Вінницький національний технічний університет
факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

БАКАЛАВРСЬКА ДИПЛОМНА РОБОТА

на тему:

Адмін-панель веб-сайту інтернет магазину

ПОЯСНЮВАЛЬНА ЗАПИСКА

Виконав студент 4 курсу, групи 1КІ-18б
спеціальності 123 — Комп'ютерна інженерія
Швець С.О.

Керівник к.т.н., доц. каф. ОТ
Черняк О.І.

Рецензент д.т.н., проф. каф. МБІС
Яремчук Ю.С.

Допущено до захисту
к.т.н., проф. Азаров О.Д.
22 червня 2022 р.

Вінниця 2022

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітньо-кваліфікаційний рівень бакалавр

Спеціальність 123 — «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач кафедри
обчислювальної техніки
д.т.н., проф. Азаров О. Д.
"02" 02 2022року

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Швецю Євгенію Олександровичу

1 Тема роботи «Адмін-панель веб-сайту інтернет магазину», керівник роботи Черняк Олександр Іванович , к.т.н., доц.,затверджені наказом вищого навчального закладу від "24" 03 2022 року №66.

2 Строк подання студентом роботи 23.02.2022.

3 Вихідні дані до роботи: розробка технології адміністрування веб-сайту, яка б надала можливість налаштування контенту для покращення швидкодії побудови та пошукової індексації сторінок,

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):вступ, аналіз сучасних методів розробки,варіантний вибір засобів для розробки програмного забезпечення, програмна реалізація, тестування розробленого програмного забезпечення.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): технічне завдання.

6 Консультанти розділів роботи наведені в таблиці 1.

Таблиця 1 — Консультанти роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Спеціальна частина	Черняк О.І. доцент кафедри ОТ		

7 Дата видачі завдання «24» березня 2022 р.

8 Календарний план наведено в таблиці2.

Таблиця 2— Календарний план

№з/п	Назва етапів дипломної Роботи	Строк виконання етапів проекту (роботи)	Пр
1	Постановка задачі роботи	09.09.2021	вико
2	Пошук матеріалів по технологіям розробки віконних додатків	12.09.2021 — 08.10.2021	вико
3	Структурне проектування додатку організації та оптимізації часу та робочого процессу	09.10.2021 — 06.11.2021	вико
4	Обгрунтування та вибір засобів реалізації системи	07.11.2021 — 25.11.2021	вико
5	Розробка головного вікна, методів створення календарю та подій	26.11.2021 — 22.12.2021	вико
6	Підготовка матеріалів та розробка алгоритму збереження та виведення списку подій	22.12.2021 — 04.02.2022	вико
7	Оформлення пояснювальної записки та ілюстративного матеріалу	04.02.2022 — 03.03.2022	вико
8	Перевірка якості виконання бакалаврської роботи та усунення недоліків	28.03.2022	вико

Студент Швець

Керівник роботи Черняк

АНОТАЦІЯ

Швець Є.О. Адмін-панель веб-сайту інтернет магазину. Бакалаврська кваліфікаційна робота зі спеціальності 123 — Комп'ютерна Інженерія, Вінниця: ВНТУ, 2022.

У бакалаврській дипломній роботі було розглянуто технології адміністрування веб-сайту, яка б надала можливість налаштування контенту для покращення швидкодії побудови та пошукової індексації сторінок. Дана робота присвячена розробці інформаційної технології адміністрування веб-сайтів. В роботі був виконаний аналіз найбільш сучасних методів адміністрування веб-сайтів. Перевірка працездатності системи протестована шляхом практичного тестування системи.

Ключові слова: адміністрування, веб-сайт, пошукова оптимізація.

ABSTRACT

Shvets E.O. Admin panel of the online store website. Bachelor's thesis in the specialty 123 — Computer Engineering, Vinnytsia: VNTU, 2022.

The bachelor's thesis covered website administration technologies that would allow you to customize the content to improve the performance and search indexing of pages. This work is devoted to the development of information technology for website administration. The analysis of the most modern methods of website administration was performed in the work. The system is tested by practical testing of the system.

Keywords: administration, website, search engine optimization.

ЗМІСТ

ВСТУП……8

1 АНАЛІЗ СУЧАСНОГО СТАНУ СИСТЕМ WEB-САЙТІВ……...10

1.1 Аналіз предметної області……..……10

1.2 Аналіз видів веб-сайтів...……..……..13

1.3 Аналіз існуючих систем керування вмістом…….....17

1.4 Постановка задачі…….....17

2 **ДОСЛІДЖЕННЯ ФАКТОРІВ ЕФЕКТИВНОСТІ ВЕБ-САЙТІВ**….....17

2.1 Метод для побудови гіпертекстової розмітки на сервері……..……..17

2.2 Проектування методу роботи з пошуковою оптимізацією проектованої технології моделювання…….....21

3 ПРОЕКТУВАННЯ ПАНЕЛІ АДМІНІСТРАТОРА З МОЖЛИВІСТЮ МОДЕЛЮВАННЯ ВЕБ- ОРІЄНТОВАНОЇ СИСТЕМИ…….....26

3.1 Вибір технологій розробки……..…..…….……26

3.2 Проектування інформаційної технології моделювання веб-сайту...30

3.3 Розробка інформаційної технології моделювання web-орієнтованої системи……..……..……..……33

ВИСНОВКИ…..……..……38

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ...……39

ДОДАТОК А Технічне завдання…… …… …..…….……43

ДОДАТОК Б Програмні коди index.html ……...…….……46

ДОДАТОК В Програмні коди JS…… … …..…….....55

ДОДАТОК Г Програмні коди CSS …… ...…….....60

					08-23.БДР.017.00.000 ПЗ		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Швець Є.О.			Адмін-панель веб-сайту інтернет магазину Пояснювальна записка	Літ.	Арк. 6 / Аркушів 70
Перевір.		Черняк О.І.					
Реценз.		Яремчук Ю.Є.				ВНТУ, гр. 1КІ-18б	
Н. Контр.		Швець С.І.					
Затверд.		Азаров О.Д.					

ДОДАТОК Д Програмні коди singin.html …….…….67

ДОДАТОК Е Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень……..……...70

ВСТУП

Кваліфікаційна робота бакалавра присвячена розробці інформаційної технологіїмоделювання та адміністрування web-сайтів.

Актуальність теми полягає в тому, що тяжко уявити собі людину яка ніколи не зустрічалась з різними інтернет ресурсами. Веб-сайти супроводжують людину на кожному її кроці. Вони використовуються для виконання великої кількості задач, та мають громіздкий діапазон застосування. Для успішної веб-орієнтованої системи дуже важливо мати потужну систему для контролю вмісту цієї системи. Такі системи існують вже доволі давно також їх називають Системами керування контентом. Серед найкращих систем керування вмістом слід відзначити Wordpress, Joomla, WooComerce. Ці системи мають досконалий функціонал, але їх дуже тяжко персоналізувати, тобто добавляти такий функціонал, який відноситься тільки для конкретного бізнесу. Створення кожної з таких систем потребує застосування величезної кількості ресурсів та вирішення великої кількості важливих проблем. Серед них можливо виділити такі:

— перевірка ринку та цільової аудиторії;

— побудова бізнес моделі роботи системи;

— створення дизайну системи;

— вибір технологій розробки;

— розробка програмного продукту;

— запровадження продукту;

— підтримка та оновлення.

На противагу цим великим та важким системам було розроблено невелику систему, яка буде мати величезні можливості для доповнення її новим функціоналом. Така система повинна включати в себе функціонал для створення нових сторінок, роботи з навігацією на сайтах, роботи з даними необхідними для пошукової оптимізації сайту. Крім вмісту сайт повинен мати високу швидкість завантаження, чого досить тяжко досягнути для повністю динамічного контенту.

Об’єкт дослідження — процес моделювання веб-сайту.

Предмет дослідження — це моделі, методи, підходи та засоби для адміністрування web-сайтів.

Метою бакалаврської роботи є розробка технології адміністрування web-сайту, яка б надала можливість налаштування контенту для покращення швидкодії побудови та пошукової індексації сторінок.

Для досягнення поставленої мети були визначені такі **задачі**:

— дослідили види та основні принципи створення web-сайтів;

— дослідити фактори та способи оптимізації, що впливають на пошукову індексацію сторінок сайту;

— спроектувати та розробити технологію адміністрування web-сайту з можливістю керування процесом побудови сторінки на сервері та налаштуванням мета-інформації, що необхідна для пошукової індексації.

Практичне значення одержаних результатів полягає в розробці технології адміністрування веб-сайтів створено програмну реалізацію у вигляді панелі адміністратора з можливістю керування процесом побудови сторінки та налаштуванням мета-інформації, що дає можливість оптимізувати процес пошукової індексації.

1 АНАЛІЗ СУЧАСНОГО СТАНУ СИСТЕМ МОДЕЛЮВАННЯ WEB-САЙТІВ.

1.1 Аналіз предметної області

Використання веб-сайтів у нашій країні майже не нова, і зараз більшість людей активно користуються кількома, а то й десятком таких систем. Ринок активно розвивається і розширює свої горизонти. Кожен бізнес повинен використовувати інформаційну систему для належного функціонування. Ці системи можуть виконувати найрізноманітніші функції, від управління внутрішніми процесами компанії до просування бренду компанії серед користувачів Інтернету. Програмне забезпечення для управління внутрішніми процесами є досить унікальним і, як правило, призначене для конкретних завдань і конкретних компаній. Для рекламних цілей підприємства часто використовують одночасно кілька ресурсів, одним з яких є особистий веб-сайт.

Сайти компаній можливо умовно розділити на 3 категорії: прості сайти візитки, повноцінні багатосторінкові сайти, інтернет застосунки. Сайт-візитка зазвичай являє собою односторінковий сайт з короткою інформацією та формою для зв’язку. Цей варіант блискуче пасує для малого бізнесу. Для середнього та великого бізнесу краще личить розробка повноцінного сайту, котрий буде з складатись із багатьох сторінок(головна, про нас, блог, новини, вакансії) та буде мати панель адміністратора для управління контентом та іншими функціями. Третя група поєднує рекламне та функціональне призначення. Іншими словами, цей програмний підхід є послугою, яку надає бізнес. Таке програмне забезпечення може включати соціальні мережі, рекламні веб-сайти, інтернет-магазини. Кожен тип програмного забезпечення необхідно розробляти з використанням різного підходу. Щоб розробити сайт-візитку, скористайтеся онлайн-конструктором або зверніться до веб-студії для розробки.

Повноцінний веб-сайт можна розробити за допомогою системи управління контентом, такі системи поширені, але не гарантують належної якості продукту, в свою чергу, може бути залучена команда розробників, які створять надійний

продукт, але для цього знадобиться більше часу та гроші. Для третього типу програмного забезпечення все одно має бути залучена величезна команда розробників і багато фінансових ресурсів.

Крім сайтів, можна повторно використовувати інші типи мережевих систем. Веб-сайт — це будь-який програмний продукт, який для належної роботи має бути інтегрований з Інтернетом. До веб-систем належать: веб-сайти, мобільні додатки, настільні програми.

Більшість веб-сайтів [1] мають прихований від звичайних користувачів розділ, який називається панеллю адміністратора. Панель адміністратора сайту — це прихована від відвідувача частина, необхідна для управління ресурсами. Тут адміністратори працюють зі структурою та змістом сайту. Панель адміністратора або адміністратор дозволяє створювати, змінювати та видаляти сторінки та розділи, текст, медіаконтент у межах, передбачених системою керування вмістом (CMS), яку ви використовуєте. Крім того, що адміністратор повинен бути швидким і легким, до нього потрібна ще одна вимога — мати можливість користуватися панеллю користувача без навичок програмування і без втручання в програмну частину сайту.

Хоча панель адміністратора невидима для користувача, її дизайн настільки ж важливий, як і створення інтерфейсу. Адже від ефективності роботи адміністратора залежить ефективність роботи панелі. Складна, заплутана й непродумана панель адміністратора, яка ігнорує користувацький досвід, не тільки ускладнює та коштує персоналу, але й збільшує ймовірність помилок, додаючи додаткові витрати на їх усунення.

1.2 Аналіз видів веб-сайтів

Як зазначалося вище, веб-сайт — це система, яка вимагає постійного підключення до Інтернету. Існує кілька поширених типів систем:

— веб-сайт;

— мобільний додаток;

— настільний додаток зазначено в рисунку 1.1.

