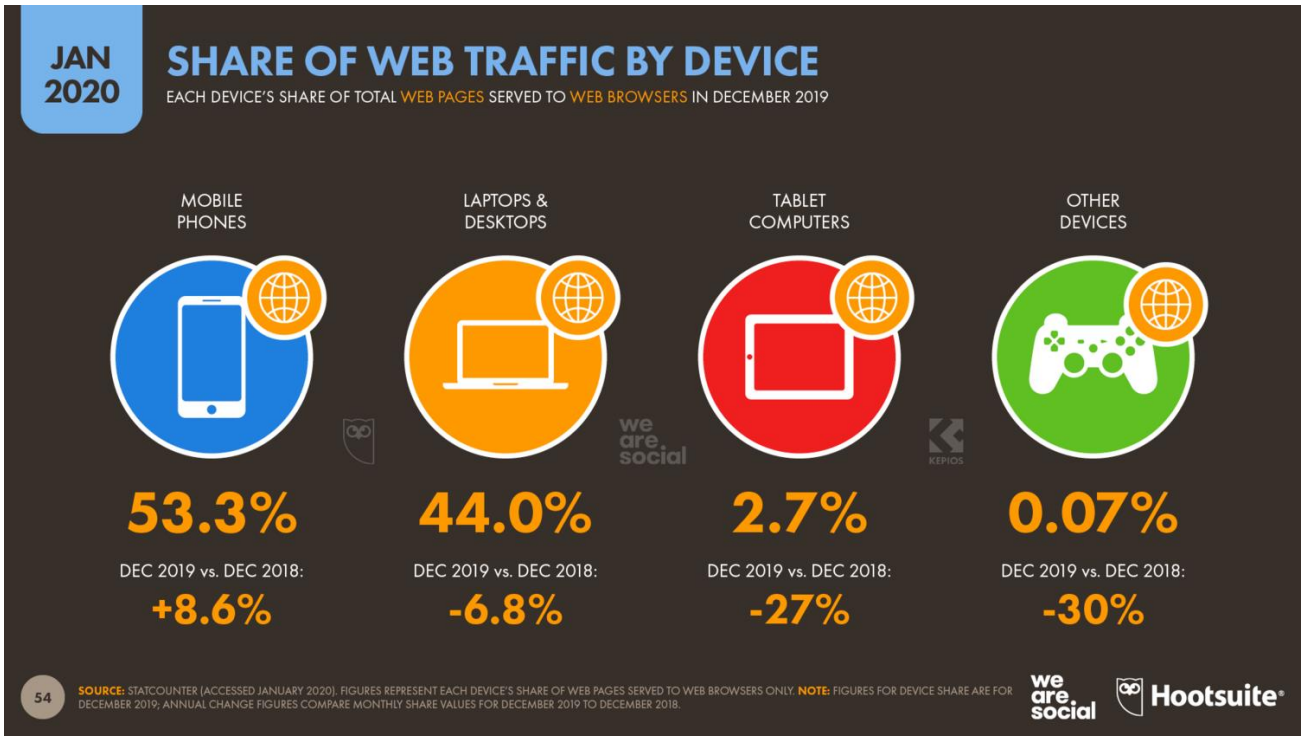


Рисунок 1.5 - Статистика використання пристроїв для web-серфінгу

Смартфон – зручний повсякденний засіб для розваг, спілкування і роботи. Оскільки попит на мобільний трафік з кожним роком зростає, вирішено створити платформу для створення адаптивних сайтів.

Переваги використання адаптивних дизайнів:

1. Відсутність необхідності розробки мобільного застосунку.
2. Лояльне сприйняття сайту користувачами та пошуковими роботами.
3. Відсутність необхідності підтримки додатку на декількох платформах.

2 ПРОГРАМНІ ЗАСОБИ ЗАСТОСУНКУ

2.1 Вибір середовища розробки

Існує три найбільш зручних середовища розробки : PHPStorm, Atom, SublimeText.

PhpStorm [6]– це найпотужніший серед всіх середовищ редактор PHP коду у якому використовуються велика кількість плагінів, які автоматично аналізують код та допомагають в розробці. Створений відомою компанією «JetBrains». Завдяки функції LiveEdit з’являється можливість відразу переглядати зміни за допомогою інтегрованого браузеру. Має влаштований Composer, Git, підтримку командного рядка та безліч інших корисних інструментів. За рахунок великої кількості реалізованих плагінів, вимагає більш продуктивні мінімальні системні характеристики для комфортної роботи. Платний, та при цьому дуже потужний.

Atom [7] – сучасний безкоштовний редактор PHP коду, в який влаштована можливість парного програмування. Для цього потрібно зробити лише кілька рухів і декілька розробників водночас може змінювати код одного файлу в різних місцях, наяву споглядавши за діями інших членів команди. Також має влаштовану систему контролю версій, гнучкий в налаштуванні.

Sublime Text [8] – текстовий редактор, доволі швидке та функціональне рішення. Має безліч корисних актуально налаштованих гарячих клавіш, які спрощують процес розробки та роблять його продуктивнішим. Ціна всього 80$ за 3 роки оновлень з подальшою підтримкою останньої скачаної версії. Безкоштовний тариф не має реклами та обмежень - лише нагадування, що можна придбати ліцензію. Невибагливий до системних параметрів, крос-платформний, потужний API та екосистема, що дозволяють розширити функціонал за допомогою плагінів.

Порівняно з іншими редакторами, у Sublime Text влаштовані всі потрібні функції для комфортної роботи, крім цього він дозволяє користуватись безліччю інших програм на фоні, оскільки майже зовсім не навантажує систему.

2.2 Функціональні модулі редактора SublimeText

На рис. 2.1 зображено інтерфейс редактора SublimeText з активованою Command Palette(Палітрою команд).

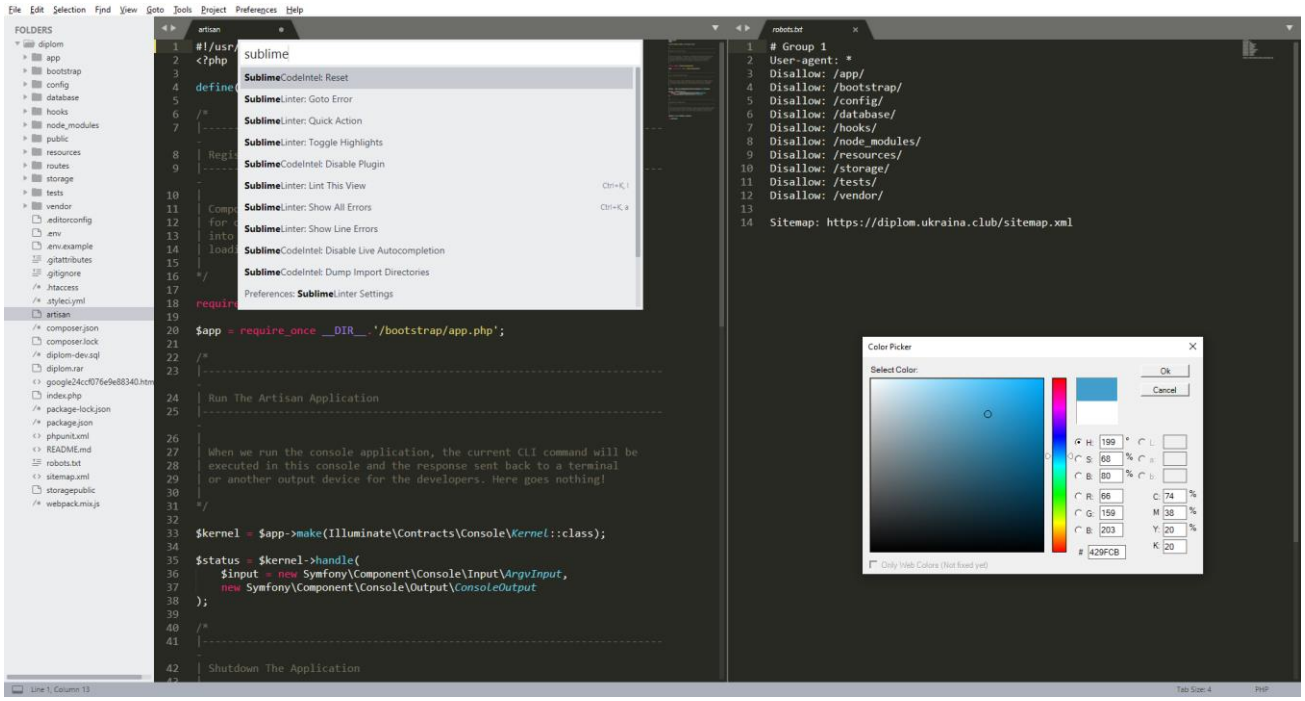


Рисунок 2.1 - Інтерфейс редактора SublimeText

Загальні модулі для розробки в редакторі SublimeText:

1. Goto Anything дозволяє швидко переміщатись між папками та файлами в проекті, а також створювати, видаляти, перейменовувати.
2. Спліт редагування дає можливість розділити свій монітор на зручну кількість частин, щоб редагувати в них код.

3. Велика кількість підтримуваних синтаксисів та типів файлів, що звільняє від необхідності встановлювати допоміжні програми. Також можливе власне налаштування підсвічування синтаксису.
4. Command Palette надає доступ до палітри команд, які при необхідності завантажують та встановлюють необхідні плагіни, такі як ColorPicker, Terminal і щось схоже на CSS Cross Browser Helper, який автоматично визначає необхідні стилі для різних браузерів та автоматично вносить зміни в файл CSS.

2.3 Функціональні модулі середовища OpenServer

Серед найзручніших додатків для розгортання локального серверу можна виділити дві програми: Denwer та OpenServer.

Denwer [9] – зручний та простий у використанні набір дистрибутивів. Ним дуже легко користуватись, але він застарілий. Остання версія підтримує PHP 5.3.13, датований 2012 роком. Найбільш вагомий недолік – довга затримка між моментом збереження файлу та його оновленням в кеші серверу, що робить даний застосунок не актуальним для комфортної розробки.

OpenServer [10] – найпотужніше програмоване середовище, створене для веб-розробників з врахуванням всіх рекомендацій та побажань. Має багато корисних серверних, PHP та СУБД модулів. Безліч можливих налаштувань керується зручним інтерфейсом. Безкоштовний.

Було вирішено використовувати OpenServer, оскільки він є найкращим рішенням на ринку.

Доступні налаштування серверу: HTTP, PHP, MySQL/MariaDB, PostgreSQL, MongoDB, Memcashed, Redis, DNS.

Серед зручних інструментів потрібно виділити phpMyAndmin, SQL Manager та консоль.

Інтерфейс додатку OpenServer зображено на рис. 2.2.

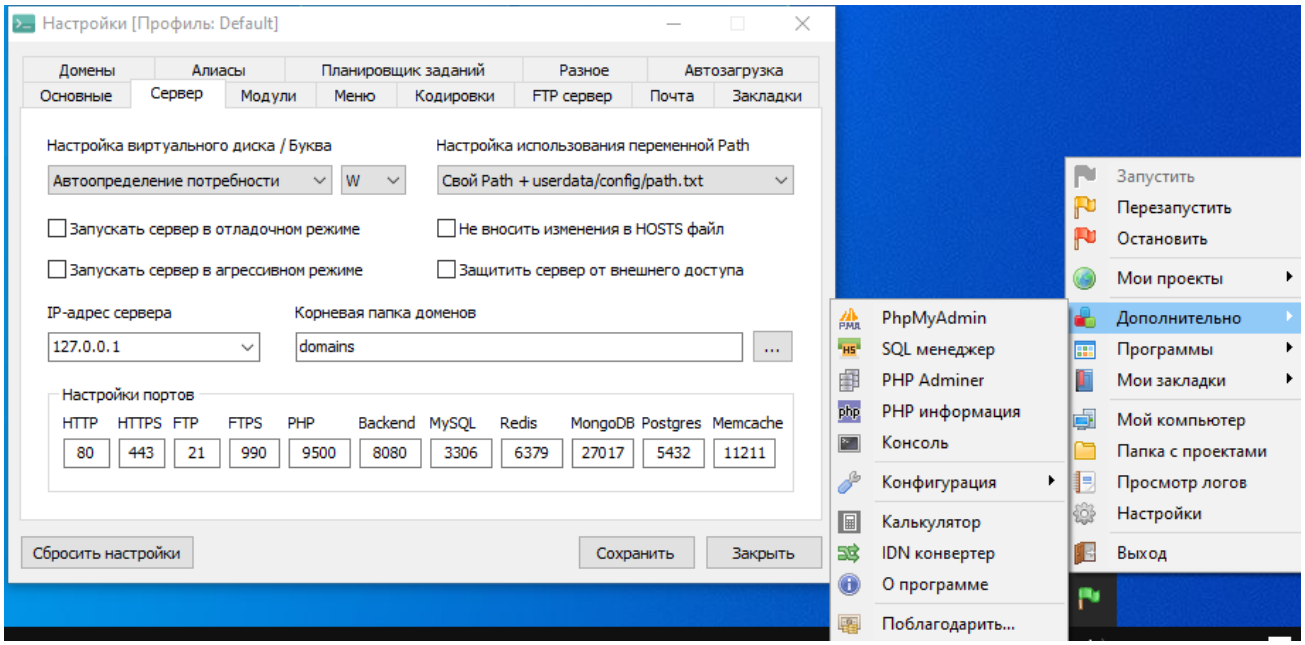


Рисунок 2.2 – Інтерфейс додатку OpenServer

2.4 Вибір фреймворку для розробки

Існує три найбільш зручних PHP фреймворка для розробки : WordPress, Symfony, Laravel.

WordPress [11] – система керування вмістом з відкритим кодом, перший випуск датовано травнем 2003 року. За рахунок своєї простоти та зручності в налаштуванні та використанні, заволодів великою кількістю фанатів та розробників. Має велику бібліотеку всіляких тем та плагінів, які дозволять з легкістю почати роботу. Та на сьогодні його актуальність зменшилась, оскільки він вже застарілий, хоч оновлення виходять до сих пір.

