UNICHECK

Ім'я користувача:
**приховано налаштуваннями конфіденційності**

ID перевірки:
**1015521889**

Дата перевірки:
**09.06.2023 10:27:03 EEST**

Тип перевірки:
**Doc vs Library**

Дата звіту:
**12.06.2023 08:45:31 EEST**

ID користувача:
**100011372**

Назва документа: **ОК42 Куспісь Яромир**

Кількість сторінок: **38** Кількість слів: **8194** Кількість символів: **60773** Розмір файлу: **1.08 MB** ID файлу: **1015175986**

## 2.61%
## Схожість

Найбільша схожість: **1.16%** з джерелом з Бібліотеки (ID файлу: **1004053580**)

Пошук збігів з Інтернетом не проводився

2.61% Джерела з Бібліотеки | 35 ..... Сторінка 40

## 0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

## 0%
## Вилучень

Немає вилучених джерел

# 1 ОГЛЯД СУЧАСНИХ CMS СИСТЕМ

## 1.1 Характеристики сучасних CMS систем

Для того, щоб детально розглянути можливості та класифікацію систем управління контентом (CMS), слід дати визначення CMS та її функціональності.

CMS – це програмне забезпечення, яке дозволяє публікувати та редагувати інформацію на сайті без залучення розробників та спеціалізованого персоналу. Від користувача не потрібні особливі знання технологій, відмінних від базових комп'ютерних навичок [1].

Сучасні CMS складаються з двох частин: back-end, яка відповідає за функціональність, обробку та зберігання інформації, та front-end, орієнтованої на роботу з користувачем. Деякі CMS мають можливість створення зовнішнього вигляду сторінок з заздалегідь розроблених дизайн-шаблонів, які можуть бути як платними, так і безплатними, такі CMS полегшують процес створення та підтримки сайтів [2]. Останнім часом спостерігається тенденція до спрощення процесу створення шаблонів та ручної верстки сайтів. На зміну звичним текстовим редакторам приходять так звані Drag&Drop інтерфейси, в яких у більшості випадків інтуїтивно зрозумілі користувачеві та не вимагають спеціальної підготовки та особливих технічних знань.

У цьому розділі детально розглянуто технологічну основу систем управління контентом (CMS) та ключові технології та інструменти, які використовуються при розробці CMS.

Технологічна основа CMS передбачає використання різних технологій та інструментів, які забезпечують її ефективну та надійну роботу. Основними технологіями, які використовуються при розробці CMS, є:

1. Бази даних. Вони є невід'ємною частиною CMS, оскільки вони використовуються для зберігання та організації контенту. Найпоширеніші бази даних, які використовуються для CMS, це MySQL та PostgreSQL.

2. PHP. Ця мова програмування є однією з найпоширеніших для розробки CMS. Вона дозволяє створювати динамічний контент та забезпечує можливість легко розширювати та модифікувати CMS.

3. JavaScript. Він використовується для створення динамічної та інтерактивної вебсторінки, що дозволяє покращити користувацький досвід динамічним вебсайтом.

4. HTML та CSS. Це є основою веброзробки та використовуються для створення структури та дизайну вебсторінок.

5. AJAX. Він використовується для створення асинхронних запитів на сервер, що дозволяє оновлювати частини вебсторінки без перезавантаження сторінки.

Крім того, для розробки CMS можуть використовуватись такі інструменти, як фреймворки, CMS-генератори, системи контролю версій. Фреймворки дозволяють спростити процес розробки, забезпечуючи готову структуру проєкту, набір інструментів та бібліотек, які дозволяють швидше та ефективніше створювати CMS.

CMS-генератори, такі як Drupal або WordPress, забезпечують готову структуру CMS та набір основних функцій, що дозволяє зосередитися на розробці унікальних функцій та дизайну вебсторінок.

Системи контролю версій, такі як Git, дозволяють зберігати та керувати версіями коду CMS, що дозволяє легко повертатися до попередніх версій, вносити зміни та співпрацювати з іншими розробниками.

Розглядаючи процес розробки CMS та його ключові етапи, їх можна поділити на:

- Проєктування архітектури;
- Розробка функціональності;
- Впровадження системи.

Проєктування архітектури є першим етапом розробки CMS, під час якого визначається структура проєкт, вибір технологій та інструментів, організація бази даних та структури даних, а також розробка концепції дизайну та користувацького

інтерфейсу [3]. На рисунку 1.1 зображено схематичну архітектуру типової традиційної CMS, де наведено основні компоненти, такі як база даних, вебсервер та користувацький інтерфейс.




Рисунок 1.1 – Схематична архітектура типової CMS

Розробка функціональності є наступним етапом та передбачає розробку ключових функцій CMS, таких як керування контентом, авторизація та автентифікація, адміністративна панель, налаштування та розширення функціональності [4].

Впровадження є останнім етапом розробки CMS, під час якого вона готується до релізу та використання. Цей етап передбачає встановлення та налаштування CMS на сервері, перенесення даних з тестового середовища, налаштування домену та підключення до інших систем [5].

Оскільки розробка CMS є складним та трудомістким процесом, важливо дотримуватися кращих практик розробки, таких як написання чистого та оптимізованого коду та забезпечення безпеки.

Крім того, під час розробки CMS необхідно враховувати вимоги до проєкту, дотримуватися стандартів безпеки та якості, та регулярно оновлювати та покращувати CMS з урахуванням змін в технологіях та потребах майбутніх користувачів.

Як зазначається в книзі "Professional Content Management Systems: Handling Digital Media Assets" [6], розробка CMS охоплює не тільки процес розробки, а й постійне управління та підтримку. Важливим елементом є створення системи моніторингу та відстеження помилок, що дозволить оперативно розв'язувати проблеми та уникати їх у майбутньому. Також важливо забезпечити підтримку та оновлення безпеки системи, в тому числі регулярне оновлення CMS та її складових, встановлення необхідних патчів та заходів для забезпечення безпеки даних.

Отже, розробка CMS – це складний процес, який потребує використання кращих практик розробки та врахування потреб користувачів. Важливо не зупинятися на досягнутому та постійно покращувати CMS з урахуванням нових технологій та потреб користувачів. У цьому розділі було детально розглянуто технологічну основу CMS, що може охоплювати використання фреймворків, CMS-генераторів та систем контролю версій такий як Git. Також було розглянуто ключові етапи розробки CMS, такі як проєктування архітектури, розробка функціональності та впровадження. Важливо дотримуватися кращих практик розробки та враховувати вимоги до проєкту під час створення таких систем.

### 1.2 Огляд популярних CMS систем

При розробці вебсайту одним з ключових завдань є вибір відповідної системи управління контентом (CMS). На сьогодні існує безліч CMS, що створюються з різними цілями та функціональними можливостями. На рисунку 1.2 зображено діаграму найпопулярніших CMS систем доступних на ринку.

Рисунок 1.2 – Найпопулярніші CMS

У проєкті було порівняно популярні CMS та зроблено висновок про їх переваги та недоліки, а саме:

1. WordPress.

WordPress є однією з найпопулярніших CMS у світі. Ця система проста в використанні та дозволяє створювати сайти будь-якої складності. Вона має безліч тем та плагінів, що дозволяють змінювати дизайн та функціонал вебсайту, а також покращують його SEO. Крім того, WordPress безплатний, що робить його доступним для багатьох користувачів [7].

Проте, зважаючи на свою популярність, WordPress став частою мішенню хакерів та зловмисників. Це може бути проблемою для користувачів, які не мають досвіду в забезпеченні безпеки вебсайту. Крім того, деякі теми та плагіни можуть бути платними, що збільшує вартість створення вебсайту. Також, якщо вебсайт потребує складних функцій, WordPress може не бути найкращим варіантом.

2. Joomla.

Joomla є іншою популярною CMS, що дозволяє створювати вебсайти будь-якої складності. Вона має велику спільноту розробників та користувачів, що

забезпечує швидкий розвиток та підтримку системи. Joomla має безліч тем та плагінів, що дозволяють змінювати дизайн та функціонал вебсайту. Крім того, Joomla має вбудовану систему SEO, що сприяє покращенню рейтингу вебсайту[8].