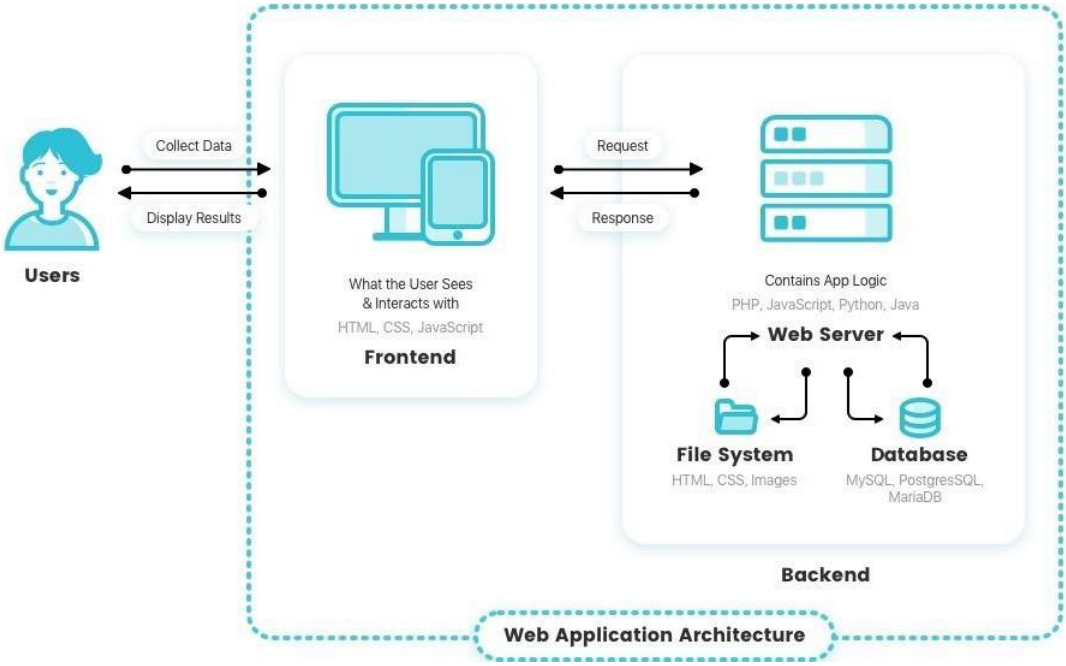


Рисунок 1.1 — Схема роботи веб-сайту

Створення сайту — надзвичайно складний процес. Етапи створення сайту, приведені нижче. Розробка структури сайту:

— визначити вихідні дані для сайту;

— визначення зовнішнього вигляду та функціональних вимог;

— формування структури сайту — розділ меню;

— розробка концепцій дизайну:

— створити макет дизайну для домашньої сторінки сайту;

— затвердження концепції дизайну — макет головної сторінки;

— створення внутрішніх сторінок веб-сайту та виявлення змін у дизайні внутрішніх сторінок;

— html-верстка, проектування та створення внутрішніх сторінок: розробка наповнення внутрішніх сторінок;

— розробка додаткових сторінок (карта сайту, результати пошуку тощо), оптимізація зображення;

— програмування: визначте завдання програмування;

— розробляти структури баз даних, писати адміністративні скрипти.

Другий метод на даний момент є основним методом створення складних,

сучасних веб-сайтів, порталів, веб-додатків. Це один із способів використання CMS. Вікіпедія дає таке визначення. CMS — це система керування вмістом (CMS) зображена на рисунку 1.2, комп'ютерна програма або система для забезпечення та організації спільних процесів для створення, редагування та керування текстовими та мультимедійними документами (вміст або вміст). Цей вміст часто розглядають як неструктуровані дані для предметних завдань у порівнянні зі структурованими даними, якими зазвичай керують бази даних. Звичайно, установка CMS була виконана на вибраному хості. Для цього потрібен принаймні доступ до FTP і дозволи MySQL.

Тому розділення дизайну та контенту є основною відмінністю між динамічними та статичними сайтами. На цій основі структура веб-сайту може бути додатково вдосконалена, наприклад, визначення різноманітних функцій користувача та автоматизація бізнес-процесів, а найголовніше — контролювати зміст вмісту веб-сайту.

Рисунок 1.2 — Система управління Web-контентом

1.3 Аналіз існуючих систем керування вмістом

Враховуючи поставлені вище завдання найбільш оптимальним засобом для розробки такого роду – застосування мови розмітки документів у Всесвітній павутині - HTML , формальної мови опису зовнішнього вигляду документа, написаного з використанням мови розмітки - CSS, прототипноорієнтованого сценарного типу мови програмування JavaScript, скриптової мови програмування загального призначення PHP з використання баз данних MySQL та системи управління вмістом - CMS. Існує безліч систем, на яких можна створити сайт будь-якої складності. Важливо грамотно вибирати CMS, виходячи з суті проекту

Системи керування вмістом [20] [21] — це програмне забезпечення, яке допомагає користувачам створювати, керувати та змінювати вміст веб-сайту, не вимагаючи спеціальних технічних знань. Простіше кажучи, система управління контентом — це інструмент, який допомагає створити веб-сайт, не писуючи весь код з нуля. Розробити власну систему управління контентом. Розглянуто найпопулярніші аналогові системи. Такими системами є: WordPress, Joomla.

WordPress — це система керування вмістом із відкритим вихідним кодом, яка широко використовується для створення веб-сайтів через простоту встановлення та використання. Першу версію WordPress створили Метт Малленвег і Майк Літтл у 2003 році. Вона починалася як проста платформа, розроблена для людей, які хотіли створити базовий блог і опублікувати його в Інтернеті. Однак з часом він став гнучким і потужним інструментом, який можна використовувати для створення веб-сайтів практично будь-якого типу.

Переваги використання WordPress. WordPress — це абсолютно безкоштовна система керування вмістом у якій файли та бази даних цього веб-сайту повністю контролюються власником. Наявність підтримки розробників. Ядро системи часто оновлюється, додаючи нові модулі, віджети, хуки, а також покращуючи безпеку та швидкість. Мережа може відповісти майже на будь-яке запитання, яке може стосуватися цієї CMS. Тисячі плагінів. Тільки WordPress включає понад 1 000 000 безкоштовних плагінів для різних типів завдань. Зручність і простота адмін панелі. Панель адміністратора WordPress дуже проста у використанні. Навіть

недосвідчені люди можуть додавати нові розділи, сторінки, новини, зображення. Для складних сайтів - панель адміністратора адаптується до будь-яких завдань за допомогою плагіна - «Додаткові користувацькі поля». Швидко за допомогою правильного дизайну шаблону, увімкненого кешування та правильної оптимізації зображення та вмісту.

Недоліки використання WordPress приклад наведено на рисунку 1.3. Необхідно постійно контролювати сайт, його безпеку та правильну роботу. Встановлення плагінів для автоматизації роботи не позбавляє від необхідності постійного моніторингу.Багато плагінів низької якості, які можуть конфліктувати після встановлення. Через це швидкість завантаження сайту буде не дуже високою і погіршить якість сайту в очах пошукових систем. Це, у свою чергу, може значно погіршити позицію сайту в результатах пошуку. WordPress не є конструктором веб-сайтів, як Wix, Shopify, Weebly або Jimdo. Отже, щоб створити сайт з унікальною архітектурою та дизайном, вам потрібно попрацювати над усіма його сторінками з нуля, перш ніж інтегрувати їх із CMS WordPress, уразливості системи.

Joomla! — відкрита універсальна система управління контентом для публікації інформації в Інтернеті. Ідеально підходить для створення веб-сайтів малого та великого бізнесу, онлайн-порталів, інтернет-магазинів, сайтів спільнот та особистих сторінок. До особливостей Joomla належать гнучкі інструменти керування обліковими записами, інтерфейс керування медіа, підтримка багатомовних сторінок, система керування рекламою, адресна книга користувачів, голосування, вбудований пошук, класифікація посилань та облік кліків, редактор WYSIWYG, система шаблонів, підтримка меню зображено на рисунку 1.4

Переваги використання Joomla, а саме універсальність полягає у порівнянні з, наприклад, Drupal, Joomla є повноцінним інструментом для створення будь-якого сайту, незважаючи на його відносну простоту, проста у використанні, панель адміністратора проста та зручна, розділи та розширення, які потрібно налаштувати, легко знайти, простота установки, вбудована система кешу полягає в тому, що надсилає в кеш раніше відкриті веб-сторінки, що прискорює

завантаження сторінки, тому на сервері немає перевантаження, що покращує продуктивність SEO.

Недоліки використання Joomla полягають у багато розширень і шаблонів для Joomla є платними, немає технічної підтримки, слабкість цієї CMS — розширення завантажуються з сумнівних джерел, вони часто стають причиною злому. Багато веб-сайтів використовують ці системи моделювання, але веб-сайти, написані за допомогою цих систем, важко оновлювати, стежити за їх безпекою та оптимізувати їх для хорошої індексації в пошукових системах. Як альтернативу таким системам можна створити власну.

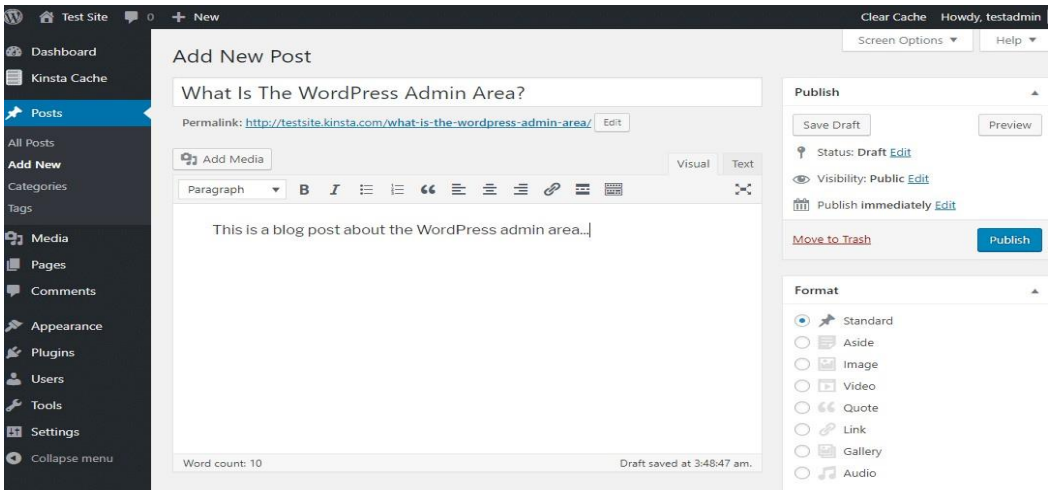


Рисунок 1.3 — Приклад адмін панелі сайту створеного з Wordpress

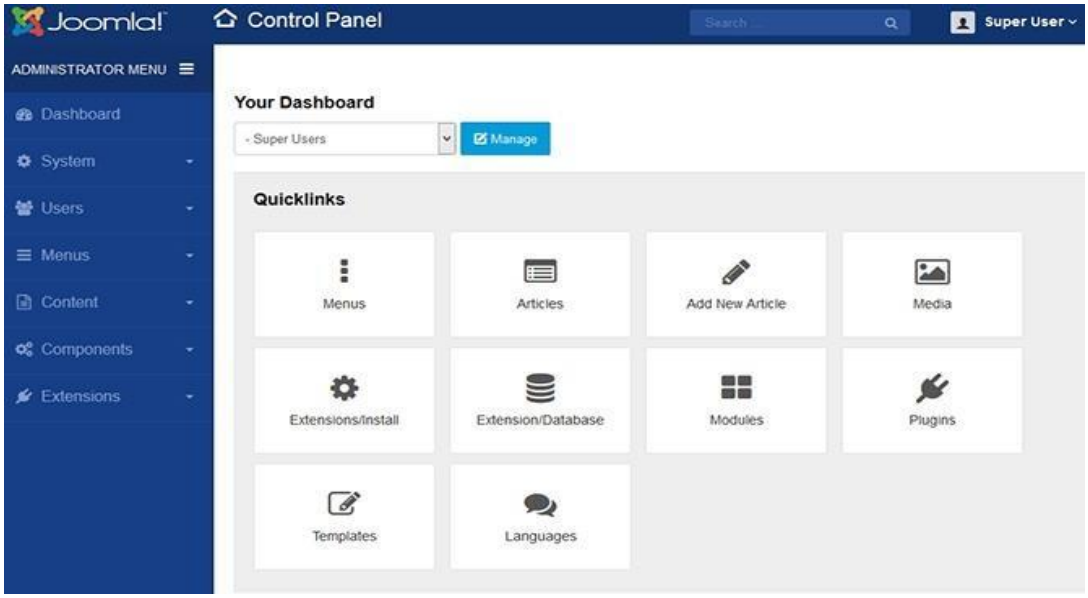


Рисунок 1.4 —Приклад адмін панелі сайту створеного з Joomla!

1.4 Постановка задачі

Метою бакалаврської роботи є розробка методів моделювання та керування веб-сайтами, які забезпечать можливість налаштовувати контент, здійснювати пошук інформації, необхідної для індексації сторінок, та забезпечувати швидкість роботи системи.

Для цього визначаються наступні завдання:

— вивчити процес побудови гіпертекстової розмітки на сервері;

— досліджувати шляхи оптимізації веб-сайту;

— дослідити фактори, які впливають на індексацію пошуку сторінок;

— розробка та розробка методів моделювання та управління.

Веб-орієнтована система, яка контролює процес створення сторінок на сервері та налаштовує метаінформацію, необхідну для індексації пошуку.

2 ДОСЛІДЖЕННЯ ФАКТОРІВ ЕФЕКТИВНОСТІ ВЕБ-САЙТІВ

2.1Метод для побудови гіпертекстової розмітки на сервері

Одним з найважливіших факторів ефективності мережевої системи є її оптимізація зображено на рисунку 2.1. Основні проблеми, які можуть вплинути на швидкість оптимізації, описані в Розділі 1.1. Щоб вирішити кожну проблему, при створенні динамічних сторінок на сервері слід розглянути загальні сценарії сервера та визначити, як застосовувати відомі рішення.

