

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр
(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБСАЙТА ДЛЯ ПІДГОТОВКИ
ДО ЗНО
SOFTWARE IMPLEMENTATION OF THE WEBSITE FOR GETTING
READY FOR EXTERNAL INDEPENDENT ASSESSMENT (ZNO)

Виконав: студент(ка) 4 курсу, групи КНТ– 227
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Вавілов М.О.
(прізвище та ініціали)

Керівник Терещенко Е.В.
(прізвище та ініціали)

Рецензент Казурова А.Є.
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти бакалавр

Спеціальність 122 Комп'ютерні науки

(код і найменування)

Освітня програма (спеціалізація) Комп'ютерні науки

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 2021 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Вавілова Миколи Олександровича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація вебсайта для підготовки до ЗНО. Software Implementation of the Website for Getting Ready for External Independent Assessment (ZNO)

керівник проєкту (роботи) Терещенко Еліна Валентинівна, к.ф.-м.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 17 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково – пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми.

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1– 4 Основна частина	Терещенко Е.В., доцент		
Нормоконтроль	Липовець М.В., асистент		

7. Дата видачі завдання “1” березня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2– 3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5– 6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Вавілов М.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Терещенко Е.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
86 с., 48 рис., 3 табл., 3 дод., 7 джерел.

JAVASCRIPT, PHP, КРОСПЛАТФОРМНІСТЬ, ІНТЕРНЕТ-ПОРТАЛ,
ВЕБЗАСТОСУНОК, БРАУЗЕР.

Об'єкт дослідження – реалізація інтернет-порталу для підготовки до ЗНО.

Предмет дослідження - сучасні інформаційні портали.

Мета роботи – розробка інтернет порталу для підготовки до ЗНО та надання актуальної інформації в цій темі.

Матеріали, методи та технічні засоби – мова програмування PHP, мова гіпертекстової розмітки HTML та опису стилів CSS, фреймворк Laravel, персональний комп'ютер з предумовленою операційною системою Microsoft Windows 10.

Результати. Створено динамічний вебсервіс за допомогою фреймворка Laravel, який дає можливість користувачам бачити останні новини й курси по ЗНО, скачувати останні тести ЗНО, та залишати відгук про сайт.

Висновки. Розроблено інтернет-портал для підготовки до ЗНО, в якого простий й зрозумілий інтерфейс, приємний дизайн, дозволяє підготуватися до ЗНО різними шляхами, є динамічним.

Галузь використання - може використовуватися користувачами будь-яких систем з підтримкою сучасних браузерів.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 86 pages, 48 figures, 3 tables, 3 appendixes, 7 sources.

JAVASCRIPT, PHP, CROSS-PLATFORM, INTERNET PORTAL, WEB APPLICATION, BROWSER.

Object of study is the implementation of an internet portal for preparation for external evaluation.

Subject of study are modern information portals.

The purpose of the work is to develop an internet portal to prepare for the external evaluation and provide relevant information on the topic.

Materials, methods and tools: PHP programming language, HTML hypertext markup language and CSS style description, Laravel framework, personal computer with pre-installed Microsoft Windows 10 operating system.

Result. A dynamic web service has been created using the Laravel framework, which allows users to see the latest news and courses on external evaluation, download the latest external evaluation tests, and leave feedback on the site.

Conclusions An internet portal for preparation for external evaluation has been developed, in which a simple and clear interface, pleasant design, allows to prepare for external evaluation in different ways, is dynamic.

The field of use – can be used by users of any system with support for modern browsers.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	9
1.1 Аналіз предметної області	10
1.2 Аналіз існуючих аналогів	11
1.3 Загальна схема системи	15
2 Розробка архітектури програми	17
2.1 Необхідні технології для створення вебсайтів	17
2.2 Технології створення динамічних сайтів	20
2.3 Вибір PHP фреймворка	22
2.4 Вибір фреймворку	24
2.5 Редактор коду PhpStorm	24
2.6 Діаграма взаємодії клієнта і сервера	26
2.7 Діаграми дій користувача у системі	28
2.8 Розробка дизайну сайту	29
3 Основні рішення щодо реалізації компонентів системи	31
3.1 Опис структури файлів проєкту	31
3.2 Створення бази даних	33
3.3 Створення логіки обробки HTTP-запитів	36
3.4 Опис шаблонів проєкту	39
4 Експлуатація та тестування програми	41
4.1 Призначення та умови застосування застосунку	41
4.2 Завантаження сайту на хостинг	42
4.3 Робота із застосунком	45
4.4 Можливі модифікації	49

4.5 Вхідні й вихідні дані	50
Висновки.....	51
Перелік джерел посилання	52
Додаток А Технічне завдання	53
Додаток Б Текст програми.....	57
Додаток В Слайди презентації.....	74

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
CLI	– Command-Line Interface;
CMS	– Content Manager System;
FAQ	– Frequently Asked Questions;
MVC	– Model-View-Controller;
ORM	– Object-Relational Mapping;
PSR	– PHP Standart Recommendation;
REST	– Representational State Transfer;
ЗНО	– зовнішнє незалежне оцінювання;
СУБД	– система управління базами даних.

ВСТУП

В наш час цифрових технологій не кожен має можливість користуватися такими джерелами інформації як книжки, журнали, посібники тощо. Традиційні засоби інформації не можуть довго зберігатися, а доступ до них не завжди можна отримати. Зараз більша частина людей має доступ до інтернету, майже без будь-яких обмежень.

Випускники 11 класу мають скласти зовнішнє незалежне оцінювання (ЗНО), результати якого відображають рівень знань. Відповідно до результату ЗНО абітурієнт може визначитися з тим, куди подавати свої документи для вступу або у вищий навчальний заклад, або у коледж.

Кожен по різному готується до ЗНО, але найчастіше всі звертають свою увагу на сайти підготовки до ЗНО. На таких сайтах в найбільш стислій та якісній формі надано інформацію щодо тем з кожного предмета, по яким потрібно готуватися, та й тести за минулі роки аби перевірити себе й свої знання.

Завданням дипломної роботи є реалізація вебсайту підготовки до ЗНО, який дозволяє користувачам вільно переглядати тести ЗНО, вирішувати їх, мати нагоду подивитися на свої помилки для подальшого їх виправлення.

Основними вимогами до сайту будуть:

- створення інтуїтивно-зрозумілого інтерфейсу;
- база даних, в якій будуть зберігатися результати кожного користувача;
- швидкий та якісний доступ до сайту.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

Інформаційний портал – це великий інтернет ресурс, що надає детальну інформацію на певну тематику. Інформаційний портал, на відміну від простого тематичного сайту, містить інструменти для взаємодії з користувачами, тематичні розсилки, пошук а також дозволяють відвідувачам спілкуватися між собою, наприклад на форумі або в чаті.

Інформаційні портали зазвичай створюються для того, щоб будь-який бажаючий міг отримати відповідь на своє питання з певної тематики, скориставшись, інформацією з сайту або запитавши у інших відвідувачів.

Одним з найбільш поширених прикладів інформаційного порталу є міські інформаційні портали, що містять новини, погоду, афішу, розклад транспорту, дошку оголошень, форум, каталог підприємств та іншу необхідну інформацію.

Класифікація по спеціалізації інформації:

- горизонтальний портал – це портал, що охоплює багато тематик, представляє набір сервісів(які обслуговуються, по можливості, всі теми) й орієнтовані на максимально широку аудиторію, на максимальне охоплення її інтересів;
- вертикальний портал – це портал вузької тематичної спрямованості, що надає різні сервіси для користувачів по певних інтересах і орієнтований на повне охоплення тематики або галузі діяльності;
- змішаний портал – портал, що сполучає в собі функції електронної торгівлі та класичні довідкові сервіси. Прикладами таких порталів є інтернет-магазини.

1.2 Аналіз існуючих аналогів

1.2.1 Опис інтернет порталу znosclub

Znosclub – інтернет портал, посвячений ЗНО. Він має повний ‘арсенал’ для того, аби будь-який бажаючий мав можливість скласти ЗНО. Тут є як і онлайн тести, так і багато новин в розділах Довідник ЗНО, або Підготовка до ЗНО. Є можливість обрати з якого предмету тебе цікавлять новини (рис. 1.1).

Інші аналоги зазвичай мають лише тести и можливість побачити свої результати та помилки, але на цьому порталі є можливість зайти на форум й пошукати допомогу там, якщо ви чогось не знайшли в тестах або в стрічці новин.