Symfony [12] – потужний засіб для створення сайтів, включає в себе пакетну та компонентну систему, яка дозволяє обирати потрібні функції, чи використовувати всі. Також має влаштовану систему тестування, сам фреймворк дуже схожий на Yii2. Перший реліз було опубліковано в 2005 році. Він зручний, але результат не виправдовує очікувань продуктивності.

Laravel [13] – найновітніший безкоштовний PHP фреймворк, гнучкий до налаштування, має високу продуктивність та широкий спектр можливостей. Дозволяє завантажувати пакети, користуватись міграціями для зручнішої роботи з базою даних, має активну спільноту, що спрощує пошук рішення проблем при розробці.

Було вирішено використовувати Laravel, оскільки це найпродуктивніше рішення.

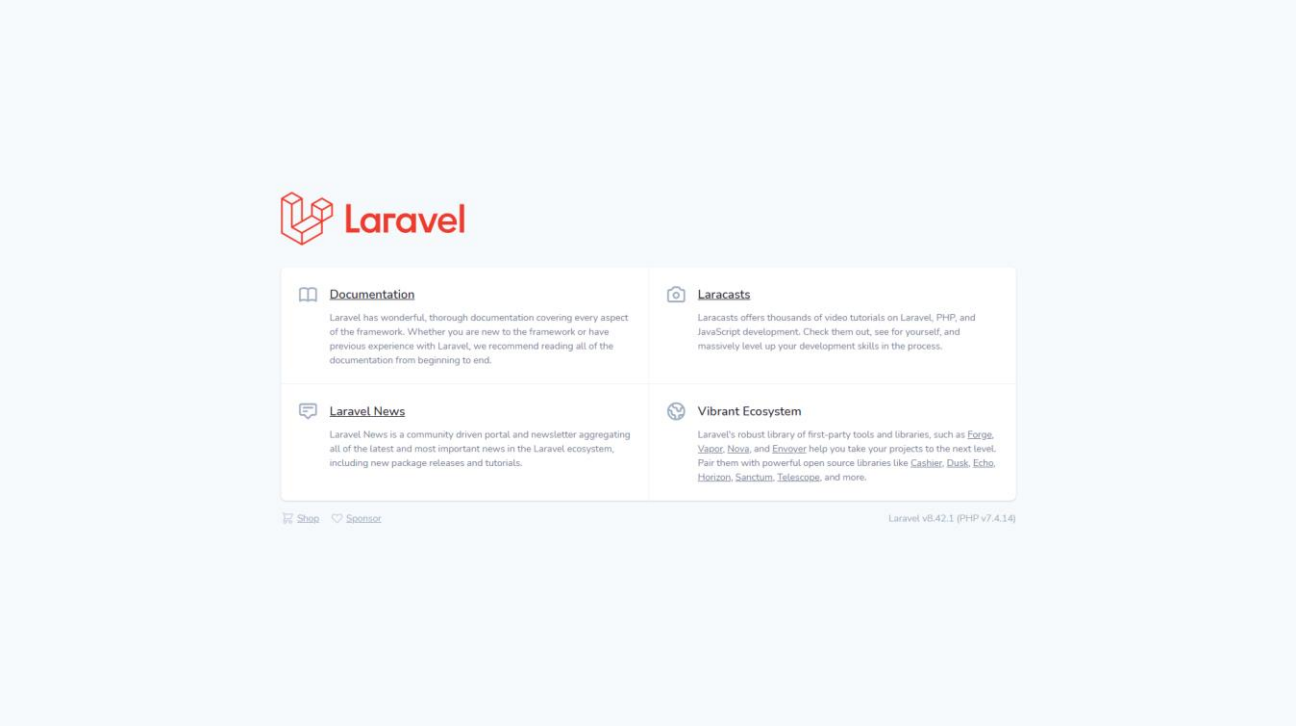


Рисунок 2.3 - Приклад пустого проекту Laravel

2.5 Laravel

Laravel – сучасний фреймворк, розрахований для розробки з використанням архітектурної моделі MVC (Model – View – Controller).

Model – модель представляє данні та методи роботи з ними. При правильному використанні стає схожою на класи в ООП.

View – представлення відповідає за взаємодію сайту з користувачем. Код компоненту View визначає зовнішній вигляд веб-сайту.

Controller – контролер відповідає за зв'язок між компонентами View та Model. Він реагує на дії користувача та вирішує що з ними робити.

Серед переваг Laravel можна виділити:

1. Міграції – система управління версіями для БД. Дозволяють зберігати її шаблон, що спрощує процеси розгортання й оновлення.
2. Влаштована система тестування та функція пагінації сторінок.
3. Використання шаблонів .blade.php . Зазвичай створюється файл іменований app.blade.php, який лише ділиться на різні секції, а код інтегрується з інших файлів, таких як: head- , header- , main- та footer.blade.php. При запуску сторінки Controller передає потрібні моделі в шаблон .blade.php, наслідуваний від app.blade.php, потім формує контент, який буде відображено в тілі екземпляру. Це і є компонент MVC - View.
4. Можливість підключення різних адмін панелей в залежності від потреб.
5. Влаштована підтримка сесій.

Структуру чистого проекту Laravel зображено на рис 2.4.

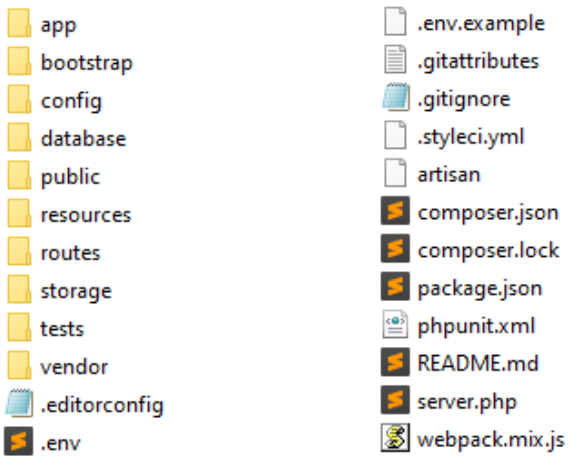


Рисунок 2.4 – Структура чистого проекту Laravel

Файл .env містить всі налаштування необхідні для коректної роботи сайту на сервері. А саме: інформацію про додаток, дані підключення до БД, логи, тривалість сесії та інші. Шаблон файлу .env зображено на рис 2.5.

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:ncKDgdOIEShHx2I8+5rYM9NgZN3o/dw5KHVkvp56yfQ=
APP_DEBUG=true
APP_URL=http://laravel.test

LOG_CHANNEL=stack
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=

BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120

MEMCACHED_HOST=127.0.0.1

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=smtp
MAIL_HOST=mail.adm.tools
MAIL_PORT=2525
MAIL_USERNAME=diplom@ukraina.club
MAIL_PASSWORD=AAaa11..
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=diplom@ukraina.club
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=mt1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

Рисунок 2.5 – Шаблон файлу .env

2.6 Вибір адмін панелі

Існує більше 10 готових адмін панелей для Laravel, більшість з них або має влаштований конструктор шаблонів, або малофункціональний.

Серед найкращих рішень було обрано 3 готових адмін панелі: Black Dashboard, Material Dashboard, Voyager.

Black Dashboard Laravel [14] – гарна зовні та потужна адмін панель. Не потрібно витрачати час на інтеграцію інтерфейсу, оскільки вже є налаштовуваний шаблон, включаючи сторінки авторизації, реєстрації, перегляду та редагування профілю. Влаштована підтримка основних функцій CRUD (Create – Read – Update – Delete).

Приклад панелі зображено на рис 2.6.

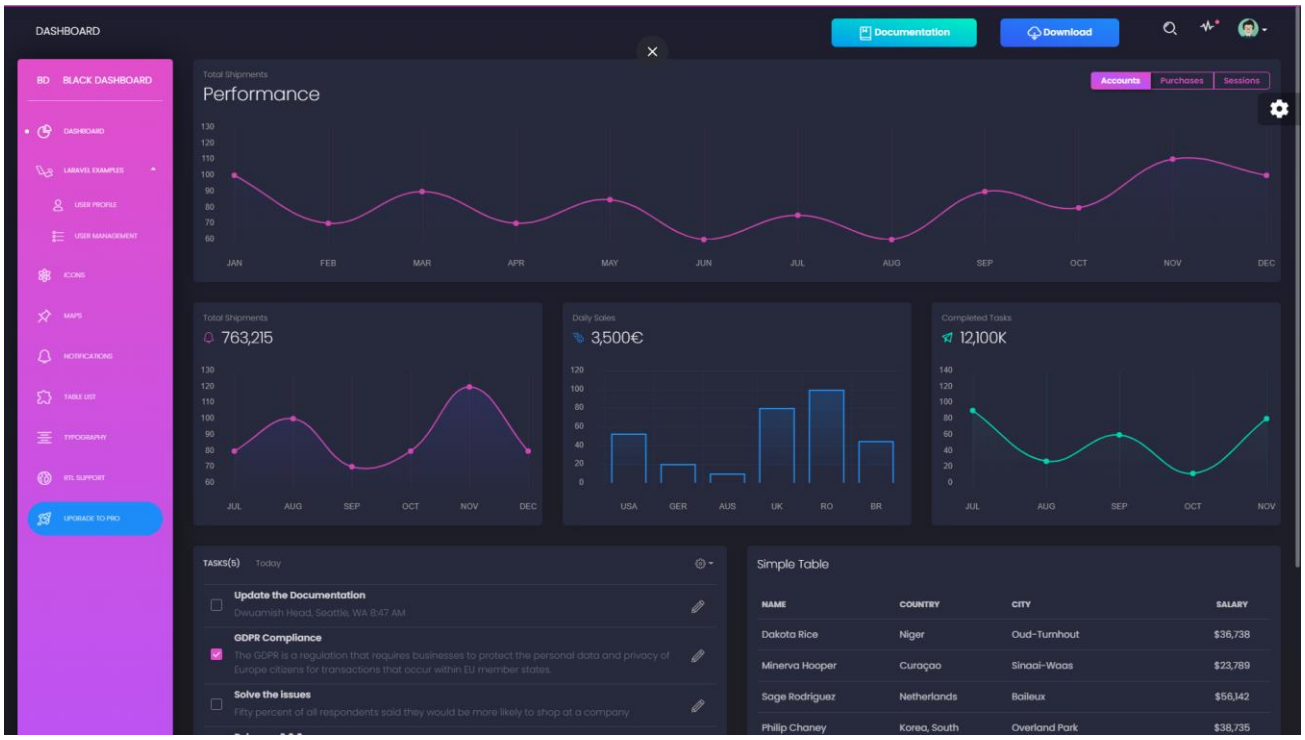


Рисунок 2.6 – Адмін панель Black Dashboard Laravel

Material Dashboard Laravel [15] – повний аналог вищевказаної адмін панелі, створений тією ж командою. Менш функціональний, але більш гнучкий до налаштування візуальної частини сайту. Зовнішній вигляд панелі зображено на рис 2.7.

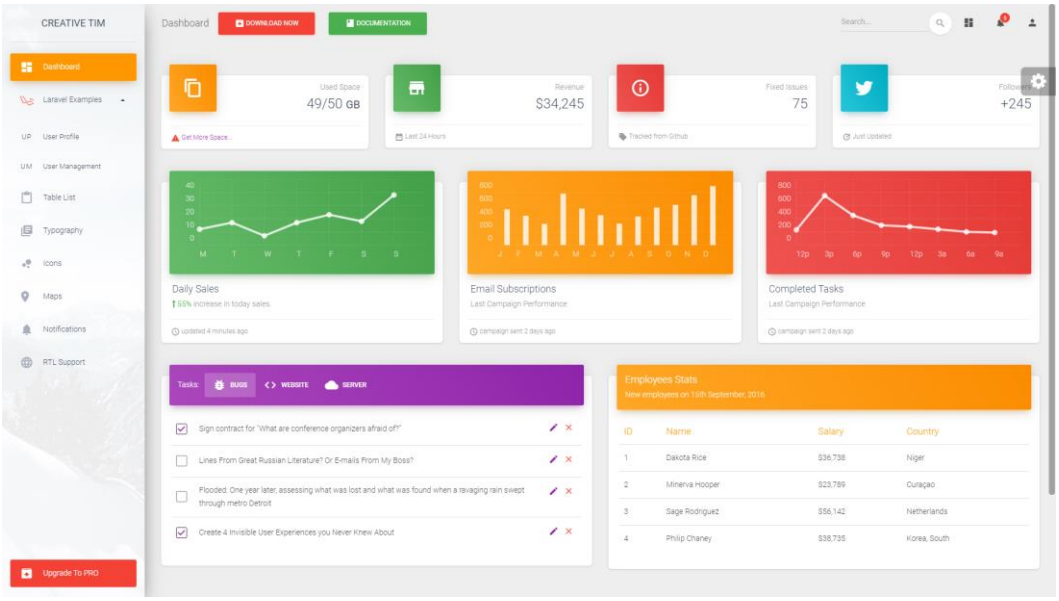


Рисунок 2.7 – Адмін панель Material Dashboard Laravel

Voyager – The Missing Laravel Admin [16]. Дана панель – це безкоштовний пакет адміністрування, який підтримує основні функції BREAD (Browse – Read – Edit – Add – Delete). Має сучасний інтерфейс та зручний у використанні. Одна з дуже корисних функцій – відображення Google Analytics статистики. Можна виділити можливість керувати БД, влаштований конструктор меню, інтегровані SEO налаштування для сторінок та фото та особливо функція управління ролями та правами. Інтерфейс адмін панелі зображений на рис 2.8.

