МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Розробка серверної частини Веб-порталу

Виконав студент 4 курсу групи К-41
Спеціальність 122 комп'ютерні науки
Ільічов Дмитро Ігорович

Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Консультант _____
_____

Рецензент к.т.н., доцент
Великодний Станіслав Сергійович

Одеса 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук, управління та адміністрування
Кафедра інформаційних технологій

Комплексна бакалаврська кваліфікаційна робота

на тему: «Розробка Веб-порталу факультету КНУА ОДЕКУ»

Склад:

Частина 1 «Розробка серверної частини Веб-порталу»

Виконавець: Ільічов Дмитро Ігорович
Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Частина 2 «Розробка клієнтської частини Веб-порталу»

Виконавець: Обуховський Ілля Юрійович
Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Староста роботи: Ільічов Дмитро Ігорович
Провідний керівник проекту: к.ф.-м.н., доцент Козловська Валентина Петрівна
Рецензент: к.т.н., доцент Великодний Станіслав Сергійович

ОДЕСА 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Комп'ютерних наук, управління та адміністрування
Кафедра Інформаційних технологій
Рівень вищої освіти бакалавр
Спеціальність 122 комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри_____
_____
" 13" квітня 2020 року_____

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ільчову Дмитру Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка Веб-порталу факультету КНУА ОДЕКУ
Ч.1 Розробка серверної частини Веб-порталу

керівник роботи Козловська Валентина Петрівна, к.ф.-м.н, доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від " 13 " квітня 2020 року № 37-С
2. Строк подання студентом роботи 30.05.2020
3. Вихідні дані до роботи Технічні вимоги до роботи Веб-порталу
Зовнішні представлення на ІС груп користувачів Веб-порталу

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) У вступі викладаються мета та задачі дипломної роботи
У перших розділах проводиться опис та аналіз предметної області, виконується її моделювання
Наводиться огляд та вибір існуючих засобів розробки програмного продукту
Описується розроблений програмний продукт та наводиться посібник користувача цього продукту
У висновках підводяться підсумки виконаної роботи

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

_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання_____13” квітня 2020 року_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Огляд та аналіз предметної області	13.04-		
		20.04		
2	Моделювання системи	21.04-		
		25.04		
3	Вибір засобів розробки Веб-порталу	26.04-		
		01.05		
4	Розробка бази даних	02.05-		
		10.05		
	Рубіжна атестація	11.05-16.05		
5	Розробка додатка адміністратора Веб-порталу	17.05-		
		21.05		
6	Оформлення пояснювальної записки	22.05-		
		29.06		
7	Перевірка дипломної роботи на оригінальність	30.05		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)			

Студент _____ __Ільічов Д.І._____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _Козловська В.П._____
(підпис) (прізвище та ініціали)

Передмова

Тема роботи: Розробка Веб-порталу факультету КНУА ОДЕКУ

Метою роботи є придбання теоретичних знань і практичних умінь з розробки сайтів із застосуванням засобів програмування PHP фрамеворк, HTML, CSS, JavaScript а так само зі створення баз даних в середовищі MySQL для створення порталу факультету КНУА ОДЕКУ з можливістю управління контентом порталу через адміністративну панель.

Задачею даної роботи є аналіз груп користувачів, проектування серверної та клієнтської частини веб-порталу, створення баз даних, вибір програмних засобів для його розробки та для верстки, опис роботи з цими програмними засобами, та створення готового до використання сайту факультету.

Частина 1: Розробка серверної частини Веб-порталу

Виконавець: Ільічов Дмитро Ігорович

Метою роботи є придбання теоретичних знань та практичних умінь з розробки серверної частини порталу із застосуванням мови програмування PHP зі зв’язкою баз даних MySQL.

Задачею даної роботи є аналіз груп користувачів адміністративної панелі, проектування серверної частини веб-порталу, створення баз даних для управління контентом порталу, вибір програмних засобів для розробки структури серверної частини, опис роботи з цими програмними засобами, та створення готового до використання сайту факультету.

Частина 2: Розробка клієнтської частини Веб-порталу

Виконавець: Обуховський Ілля Юрійович

Метою роботи є придбання теоретичних знань та практичних умінь з розробки клієнтської частини порталу яка виконується в браузері користувача із застосуванням мови програмування JavaScript, мови гіпертекстової розмітки HTML та каскадними таблицями стилів CSS.

Задачею даної роботи є аналіз цільової аудиторії порталу, проектування клієнтської частини веб-порталу, створення адаптивного дизайну, вибір програмних засобів для розробки структури клієнтської частини, опис роботи з цими програмними засобами, та створення готового до використання шаблону портала факультета.

Староста комплексної кваліфікаційної роботи _____ Ільічов Д.І. _____
(підпис) (прізвище та ініціали)

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

HTML – HyperText Markup Language – мова гіпертекстової розмітки

БД – база даних

PHP – Hypertext Preprocessor – Personal Home Page – скриптова мова

CMS – Content Management System – система управління контентом

MVC – Model Viev Controller – модель, уявлення, контролер

ООП – об’єктно-орієнтоване програмування

SQL – Structured query language – мова структурованих запитів

HTTP – HyperText Transfer Protocol – протокол передачі гіпертексту

IMAP – Internet Message Access Protocol – протокол прикладного рівня

FTP – File Transfer Protocol – протокол передачі файлів

POP – Post Office Protocol – протокол поштового відділення

SNMP – Simple Network Management Protocol – простий протокол мережевого управління

СКБД – система керування базами даних

CGI – Common Gateway Interface – загальний інтерфейс шлюзу

XHTML – Extensible HyperText Markup Language – розширювана мова гіпертекстової розмітки

AJAX – Asynchronous JavaScript And XML

CRUD – Create, Read, Update, Delete

Eloquent ORM – Система об’єкстно реляційного відображення

API – Application programming interface – програмний інтерфейс

JSON – JavaScript Object Notation – текстовий формат обміну даними

VCS – Version Control System – система керування версіями

IDE – Integrated Development Environment – інтегроване середовище розробки

ІСР – інтегроване середовище розробки

ВСТУП

Сьогодні кожна організація чи заклад має власний web-портал. Веб-портал – це місце, де можна опублікувати детальну інформацію про навчальний заклад, послуги, умови замовлення та виконання послуг, контакти і реквізити. Це дозволяє розширити поле діяльності і залучити тим самим увагу, для освітніх сайтів, у нашому випадку портал факультету КНУА ОДЕКУ, це висвітлення діяльності навчального закладу, надання повної та актуальної інформації студентам, співробітникам, абітурієнтам та їх батькам.

Мета роботи полягає в розробці серверної частини web-порталу факультету, який був би інформативним, зручним та гнучким в управлінні для адміністраторів сайту і задовольняє необхідним потребам факультету.

Задачею даної роботи є аналіз груп користувачів, проектування серверної частини веб-порталу, створення баз даних, вибір програмних засобів для його розробки, опис роботи з цими програмними засобами, та створення готового до використання сайту факультету.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Перш ніж приступити до реалізації поставленого завдання, потрібно проаналізувати предметну область даної роботи. Web-портал – це багатосторінковий, структурований web-сайт, розроблений за допомогою системи керування вмістом сайту, що є динамічним варіантом. Завданням такого сайту є надання можливості користувачеві ознайомитися з повною інформацією про факультет і її життєдіяльності. Для роботи порталу потрібно визначити групу користувачів для управління порталом. Розроблюваний web-сайт структурує обсяги інформації, пов'язані з факультетом, а саме: інформацію про новини, викладачів, контактні дані, будь яких змін в роботі факультету.

1.1 Користувачі інформаційної системи

Перш за все до числа користувачів інформаційних систем відносяться фахівці в предметній області системи, для задоволення інформаційних потреб яких система створюється. Користувачів цієї категорії називають кінцевими користувачами. Будемо вважати, що користувачами системи є не тільки кінцеві користувачі, а й програмні засоби додатків, які застосовують інформаційні ресурси даної інформаційної системи для вирішення власних завдань. У деяких інформаційних системах контингент користувачів не зафіксовано. Інформаційні ресурси таких систем вільно надаються будь-якому користувачеві. В інших системах для того, щоб стати користувачем, необхідно отримати від системного адміністратора необхідні повноваження доступу до системи, а іноді і до деяких її інформаційних ресурсів[1][1)].

Для функціонування веб порталу, потрібні такі користувачі, як адміністратор і модерація, вони повинні стежити за публікаціями, і додаванням контенту на портал керуючи даними в БД, та звичайні

[1)][1] Рад Б. Я. Архітектура інформаційних систем / Б. Я. Рад, А. І. Водяхо, В. А. Дубенецький, В. В. Цехановскій. - М.: Академія, 2012. 220с.

користувачі, цей контингент користувачів не фіксований, вони отримують інформацію перебуваючи на веб порталі.

До основних користувачів порталу кафедри чиї інформаційні запити портал повинен задовольняти відносяться:

- студенти яким читаються дисципліни співробітниками факультету;
- студенти напрямків і профілів, які виконують курсові та випускні клаліфікаціонние роботи на факультеті;
- аспіранти факультету;
- абітурієнти, які вибирають майбутню спеціальність;
- студенти молодших курсів, які вибирають подальшу спеціалізацію;
- батьки студентів, які навчаються на факультеті;
- викладачі факультету;
- керівництво і співробітники університету;
- потенційні партнери факультету;
- співробітники споріднених факультетів з інших вузів;
- випускники факультету;
- роботодавці студентів-випускників факультету;
- наукова і педагогічна громадськість.

Головна група користувачів як «адміністратор», повинна мати права, які дають повний контроль над порталом, призначення ролей іншим користувачам, створення нових баз даних, нових пунктів адміністративного меню і так далі. Група користувачів «модератор» повинна мати права на зміни, додавання, та видалення новинних постів, створення категорій та нових сторінок з контентом, щоб висвітлювати останні новини і зміни в житті кафедри. Щоб користувач отримав відповідні права, адімістратор порталу повинен внести його в дану категорію коричтувачів видав роль модератора.

1.2 Вимоги до інформаційної системи веб порталу факультету

Інформаційна система повинна:

- надавати можливість реєструватися;
- надавати можливість додавання нових таблиць в БД через зручний інтерфейс;
- надавати можливість створювати та встановлювати ролі користувачам;
- надавати зручний інтерфейс для внесення новин на портал;
- надавати можливість створення категорій;
- надавати можливість додавання контенту через зручний інтерфейс для створення нових сторінок;
- надавати можливість змінювати, переглядати і видаляти новини та нові сторінки.

1.3 Аналіз статичного та динамічного порталу.

Інформація, яка зберігається в глобальній мережі Інтернет, знаходиться на web-серверах, на яких, в свою чергу, встановлено спеціальне ПЗ, що дозволяє звичайним користувачам знаходити інформацію яка їм потрібна. Велика частина такої інформації впорядкована у вигляді web-сайта, кожен має свою адресу або ім'я в Інтернеті. Для того, щоб переглядати web-сайти, на комп'ютері користувача має бути встановлено спеціальне програмне забезпечення – браузер. Виходячи з того, яку адресу (ім'я) задається в пошуковому рядку "Адреса", браузер буде відображати відповідну інформацію у своєму робочому вікні. web-сайт складається з пов'язаних між собою web сторінок. Кожна сторінка містить у собі файл з розширенням .html, в якому міститься текстова інформація і спеціальні команди HTML-код, який, в свою чергу, визначає вид інформації для відображення в браузері. Для відображення інформації про певну наочну область

використовують бази даних. Всі web-сайти прийнято розділяти на дві групи: статичні і динамічні. Статичні сайти – це набір статичних HTML-сторінок, які пов'язані між собою посиланнями. Статичні HTML-сторінки створюються вручну, зберігається і завантажується на сервер, після чого при кожному зверненні до сайту представляються користувачеві в такому вигляді в якому були створені, тобто не зміна.

Рисунок 1.1 – Принцип роботи запиту статичної сторінки

Для того щоб внести зміни в інформації статичних сторінках, потрібно вручну редагувати програмний код сторінки. У статичних сайтів є як свої переваги, так і недоліки. Переваги статичних сайтів наступні:

– статичні сайти це мінімальне навантаження на сервер;
– дуже швидке завантаження і оновлення сторінок;
– розробка статичних сайтів обходиться набагато дешевше;
– перенесення на новий хостинг дуже легке.

Серед недоліків таких сайтів виділяється складність оновлення сайту, внесення будь-яких змін. Управління статичним сайтом неможливо без знань і умінь в області web-програмування – це тягне за собою додаткові витрати при необхідності щоб додати щось з нових матеріалів на web-сайт, нових розділів або категорій.