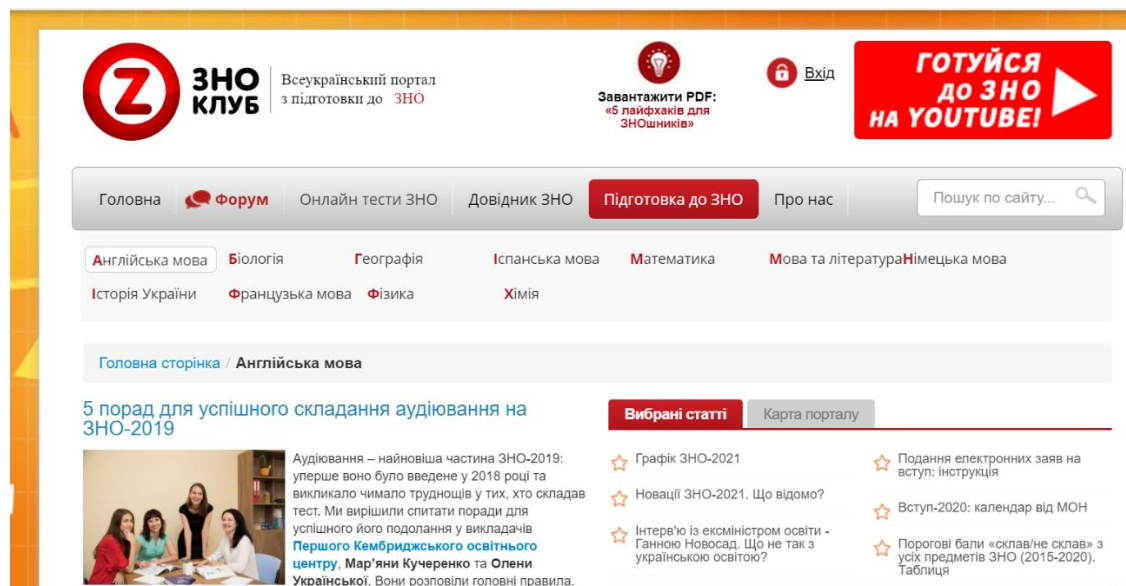


Рисунок 1.1 – Інтерфейс Znosclub. Головна сторінка

Загальні переваги платформи:

- стрічка новин з актуальною інформацією про ваш предмет;
- форум на якому можуть допомогти інші користувачі;
- Власний особистий кабінет з можливістю швидко користуватись форумом.

1.2.2 Опис інтернет порталу osvita

Власник сайту osvita надає простий та зрозумілий інтерфейс користувачеві, приємний для очей стиль виконання й кольорову палітру в будь-якому розділі його порталу. На даній інтернет сторінці, окрім самого ЗНО, можна знайти й дорожню картку учасника ЗНО.

На даному сайті (рис. 1.2) на відміну від попереднього, усі новини офіційні й є можливість зміни мови на російську при бажанні, невеликий бонус тут показана сьогоднішнє число й місяць.

Для тих хто заглядає наперед є корисні вкладки Вища освіта та Освіта за кордоном, а якщо у вас питання щодо законів є й окрема вкладка для цього.

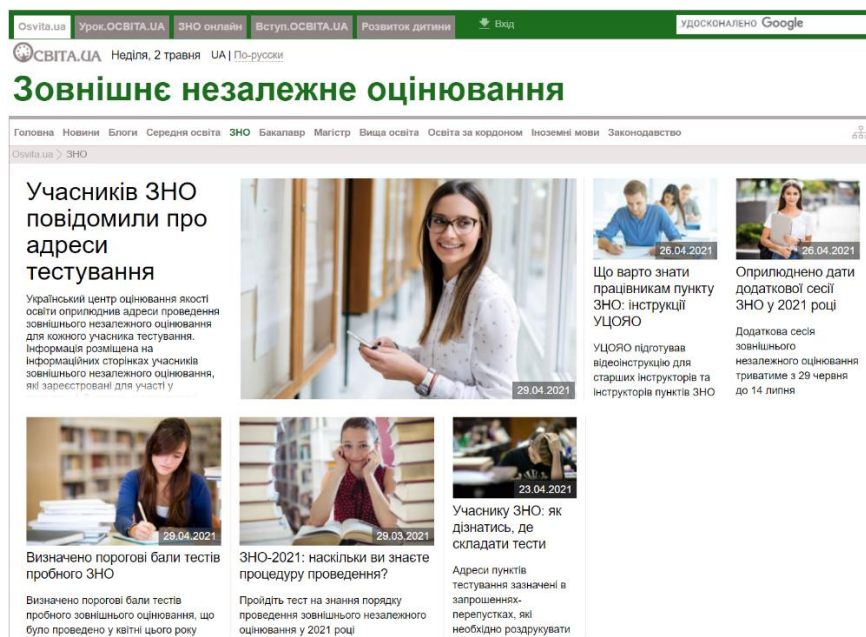


Рисунок 1.2 – Інтерфейс Osvita. Головна сторінка

Корисні відмінності від минулого сайту:

- багато корисної та офіційної інформації;
- в кожній частині цього порталу є корисний функціонал;

- має свою пошукову строку по усьому порталу;
- безкоштовний доступ до інформації.

1.2.3 Опис інтернет порталу naurok

Ця інтернет платформа як можна побачити на (рис. 1.3) більш сфокусована на поданні курсів, які на відміну від тестів своїх попередників не безкоштовні, але ви працюєте с живою людиною онлайн, ви можете задавати питання якщо щось не зрозуміло, спілкуєтесь з іншими, маєте домашнє завдання яке допомагає не забути здобуті знання та навички.

Для тих хто сумнівається в цьому сайті є відгуки та контакти завдяки яким можна задати всі турбуючи вас питання.

Тести на цьому порталі також присутні, вони більш узької направленості, а саме основні теми уроків, дуже легко знайти саме те чого ви недовчили.

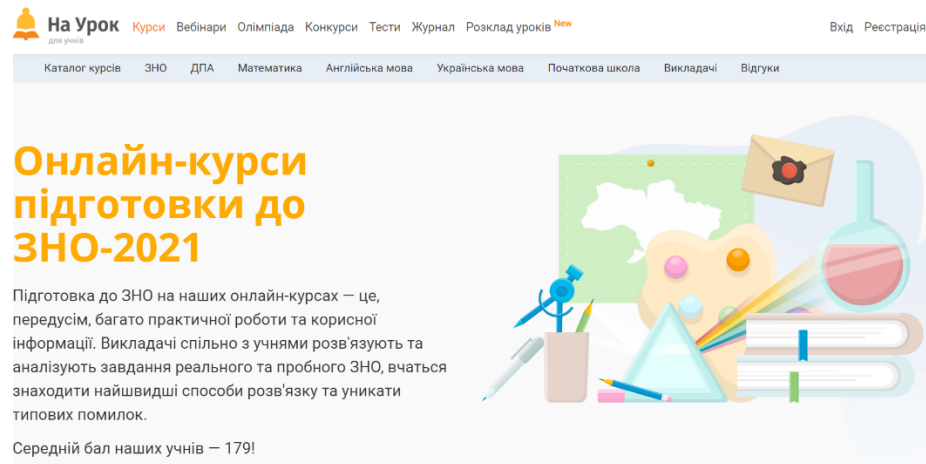


Рисунок 1.3 – Інтерфейс Naurok. Головна сторінка

Переваги платформи:

- курси з реальними викладачами та учнями;
- гаряча лінія підтримки й інша контактна інформація;

- більш цілеспрямовані тести, розбір по темам;
- олімпіади й конкурси для саморозвитку.

1.2.4 Опис Інтернет порталу besmart study

Останній, але не менш корисний сайт be smart (рис. 1.4), за підтримки міністерства освіти і науки України. Новий сайт з не багатою але якісною базою тестів пробного ЗНО. Підійде для тих, хто осилив попередників, й шукає швидкі й безкоштовні тести.

На відміну від минулих інтернет порталів має в базі даних відео лекції які можна включити прямо під час виконання тестів.

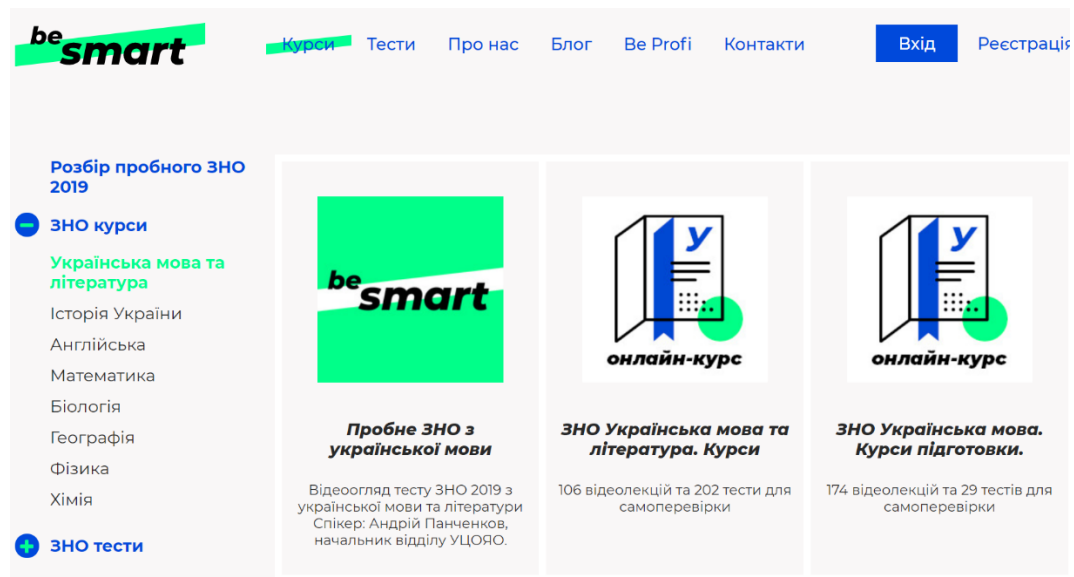


Рисунок 1.4 – Інтерфейс Besmart. Головна сторінка

Переваги платформи:

- відео лекції;
- підтримка міністерства освіти й свіжа база курсів;
- блог з новинами ресурсу.