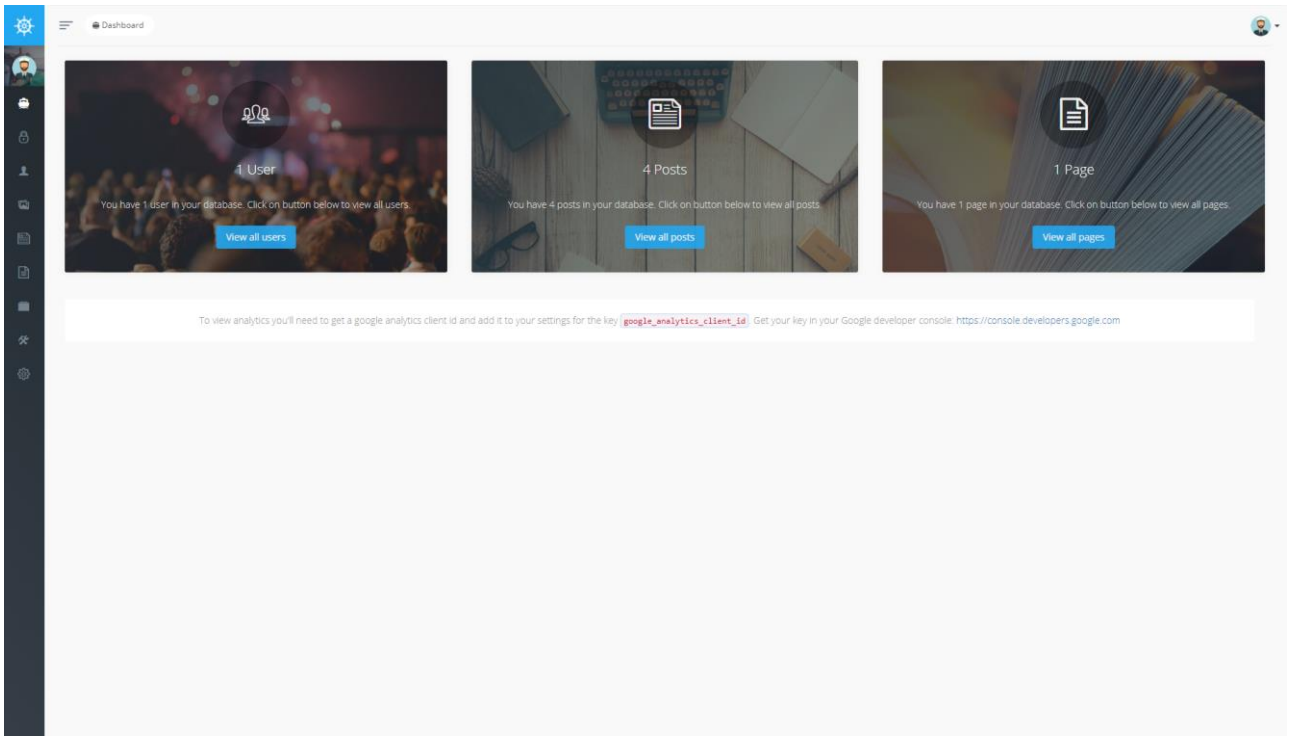


Рисунок 2.8 – Voyager – The Missing Laravel Admin

2.7 Використання платформи Lucid для створення макету дизайну платформи

Під час розробки платформи необхідно створити інтуїтивний інтерфейс, щоб кінцевий споживач отримав найкращий досвід користування. Для цієї цілі, було обрано веб-додаток Lucid. Для найкращого результату необхідно одразу розробити дизайн для комп’ютерної та мобільної версій.

Для даних цілей платформа Lucid підходить, оскільки:

1. Безкоштовна.
2. Має безліч влаштованих фігур.
3. Імпорт фігур.
4. Підтримує колективні проекти.
5. Зручна в використанні.

Приклад макету для платформи зображений на рис. 2.9.

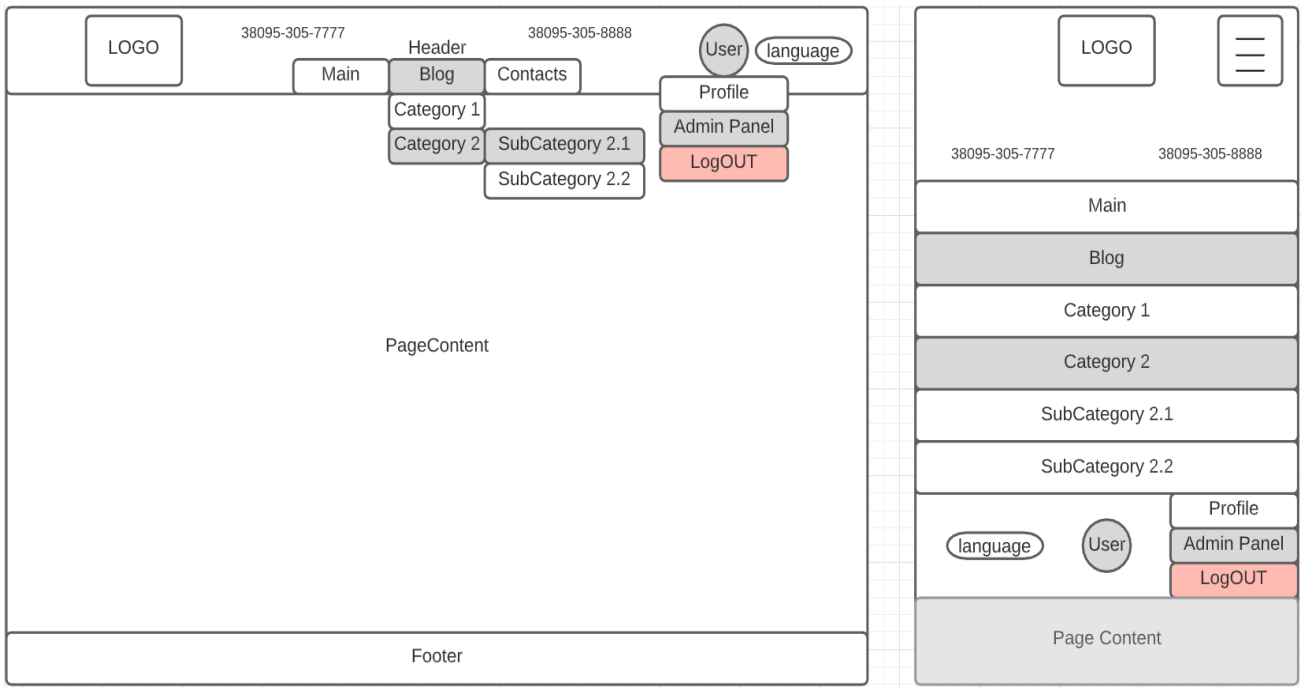


Рисунок 2.9 – Макет дизайну платформи

Lucid [17] – безкоштовний сервіс який допомагає швидко візуалізувати свої думки використовуючи об’єкти блок-схем, тексту та наліпок.

3 МОДЕЛЮВАННЯ ТА РОЗРОБКА ЗАСТОСУНКУ

3.1 Постановка задачі та опис можливостей платформи

Сьогодні в мережі Інтернет нараховується біля 200 мільйонів активних веб-сайтів та більше 4 мільярдів користувачів використовуюсь свої смартфони для споживання контенту.

Ніша створення залишається прибутковою, але має певні сумнівні перспективи так, як все в нашому світі автоматизується та процес створення сайтів теж. На ринку розробників з'явились конкуренти – сайти-конструктори, які дозволяють користувачу створити власний сайт без знать в програмуванні та управлінні сервером. Але вони не є актуальними для більшості проектів, оскільки мають ризик без сліду зникнути разом із сайтом. Більшим аргументом є не завжди виправдана ціна за ту якість послуг, які отримує користувач. Сам процес створення сайту в такому редакторі лише приносить задоволення, але в результаті ми отримуємо низькоякісну сторінку, оскільки в кожного конструктора свої недоліки та досягти бажаного результату ніяк не вийде. Це стосується замовників, які цінують свій та користувацький час, і прагнуть до ідеалу.

Завдання, яке має покривати застосунок полягає у тому, щоб надати можливість розробнику швидко створити сайт за допомогою готової платформи не витрачаючи час на основні та обов'язкові етапи роботи, при цьому отримавши результат, який задовільнить користувача, роботів Google та замовника.

Важливо розробити платформу, яка буде містити функціонал та характеристики, щоб покривати вимоги SEO пошукових роботів.

Інтерфейс повинен бути інтуїтивним.

Основні можливості функціоналу розроблюваної платформи: швидке створення адаптивного сайту з дотриманням правил SEO, адмін панель для управління контентом, особиста сторінка користувача, коментування публікацій, генерація рядку breadcrumbs.

Можливості розроблюваної платформи:

1. Розробник налаштовує файл .env, та вказує список дозволених мов, після чого може відразу починати створювати новий сайт на будь-якому зручному йому сервері.
2. Після цього, розробник інтегрує індивідуально обраний функціонал, а також налаштовує файли CSS та їх підключення, щоб вони відповідали рекомендаціям.
3. Розробник має змогу в любий момент створити нову роль, наприклад для клієнта, випускового редактора або менеджера з різними правами доступу.
4. Клієнт отримує доступ до адмін панелі сайту, де може редагувати наповнення сторінок на різних мовах без досвіду в програмуванні, а також налаштовувати певні блоки сайту такі, як: логотип, номер телефону та інші, які будуть завідома обговорені.
5. Крім цього, клієнт має можливість генерувати XML карту сайту в адмін панелі.
6. Клієнт має можливість під’єднати ботів Viber та Telegram та сповіщати користувачів про нові публікації.
7. Налаштування пунктів меню та його вкладеність налаштовується в адмін панелі, кожен з них можна прикрасити з інтегрованої 5000 колекції іконок.
8. Відображення вкладеності меню на сайті.
9. Користувач, звертаючись до адреси сайту, швидко отримує контент сторінки, за рахунок асинхронного завантаження елементів та інших оптимізуючих процесів.
10. Для закінчення реєстрації користувача, сайт вимагає скористатись підтвердженням, надісланим на вказану електронну адресу.
11. Можлива автентифікація за допомогою сервісів Google, Facebook, Telegram.

12. Після успішної авторизації, користувач отримує доступ до власного профілю та його налаштувань, а також можливість залишати коментарі та відповідати на них.
13. Сайт легко адаптується під прямокутні екрани стандартних та великих розмірів всіх девайсів, починаючи від смартфонів.
14. Екземпляр публікації на кожній з налаштованих мов має власну адресу, що дозволяє відображати ці сторінки в пошуковій видачі, охоплювати більшу кількість користувачів та підвищити загальний рейтинг домену за рахунок кількості відвідувачів.
15. При перегляді публікації можливо переміщатись по сайту за допомогою рядка breadcrumbs з посиланнями на батьківські сторінки.
16. Пагінація сторінок.

Головна особливість платформи – в результаті ми отримуємо SEO-дружелюбний сайт, який працює швидко та в ньому дотримано й реалізовано цілий ряд правил та забаганок аналізуючих сервісів Google. Користуючись платформою, розробник звільняється від часових витрат на написання коду та виправлення багів, а також має змогу швидко налаштувати сайт під потреби клієнта. Оптимізовані методи завантаження сторінки дають перевагу над конкурентними рішеннями та позитивно впливають на користувацький відгук та результати оцінки сайту пошуковими роботами. Влаштований редактор сторінок, конструктор меню та налаштовувана багатомовність дозволяють покрити основні потреби для швидкого створення сайту. Платформа звільняє клієнта від тарифних витрат на сайтах-конструкторах та дає кращу продуктивність.

3.2 Розробка моделей платформи

3.2.1 UML діаграма діяльності

UML (Unified Modeling Language) [18] - уніфікована мова моделювання. - це мова графічного опису для об'єктного моделювання в області розробки ПЗ. У процесі моделювання система розглядається з точки зору моделей. Їхні уявлення подаються у формі діаграм, які є графічним представленням елементів розроблюваної системи.

Діаграми поділяються на:

1. Діаграма використання (Use Case)
2. Діаграма класів (Class)
3. Діаграма об'єктів (Object)
4. Діаграма станів (State chart)
5. Діаграма діяльності (Activity)
6. Діаграма послідовності (Sequence)
7. Діаграма кооперації (Collaboration)
8. Діаграма компонентів (Component)
9. Діаграма розміщення (Deployment)

Діаграма діяльності (Activity diagram) – це певний набір правил для опису основної частини логіки функціоналу застосунку рис. 3.1. Діаграма показує послідовність подій, які реалізовують загалом певну програму. Це своєрідна піраміда подій, що базуються на відкритті основної концепції застосунку. Діаграма діяльності відноситься до логічної моделі, оскільки візуалізує особливості реалізації класів та при необхідності показати алгоритми їхнього виконання.

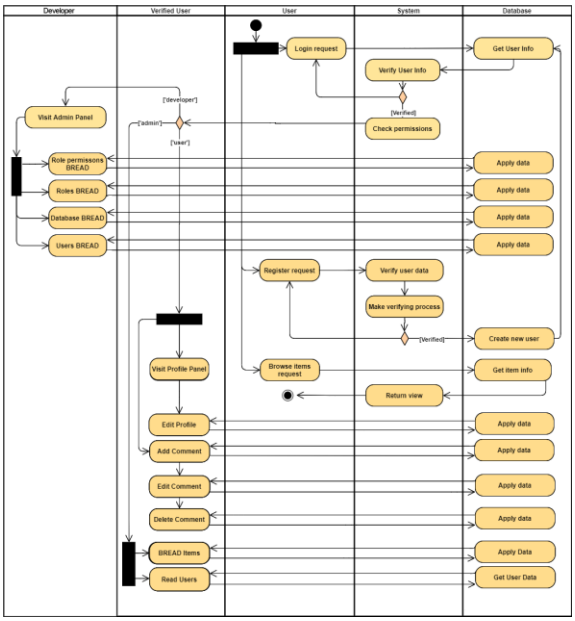


Рисунок 3.1 - UML діаграма діяльності

3.2.2 UML діаграма використання

Діаграма використання (Use case diagram) – узагальнене уявлення функціонального призначення системи. Діаграма використання надає можливість швидкого пошуку варіанту для досягнення мети користувачем. Також описує функціонал будь-якої системи, бізнес-логіку та ролі, які використовуються в схемі. Зображує маршрутизацію та масштаби розроблюваної платформи. Use case допомагає швидше орієнтуватись у взаємодіях існуючих ролей, які співпрацюють з програмою. З діаграмою використання застосовуються два типи сутностей: дійові особи та варіанти використання. Між ними встановлюються такі основні типи відносин:

1. Асоціація – між дійовою особою та варіантом використання.
2. Узагальнення – вказує спільність ролей.
3. Залежності – між варіантами використання.