3. Drupal.

Drupal є ще однією з популярних CMS, яка зазвичай використовується для створення складних вебсайтів, таких як інтернет-магазини, форуми та соціальні мережі. Drupal має потужну систему прав доступу, що дозволяє керувати доступом користувачів до контенту сайту. Вона має безліч модулів, які дозволяють додавати різні функції до вебсайту. Крім того, Drupal є досить надійним і безпечним CMS, що робить його хорошим варіантом для бізнес-вебсайтів[9].

4. Magento.

Magento є CMS, спеціально розробленою для створення інтернет-магазинів. Вона має великий набір функцій, що дозволяють налаштовувати товари, замовлення, оплату та доставлення. Magento також має потужну систему SEO, що дозволяє вебсайту бути високо в результатах пошуку[10].

Проте, Magento важка для розгортання та підтримки у майбутньому.

5. Власна CMS.

Одним з варіантів розробки вебсайту є створення власної CMS. Це дає можливість створити систему, яка повністю відповідає потребам та вимогам проєкту. Власна CMS може бути більш ефективною в тих випадках, коли такою системою потрібно вирішити особливі завдання.

Створення власної CMS це складний процес, проте за наявності відповідних знань та навичок, це вигідніше ніж використовувати готові комерційні рішення.

6. WIX.

WIX є вебплатформою, що дозволяє створювати сайти без необхідності програмування. Вона має безліч шаблонів та інструментів, які дозволяють швидко та просто створювати вебсайти будь-якої складності. Крім того, WIX як і інші має вбудований функціонал SEO, що дозволяє вебсайту бути високо в результатах пошуку.

Проте, використання WIX є обмеженим у порівнянні зі складнішими CMS. Також, деякі функції та можливості можуть бути доступними лише за певну плату.

7. Shopify.

Shopify спеціально розроблений для створення інтернет-магазинів. Він має безліч функцій, що дозволяють налаштовувати товари, замовлення, оплату та доставлення. Shopify також має вбудований функціонал SEO, що дозволяє вебсайту бути вище в результатах пошуку. Крім того, Shopify має безліч шаблонів та плагінів, що дозволяють змінювати дизайн та функціонал вебсайту.

Проте, використання Shopify може не зовсім підходити для вебсайтів, що не є інтернет-магазинами. Також, Shopify є платним, що збільшує вартість створення вебсайту.

8. Squarespace.

Squarespace дозволяє створювати вебсайти будь-якої складності без необхідності програмування. Він має безліч шаблонів та інструментів, що дозволяють швидко та просто створювати вебсайти. Squarespace також має вбудований функціонал SEO та аналітики, що дозволяє вебсайту знаходитись високо в результатах пошуку та отримувати детальну інформацію про відвідувачів.

Отже, кожна CMS має свої переваги та недоліки. WordPress – це хороший варіант для створення вебсайтів будь-якої складності, Joomla – для середньої складності вебсайтів, Drupal – для складних вебсайтів а Magento та Shopify – для інтернет-магазинів. WIX є хорошим варіантом для створення вебсайтів будь-якої складності без необхідності програмування. Shopify хороший варіантом для створення інтернет-магазинів, тому що він має безліч функцій, що дозволяють налаштовувати товари, замовлення, оплату та доставлення. Squarespace підходить для створення вебсайтів будь-якої складності без необхідності програмування. Він має безліч шаблонів та інструментів, що дозволяють швидко та просто створювати вебсайти.

При виборі CMS для вебсайту, необхідно враховувати бюджет проєкту, потреби та вимоги. Також варто врахувати майбутні плани розвитку вебсайту,

оскільки перехід з однієї CMS на іншу може бути досить складним та дорогим процесом. Тому, враховуючи що буде розроблятись серверна частина CMS системи для начальної системи, є обґрунтованим рішення використовувати власну CMS систему, щоб можна було реалізувати власну серверну частину проєкту, чого не дають інші комерційні CMS системи, або дають з обмеженнями.

### 1.3 Налаштування і підтримка вебсайту побудованого на CMS.

Для належної роботи вебсайту на CMS потрібно налаштувати базу даних, яка відповідає за зберігання та організацію контенту [11]. Найпоширеніші бази даних, які використовуються для CMS, це MySQL та PostgreSQL. Крім того, вебсервер повинен бути налаштований на роботу з CMS. Саме налаштування вебсервера передбачає встановлення необхідних пакетів, конфігурацію вебсервера та налаштування прав доступу.

Підтримка вебсайту на CMS передбачає багато аспектів, таких як стеження за оновленнями та забезпечення безпеки. Своєчасне оновлення CMS мінімізує проблеми з безпекою та підвищує продуктивність. Крім того, щоб забезпечити безпеку, потрібно використовувати надійні паролі та регулярно робити резервну копію даних на хостингу. Важливо визначити основні загрози та ризики, які можуть виникнути при роботі вебсайту на CMS, та прийняти заходи щодо їх запобігання. Оскільки безпека — це постійний процес, підтримка вебсайту на CMS має бути постійною. Це дозволяє забезпечити безпеку, яка є однією з найбільш важливих складових вебсайту на CMS.

Розробка вебсайту на CMS передбачає використання різних технологій та інструментів, таких як PHP, JavaScript, HTML та CSS. Потрібно мати розуміння основних принципів роботи з CMS, знати необхідні технології та інструменти, але не обмежуватися лише ними. Важливо також розуміти вимоги проєкту та уважно їх аналізувати. Це дозволяє створювати вебсайти, які не просто працюють належним чином, але й задовольняють поставлені перед ними цілі.

Під час розробки вебсайту на CMS важливо враховувати потреби майбутніх користувачів та їх очікування від системи. Цього можна досягти, консультуючись з керівником проєкту, а також дивлячись на приклади успішних CMS, щоб розробити та забезпечити зручний та інтуїтивно зрозумілий інтерфейс для користувачів. Також для цього можуть використовуватися різні методи тестування та аналізу користувацького досвіду, наприклад, тестування з використанням A/B-тестів чи опитування користувачів. Це дозволяє покращити якість вебсайту та зробити його більш зручним та зрозумілим для користувачів.

У процесі розробки вебсайту на CMS важливо враховувати потреби майбутніх користувачів та мати стратегію її розвитку. Це дозволяє створювати вебсайт, який відповідає потребам бізнесу та допомагає досягти поставлених цілей. Також варто пам'ятати, що вебсайт на CMS потребує постійної підтримки та розвитку, тому важливо мати план на майбутнє, що охоплює як технічні, так і стратегічні питання [11].

Крім того, використання CMS для побудови вебсайту дозволяє ефективно управляти контентом та ресурсами, що дозволяє прискорити роботу та збільшити продуктивність вебсайту. Проте, використання CMS має свої недоліки, такі як обмеження в можливостях редагування дизайну вебсайту та можливі виникнення проблем з безпекою при використанні старіших версій CMS.

У підсумку, використання CMS є ефективним інструментом для розробки та підтримки вебсайту. При правильному використанні вона може допомогти збільшити продуктивність та прискорити роботу вебсайту, а також забезпечити зручний та інтуїтивно зрозумілий інтерфейс для користувачів. Проте, для успішної роботи на CMS потрібно мати професійний підхід та ретельно продуману стратегію. Вебсайт на CMS має відповідати потребам проєкту та користувачів, бути зручним та безпечним. Актуальність та оновленість CMS, а також постійна підтримка та розвиток вебсайту — це ключові фактори успішної CMS системи.

## 2 ВИБІР ЗАСОБІВ РОЗРОБКИ CMS СИСТЕМИ

### 2.1 Вибір мови програмування для CMS

Вибір мови програмування для розробки CMS системи є важливим кроком, від якого залежить багато речей у проєкті. Однією з найпопулярніших мов програмування для створення CMS систем є PHP, який широко використовується у веброзробці [12]. У цьому розділі розглянуто основні фактори, що слід враховувати при виборі мови програмування для розробки CMS системи, зокрема PHP.