1.3 Загальна схема системи

Завдяки проведеним дослідженням аналогів створюваного інтернет ресурсу, було визначено вимогу до функціоналу проєкту. Технічне завдання наведено у додатку А.

Розглянемо загальну схему роботи системи.

Спочатку користувач переходить на інтернет портал. Одразу бачимо головну сторінку portalу, на якій містяться новини, пошук для цих самих новин, розділ з курсами ЗНО минулих років, й пробники цього року, інформаційна вкладка про цей сайт з контактами й технічною підтримкою й відгуки. Для роботи з тестами немає необхідності в реєстрації, але при бажанні зберігати свої результати й мати можливість до них повертатися необхідна зареєструватися та авторизуватися.

Основний функціонал сайту має вміщувати в себе:

- головна сторінка;
- вкладка реєстрації;
- вкладка авторизації;
- вкладка новин;
- вкладка тести;
- вкладка курсів;
- вкладка про нас;
- вкладка FAQ;
- вкладка відгуків.

Головна сторінка буде вміщувати в себе основну панель для переходу по вкладкам, кнопки реєстрації та авторизації, більш зжятий варіант стрічки новин та коротка довідка про нас.

Сторінка РЕЄСТРАЦІЇ має поля для вводу електронної пошти, ім'я користувача та пароллю, кнопки для підтвердження реєстрації та переходу до сторінку авторизації у разі, якщо користувач вже зареєстрований.

Сторінка АВТОРИЗАЦІЇ має поля для вводу логіну та пароллю й підтвердження введених даних, а також кнопку на випадок, якщо користувач забув свій логін або пароль.

Вкладка НОВИНИ вміщуватиме в себе більш розгорнуту стрічку новин с головної сторінки, на якій будуть викладені останні публікації з офіційних джерел інформації, та будуть розсортовані по даті викладення від нових до старих за датою їх публікування.

Вкладка ТЕСТИ містить у собі тести по ЗНО, вони будуть надані у форматі PDF кожен для свого предмета. Окрім самих тестів буде надано відповіді до тестів, шкала оцінки та для деяких спец. завдання з розгорнутим вирішенням.

На вкладці курсів буде запропоновано багато різних ресурсів, на яких можна буде обрати підходящий для кожного окремого користувача курс, з будь якого предмета для ЗНО предмета.

На вкладці ПРО НАС буде вказано робочі години нашої підтримки, номер гарячої лінії, по якому можна буде зв'язатися з нашим технічним спеціалістом, якщо виникла якась помилка на сайті, робоча пошта для бажаючих співпрацювати з нами й посилання на наші соціальні мережі.

На вкладці FAQ можна переглянути найчастіші запитання пов'язані з сайтом або ЗНО.

На вкладці відгуки буде відображено відгуки користувачів порталу, їх враження від користі інформації на сайті, актуальності тестів, та можливі побажання.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Необхідні технології для створення вебсайтів

2.1.1 HTML

HTML (англ. HyperText Markup Language — мова розмітки гіпертексту) — це мова тегів, засобами якої здійснюється розмічання вебсторінок для мережі Інтернет. Браузери отримують HTML-документи з вебсервера або з локальної пам'яті й передають документи в мультимедійні вебсторінки. HTML описує структуру вебсторінки семантично і спочатку включені сигнали для зовнішнього вигляду документа.

Елементи HTML є будівельними блоками сторінок HTML. За допомогою конструкцій HTML, зображення та інші об'єкти, такі як інтерактивні форми, можуть бути вбудовані у візуалізовану сторінку. HTML надає засоби для створення структурованих документів, позначаючи структурну семантику тексту, наприклад заголовки, абзаци, списки, посилання, цитати та інші елементи. Елементи HTML окреслені тегами, написаними з використанням кутових дужок. Теги, такі як і безпосередньо вводять вміст на сторінку. Інші теги, такі як `<img />` `<input />` `<p>` оточують і надають інформацію про текст документа і можуть включати інші теги як піделементи. Браузери не показують теги HTML, але використовують їх для інтерпретації вмісту сторінки.

HTML може вбудовувати програми, написані на мові сценаріїв, наприклад JavaScript, що впливає на поведінку та вміст вебсторінок. Включення CSS визначає вигляд і компоновання вмісту. World Wide Web Consortium (W3C), який супроводжує стандарти HTML та CSS, заохочує використання CSS над явним презентаційним HTML з 1997 року.

HTML впроваджує засоби для:

- створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
- отримання інформації із Всесвітньої мережі через гіперпосилання;
- створення інтерактивних форм;
- включення зображень, звуку, відео, та інших об'єктів до тексту [1].

2.1.2 CSS

CSS - це формальна мова, яка слугує для опису оформлення зовнішнього вигляду документа, створеного з використанням мови розмітки (HTML, XHTML, XML). Назва походить від англійського Cascading Style Sheets, що означає «каскадні таблиці стилів».

Навіщо використовується CSS.

Призначення CSS - відокремлювати те, що задає зовнішній вигляд сторінки, від її змісту. Якщо документ створено тільки з використанням HTML, то в ньому визначається не тільки кожен елемент, але і спосіб його відображення (колір, шрифт, положення блоку і тощо). Якщо ж підключені каскадні таблиці стилів, то HTML описує тільки черговість об'єктів. А за все їх властивості відповідає CSS. В HTML досить прописувати клас, не перераховуючи всі стилі кожен раз.

Така технологія:

- забезпечує відносно просту і швидку розробку, тому що одного разу створене оформлення можна застосовувати до багатьох сторінок;
- підвищує гнучкість і зручність редагування - досить внести правку в CSS, щоб оформлення змінилося всюди;
- робить код більш простим, знижуючи повторюваність елементів. Його простіше читати програмістам і пошуковим роботам;

- прискорює час завантаження, тому що CSS може кешуватися при першому відкритті, а в наступних зчитуються тільки структура і дані;
- збільшує кількість візуальних рішень для подання вмісту;
- забезпечує можливість легко застосовувати до одного документу різні стилі (наприклад, створювати адаптовану версію для мобільних пристроїв або спеціальні стилі для людей з вадами зору).

Тобто каскадні таблиці служать не тільки для втілення дизайну, а й кардинально змінюють підхід до побудування сайтів, спрощуючи працю розробників і забезпечуючи гнучкість реалізації. Ось для чого потрібен CSS [2].

2.1.3 JavaScript

JavaScript - це об'єктно-орієнтована мова скриптів, яка використовується для розробки вбудованих додатків, які виконуються як на стороні клієнта, так і на стороні сервера.

Частіше за все вони виконуються в формі JavaScript в клієнтській частині і реалізуються в додатку у вигляді інтегруємого веббраузером компонента, що дозволяє розробляти покращені інтерфейси і динамічні вебсайти.

JavaScript використовує можливості середовища, в якому виконуються написані на ньому сценарії. JavaScript являється діалектом стандарту ECMAScript і охарактеризований як динамічна мова скриптів.

На JavaScript вплинули багато мов. Його не рідко порівнювали з іншими мовами програмування, але в підсумку прийшли до висновку, що JavaScript це в дійсності особлива мова [3].

JavaScript доволі компактна та гнучка мова. Розробники забезпечили велике розмаїття інструментів, що доповнюють основу мови JavaScript, які відкривають величезну кількість додаткового функціоналу з мінімальними зусиллями. Серед них:

- програмні інтерфейси (APIs) для браузерів — API, які вбудовані у браузери, що надають функціонал на зразок динамічного створення HTML та застосування CSS-стилів, збір та обробка відео-потоків з вебкамери користувача, генерація 3D-графіки та аудіо-семплів;
- API третіх осіб, що дозволяють розробникам інтегрувати у власні сайти функціонал інших провайдерів, таких як Twitter або Facebook;
- фреймворки та бібліотеки третіх осіб, які ви можете застосувати до вашого HTML, щоб прискорити створення сайтів та застосунків [4].

2.1.4 PHP

PHP (рекурсивний акронім словосполучення PHP: Hypertext Preprocessor) - це поширена мова програмування загального призначення з відкритим вихідним кодом. PHP спеціально сконструйований для веброзробок і його код може впроваджуватися безпосередньо в HTML.

Замість рутинного виводу HTML-коду командами мови (як це відбувається, наприклад, в Perl або C), скрипт PHP містить HTML з вбудованим кодом. Код PHP відділяється спеціальними початковими й кінцевими тегами `<?php` и `?>`, які дозволяють "переключатися" в "PHP-режим" и виходити з нього.

PHP вкрай легкий для засвоєння, але разом з тим здатен задовільнити запити професіональних програмістів. Не лякайтесь великого списку можливостей PHP. Ви можете швидко розпочати, й вже протягом кількох годин зможете створювати прості PHP-скрипти [5].

2.2 Технології створення динамічних сайтів

Статичні сайти — це прості з незмінними вебсторінками сайти, які зберігаються на у готовому вигляді на сервері та оновлюються в ручну

власниками, кожен користувач такого сайту буде отримувати таку саму сторінку на свій запит до сервера як и будь який інший.

На противагу статичним сайтам, з часом прийшли динамічні, оскільки потреби користувачів з часом становились все більш складними та вимогливими, та й розробка почала еволюціонувати.

Динамічні сайти можуть автоматично змінюватися в плані змісту і вигляду залежно від контексту дій та уподобань користувача, часу та інших факторів, вебсторінки генеруються з використанням програмного коду та інформації з баз даних і передаються браузеру, вони оновлюються часто і не обов'язково фахівцями і власниками [6].