Динамічний сайт – це сайт, в якому є можливість редагувати вміст сторінок сайту, без звернення до web-програмування. Відображення сторінок

таких web-сайтів засноване на шаблонної структурі. У більшості випадків наповнення інформацією сторінок береться з БД. Під час запиту сторінки користувачем відбувається вибірка з бази даних, яка вставляється в шаблон, утворюючи при цьому нову web-сторінку, і пересилається web-сервером в призначений для користувача браузер, який і відображає її[2][1)].

Рисунок 1.2 – Прицнип роботи запиту динамічної сторінки з сервера

Можливість вносити зміни в усі сторінки може тільки певна група користувачів, такі як, адміністратори або привілейовані користувачі. В окремих випадках вносити зміни в контент web-сторінки допускаються і анонімні користувачі (наприклад, відправлення повідомлень на форум, або створення своєї теми). На відміну від статичних, динамічні сайти набагато більш гнучкі в управлінні.

Крім перерахованих переваг, динамічні сайти мають і ряд недоліків. У порівнянні зі статичними сайтами, динамічні важчі, дають велике навантаження на сервер – отже, вони більш вимогливі до хостингу, ресурсів сервера[3][2)].

[1)][2] Фрідман В. А., Александров А. В., Сергєєв Г. Г., Костін С. П. Будівництво Web-сайтів; Тріумф - Москва, 2011. - 288с.

[2)][3] Венедюхін Олександр, Воробйов Андрій Створення сайтів; Ексмо - Москва, 2011. - 528с.

1.4 Постановка задачі

Метою дипломної роботи є придбання теоретичних знань і практичних умінь з розробки серверної частини web-сайтів із застосуванням засобів PHP framework, а так само по створенню баз даних в середовищі MySQL. Об'єктом дослідження є структура факультету КНУА. Для реалізації мети дипломної роботи необхідно вирішити наступні завдання:

- вивчити теоретичні складові створення web-сайтів;
- провести моделювання інформаційної системи;
- виявлення груп користувачів;
- провести аналіз ПЗ для створення web-сайтів;
- розглянути мови програмування, що використовуються в web технологіях;
- провести аналіз засобів створення баз даних для web-сайтів;
- створити структуру серверної частини;
- створення web-сайту на основі системи керування вмістом, яка б дозволяла вносити в нього зміни не вдаючись до технічних фахівців;
- створити базу даних для сайту;
- провести аналіз працездатності створеного сайту.

Результатом виконання поставлених завдань повинен стати web-портал факультету КНУА ОДЕКУ.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Потрібно визначити групи користувачів, які будуть взаємодіяти з web порталом. Як різні групи використовують систему, та за якими сценаріями вони мають можливість з нею працювати.

2.1 Визначення груп користувачів

Основна група користувачів інформаційної системи факультету це адміністратори та модератори які вносять зміни в системі, керують порталом. Інші користувачі мають можливість отримувати цю інформацію як тільки вона з’явиться.

У групи користувачів «адміністратор» повинні бути такі варіанти використання системи як:

- за необхідністю додавати користувачів в систему;
- переглядати, редагувати і видаляти зі списку користувачів;
- створювати і видавати ролі користувачам системи;
- управляти адміністративною панеллю, додавати нові пункти меню за необхідністю;
- створення нових таблиць в базі даних, а так само їх перегляд редагування та видалення.

У групи користувачів «модератор» варіанти використання системи такі:

- вносити в базу даних новини (пости);
- переглядати, редагувати видаляти новини з баз даних;
- створювати категорії для поділу контенту;
- створювати нові окремі сторінки в БД;
- переглядати, редагувати, видаляти сторінки з баз даних.

Та група звичайних користувачів які прийшли на портал щоб отримати інформацію яка відображена на виході, вони не мають доступ до адміністративної панелі.

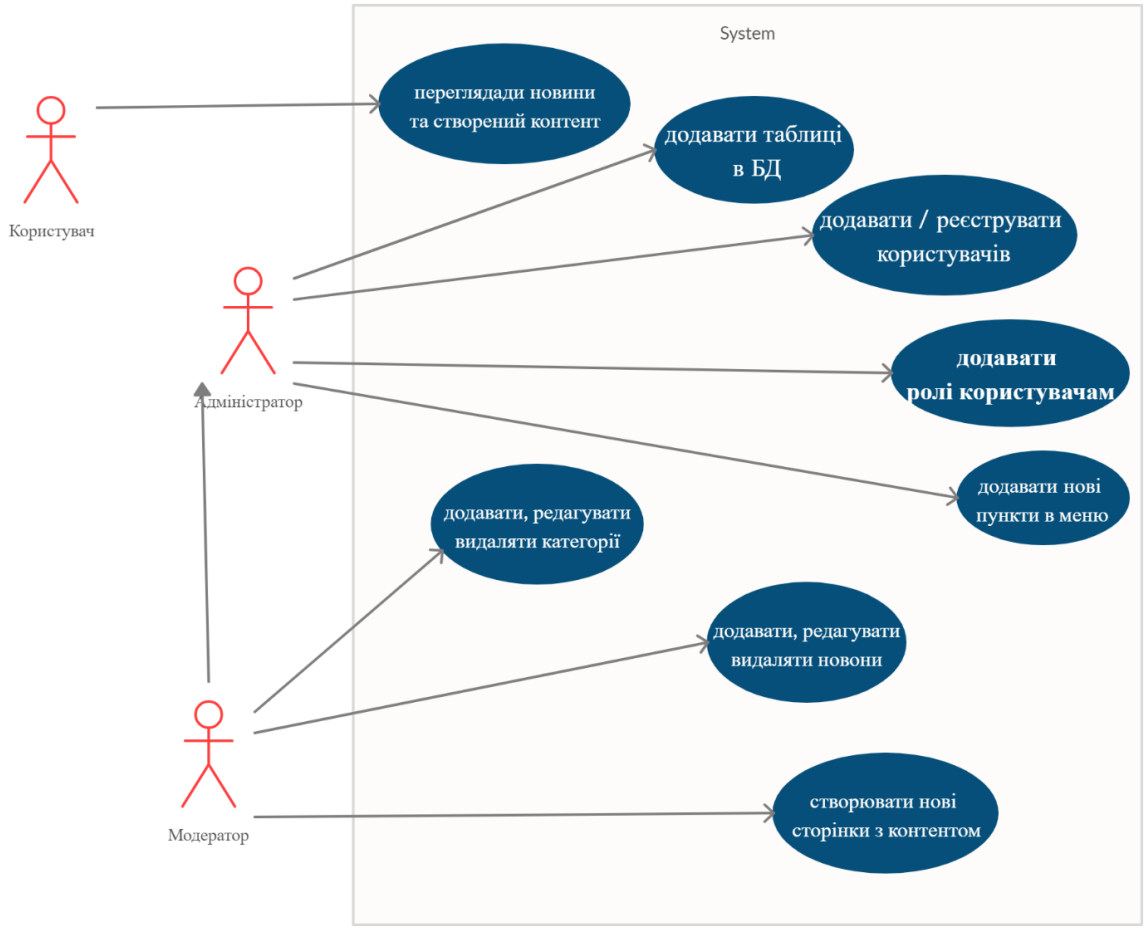

Рисунок 2.1 – Діаграма варіантів використання ІС

2.2 Логічна структура адмініністративної панелі

Логічна структура адміністративної панелі – це сукупність всіх сторінок для управління контентом, розташованих з урахуванням ієрархії. Тобто, взаємозв'язок сторінок, в яких простежується їх приналежність до розділів, категорій, підкатегорій і іншим типам сторінок.

Структуру треба побудувати таким чином, щоб спростити адміністраторам та модераторам перехід між різними пунктами меню для

керування контентом для змін контенту, за якою вони прийшли. Від цього залежить зручність користувача адміністративної панелі, час, який він проведе за роботою, цільові дії, які він зробить. Заплутана система навігації і надмірне структурування контену не підходить до роботи.

У користувача панелі повинна бути можливість легко потрапити на будь-яку сторінку. Для цього використовуйте рядки навігації, так звані «хлібні крихти». Вони допоможуть користувачеві знайти потрібний розділ і не заблукати. Виглядають вони як зображено на рисунку 2.2

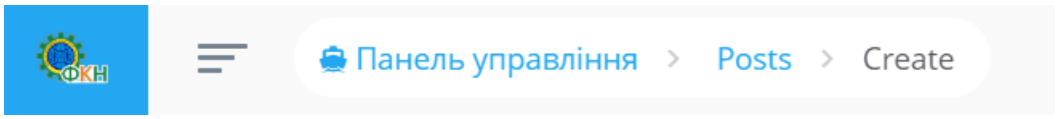

Рисунок 2.2 – Приклад «хлібні крихти»

Логічна структура адміністративної панелі зображена на рисунку 2.3 на якій є такі сторінки:

– користувачі, сторінка для управління користувачами;
– ролі сторінка для додавання, редагування та видалення ролей;
– медія, для управління медія файлами на сайті;
– пости, для додавання в базу постів, їх редагування і видалення з баз;
– категорії, управління категоріями;
– сторінки, для додавання в базу нових сторінок з контентом;
– налаштування, для зміни аватарки в меню і фону;
– інструменти;
– конструктор меню, для додавання пунктів меню;
– база Даних для створення нових таблиць в базі.

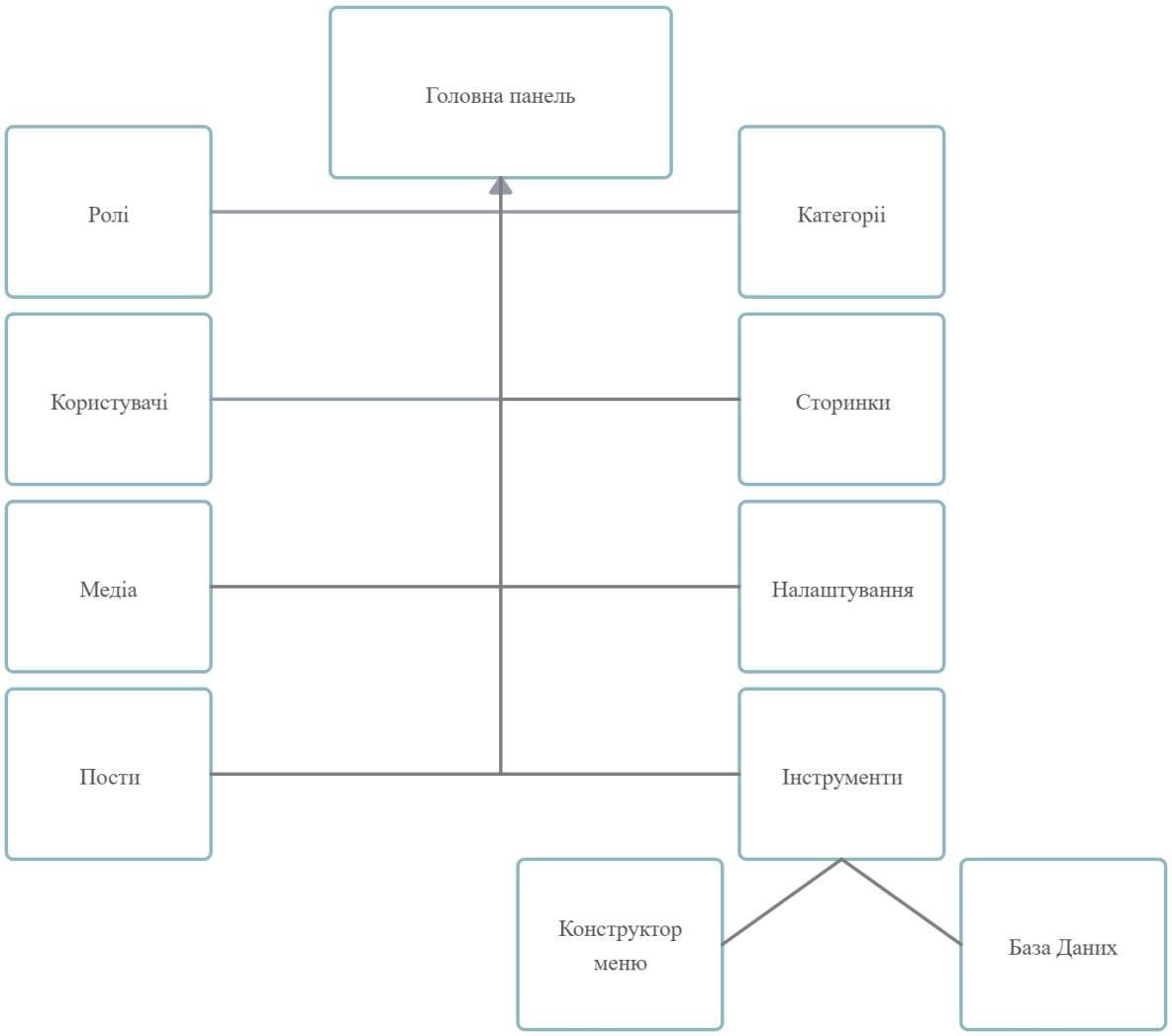

Рисунок 2.3 – Логічна структира адміністративної панелі

3 ВИБІР ПРОГРАМНИХ ЗАСОБІВ

3.1 Вибір застосування

Часи унікальних сайтів, написаних веб майстрами самостійно від початку і до кінця, відходять у минуле. Сучасні вимоги до сайтів такі, що код для необхідного функціоналу в багатьох випадках довелося б писати роками. На допомогу програмістам приходять такі ефективні інструменти: PHP фреймворки та CMS.