Розвиток мов програмування просувається величезною швидкістю. З роками з'являються нові мови програмування, фреймворки та бібліотеки, що спрощують життя розробників. Хтось вважає, що мови програмування, яким вже понад 25 років, стали застарілими та непридатними для створення сучасних застосунків. Однак, насправді ситуація має інший вигляд.

Мови програмування продовжують удосконалюватись та ставати кращими із часом. Навіть мови програмування, яким понад 60 років, досі використовуються у ряді випадків. Наприклад, FORTRAN, створений компанією IBM у 50-х роках XX століття, є однією з таких старих, але ще досі актуальних мов програмування.

Подібною по старовині мовою, що має трохи менше ніж 30 років, є PHP, який з'явився у 1995 році. Цю мову можна назвати першою динамічною мовою програмування для створення динамічних вебсайтів. 80% вебсайтів у світі написано на PHP, що свідчить про його стабільну популярність серед розробників протягом років.

Щорічно проводяться численні конференції, де збираються розробники, програмісти та студенти, що працюють з PHP. Вважається, що ця мова програмування буде розвиватися та удосконалюватися ще довгий час. Тому вибір саме цієї мови програмування є обґрунтованим, також при виборі PHP увага зверталась на такі характеристики:

1. Компетенції розробників: навички та досвід команди розробників відіграють важливу роль у виборі мови програмування. PHP є простим у навчанні

та має велику кількість ресурсів, що сприяє легкому вивченню та підтримці продукту [13].

2. Специфіка проєкту: різні мови програмування мають свої сильні та слабкі сторони. Наприклад, PHP підходить для створення вебсайтів з динамічним контентом та базами даних таких як CMS система, але, наприклад, може бути менш пригідним для створення розширюваних вебдодатків з інтенсивними обчисленнями [14].

3. Вимоги до продуктивності та масштабованості: Різні мови програмування мають різні характеристики продуктивності та масштабованості. PHP має хорошу продуктивність для вебсайтів та додатків середнього розміру, але може мати обмеження при роботі з великими системами [15].

4. Екосистема: Вибір мови програмування також залежить від наявності бібліотек, фреймворків та інструментів, що спрощують процес розробки та підтримки CMS системи. PHP має широку екосистему з великою кількістю фреймворків (наприклад, Laravel, Symfony, CodeIgniter), бібліотек та інструментів, що дозволить прискорити роботу та створювати масштабовані та безпечні вебсайти [16].

5. Інтеграція з іншими технологіями: Мова програмування повинна добре інтегруватися з іншими технологіями, що використовуються у проєкті, такими як база даних та сервер. PHP добре працює з різними базами даних (наприклад, MySQL, PostgreSQL, SQLite) та вебсерверами (наприклад, Apache, Nginx), що робить його гнучким відносно інтеграції та відмінним вибором. [17].

6. Спільнота: Наявність активної спільноти розробників є важливим критерієм при виборі мови програмування, оскільки це забезпечує підтримку, доступність ресурсів для навчання та розвитку. PHP має велику спільноту розробників, яка активно підтримує мову, створює нові бібліотеки, фреймворки та інструменти.

7. Безпека: Безпека є важливим аспектом у розробці вебсайтів та додатків, особливо для CMS систем, які у деяких випадках можуть обробляти персональні дані користувачів та ділову інформацію. PHP має вбудовані функції

безпеки, а також додаткові бібліотеки та фреймворки, які допомагають у захисті від загроз та забезпечують захист даних [18].

8. Ліцензія та вартість: Розробка CMS системи на PHP є відносно економічною, оскільки PHP розповсюджується вільно. Це означає, що можна використовувати PHP без витрат на ліцензії або обмежень щодо використання його в комерційних проєктах.

9. Багатоплатформність: PHP є багатоплатформенною мовою програмування, що дозволяє розробляти вебсайти та додатки, які можуть працювати на різних платформах та операційних системах. Це спрощує розгортання та міграцію CMS системи на різні сервери чи хмарні рішення [19].

10. Підтримка довгострокового розвитку: Вибір мови програмування повинен також враховувати довгострокові перспективи розвитку та підтримки системи. PHP, з його стабільною спільнотою та активним розвитком, є хорошим варіантом для створення CMS систем, які потребують постійної підтримки та розвитку [20].

Також вибираючи мову програмування PHP потрібно звернути увагу на ООП. ООП – це абревіатура, яка розшифровується як об'єктно-орієнтоване програмування. В ООП все будується на класах, методах та об'єктах. Клас – це свого роду шаблон, який створюється для того, щоб потім на його основі створити об'єкт. Класи мають свої функції та методи, яким можна задавати область видимості. ООП базується на поліморфізмі, інкапсуляції та наслідуванні. Розгляньмо всі ці поняття:

1. Поліморфізм – це здатність об'єктів різних класів працювати з одним інтерфейсом, дозволяючи тим самим використовувати однакові методи для різних типів даних. Поліморфізм полегшує розробку програмного коду, роблячи його більш гнучким та зручним для модифікацій.
2. Інкапсуляція – це принцип, який об'єднує дані та функції, що працюють з цими даними, у єдиний компонент, зазвичай клас. Інкапсуляція дозволяє приховати деталі реалізації від користувача класу, тим самим ізолюючи взаємодію між різними компонентами системи.

3. Наслідування – це процес, за допомогою якого один клас (дочірній) може отримати властивості та методи іншого класу (батьківського). Наслідування дозволяє уникати дублювання коду, спрощує модифікацію програмного коду та підвищує його повторне використання.

Враховуючи все вищесказане, можна зробити висновок, що PHP є найкращим та обґрунтованим вибором мови програмування для розробки CMS системи. Завдяки широкому спектру переваг, таких як простота вивчення, велика спільнота розробників, гнучкість інтеграції, відмінна екосистема, безпека, портабельність та довгострокова підтримка, а також ООП. PHP точно може задовольнити потреби більшості проєктів, пов'язаних з розробкою CMS систем в тому числі й цього.

### 2.2 Вибір бази даних для CMS

При розробці CMS-системи одним з ключових аспектів є вибір бази даних для зберігання інформації та управління контентом. Важливо обрати базу даних, яка відповідає потребам проєкту, має хорошу продуктивність та масштабується згідно з потребами проєкту. У цьому розділі розглядається вибір MySQL як бази даних для CMS.

MySQL є однією з найпопулярніших відкритих систем управління базами даних, яка використовує SQL (Structured Query Language) для обробки та отримання даних. Вона відома своєю надійністю, продуктивністю та легкістю використання. Для підключення MySQL буде використовуватись середовище розробки phpMyAdmin, зображення якої показано на рисунку 2.1.

Рисунок 2.1 – Середовище розробки phpMyAdmin

MySQL з'явилася як спроба використати mSQL для внутрішніх потреб компанії, а саме для таблиць з використанням ISAM – низькорівневих підпрограм для індексного доступу до даних. В результаті було створено новий SQL-інтерфейс, хоча API-інтерфейс залишився від mSQL.

Логотип MySQL у вигляді дельфіна має ім'я "Sakila". Він був обраний з-поміж пропозицій простих користувачів щодо імені дельфіна.