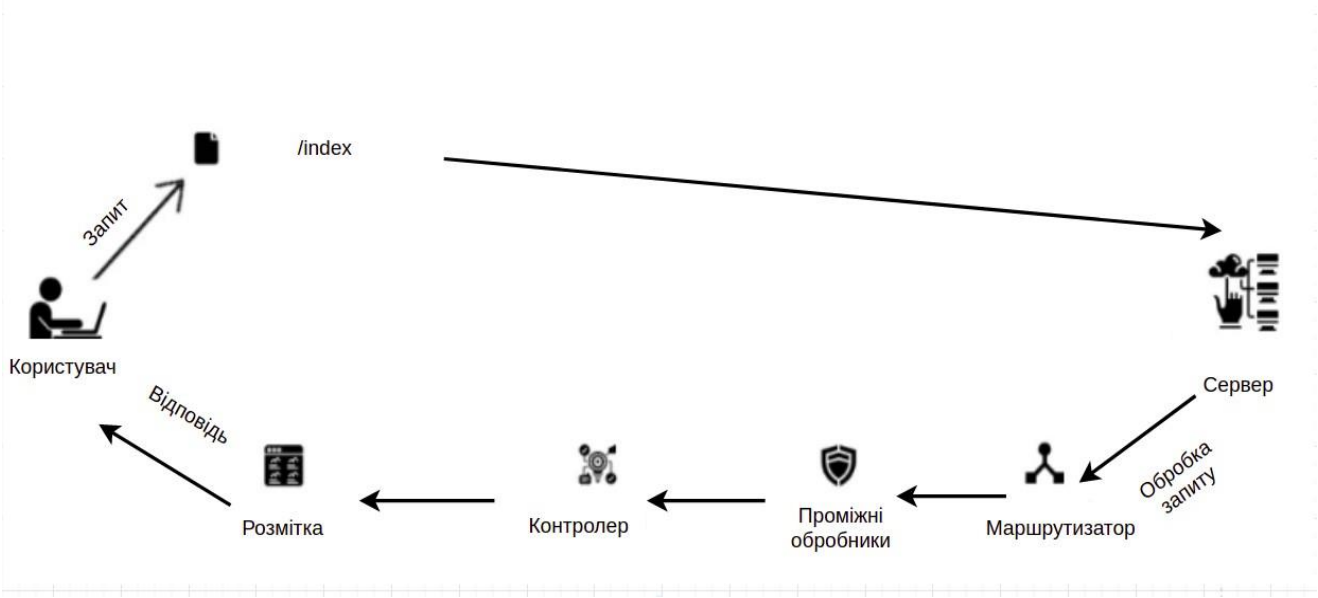


Рисунок 2.1 — Схема роботи веб-системи

Як зображено на рисунку 2.1, веб-система працює за таким алгоритмом:

— користувач відкриває браузер і вводить адресу сайту в поле пошуку;

— введена користувачем адреса перетворюється в IP-адресу сайту за

— допомогою можливостей системи доменних імен;

— сервер отримує запит користувача та надсилає результат обробки запиту у вигляді гіпертекстової розмітки;

— браузер користувача отримує відповідь від сервера і виводить результат на екран.

Це загальна схема веб-сайту, і вона не відображає процес, що відбувається на сервері. Для того, щоб зрозуміти, як будувати системи контролю вмісту, потрібно подумати про них більш детально.

У цьому випадку сервер можна розділити на наступні кроки:

— отримати вхідні дані;

— пошук логіки, що відповідає запиту;

— перевіряти вхідні дані;

— виконувати бізнес-логіку та запити до бази даних;

— замінити динамічні дані в гіпертекстовій розмітці;

— повернути результат клієнту.

Ця схема дуже проста і ідеально підходить для повноцінних гіпертекстових сторінок з деякими блоками динамічного вмісту. Однак, коли структура сторінок сайту не визначена заздалегідь, а сформована певними налаштуваннями, така схема набагато складніша.

На перший погляд використання редактора WYSIWYG зображеному на рисунку 2.2 у цьому випадку може здатися бажаним, але це не так. Такий редактор варто використовувати для дописів блогів, де стилізація контенту другорядна, але він не підходить для створення повноцінних сторінок. З таким редактором створення нової сторінки може стати набагато більш схожим на макет процесу, інша проблема полягає в тому, що вміст буде повністю статичним. Вміст можна редагувати, але не можна поєднувати з даними в базі даних. Тому вам потрібно знайти інший спосіб відтворення динамічного вмісту, щоб створити ефективну веб-систему.

HTML (англ. HyperText Markup Language — Мова розмітки гіпертекстових документів)(рис.2.2) — основана на SGML текстова мова розмітки, призначена для маркування документів, що містять текст, зображення, гіперпосилання, тощо. HTML-документи лежать в основі Веб, і відображаються із допомогою веб-браузерів. Разом із видимою інформацією, HTML-документи містять додаткові метадані, такі як, наприклад, мова тексту, автор документа, стислий підсумок. Мова розмітки розроблялась консорціумом W3C, остання версія — 5, очікується, що HTML буде замінена розширюваною мовою розмітки гіпертексту (XHTML) . Найпростіша веб-сторінка складається з текстових блоків, декількох рисунків, горизонтальних розмежувальних ліній та гіперпосилань.

Рисунок 2.2 — Логотип HTML

Більш складні веб-сторінки містять фрейми, елементи керування, динамічні ефекти та анімовані об'єкти. Крім свого стандартного застосування таблиці в html-мові також використовують, якщо потрібно розташувати якийсь текст чи об'єкт у певному місті на веб-сторінці

Основною ідеєю вирішення цієї проблеми є запозичення передньої частини фреймворка мови програмування JavaScript (рисунок 2.3). Ідея полягає в тому, щоб гіпертекстові блоки розмітки, які називаються компонентами. Ці компоненти наповнені динамічними даними та об’єднані в цілісну веб-сторінку.

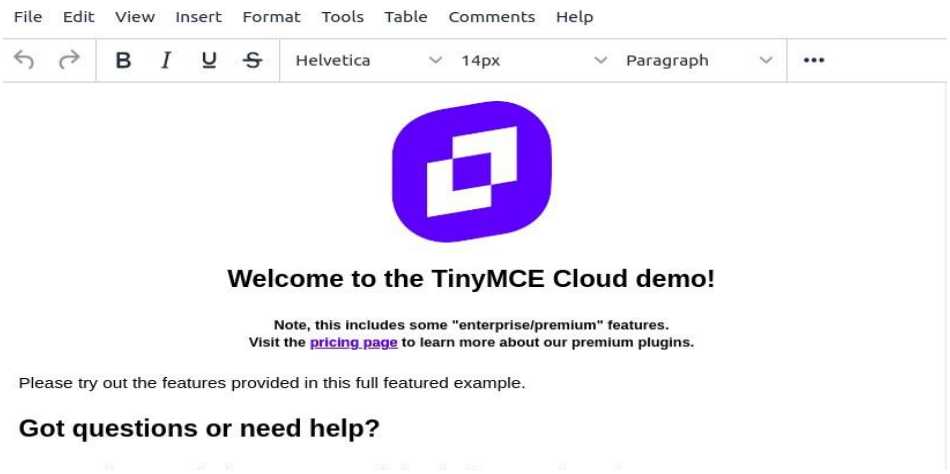


Рисунок 2.3 — Приклад WYSIWYG-редактором під назвою TinyMCE

Для досягнення цього завдання необхідно удосконалити процес побудови гіпертекстової розмітки веб-сторінок на сервері з використанням усіх відомих методів оптимізації.

Початок побудови сторінки полягає в отриманні інформації про блоки, які використовуються при моделюванні цієї сторінки. Отримавши цю інформацію, вам потрібно отримати інформацію про кожен блок з конфігураційного файлу. Ці файли описують запити до бази даних, необхідні для відтворення конкретних блоків, інформацію про власні стилі, наявність скриптів. Як тільки ви отримаєте цю інформацію, вам потрібно виконати запит до бази даних для кожного блоку паралельно, потім завантажити теги блоку та розмістити динамічні дані на місці. Отриману розмітку блоку потрібно об’єднати в одне ціле і помістити в основний файл розмітки всередині «тіла» розмітки. Окрім створення розмітки вмісту сторінки, вам також потрібно створити стилі на основі інформації, яку ви отримуєте з конфігураційного файлу, який слід застосувати до створюваної сторінки. Для цього завантажте стилі для кожного компонента, що використовується на сторінці, у пам’ять сервера. Потім їх потрібно з’єднати в одне ціле. Стилі слід перевіряти на дублювання під час композиції. Результат слід помістити на вкладку «стилі» основного файлу розмітки. Подібні кроки потрібні у випадку створення клієнтського коду. Єдина відмінність полягає в тому, що його слід помістити не в тег «style», а в тег «script».

На основі всього вищесказаного був розроблений метод візуалізації [22] для побудови повністю динамічної та оптимізованої веб-сторінки (рис. 2.4), який складається з наступних кроків:

— отримати інформацію про компоненти, які використовуються на сайті;

— завантажити гіпертекстову розмітку для кожного компонента;

— запитувати базу даних на наявність динамічного вмісту;

— створювати файли стилів відповідно до використовуваних компонентів;

— згенерувати весь необхідний код JavaScript;

— відправити згенеровані результати у браузер користувача.

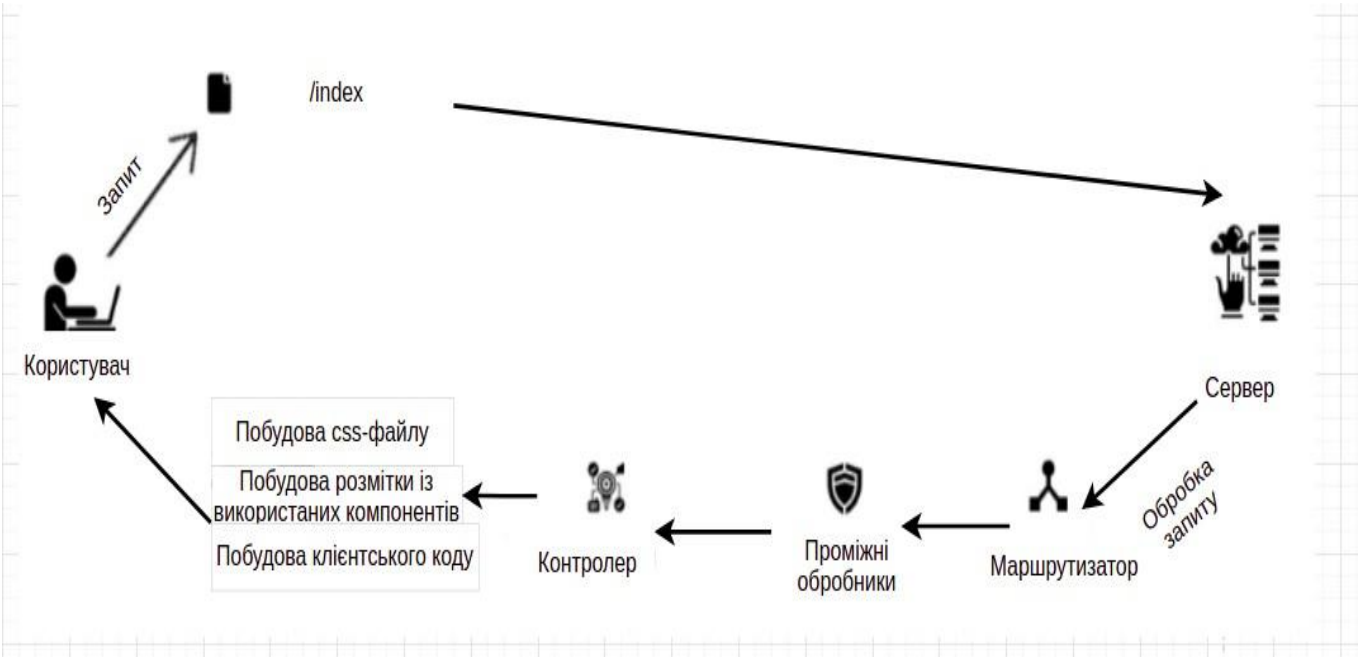


Рисунок 2.4 – Рендеринг оптимізованої веб сторінки

Реалізація такого методу створення сторінок на сервері гарантує швидкість роботи веб-системи, оскільки через Інтернет будуть передаватися лише необхідні дані.

2.2 Проектування методу роботи з пошуковою оптимізацієюпроектованої технології моделювання

Як обговорювалося в Розділі 1.1, пошукова оптимізація є важливою частиною успіху будь-якої веб-системи. Основними компонентами, які впливають на SEO, є системний контент і мета інформація.

Використання SEO в розробленій системі дає повний контроль над вмістом веб-сайту та його метаінформацією.

Метаінформація заповнюється спеціальними тегами в гіпертекстовій розмітці веб-сторінок. Основними мета-тегами [23] є заголовок, опис, ключові слова. Крім основного тега є допоміжні теги, з яких og-теги мають бути виділені. Ці етикетки використовують технологію OpenGraph зображеному на рисунку 2.5 [24]. Facebook запустив Open Graph у 2010 році. Він розроблений для інтеграції Facebook з іншими веб-сайтами. Це допомагає сайту стати багатим об’єктом у розділі соціальних мереж, тобто дозволяє веб-сторінці мати ту саму

функціональність, що й інші об’єкти Facebook. Технологія дозволяє спілкуватися в соціальних мережах.

Рисунок 2.5 — Результат використання технології OpenGraph

Відображати інформацію про сторінку правильно. Це стане в нагоді, коли користувач ділиться посиланням на сторінку в соціальній мережі, а інші користувачі можуть побачити коротку інформацію про сторінку, не відвідуючи її, як показано на рисунку 2.6.

Є чотири основні теги og, які є обов’язковими: og-title, og-description, og-type, og-description.

Наступним кроком для досягнення хорошої продуктивності, що використовується в технічному проектуванні, є використання мікроданих [25]. Ці дані не змінюють відображення сторінки користувача. Вони лише вказують пошуковим роботам тип інформації в певному місці гіпертекстової розмітки сторінки. Чудовим прикладом використання мікротегів є тегування блоків із відповідями на поширені запитання. На малюнку 2.5 наведено приклад результату використання мікроданих.

Ще одна технологія, яка допомагає покращити рейтинги веб-сайтів, — це AMP [26]. Це технологія оптимізації веб-сторінок під мобільні пристрої, розроблена Google у 2015 році.

Технологія AMP має багато обмежень для розробників, але гарантує, що при дотриманні всіх умов сторінка матиме кращу оцінку. До літа 2021 року сторінки, створені за допомогою технології AMP у пошуку, відображатимуться як спеціальна розмітка у вигляді блискавок, що додатково демонструє важливість технології, як показано на рисунку 2.7

Технологія складається з наступних компонентів:

— AMP HTML – мова HTML, де деякі теги замінюються еквівалентними тегами AMP, а деякі теги заборонені.