3.1.1 Опис CMS

Серед поширених CMS – WordPress, Drupal, 1С Бітрікс. Це по суті програмне забезпечення, яке являє собою конструктор з уже готових деталей. Ці деталі – різні модулі та плагіни в готовому вигляді. При розробці сайту на CMS завдання розробника полягає в установці цієї системи управління і в підключенні необхідних модулів. Якщо виникає необхідність у розробці якогось нестандартного або складного модуля, в більшості випадків реалізувати це буде складно, а іноді навіть неможливо.

Переваги CMS:

- просто і зручно використовувати;
- доступний широкий функціонал за рахунок доповнень, тем, розширень;
- сайт можна створити за короткий проміжок часу;
- наявність документації.

Недоліки:

- не підходять для нетипових завдань;
- популярні CMS уразливі;

– необхідно стежити за оновленням CMS і сумісністю версій доповнень; підвищене споживання ресурсів, особливо при використанні плагінів.

3.1.2 Опис PHP Framework

PHP – Фреймворк (framework) – готовий каркас з набором модулів, компонентів, розширення для швидкої, простої і якісної розробки програм. Робота з фреймворками вимагає певних знань. Розробляючи проект на фреймворку, робота ведеться практично з нуля. У кожному фреймворку є набір заготовлених модулів, але вони використовуються в якості прикладу. Програміст сам вибудовує логіку проекту, структуру бази даних. При цьому з допомогою фреймворка можна розробити функціонал будь-якого рівня складності, реалізувати будь-яку нестандартну задачу, і зробити проект унікальним.

Одним з головних переваг у використанні фреймворків є те, що фреймворк визначає уніфіковану структуру для побудованих на його базі додатків. Тому додатки на фреймворку значно простіше супроводжувати і допрацьовувати, так як стандартизована структура організації компонентів зрозуміла всім розробникам на цій платформі і не потрібно довго розбиратися в архітектурі, щоб зрозуміти принцип роботи програми або знайти місце реалізації того чи іншого функціоналу. Більшість фреймворків для розробки веб-додатків використовує парадигму MVC (модель-уявлення-контролер) – тобто дуже в багатьох фреймворках ідентичний підхід до організації компонентів програми та це ще більше спрощує розуміння архітектури додатку навіть незнайомому розробнику фреймворку. Проектування архітектури ПЗ при розробці на фреймворку теж дуже спрощується – в методології фреймворків звичайно закладені кращі практики програмної інженерії та просто дотримуючись цих правил можна уникнути багатьох проблем і помилок в проектуванні. По суті, фреймворк – це безліч

конкретних і абстрактних класів, пов'язаних між собою і впорядкованих згідно з методологією фреймворка. Конкретні класи зазвичай реалізують взаємні відносини між класами, а абстрактні класи являють собою точки розширення, в яких закладений у фреймворк функціонал може бути використаний «як є» або адаптований під завдання конкретного додатка. Для забезпечення розширення можливостей в більшості фреймворків використовуються техніки ООП: наприклад, частини програми можуть успадковуватися від базових класів фреймворка або окремі модулі можуть бути підключені як домішки. Екосистеми веб-фреймворків також багаті на готові реалізації багатьох функціональних можливостей. Розробникам при роботі над типовими завданнями не треба «винаходити велосипеди», так як вони можуть скористатися вже створеної спільнотою реалізацією. А це не тільки скорочує витрати часу і грошей, але і дозволяє домогтися більш високої стабільності рішення – компонент, який використовується і допрацьовується тисячами інших розробників зазвичай більш якісно реалізований і краще протестований на всіляких сценаріях, ніж рішення, яке може в адекватні терміни розробити один розробник або навіть невелика команда.

З огляду на поставлені вище завдання, найбільш оптимально використовувати PHP фреймворк для розробки веб порталу факультета.

3.2 Вибір мови програмування серверної частини

Для програмування була обрана мова PHP (англійською Hypertext Preprocessor – гіпертекстовий препроцесор) – мова програмування, створена для породження HTML-сторінок зі сторони веб-сервера.

Дана мова програмування створена саме для ведення Web-розробок і може впроваджуватися безпосередньо в програмний код Web-сторінки. Для впровадження коду PHP його відокремлюють спеціальними початковим і кінцевим тегами, за допомогою яких процесор PHP визначає початок і кінець

ділянки HTML-коду, який містить PHP-команди. Мова програмування PHP містить в собі велику кількість вбудованих функцій, таких як: функція роботи з файловою системою, функція роботи з протоколом HTTP, функція роботи з датою і часом, функція обробки рядків і масивів. Програмний продукт, написаний на мові PHP, складається з набору спеціальних конструкцій. В якості таких конструкцій можуть служити будь-які елементи, які використовуються в коді PHP, такі як цикли, оператори, функції[4][1)].

Головна особливість мови полягає в тому, що вона добре взаємодіє з усіма сучасними Web-технологіями і підтримує велику кількість сучасних Web-протоколів, а саме: протокол прикладного рівня для доступу до електронної пошти IMAP, протокол FTP, протокол POP, протокол SNMP. Також мова PHP відмінно підходить для роботи з базами даних. В більшості сучасних СКБД здійснена підтримка скриптового мови програмування PHP. Веб-сервер інтерпретує код PHP у код HTML, який передає клієнту. На відміну від мови JavaScript, користувач не бачить PHP-коду, бо браузер клієнта отримує готовий html-код.

Коли користувач звертається до сервера він пише в рядку адреси браузера адресу сторінки чи переходить за якимось посиланням, тобто створює запит.

У найпростішому випадку викликають HTML сторінку – текстовий файл. Браузер, посилає запит на сервер, отримує код сторінки, додатково завантажує необхідні файли, прописані в коді сторінки і необхідні для відображення її дизайну та її коректної роботи, та показує результат у своєму вікні. Такі сторінки мають статичний характер.

А коли викликають сторінку, написану мовою php у цьому випадку відбувається зовсім інше. При зверненні до сервера Web машина обробляє дані, отримані з файлу *.php, після чого сервер відправляє результат опрацювання цих даних у браузер користувача. Такі сторінки називають динамічними.

[1)][4] Офіційний сайт скриптової мови PHP. URL: http://php.net/

З допомогою мови PHP можна зробити будь-що, що роблять CGI програми (англійською Common Gateway Interface – загальний інтерфейс шлюзу – стандарт інтерфейсу, який використовують для організації взаємодії програми веб-сервера із зовнішньою програмою). Наприклад, опрацювання даних форм, породження вмісту динамічних сторінок, відправлення та отримання куків. Але PHP має значно ширші можливості.

Основні області застосування PHP коду це написання скриптів для виконання на стороні сервера. PHP традиційно і найширше використовують саме таким чином. Потрібно запустити веб-сервер з встановленою на ньому мовою PHP. Через веб-сервер можна пропускати сторінку з кодом PHP, та проглядати результат її роботи через веб-браузер. Все це можна робити на домашньому комп'ютері[5][1)].

Створення скриптів для виконання в командному рядку без будь-якого сервера чи браузера. Цей тип використання ідеальний для скриптів, що регулярно запускають через cron (на *nix чи Linux) чи його аналог Планувальник Задач (на Windows). Ці скрипти можна використовувати для простого опрацювання текстів.

Створення програм з графічним інтерфейсом. Хоча мова PHP не найкраща для такого використання.

PHP можна використовувати при більшості операційних систем: Linux, багато варіантів Unix (включаючи HP-UX, Solaris та OpenBSD), Microsoft Windows, Mac OS X, RISC OS та, можливо, інших.

PHP має також підтримку для більшості сучасних веб-серверів: Apache, IIS та багатьох інших. Окрім цього, підтримка є на будь-якому веб-сервері, що використовує бінарні коди FastCGI – клієнт-серверного протоколу взаємодії веб-сервера та програм.

PHP не обмежено створенням лише коду HTML: можливе подання зображень, PDF-файлів та навіть Flash роликів, що які породжують «на

[1)][5] Зандстра. PHP. Об'єкти, шаблони і методики програмування. / М. Зандстра- М: Вільямс, 2011. 560с.

льоту». Можна легко виводити будь-який текст XHTML та будь-який інший XML-файл. PHP може породжувати такі файли та зберігати їх у файловій системі, а не лише виводити текст, формувати кеш на стороні сервера для динамічного вмісту.

Одна з найсильніших та найістотніших особливостей PHP – це підтримка широкого кола баз даних: Adabas D, dBase, Direct MS-SQL, Empress, FilePro (read-only), FrontBase, Hyperwave, IBM DB2, Informix, Ingres, InterBase, mSQL, MySQL, ODBC, Oracle (OCI7 и OCI8), Ovrimos, PostgreSQL, Solid, Sybase, Unix dbm.

3.3 Вибір PHP фраемворку

Призначення цього проекту це основний фактор для прийняття рішення при виборі фреймворка. Потрібно створити веб-додаток, немає потреби використовувати фреймворк промислового рівня, що вимагає великої кількості ресурсів. На даний момент існує багато фреймворків, але найпопулярніші це Laravel, Symfony, Yii2.

3.3.1 Опис фраемворку Yii2

Yii2 відомий підвищеною безпекою і швидким завантаженням. Цей фреймворк застосовує принцип не-повторення коду. Використовуючи Ajax і JQuery, Yii2 спрощує розробку додатків з високою масштабованість[6][1)].

Підвищена безпека. Yii2 має численні показники безпеки, покликані захистити ваш сайт або веб-додаток від атак. Вам більше не доведеться хвилюватися про такі речі як підробка cookie, SQL injection, міжсайтова підробка запитів (CSRF), міжсайтовий скриптинг (CSS).

[1)][6] Офіційний сайт документації фраемвока YII2. URL: https://www.yiiframework.com/doc/guide/2.0/ru

Більш швидка розробка. Оскільки Yii генерує базові операції CRUD (створення, читання, оновлення та видалення), команда розробників може працювати швидше, а це знижує вартість розробки.

Легка установка. В ході процесу розробки важлива кожна хвилина. Yii економить час розробників, надаючи шаблон інсталяції. Завдяки цьому на установку і настройку фреймворка йде менше часу.

3.3.2 Опис фраемворку Symfony

З часу своєї появи в 2005 році фреймворк Symfony надає веб-розробникам вбудований функціонал для тестування, а також можливість працювати з компонентами і багаторазово використовувати код. Цей фреймворк володіє і іншими перевагами:

Швидка швидкість завантаження. Продуктивність додатка має велике значення. Вона багато в чому визначає, чи будуть споживачі користуватися цим додатком. Symfony 4.2 завантажує REST API за 2мс. Це робить Symfony найшвидшим PHP-фреймворком[7][1)].

Гнучкість. Symfony пристосовується до будь-яких вимог проекту. У цьому фреймворку є Event Dispatcher, завдяки якому Symfony є повністю конфігурованим. Крім того, всі компоненти Symfony управляються незалежно один від одного. Вичерпна документація. Цей фреймворк має зрозумілу і детальну документацію, яка буде корисна як початківцям, так і досвідченим розробникам. У цій документації добре пояснюються всі компоненти фреймворка, а також наводяться приклади.

Надійність. Компанія SensioLabs, що займається розробкою Symfony, підтримує цей фреймворк ось уже 13 років. При виникненні будь-яких проблем ви можете зв'язатися з розробниками.

[1)][7] Документація фраемворку Symfony. URL: https://symfony.com.ua/doc/current/index.html

3.3.3 Опис фраемворку Laravel

Laravel – це фреймворк для web-додатків з виразним і елегантним синтаксисом. Він дозволить спростити вирішення основних наболілих завдань, таких як аутентифікація, маршрутизація, сесії і кешування. Laravel – це спроба об'єднати все найкраще, що є в інших PHP фреймворків[8][1)].