MySQL є потужним та популярним вибором для бази даних при розробці CMS системи. Вона надійна, продуктивна та має можливість масштабуватись, що відповідає вимогам багатьох проєктів. Однак, слід враховувати недоліки при виборі бази даних для CMS. Якщо розглядати основні переваги та недоліки MySQL для побудови CMS, то вони включають:

- Висока продуктивність: MySQL оптимізована для високошвидкісної обробки запитів і має низьку навантаженість на ресурси сервера;
- Оптимізація запитів: MySQL має потужну оптимізацію запитів, яка допомагає забезпечити максимальну продуктивність та ефективність обробки даних. Вона автоматично аналізує запити та використовує оптимальний план виконання для швидкого отримання результатів [21];

- Масштабованість: MySQL легко масштабується для підтримки вебсайтів з великою кількістю користувачів та великим обсягом даних;
- Підтримка різних типів таблиць: MySQL дозволяє використовувати різні типи таблиць, такі як MyISAM, InnoDB, MEMORY, NDB Cluster та інші. Кожен з цих типів таблиць має свої властивості та характеристики, які можуть бути потрібні для різних сценаріїв використання [22];
- Візуальні інструменти керування: Для спрощення роботи з MySQL доступні різні візуальні інструменти керування, такі як phpMyAdmin, MySQL Workbench, Adminer та інші. Вони допомагають адміністраторам баз даних та розробникам легко керувати структурою бази даних, даними та налаштуваннями [23];
- Оновлення без втрати даних: MySQL дозволяє здійснювати оновлення без втрати даних. Це забезпечує безперебійну роботу вебдодатків та CMS під час оновлення та розширення функціонала бази даних [24];
- Безпека: MySQL пропонує різні рівні доступу до даних та безпеки, що дозволяє захищати інформацію та контролювати доступ до неї [25];
- Відкритий код: MySQL є відкритою системою, що дозволяє розробникам адаптувати її до своїх потреб та використовувати безплатно [26].

Однак, MySQL також має деякі недоліки, які слід враховувати при виборі бази даних для CMS:

- Обмежена підтримка транзакцій: MySQL має обмежену підтримку транзакцій, що може стати проблемою для деяких CMS-проєктів, особливо для тих, які потребують складніших операцій з даними [27];
- Відсутність підтримки NoSQL: MySQL є реляційною базою даних і не підтримує технології NoSQL, що може бути обмеженням для деяких варіантів використання, коли проєкту потрібна гнучкість структури даних [28];
- Відносно повільна робота з документами XML та JSON: Попри те, що MySQL підтримує зберігання та обробку даних у форматах XML та JSON, вона може працювати повільніше, ніж спеціалізовані бази даних для цих форматів [29].

Завдяки доступним візуальним інструментам керування, таким як phpMyAdmin, MySQL Workbench та Adminer, адміністратори баз даних та розробники можуть легко керувати структурою бази даних, даними та налаштуваннями. MySQL також підтримує оновлення без втрати даних, забезпечуючи безперебійну роботу вебдодатків та CMS під час оновлення та розширення функціонала бази даних.

MySQL має потужний оптимізатор запитів, який аналізує SQL-запити та визначає найефективніший спосіб виконання. Розробники можуть використовувати підказки для оптимізатора, щоб керувати його поведінкою та покращувати продуктивність. Крім того, MySQL підтримує рівень аудиту безпеки, який дозволяє адміністраторам відстежувати зміни в доступі до даних, зміни структури бази даних та інші події, що стосуються безпеки.

MySQL також має систему резервного копіювання, яка дозволяє створювати резервні копії баз даних, що можуть бути відновлені в разі втрати даних або пошкодження. Різні типи резервних копій, такі як логічні, фізичні та згортки, можуть бути використані для різних сценаріїв відновлення. Окрім того, MySQL підтримує обробку просторових даних, що дозволяє зберігати, індексувати та обробляти геометричні об'єкти. Однак під час розробки CMS, слід враховувати недоліки MySQL, такі як обмежена підтримка транзакцій та відсутність підтримки NoSQL та відносно повільна робота з документами XML та JSON.

Враховуючи все вищевказане, MySQL є відмінним вибором для розробки CMS, завдяки своїй гнучкості, масштабованості, продуктивності та відкритому коду. Враховуючи популярність PHP та MySQL на ринку веброзробки, це поєднання технологій забезпечує широку спільноту розробників та доступ до багатьох ресурсів для підтримки та розвитку проєкту. Використання MySQL для CMS дає можливість отримати стабільну, продуктивну та гнучку систему, що робить цей вибір обґрунтованим.

# 3 РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ CMS СИСТЕМИ

## 3.1 Створення бази даних

Одним з ключових етапів розробки серверної частини CMS системи є створення бази даних для зберігання контенту. У цьому розділі буде розглянуто процес створення бази даних MySQL на локальному сервері, а також через вебпанель cPanel.

Для роботи з базою даних необхідно запустити клієнтську програму MySQL. Потрібно відкрити термінал на комп'ютері та ввести наступну команду:

mysql -u ім'я_користувача –p

Де "ім'я_користувача" повинно бути замінено на реальне ім'я користувача, який створює базу даних. Згодом потрібно буде ввести пароль, коли він буде запитаний. З метою створення нової бази даних, введіть наступну команду в командному рядку MySQL:

create database cms;

Натисніть Enter для підтвердження команди. Тепер є нова порожня база даних, до якої можна додати таблиці та контент. Після створення бази даних можна завершити роботу з клієнтською програмою MySQL, ввівши наступну команду:

Exit

Також потрібно натиснути Enter для виходу з програми.

Якщо розглядати створення через вебпанель, то на деяких вебсерверах бази даних створюються за допомогою вебінструментів, таких як cPanel.

cPanel – це комерційна вебпанель керування хостингом, що дозволяє адміністраторам та користувачам керувати своїми вебсерверами та хостинговими обліковими записами через зручний графічний інтерфейс. cPanel надає можливість створювати та керувати базами даних, електронними скриньками, доменами, FTP-аккаунтами та іншими компонентами вебсервера. На рисунку 3.1 зображена головна панель керування cPanel.

Рисунок 3.1 – Головна панель керування cPanel

Однією з особливостей cPanel є інтегрований менеджер баз даних, який дозволяє створювати, модифікувати та видаляти бази даних безпосередньо через вебінтерфейс. Також він надає доступ до інших інструментів керування базами даних, таких як phpMyAdmin, який дозволяє керувати таблицями, індексами, запитами та іншими аспектами баз даних MySQL.

Для створення бази даних через cPanel, спочатку потрібно увійти в панель керування свого хостингового облікового запису. Далі потрібно знайти розділ "Бази даних" або "Databases", який зазвичай розташований в головному меню або в бічній панелі інструментів. Після цього, вибрати пункт "MySQL бази даних" або "MySQL Databases". В цьому розділі можна створити нову базу даних, вказавши її назву в спеціальному полі, а потім натиснувши кнопку "Створити базу даних" або "Create Database". Після того, як база даних буде створена, можна буде додати користувача. Це можна зробити, вказавши ім'я користувача, вибравши пароль і визначивши відповідні привілеї. Таким чином, cPanel значно спрощує процес створення та керування базами даних на вебсервері, надаючи графічний інтерфейс, який робить ці операції доступними для користувачів з різним рівнем знань та досвіду, незалежно від їх технічної обізнаності.

### 3.2 Створення таблиць бази даних та їх завантаження

Для навчальної CMS системи будуть використовуватись лише дві таблиці бази даних: статті (articles), а також категорії (categories), які будуть зберігати всі статті(публікації) та категорії статей в системі. Створимо схему для таблиці, яка описує типи даних, які може зберігати таблиця, а також іншу інформацію про таблицю. Спочатку має бути створено текстовий файл під назвою tables.sql та додано наступний код:

```
DROP TABLE IF EXISTS articles;
CREATE TABLE articles(
  id              smallint unsigned NOT NULL auto_increment,
  publicationDate date NOT NULL,
  categoryId      smallint unsigned NOT NULL,
  title           varchar(255) NOT NULL,
  summary         text NOT NULL,
  content         mediumtext NOT NULL,
  PRIMARY KEY     (id)
); DROP TABLE IF EXISTS categories;
CREATE TABLE categories(
  id              smallint unsigned NOT NULL auto_increment,
  name            varchar(255) NOT NULL,
  description     text NOT NULL,
  PRIMARY KEY     (id));
```

Цей код визначає схему таблиць статей. Він написаний на SQL, мові, яка використовується для створення та керування базами даних у MySQL.