UML діаграма використання зображена на рис. 3.2.

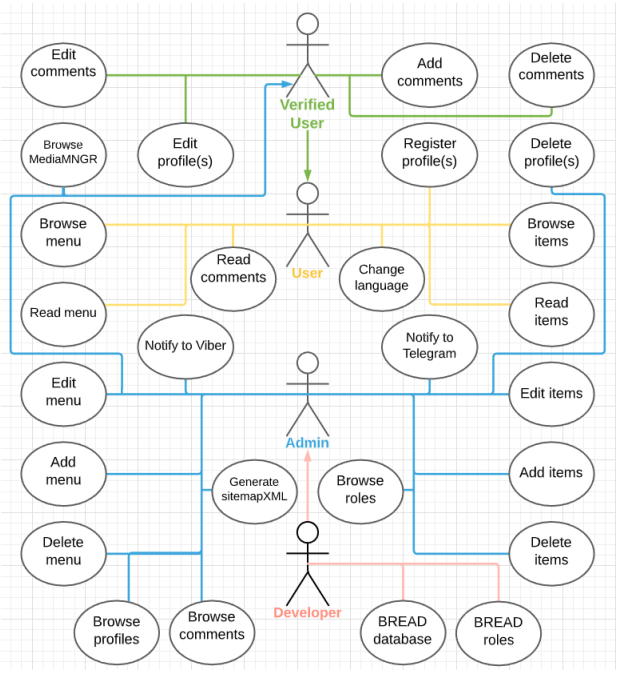


Рисунок 3.2 - UML діаграма використання

3.3 База даних MySQL

MySQL – реляційна СУБД. Є чудовим рішенням для середніх і малих проектів, оскільки має гарні властивості безпеки, швидкості та кількість можливостей. Але при більш масштабних проектах доведеться використовувати PostgreSQL, оскільки він більш лояльний до одночасних запитів типу «читання – запис», а також більш повільним, тому перехід рекомендовано робити лише при необхідності. Легка в налаштуванні та користуванні.

Переваги MySQL:

1. Простота налаштування БД.
2. Необмежена к-ть користувачів.
3. Масштабованість.
4. Система безпеки.

Управління БД відбувається за допомогою веб-інтерфейсу phpMyAdmin.

Головне вікно phpMyAdmin зображено на рис. 3.3.

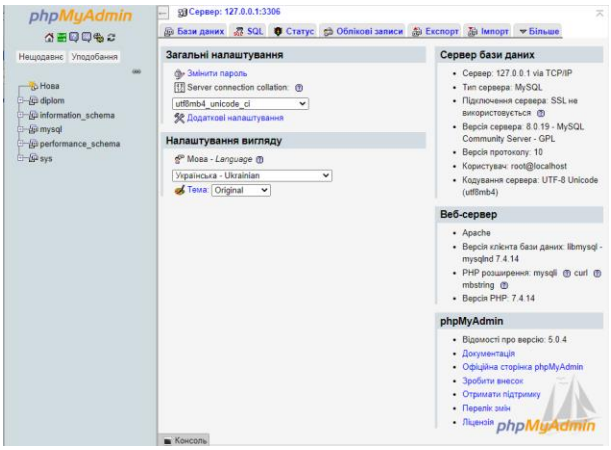


Рисунок 3.3 – Головне вікно phpMyAdmin

Для підключення БД до сайту, спочатку необхідно її створити, також додати нову таблицю і відповідно налаштувати файл .env в кореневій папці платформи. Для успішного результату, потрібно вказати: тип БД, хост, порт, назва БД та дані користувача. Підключення до БД зображено на рис 3.4.

```
10  DB_CONNECTION=mysql
11  DB_HOST=diplom2021.mysql.tools
12  DB_PORT=3306
13  DB_DATABASE=project_database
14  DB_USERNAME=admin
15  DB_PASSWORD=password
```

Рисунок 3.4 – Підключення до БД

Після підключення сайту до БД, потрібно імпортувати вихідні дані для початку роботи. Це легко зробити за допомогою влаштованих інструментів phpMyAdmin (Import) або з використанням міграцій.

Міграції – інструмент, який влаштовано в Laravel. Він допомагає розгортати нову БД та вносити зміни в існуючу. Для застосування змін в файлах міграції необхідно виконати команду php artisan migrate, рис 3.5.

```
C:\OpenServer\domains\diplom>php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated:  2014_10_12_000000_create_users_table (146.18ms)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated:  2014_10_12_100000_create_password_resets_table (53.17ms)
Migrating: 2016_01_01_000000_add_voyager_user_fields
Migrated:  2016_01_01_000000_add_voyager_user_fields (101.34ms)
Migrating: 2016_01_01_000000_create_data_types_table
Migrated:  2016_01_01_000000_create_data_types_table (256.90ms)
Migrating: 2016_01_01_000000_create_pages_table
Migrated:  2016_01_01_000000_create_pages_table (65.46ms)
Migrating: 2016_01_01_000000_create_posts_table
Migrated:  2016_01_01_000000_create_posts_table (58.37ms)
Migrating: 2016_02_15_204651_create_categories_table
Migrated:  2016_02_15_204651_create_categories_table (177.41ms)
Migrating: 2016_05_19_173453_create_menu_table
Migrated:  2016_05_19_173453_create_menu_table (190.53ms)
Migrating: 2016_10_21_190000_create_roles_table
Migrated:  2016_10_21_190000_create_roles_table (47.66ms)
```

Рисунок 3.5 – Виконання команди php artisan migrate

Приклад міграції create_users_table зображено на рис 3.6.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->string('google_id');
            $table->string('facebook_id');
            $table->string('telegram_id');
            $table->text('avatar')->nullable();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password')->nullable();
            $table->rememberToken();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('users');
    }
}
```

Рисунок 3.6 – Міграція таблиці «users»

Створення таблиць відбувається наступним чином : потрібно перезаписати метод up() батьківського класу Migration. В ньому ми описуємо створення об’єкту схеми(Schema) за допомогою статичної функції create(), де вказуємо властивості атрибутів.

У таблиці users є такі поля : id, role_id, name, password, google_id, facebook_id, telegram_id, avatar, email_verified_at, settings, remember_token, created_at, updated_at, які також мають різні типи даних та властивості. Полю id необхідно присвоїти такі параметри: AUTO_INCREMENT, PRIMARY,UNIQUE. Поле email у кожному записі має бути унікальним, бо це вказує властивість UNIQUE. Для атрибутів role_id, avatar, email_verified_at, password, remember_token, settings, google_id, facebook_id, telegram_id необхідно дозволити пусті значення NULL. Поле remember_token використовується для автентифікації користувачів: він містить дані файлів cookie, які оновлюються, що робить їх викрадення марним для шахраїв. Таблиця users зображена на рис. 3.7.

#	ім'я	Тип	порівняння	атрибути	Null	За замовчуванням	Коментарі	додатково
1	id	bigint (20)		UNSIGNED	немає	*немає*		AUTO_INCREMENT
2	role_id	bigint (20)		UNSIGNED	Так	*NULL*		
3	name	varchar (255)	utf8mb4_unicode_ci		немає	*немає*		
4	email	varchar (255)	utf8mb4_unicode_ci		немає	*немає*		
5	avatar	varchar (255)	utf8mb4_unicode_ci		Так	/public/storage/users/default.png		
6	email_verified_at	timestamp			Так	*NULL*		
7	password	varchar (255)	utf8mb4_unicode_ci		немає	*немає*		
8	remember_token	varchar (100)	utf8mb4_unicode_ci		Так	*NULL*		
9	settings	text	utf8mb4_unicode_ci		Так	*NULL*		
10	created_at	timestamp			Так	*NULL*		
11	updated_at	timestamp			Так	*NULL*		
12	google_id	varchar (255)	utf8mb4_unicode_ci		Так	*NULL*		
13	facebook_id	varchar (255)	utf8mb4_unicode_ci		Так	*NULL*		
14	telegram_id	varchar (255)	utf8mb4_unicode_ci		Так	*NULL*		

Рисунок 3.7 – Таблиця users

Далі потрібно підключити БД до розроблюваної платформи за допомогою моделей(Models). Вони допомагають зручно користуватись таблицями з БД, представляючи їх як класи. Для цього необхідно створити моделі для кожної таблиці, якщо потрібно часто до неї звертатись. Для створення нової моделі потрібно скористатись командою php artisan make:model User рис. 3.8.

```
C:\OpenServer\domains\diplom>php artisan make:model User
Model created successfully.
```

Рисунок 3.8 – Створення моделі User

При виконанні команди створюється клас User в файлі User.php, рис. 3.9, який наслідується від класу Model, котрий містить безліч функцій для роботи з БД таких, як: читання, запис інформації та інших.

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class User extends Model
{
    use HasFactory;
}
```

Рисунок 3.9 – Стандартна модель User

Потім потрібно підготувати модель для коректного виконання всіх функцій таких, як: авторизація, підтвердження за допомогою email, використання адмін панелі, коментування елементів. Кінцевий код моделі User зображено на рис. 3.10.

```php
<?php

namespace App;

use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Notifications\Notifiable;
use Laravelista\Comments\Commenter;

class User extends \TCG\Voyager\Models\User implements MustVerifyEmail
{
    use Notifiable, Commenter;

    protected $fillable = [
        'name', 'email', 'password',
    ];

    protected $hidden = [
        'password', 'remember_token',
    ];

    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}
```

Рисунок 3.10 – Налаштована модель User

Надалі використовуються моделі в контролерах(Controllers). Для цього потрібно імпортувати модель командою use App\User і користуватись статичними функціями управління БД. На рис. 3.11 зображено фрагмент контролеру ProfileController.php. Він відповідає за відображення профілю користувача. На даному фрагменті зображено метод, який надає користувачу доступ до профілю у випадку, якщо користувач авторизувався, якщо ні – контролер повертає Redirect на сторінку Login.

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Password;
use App\User;
use Auth;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Storage;
use Intervention\Image\ImageManager as Image;
use Exception;

class ProfileController extends Controller
{

    public function index()
    {
        try{
            $profile = User::where('id', '=', Auth::user()->id)->firstOrFail();
            if(is_null($profile->email_verified_at)) return view('auth.verify');
            return view('profile.index', ['profile' => $profile]);
        }
        catch (Exception $e){
            return redirect()->route('login');
        }
    }
```

Рисунок 3.11 – Контролер ProfileController.php

Звернувши увагу на код, можна побачити наступне: на 20 лінії оголошуємо змінну $profile та присвоюємо значення моделі User, команда 21 рядка перевіряє чи користувач пройшов верифікацію, у випадку позитивного результату - 22 рядок викликає функцію відображення представлення (View). Викликаючи функцію view, в аргументах вказується назву файлу та додаткові змінні, у даному випадку - $profile.

View – останній нерозглянутий елемент структури MVC. Собою він являє бібліотеку файлів, що мають розширення .blade.php та містять необхідний

HTML для відображення та PHP код для генерації контенту. Створюючи проект, необхідно: структурувати частини сторінки, розмістити код по різним файлам, раціонально користуватись імпортом. Фрагмент представлення layouts/app.blade.php, яке відповідає за відображення сторінки профілю користувача зображено на рис. 3.12.

```
22      <title>@yield('title')</title>
23  </head>
24  <body>
25
26      @include('layouts.header')
27
28      <input type="checkbox" id="check-main">
29      <main id="main">
30      @include('layouts.leftside')
31
32          <article role="main">
33          @yield('breadcrumbs')
34          @yield('content')
35          </article>
36
37      @include('layouts.rightside')
38      </main>
39
40      @include('layouts.footer')
41
42  </body>
43  </html>
```

Рисунок 3.12 – Файл layouts/app.blade.php

Фрагмент представлення profile/index.blade.php, яке відповідаює за відображення сторінки профілю користувача зображено на рис. 3.13.

```
1   @extends('layouts.app')
2   @section('title')
3       <?= $profile->name ?>
4   @endsection
5   @section('content')
6   <h1>{{ __('profile.hi') }} , {{ $profile->name }} !</h1>
7   <div id="profile">
8       <div id="profile-block">
9           @include('profile.profile_menu')
10      </div>
11      <div id="profile-info">
12          <div id="text-profile-info">
13              <h2>{{ __('profile.name') }}</h2>
14              <span>{{ $profile->name }}</span>
15              <h3>{{ __('profile.email') }}</h3>
16              <span>{{ $profile->email }}</span>
17              <br>
18              <div class="created-at">
19                  {{ __('profile.account_created_at') }}
20                  <span >{{ $profile->created_at }}</span>
21              </div>
22          </div>
23          <div class="profile-logo">
24              <img src="{{$profile->avatar}}">
25          </div>
26      </div>
27  </div>
28  @endsection
```

Рисунок 3.13 – Файл profile/index.blade.php

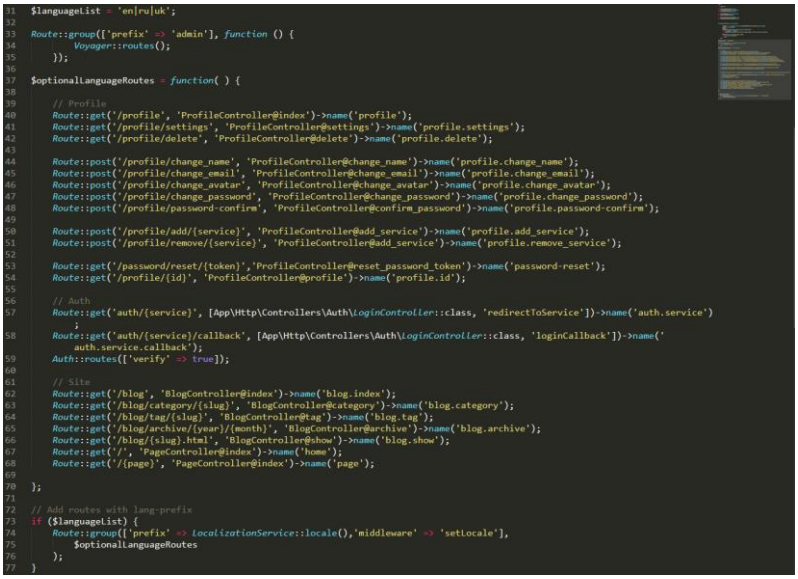
3.4 Проектування маршрутизації користувача

За маршрутизацію сторінок в Laravel відповідають роути(routes). Це абстрактні змінні, які мають характеристики GET і POST. При написанні роуту вказується: адреса, яка активує його, функція в контролері, яка обробить запит, а також можна ідентифікувати іменем, для можливості виклику з коду. Приклад роуту «profile», який активується при відкритті сторінки diplom.site/profile зображено на рис. 3.14.