Він був створений як альтернатива фреймворку Codeigniter, в якому було недостатньо корисних функцій для розробки веб-додатків. В якості основи Laravel виступають компоненти попереднього фреймворка - Symfony.

Фреймворк Laravel дуже популярний серед західних розробників веб-додатків. За допомогою менеджера пакетів Composer, фреймворк Laravel дозволяє легко встановлювати і підключати різні компоненти для використання в веб-додатку. Реалізація шаблону ActiveRecord – Eloquent ORM, дозволяє встановити відносини між об'єктами бази даних веб-додатків і вибудовувати зручні запити для маніпуляції даними. Механізм автозавантаження класів дозволяє не підключати вручну файли через include і запобігає завантаженню що не використовуються. Зручна система міграцій допомагає спростити розгортання і оновлення веб-додатки. У Laravel є вбудована підтримка движка шаблонів Blade, за допомогою якого можна робити прості уявлення веб-додатки використовуючи спеціальний синтаксис. При створенні програми можна використовувати Artisan – інтерфейс командного рядка для введення вбудованих команд, а також створення своїх власних. У Laravel є багато корисних функцій, що дозволяють зробити процес розробки веб-додатків швидким, простим і якісним. З подібних функцій можна відзначити dd () – зручний аналог стандартної функції PHP var_dump (). Функція виводить інформацію змінної в більш зрозумілій формі, розділяючи дані на дерево атрибутів і значень, в можливістю пошуку і переходу по ним.

[1)][8] Офіційний сайт документації фраемворка Laravel. URL: http://laravel.su/

Основні переваги Laravel:

- Досить і зрозуміла документація.
- Навколо фреймворка створена потужна екосистема. Різні курси, конференції, навчальні матеріали дозволяють зібрати навколо фреймворка велику кількість розробників і спонсорів, які зацікавлені в розвитку інструменту і приймають в цьому участь.
- Одним з найбільш очевидних плюсів Laravel, є гнучка система маршрутизації, що дозволяє скласти найрізноманітніші перевірки маршруту веб-додатка. Можливість виділити маршрути в спеціальні групи, використовувати простір імен, вказати параметри маршруту, використовувати регулярні вирази, налаштувати піддомени маршрутизації і багато іншого.
- У Laravel багато синтаксичного цукру. Синтаксис API фреймворка досить простий і зрозумілий. Тут немає довгих і складних конструкцій, а тільки короткі і продумані назви функцій.
- Laravel містить зручний механізм обробки помилок і виключень.
- Фреймворк включає в себе вбудовані механізми аутентифікації і авторизації користувачів, яку можна переналаштувати під свої потреби.
- Laravel надає з коробки механізми для кешування веб-додатки за допомогою Memcached і Redis. Крім цього є зручні функції для використання простого файлового кешування даних.
- Laravel надає чистий і простий API поверх популярної бібліотеки SwiftMailer з драйверами для SMTP, Mailgun, SparkPost, Amazon SES і sendmail, щоб зробити відправку пошти через локальну або хмарну службу за вибором. У тому числі є механізм для побудови черг відправки пошти.
- Laravel Cashier забезпечує виразний, вільний інтерфейс до сервісів білінгу по підписці Stripe і Braintree.

Провівши аналіз даних фреймворків вибір впав на Laravel. Переваги створення бізнес-додатків з використанням Laravel дійсно величезні. Нові функції вводяться з кожним оновленням. Framework допомагає створювати чудові додатки, використовуючи простий і ефективний синтаксис. Ось чому Laravel підходить для даного проекту.

3.4 Менеджер залежностей Composer

Composer – це пакетний менеджер залежностей, призначений для спрощення завантаження і установки сторонніх PHP бібліотек в проект. Наприклад, за допомогою нього можна дуже просто додати в проект пакети, а також розгорнути інші проекти, які поширюються разом з файлом «composer.json».

«Composer.json» – це текстовий файл, в якому в форматі JSON описані всі сторонні пакети від яких залежить даний проект.

Наприклад, для того щоб в певний проект, що розробляється додати сторонні бібліотеки, в ньому можна просто створити «composer.json» і описати в цьому файлі всі необхідні залежності. Після цього для установки всіх необхідних зовнішніх PHP пакетів в проект досить буде ввести в консолі всього одну команду (composer install).

Крім цього, Composer застосовується не тільки для установки PHP бібліотек. За допомогою Composer здійснюється також установка різних PHP фреймворків наприклад Laravel, для його установки є команда в композер 'laravel new імя_проекта'

Composer являє собою звичайний PHP скрипт, тобто програму, написану на мові PHP.

Основна мета цієї програми полягає в тому, щоб надати веб-розробнику зручний інструмент, за допомогою якого він зможе дуже просто завантажувати і встановлювати пакети в проект, їх оновлювати, а також при

необхідності здійснювати їх видалення. Всі ці дії Composer дозволяє виконати за допомогою введення всього однієї або декількох команд.

3.5 Система керування даними

Система керування базами даних – це сукупність мовних і програмних засобів, яка здійснює доступ до даних, дозволяє їх створювати, змінювати і видаляти, забезпечує безпеку даних і так далі. Загалом СКБД – це система, що дозволяє створювати бази даних і маніпулювати даними з них. А здійснює цей доступ до даних СКБД за допомогою спеціальної мови – SQL.

SQL – мова структурованих запитів, основним завданням якого є надання простого способу зчитування і запису інформації в базу даних. Отже, найпростіша схема роботи з базою даних зображена на (рисунку 3.1).

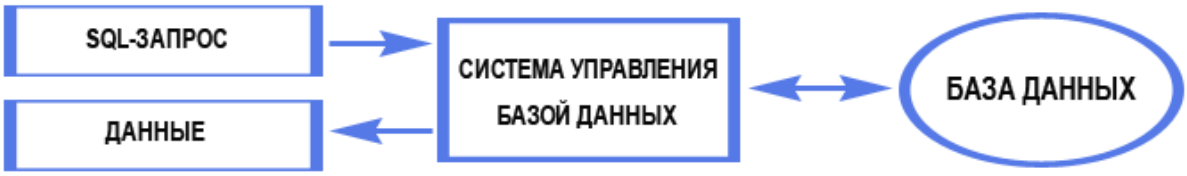

Рисунок 3.1 – Схема роботи з базою даних

За характером використання СКБД ділять на ті що розраховані на одного користувача (призначені для створення і використання БД на персональному комп'ютері) і розраховані на багато користувачів (призначені для роботи з єдиною БД декількох комп'ютерів, об'єднаних в локальні мережі). На сьогоднішній день число використовуваних СКБД обчислюється десятками. Найбільш відомі СКБД – Microsoft Visual FoxPro і Access, розраховані на багато користувачів - MS SQL Server, Oracle та MySQL.

3.5.1 MySQL

PHP дозволяє використовувати різні систему управління базами даних, але найбільш популярною на сьогоднішній день в зв'язці з PHP є MySQL. MySQL являє безкоштовне програмне забезпечення, що дозволяє взаємодіяти з базами даних за допомогою команд мови SQL. Це система управління реляційними базами даних. У реляційній базі даних дані зберігаються в окремих таблицях, завдяки чому досягається виграш у швидкості й гнучкості.

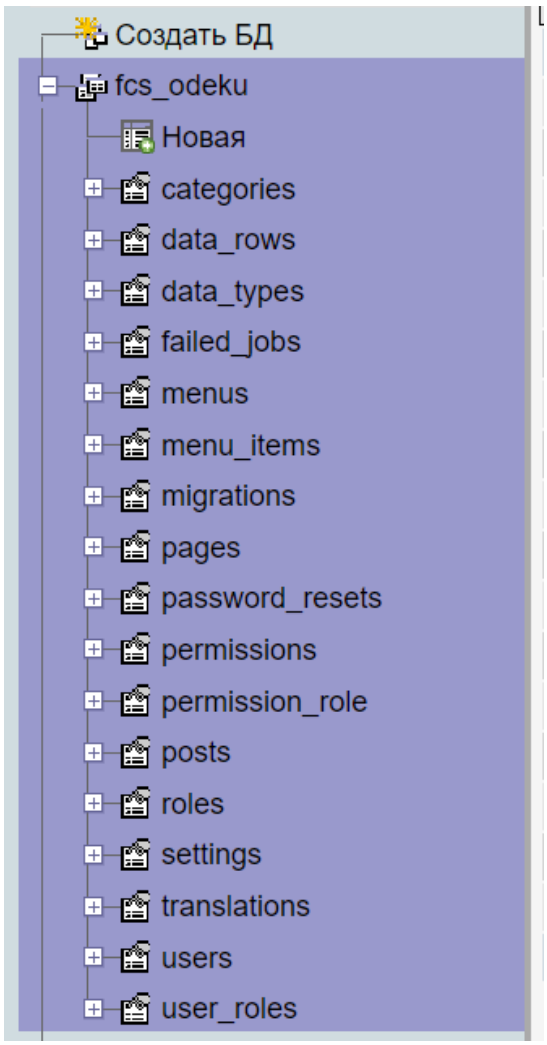

Рисунок 3.2 – Приклад бази даних з таблицями

Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість поєднувати при виконанні запиту дані з декількох

таблиць. SQL як частина системи MySQL можна охарактеризувати як мову структурованих запитів, що використовується для доступу до баз даних. MySQL складається з двох частин: серверної і клієнтської.

Сервер MySQL постійно працює на комп'ютері. Клієнтські програми (наприклад, скрипти PHP) посилають серверу MySQL SQL – запити через механізм сокетів (тобто за допомогою мережевих засобів), сервер їх обробляє і запам'ятовує результат. Тобто скрипт (клієнт) вказує, яку інформацію він хоче отримати від сервера баз даних. Потім сервер баз даних посилає відповідь (результат) клієнтові (скрипту)[9][1)].

Структура MySQL трирівнева: бази даних – таблиці – записи. Бази даних і таблиці MySQL фізично представляються файлами з розширеннями frm, MYD, MYI. Логічно таблиця являє собою сукупність записів[10][2)].

А записи – це сукупність полів різного типу. Ім'я бази даних MySQL унікально в межах системи, а таблиці – в межах бази даних, поля – в межах таблиці. Один сервер MySQL може підтримувати одразу декілька баз даних, доступ до яких може розмежовуватись логіном і паролем.

Структура таблицы | Связи

	#	Имя	Тип
☐	1	id	int(10)
☐	2	author_id	int(11)
☐	3	category_id	int(11)
☐	4	title	varchar(255)
☐	5	seo_title	varchar(255)
☐	6	excerpt	text
☐	7	body	text
☐	8	image	varchar(255)
☐	9	slug	varchar(255)
☐	10	meta_description	text
☐	11	meta_keywords	text
☐	12	status	enum('PUBLISHED', 'DRAFT', 'PENDING')
☐	13	featured	tinyint(1)
☐	14	created_at	timestamp
☐	15	updated_at	timestamp

Рисунок 3.3 – Приклад таблиці з полями

[1)][9] Мішель Е. Д. Вивчаємо PHP і MySQL / Е. Д. Мішель, А. Ф. Джон. - СПб.: Символ-Плюс, 2008. 442с.

[2)][10] MySQL керівництво адміністратора. - М.: Вильямс, 2005. - 621с.

База даних з точки зору MySQL – це звичайний каталог, що містить файли певного формату – таблиці. Таблиці складаються із записів, а записи, у свою чергу, складаються з полів. Поле має два атрибути - ім'я і тип.

3.5.2 phpMyAdmin

Є кілька способів, як можна отримати доступ і працювати з базою даних MySQL. Наприклад за допомогою командного рядка.

Цей спосіб вимагає від користувача певний рівень професіоналізму і знань. Потрібно знати відповідні команди і правила роботи з командним рядком.

За допомогою веб-додатків, які дозволяють працювати з базою даних через графічний інтерфейс.

Цей спосіб більше підходить для простих користувачів і має інтуїтивно зрозумілий інтерфейс. Навіть якщо людина ніколи не працював з базою даних, якісь основні операції він в будь-якому випадку зможе зробити.

Одним з таких веб-додатків для роботи з базою даних MySQL є phpMyAdmin.

phpMyAdmin – це просте, зручне і добре документоване рішення, яке переведено на безліч мов. Навіть якщо у користувача виникають якісь питання по роботі з цим додатком, в Інтернеті можна без зусиль знайти відповіді на більшість стандартних запитань. PhpMyAdmin написаний на мові PHP.

Що можна робити за допомогою phpMyAdmin:

– Створювати і коригувати бази даних, таблиці, записи;

– Створювати користувачів;

– Можливість виконувати SQL-команди;

– Система пошуку по базі даних;

Загалом, є всі необхідні кошти, які необхідні для роботи з базою даних.