Для створення динамічних сайтів є два варіанти. Перший варіант вимагаючий менше зусиль – це скористатися вже існуючими системами з використанням CMS. Другий за допомогою фреймворків – це такий каркас для початку розробки.

CMS – це програмне забезпечення для організації вебсайтів чи інших інформаційних ресурсів в інтернеті. Цих систем існують тисячі, кожна з своїм призначенням й функціоналом. Перевагою даних систем є вже готовий набір інструментів й зазвичай вже готова та зручна адмін-панель.

CMS мають декілька способів роботи:

- генерація сторінок за запитом. Системи такого типу працюють на основі зв'язки «модуль редагування → база даних → модуль представлення». Модуль представлення генерує сторінку з контентом при запиті на нього на основі інформації з бази даних. Інформація в БД змінюється за допомогою модуля редагування. Сторінки заново створюються сервером при кожному запиті, а це створює навантаження на сервер. Але це навантаження може бути багатократно зменшене при використанні методів кешування, які є в сучасних вебсерверах;

- генерація сторінок при редагуванні. Системи цього типу при редагуванні сторінок вносять зміну у вміст сайту та створюють набір статичних сторінок. При такому способі втрачається інтерактивність між відвідувачами сайтів та контентом даного сайту;

- змішаний тип. Як зрозуміло із назви, цей тип поєднує в собі переваги перших двох. Може бути реалізований шляхом кешування — модуль представлення генерує сторінку один раз, надалі вона через деякий час буде в декілька разів швидше завантажуватися із кешу. Кеш може оновлюватись як автоматично, через деякий час чи при внесенні змін у певні розділи сайту, так і вручну за командою адміністратора. Другий підхід — збереження певних інформаційних блоків на етапі редагування сайту і збирання сторінок з цих блоків при запиті відповідної сторінки користувачем [7].

Фреймворки базуються на шаблоні MVC – він передбачає поділ системи на три взаємопов'язані частини: модель даних, вигляд та модуль виконання. Це відокремлення застосовується для мінімізації впливу цих самих частин одна на одну під час їх змін. Завдяки такому гнучкому типу програмного забезпечення полегшуються подальші зміни в програмі й робить структуру більш впорядкованою.

2.3 Вибір PHP фреймворка

2.3.1 Фреймворк CodeIgniter

CodeIgniter – це популярний MVC фреймворк на мові PHP який вийшов в світ у 2006, та швидко набрав популярність завдяки простоті і швидкості. Покинув свого творця у 2013 році перейшовши до рук Технологічного Інституту Британської Колумбії. Остання версія 4.0.4 яка підтримує працю на будь якому хостингу з підтримкою PHP 5.1 й вище.

Простота цього фреймворку пояснюється завдяки таким факторам:

- якісна й повна документація з прикладами, власна Wiki;
- велика кількість відео-уроків;
- не має певних структурних обмежень;
- не вимагає вчити як користуватися генераторами кода з командної строки.

2.3.2 Фреймворк Symfony

Symfony – це один з найпопулярніших фреймворків в світі веброзробки. Написаний на базі MVC й має в своїй структурі інші патерни об'єктно-орієнтованого програмування.

Коли Symfony2 перейшов з PHP 5.2.7 на версію PHP 5.3 й став використовувати Composer, для нього було закінчено документацію й повністю переведено на англійську мову. Одразу почалась міграція крупних проектів на Symfony: OroCRM, EzPublish, PHPBB, Piwik, PrestaShop та багато інших – частина використала повністю цей каркас, а інші використали лише окремі програмні компоненти.

Компоненти Symfony гарно задокументовані, їх підтримує велика спільнота й вони постійно розвиваються.

Найбільш популярними компонентами являються:

- компонент Console - надає повноцінну консоль з аргументами й управлінням параметрами, необхідними для PHP в режимі CLI;
- компоненти Debug і VarDumper - які надають кращий контроль стека у випадку помилки і функцію дампа в якості заміни var_dump;
- компоненти Finder і Filesystem - надає комфортний API для пошуку й маніпуляцій с папками та файлами;
- компоненти Polyfill-* - дозволяє використовувати функції, недоступні через версію PHP або через відсутність необхідних розширень.

2.3.3 Фреймворк Laravel

Laravel – це гнучкий та найбільш відомий фреймворк, який підходить для вирішення багатьох задач, від простих до найбільш складних та нестандартних. Завдяки широкому функціоналу є можливість відтворювати структуровану архітектуру, будувати кеширування та маршрутизацію RESTful. Одна з перших ця платформа почала працювати з стандартом PSR-4, за допомогою вдається структурувати ресурс відповідно до заказу.

Обираючи Laravel, ви отримуєте можливість швидко створювати вебресурси з нуля. При цьому програміст отримує велике різноманіття інструментів для вирішення будь-яких задач.

Цей фреймворк визнано одним з найкращих завдяки різноманіттю інструментів, бібліотек, фактично необмеженими можливостями та доступності.

2.4 Вибір фреймворку

Основаючись на проведених дослідженнях, а також беручи до уваги знайомство з Laravel, було вирішено використовувати фреймворк Laravel для розробки інтернет порталу.

2.5 Редактор коду PhpStorm

PhpStorm – платний редактор коду (IDE), який використовується для створення вебзастосунків (рис. 2.1). В цього редактору є безкоштовний пробний період в 30 днів. Він має не складний інтерфейс, в якому не складно розібратися, вбудовані бібліотеки функцій які спростять початок роботи на даній платформі та автопідстроювання коду, виправлення. Цей програмний

продукт дає можливість працювати з використанням PHP, SQL, XML, Ruby має підтримку баз даних, повна підтримка фронтенд-технологій HTML, CSS, JavaScript, Sass, Less, Stylus. При цьому доступні рефакторинг, отладка і юніт-тестування. Завдяки функції Live Edit всі зміни можна одразу ж побачити в браузері. Вбудовані Інструменти допомагають зручно вирішувати одноманітні задачі. IDE інтегрована з системами контролю версій, підтримує віддалене розгортання, бази даних і SQL, інструменти командної строки, Docker, OpenServer, Composer, REST-клієнт й інші інструменти.

Основні переваги PhpStorm:

- виправлення, підсвічування коду та його звертання;
- можливість розробляти тести PHPUnit в IDE й одразу запускати їх;
- універсальний інтерфейс системи контролю версій, для зручної роботи з проєктами які використовують Git, SVN, Mercurial і Perforce однаково зручно.

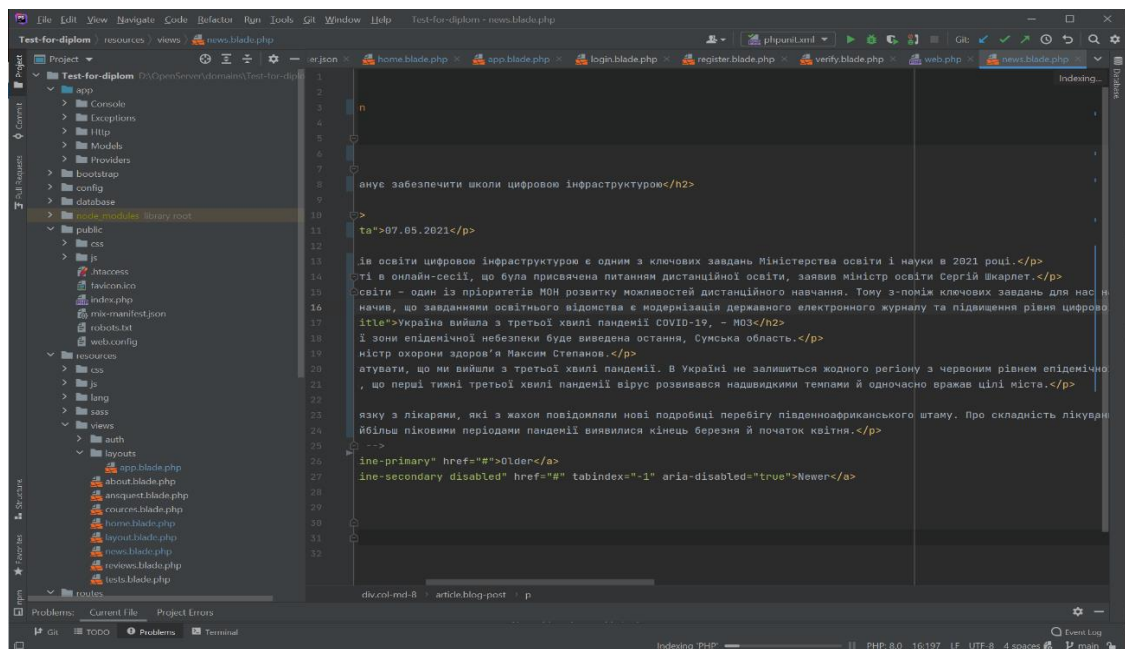


Рисунок 2.1 – Редактор коду PhpStorm

2.6 Діаграма взаємодії клієнта і сервера

В Laravel для взаємодії клієнтів та серверу використовуються запити(маршрути). Існують два види запитів, POST і GET запити, перший призначений для прийняття даних на сервер, а інші для отримання змісту з бажаного ресурсу. В залежності від запиту, після фільтрації сервером відбувається виклик функції контролера який знаходиться в файлі `app/Http/Controllers/{Ваш контролер}`. Контролер можна створити за допомогою команди `php artisan make:controller {ім'яController}`. Маршрути зазвичай визначають в файлі `routes.php` який знаходиться в папці `/app/Http/routes.php`.