```
Route::get('/profile', 'ProfileController@index')->name('profile');
```

Рисунок 3.14 – Роут «profile»

Вміст файлу web.php можна переглянути на рис. 3.15.



```
$languageList = 'en|ru|uk';

Route::group(['prefix' => 'admin'], function () {
        Voyager::routes();
    });

$optionalLanguageRoutes = function( ) {

    // Profile
    Route::get('/profile', 'ProfileController@index')->name('profile');
    Route::get('/profile/settings', 'ProfileController@settings')->name('profile.settings');
    Route::get('/profile/delete', 'ProfileController@delete')->name('profile.delete');

    Route::post('/profile/change_name', 'ProfileController@change_name')->name('profile.change_name');
    Route::post('/profile/change_email', 'ProfileController@change_email')->name('profile.change_email');
    Route::post('/profile/change_avatar', 'ProfileController@change_avatar')->name('profile.change_avatar');
    Route::post('/profile/change_password', 'ProfileController@change_password')->name('profile.change_password');
    Route::post('/profile/password-confirm', 'ProfileController@confirm_password')->name('profile.password-confirm');

    Route::post('/profile/add/{service}', 'ProfileController@add_service')->name('profile.add_service');
    Route::post('/profile/remove/{service}', 'ProfileController@add_service')->name('profile.remove_service');

    Route::get('/password/reset/{token}','ProfileController@reset_password_token')->name('password-reset');
    Route::get('/profile/{id}', 'ProfileController@profile')->name('profile.id');

    // Auth
    Route::get('auth/{service}', [App\Http\Controllers\Auth\LoginController::class, 'redirectToService'])->name('auth.service')
        ;
    Route::get('auth/{service}/callback', [App\Http\Controllers\Auth\LoginController::class, 'loginCallback'])->name('
        auth.service.callback');
    Auth::routes(['verify' => true]);

    // Site
    Route::get('/blog', 'BlogController@index')->name('blog.index');
    Route::get('/blog/category/{slug}', 'BlogController@category')->name('blog.category');
    Route::get('/blog/tag/{slug}', 'BlogController@tag')->name('blog.tag');
    Route::get('/blog/archive/{year}/{month}', 'BlogController@archive')->name('blog.archive');
    Route::get('/blog/{slug}.html', 'BlogController@show')->name('blog.show');
    Route::get('/', 'PageController@index')->name('home');
    Route::get('/{page}', 'PageController@index')->name('page');

};

// Add routes with lang-prefix
if ($languageList) {
    Route::group(['prefix' => LocalizationService::locale(),'middleware' => 'setLocale'],
        $optionalLanguageRoutes
    );
}
```

Рисунок 3.15 – Файл web.php

3.5 Розробка дизайну сайту за допомогою метамови SCSS

SASS – метамова на базі CSS, яка має 2 синтаксиси: sass, scss. Вони відрізняються лише тим, що в scss використовуються фігурні дужки, що є більш

зручним. SASS дозволяє оголошувати та використовувати змінні. Зміст файлу app.scss зображено на рис. 3.16.

```scss

// Variables
@import 'variables';



/*****************************MAIN FIRST*****************************/
/*****************************MAIN FIRST*****************************/

main, .desktop, .auth-block, article[role=main],.form-group,.pagination{
display: flex;
}

header, nav, main, article[role=main], footer,#close-menu-button-label, #close-menu-button,.form-group{
width: 100%;
}

body {
    position: relative;
    font-family: "SF Pro Display", sans-serif;
    font-weight: 400;
    margin: 0;
    height: 100vh;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
}
h1,h2,h3,h4,h5,h6{
    margin: auto;
    margin-top: 0.5em;
    margin-bottom: 0.5em;
}
```

Рисунок 3.16 – Файл app.scss

До розробки дизайну було використано особливий підхід, який спрямований на оптимізацію швидкості завантаження сторінки та зручну модифікацію в майбутньому.

Весь дизайн створюється в файлі app.scss, після чого за допомогою команди npm run dev компілюється в файл app.css, рис. 3.17.

```
C:\OpenServer\domains\diplom>npm run dev

> @ dev C:\OpenServer\domains\diplom
> npm run development


> @ development C:\OpenServer\domains\diplom
> cross-env NODE_ENV=development node_modules/webpack/bin/webpack.js --progress --hide-modules --config=node_modules/laravel-mix/setup/webpack.co
nfig.js

98% after emitting SizeLimitsPlugin

 DONE  Compiled successfully in 12646ms                                                                                                18:02:41