— AMP JS - Працює за допомогою власної бібліотеки JS, що дозволяє елементам сторінки завантажуватися асинхронно.

— кешування Google AMP - Під час процесу індексування сторінок AMP

— пошукова система кешує свої дані та відтворює їх зі своїх серверів.
пошукова система кешує свої дані та відтворює їх зі своїх серверів.

У проектованій системі моделювання пропонується вводити ці дані під час створення сутності сторінки, а також надається можливість редагування цієї інформації у будь який момент часу.

Заповнення мета-інформації та використання системного вмісту є єдиним легальним способом використання SEO в системі, але іноді цього недостатньо для досягнення хороших результатів. З огляду на це була створена спеціальна функція для системи проектування для моделювання веб-систем, яка перевіряє кожного користувача, що входить до системи. Якщо таким користувачем є пошуковий бот, сервер повертає спеціальну версію сторінки без клієнтського коду. Цю перевірку легко реалізувати, оскільки кожен запит до сервера має заголовок «User-Agent», який описує основну інформацію про систему користувача. Заголовки від звичайних користувачів відрізняються від заголовків від ботів, що дозволяє легко ідентифікувати та приховати певну інформацію. Це пришвидшує завантаження сторінки, що також є важливим показником при ранжуванні сторінок.

Незаконні методи візуалізації сторінок на сервері з використанням усіх вищезазначених методів та інструментів у поєднанні з SEO забезпечують хорошу продуктивність і, таким чином, хорошу конверсію користува.

Рисунок 2.6 — Результат використання мікроданих

Рисунок 2.7 — Результат технології AMP

Каскадні таблиці стилів (CSS) (рисунок 2.8) є потужним інструментом для керування зовнішній вигляд вашого веб-сайту, вплине на уявлення документа або набору документів.

Переваги CSS:

— CSS дозволяє значно багатший, ніж документ виступу HTML все дозволено, навіть у розпал презентаційних запал HTML;

— CSS дозволяє задати кольору на текст і на тлі будь-якого елементу;

— CSS дозволяє створювати рамки навколо будь-якого елемента, а також — збільшення або зменшення простору навколо них.

Рисунок 2.8 — Логотип CSS

CSS дозволяє змінити спосіб текст пишеться з великої літери, прикрашені (наприклад, основний), розташованих на відстані, і навіть будь воно відображається на всіх. І CSS дозволяє виконувати багато інших ефектів. За допомогою CSS, ви також можете створювати спеціальні ефекти, включаючи текст перекидання, а також контролювати розміщення графіки та дизайну макета сторінки. Каскадні таблиці стилів являє собою набір інструкцій, який вказує веббраузер, як представити або дисплей, різні HTML-елементи, такі як, який шрифт використовувати, який розмір тексту абзацу повинен бути, який колір тексту заголовка має бути, або Не повинно бути колір фону стосовно до конкретних елементів, і так далі. Ви можете використовувати кілька таблиць стилів, щоб налаштувати дисплей вашого веб-сайту в різних браузерах, різних платформ і різних пристроїв

3 ПРОЕКТУВАННЯ ПАНЕЛІ АДМІНІСТРАТОРА З МОЖЛИВІСТЮ МОДЕЛЮВАННЯ ВЕБ- ОРІЄНТОВАНОЇ СИСТЕМИ

3.1 Вибір технологій розробки.

У розробці проекту братимуть участь наступні технології:

— операційна система – будь-яка (вибір операційної системи важливий лише на етапі розгортання продукту);

— база даних – MySQL (XAMPP)[28]:

— мови - JS, CSS, HTLM[29].

PostgreSQL — це розширена реляційна база даних корпоративного рівня з відкритим кодом, яка підтримує запити SQL (реляційні) і JSON (нереляційні). SQL — це декларативна мова програмування, за допомогою якої користувачі можуть взаємодіяти з базами даних для створення запитів, оновлення та керування реляційними базами даних, створення та зміни схем баз даних, а також керування системами доступу до бази даних. PostgreSQL – це надзвичайно стабільна система управління базами даних, що підтримується більш ніж 20-річним розвитком спільноти, що сприяло її високому рівню стійкості, цілісності та цілісності. PostgreSQL використовується як основне сховище даних або сховище даних для багатьох веб-, мобільних, геопросторових та аналітичних додатків. Остання основна версія — PostgreSQL зображена на рисунку 3.1.

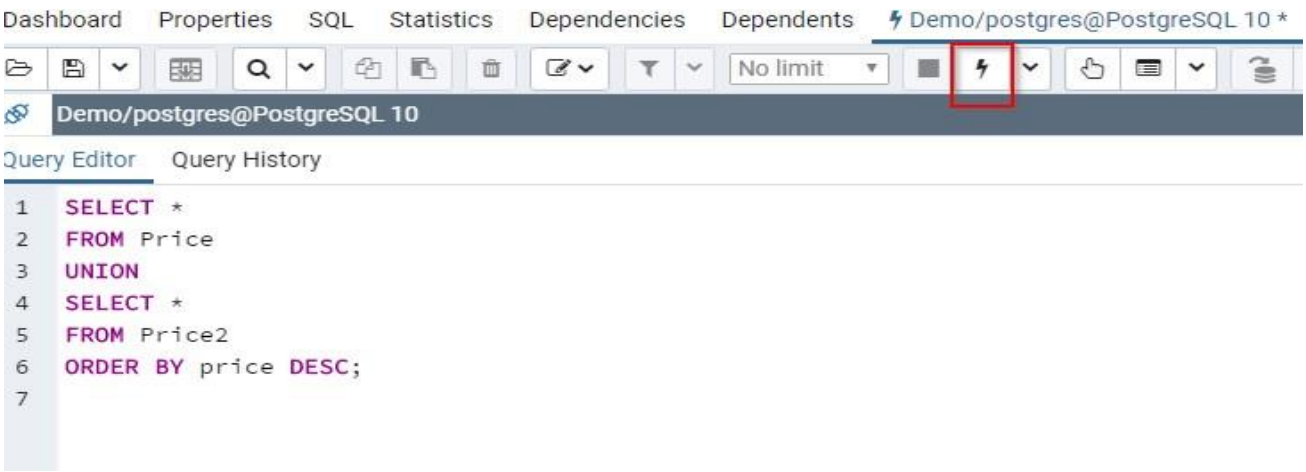


Рисунок 3.1 – Приклад запиту до бази даних

JS — це платформа, орієнтована на ергономіку, стабільність і швидкість

розробника. AdonisJS був написаний з нуля з чіткими принципами та цілями, щоб бути надійною інтегрованою системою. Він також дотримується тих же принципів MVC, що використовуються в багатьох популярних фреймворках, таких як Laravel, Rails і Spring. Він зосереджений на користувацькому досвіді, стабільності та швидкості для розробників. Як повноцінна веб-платформа, AdonisJS має набір вбудованих функцій і модулів, які відрізняють його від інших фреймворків Node.js.

Текст Sublime, який ми будемо використовувати для розробки, має вбудовану систему маршрутизації, яка дозволяє описувати статичні шляхи до певних ресурсів і динамічні шляхи з одним або кількома параметрами. Ще однією перевагою adonisjs є можливість створення проміжного програмного забезпечення, тобто проміжних функцій обробки, які є дуже важливою частиною практично будь-якої веб-орієнтованої системи. Хоча зараз популярно використовувати односторінкові програми (SPA), де весь вміст сторінки перетворюється на JavaScript без перезавантаження, що є хорошим рішенням для складних інтерфейсів користувача, рендеринг на стороні сервера все ще має свої програми, тобто сторінки в ресурсах, які повинні бути добре індексується пошуковими системами. Adonisjs має вбудований HTML-шаблон, який дозволяє відтворювати його на стороні сервера. Крім того, до переваг фреймворку можна віднести наявність конструктора SQL-запитів.

Перевірка введення є дуже важливою частиною будь-якої серверної програми, adonisjs має вбудований засіб перевірки даних з багатьма функціями перевірки. Крім перерахованого вище, adonisjs має кілька інших модулів, які значно спрощують написання коду порівняно з іншими.

Після вибору методик розвитку необхідно організувати ефективну роботу з ними. Налаштування цих інструментів окремо на комп’ютері кожного розробника займе багато часу. Щоб обійти це, вам потрібно запакувати ці технології в контейнер, який потім можна буде завантажити та запустити в будь-якій системі. Ця проблема не нова, і наразі існують інструменти для її вирішення. Найкращим інструментом є Docker (рисунок 3.2).

Рисунок 3.2 – Логотип docker

Docker — це платформа для контейнерування з відкритим кодом. Це дозволяє розробникам упаковувати програми в стандартизовані виконувані компоненти, які поєднують вихідний код програми з бібліотеками операційної системи (ОС) і залежностями, необхідними для виконання цього коду в будь-якому середовищі. Контейнери полегшують доставку розподілених додатків і стають все більш популярними, оскільки організації переходять до хмарної розробки та гібридних багатохмарних середовищ. Docker — це, по суті, набір інструментів, які дозволяють розробникам створювати, розгортати, запускати, оновлювати та зупиняти контейнери за допомогою простих команд, автоматично заощаджуючи час за допомогою єдиного API (рисунок 3.3).

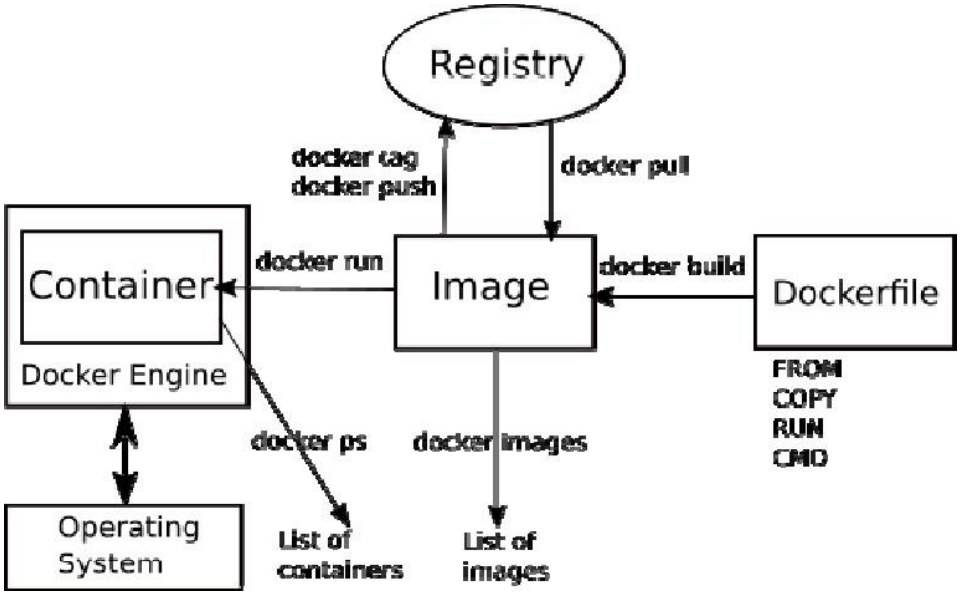


Рисунок 3.3 – Структура docker

Контейнери стають можливими завдяки функції ізоляції процесів і віртуалізації, вбудованих у ядро Linux. Ці функції, такі як групи керування (C-групи) для розподілу ресурсів між процесами та просторами імен, щоб обмежити доступ процесу або видимість до інших ресурсів або областей системи, дозволяють кільком компонентам програми працювати однаково в одному спільних ресурсах. Виконуючи роль гіпервізора, кільком віртуальним машинам (ВМ) можна використовувати центральний процесор, пам’ять та інші ресурси одного апаратного сервера.

В результаті технологія контейнерів пропонує всі функціональні можливості переваги віртуальних машин, включаючи ізоляцію додатків, економічно масштабоване масштабування та одноразове використання, а також важливі додаткові переваги. Менша вага: на відміну від віртуальних машин, контейнери не розміщують всю операційну систему та корисне навантаження гіпервізора; вони містять лише процеси операційної системи та залежності, необхідні для виконання коду. Розмір контейнера вимірюється в мегабайтах (порівняно з гігабайтами для деяких віртуальних машин), покращує використання ємності пристрою та має швидший час запуску.Також підвищена ефективність використання ресурсів: за допомогою контейнерів ви можете запускати в кілька разів більше копій програм, ніж віртуальні машини на одному обладнанні. Це може зменшити витрати на хмару.

Покращена продуктивність розробника: контейнери швидше та легше розгортати, надавати та перезапускати, ніж віртуальні машини. Це робить їх ідеальними для систем безперервної інтеграції та безперервної доставки (CI/CD), а також для команд розробників, які впроваджують Agile та DevOps практики.

Щоб контейнерувати всі необхідні служби, вам потрібно описати свій Dockerfile (файл директиви запуску) для кожної служби та створити файл docker-compose (рисунок 3.4), який описує загальні правила запуску програмного забезпечення. Compose — це інструмент для визначення та запуску багатьох контейнерів Docker. За допомогою Compose ви можете використовувати файл YAML для налаштування служб вашої програми.

Після написання цих інструкцій ви можете запустити програму лише однією командою, яка автоматично запустить всі необхідні служби всередині ізольованого контейнера. Docker також дозволяє вам не турбуватися про версію вашого сервісу. Docker завантажує версію програмного забезпечення, описану в Dockerfile (рисунок 3.5), що гарантує, що система може працювати на будь-якому комп'ютері, економлячи багато часу розробникам.