Для більшого розуміння процесу давайте розглянемо діаграмі активності (рис. 2.2). Для цього процесу необхідні три деталі – веббраузер, який буде виконувати роль клієнта, вебпроект створених на базі фреймворку Laravel, розташований на сервері, і база даних.

Клієнт коли звертається до сервера робить запит, який сервер обробляє фільтрами. У фреймворка Laravel є власні базові фільтри розташовані у `/app/Http/Middleware`. Можна створювати й власні фільтри за допомогою команди `php artisan make: middleware {ім'я фільтру}`.

В наведеній нижче діаграмі зрозуміло показано процес проходження авторизації користувача в системі для написання відгуку.

В Laravel для того, аби зробити систему авторизації досить лише написати в консоль команди `php artisan make:auth` і `php artisan migrate`, і вже можна отримати заготовку для вашої системи авторизації, файл конфігурації буде розташований в `/config/auth.php`. Цей файл вже має декілька гарно описаних опцій для тонкого налаштування поведінки служб авторизації.

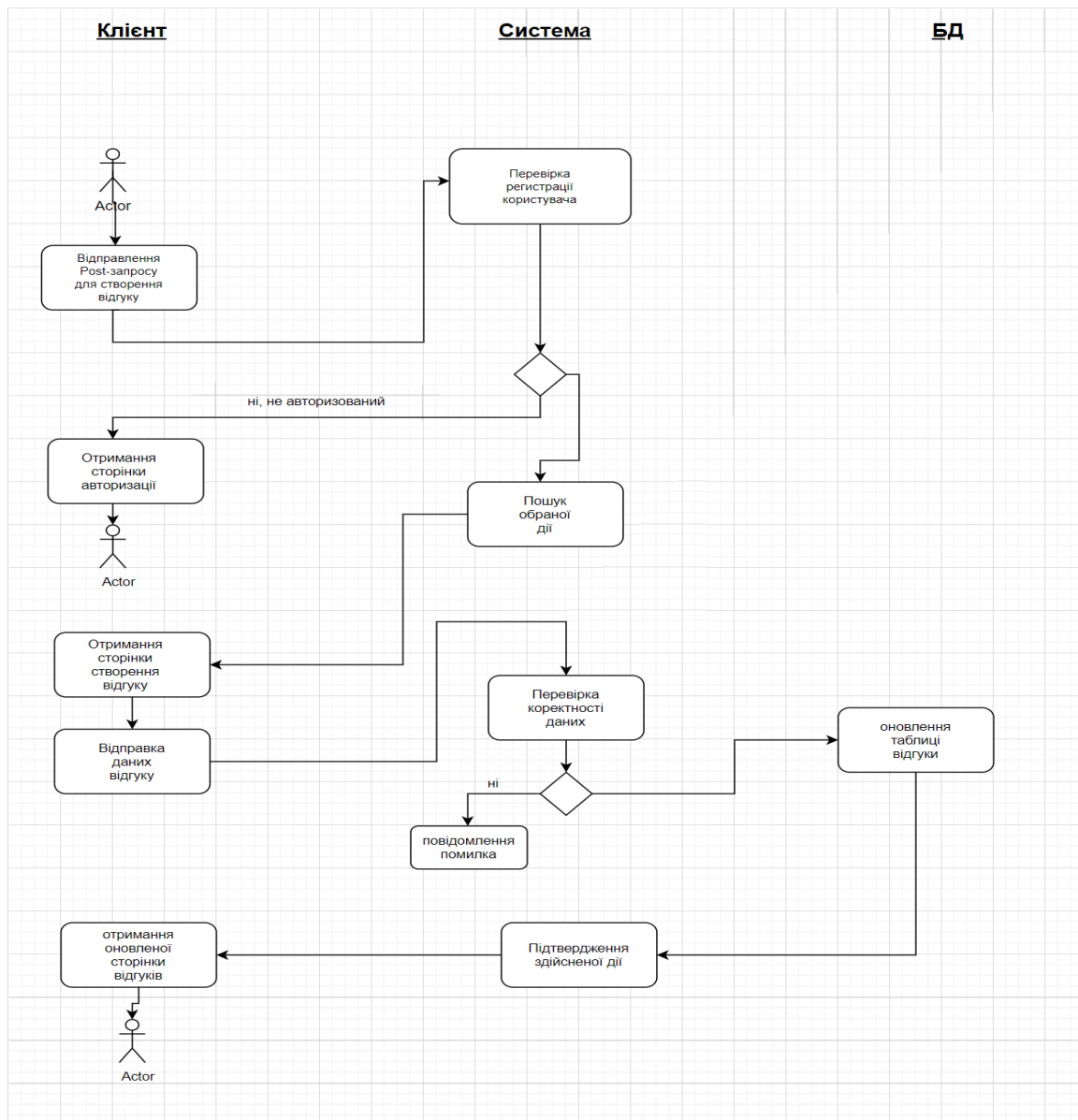


Рисунок 2.2 – Діаграма активності взаємодії клієнта й сервера

Для перевірки авторизації користувача існують фільтри, для створення і використання яких в Laravel є така багатofункціональна система – ORM Eloquent або система об’єктно-реляційного відображення. При використанні Eloquent кожна БД має відповідну модель, яка використовується для взаємодії з таблицею. Окрім отримання даних з таблиці БД, моделі Eloquent також дозволяють вставляти, оновлювати й видаляти дані з таблиць.

Після обробки фільтром запиту сервер або відправить користувача на сторінку авторизації, якщо він не авторизований, або запить пройде далі. Після

цього, на сервері пройде пошук дії обраної користувачем, він отримає сторінку створення свого відгуку, й після написання відправить його на сервер, де він має пройти перевірку коректності даних. В разі коли є помилка в даних, він отримає повідомлення про помилку, якщо все правильно сервер пропускає дані до бази даних, після цього на сервері пройде підтвердження здійсненої дії й користувач побачить оновлену сторінку відгуків.

2.7 Діаграми дій користувача у системі

Розглянемо активність прецедента «Створення відгуку» (рис. 2.3).

Основні дії:

- користувач переходить на сторінку авторизації;
- користувач водить логін та пароль, натискає кнопку «авторизація».

Система перевіряє чи всі поля заповнені, якщо ні то альтернативний потік A1, якщо так, то система перевіряє чи є користувач з такими даними в системі, якщо ні – альтернативний потік A2, якщо є, то авторизація проходить успішно й користувач входить в систему;

- користувач переходить на сторінку відгуків;
- користувач вводить данні у поля електронної пошти, теми, та повідомлення й натискає кнопку відправити. Система перевіряє чи всі поля заповнені коректно, якщо ні – альтернативний потік A3, якщо так, то відбувається створення відгуку, сторінка з відгуками оновлюється і користувач бачить свій відгук.

Альтернативний потік.

A1. Система видає повідомлення «Помилка. Не всі поля заповнені.». Відбувається перехід до кроку 1 основного потоку.

А2. Система видає повідомлення «Помилка. Користувача з такими даними не знайдено. Перевірте правильність даних.». Відбувається перехід до кроку 1 основного потоку.

А3. Система видає повідомлення «Помилка. {поле} Заповнено не коректно.» відбувається перехід до кроку 4 основного потоку.

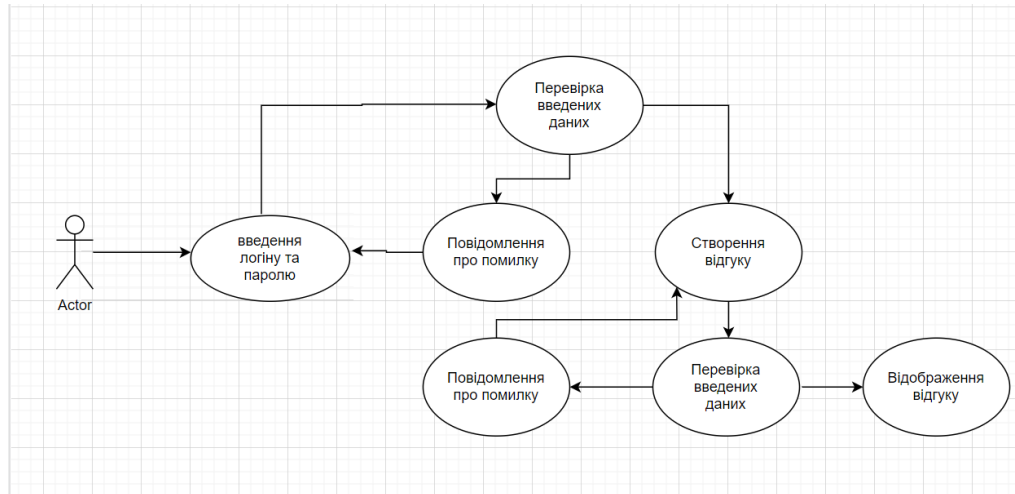


Рисунок 2.3 – Діаграма прецедента «Створення відгуку»

2.8 Розробка дизайну сайту

Проектування загального вигляду сайту містить у собі загальне уявлення інтерфейсу, характерні риси які б відображали його функціонал. Шаблон по якому буде проводитися розробка має не містити у собі помилки, бо по ньому буде проводитися розробка, й наповнення сайту.