       Asset      Size  Chunks             Chunk Names
/css/app.css  21.7 KiB     mix  [emitted]  mix
Notifications are disabled
Reason: DisabledForUser Please make sure that the app id is set correctly.
Command Line: C:\OpenServer\domains\diplom\node_modules\node-notifier\vendor\snoreToast\snoretoast-x64.exe -pipeName \\.\pipe\notifierPipe-277c6b
c0-44c1-4c89-831e-7f378e0cbdad -p C:\OpenServer\domains\diplom\node_modules\laravel-mix\icons\laravel.png -m "Build successful" -t "Laravel Mix"
```

Рисунок 3.17 – Виконання команди npm run dev

Отриманий файл необхідно розділити на інші файли. Деякі з них містять лише стилі, які вказують точні положення в просторі та розмір елементів. Їх необхідно підключати перед завантаженням відповідного контенту. Інші ж –

містять стилі які відповідають за кольори, фони. Їх потрібно підключати в кінці <body>. Також було вирішено пізніше завантажувати всі стилі, які пов'язані зі зміною стану елементів, оскільки керування ними відбувається лише після завантаження всієї сторінки. Цей трюк прискорює завантаження сторінки зі сторони користувача та покращує декілька параметрів, за якими потрібно слідкувати за допомогою панелі Network в браузері або скористатись сервісом PageSpeed Insights від Google, який допомагає пошуковій системі генерувати оцінку сайту. Результати оптимізації зображено на рис. 3.18.

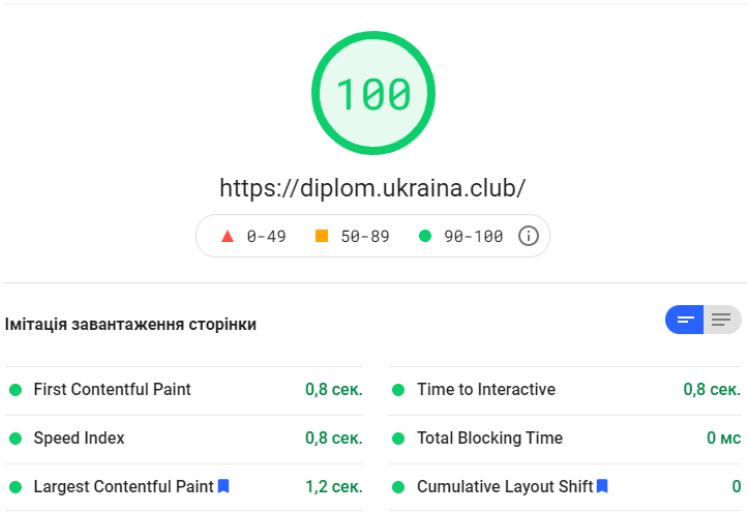


Рисунок 3.18 – Результати PageSpeed Insights

3.6 Впровадження багатомовності в проект

Існує декілька способів реалізувати багатомовність на сайті. Вони відрізняються складністю впровадження та SEO-оптимізованістю. Перший метод дозволяє використовувати плагіни для автоматичного перегляду, наприклад, від Google, що дає можливість будь-кому обрати зручну для себе мову зі списку дозволених. При індексуванні сайту, роботи аналізують лише одну мову сайту, що ніяк не допомагає просувати сайт в пошуковій видачі.

Інший метод передбачає використання піддоменів для додаткових мов, налаштованих на сайті. В результаті можна отримати декілька сайтів з доменними

іменами типу site.com, en.site.com, ch.site.com. Такий метод також не є SEO-оптимізованим, оскільки кожна мова належить різному сайту і необхідно просувати кожен домен нераціонально.

Вірним рішенням є створення префіксу для додаткових мов, в адресі сайту. Наприклад: site.com/index.html, site.com/en/index.html, site.com/ch/index.html. Дані перекладу завантажуються з БД та редагуються за допомогою адмін панелі, що звільняє від неточних помилок в перекладі ботом. Перелік мов вказується в файлі config/app.php та відразу відображається серед дозволених на сайті. Приклад налаштування та результат зображено на рис. 3.19.

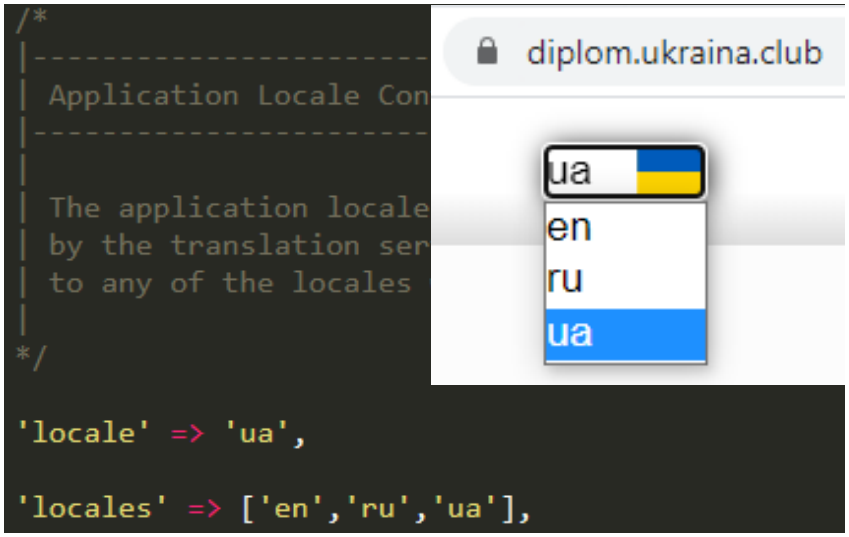


Рисунок 3.19 – Приклад налаштування багатомовності

3.7 Тестування платформи

Test Case – це текстовий документ, суть якого є виконання деякого порядку дій для тестування певного функціоналу.

Таблиця 3.1 – Функціонал адмін панелі Laravel

Передумови : перейти на сайт з вкладенням /admin в посиланні.

Умова	Опис тесту	Очікуваний результат

Ввійти в адмін панель .	1. Ввести дані авторизації. 2. Натиснути «Увійти».	Успішна авторизація в адмін панель.
Перегляд ролей.	1. Перейти на сторінку «Ролі». 2. Відтворення списку ролей в системі. 3. Натиснути на кнопку «Переглянути». 4. Відображення імені та назви ролі.	Відтворення існуючих ролей в системі.
Редагування існуючих категорій.	1. Перейти на сторінку «Категорії». 2. Відтворення списку категорій в системі. 3. Натиснути на кнопку «Редагувати». 4. Налаштування батьківської категорії, відображуваного імені. 5. Натиснути кнопку «Зберегти».	Зміни в налаштуванні категорії збережено.
Створення категорій	1. Перейти на сторінку «Категорії». 2. Відтворення списку категорій в системі. 3. Натиснути на кнопку «Створити». 4. Налаштування батьківської	Нову категорію було створено.

	категорії, відображуваного імені. 5. Натиснути кнопку «Зберегти».	
Створення постів	1. Перейти на сторінку «Пости». 2. Відтворення списку постів в системі. 3. Натиснути на кнопку «Створити». 4. Налаштування батьківської категорії, відображуваного імені, фото, опису, SEO налаштувань та повідомлень в соц. мережі. 5. Натиснути кнопку «Зберегти».	Новий пост було створено. Посилання в обрані боти було надіслано.
Редагування постів	1. Перейти на сторінку «Пости». 2. Відтворення списку постів в системі. 3. Натиснути на кнопку «Редагувати». 4. Налаштування батьківської категорії, відображуваного імені, фото, опису, SEO налаштувань та повідомлень в соц. мережі. 5. Натиснути кнопку «Зберегти».	Новий пост було успішно змінено. Посилання в обрані боти було надіслано.

Таблиця 3.2 – Тестування функціоналу сайту

Сторінка «Блог»	1. Перейти на сайт. 2. Натиснути на кнопку «Блог». 3. Обрати категорію та підкатегорію. 4. Список постів відображується коректно. 5. Список вкладеності сторінок відображується одразу після вибору категорії. 6. Натиснути на будь – який пост. 7. Змінилась вкладеність сторінки(посилання та breadcrumbs). 8. Відображується контент посту та коментарі.	Публікацію відображено.