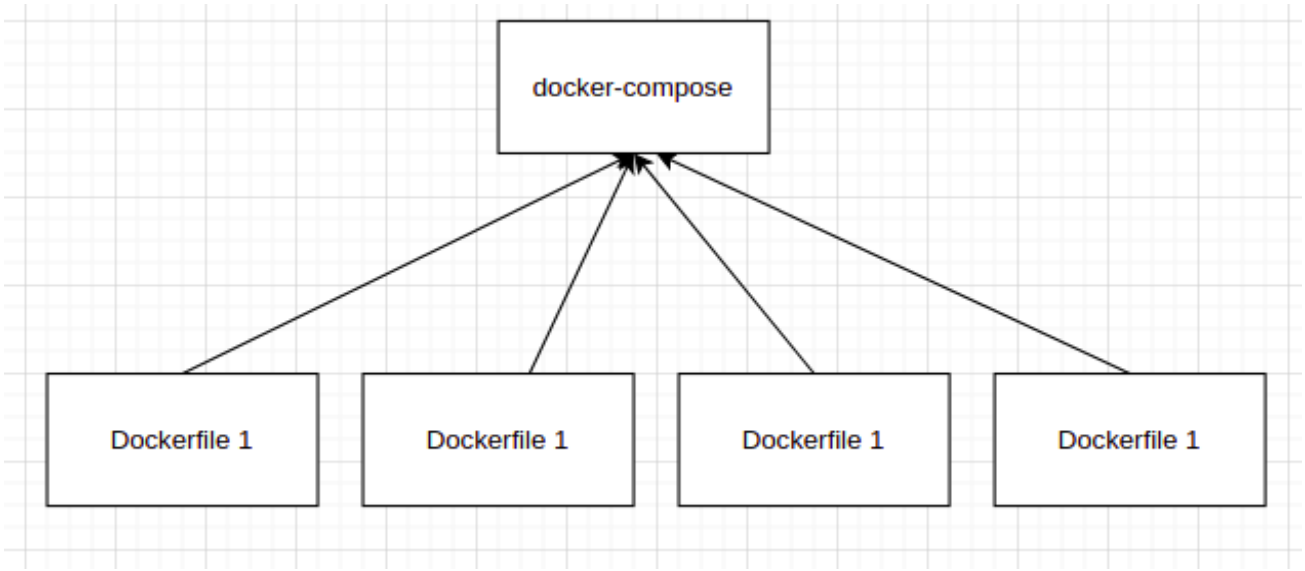


Рисунок 3.4 — Схема роботи docker-compose

```
FROM nginx:alpine

FROM postgres:12.5

# If you need POSTGRES_MULTIPLE_DATABASES
# COPY ./docker/postgres/create-multiple-postgresql-databases.sh /docker-entrypoint-initdb.d

# If you need install DB extensions
COPY ./docker/postgres/extensions.sql /docker-entrypoint-initdb.d
```

Рисунок 3.5 — Dockerfile для nginx

Після виконання описаних вище дій маємо проміжний результат — базовий шаблон web-орієнтованої системи.

3.2 Проектування інформаційної технології моделювання web-орієнтованої системи.

Основою кожної системи є база даних. Для створення динамічних сторінок, створених з панелі адміністратора, розроблена схема бази даних, що складається з двох основних таблиць: «сторінки» та «компоненти». Схема для цієї частини бази даних показана на рисунку 3.6. У таблиці "сторінки" зберігаються заголовки сторінок, унікальні ідентифікатори, які використовуються для створення посилань, мета-інформація та логічне значення, що відповідає за доступність сторінки.

У цій таблиці зберігається інформація, до якої належить компонент, тип компонента, порядок компонента на сторінці та поле json «дані», яке зберігає динамічні дані. Міграції баз даних для створення таблиць наведена на рисунку 3.7.

Окрім таблиць, що необхідні для побудови сторінки, було створено таблицю в якій зберігається інформація про навігацію системи, що моделюється. Ця таблиця називається “navigation” її схему зображено на рисунку 3.6.

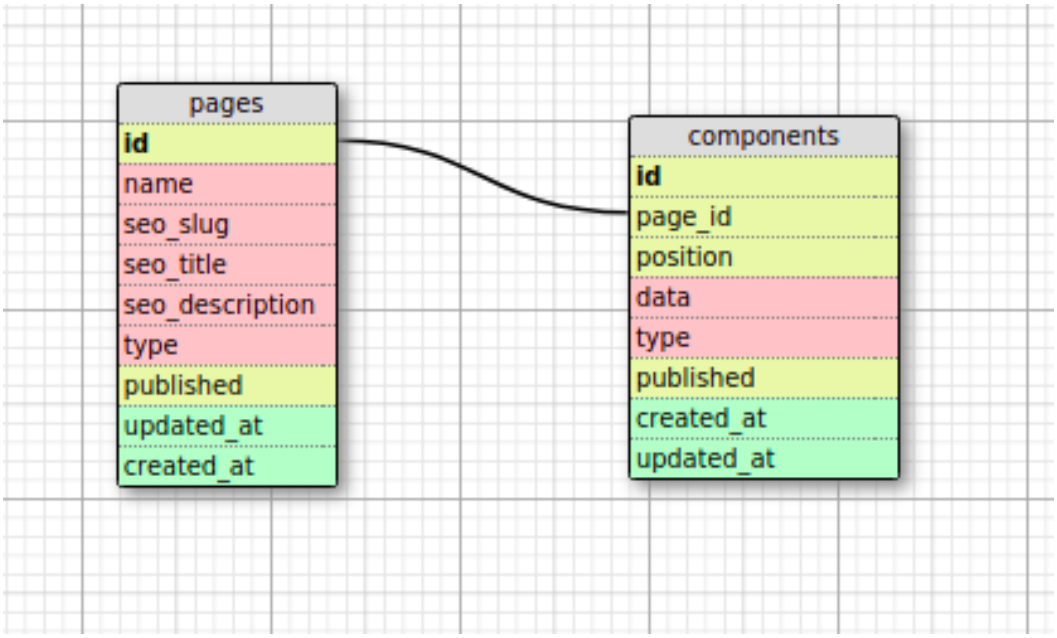


Рисунок 3.6 – Схема бази даних веб-сторінок

```
class ComponentsSchema extends Schema {
  up () {
    this.create('components', (table) => {
      table.increments()
      table.string('name')
      table.string('type')
      table.integer('position')
      table.integer('page_id').references('id').inTable('pages').onDelete('CASCADE').onUpdate('CASCADE')
      table.json('data')
      table.boolean('published').defaultTo(false);
      table.timestamps()
    })
  }
```

Рисунок 3.7 – Міграції бази даних для таблиці компонентів

Дуже цікавим елементом цієї таблиці є зовнішній ключ, який посилається на записи в одній таблиці. Це рішення може створювати вкладення, що дуже важливо для навігації по веб-системах.

На додаток до наведених вище таблиць також створюються таблиці для зберігання адміністраторів, ролей і дозволів.

Для полегшення роботи з базою даних ми опишемо модель (рисунок 3.8) за допомогою шаблону оформлення під назвою «ActiveRecord».

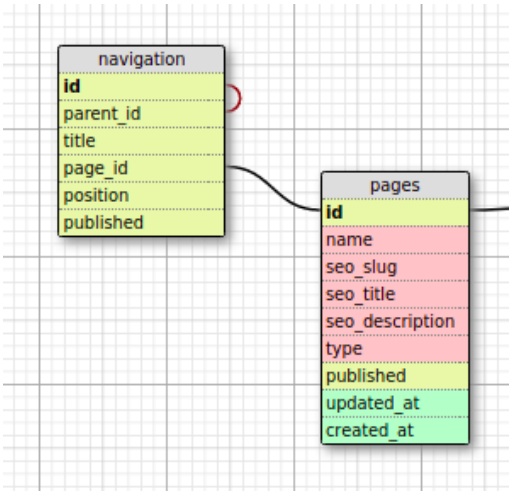


Рисунок 3.8 – Схема таблиці “navigation” та її зв’язок із таблицею “pages”

Шаблони Active Record(рисунок 3.9) — це шаблони дизайну для доступу до реляційних баз даних. Модель бази даних описує реляційні методи життєвого циклу.

```
class Page extends Model {
  static boot () {
    super.boot()
    this.addHook('afterSave', async (instance) => {
      await instance.load('allComponents')
      await Promise.all(instance.getRelated('allComponents').rows.map(async (component) => {
        if(Config.get(`admin.components.${component.type}`).not_allowed.includes(instance.type)) {
          component.published = false;
          return await component.save();
        }
        return component;
      }))
    })
  }

  static get table() {
    return 'pages';
  }

  allComponents() {
    return this.hasMany('Pages/Models/Component', 'id', 'page_id')
  }

  components() {
    return this.hasMany('Pages/Models/Component', 'id', 'page_id').where('published', true).orderBy('position', 'asc')
  }
```

Рисунок 3.9 – Модель для роботи із таблицею “pages”

Схему взаємодії кожної сутності з бази даних при побудові динамічної сторінки зображено на рисунку 3.10.

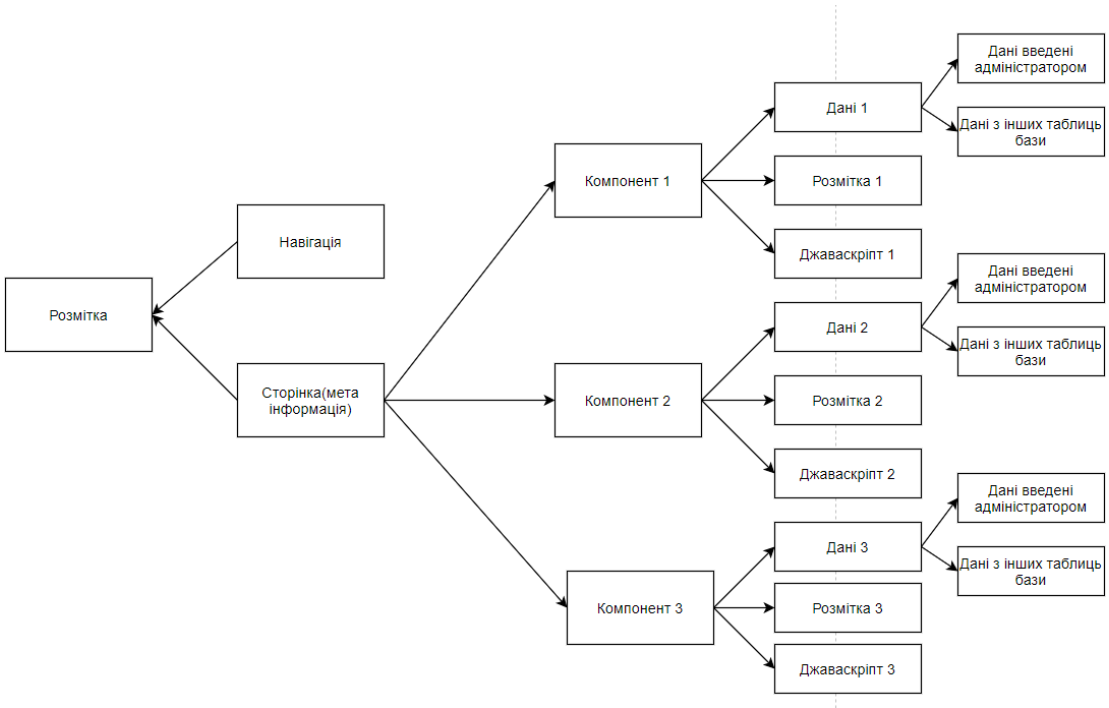


Рисунок 3.10 – Модель взаємодії сутностей бази даних при побудові динамічноїсторінки

3.3 Розробка інформаційної технології моделювання web-орієнтованої системи

Після створення схеми бази даних було прийнято рішення розробитивласну архітектуру на основі тієї, що вже була реалізована у фреймворку Adonisjs.

Архітектура — це принцип організації коду системи. Для зручної роботи з кодом було вирішено розділити його на модулі. В кожному модулі будуть зберігатися власні контроллери, маршрутизатори, проміжні обробники, конфігураційніфайли, розмітка та інше. Для зручності роботи із такою архітектурою було написано код, який автоматично реєструє модулі у сервері node.js (рисунок 3.11). Цей код складається функцій для реєстрації розмітки, конфігурації, маршрутизації, побудови меню адмін панелі.

```
registerConfigs(modulePath, moduleName) {
  const configsPath = path.resolve(modulePath, moduleName, 'Configs');
  if (fs.existsSync(configsPath)) {
    const moduleMenu = requireAll(configsPath);
    const { _config } = Config;

    Object.assign(Config, { _config: deepmerge(_config, moduleMenu) });
  }
}
```

Рисунок 3.11 – Код для роботи із конфігурацією

Для цього прописується спеціальний код, який виконується при першому запуску програми. До початкових даних відносяться: базові ролі панелі адміністратора, доступи та перший адміністратор. Приклад коду для заповнення бази даних:

Для надання доступів адміністратору було написано спеціальну функцію. Ця функція збирає інформацію про доступи, яка зберігається у модулях, додає їх до бази та приєднує їх до адміністратора. Приклад використання функції зображено на рисунку 3.12.

```
node@7e5c6bb6f1a1:~/app$ adonis module:sync
articles_view
articles_create
articles_edit
articles_delete
authors_view
authors_create
authors_edit
authors_delete
developers_view
developers_create
developers_edit
developers_delete
admin_users_view
admin_users_create
admin_users_edit
admin_users_delete
admin_roles_view
admin_roles_create
admin_roles_edit
admin_roles_delete
navigation_view
navigation_create
navigation_edit
navigation_delete
```

Рисунок 3.12. – Використання команди для надання доступів адміністратору.

Наступним етапом розробки системи є власне створення модулів роботи із навігацією та сторінками. У цих модулях повинні бути реалізовані функції для додавання, перегляду, редагування та видалення записів. Для відображення даних, що знаходяться у таблиці було використано бібліотеку під назвою Datatables. Використання цієї бібліотеки вимагає написання сервісу для опрацювання запитів. У цьому сервісі повинні бути реалізовані функції для пагінації, пошуку, сортування даних. Для створення та редагування даних було прийнято рішення використовувати модальні вікна.