Потрібно відзначити, що практично всі хостери встановлюють саме phpMyAdmin в свої панелі управління для роботи з базою даних. Ця програма є одним з найпопулярніших серед інших додатків в цьому класі.

3.6 Локальний сервер

Локальний сервер – це емулятор хостингу. Потрібен він для можливості створити сайт у себе на комп'ютері, і в подальшому перенести його на хостинг. Більшість же сайтів створюються на основі фреймворку або системи управління контентом (CMS). Для їх коректної роботи потрібно середовище, в якому працюють бази даних, серверні мови програмування.

Щоб не встановлювати ці компоненти окремо (наприклад, веб-сервер Apache, бази даних MySQL, мови програмування PHP, Perl) – зазвичай використовують готовий набір програм.

Для проекту було обрано локальний сервер Open Server. Він являє собою так званий WAMP комплекс, суть якого можна зрозуміти з розшифровки цієї абревіатури:

- Windows - операційна система, для роботи в якій призначений даний локальний сервер;
- Apache - web-сервер, який «піднімається» при запуску програми Open Server;
- MySQL - дуже популярна система управління базами даних;
- PHP - інтерпретатор серверної мови програмування, на якому написано більшість CMS і створено безліч інших веб-додатків.

Даним скороченням описується комплектація пакетів програм, під відповідну ОС, що позначається першою літерою. Походить від LAMP, де перша буква означає Linux. Існує також MAMP, під Mac OS. Дані пакети в основному використовуються для веб-розробки.

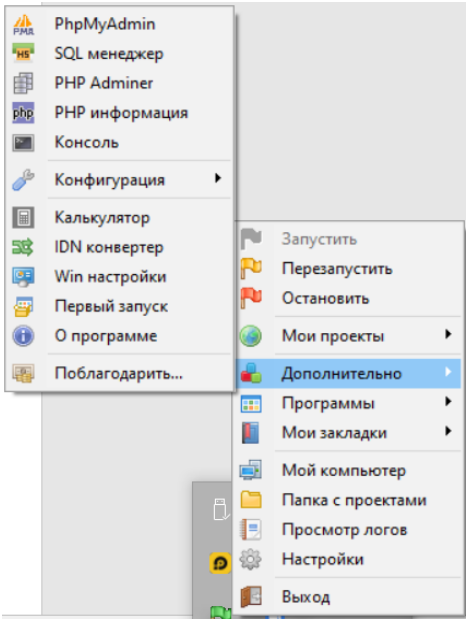

Рисунок 3.4 – Панель управління локального сервера Open Server

3.7 Вибір системи для управління версіями проекту

Система управління версіями (Version Control System, VCS) – це програмне забезпечення для полегшення роботи зі змінною інформацією. Система управління версіями дозволяє зберігати кілька версій одного і того ж документа, при необхідності повертатися до попередніх версій, визначати, хто і коли зробив ту чи іншу зміну, і багато іншого. Такі системи найбільш широко використовуються при розробці програмного забезпечення для зберігання вихідних кодів програми, що розробляється.

3.7.1 Версіонування

Розглянемо приклад програміста, який закінчив працювати над проектом і відправив фінальну версію замовнику. У програміста є папка, в якій зберігається фінальна версія проекту:

source /

project_index_final.php

Програміст закінчив роботу, але замовник надіслав у відповідь правки. Щоб була можливість повернутися до старої версії проекту, програміст створив новий файл project_index_final_2.php, вніс зміни і відправив замовнику:

source /

project_index_final.php

project_index_final_2.php

Цим все не обмежилося, в результаті структура проекту розрослася і стала виглядати так:

source /

project_index_final.php

project_index_final_2.php

...

project_index_final_19.php

...

project_index_latest_final.php

project_index_latest_final_Final.php

У професійній розробці використання нових файлів для версіонірованія є поганою практикою. Зазвичай у розробників в папці проекту зберігається безліч файлів. Також над одним проектом може працювати кілька людей. Якщо кожен розробник для версіонірованія створюватиме новий файл, трохи змінюючи назву попередньої версії, то незабаром в проекті почнеться хаос і ніхто не буде розуміти які файли потрібно відкривати.

3.7.2 Git

Для вирішення проблеми зі збереженням нової версії файлів зручно використовувати систему Git так як в середовищі розробки яка була обрана для проекту є плагіни і підтримка роботи з цією системою. Роботу Git можна порівняти з процесом збереження та завантаження в комп'ютерних іграх:

– якщо попереду чекає важкий бій, то перед цим краще заздалегідь зберегтися;
– щоб це зробити, потрібно виконати спеціальну команду;
– після чого збереження потрапляє в спеціальну папку і містить стан гри.

Тепер при необхідності завжди є можливість повернутися до попередньої версії гри.

Файли, необхідні для роботи програми, зберігаються в робочій області. В папці saves зберігається історія всіх збережень гри. Git зберігає код вашого проекту за таким же принципом: збереження потрапляють в спеціальну приховану папку, а робочою областю є вміст кореневої папки. Основні поняття:

Репозиторій. Проект, в якому була инициализирована система Git, називається репозиторієм. При ініціалізації в проект додається прихована папка .git. Репозиторій зберігає всі робочі файли та історію їх змін.

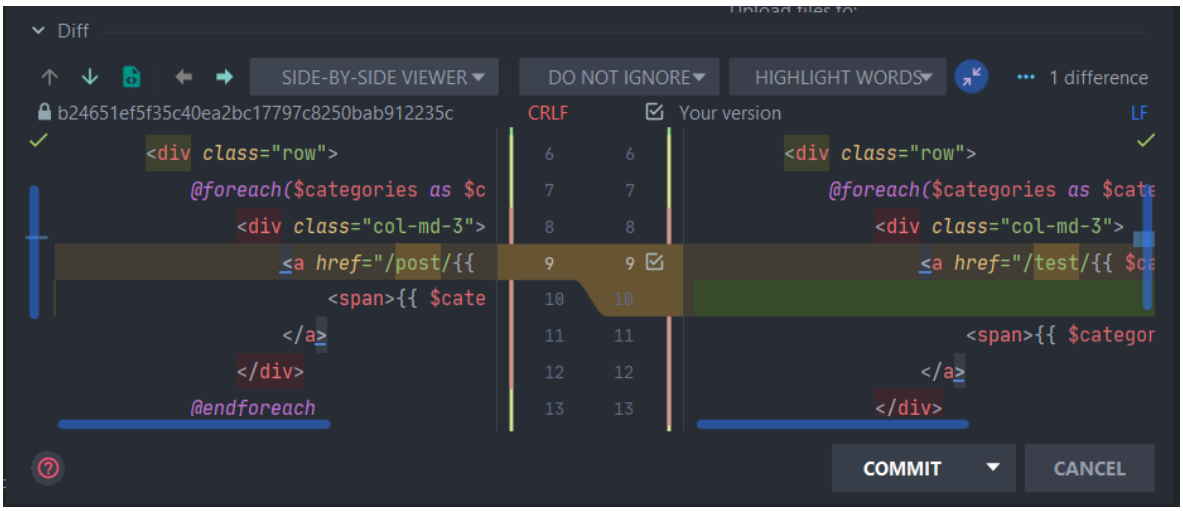

Рисунок 3.5 – Приклад комміту

Точно так же, як і в грі, в системі контролю версій Git можна зберегти поточний стан проекту. Для цього є спеціальна команда – commit. Вона робить так, що нова версія проекту зберігається і додається в сховище. У

файлі зі збереженням відображаються: всі зміни, які відбувалися в робочій області, автор змін і короткий коментар, що описує суть змін. Кожен Комміт зберігає повний стан робочої області, її папок і файлів проекту. У підсумку проект працює так:

Репозиторій зберігає всі версії проекту. У разі передачі цього проекту іншій людині, він побачить все, що з ним відбувалося до цього.

Нічого не втрачається і не видаляється безслідно. При видаленні файлу в новій версії додається запис про те, що файл був видалений.

Завжди можна повернутися до будь-якої з версій проекту, завантаживши її зі сховища в робочу область.

3.7.3 GitHub

GitHub – сервіс онлайн-хостингу репозиторіїв, що володіє всіма функціями розподіленого контролю версій і функціональністю управління вихідним кодом – все, що підтримує Git і навіть більше. Зазвичай він використовується разом з Git і дає розробникам можливість зберігати їх код онлайн, а потім взаємодіяти з іншими розробниками в різних проектах. Також GitHub може похвалитися контролем доступу, багтрекінгом, управлінням завданнями для кожного проекту. Мета GitHub – сприяти взаємодії розробників. До проекту, завантаженому на GitHub, можна отримати доступ за допомогою інтерфейсу командного рядка Git і Git-команд. Також є й інші функції, такі як документація, запити на прийняття змін (pull requests), історія коммітов, інтеграція з безліччю популярних сервісів, email-повідомлення, емодзі, графіки, вкладені списки завдань. Git – це інструмент, що дозволяє реалізувати розподілену систему контролю версій, а GitHub – це сервіс для проектів, що використовують Git.

Якщо робота над проектом ведеться в команді, то перед тим як почати писати код потрібно отримати останню версію проекту. Для цього потрібно виконати команду pull. Так ми забираємо всі зміни, які були здійснені з часу

останньої синхронізації з віддаленим репозиторієм. Тепер вони у нас в репозиторії на локальному комп'ютері.

Щоб відправити колегам останню версію проекту виконуємо команду push. Якщо у віддаленому репозиторії з моменту останньої синхронізації не було ніяких змін, то всі збережені зміни успішно завантажаться в хмару і колеги отримають останню версію проекту, виконавши команду pull. Якщо ж були зміни, то Git попросить вас перед відправкою підтягнути останні версії, зробивши pull.

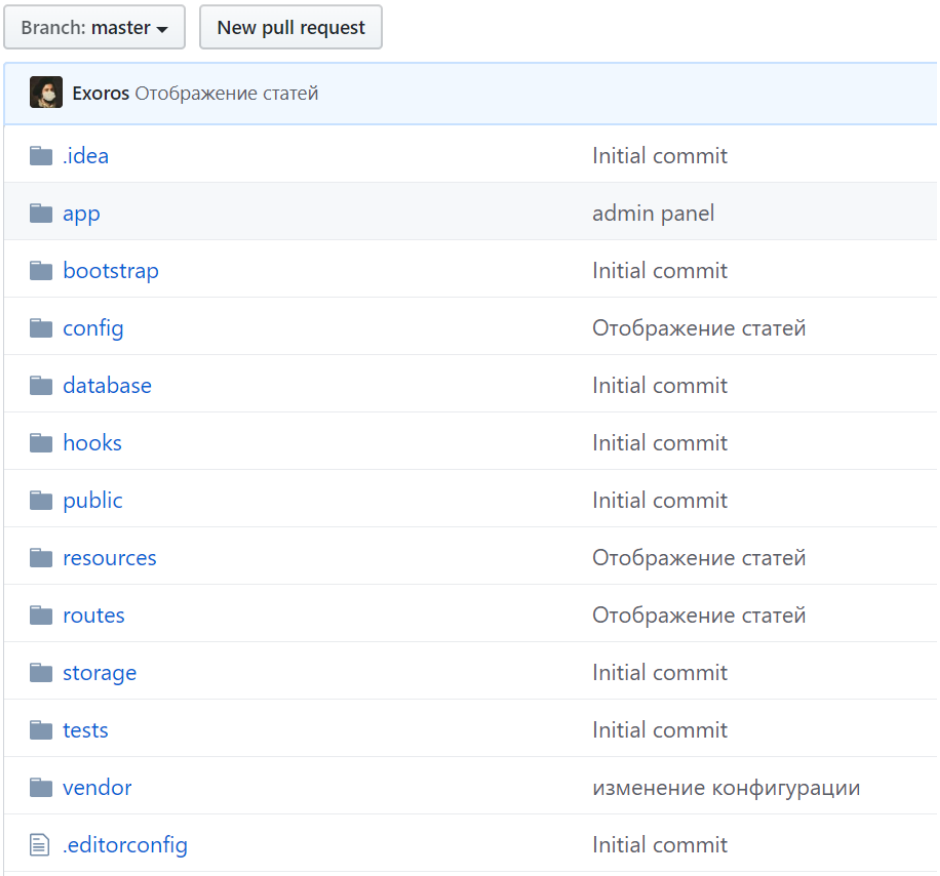

Рисунок 3.6 – Приклад команди push на віддалений репозиторій

4 ПРОЕКТНІ РІШЕННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Концепція MVC

Структура коду Laravel framework відповідає популярному паттерну проектування MVC, тобто в ньому можна виділити моделі (models), уявлення (views) і контролери (controllers)[11][1)].