Залишити коментар	1. Перейти на публікацію. 2. Список коментарів відображується одразу після контенту публікації. 3. Ввести коментар в поле для вводу. 4. Натиснути кнопку «Відправити»	Коментар опубліковано.

4 МАРШРУТИЗАЦІЯ

4.1 Маршрутизація користувача

Для користування сайтом користувачу потрібно звернутись до доменного імені, після чого відкривається головна сторінка з стандартно налаштованою мовою. Приклад сторінки на комп’ютері та телефоні зображено на рис.4.1.

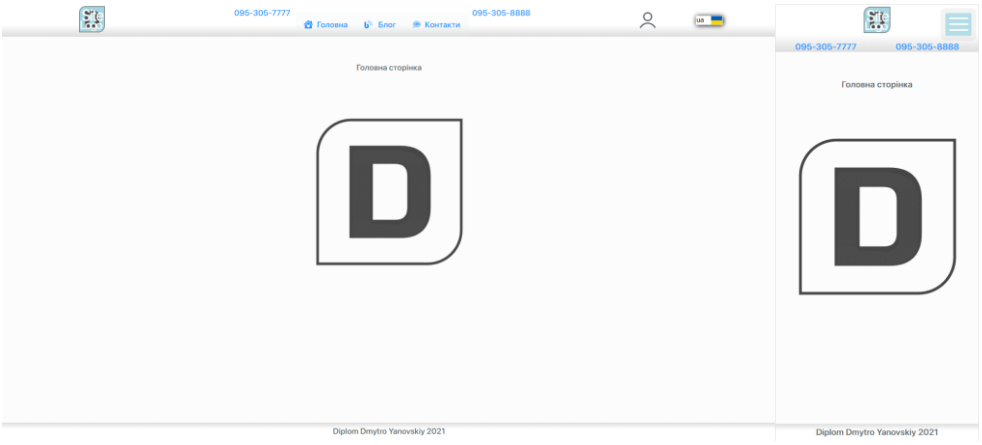


Рисунок 4.1 – Головна сторінка сайту

Для того, щоб зареєструватись, користувачу потрібно пройти перейти на сторінку реєстрації, рис. 4.2.

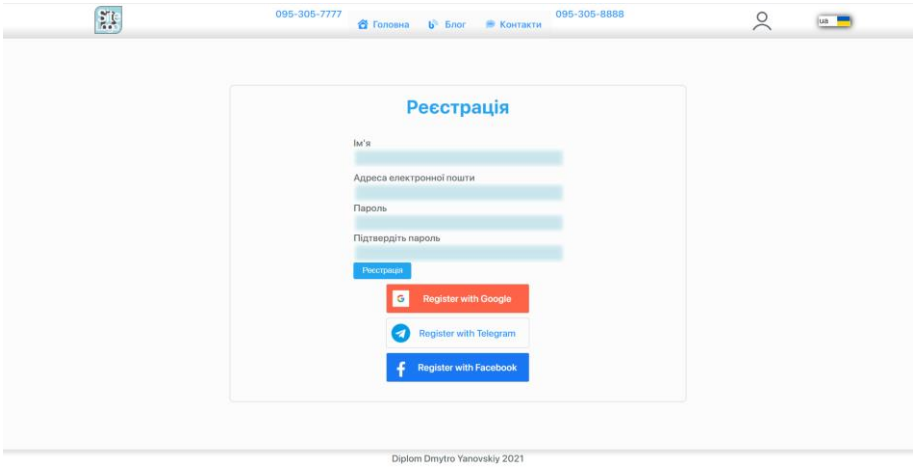


Рисунок 4.2 – Відеофайл, що завантажив користувач

Створити новий профіль можливо за допомогою підтвердження електронної адреси або скориставшись авторизацію за допомогою сервісів Google, Facebook, Telegram. Приклад роботи одного з сторонніх методів авторизації зображено на рис. 4.3.

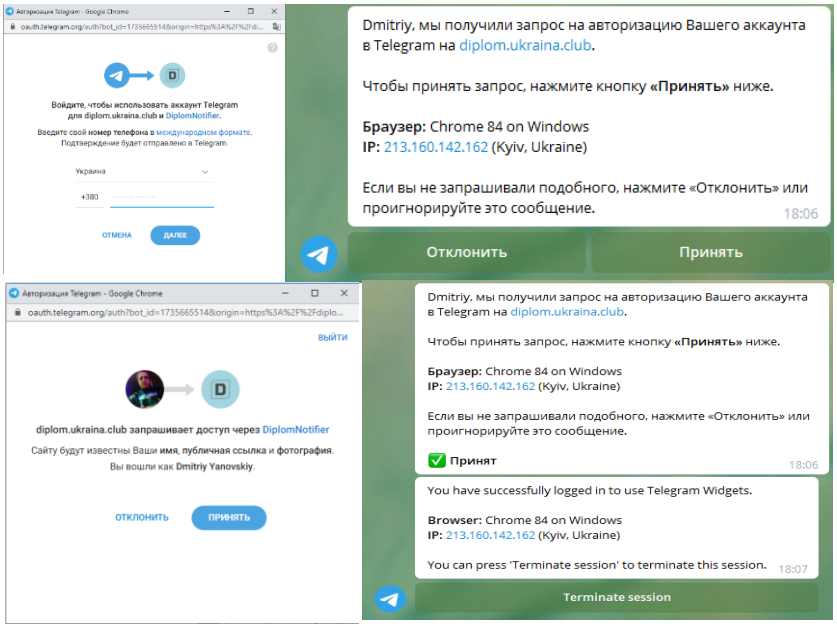


Рисунок 4.3 – Авторизація до допомогою Telegram

Після успішної авторизації, користувач потрапляє на сторінку свого профілю. Інтерфейс зображено на рис. 4.4.

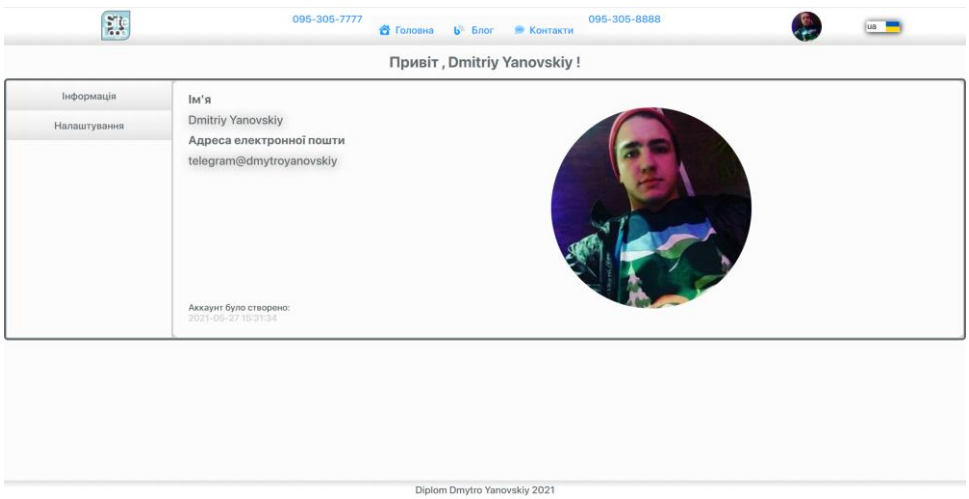


Рисунок 4.4 – Інтерфейс профілю користувача

Редагувати свій профіль користувач може за допомогою сторінки «Налаштування профілю», рис 4.5.

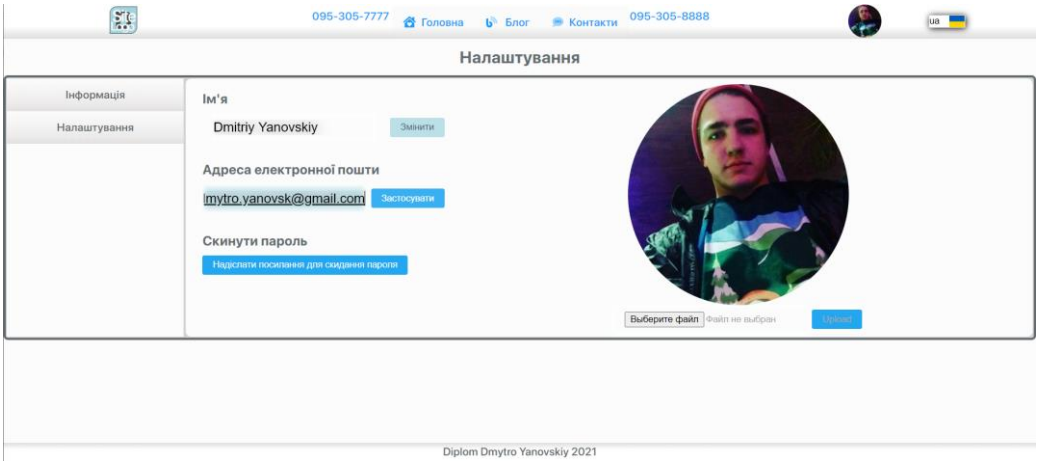


Рисунок 4.5 – Інтерфейс сторінки «Налаштування профілю»

Скориставшись меню сайту, можна відкрити блог та користуватись пагінацією. Дозволено переглядати як всі публікації, так і відібрані за категорією. Приклад відображення блогу зображено на рис 4.6.

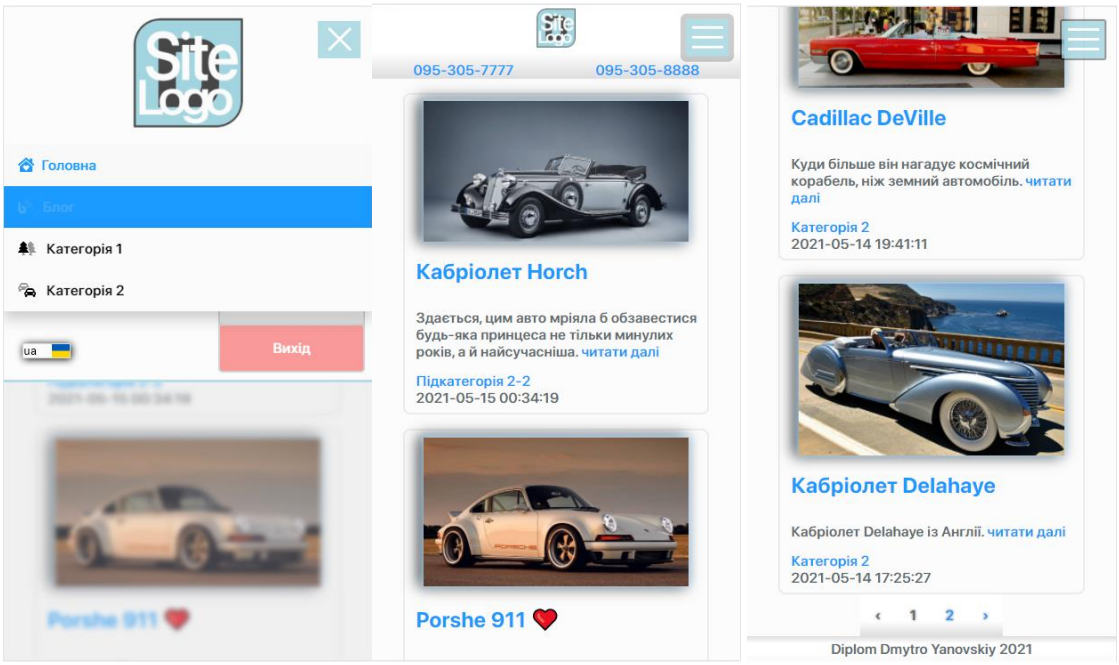


Рисунок 4.6 – Блог сайту

При натисканні на пост відкривається стаття з головним контентом, інформацією про автора та головне - рядком breadcrumb, який допомагає користувачу з маршрутизацією на сайті. Також відображаються коментарі, де авторизований користувач може залишати повідомлення, рис. 4.7.

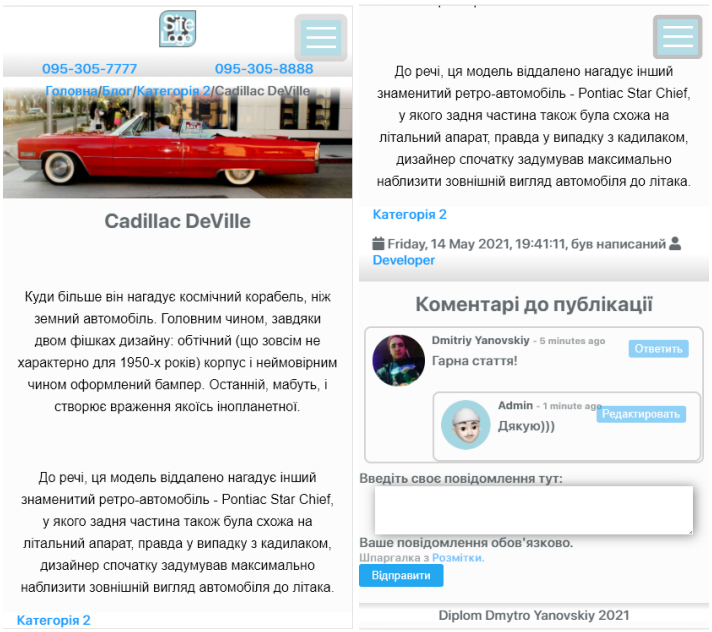


Рисунок 4.7 – Відображення коментарів

4.1 Маршрутизація адміністратора(клієнта)

Доступ до адмін панелі можна отримати авторизувавшись на сторінці /admin/login. Інтерфейс сторінки зображено на рис. 4.8.

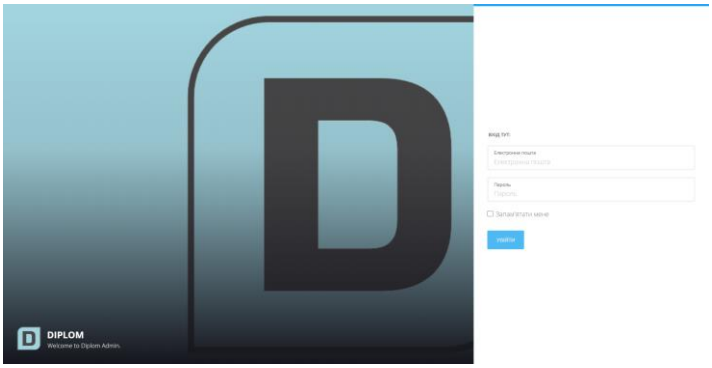


Рисунок 4.8 – Сторінка авторизації адміністатора

Авторизований на сайті адміністратор може скористатись кнопкою «Admin Panel» для швидкого переходу в адмін панель. Інтерфейс адміністратора на сайті зображено на рис 4.9.

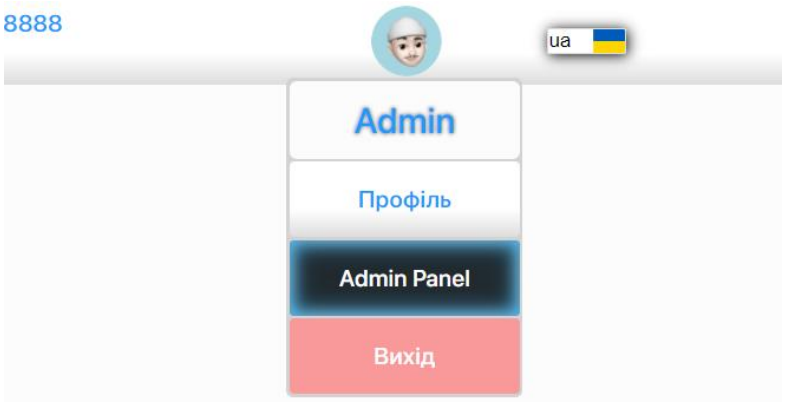


Рисунок 4.9 - Інтерфейс адміністратора

В адмін панель адміністратор отримує доступ до всіх необхідних пунктів меню та функцій для продуктивної роботи. Інтерфейс адмін панелі зображено на рис. 4.10.

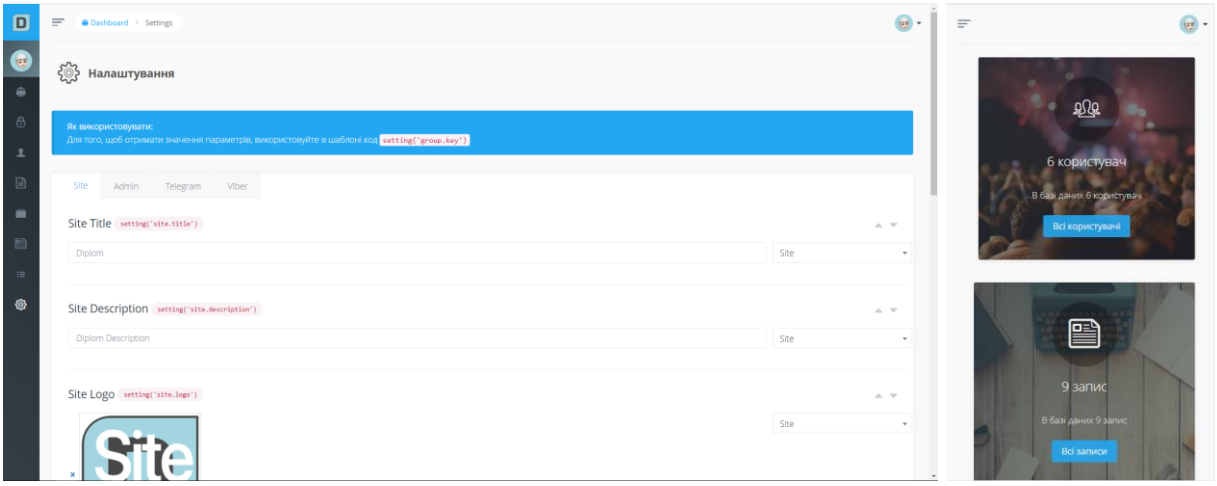


Рисунок 4.10 - Інтерфейс адмін панелі

Для редагування меню на сайті, адміністратору потрібно перейти на сторінку «Редактор меню», із запропонованих обрати меню для сайту і далі можна змінювати меню згідно своїх потреб. Інтерфейс редактора меню зображено на рис. 4.11. Переклад кожного елементу налаштовується власноруч для кожної мови.

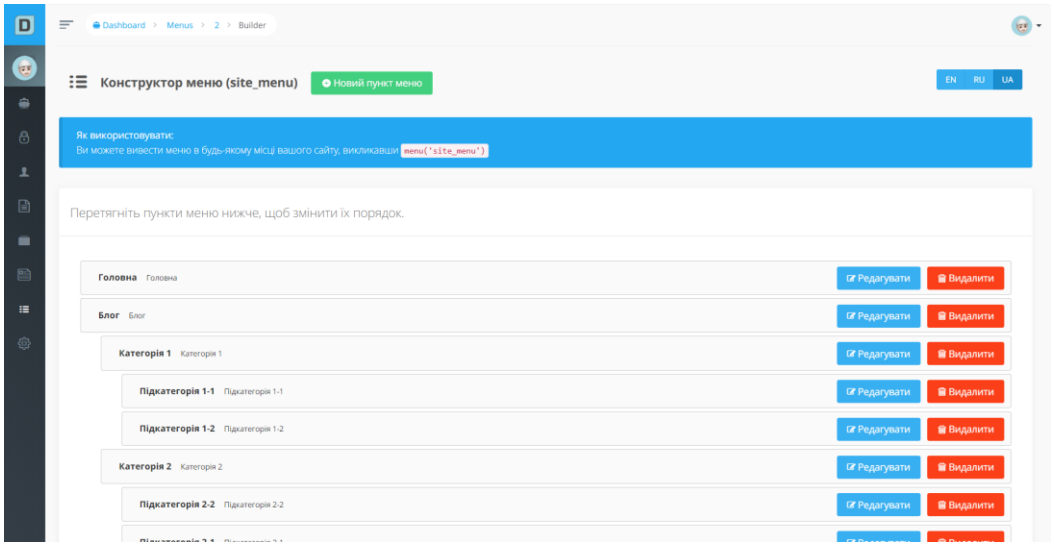


Рисунок 4.11 – Редагування меню

При редагуванні посту, адміністратор отримує зручний інтерфейс, рис. 4.8, та можливість сповіщати користувачів соціальних ботів про нову публікацію на сайті, рис 4.9. Якщо активувати сповіщення в соц. мережі, отримаємо результат зображений на рис. 4.10 та рис. 4.11.

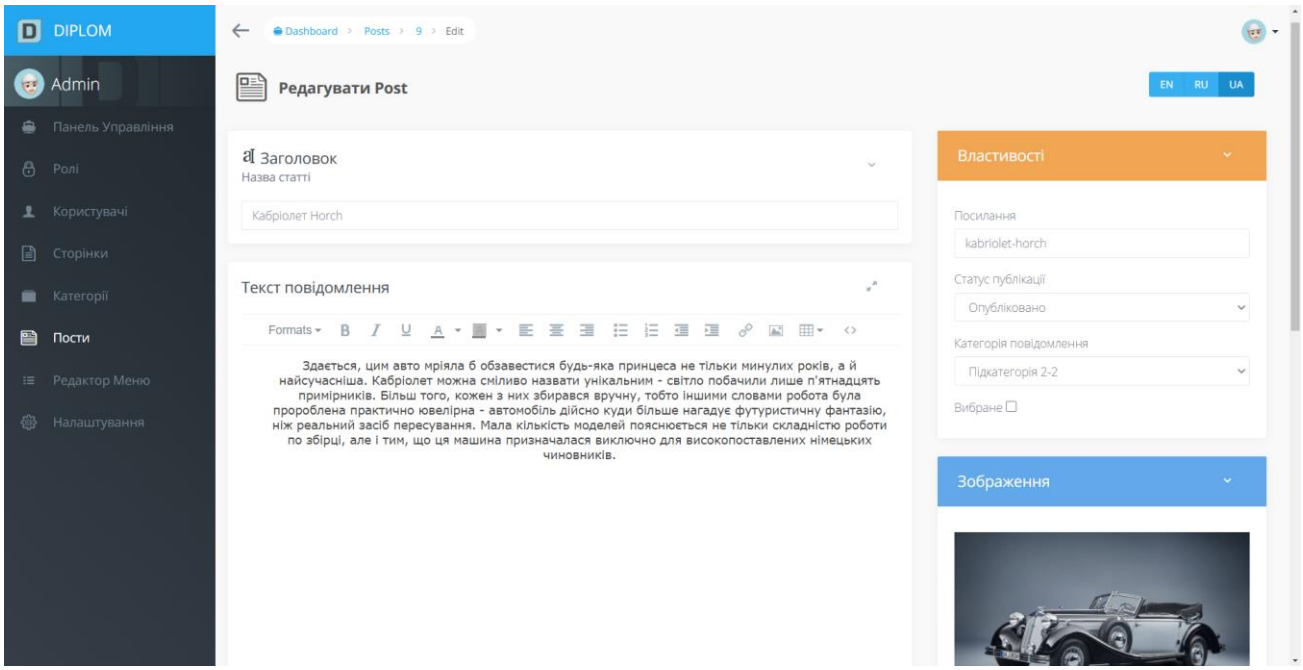


Рисунок 4.8 – Редагування посту

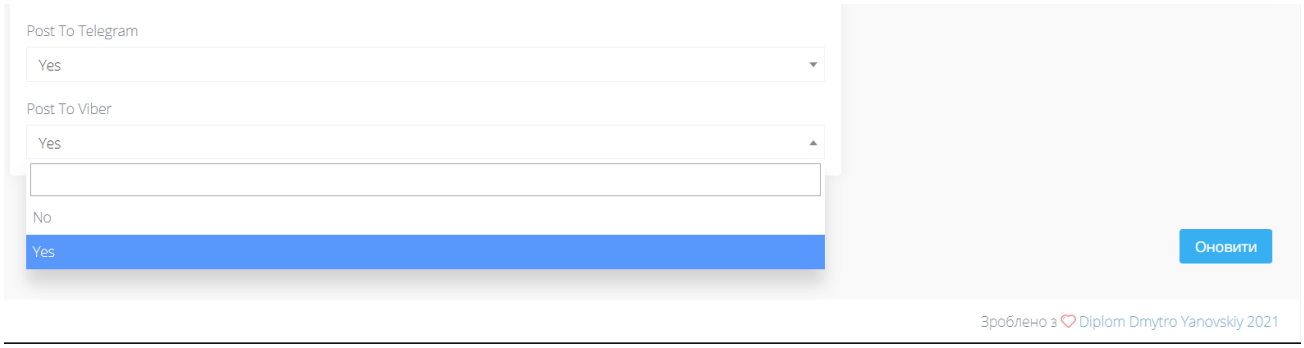


Рисунок 4.7 – Вибір соціальних мереж для сповіщення

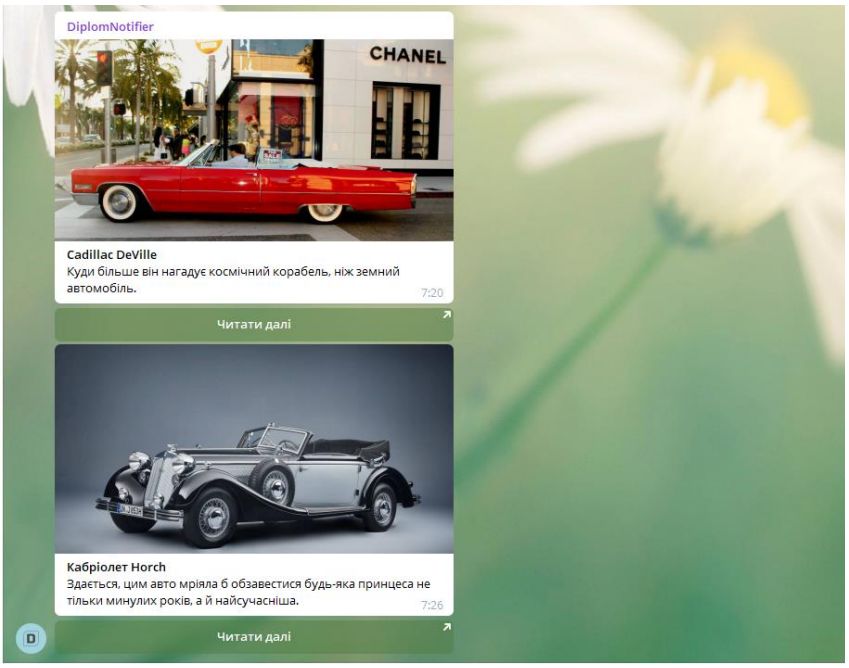


Рисунок 4.8 – Сповіщення Telegram

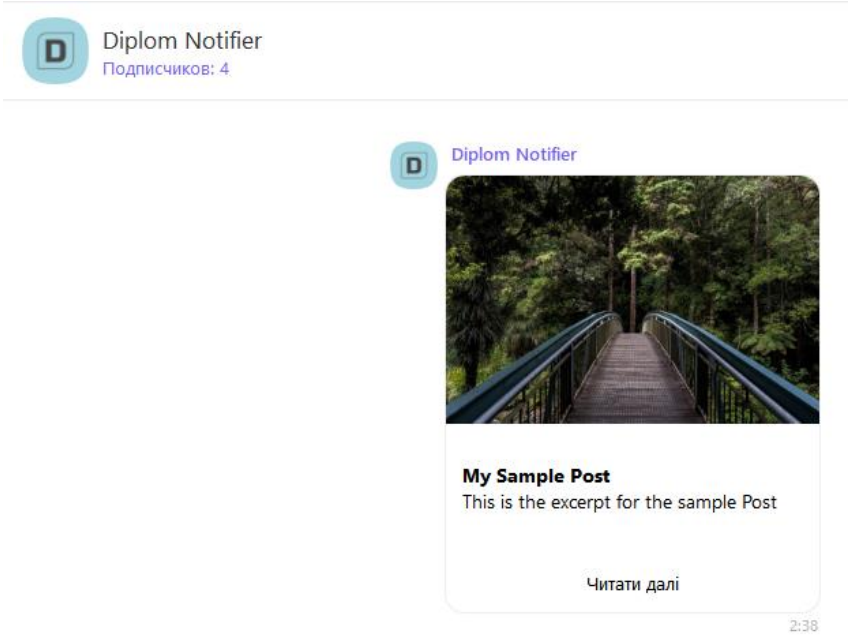


Рисунок 4.9 – Сповіщення Viber

ВИСНОВКИ

У дипломній роботі розроблялась платформа для створення адаптивних сайтів на базі фреймворку Laravel.

1. Проаналізувавши актуальність проблеми та попит на ринку, було зроблено висновок про необхідність створення платформи для створення адаптивних сайтів з дотриманням усіх вимог SEO.
2. Досліджено аналоги – їх недоліки та переваги, побудувавши порівняльну таблицю, визначено вимоги до платформи, маршрутизацію додатку, дизайн.
3. Враховані побажання більшості користувачів та функціонал реалізовано в застосунку.
4. Після аналізу інструментів phpMyAdmin, PHP, CSS, Laravel, OpenServer було вирішено використовувати цей пакет рішень, оскільки він повністю покриває необхідні потреби.
5. Встановлено попит функцій, що виконує платформа. Головною перевагою якої є висока швидкість завантаження сторінок та можливість публікувати нові пости в соціальні мережі автоматично.
6. Виконано тестування застосунку на ПК та телефоні. Визначено, що сайт вірно відображається на екранах різних розмірів.

СПИСОК ЛІТЕРАТУРИ

1. wix.com [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Wix] – Режим доступу : https://www.wix.com/.