Після створення схем таблиць необхідно завантажити їх в MySQL, щоб створити самі таблиці. Найпростіший спосіб зробити це – відкрити вікно термінала, перейти до папки, що містить файл tables.sql, і виконати цю команду:

```
mysql -u username -p cms < tables.sql
```

де “username” – ім'я користувача MySQL. “cms” – назва бази даних, яку було створено у пункті 3.1.

Також потрібно буде ввести пароль при введенні запиту. MySQL завантажить та виконає код у файлі tables.sql, створюючи таблицю статей в базі даних cms. Також можна використовувати вебадміністратор, наприклад phpMyAdmin, для запуску коду tables.sql та створення таблиці. phpMyAdmin доступний у більшості хостинг провайдерів. Нижче наведено детальну структуру таблиці статей(articles):

- Id: Поле id має тип даних smallint unsigned, що дозволяє зберігати цілі числа від 0 до 65 535. Це дозволяє CMS системі мати до 65 535 статей. Також вказується атрибут NOT NULL, що означає, що поле не може бути порожнім (null). Додатково додаємо атрибут auto_increment, який каже MySQL автоматично призначати нове, унікальне значення для поля id статті при створенні запису статті;
- PublicationDate: Поле publicationDate має тип даних date, що дозволяє зберігати значення дати. Це поле зберігає дату публікації кожної статті;
- CategoryId: Поле CategoryId має тип даних smallint unsigned, що дозволяє зберігати цілі числа від 0 до 65 535. Це поле використовується для пов'язування кожної статті з відповідною категорією. Значення в цьому полі відповідає полю id в таблиці categories. Це реалізує відношення "багато-до-одного" між таблицею articles і categories, оскільки кожна стаття може належати лише одній категорії, але одна категорія може містити багато статей. Атрибут NOT NULL забезпечує, що кожна стаття обов'язково повинна мати присвоєну категорію;
- Title: Поле title має тип даних varchar(255), що дозволяє зберігати рядки довжиною до 255 символів. Це поле зберігає заголовок кожної статті;
- Summary: Поле summary має тип даних text, що дозволяє зберігати рядки довжиною до 65 535 символів. Це поле зберігає короткий зміст кожної статті;
- Content: Поле content має тип даних mediumtext, що дозволяє зберігати рядки довжиною до 16 777 215 символів. Це поле зберігає вміст кожної статті у форматі HTML.

Таблиця categories має такі поля:

- Id: Поле id має тип даних smallint unsigned, що дозволяє зберігати цілі числа від 0 до 65 535. Така місткість дозволяє CMS системі мати до 65 535 різних категорій. Вказуємо атрибут NOT NULL, що забезпечує обов'язкову наявність значення в цьому полі, воно не може бути порожнім (null). Атрибут auto_increment гарантує, що MySQL автоматично призначить нове, унікальне значення для поля id категорії при створенні нового запису;
- Name: Поле name має тип даних varchar(255), що дозволяє зберігати рядки довжиною до 255 символів. Це поле зберігає назву кожної категорії. Оскільки вказуємо атрибут NOT NULL, кожна категорія обов'язково повинна мати назву;
- Description: Поле description має тип даних text, що дозволяє зберігати рядки довжиною до 65 535 символів. Це поле зберігає опис кожної категорії. Як і в полі name, вказуємо атрибут NOT NULL, тому кожна категорія повинна мати опис.

Після створення таблиць articles та categories, можна продовжити розробку серверної частини CMS системи, а саме створенням конфігураційного файлу, а також класів. Проте, спочатку потрібно узагальнити та зобразити структуру бази даних за допомогою ER-діаграми.

### 3.3 ER-діаграма бази даних

ER-діаграма, або діаграма відношень сутностей, є ключовим інструментом, що відіграє невід'ємну роль при проєктуванні та візуалізації баз даних. Вона дозволяє зручно представити структуру бази даних, відобразивши сутності (таблиці), їх атрибути (поля) та взаємозв'язки між ними. У нашому випадку є дві сутності: articles та categories. Сутність "articles", або таблиця статей, включає в себе наступні атрибути:

- id – унікальний ідентифікатор статті;
- publicationDate – дата публікації статті;
- categoryId – ідентифікатор категорії, до якої належить стаття;

- title – заголовок статті;
- summary – короткий опис статті;
- content – повний текст статті;

Таблиця categories містить наступні атрибути:

- id – унікальний ідентифікатор категорії;
- name – назва категорії;
- description – опис категорії.

Між цими двома таблицями існує відношення "багато до одного", яке вказує, що одна стаття може належати лише одній категорії, але категорія може включати багато статей. На ER-діаграмі це зображується у вигляді лінії, яка з'єднує сутності "articles" та "categories", із вказівкою на тип відношення між ними.

ER-діаграми важливі, оскільки вони допомагають розуміти структуру бази даних, без необхідності детального вивчення коду SQL, та спрощують процес розробки та оптимізації баз даних.

ER-діаграми практично завжди використовуються при проєктуванні баз даних. Вони допомагають розробникам побачити всю картину відношень між сутностями, що в свою чергу сприяє більш ефективному процесу розробки і, в подальшому, оптимізації бази даних. На рисунку 3.2 наведено ER-діаграму бази даних для CMS системи:




Рисунок 3.2 – ER-діаграма бази даних

### 3.4 Конфігураційний файл CMS

Потрібно створити конфігураційний файл для зберігання різних налаштувань для CMS. Цей файл буде використовуватися всіма скриптами CMS системи. В папці проєкту створено файл config.php з наступним кодом:

```
<?php
ini_set("display_errors", true);
date_default_timezone_set("Europe/Kiev");
define("DB_DSN", "mysql:host=localhost;dbname=cms");
define("DB_USERNAME", "username");
define("DB_PASSWORD", "password");
define("CLASS_PATH", "classes");
define("TEMPLATE_PATH", "templates");
define("HOMEPAGE_NUM_ARTICLES", 5);
define("ADMIN_USERNAME", "admin");
define("ADMIN_PASSWORD", "mypass");
require(CLASS_PATH . "/Article.php");
function handleException($exception) {
echo "Sorry, a problem occurred. Please try later.";
error_log($exception->getMessage());}
set_exception_handler('handleException');?>
```

Цей файл містить різні налаштування, такі як:

- Відображення помилок у браузері
- Встановлення часового поясу сервера
- Встановлення деталей доступу до бази даних (DSN, ім'я користувача, пароль)
- Встановлення шляхів до класів і шаблонів
- Встановлення кількості статей, які будуть відображатися на головній сторінці
- Встановлення імені користувача та паролю адміністратора
- Підключення класу Article
- Створення функції обробки винятків handleException

Отже, було створено конфігураційний файл для CMS системи. Цей файл буде використовуватися всіма скриптами CMS системи для доступу до різних корисних налаштувань та зберігання глобальних налаштувань. Тепер можна продовжити розробку серверної частини CMS системи, створивши класи, шаблони та інші компоненти, які будуть використовуватись у проєкті.

### 3.5 Створення класу Article

У папці cms створено папку classes, всередині цієї папки створено новий файл під назвою Article.php та додано код, який наведено у додатку А.

Клас Article надає можливість створювати, оновлювати та видаляти статті. Він відповідає за зберігання статей у базі даних та їх отримання з неї. Після створення цього класу, робота зі статтями в інших скриптах CMS буде значно простішою.

Властивості класу включають $id, $publicationDate та інші. Кожен об'єкт Article зберігає дані статті в цих властивостях. Властивості класу відображають назви полів у таблиці бази даних статей.

Клас Article наслідує шаблон проєктування Active Record, який передбачає відображення властивостей класу на відповідні поля бази даних та методи для зберігання та отримання записів з бази даних.

Конструктор класу construct() – це спеціальний метод, який автоматично викликається двигуном PHP при створенні нового об'єкта Article. Конструктор приймає необов'язковий масив $data, що містить дані для заповнення властивостей нового об'єкта. В тілі конструктора відбувається заповнення властивостей.