Шаблон описує простий спосіб побудови структури додатка, метою якого є відділення бізнес-логіки від призначеного для користувача інтерфейсу. В результаті, додаток легше масштабується, тестується, супроводжується і звичайно ж реалізується.

Модель – містить бізнес-логіку додатка і включає методи вибірки (це можуть бути методи ORM), обробки (наприклад, правила валідації) і надання конкретних даних, що часто робить її дуже товстою, що цілком нормально. Модель не повинна безпосередньо взаємодіяти з користувачем. Всі змінні, що відносяться до запиту користувача повинні оброблятися в контролері.

Вид – використовується для завдання зовнішнього відображення даних, отриманих з контролера і моделі. Види містять HTML-розмітку і невеликі вставки PHP-коду для обходу, форматування і відображення даних. Види не повинні безпосередньо звертатися до бази даних. Цим повинні займатися моделі.

Контролер – сполучна ланка, що з'єднує моделі, види і інші компоненти в робочій додаток. Контролер відповідає за обробку запитів користувача. Контролер не повинен містити SQL-запитів. Їх краще тримати в моделях. Контролер не повинен містити HTML і інший розмітки. Її варто виносити в види.

[1)] [11] Довідник «Паттерни проектування». URL: http://design-pattern.ru/patterns/mvc.html

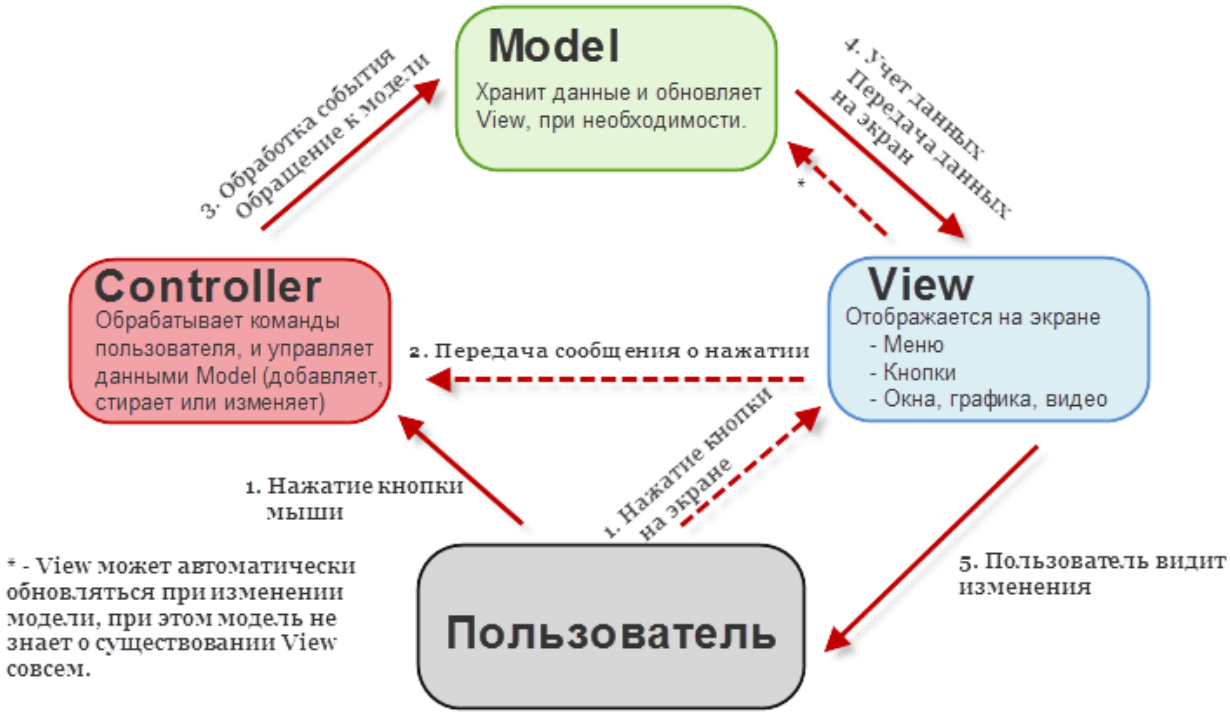

Рисунок 4.1 – Прицнип роботи MVC патерна

Типову послідовність роботи MVC-додатка можна описати таким чином:

При заході користувача на веб-ресурс, скрипт ініціалізації створює екземпляр додатку і запускає його на виконання.

При цьому відображається вид, скажимо головної сторінки сайту.

Додаток отримує запит від користувача і визначає контролер і дію. У разі головної сторінки, виконується дія за замовчуванням (index).

Додаток створює екземпляр контролера і запускає метод дії,

в якому, наприклад, міститися виклики моделі, що зчитують інформацію з бази даних.

Після цього, дія формує уявлення з даними, отриманими з моделі і виводить результат користувачеві.

4.2 Програми для проектування структури і контенту

Для написання веб порталу підходить і звичайний текстовий редактор блокнот, але краще використовувати середовище розробки з великим функціоналом, наприклад інтегроване середовище розробки, це збільшить швидкість роботи в рази, швидка орієнтація по проекту, і безліч інструментів. Інтегроване середовище розробки – це система програмних засобів, яка використовується програмістам для розробки програмного забезпечення. В англійській мові таке середовище називається Integrated development environment або скорочено IDE.

ІСР зазвичай включає в себе текстовий редактор, компілятор, інтерпретатор, засоби автоматизації розробки і збірки програмного забезпечення і відладчик. Іноді також містить засоби для інтеграції з системами управління версіями і різноманітні інструменти для спрощення конструювання графічного інтерфейсу користувача. Багато сучасних середовищ розробки також включають вікно перегляду програмних класів, інспектор об'єктів і діаграму ієрархії класів – для використання при об'єктно-орієнтованої розробки ПЗ. Більшість сучасних ІСР призначені для розробки програм на кількох мовах програмування одночасно.

Один з окремих випадків ІСР – середовища візуальної розробки, які включають в себе можливість візуального редагування інтерфейсу програми.

Основним вікном, є текстовий редактор, який використовується для введення вихідного коду в ІСР і орієнтований на роботу з послідовністю символів в текстових файлах. Такі редактори забезпечують розширену функціональність – підсвічування синтаксису, сортування рядків, шаблони, конвертацію кодувань, показ кодів символів. Іноді їх називають редакторами коду, так як основне їхнє призначення – написання вихідних кодів комп'ютерних програм.

Підсвічування синтаксису – виділення синтаксичних конструкцій тексту з використанням різних кольорів, шрифтів і накреслень. Зазвичай

застосовується в текстових редакторах для полегшення читання вихідного тексту, поліпшення візуального сприйняття. Часто застосовується при публікації вихідних кодів в Інтернет.

Для проекту було вирішено використовувати IDE JetBrains PhpStorm з використанням відповідних плагінів такі як Laravel, плагін для фраемворка, Git і Github для роботи з контролем версій проекту і безліч інших.

Програмне забезпечення JetBrains PhpStorm є спеціалізований засіб web-розробки, орієнтовані на web-додатки та інші види програм, які можна створювати за допомогою мови PHP і з використанням HTML, JavaScript і CSS. Рішення PhpStorm здійснює розгортання і синхронізацію проектів через протокол FTP. Середа PhpStorm пропонує функції автоматичного завершення мовних конструкцій PHP в коді, інспектування коду, різні алгоритми рефакторинга і швидку навігацію по коду.

Реалізований в PhpStorm графічний PHP-відладчик підтримує умовні точки зупину, відстеження значень і автоматизований вхід в налагодження окремих процедур. Для тестування додатків підтримується каркас тестових модулів PhpUnit і графічний інтерфейс для запуску тестів. При редагуванні коду виділяються конструкції синтаксису, здійснюється розширене форматування конфігурації, виявлення помилок в режимі реального часу і завершення коду. PhpStorm-редактор враховує коментарі до коду при його завершенні, автоматично вибираючи оптимальне рішення проблеми. PHP-рефакторинг і редагування шаблонів гарантує зміну проекту в найкоротші терміни. PhpStorm дозволяє візуалізувати код в ієрархічності вигляді і забезпечує швидку навігацію по всім елементам.

Завдяки застосуванню PHPUnit-тестів можна швидко переглядати результати генерації коду окремих блоків або всієї програми. Якщо тест був проведений невдало, продукт дозволяє переглядати окремі кодові рядки, в яких була виявлена помилка. PhpStorm забезпечує налагодження коду JavaScript і надає широкий діапазон можливостей: знаходження точки зупину

в HTML і JavaScript, настройка параметрів точки зупину, тестування синтаксису коду в режимі реального часу.

Сотні інспекцій піклуються про верифікації коду, аналізуючи проект цілком під час розробки. Підтримка PHPDoc, code (re) arranger, форматування коду з конфігурацією стилю коду і інші можливості допомагають розробникам писати охайний і легко-підтримуваний код.

Підтримуються передові технології веб-розробки, включаючи HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMAScript Harmony, шаблони Jade, Zen Coding, Emmet, і, звичайно ж, JavaScript.

PhpStorm включає в себе всю функціональність WebStorm (HTML / CSS редактор, JavaScript редактор) і додає повнофункціональну підтримку PHP і баз даних / SQL.

Основні можливості середовища це:

- Інтелектуальний редактор PHP коду з підсвічуванням синтаксису, автодоповнення коду, розширеними настройками форматування коду, запобіганням помилок нальоту;
- Підтримує PHP 7.0, 5.6, 5.5, 5.4 і 5.3, генератори, співпрограми і все синтаксичні поліпшення;
- PHP рефакторингом, code (re) arranger, детектор дубльованого коду;
- Підтримка Vagrant, Composer, вбудований REST клієнт, Command Line Tools, SSH консоль;
- Підтримка фреймворків (MVC view для Symfony2) і спеціалізовані плагіни для провідних PHP фреймворків (Symfony, Magento, Drupal, Yii, Laravel і багато інших);
- Візуальний відладчик для PHP додатків, валідація конфігурації відладчика, PHPUnit з покриттям коду (підтримка PHPUnit 5), а також інтеграція з профілювальником;
- HTML, CSS, JavaScript редактор. Налагодження і модульне тестування для JS. Підтримка HTML5, CSS, Sass, SCSS, Less, Stylus,

Compass, CoffeeScript, TypeScript, ECMAScript Harmony, Emmet і інших передових технологій веб-розробки;

- Повний набір інструментів для фронтенд-розробки;
- Підтримка стилів коду, вбудовані стилі PSR1 / PSR2, Symfony2, Zend, Drupal та інші;
- Інтеграція з системами управління версіями, включаючи уніфікований інтерфейс;
- Віддалене розгортання додатків і автоматична синхронізація з використанням FTP, SFTP, FTPS і ін; Live Edit: зміни в коді можна миттєво переглянути в браузері без перезавантаження сторінки;
- PHP UML;
- Інтеграція з баг-трекера;
- Інструменти роботи з базами даних, SQL редактор;
- Крос-платформенность (Windows, Mac OS X, Linux).

4.3 Концептуальне проектування

Концептуальне проектування – це побудова семантичної моделі предметної області та її опис. Процес такого проектування створюється без спрямованості на яку-небудь конкретну систему управління базами даних. ER-модель або ER-діаграма – це модель даних, яка дає можливість описати концептуальні схеми предметної області. Моделлю даних є графічне опис предметної області за допомогою стандартизованого набору позначень. Для застосування в обраній СКБД будується логічна модель основі ER-моделі. Логічна модель відображає процес проектування інформаційної моделі, яка виконується на основі створеної моделі даних, але без урахування певної СУБД і інших фізичних обмежень. Основні елементи, що входять до складу ER-моделей:

- об'єкти предметної області (сутності);
- безпосередньо зв'язку між об'єктами;

– властивості об'єктів (атрибути).

У кожній асоціації можна виділити два кінця (згідно парі сутностей, пов'язаних між собою), на кожному з яких встановлюється ступінь кінця зв'язку (кількість пов'язують примірників сутності). Моделювання зв'язків між об'єктами визначається типом зв'язку і класом приналежності. Основні типи зв'язків:

– «один до одного» (1: 1);

– «один до багатьох» (1: ∞);

– «багато до багатьох» (∞: ∞);

Клас приналежності є обов'язковим, якщо всі об'єкти одного класу беруть участь в зв'язку з об'єктами іншого класу або необов'язковим, якщо об'єкти одного класу не беруть участі в зв'язку з об'єктами іншого класу.