2. tilda.cc [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Tilda] – Режим доступу : https://www.tilda.cc/.
3. ucoz.ua [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [uCoz] – Режим доступу : https://www.ucoz.ua/.
4. blogger.com [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Blogger] – Режим доступу : https://www.blogger.com/.
5. Вся статистика інтернету на 2020 рік [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [WebCanape] – Режим доступу : https://www.web-canape.ru/business/internet-2020-globalnaya-statistika-i-trendy/
6. PhpStorm [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [PhpStorm] – Режим доступу : https://www.jetbrains.com/phpstorm/.
7. Atom [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Atom] – Режим доступу : https://www.atom.io/.
8. SublimeText [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [SublimeText] – Режим доступу : https://www.sublimetext.com/.
9. Denwer [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Denwer] – Режим доступу : https://www.denwer.ru/.
10. Open Server Panel [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [OpenServer] – Режим доступу : https://www.ospanel.io/.
11. Wordpress [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Wordpress] – Режим доступу : https://uk.wordpress.org/.
12. Symfony [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Symfony] – Режим доступу : https://www.symfony.com/.
13. Laravel – The PHP Framework [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Laravel] – Режим доступу : https://www. laravel.com/.

14. Black Dashboard Laravel [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Creative Tim] – Режим доступу : https://www.creative-tim.com/product/black-dashboard-pro-laravel.
15. Material Dashboard Laravel [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Creative Tim] – Режим доступу : https://www.creative-tim.com/product/material-dashboard-laravel.
16. Voyager – The Missing Laravel Admin [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Control Group] – Режим доступу : https://voyager.devdojo.com/.
17. Lucid Software [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [Lucid Software] – Режим доступу : https://www.lucid.co/.
18. UML - [Електронний ресурс] : [Інтернет портал]. - Електронні дані. - [ЗНУ] – Режим доступу : http://sites.znu.edu.ua/webprog/lect/1238.ukr.html.

Додаток А

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА ПЛАТФОРМИ ДЛЯ СТВОРЕННЯ АДАПТИВНИХ САЙТІВ НА БАЗІ ФРЕЙМВОРКУ LARAVEL

Виконав: студент 4 курсу, групи ПД-41
Яновський Д.А.
Науковий керівник: к.т.н. доцент
Негоденко О.В.

Київ 2021

АНАЛІЗ ІСНУЮЧИХ АНАЛОГІВ

Назва конструктору	Переваги	Недоліки
Wix	1. Підтримка ручного та автоматичного перекладу. 2. Адаптивний дизайн.	1. Швидкість завантаження сторінок. 2. Влаштована реклама навіть з платними тарифами.
Tilda	1. Простота використання. 2. Можливість експорту коду.	1. Немає підтримки багатьох мов. 2. Відсутність функціоналу перевірки знань.
uCoz	1. Можливість вносити корективи в CSS файл.	1. Застарілий дизайн. 2. Немає підтримки багатьох мов.
Blogger	1. Влаштовані Google сервіси. 2. Швидке завантаження сторінки. 3. Безкоштовний.	1. Занадто простий та малофункціональний інтерфейс. 2. Відсутність змоги маштабувати свій сайт за потреби.

МЕТА, ОБ’ЄКТ, ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: розробка платформи з можливістю швидкого створення бази для сайту, а також для подальшого маштабування без зайвих фінансових витрат.

Об’єкт дослідження: створення та розгортання сайтів з дотриманням усіх вимог SEO.

Предмет дослідження: – платформа для створення адаптивних сайтів.

ТЕХНІЧНЕ ЗАВДАННЯ

Розробити платформу для створення адаптивних сайтів, яка має виконувати наступні вимоги:

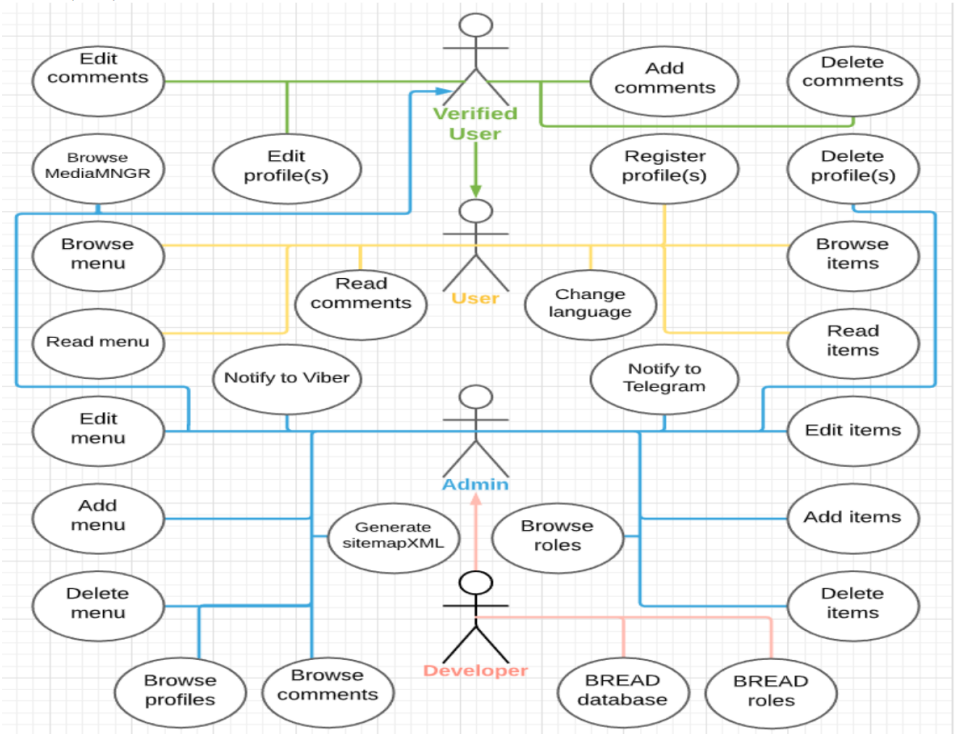
1. Швидке завантаження сторінок.
2. Можливість керувати ролями користувачів.
3. Авторизація за допомогою сторонніх сервісів.
4. Надсилання сповіщення в соціальні боти.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

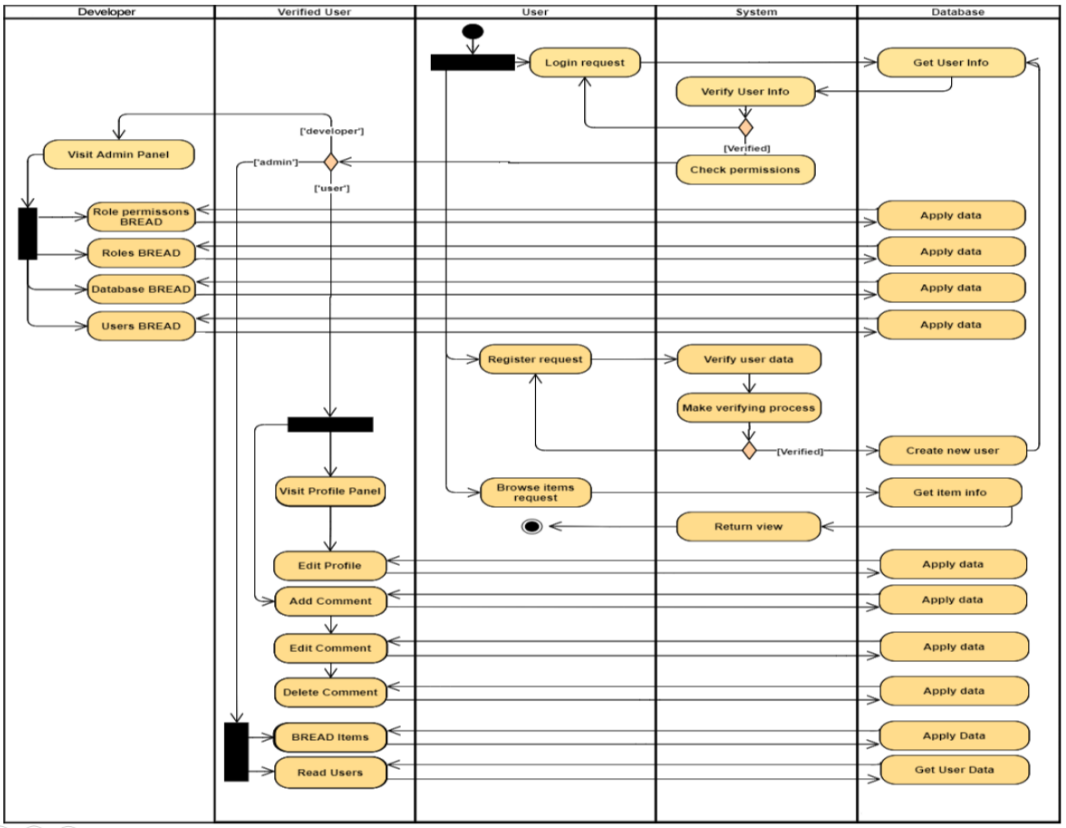
Lucid

Open Server

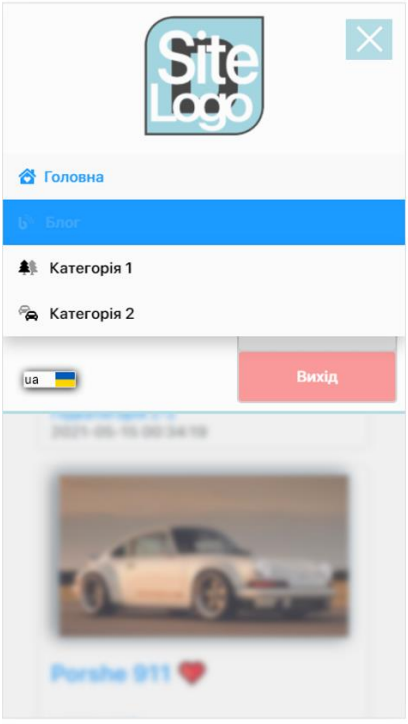
ДІАГРАМА ВИКОРИСТАННЯ

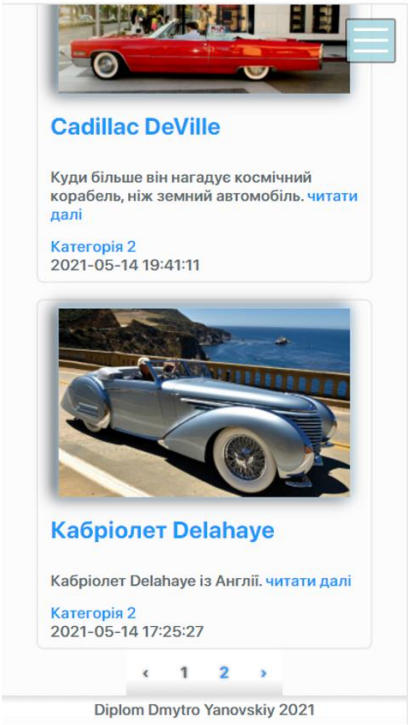


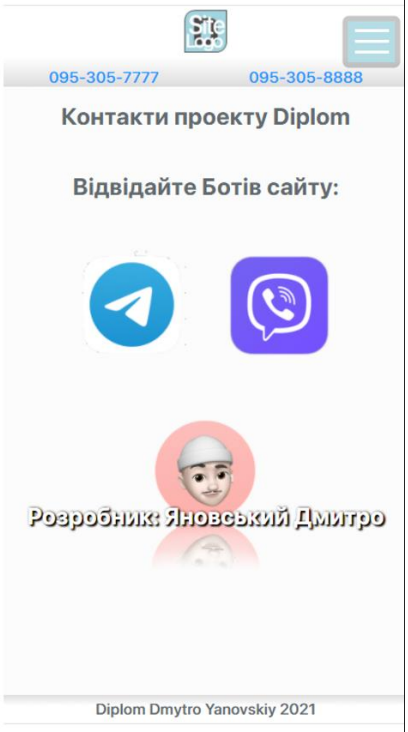
ДІАГРАМА ДІЯЛЬНОСТІ



САЙТ







ВИСНОВКИ

На основі результатів дослідження розроблено платформу для створення адаптивних сайтів, що відповідає усім вимогам SEO.
Сайт зручно використовувати на будь-якій діагоналі екрану.

1. Проаналізовано аналоги.
2. Досягнута швидкість завантаження сторінки рівна ~0.8с.
3. Проаналізовано інструменти для розробки платформи.
4. Встановлено технічне завдання розроблюваної платформи.
5. Проаналізовано маршрутизацію та інтерфейс додатку.
6. Виконано тестування додатку.

АПРОБАЦІЇ

Результати дослідження бакалаврської роботи апробовані:

1. На всеукраїнській науково-технічній конференціїї: «Застосування програмного забезпечення в інфокомунікаційних технологіях»
2. На всеукраїнскій науково-технічній конференції «Сучасні інфокомунікаційні технології»