Модуль роботи з динамічними веб-сторінками є досить складним та складається із багатьох функцій та налаштувань. Для того щоб не навантажувати інтерфейс було прийнято рішення розділити процес створення сторінки на декілька кроків. Спочатку створюється сторінка заповнення мета-інформації. Далі відбувається перехід до списку комнонентів сторінки. Після цього можно добавити новий елемент на сторінку. Редагування даних елементу.

Такий процес створення та наповнення сторінки є дуже простим у використанні та не потребує ніяких додаткових навичок. Для того, щоб реалізувати можливість кожному компоненту мати власний файл стилів та власний файл із клієнтським кодом, було прийнято рішення створювати окрему папку для кожного із таких компонентів, що зображено на рисунку 3.13. У цій папці обов’язково знаходиться розмітка, яка повертається користувачу. Окрім цієї розмітки у папці ще можуть знаходитись: форма для заповнення інформації, файл із стилями, файл із клієнтським кодом.

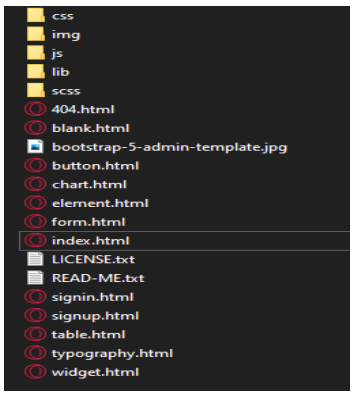


Рисунок 3.13 – Структура папки компонентів

Кожний компонент має свої налаштування, які зберігаються увідповідному конфігураційному файлі. У цьому конфігураційному файлі вказується на яких типах сторінок цей компонент може використовуватися, тип цього компоненту(статичний чи динамічний), чи потрібно завантажувати стилі, запити до бази даних, які необхідно зробити або під час рендерингу, або під час редагування інформації(рисунок 3.14).

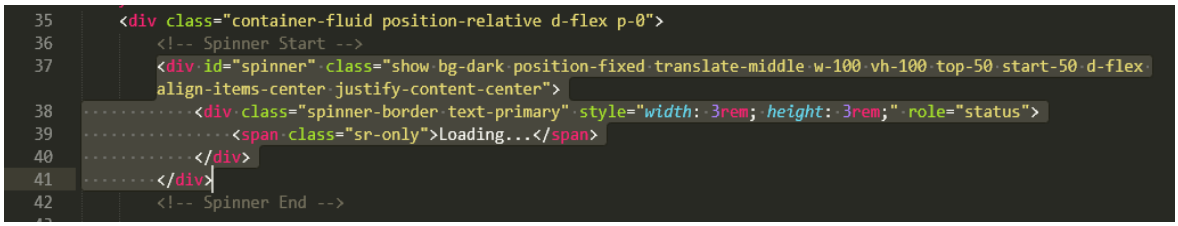


Рисунок 3.14 – Конфігурація одного із компонентів.

Наступним кроком розробки системи керування контентом є валідація вхідних даних для кожного компоненту. Для цього був створений ще один (рисунок 3.15)

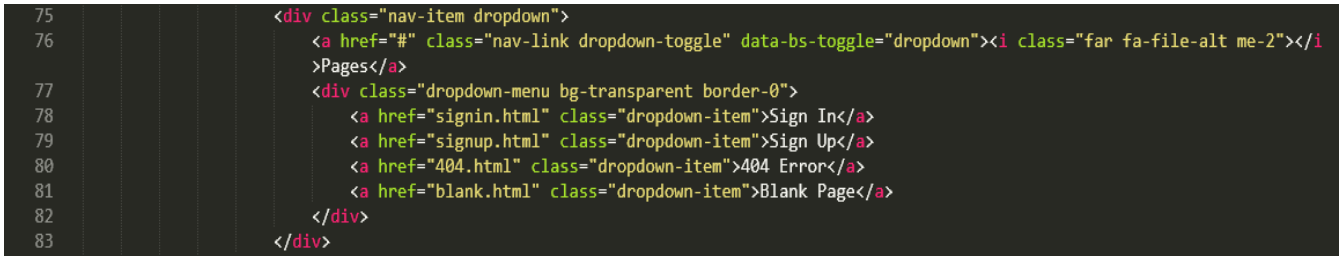


Рисунок 3.15 – Валідація вхідних даних для кожного компонента.

Ці валідаційні правила автоматично підвантажуються перед у головний валідатор перед збереженням інформації. Коли адміністратор зберігає інформацію про компонент спочатку дані потрапляють у головний валідатор, де перевіряється наявність такого компоненту в системі та підвантажуються валідаційні правила спеціально для цього компонента.

Після того, як було реалізовано можливість наповнювати систему компонентами та перевіряти вхідні дані, можна перейти до створення контроллеру, який власне буде ці дані зберігати. У контролері було описано декілька основних методів, серед яких можна виділити метод, для зміни порядку

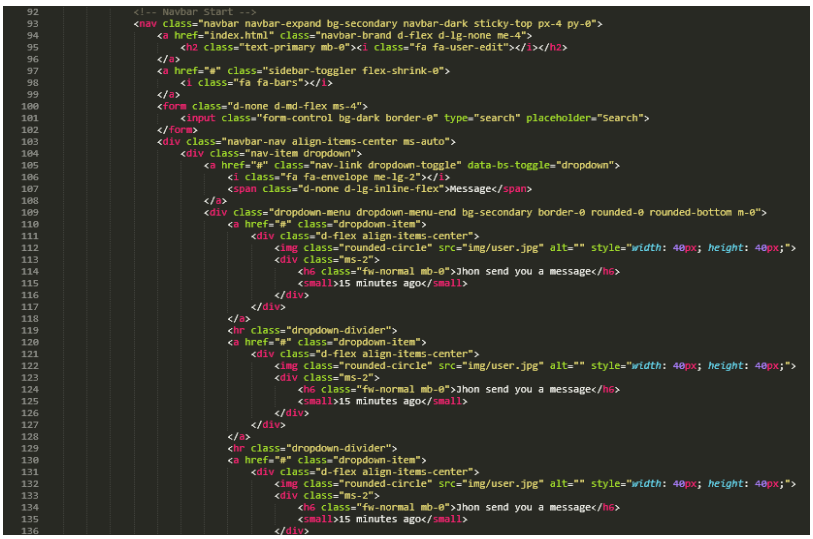
відображення компонентів на сторінці (reorder).

Після створення функціоналу для створення сторінок та наповнення їх контентом залишилось тільки реалізувати рендеринг цієї сторінки. Для реалізації цього було використано технологію розроблену у підрозділі 2.1. У цій функції буде формуватися весь необхідний для веб-сторінки код: гіпертекстова розмітка розмітка, стилі, клієнтський код, запити до бази даних зображено на рисунку 3.16.

```
(function ($) {
    "use strict";

    // Spinner
    var spinner = function () {
        setTimeout(function () {
            if ($('#spinner').length > 0) {
                $('#spinner').removeClass('show');
            }
        }, 1);
    };
    spinner();


    // Back to top button
    $(window).scroll(function () {
        if ($(this).scrollTop() > 300) {
            $('.back-to-top').fadeIn('slow');
        } else {
            $('.back-to-top').fadeOut('slow');
        }
    });
    $('.back-to-top').click(function () {
        $('html, body').animate({scrollTop: 0}, 1500, 'easeInOutExpo');
        return false;
    });


    // Sidebar Toggler
    $('.sidebar-toggler').click(function () {
        $('.sidebar, .content').toggleClass("open");
        return false;
    });


    // Progress Bar
    $('.pg-bar').waypoint(function () {
        $('.progress .progress-bar').each(function () {
            $(this).css("width", $(this).attr("aria-valuenow") + '%');
        });
    }, {offset: '80%'});


    // Calender
    $('#calender').datetimepicker({
        inline: true,
        format: 'L'
    });


    // Testimonials carousel
    $(".testimonial-carousel").owlCarousel({
        autoplay: true,
        smartSpeed: 1000,
        items: 1,
        dots: true,
        loop: true,
        nav : false
    });


    // Chart Global Color
    Chart.defaults.color = "#6C7293";
    Chart.defaults.borderColor = "#000000";
```

Рисунок 3.16 – Код головного валідатора.

В результаті описаних вище кроків було створено повноцінну систему моделювання веб-орієнтованих систем, яка може конкурувати із технологіями-аналогами. Завдяки власному підходу до рендерингу сторінок на сервері ця технологія гарантує швидкість роботи системи навіть при великій кількості компонентів та запитів до бази даних зображено на рисунку 3.17.



```
<!-- Navbar Start -->
<nav class="navbar navbar-expand bg-secondary navbar-dark sticky-top px-4 py-0">
    <a href="index.html" class="navbar-brand d-flex d-lg-none me-4">
        <h2 class="text-primary mb-0"><i class="fa fa-user-edit"></i></h2>
    </a>
    <a href="#" class="sidebar-toggler flex-shrink-0">
        <i class="fa fa-bars"></i>
    </a>
    <form class="d-none d-md-flex ms-4">
        <input class="form-control bg-dark border-0" type="search" placeholder="Search">
    </form>
    <div class="navbar-nav align-items-center ms-auto">
        <div class="nav-item dropdown">
            <a href="#" class="nav-link dropdown-toggle" data-bs-toggle="dropdown">
                <i class="fa fa-envelope me-lg-2"></i>
                <span class="d-none d-lg-inline-flex">Message</span>
            </a>
            <div class="dropdown-menu dropdown-menu-end bg-secondary border-0 rounded-0 rounded-bottom m-0">
                <a href="#" class="dropdown-item">
                    <div class="d-flex align-items-center">
                        <img class="rounded-circle" src="img/user.jpg" alt="" style="width: 40px; height: 40px;">
                        <div class="ms-2">
                            <h6 class="fw-normal mb-0">Jhon send you a message</h6>
                            <small>15 minutes ago</small>
                        </div>
                    </div>
                </a>
                <hr class="dropdown-divider">
                <a href="#" class="dropdown-item">
                    <div class="d-flex align-items-center">
                        <img class="rounded-circle" src="img/user.jpg" alt="" style="width: 40px; height: 40px;">
                        <div class="ms-2">
                            <h6 class="fw-normal mb-0">Jhon send you a message</h6>
                            <small>15 minutes ago</small>
                        </div>
                    </div>
                </a>
                <hr class="dropdown-divider">
                <a href="#" class="dropdown-item">
                    <div class="d-flex align-items-center">
                        <img class="rounded-circle" src="img/user.jpg" alt="" style="width: 40px; height: 40px;">
                        <div class="ms-2">
                            <h6 class="fw-normal mb-0">Jhon send you a message</h6>
                            <small>15 minutes ago</small>
                        </div>
```

Рисунок 3.17 – Код наповнення сторінки

ВИСНОВКИ

У процесі виконання дипломної роботи, проведено ретельний аналіз існуючих платформ та технологій котрі використовують для їх створення, виділено основні схеми побудови ВЕБ-сторінок та додатків, як результат виконання бакалаврської дипломної роботи, розроблену власну панель адміністратора.

В даній роботі було досліджено види та основні принципи створення web-сайтів, розглянуто засоби для їх моделювання. Такі засобами називаються системами контролю вмісту.

Досліджено фактори та способи оптимізації, що впливають на пошукову індексацію сторінок сайту.

Спроектоватовано та розроблено технологію адміністрування web-сайту з можливістю керування процесом побудови сторінки на сервері та налаштуванням мета-інформації, що необхідна для пошукової індексації.

У процесі розробки панелі адміністратора виявлено певні недоліки, а саме: відсутність активності деяких ключових елементів, до прикладу “To do list”, вузько налаштована система чатів, деякі труднощі із авторизацією користувача у формі, а саме – при великому об’ємі користувачів, база даних дає незначні збої.

У подальшому, планується більш повноцінна підтримка та фінансування проекту, а також модернізація із урахуванням перечислених недоліків, та репортів ком’юніті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Моделювання web-орієнтованих систем [Електронний ресурс]. – Режим доступу: http://ena.lp.edu.ua:8080/bitstream/ntb/19246/1/3-Boyko-16-25.pdf

2. Admin Dashboard [Електронний ресурс]. – Режим доступу: https://www.reviewboard.org/docs/manual/3.0/admin/admin-ui/dashboard/

3. Techniques of website speed optimization [Електронний ресурс]. – Режим доступу: https://www.altexsoft.com/blog/engineering/12-techniques-of- website-speed-optimization-performance-testing-and-improvement-practices/

4. Website conversion rate [Електронний ресурс]. – Режим доступу: https://www.geckoboard.com/best-practice/kpi-examples/website-conversion-rate/

5. Google pagespeed Insights [Електронний ресурс]. – Режим доступу: https://kinsta.com/blog/google-pagespeed-insights/

6. Lighthouse [Електронний ресурс]. – Режим доступу: https://developers.google.com/web/tools/lighthouse

7. Efficiently load javascript with defer and async [Електронний ресурс]. – Режим доступу: https://flaviocopes.com/javascript-async-defer/

8. What is minification [Електронний ресурс]. – Режим доступу: https://www.imperva.com/learn/performance/minification/

9. What is SEO? [Електронний ресурс]. – Режим доступу: https://searchengineland.com/guide/what-is-seo

10. Content relevance [Електронний ресурс]. – Режим доступу: https://www.searchmetrics.com/glossary/content-relevance/

11. Rankings [Електронний ресурс]. – Режим доступу: https://www.searchmetrics.com/glossary/rankings/

12. White hat SEO [Електронний ресурс]. – Режим доступу: https://whatis.techtarget.com/definition/white-hat-SEO

13. What is black hat SEO? [Електронний ресурс]. – Режим доступу: https://www.wordstream.com/black-hat-seo

14. What is an operating system? [Електронний ресурс]. – Режим доступу:

https://edu.gcfglobal.org/en/computerbasics/understanding-operating-systems/1/

15. Web server [Електронний ресурс]. – Режим доступу:https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server

16. Database management system [Електронний ресурс]. – Режим доступу: https://www.techopedia.com/definition/24361/database-management-systems-dbms

17. Server Side web frameworks [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Web_frameworks

18. Website [Електронний ресурс]. – Режим доступу:https://www.searchmetrics.com/glossary/website/

19. How a website works [Електронний ресурс]. – Режим доступу: https://www.superwebpros.com/blog/how-does-a-website-work/

20. What is a CMS [Електронний ресурс]. – Режим доступу: What is a CMS [Електронний ресурс]. – Режим доступу: https://kinsta.com/knowledgebase/content-management-system/

21. The pros and cons of using a CMS [Електронний ресурс]. – Режим доступу: https://flickerleap.com/pros-cons-using-cms-build-website/

22. Server side rendering definition [Електронний ресурс]. – Режим доступу: https://www.omnisci.com/technical-glossary/server-side-rendering

23. SEO meta tags [Електронний ресурс]. – Режим доступу: https://www.wordstream.com/meta-tags

24. The OpenGraph protocol [Електронний ресурс]. – Режим доступу: https://ogp.me/

25. Schema.org Documentation [Електронний ресурс]. – Режим доступу: https://schema.org/docs/documents.html

26. The complete guide to accelerated mobile pages [Електронний ресурс]. – Режим доступу: https://instapage.com/blog/amp

27. What is nginx [Електронний ресурс]. – Режим доступу: https://kinsta.com/knowledgebase/what-is-nginx/

28. PostgreSQL [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/

29. Adonisjs [Електронний ресурс]. – Режим доступу: https://adonisjs.com/

30. Docker [Електронний ресурс]. – Режим доступу: https://www.docker.com/

ДОДАТОК А

Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
……. Завідувач кафедри ОТ
_____ проф., д.т.н. О.Д. Азаров
«___»_____2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання дипломної роботи
«Адмін-панель веб сайту інтернет мгазину»
08-23.БДР.017.00.000 ПЗ

Науковий керівник к.т.н., доц. каф. ОТ
_____ Черняк О.І.