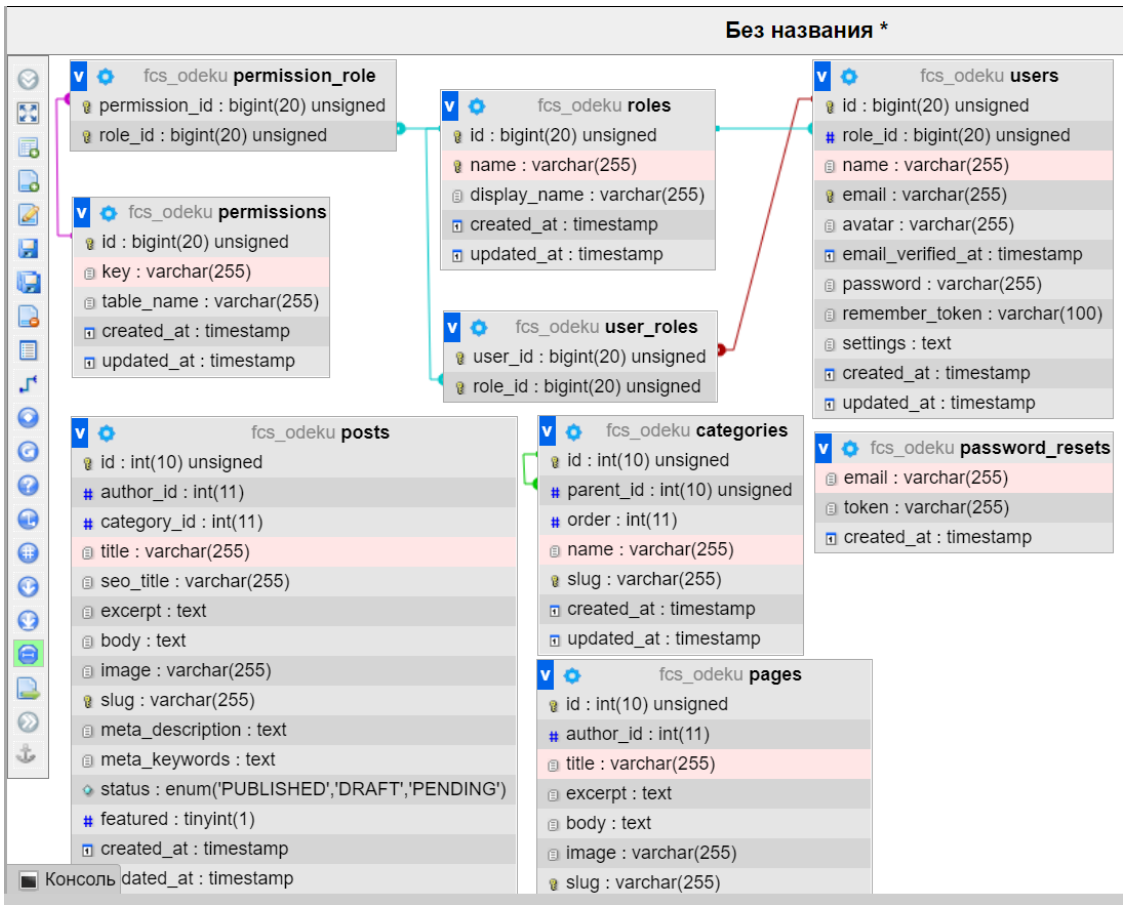

Рисунок 4.2 – Діаграма БД fcs_odeku

4.4 Створення баз даних, міграції

Для створення бази даних використовується середа phpMyAdmin, була створена база fcs_odeku.

```
APP_NAME=FCS
APP_ENV=local
APP_KEY=base64:GHBpPk0mvhQG9d5dwIHoUbe47BD8pyDkdwWw7WSoupI=
APP_DEBUG=true
APP_URL=http://fcs.odeku.edu.ua

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=fcs_odeku
DB_USERNAME=root
```

Рисунок 4.3 – Підключення бази даних fcs_odeku в .env файлі

Для створення таблиць в базі даних були використані міграції. Міграції – це щось на зразок системи контролю версій для бази даних. Вони дозволяють команді програмістів змінювати структуру БД, в той же час залишаючись в курсі змін інших учасників. Міграції зазвичай йдуть рука об руку з конструктором таблиць для більш простого поводження з архітектурою застосування.

Використовуючи команду php artisan make:migration створюється міграція наприклад для пунктів меню на рисунку 4.4.

```php
public function up()
{
    Schema::create('menus', function (Blueprint $table) {
        $table->increments( column: 'id');
        $table->string( column: 'name')->unique();
        $table->timestamps();
    });

    Schema::create('menu_items', function (Blueprint $table) {
        $table->increments( column: 'id');
        $table->unsignedInteger( column: 'menu_id')->nullable();
        $table->string( column: 'title');
        $table->string( column: 'url');
        $table->string( column: 'target')->default( value: '_self');
        $table->string( column: 'icon_class')->nullable();
        $table->string( column: 'color')->nullable();
        $table->integer( column: 'parent_id')->nullable();
        $table->integer( column: 'order');
        $table->timestamps();
    });

    Schema::table('menu_items', function (Blueprint $table) {
        $table->foreign( columns: 'menu_id')->references( columns: 'id')->on( table: 'menus')->onDelete( action: 'cascade');
    });
}
```

Рисунок 4.4 – Використання міграцій для створення таблиць

Таким же чином були створені всі таблиці бази даних, пости, категорії, сторінки, ролі, користувачі і так далі що в підсумковому варіанті має вигляд який зображен на рисунку 4.5

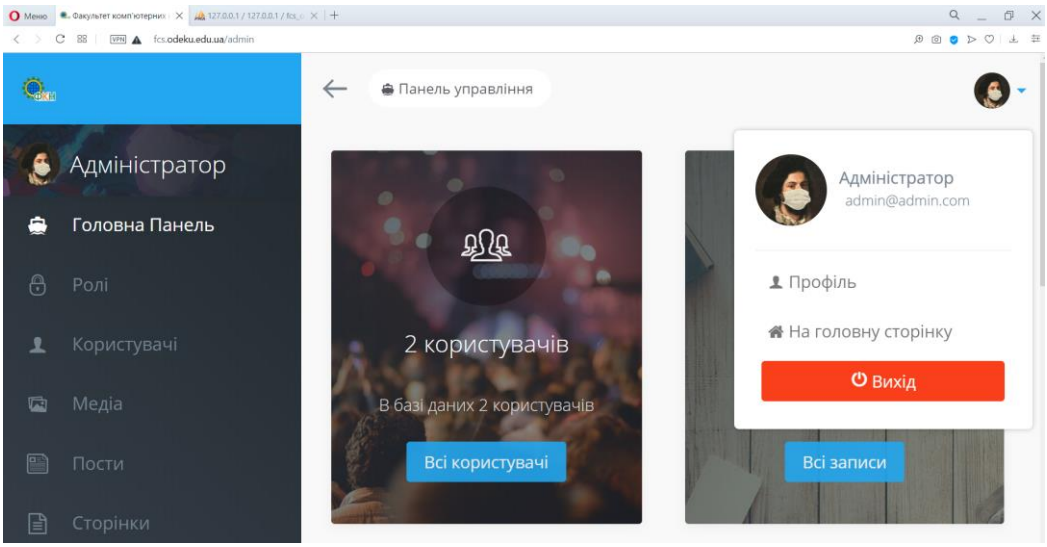

Рисунок 4.5 – Адміністративна панель

4.5 Робота з таблицями через модель, створення відношень

Система об'єктно-реляційного відображення (ORM) Eloquent – красива і проста реалізація шаблону ActiveRecord в Laravel для роботи з базами даних. Кожна таблиця має відповідний клас-модель, який використовується для роботи з цією таблицею. Моделі дозволяють запитувати дані з таблиць, а також вставляти в них нові записи. Всі моделі Eloquent успадковують клас Illuminate \ Database \ Eloquent \ Model. Найпростіший спосіб створити екземпляр моделі - за допомогою Artisan-команди make: model. Приклад створенної моделі для роботи с таблицею posts та створення відношень с таблицею categories зображен на рисунку 4.6

```php
class Post extends Model
{
    use Translatable;
    use Resizable;

    protected $translatable = ['title', 'seo_title', 'excerpt', 'body', 'slug', 'meta_description', 'meta_keywords'];

    const PUBLISHED = 'PUBLISHED';

    protected $guarded = [];

    public function save(array $options = [])
    {
        // If no author has been assigned, assign the current user's id as the author of the post
        if (!$this->author_id && Auth::user()) {
            $this->author_id = Auth::user()->getKey();
        }

        return parent::save();
    }
```

```php
    public function category()
    {
        return $this->belongsTo(Voyager::modelClass('Category'));
    }
}
```

Рисунок 4.6 – Модель Post для роботи з таблицями posts та створення зворотного відношення belongsTo до моделі Category таблиці categories

4.6 Механізм маршрутизації та контролери

Маршрутизація призначена для направлення запиту до відповідного контролера. Маршрути додатки можуть бути визначені в файлі app / Http / routes.php.

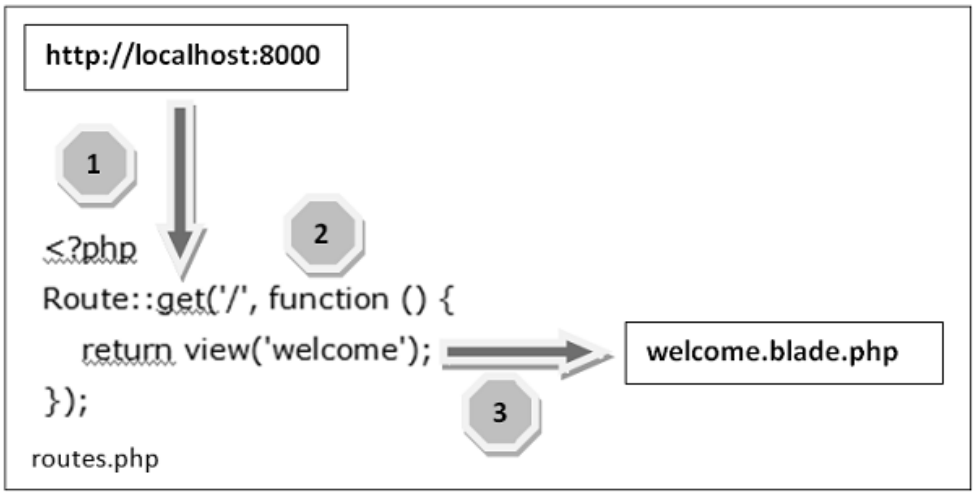

Рисунок 4.7 – Механізм маршрутизації

Замість того, щоб визначати всю логіку обробки запитів у вигляді замикань в файлах маршруту, було організовано її за допомогою класів контролерів. Контролери можуть групувати пов'язану з обробкою HTTP-запитів логіку в окремий клас. Контролери зберігаються в папці app / Http / Controllers. Для відображення наприклад постів був створений контролер в якому потрібно створити метод для відображення постів, назва методу show() в якому створена змінна $posts і метод повертає шаблон який в результаті буде відображати все з баз даних на виході, шаблон це файл news.blade. На рисунку 4.6 бачимо приклад класу PostController в якому знаходиться метод для відображення постів show() та зміна $posts бере всі значення all() звертаєчись до модели Post в якої створена структура татблиці з постами.

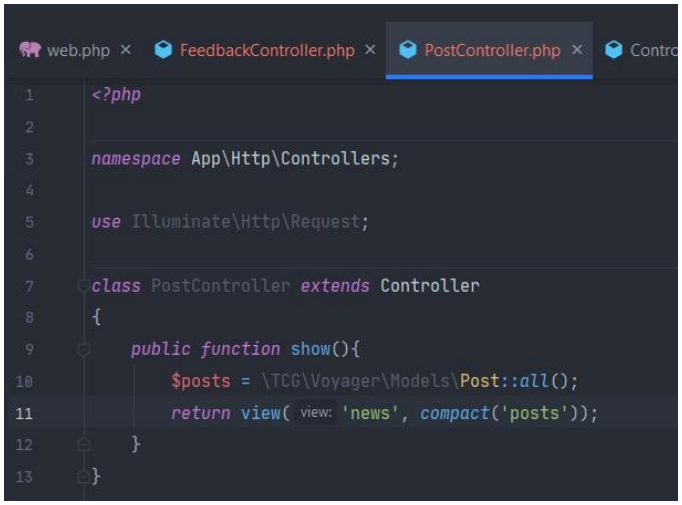

Рисунок 4.8 – Класс контролера PostController

4.7 Види (views) шаблонізатор Blade

Для відображення інформації потрібно створити файл шаблону blade.php. Blade – простий, але потужний шаблонизатор, що входить до складу Laravel. Blade заснований на концепції наслідування шаблонів і секціях. В шаблони вставляється php код, html розмітка, цикли для перебору даних з баз та багато іншого. Всі шаблони знаходяться в папці views.