Кожен, хто задіяний у проєкті має мати уявлення про свою частину праці, вайрфрейм чітко підходить для початку.

Вайрфрейм – макет в чорно-білому вигляді з упором на структуру й наповнення, а не візуальну частину.

Зовнішній вигляд такого макету має бути естетичним, вміщувати в себе головні елементи інтерфейсу й його приблизну візуалізацію.

Приклад розроблення вайрфрейм для головної сторінки інтернет ресурсу наведено на (рис. 2.4) та сторінки для залишення відгуку на (рис. 2.5).

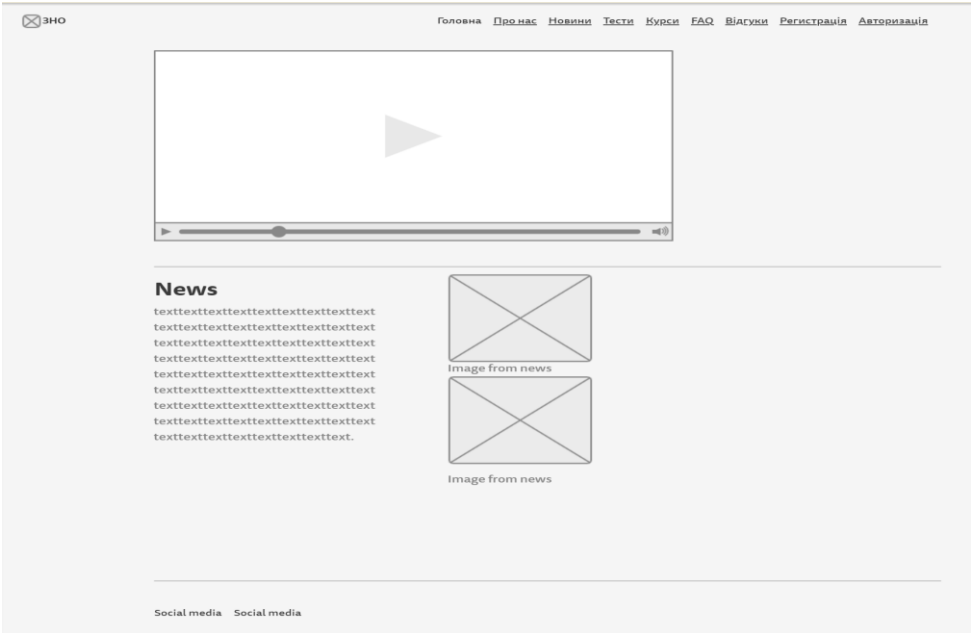


Рисунок 2.4 – Вайрфрейм для головної сторінки інтернет ресурсу

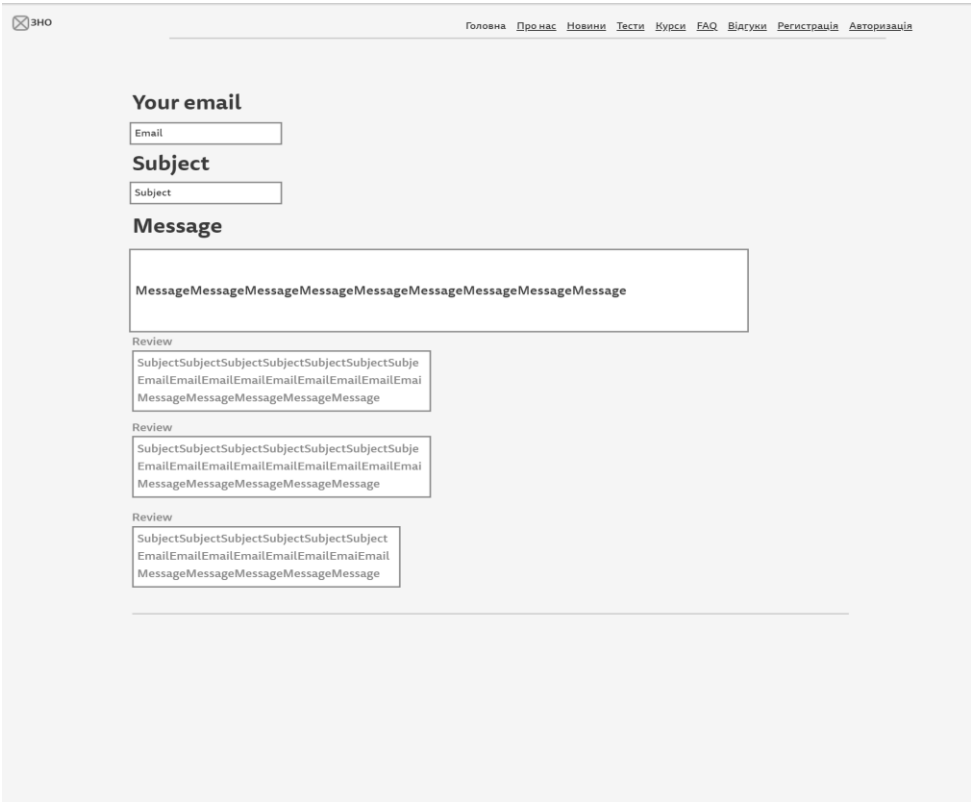


Рисунок 2.5 – Вайрфрейм для сторінки відгуків

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Опис структури файлів проєкту

Структура проєкту зображена на (рис. 3.1).

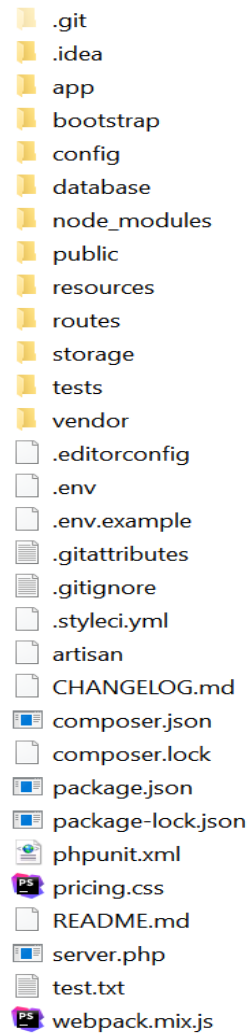


Рисунок 3.1 – Структура проєкту

Структура проєкту складається з:

- `.env` – файл, який містить інформацію необхідну для підключення бази даних до хостингу;
- `artisan` – утиліта яка використовується для виконання різних службових задач сервера;
- `app` – папка, яка містить у собі контролери `app/Controllers` та фільтри `app/Middleware`;
- `app.blade` – файл який містить основний код програми;
- `Bootstrap` – каталог який містить у собі `app.php`, файл який запускає фреймворк. В каталозі також знаходиться каталог кеша, який містить сгенеровані файли для оптимізації продуктивності(кеш маршрутів й служб);
- `Config` - папка яка містить файли конфігурації вебзастосунку;
- `Database` – папка яка містить у собі базу даних й файли міграцій;
- `Public` – папка яка містить файл `index.php` котрий є вхідною точкою усіх запитів, які поступають до сайту. А також папка містить у собі ваші зображення, стилі для яких відведена папка `css` та JavaScript в папці `js`;
- `Httpaccess` – файл конфігурацій вебсервера Apache, котрий дозволяє управляти роботою сервера й настройками сайту;
- `Web.config` – файл який визначає поведінку додатків ASP.NET. Microsoft .NET Framework;
- `Resources` – папка в якій містяться `css` і `js` компоненти, вихідні ресурси в папці `Sass`, та папка `views` в якій вкладені файли шаблони HTML;
- `Routes` – папка яка містить у собі всі маршрути з файлом налаштування маршрутизації `web.php` для кожного посилання;
- `Storage` – папка яка містить скомпільовані blade-шаблони, файли сесії, кеш файлів, й інші файли створювані фреймворком;

- Tests – папка яка містить автотести, з початку має в собі готовий приклад PHPUnit;
- Vendor – містить composer залежності.

3.2 Створення бази даних

У більшості сайтів присутня база даних, при праці з якою ми будемо створювати нові поля, таблиці, змінювати їх типи. Для роботи з базою даних необхідні спеціальні програми по типу phpMyAdmin, phpAdminer.

Контроль версій баз даних може здійснюватися за допомогою міграцій. Вони дозволяють вам змінювати структуру бази даних й дозволяють іншим учасникам бачити зміни. Для більш приємного користування краще використовувати конструктор таблиць як наприклад Migrations і Schema разом з міграціями. Цей конструктор є незалежним від БД інтерфейсом маніпулювання таблицями, який горно працює з усіма СУБД.

Міграції у Laravel розташовані як файли класів міграцій, в папці database/migration.

Основні плюси міграцій:

- управління версіями БД, відкат до попередньої за необхідності;
- можливість перейти з одної програми для роботи з БД в інші завдяки використанню сиду БД;
- підтримуються більшістю фреймворків.

Для налаштування бази даних потрібно змінити файл .env наступний чином:

```
DB_HOST=localhost або 127.0.0.1
DB_DATABASE=your_database_name
DB_USERNAME=database_username
DB_PASSWORD=database_password
```

Далі робимо міграцію за допомогою команди: `php artisan make:migration news`, після чого у файлі проєкту `/app` було створено модель `news.php`, а у `database/migration` створено файл `news_table.php`. До наших міграцій там вже було два файли. Перший файл (`2014_10_12_000000_create_users_table.php`) котрий генерує таблицю користувачів в БД. Другий файл (`2014_10_12_100000_create_password_resets_table.php`) це таблиця необхідна для відновлення паролю.