Студента групи 1КІ-18б
_____ Швець Є.О.

Вінниця 2022

1 Підстава для виконання бакалаврської дипломної роботи (БДР)

Підставою для розробки даної бакалаврської дипломної роботи є наказ ВНТУ № від «__» ___ 2022 року та рішення засідання кафедри обчислювальної техніки (протокол №_1_ від «_____»__ 2022 року).

2 Вихідні дані для виконання БДР

Розробка технології адміністрування веб-сайту, яка б надала можливість налаштування контенту для покращення швидкодії побудови та пошукової індексації сторінок.

3 Перелік задач

Перелік задач, що повинні бути виконані:

— дослідили види та основні принципи створення web-сайтів;

— дослідити фактори та способи оптимізації, що впливають на пошукову індексацію сторінок сайту;

— спроектувати та розробити технологію адміністрування web-сайту з можливістю керування процесом побудови сторінки на сервері та налаштуванням мета-інформації, що необхідна для пошукової індексації;

4 Порядок контролю виконня та захисту БДР

До захисту подається: пояснювальна записка БДР, графічні і ілюстративні матеріали, протокол попереднього захисту БДР на кафедрі, відгук наукового керівника, відгук рецензента, анотації до БДР українською та іноземною мовами, довідка про відповідність оформлення БДР діючим вимогам.

5 Порядок контролю виконня та захисту БДР

5.1 Робота виконується в три етапи, таблиця А.1

Таблиця А.1 — Етапи виконання роботи

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Інформаційний пошук та огляд інформаційних джерел	12.09.2021	25.11.2021	Розділи 1 та 2
2	Дослідження підходів та розробка схемної реалізації	26.11.2021	04.02.2022	Чернетки матеріалів
3	Підготока матеріалів пояснювальної записки	05.02.2022	28.03.2022	Пояснювальна записка

6 Порядок контролю та прийому

Виконання етапів графічної та розрахункової документації БДР контролюється науковим керівником згідно зі встановленими термінами. Захист БДР відбувається на засіданні Екзаменаційної комісії, затвердженої наказом ректора.

6 Вимоги до оформлювання та порядок виконання БДР

При оформлюванні БДР використовуються:

— ДСТУ 3008 : 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302 : 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— ГОСТ 2.104-2006 «Едина система конструкторської документації. Основні написи»;

— документами на які посилаються у вище вказаних.

Технічне завдання до виконання отримав _____ Швець Є.О.

ДОДАТОК Б

Програмні коди index.html

```
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>DarkPan - Bootstrap 5 Admin Template</title>
    <meta content="width=device-width, initial-scale=1.0" name="viewport">
    <meta content="" name="keywords">
    <meta content="" name="description">
    <!-- Favicon -->
    <link href="img/favicon.ico" rel="icon">
    <!-- Google Web Fonts -->
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;600&fami
ly=Roboto:wght@500;700&display=swap" rel="stylesheet">

    <!-- Icon Font Stylesheet -->
    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css" rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-
icons.css" rel="stylesheet">

    <!-- Libraries Stylesheet -->
    <link href="lib/owlcarousel/assets/owl.carousel.min.css" rel="stylesheet">
    <link href="lib/tempusdominus/css/tempusdominus-bootstrap-4.min.css"
rel="stylesheet" />

    <!-- Customized Bootstrap Stylesheet -->
```

```
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- Template Stylesheet -->
    <link href="css/style.css" rel="stylesheet">
</head>
<body>
    <div class="container-fluid position-relative d-flex p-0">
        <!-- Spinner Start -->
        <div id="spinner" class="show bg-dark position-fixed translate-middle w-100 vh-100 top-50 start-50 d-flex align-items-center justify-content-center">
            <div class="spinner-border text-primary" style="width: 3rem; height: 3rem;" role="status">
                <span class="sr-only">Loading...</span>
            </div>
        </div>
        <!-- Spinner End -->
        <!-- Sidebar Start -->
        <div class="sidebar pe-4 pb-3">
            <nav class="navbar bg-secondary navbar-dark">
                <a href="index.html" class="navbar-brand mx-4 mb-3">
                    <h3 class="text-primary"><i class="fa fa-user-edit me-2"></i>DarkPan</h3>
                </a>
                <div class="d-flex align-items-center ms-4 mb-4">
                    <div class="position-relative">
                        <img class="rounded-circle" src="img/user.jpg" alt="" style="width: 40px; height: 40px;">
                        <div class="bg-success rounded-circle border border-2 border-white position-absolute end-0 bottom-0 p-1"></div>
                    </div>
```

```
                <div class="ms-3">
                    <h6 class="mb-0">Eugene</h6>
                    <span>Admin</span>
                </div>
            </div>
            <div class="navbar-nav w-100">
                <a href="index.html" class="nav-item nav-link active"><i class="fa
fa-tachometer-alt me-2"></i>Dashboard</a>
                <div class="nav-item dropdown">
                    <a href="#" class="nav-link dropdown-toggle" data-bs-
toggle="dropdown"><i class="fa fa-laptop me-2"></i>Elements</a>
                    <div class="dropdown-menu bg-transparent border-0">
                        <a href="button.html" class="dropdown-item">Buttons</a>
                        <a href="typography.html" class="dropdown-
item">Typography</a>
                        <a href="element.html" class="dropdown-item">Other
Elements</a>
                    </div>
                </div>
                <a href="widget.html" class="nav-item nav-link"><i class="fa fa-th
me-2"></i>Widgets</a>
                <a href="form.html" class="nav-item nav-link"><i class="fa fa-
keyboard me-2"></i>Forms</a>
                <a href="table.html" class="nav-item nav-link"><i class="fa fa-table
me-2"></i>Tables</a>
                <a href="chart.html" class="nav-item nav-link"><i class="fa fa-chart-
bar me-2"></i>Charts</a>
                <div class="nav-item dropdown">
                    <a href="#" class="nav-link dropdown-toggle" data-bs-
toggle="dropdown"><i class="far fa-file-alt me-2"></i>Pages</a>
```

```
                    <div class="dropdown-menu bg-transparent border-0">
                        <a href="signin.html" class="dropdown-item">Sign In</a>
                        <a href="signup.html" class="dropdown-item">Sign Up</a>
                        <a href="404.html" class="dropdown-item">404 Error</a>
                        <a href="blank.html" class="dropdown-item">Blank Page</a>
                    </div>
                </div>
            </div>
        </nav>
    </div>
    <!-- Sidebar End -->
    <!-- Content Start -->
    <div class="content">
        <!-- Navbar Start -->
                            <td><input class="form-check-input"
type="checkbox"></td>
                            <td>01 Jan 2045</td>
                            <td>INV-0123</td>
                            <td>Eugene</td>
                            <td>$123</td>
                            <td>Paid</td>
                            <td><a class="btn btn-sm btn-primary"
href="">Detail</a></td>
                        </tr>
                    </tbody>
                </table>
            </div>
        </div>
        <!-- Recent Sales End -->
```

```
        <!-- Widgets Start -->
        <div class="container-fluid pt-4 px-4">
            <div class="row g-4">
                <div class="col-sm-12 col-md-6 col-xl-4">
                    <div class="h-100 bg-secondary rounded p-4">
                        <div class="d-flex align-items-center justify-content-between
mb-2">
                            <h6 class="mb-0">Messages</h6>
                            <a href="">Show All</a>
                        </div>
                        <div class="d-flex align-items-center border-bottom py-3">
                            <img class="rounded-circle flex-shrink-0" src="img/user.jpg"
alt="" style="width: 40px; height: 40px;">
                            <div class="w-100 ms-3">
                                <div class="d-flex w-100 justify-content-between">
                                    <h6 class="mb-0">Eugene</h6>
                                    <small>15 minutes ago</small>
                                </div>
                                <span>Short message goes here...</span>
                            </div>
                        </div>
                        <div class="d-flex align-items-center border-bottom py-3">
                            <img class="rounded-circle flex-shrink-0" src="img/user.jpg"
alt="" style="width: 40px; height: 40px;">
                            <div class="w-100 ms-3">
                                <div class="d-flex w-100 justify-content-between">
                                    <h6 class="mb-0">Eugene</h6>
                                    <small>15 minutes ago</small>
                                </div>
                                <span>Short message goes here...</span>
```

```
                    </div>
                </div>
                <div class="d-flex align-items-center border-bottom py-3">
                    <img class="rounded-circle flex-shrink-0" src="img/user.jpg"
alt="" style="width: 40px; height: 40px;">
                    <div class="w-100 ms-3">
                        <div class="d-flex w-100 justify-content-between">
                            <h6 class="mb-0">Eugene</h6>
                            <small>15 minutes ago</small>
                        </div>
                        <span>Short message goes here...</span>
                    </div>
                </div>
                <div class="d-flex align-items-center pt-3">
                    <img class="rounded-circle flex-shrink-0" src="img/user.jpg"
alt="" style="width: 40px; height: 40px;">
                    <div class="w-100 ms-3">
                        <div class="d-flex w-100 justify-content-between">
                            <h6 class="mb-0">Eugene</h6>
                            <small>15 minutes ago</small>
                        </div>
                        <span>Short message goes here...</span>
                    </div>
                </div>
            </div>
        </div>
        <div class="col-sm-12 col-md-6 col-xl-4">
            <div class="h-100 bg-secondary rounded p-4">
                <div class="d-flex align-items-center justify-content-between
mb-4">
```

```
                <div class="d-flex align-items-center border-bottom py-2">
                    <input class="form-check-input m-0" type="checkbox">
                    <div class="w-100 ms-3">
                        <div class="d-flex w-100 align-items-center justify-content-
between">
                            <span>Short task</span>
                            <button class="btn btn-sm"><i class="fa fa-
times"></i></button>
                        </div>
                    </div>
                </div>
                <div class="d-flex align-items-center border-bottom py-2">
                    <input class="form-check-input m-0" type="checkbox"
checked>
                    <div class="w-100 ms-3">
                        <div class="d-flex w-100 align-items-center justify-content-
between">
                            <span><del>Short task</del></span>
                            <button class="btn btn-sm text-primary"><i class="fa fa-
times"></i></button>
                        </div>
                    </div>
                </div>
                <div class="d-flex align-items-center border-bottom py-2">
                    <input class="form-check-input m-0" type="checkbox">
                    <div class="w-100 ms-3">
                        <div class="d-flex w-100 align-items-center justify-content-
between">
                            <span>Short task</span>
                            <button class="btn btn-sm"><i class="fa fa-
```

```
times"></i></button>
                                </div>
                            </div>
                        </div>
                        <div class="d-flex align-items-center pt-2">
                            <input class="form-check-input m-0" type="checkbox">
                            <div class="w-100 ms-3">
                                <div class="d-flex w-100 align-items-center justify-content-
between">
                                    <span>Short task</span>
                                    <button class="btn btn-sm"><i class="fa fa-
times"></i></button>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <!-- Widgets End -->
        <!-- Footer Start -->
    <!-- Back to Top -->
    <a href="#" class="btn btn-lg btn-primary btn-lg-square back-to-top"><i
class="bi bi-arrow-up"></i></a>
</div>

<!-- JavaScript Libraries -->
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/dist/js/bootstrap.bundle.min.js">
```

```
</script>
    <script src="lib/chart/chart.min.js"></script>
    <script src="lib/easing/easing.min.js"></script>
    <script src="lib/waypoints/waypoints.min.js"></script>
    <script src="lib/owlcarousel/owl.carousel.min.js"></script>
    <script src="lib/tempusdominus/js/moment.min.js"></script>
    <script src="lib/tempusdominus/js/moment-timezone.min.js"></script>
    <script src="lib/tempusdominus/js/tempusdominus-bootstrap-
4.min.js"></script>
    <!-- Template Javascript -->
    <script src="js/main.js"></script>
</body>
</html>
```