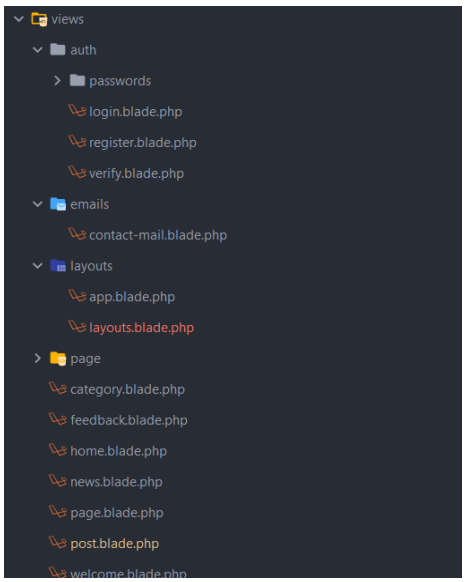

Рисунок 4.9 – Папка з шаблонами

Так для відображення постів, для редагування та видалення мають свої шаблони.

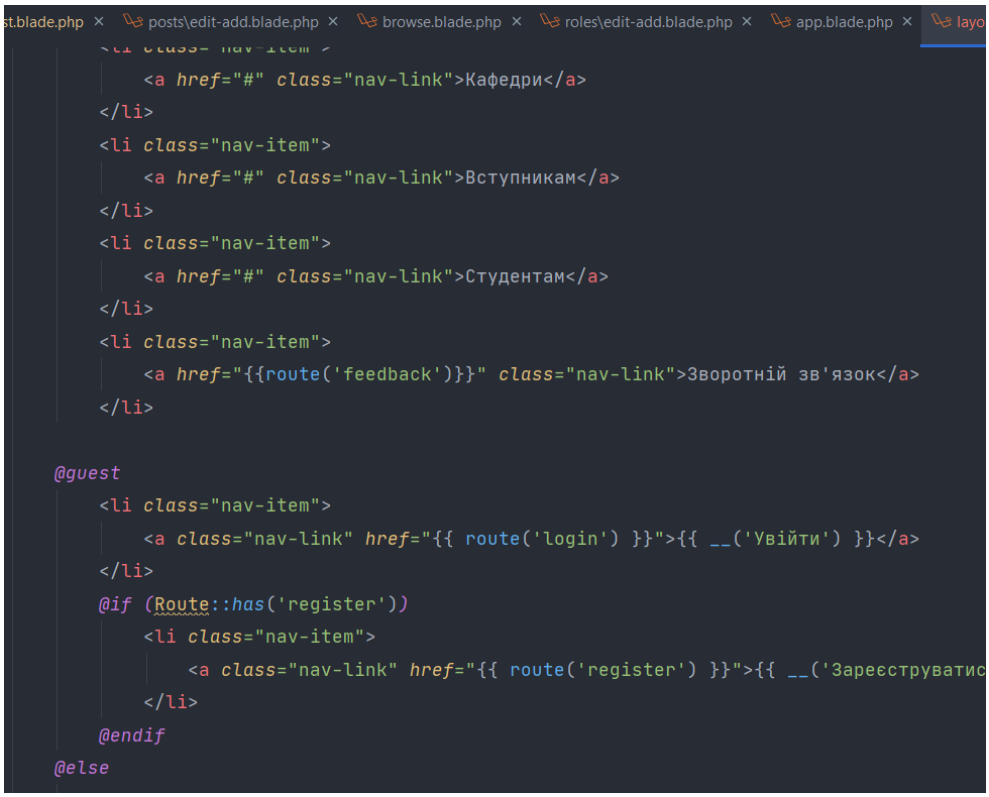

Рисунок 4.10 – Приклад коду головного шаблону layouts.blade.php

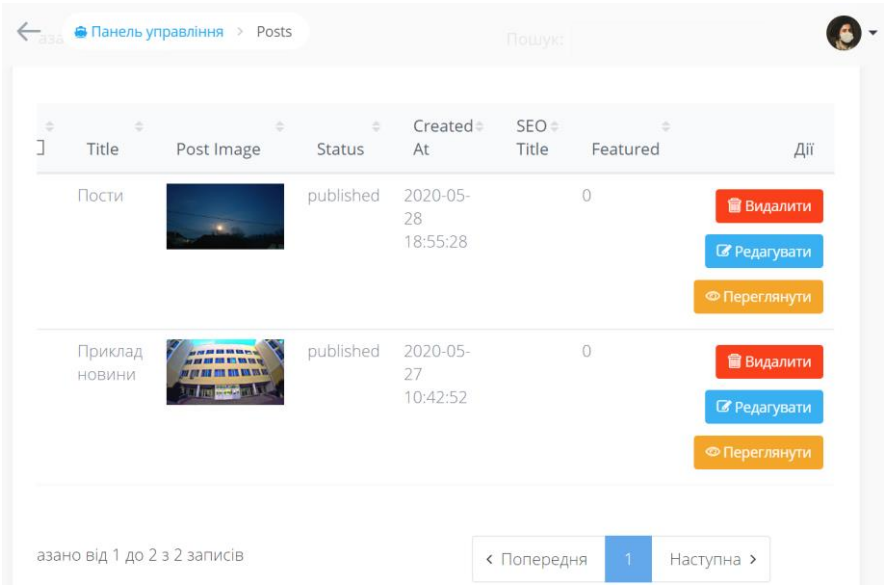

Рисунок 4.11 – Приклад сторінки для управління постами

ВИСНОВКИ

У результаті виконання даного дипломного проекту був зроблений веб-портал для факультету КНУА ОДЕКУ за допомогою программного засобу framework Laravel.

При розробці порталу було проаналізовано сучасні версії програмного забезпечення та актуальні засоби розробки сайтів.

При проектуванні були розглянуті всі можливості для того щоб користування порталом було найбільш зручним.

Розроблений портал задовольняє всім вимогам, поставленим на етапі постановки завдання, а саме:

- було виявлено група користувачів;
- використовуючи методологію моделювання UML було спроектовано роботу системи;
- обрані програмні засоби для розробки серверної частини порталу;
- створена база даних.

В результаті маємо адміністративну панель з різними можливостями для управління контентом через зручний інтерфейс, а саме:

- керування користувачами;
- призначення ролей для користувачів;
- додавання, редагування та видалення в базі даних новин, категорій, нових сторінок.

В якості подальшого використання порталу представляється можливим доробка структури порталу з метою подальшого підвищення його інформативності, функціональності та зручності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Рад Б. Я. Архітектура інформаційних систем / Б. Я. Рад, А. І. Водяхо, В. А. Дубенецький, В. В. Цехановскій. - М.: Академія, 2012. 220с.
2. Фрідман В. А., Александров А. В., Сергєєв Г. Г., Костін С. П. Будівництво Web-сайтів; Тріумф - Москва, 2011. - 288с.
3. Венедюхін Олександр, Воробйов Андрій Створення сайтів; Ексмо - Москва, 2011. - 528с.
4. Офіційний сайт скриптової мови PHP. URL: http://php.net/
5. Зандстра. PHP. Об'єкти, шаблони і методики програмування. / М. Зандстра- М: Вільямс, 2011. 560с.
6. Офіційний сайт документації фраемвока YII2. URL: https://www.yiiframework.com/doc/guide/2.0/ru
7. Документація фраемворку Symfony. URL: https://symfony.com.ua/doc/current/index.html
8. Офіційний сайт документації фраемворка Laravel. URL: http://laravel.su/
9. Мішель Е. Д. Вивчаємо PHP і MySQL / Е. Д. Мішель, А. Ф. Джон. - СПб.: Символ-Плюс, 2008. 442с.
10. MySQL керівництво адміністратора. - М.: Вильямс, 2005. - 621с.
11. Довідник «Паттерни проектування». URL: http://designpattern.ru/patterns/mvc.html

ДОДАТОК А

Лістинг класів міграцій

```
class CreateUsersTable extends Migration{
@return void
    public function up(){
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();});}
@return void
public function down(){
        Schema::dropIfExists('users');}
}class CreatePasswordResetsTable extends Migration{
    @return void
     public function up(){
        Schema::create('password_resets', function (Blueprint $t
able){
            $table->string('email')->index();
            $table->string('token');
            $table->timestamp('created_at')->nullable();});}
@return void
public function down(){
        Schema::dropIfExists('password_resets');}}
class CreatePagesTable extends Migration{
    @return void
    public function up(){
    Schema::create('pages', function (Blueprint $table) {
            $table->increments('id');
            $table->integer('author_id');
            $table->string('title');
            $table->text('excerpt')->nullable();
```

```
            $table->text('body')->nullable();
            $table->string('image')->nullable();
            $table->string('slug')->unique();
            $table->text('meta_description')->nullable();
            $table->text('meta_keywords')->nullable();
            $table->enum('status', Page::$statuses)-
>default(Page::STATUS_INACTIVE);
            $table->timestamps();});}
@return void
public function down(){
        Schema::drop('pages');}}
class CreatePostsTable extends Migration{
@return void
public function up(){
Schema::create('posts', function (Blueprint $table) {
            $table->increments('id');
            $table->integer('author_id');
            $table->integer('category_id')->nullable();
            $table->string('title');
            $table->string('seo_title')->nullable();
            $table->text('excerpt');
            $table->text('body');
            $table->string('image')->nullable();
            $table->string('slug')->unique();
            $table->text('meta_description');
            $table->text('meta_keywords');
            $table-
>enum('status', ['PUBLISHED', 'DRAFT', 'PENDING'])-
>default('DRAFT');
            $table->boolean('featured')->default(0);
            $table->timestamps();});}
@return void
public function down(){
        Schema::drop('posts');}}
class CreateCategoriesTable extends Migration{
@return void
public function up(){
```

```
Schema::create('categories', function (Blueprint $table) {
            $table->increments('id');
            $table->integer('parent_id')->unsigned()-
>nullable()->default(null);
            $table->foreign('parent_id')->references('id')-
>on('categories')->onUpdate('cascade')->onDelete('set null');
            $table->integer('order')->default(1);
            $table->string('name');
            $table->string('slug')->unique();
            $table->timestamps();});};}
@return void
public function down(){
Schema::drop('categories');}}
class AddUserFields extends Migration{
public function up(){
Schema::table('users', function ($table) {
if (!Schema::hasColumn('users', 'avatar')) {
          $table->string('avatar')->nullable()->after('email')-
     >default('users/default.png');}
          $table->bigInteger('role_id')->nullable()-
     >after('id');});};}
public function down(){
if (Schema::hasColumn('users', 'avatar')) {
Schema::table('users', function ($table) {
          $table->dropColumn('avatar');});};}
if (Schema::hasColumn('users', 'role_id')) {
Schema::table('users', function ($table) {
          $table->dropColumn('role_id');});};}}}
class CreateMenuTable extends Migration{
@return void
public function up(){
Schema::create('menus', function (Blueprint $table) {
          $table->increments('id');
          $table->string('name')->unique();
          $table->timestamps();});
Schema::create('menu_items', function (Blueprint $table) {
          $table->increments('id');
          $table->unsignedInteger('menu_id')->nullable();
```

```
            $table->string('title');
            $table->string('url');
            $table->string('target')->default('_self');
            $table->string('icon_class')->nullable();
            $table->string('color')->nullable();
            $table->integer('parent_id')->nullable();
            $table->integer('order');
            $table->timestamps();});
Schema::table('menu_items', function (Blueprint $table) {
            $table->foreign('menu_id')->references('id')-
     >on('menus')->onDelete('cascade');});}
@return void
public function down(){
Schema::drop('menu_items');
Schema::drop('menus');}}
class CreateRolesTable extends Migration{
@return void
public function up(){
Schema::create('roles', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string('name')->unique();
            $table->string('display_name');
            $table->timestamps();});}
@return void
public function down(){
Schema::drop('roles');}}
class CreatePermissionTable extends Migration{
@return void
public function up(){
Schema::create('permissions', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string('key')->index();
            $table->string('table_name');
            $table->timestamps();});}
@return void
public function down(){
Schema::dropIfExists('permissions');}}
```

```
class CreatePermissionRoleTable extends Migration{
@return void
public function up(){
Schema::create('permission_role', function (Blueprint $table) {
          $table->bigInteger('permission_id')->unsigned()-
     >index();
          $table->foreign('permission_id')->references('id')-
     >on('permissions')->onDelete('cascade');
          $table->bigInteger('role_id')->unsigned()->index();
          $table->foreign('role_id')->references('id')-
     >on('roles')->onDelete('cascade');
          $table->primary(['permission_id', 'role_id']);});};}
@return void
public function down(){
Schema::dropIfExists('permission_role');}}
```