Клас міграції містить два методи: `up` і `down`. `Up` необхідний для додавання нових таблиць, стовпців або індексів в БД. `Down` просто відмінняє операції виконані в методі `up`.

Метод `schema::create` слугує для створення таблиці. Приймає два аргументи, перший - це ім'я таблиці, другий – замикання, яке в якості аргументу приймає об'єкт `blueprint`, який використовується для визначення нової таблиці.

На (рис. 3.2) зображено створення таблиці для збереження нових публікацій у базі даних.

```
public function up()
{
    Schema::create( table: 'news', function (Blueprint $table) {
        $table->id();
        $table->text( column: 'title');
        $table->longText( column: 'body');
        $table->timestamp( column: 'created_at')->useCurrent();
        $table->timestamp( column: 'updated_at')->default(DB::raw( 'value: 'CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP')));
    });
}

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

Рисунок 3.2 – Створення таблиці для створення нових постів новин у базі даних

База даних інтернет ресурсу складається з основних таблиць, перелік яких наведено в (табл. 3.1-3.2).

Таблиця 3.1 – структура таблиці Contacts

Назва поля	Призначення	Тип даних
id	Номер в базі даних	bigint unsigned
created_at	Дата створення відгуку	timestamp
updated_at	Дата оновлення відгуку	timestamp
email	Електронна пошта користувача	varchar
subject	Тема відгуку	varchar
message	Відгук	text

Таблиця 3.2 – Структура таблицы News

Назва поля	Призначення	Тип даних
Id	Номер в базі даних	bigint unsigned
Title	Тема новин	text
Body	Основний текст новин	longtext
created_at	Дата публікації статі	timestamp
updated_at	Дата Оновлення статі	timestamp

Та ще одної таблиці (табл. 3.3).

Таблиця 3.3 – Структура таблиці Sources

Назва поля	Призначення	Тип даних
id	Номер в базі даних	bigint unsigned
title	Назва курсу	text
description	Опис курсу	longtext
link	Посилання на курс	text
created_at	Дата створення публікації курсу	timestamp
updated_at	Дата створення публікації курсу	timestamp

Для зберігання даних між Oracle і MySQL було обрано останню у зв'язку з знайомством з нею.

3.3 Створення логіки обробки HTTP-запитів

Запрос починається з того, що користувач вводить в строку запросу url, на якій він хоче перейти.

Цей запрос потрапляє до файлу public/index.php котрий є конфігурацією вебсерверу. Далі цей файл загрузає автозавантажувач, створений менеджером пакетів Composer, а потім бере екземпляр додатку Laravel із bootstrap/app.php .

Першим що зробить Laravel буде створювання екземпляру додатку, контроллеру. Потім запрос потрапляє до HTTP-ядра, або ядру консолі залежно

від типу запиту. Саме ядро HTTP розширяє клас Illuminate/Foundation/Http/Kernel, котрий визначає масив завантажувачів(bootstrappers), котрі запусаються до виконання запросу.

Ці завантажувачі налаштовують обробітку помилок, ведення журналу, визначають середу розробки додатку і виконують інші задачі, котрі необхідно виконати до фактичного виконання запиту. Також у ядрі буде визначено список HTTP фільтрів.

Логіка запитів в даному проекті організована через контролери.

Створюємо контролер MainController.php . Цей контролер має клас MainController який наслідує Controller.php вбудований клас Laravel.

Клас MainController має функції:

- функція home();
- функція about();
- функція news();
- функція tests();
- функція reviews();
- функція reviews_check(Request \$request);
- функція ansquest();
- функція sources();
- функція home().

Для виведення новин та курсів використовуються методи, які описані нижче:

```
$news = News::query()
->orderBy('created_at','DESC')
->limit(2)
->get();
$sources = Course::query()
->orderBy('created_at','DESC')
```

->limit(2)

->get();

Метод orderBy – це сортування публікацій за датою створення від нової да останньої.

Метод limit – це ліміт на кількість елементів на сторінці, в даному випадку лише 2 курси і 2 новини.

Метод get – отримує вже відфільтровані дані після двох попередніх методів які були свого родами фільтрами.

Ця функція виводить шаблон home.blade.php, який передається змінні news і course.

Функція about призначена для виведення шаблону about.blade.php.

Функція news – призначена для виведення шаблону news.blade.php.

Функція tests – призначена для виведення шаблону tests.blade.php.

Функція reviews – призначена для виведення шаблону reviews.blade.php.

Функція reviews_check – призначена для перевірки правильності введених даних при залишенні відгуку на сайті, через валідацію, а також збереження даних у БД, та пересилання назад до reviews після вдалого залишення відгуку. Для того аби зберегти інформацію о відгуку та залишити її на сторінці відгуків необхідно доступ до полей таблиці Contacts, які заповнюють дані відгуків:

- поле \$review = new Contact(); - створення нового відгуку користувача в БД;

- поле \$review->email = \$request->input('email'); - електронна пошта користувача готується для внесення в БД;

- поле \$review->subject = \$request->input('subject'); - тема відгуку готується для внесення в БД;

- поле \$review->message = \$request->input('message'); - текст відгуку користувача готується для внесення в БД;

- поле `$review->save()`; - збереження інформації в БД.

Після збереження даних в базу даних, користувача переправляє на шаблон

`reviews.blade.php` та показує відгук на сторінці.

Функція `ansquest` – відповідає за виведення шаблону `ansquest.blade.php`.

Функція `sources` – відповідає за виведення шаблону `sources.blade.php`.

3.4 Опис шаблонів проєкту

Шаблонізатор Blade – простий, але потужний шаблонізатор, який входить до складу Laravel. Blade заснований на концепції успадкування шаблонів і секцій тому можуть використовуватися повторно. Він не обмежує в використанні `php` конструкцій. Все шаблони мають мати розширення `.blade.php`. Шаблони компілюються в `php`-код і кешуються фреймворком тому цей шаблонізатор не заторможує роботу фреймворка.

Розглянемо шаблони проєкту:

- `layouts/app` – шаблонний шаблон, бо всі його елементи відображені на всіх сторінках проєкту однаково (панель навігації по сайту);
- `home` – шаблон головної сторінки сайту, на якій є частково новини й курси для звернення уваги до них;
- `news` – шаблон який має більш розгорнутий вигляд новин у порівнянні з головною сторінкою, новини викладені від першої до останньої;
- `tests` – шаблон у якому розміщені тести до ЗНО за минулі роки, відповіді до них, система оцінювання завдань та балів, та розгорнута відповідь для деяких складних питань;
- `ansquest` – шаблон сторінки FAQ(найчастіші запитання) де користувач може побачити питання, які найчастіше задають нам інші користувачі;

- reviews – шаблон сторінки на який знаходяться відгуки авторизованих користувачів сайту, і форма через яку можна залишити свій відгук;
- sources – шаблон сторінки з коротким описом та посиланням на курси;
- login – шаблон який відповідає за вигляд сторінки авторизації користувача;
- register – шаблон котрий відповідає за вигляд сторінки реєстрації користувача.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення та умови застосування застосунку

Інтернет ресурс для перегляду інформації корисною для здачі ЗНО складається з клієнтської та серверної частини, виконаний у вигляді інформаційного порталу.

Вебзастосунок надає можливість користувачам побачити останні новини які відносяться до ЗНО, пройти курси підготовки до ЗНО, загрузити тести з різних предметів ЗНО та перевірити свої знання, поділитися своїм відгуком про сайт з іншими.

Для використання сервісу, його необхідно завантажити на сервер, або запустити через віртуальних сервер, який має будь яку архітектуру й сервер має відповідати таким вимогам:

- PHP >= 5.4;
- Mcrypt PHP Extension;
- OpenSSL PHP Extension;
- Mbstring PHP Extension.

Для мінімальних системних вимог до сервера було обрано мінімальну систему з сайту американського провайдеру віртуальних приватних серверів DigitalOcean:

- оперативна пам'ять 4 GB;
- 1vCPUs;
- 5 GB пам'яті жорсткого диску SSD.

4.2 Завантаження сайту на хостинг

Хостинг – це платна послуга з розміщення інформації на сервері, що підключений до інтернету. В неї входить оренда місця на сервері та його використання його потужності.

Свої проєкти можна розміщувати на хостингу двома способами.

Перший це за допомогою хостинг-провайдерів. Ваш проєкт розміщується на виділеному місці на сервері з передвстановленою мовою програмування та оперативною системою Linux за замовчуванням. А вам як адміністратору надається доступ до адмін-панелі вашого проєкту на хостингу, для внесення змін та коригування свого проєкту.

Другим варіантом є хостинг послуга VDS або VPS, де користувачу надається віртуальний сервер. Цей сервер емулює роботу реального фізичного сервера – є root-доступ, можлива установка власної операційної системи і програмного забезпечення.

Порівнюємо віртуальний хостинг з фізичним.

Плюси віртуального серверу:

- невелика вартість – оренда віртуального серверу набагато дешевша аналогу фізичного серверу;
- просте масштабування – віртуальному серверу дуже просто додати потужності у вигляді додаткового об'єму оперативної пам'яті, процесорних ядер, або жорсткого диску.

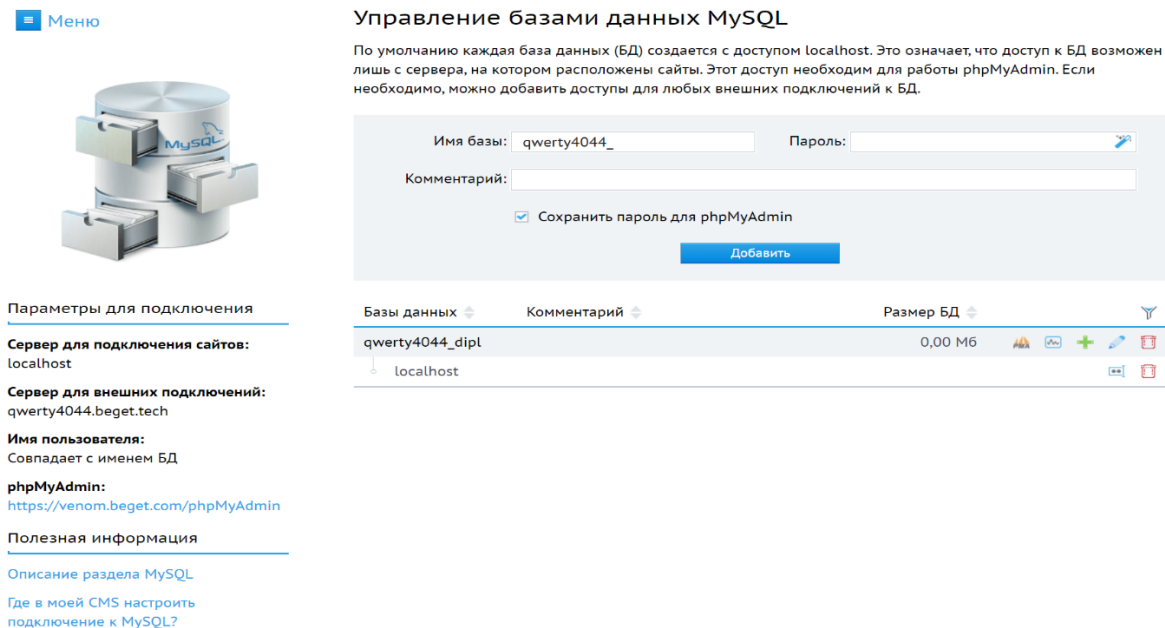
Мінуси віртуального сервера у порівнянні з фізичним:

- усі віртуальні сервера працюють на єдиній хост-машині, у умовах розподілення ресурсів, використовують одну на всіх дискову систему і загальний високошвидкісних каналах зв'язку, що деякі користувачі можуть зловживати використанням ресурсів і заважають іншим клієнтам.

Потрібно створити базу даних та нового користувача, та зберегти дані для входу в неї. Ці дані редагуються в файлі `.env`. Дані необхідні для з'єднання з застосунків з базою даних прописуються в строках `DB_CONNECTION` та `DB_PORT` (рис. 4.1) та імпортуємо БД через phpMyAdmin (рис. 4.2).

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=zno_reviews
DB_USERNAME=root
DB_PASSWORD=root
```