Конструктор фільтрує дані перед зберіганням у властивостях. Властивості id та publicationDate перетворюються на цілі числа за допомогою (int). Назва та короткий опис фільтруються за допомогою регулярного виразу.

Метод storeFormValues() схожий на конструктор, оскільки зберігає дані, надані у масиві, властивостях об'єкта. Основна відмінність полягає в тому, що

storeFormValues() може обробляти дані у форматі, що надсилається через форми "Нова стаття" та "Редагувати статтю". Зокрема, він може обробляти дати публікації у форматі YYYY-MM-DD, перетворюючи дату в формат UNIX-таймстемпі, пригідній для зберігання в об'єкті.

UNIX-таймстемпи – це цілі числа, які представляють кількість секунд між опівночі 1 січня 1970 року та датою чи часом у питанні. У PHP часто використовуються UNIX-таймстемпи для роботи з датами та часом, оскільки їх легко зберігати та обробляти.

Мета цього методу полягає в тому, щоб спростити зберігання даних, що надсилаються формами, для наших адміністративних сценаріїв. Все, що їм потрібно зробити, це викликати storeFormValues(), передаючи масив даних форми.

Метод визначається з ключовим словом static, що дозволяє викликати метод без створення об'єкта:

```
public static function getById( $id ) {
```

Сам метод використовує PDO для з'єднання з базою даних, отримання запису статті за допомогою SQL-виразу SELECT та зберігання даних статті в новому об'єкті Article, який повертається коду що викликається.

PDO (PHP Data Objects) – це об'єктно-орієнтована бібліотека, вбудована в PHP, яка полегшує роботу PHP-скриптів з базами даних.

Ще одним методом для роботи з базою даних є getList(). Цей метод дозволяє отримувати список статей з бази даних з можливістю встановлення обмеження кількості записів та сортування. Метод getList() визначається наступним чином:

```
public static function getList( $numRows = 1000000, $order = "publicationDate DESC" ) {
```

Цей метод приймає два параметри: $numRows, який встановлює максимальну кількість статей для повернення, та $order, який визначає порядок сортування статей. За замовчуванням встановлюється велика кількість записів та сортування за датою публікації.

В середині методу getList() використовується PDO для виконання SQL-запиту, який отримує список статей згідно з вказаними параметрами. Результати запиту обробляються і повертаються у вигляді масиву об'єктів класу Article:

```
$articles = array();
while ( $row = $st->fetch() ) {
$article = new Article( $row );
$articles[] = $article;}
return $articles;
```

Метод insert() дозволяє створювати нові записи статей в базі даних. Цей метод визначається наступним чином:

```
public function insert() {
```

Цей метод використовує об'єкт PDO для виконання SQL-запиту INSERT, який створює новий запис статті в таблиці бази даних, використовуючи значення властивостей об'єкта Article.

Після успішного виконання запиту INSERT, метод insert() встановлює ID новоствореного запису для об'єкта Article:

```
$this->id = (int)$con->lastInsertId();
```

Метод update() дозволяє оновлювати наявні записи статей в базі даних. Цей метод визначається наступним чином:

```
public function update() {
```

Цей метод використовує об'єкт PDO для виконання SQL-запиту UPDATE, який оновлює запис статті в таблиці бази даних з використанням значень властивостей об'єкта Article. Запит UPDATE використовує ідентифікатор статті ($this->id) для знаходження відповідного запису в таблиці.

Метод delete() дозволяє видаляти записи статей з бази даних. Цей метод визначається наступним чином:

```
public function delete() {
```

Цей метод використовує об'єкт PDO для виконання SQL-запиту DELETE, який видаляє запис статті з таблиці бази даних. Запит DELETE використовує ідентифікатор статті ($this->id) для знаходження відповідного запису в таблиці. Після видалення запису, властивість id об'єкта Article встановлюється в null.

Тепер, коли було створено клас Article, який включає методи для роботи з базою даних, можна використовувати його для створення, оновлення, видалення та отримання записів статей у CMS системі.

Використання класу Article дозволить легко маніпулювати даними статей в різних частинах системи, таких як адміністративна панель або на головній сторінці сайту.

### 3.6 Створення адміністративної панелі admin.php

У цьому розділі розглянуто створення адміністративної панелі CMS за допомогою файлу admin.php. Файл admin.php відповідає за всі адміністративні функції CMS, такі як авторизація, створення, редагування та видалення статей.

У папці cms потрібно створити новий файл під назвою admin.php та додати код, який наведено у додатку Б. Якщо описувати функції коду детально: на початку файлу admin.php визначаються дії, які можуть виконуватися в адміністративній панелі, такі як авторизація, створення нової статті, редагування статті, видалення статті та відображення списку статей. Ці дії передаються через параметри URL, тому використовується змінна $action для того, щоб визначити поточну дію. За замовчуванням використовується дія "listArticles" для відображення списку статей.

Перевірка авторизації користувача відбувається на початку файлу admin.php. Перевіряємо, чи користувач вже авторизований, використовуючи змінну сесії $username. Якщо користувач не авторизований та спробує отримати доступ до адміністративної панелі, система перенаправляє його на сторінку авторизації.

Після перевірки авторизації використовуємо оператор switch для виконання відповідної дії в адміністративній панелі. Кожна дія відповідає відповідній функції, яка обробляє дію та відображає потрібний шаблон. Наприклад, функція newArticle() обробляє створення нової статті, а функція editArticle() дозволяє редагувати наявну статтю.

У файлі admin.php активно використовується клас Article для роботи зі статтями. Коли користувач обирає створення нової статті або редагування наявної статті, використовується шаблон editArticle.php для відображення відповідної форми. Цей шаблон перевикористовується для обох випадків, і він адаптується залежно від того, чи створюється нова стаття, чи редагується наявна. Форма містить поля для введення заголовка, короткого опису, дати публікації та повного тексту статті.

Після заповнення форми створення або редагування статті, користувач може зберегти зміни. У випадку створення нової статті, стаття буде додана до бази даних, а у випадку редагування, зміни будуть застосовані до відповідної статті в базі даних. Обидва процеси здійснюються за допомогою методів класу Article, таких як insert() та update().

Коли користувач обирає видалити статтю, спочатку перевіряється, чи існує стаття в базі даних. Якщо стаття знайдена, вона видаляється за допомогою методу delete() класу Article. Після успішного видалення статті користувача перенаправляють на сторінку зі списком статей та відображається повідомлення про успішне видалення.

Остання функція admin.php, listArticles(), відображає список всіх статей в CMS для редагування. Функція використовує метод getList() класу Article для отримання списку статей. Зібрані дані передаються у шаблон listArticles.php для відображення списку статей.

У файлі admin.php передбачена обробка помилок та відображення статусних повідомлень. Наприклад, якщо користувач спробує відредагувати або видалити статтю що не існує, він отримає відповідне повідомлення про помилку. Також, після успішного збереження змін у статті або видалення статті, користувачу відображається статусне повідомлення про це.

Адміністративна панель використовує шаблони для відображення відповідного контенту та елементів керування. Усі шаблони зберігаються в окремій папці, з назвою "templates". За допомогою шаблонізації можна легко змінювати вигляд адміністративної панелі, змінюючи лише відповідні шаблони.

### 3.7 Структура файлів та папок CMS

Для забезпечення легкої розробки та підтримки CMS, важливо утримувати чітку структуру файлів та папок. У CMS системі використовується така структура:

1. config.php: Файл конфігурації, який містить налаштування для підключення до бази даних, ім'я користувача та пароль для доступу до адміністративної панелі.
2. index.php: Головний файл, який відображає статті на сайті.
3. admin.php: Файл адміністративної панелі, який містить функції для керування статтями.
4. classes/: Папка з класами, такими як Article, які використовуються для роботи зі статтями та іншими сутностями.

- Article.php: Файл класу Article, який містить методи для роботи зі статтями.

5. templates/: Папка з шаблонами для відображення вмісту на сайті.