ДОДАТОК В

Програмні коди JS

```
(function ($) {
"use strict";
// Spinner
var spinner = function () {
setTimeout(function () {
if ($('#spinner').length > 0) {
$('#spinner').removeClass('show');}}, 1);};
spinner();
// Back to top button
$(window).scroll(function () {
if ($(this).scrollTop() > 300) {
$('.back-to-top').fadeIn('slow');} else {
$('.back-to-top').fadeOut('slow');}});
$('.back-to-top').click(function () {
$('html, body').animate({scrollTop: 0}, 1500, 'easeInOutExpo');
return false;});
// Sidebar Toggler
$('.sidebar-toggler').click(function () {
$('.sidebar, .content').toggleClass("open");
return false;});
// Progress Bar
$('.pg-bar').waypoint(function () {
$('.progress .progress-bar').each(function () {
$(this).css("width", $(this).attr("aria-valuenow") + '%');});
}, {offset: '80%'});
// Calender
$('#calender').datetimepicker({
```

```
inline: true,
format: 'L'});
// Testimonials carousel
$(".testimonial-carousel").owlCarousel({
autoplay: true,
smartSpeed: 1000,
items: 1,
dots: true,
loop: true,
nav : false});
// Chart Global Color
Chart.defaults.color = "#6C7293";
Chart.defaults.borderColor = "#000000";
// Worldwide Sales Chart
var ctx1 = $("#worldwide-sales").get(0).getContext("2d");
var myChart1 = new Chart(ctx1, {
type: "bar",
data: {
labels: ["2016", "2017", "2018", "2019", "2020", "2021", "2022"],
datasets: [{
label: "USA",
data: [15, 30, 55, 65, 60, 80, 95],
backgroundColor: "rgba(235, 22, 22, .7)"},{
label: "UK",
data: [8, 35, 40, 60, 70, 55, 75],
backgroundColor: "rgba(235, 22, 22, .5)"},{
label: "AU",
data: [12, 25, 45, 55, 65, 70, 60],
backgroundColor: "rgba(235, 22, 22, .3)"}]},
options: {
```

```
responsive: true}});
// Salse & Revenue Chart
var ctx2 = $("#salse-revenue").get(0).getContext("2d");
var myChart2 = new Chart(ctx2, {
type: "line",
data: {
labels: ["2016", "2017", "2018", "2019", "2020", "2021", "2022"],
datasets: [{
label: "Salse",
data: [15, 30, 55, 45, 70, 65, 85],
backgroundColor: "rgba(235, 22, 22, .7)",
fill: true},{
label: "Revenue",
data: [99, 135, 170, 130, 190, 180, 270],
backgroundColor: "rgba(235, 22, 22, .5)",
fill: true}]},
options: {
responsive: true}});
// Single Line Chart
var ctx3 = $("#line-chart").get(0).getContext("2d");
var myChart3 = new Chart(ctx3, {
type: "line",
data: {
labels: [50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150],
datasets: [{
label: "Salse",
fill: false,
backgroundColor: "rgba(235, 22, 22, .7)",
data: [7, 8, 8, 9, 9, 9, 10, 11, 14, 14, 15]}]},
options: {
```

```
responsive: true}});
// Single Bar Chart
var ctx4 = $("#bar-chart").get(0).getContext("2d");
var myChart4 = new Chart(ctx4, {
type: "bar",
data: {
labels: ["Italy", "France", "Spain", "USA", "Argentina"],
datasets: [{
backgroundColor: [
"rgba(235, 22, 22, .7)",
"rgba(235, 22, 22, .6)",
"rgba(235, 22, 22, .5)",
"rgba(235, 22, 22, .4)",
"rgba(235, 22, 22, .3)"],
data: [55, 49, 44, 24, 15]}]},
options: {
responsive: true}});
// Pie Chart
var ctx5 = $("#pie-chart").get(0).getContext("2d");
var myChart5 = new Chart(ctx5, {
type: "pie",
data: {
labels: ["Italy", "France", "Spain", "USA", "Argentina"],
datasets: [{
backgroundColor: [
"rgba(235, 22, 22, .7)",
"rgba(235, 22, 22, .6)",
"rgba(235, 22, 22, .5)",
"rgba(235, 22, 22, .4)",
"rgba(235, 22, 22, .3)"],
```

```
data: [55, 49, 44, 24, 15]}]},
options: {
responsive: true}});
// Doughnut Chart
var ctx6 = $("#doughnut-chart").get(0).getContext("2d");
var myChart6 = new Chart(ctx6, {
type: "doughnut",
data: {
labels: ["Italy", "France", "Spain", "USA", "Argentina"],
datasets: [{
backgroundColor: [
"rgba(235, 22, 22, .7)",
"rgba(235, 22, 22, .6)",
"rgba(235, 22, 22, .5)",
"rgba(235, 22, 22, .4)",
"rgba(235, 22, 22, .3)"],
data: [55, 49, 44, 24, 15]}]},
options: {
responsive: true}});
})(jQuery);
```

ДОДАТОК Г

Програмні коди CSS

```
:root {
    --primary: #EB1616;
    --secondary: #191C24;
    --light: #6C7293;
    --dark: #000000;}
.back-to-top {
    position: fixed;
    display: none;
    right: 45px;
    bottom: 45px;
    z-index: 99;}
/*** Spinner ***/
#spinner {
    opacity: 0;
    visibility: hidden;
    transition: opacity .5s ease-out, visibility 0s linear .5s;
    z-index: 99999;}
#spinner.show {
    transition: opacity .5s ease-out, visibility 0s linear 0s;
    visibility: visible;
    opacity: 1;}
/*** Button ***/
.btn {
    transition: .5s;}
.btn-square {
    width: 38px;
    height: 38px;}
.btn-sm-square {
```

```
    width: 32px;
    height: 32px;}
.btn-lg-square {
    width: 48px;
    height: 48px;}
.btn-square,
.btn-sm-square,
.btn-lg-square {
    padding: 0;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    font-weight: normal;
    border-radius: 50px;}
/*** Layout ***/
.sidebar {
    position: fixed;
    top: 0;
    left: 0;
    bottom: 0;
    width: 250px;
    height: 100vh;
    overflow-y: auto;
    background: var(--secondary);
    transition: 0.5s;
    z-index: 999;}
.content {
    margin-left: 250px;
    min-height: 100vh;
    background: var(--dark);
```

```
    transition: 0.5s;}
@media (min-width: 992px) {
    .sidebar {
        margin-left: 0;}
    .sidebar.open {
        margin-left: -250px;}
    .content {
        width: calc(100% - 250px);}
    .content.open {
        width: 100%;
        margin-left: 0;}}
@media (max-width: 991.98px) {
    .sidebar {
        margin-left: -250px;}
    .sidebar.open {
        margin-left: 0;}
    .content {
        width: 100%;
        margin-left: 0; }}
/*** Navbar ***/
.sidebar .navbar .navbar-nav .nav-link {
    padding: 7px 20px;
    color: var(--light);
    font-weight: 500;
    border-left: 3px solid var(--secondary);
    border-radius: 0 30px 30px 0;
    outline: none;}
.sidebar .navbar .navbar-nav .nav-link:hover,
.sidebar .navbar .navbar-nav .nav-link.active {
    color: var(--primary);
```

```
    background: var(--dark);
    border-color: var(--primary);}
.sidebar .navbar .navbar-nav .nav-link i {
    width: 40px;
    height: 40px;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    background: var(--dark);
    border-radius: 40px;}
.sidebar .navbar .navbar-nav .nav-link:hover i,
.sidebar .navbar .navbar-nav .nav-link.active i {
    background: var(--secondary);}
.sidebar .navbar .dropdown-toggle::after {
    position: absolute;
    top: 15px;
    right: 15px;
    border: none;
    content: "\f107";
    font-family: "Font Awesome 5 Free";
    font-weight: 900;
    transition: .5s;}
.sidebar .navbar .dropdown-toggle[aria-expanded=true]::after {
    transform: rotate(-180deg);}
.sidebar .navbar .dropdown-item {
    padding-left: 25px;
    border-radius: 0 30px 30px 0;
    color: var(--light);}
.sidebar .navbar .dropdown-item:hover,
.sidebar .navbar .dropdown-item.active {
```

```
    background: var(--dark);}
.content .navbar .navbar-nav .nav-link {
    margin-left: 25px;
    padding: 12px 0;
    color: var(--light);
    outline: none;}
.content .navbar .navbar-nav .nav-link:hover,
.content .navbar .navbar-nav .nav-link.active {
    color: var(--primary);}
.content .navbar .sidebar-toggler,
.content .navbar .navbar-nav .nav-link i {
    width: 40px;
    height: 40px;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    background: var(--dark);
    border-radius: 40px;}
.content .navbar .dropdown-item {
    color: var(--light);}
.content .navbar .dropdown-item:hover,
.content .navbar .dropdown-item.active {
    background: var(--dark);}
.content .navbar .dropdown-toggle::after {
    margin-left: 6px;
    vertical-align: middle;
    border: none;
    content: "\f107";
    font-family: "Font Awesome 5 Free";
    font-weight: 900;
```

```
    transition: .5s;}
.content .navbar .dropdown-toggle[aria-expanded=true]::after {
    transform: rotate(-180deg);}
@media (max-width: 575.98px) {
    .content .navbar .navbar-nav .nav-link {
        margin-left: 15px; }}
/*** Date Picker ***/
.bootstrap-datetimepicker-widget.bottom {
    top: auto !important;}
.bootstrap-datetimepicker-widget .table * {
    border-bottom-width: 0px;}
.bootstrap-datetimepicker-widget .table th {
    font-weight: 500;}
.bootstrap-datetimepicker-widget.dropdown-menu {
    padding: 10px;
    border-radius: 2px;}
.bootstrap-datetimepicker-widget table td.active,
.bootstrap-datetimepicker-widget table td.active:hover {
    background: var(--primary);}
.bootstrap-datetimepicker-widget table td.today::before {
    border-bottom-color: var(--primary);}
/*** Testimonial ***/
.progress .progress-bar {
    width: 0px;
    transition: 2s;}
/*** Testimonial ***/
.testimonial-carousel .owl-dots {
    margin-top: 24px;
    display: flex;
    align-items: flex-end;
```

```
    justify-content: center;}
.testimonial-carousel .owl-dot {
    position: relative;
    display: inline-block;
    margin: 0 5px;
    width: 15px;
    height: 15px;
    border: 5px solid var(--primary);
    border-radius: 15px;
    transition: .5s;}
    border-color: var(--primary);}
```

ДОДАТОК Д

Програмні коди singin.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>DarkPan - Bootstrap 5 Admin Template</title>
    <meta content="width=device-width, initial-scale=1.0" name="viewport">
    <meta content="" name="keywords">
    <meta content="" name="description">
    <!-- Favicon -->
    <link href="img/favicon.ico" rel="icon">
    <!-- Google Web Fonts -->
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;600&fami
ly=Roboto:wght@500;700&display=swap" rel="stylesheet">
    <!-- Icon Font Stylesheet -->
    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css" rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-
icons.css" rel="stylesheet">
    <!-- Libraries Stylesheet -->
    <link href="lib/owlcarousel/assets/owl.carousel.min.css" rel="stylesheet">
    <link href="lib/tempusdominus/css/tempusdominus-bootstrap-4.min.css"
rel="stylesheet" />
    <!-- Customized Bootstrap Stylesheet -->
    <link href="css/bootstrap.min.css" rel="stylesheet">
```

```
    <!-- Template Stylesheet -->
    <link href="css/style.css" rel="stylesheet">
</head>
<body>
    <div class="container-fluid position-relative d-flex p-0">
        <!-- Spinner Start -->
        <div id="spinner" class="show bg-dark position-fixed translate-middle w-100
vh-100 top-50 start-50 d-flex align-items-center justify-content-center">
            <div class="spinner-border text-primary" style="width: 3rem; height:
3rem;" role="status">
                <span class="sr-only">Loading...</span>
            </div>
        </div>
        <!-- Spinner End -->
        <!-- Sign In Start -->
        <div class="container-fluid">
            <div class="row h-100 align-items-center justify-content-center"
style="min-height: 100vh;">
                <div class="col-12 col-sm-8 col-md-6 col-lg-5 col-xl-4">
                    <div class="bg-secondary rounded p-4 p-sm-5 my-4 mx-3">
                        <div class="d-flex align-items-center justify-content-between mb-
3">
                            <a href="index.html" class="">
                                <h3 class="text-primary"><i class="fa fa-user-edit me-
2"></i>DarkPan</h3>
                            </a>
                            <h3>Sign In</h3>
                        </div>
                        <div class="form-floating mb-3">
                            <input type="email" class="form-control" id="floatingInput"
```

```
placeholder="name@example.com">
                    <label for="floatingInput">Email address</label>
                </div>
                <div class="form-floating mb-4">
                    <input type="password" class="form-control"
id="floatingPassword" placeholder="Password">
                    <label for="floatingPassword">Password</label>
                </div>
                <div class="d-flex align-items-center justify-content-between mb-
4">
                    <div class="form-check">
                        <input type="checkbox" class="form-check-input"
id="exampleCheck1">
                        <label class="form-check-label" for="exampleCheck1">Check
me out</label>
                    </div>
                    <a href="">Forgot Password</a>
                </div>
                <button type="submit" class="btn btn-primary py-3 w-100 mb-
4">Sign In</button>
                <p class="text-center mb-0">Don't have an Account? <a
href="">Sign Up</a></p>
            </div>
        </div>
    </div>
    <!-- Sign In End -->
</div>
</html>
```

ДОДАТОК Е

ПРОТОКОЛ

ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: _____

_____

_____

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ кафедра обчислювальної техніки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 96,7% Схожість 3,3%

Аналіз звіту подібності (відмітити потрібне):

- ✓ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.
- □ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
- □ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Захарченко С.М.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Швець Є.О.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Черняк О.І.
(підпис) (прізвище, ініціали)