- archive.php: Шаблон для відображення архіву статей на сайті.
- homepage.php: Шаблон для головної сторінки сайту зі списком статей.
- viewArticle.php: Шаблон для відображення повної статті.

6. templates/admin: Папка з шаблонами для адміністративної панелі.

- listArticles.php: Шаблон для відображення списку статей в адміністративній панелі.
- editArticle.php: Шаблон для відображення форми створення або редагування статті.
- loginForm.php: Шаблон для відображення форми входу в адміністративну панель сайту.

7. templates/include: Ця папка містить загальні шаблони, які використовуються на всьому сайті.

- footer.php: Файл шаблону нижньої частини сторінки (футеру), який містить HTML-код для відображення на всіх сторінках сайту

- header.php: Файл шаблону заголовка, який містить HTML-код для відображення заголовка на всіх сторінках сайту.

8. tables.sql: Файл SQL, який містить запити для створення таблиці articles в базі даних.

9. style.css: Файл каскадних таблиць стилів (CSS), який визначає візуальний вигляд та форматування елементів на сайті.

Ця структура дозволяє легко знайти та змінити будь-який файл або шаблон, необхідний для роботи CMS. За необхідності в майбутньому можна додавати додаткові файли, папки та шаблони для реалізації нових функцій CMS системи.

### 3.8 Створення back-end сторінок

Першим шаблоном, який буде створено у папці admin, є loginForm.php, код якого наведено у додатку В. Ця сторінка містить форму входу для адміністратора, яка відправляє дані на admin.php?action=login. У формі є приховане поле "login", яке використовує функція login() з попередніх кроків для перевірки, чи була відправлена форма.

Форма також містить область для відображення помилок (наприклад, неправильне ім'я користувача або пароль), поля для введення імені користувача та пароля, а також кнопку "Увійти".

У файлі loginForm.php, використовується підключення до загальних шаблонів "header.php" та "footer.php", які містять елементи верхньої та нижньої частин сторінки. Це полегшує управління виглядом сайту, оскільки зміни в шапці та футері(підвалі) можуть бути виконані в одному місці, і вони автоматично застосовуються до всіх сторінок, які використовують ці шаблони.

У формі входу додається атрибут "required" до полів імені користувача та пароля, які забезпечують, що користувач не зможе надіслати форму без заповнення цих полів. Це забезпечує певний рівень клієнтської перевірки, який допомагає запобігти некоректному введенню даних.

Також, використовується атрибут "placeholder" для полів імені користувача та пароля, які відображають підказки в полі введення, коли користувач ще не вводив свої дані. Ці підказки забезпечують краще розуміння того, що користувач повинен ввести у кожному полі.

Крім того, атрибут "autofocus" дозволяє автоматично встановити фокус введення на поле імені користувача, коли сторінка завантажується. Це полегшує процес входу, оскільки користувачам не потрібно клацати на полі введення перед введенням своїх даних.

Шаблон loginForm.php є важливою частиною адміністративної панелі, оскільки він надає можливість адміністраторам безпечно входити у систему та отримувати доступ до функцій керування контентом.

Загалом, файл loginForm.php є ключовим компонентом адміністративної частини CMS, який допомагає забезпечити безпечний вхід для адміністраторів та ефективне керування контентом на вебсайті.

Другий шаблон адміністративної панелі у папці admin з назвою listArticles.php, код якого наведено у додатку Г, відображає список статей для редагування адміністратором. Після відображення повідомлень про помилки або статус, він проходить через масив об'єктів Article, які зберігаються у $results['articles'], відображаючи дату публікації та назву кожної статті в рядку таблиці. Цей шаблон також додає подію JavaScript onclick до кожного рядка таблиці статей, щоб адміністратор міг клацнути на статті для її редагування.

У файлі listArticles.php відображається список статей, які можуть бути відредаговані адміністратором. Він також показує повідомлення про помилки або статуси, використовуючи PHP для обробки даних від сервера, включаючи виведення дати публікації статті у форматі 'j M Y' та забезпечення безпеки даних за допомогою функції htmlspecialchars() для відображення імені користувача в блоку "adminHeader". Шаблон також включає загальну кількість статей, а також посилання для додавання нової статті адміністратором.

listArticles.php є важливим елементом адміністративної панелі CMS, оскільки він надає адміністраторам зручний та ефективний інтерфейс для

керування статтями на вебсайті. Цей файл відрізняється від loginForm.php, оскільки спрямований на роботу з вже наявними статтями та їх редагування, тоді як loginForm.php стосується авторизації адміністратора для доступу до адміністративної панелі.

Створення сторінки editArticle.php полягає в розробці форми для створення нових статей та редагування наявних статей. Цей шаблон розміщується у папці admin і містить код, наведений у додатку Д.

Форма editArticle.php має наступні основні елементи:

- Заголовок адміністративної панелі з інформацією про поточного користувача та посиланням для виходу.
- Назву сторінки, яка відображається на підставі значення змінної $results['pageTitle'].
- Ряд поля введення для назви статті, резюме, вмісту та дати публікації.
- Кнопки для збереження змін, скасування та видалення статті (якщо вона вже існує).

Форма editArticle.php спільно використовується для створення нових статей та редагування наявних статей. Відповідно до значення, переданого у змінній $results['formAction'], форма відправляє дані на admin.php?action=newArticle або admin.php?action=editArticle.

Важливою властивістю цієї форми є використання функції htmlspecialchars() для передачі всіх даних перед їх виведенням у розмітці. Це забезпечує належну безпеку та гарантує, що значення полів форми будуть належним чином екрановані.

Також варто звернути увагу на використання атрибута formnovalidate HTML5 на кнопці "Скасувати". Цей атрибут вказує браузеру, що форму не потрібно перевіряти, якщо користувач натисне кнопку "Скасувати".

Цей шаблон забезпечує інтуїтивно зрозумілий інтерфейс для адміністратора, що дозволяє легко створювати та редагувати статті на вебсайті.

Для управління категоріями статей у CMS системі, створено back-end сторінку listCategories.php, код якої наведено у додатку Е. Цей шаблон відображає список категорій статей, доступних для редагування адміністратором.

Спочатку включаємо необхідні шаблони для заголовка сторінки та адміністративної панелі. Згодом, якщо є повідомлення про помилки чи статуси, вони відповідно відображаються. Далі шаблон проходить крізь масив об'єктів категорій, які зберігаються у $results['categories'], відображаючи назву кожної категорії в рядку таблиці. Також додається подія JavaScript onclick до кожного рядка таблиці категорій, щоб адміністратор міг клацнути на категорію для її редагування. У кінці таблиці відображається загальна кількість категорій, а також посилання для додавання нової категорії адміністратором. Цей шаблон listCategories.php забезпечує адміністраторам легкий та ефективний спосіб перегляду та управління категоріями статей на вебсайті, що є важливою складовою CMS системи.

Також, Шаблон editCategory.php, код якого наведено у додатку Ж, дозволяє адміністратору створювати нові категорії або редагувати наявні категорії статей. За допомогою цього шаблону, адміністратор може вводити назву категорії та короткий опис для кожної категорії. Форма відправляє дані на admin.php, який виконує відповідну функцію (створення або редагування категорії) на основі значення $results['formAction'].

Враховуючи розроблені шаблони адміністративної панелі та їх технічні аспекти, можна зробити наступний висновок:

Ці шаблони адміністративної панелі сприяють створенню потужної та гнучкої CMS, яка забезпечує адміністраторам інтуїтивно зрозумілий та ефективний інтерфейс для управління контентом вебсайту. Завдяки ретельно продуманому дизайну та реалізації шаблонів, адміністратори зможуть легко створювати, редагувати та видаляти статті.

Крім того, використання сучасних технологій, таких як HTML, PHP, забезпечує гарантію сумісності з більшістю сучасних браузерів та платформ, а

також підтримку різних типів контенту, що забезпечує зручність та простоту використання для адміністраторів.

У цілому, розроблені шаблони адміністративної панелі сприяють успішній реалізації серверної частини CMS системи, що дозволяє створювати вебсайти з високим ступенем налаштованості та масштабованості, відповідаючи сучасним вимогам та стандартам розробки.

### 3.9 Встановлення на хостинг

У цьому розділі розглянуто процес встановлення розробленої CMS-системи на хостинг. Встановлення CMS на хостинг вимагає дотримання певних кроків, які описані нижче:

1. Вибір хостингу: Потрібно обрати надійного хостинг-провайдера, який відповідає вимогам щодо пропускної здатності, простору диска, підтримки PHP та MySQL, а також можливостей резервного копіювання та безпеки.
2. Реєстрація домену: Зареєструйте доменне ім'я, яке буде асоціюватися з вашим вебсайтом і CMS-системою. Краще використовувати стандартні домени верхнього рівня такі як .com, .org чи .net.
3. Налаштування DNS: Налаштуйте DNS-записи для вашого домену, під’єднавши його до сервера, на якому розташований ваш хостинг.
4. Завантаження файлів CMS: Завантажте файли розробленої CMS-системи на хостинг через FTP-клієнт або файловий менеджер, який надає хостинг-провайдер. Розмістіть файли у відповідних папках на сервері (зазвичай це папка public_html.
5. Створення бази даних: Створіть базу даних на сервері за допомогою панелі управління хостингом або іншого інструменту, який надається хостинг-провайдером. Запам'ятайте ім'я бази даних, користувача та пароль, оскільки вони знадобляться для налаштування конфігураційного файлу CMS.

6. Налаштування конфігураційного файлу: Відредагуйте конфігураційний файл CMS, вказавши правильні параметри підключення до бази даних, а також адресу сайту та шляхи до папок на сервері.

7. Імпорт структури бази даних: Імпортуйте структуру таблиць бази даних, яку було створено під час розробки CMS, в новостворену базу даних на хостингу.

8. Налаштування коректної часової зони: Перевірте та встановіть коректну часову зону на сервері хостингу, щоб уникнути проблем з відображенням дат та часу на сайті. Вам можуть знадобитися параметри PHP, такі як date_default_timezone_set, для встановлення правильної часової зони.

9. Тестування сайту: Після завершення усіх кроків перевірте, чи працює сайт належним чином. Тестуйте різні функції та можливості CMS, переконайтеся, що вони працюють коректно і без збоїв.

10. Резервне копіювання: Зробіть резервну копію всіх файлів та бази даних CMS-системи, щоб мати можливість відновити сайт у разі проблем або втрати даних.

Після вдалого завершення всіх вищеописаних кроків серверна частина CMS-системи буде належним чином встановлена на хостинг і готова до роботи. Завдяки різноманітним можливостям CMS-системи, ви зможете налаштувати її згідно з потребами та вимогами проєкту, а також забезпечити комфортну роботу з сайтом, а також змінювати, додавати та редагувати наявні статті.

Дану CMS систему було встановлено на домені та хостингу який було придбано у одного з найкращих доменних реєстраторів – компанії Namecheap, яка також надає безплатні домени для студентів університетів. У моєму випадку це був домен yarik-kryt.com а також Shared Stellar хостинг з вебпанеллю cPanel. На рисунку 3.3 зображено готовий сайт, розміщений у мережі Інтернет.

Рисунок 3.3 – CMS-система розміщена в мережі Інтернет

CMS-система дозволяє зайти в адміністративну панель, і звідти створювати, додавати статті, а також змінювати вже наявні. На рисунку 3.4 зображено вигляд входу в адміністративну панель:




Рисунок 3.4 – Вхід в адміністративну панель

На рисунку 3.5 зображено вигляд головного меню управління адміністративної панелі:

Рисунок 3.5 – Головне меню адміністративної панелі

### 3.10 Способи використання CMS системи

Системи управління контентом є невід'ємною частиною сучасного вебсередовища. Вони надають можливість управління вмістом вебсайтів, не вимагаючи від користувачів глибоких знань у програмуванні або веброзробці. Для розрізнення потенціальних сфер застосування CMS системи, можна виділити декілька ключових типів вебсайтів, для яких вони найбільш корисні:

- Освітній ресурс: Сайти для освіти та навчання часто використовують CMS для публікації навчальних матеріалів, домашніх завдань, а також розкладу занять для студентів. Вони дозволяють педагогам легко оновлювати вміст і надавати актуальну інформацію студентам в режимі реального часу.
- Блог: Блоги є одними з найпопулярніших типів сайтів, які використовують CMS. Блогери можуть публікувати свої думки та ідеї, створюючи

спільноту читачів. CMS також дозволяють публікацію гостьових постів, що збільшує обмін ідеями та думками між різними авторами.

- Сайт з новинами: Новинні вебсайти використовують CMS для публікації останніх новин та оновлень. CMS дозволяють створювати різні категорії для новин, що полегшує навігацію для читачів. Вони також можуть бути використані для публікації прес-релізів та інших важливих оголошень.
- Портфоліо: CMS є важливим інструментом для професіоналів, які хочуть демонструвати свої роботи онлайн. Вони можуть використовувати CMS для публікації своїх робіт, відгуків клієнтів, а також для розповіді про послуги, які вони надають. Портфоліо вебсайтів часто використовують CMS для виставки та продажу творчих робіт, таких як фотографії, малюнки, дизайн вебсайтів або будь-які інші професійні досягнення.

Всі ці сфери застосування підходять під CMS систему, що була створена у цьому проєкті. Функціонал систем управління контентом гнучкий і може бути налаштований відповідно до специфічних потреб будь-якого вебсайту. Основні переваги використання CMS включають простоту використання, гнучкість, масштабованість і здатність легко управляти великими обсягами вмісту. Для кращого розуміння, на рисунку 3.6 наведена ментальна карта, що ілюструє різні способи використання CMS системи.



Рисунок 3.6 – Ментальна карта способів використання CMS системи

# Схожість

Джерела з Бібліотеки 35

| | | | | |
|---|---|---|---|---|
| 1 | Студентська робота | ID файлу: 1004053580 | Навчальний заклад: National Aviation University | 1.16% |
| 2 | Студентська робота | ID файлу: 1009473014 | Навчальний заклад: Zaporizhzhya National University | 0.23% |
| 3 | Студентська робота | ID файлу: 106325 | Навчальний заклад: National University of Life and Environm 19 Джерело | 0.17% |
| 4 | Студентська робота | ID файлу: 1015069756 | Навчальний заклад: National Technical University of Ukraine “Kyi... | 0.12% |
| 5 | Студентська робота | ID файлу: 1015138923 | Навчальний заклад: National Technical University of Ukraine “Kyi... | 0.12% |
| 6 | Студентська робота | ID файлу: 1015139233 | Навчальний заклад: Yuriy Fedkovych Chernivtsi National University | 0.12% |
| 7 | Студентська робота | ID файлу: 1012385035 | Навчальний заклад: National Aviation University | 0.12% |
| 8 | Студентська робота | ID файлу: 1008333063 | Навчальний заклад: National Aviation University 2 Джерело | 0.11% |
| 9 | Студентська робота | ID файлу: 1009743473 | Навчальний заклад: National Technical University of Ukra 3 Джерело | 0.11% |
| 10 | Студентська робота | ID файлу: 1014748845 | Навчальний заклад: Taras Shevchenko National University of Kyiv | 0.11% |
| 11 | Студентська робота | ID файлу: 1014886988 | Навчальний заклад: State University Kyiv National Economic Univ... | 0.1% |
| 12 | Студентська робота | ID файлу: 1011443755 | Навчальний заклад: Lviv Polytechnic National University | 0.1% |
| 13 | Студентська робота | ID файлу: 1014727337 | Навчальний заклад: Lviv Polytechnic National University | 0.1% |
| 14 | Студентська робота | ID файлу: 1014813482 | Навчальний заклад: Lviv Polytechnic National University | 0.1% |