Рисунок 4.1 – Редагування строк у файлі `.env`



The screenshot shows the phpMyAdmin interface. On the left, there is a sidebar with a 'Menu' button and a MySQL logo. The main area is titled 'Управление базами данных MySQL'. Below the title, there is a form for adding a new database. The form includes fields for 'Имя базы:' (qwerty4044_), 'Пароль:' (empty), and 'Комментарий:' (empty). There is a checkbox labeled 'Сохранить пароль для phpMyAdmin' which is checked. A 'Добавить' button is at the bottom of the form. Below the form, there is a table listing existing databases. The table has columns for 'Базы данных', 'Комментарий', and 'Размер БД'. The table contains two rows: 'qwerty4044_dipl' with a size of '0,00 МБ' and 'localhost'.

Параметры для подключения

Сервер для подключения сайтов:
localhost

Сервер для внешних подключений:
qwerty4044.beget.tech

Имя пользователя:
Совпадает с именем БД

phpMyAdmin:
<https://venom.beget.com/phpMyAdmin>

Полезная информация

[Описание раздела MySQL](#)

[Где в моей CMS настроить подключение к MySQL?](#)

Управление базами данных MySQL

По умолчанию каждая база данных (БД) создается с доступом localhost. Это означает, что доступ к БД возможен лишь с сервера, на котором расположены сайты. Этот доступ необходим для работы phpMyAdmin. Если необходимо, можно добавить доступы для любых внешних подключений к БД.

Имя базы: qwerty4044_ Пароль:

Комментарий:

☒ Сохранить пароль для phpMyAdmin

Добавить

Базы данных	Комментарий	Размер БД
qwerty4044_dipl		0,00 МБ
localhost		

Рисунок 4.2 – Імпорт бази даних до хостингу

У панелі phpMyAdmin через «Імпорт» імпортуємо нашу базу даних для проєкту (рис. 4.3) та завантажуюмо файли проєкту на хостинг (рис. 4.4).

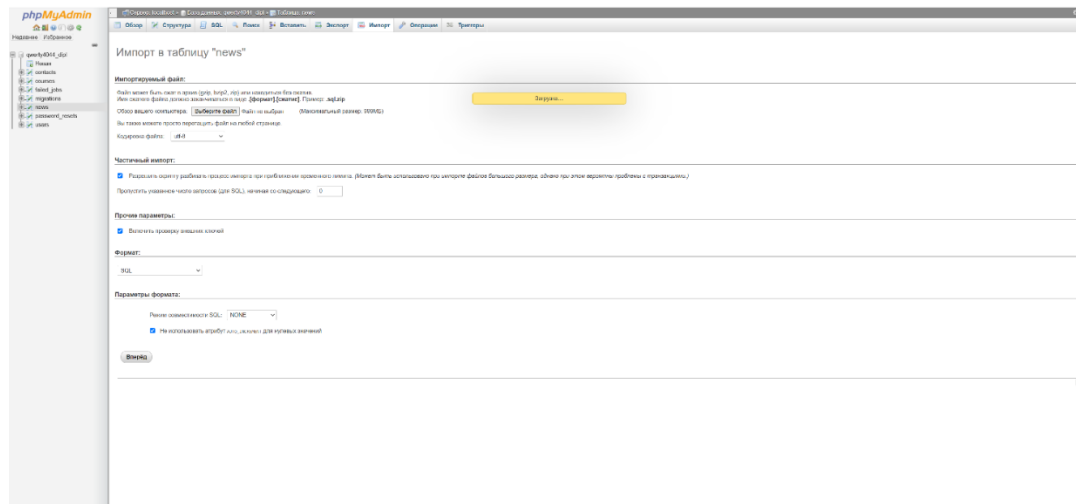


Рисунок 4.3 – Панель phpMyAdmin, розділ імпорт

Локальный сервер Удалённый сервер Список Сайтов Обновить Новая папка Загрузить Файлы Создать Архив Скачать Архив Поиск Файлов Поиск Текста Анализ Размера					
Локальный сервер .../qwerty4044.beget.tech/public_html					
Имя	Тип	Размер	Владелец	Атри...	Изменён
..	[DIR]				
.git	[DIR]		qwerty4044	700	13.05.2021 00:10...
.idea	[DIR]		qwerty4044	700	13.05.2021 00:10...
app	[DIR]		qwerty4044	700	13.05.2021 00:10...
bootstrap	[DIR]		qwerty4044	700	13.05.2021 00:10...
cgi-bin	[DIR]		qwerty4044	700	12.05.2021 23:47...
config	[DIR]		qwerty4044	700	13.05.2021 00:10...
database	[DIR]		qwerty4044	700	13.05.2021 00:10...
node_modules	[DIR]		qwerty4044	700	13.05.2021 00:10...
public	[DIR]		qwerty4044	700	13.05.2021 00:10...
resources	[DIR]		qwerty4044	700	13.05.2021 00:10...
routes	[DIR]		qwerty4044	700	13.05.2021 00:10...
storage	[DIR]		qwerty4044	700	13.05.2021 00:10...
tests	[DIR]		qwerty4044	700	13.05.2021 00:10...
vendor	[DIR]		qwerty4044	700	13.05.2021 00:10...
.editorconfig		235 ...	qwerty4044	600	05.05.2021 15:29...
.env		882 ...	qwerty4044	600	12.05.2021 22:05...
.env	exa...	860 ...	qwerty4044	600	05.05.2021 15:29...
.gitattributes		116 ...	qwerty4044	600	05.05.2021 15:29...
.gitignore		204 ...	qwerty4044	600	05.05.2021 15:29...
.styleci		194 ...	qwerty4044	600	05.05.2021 15:29...
CHANGELOG	md	9.7 KB	qwerty4044	600	05.05.2021 15:29...
README	md	3.8 KB	qwerty4044	600	05.05.2021 20:38...
Test-for-diplom	rar	788...	qwerty4044	700	12.05.2021 23:59...
artisan		1.7 KB	qwerty4044	600	05.05.2021 15:29...
composer	json	1.6 KB	qwerty4044	600	08.05.2021 13:14...
composer	lock	263...	qwerty4044	600	08.05.2021 13:13...
package-lock	json	1.2 MB	qwerty4044	600	08.05.2021 13:17...
package	json	767 ...	qwerty4044	600	08.05.2021 13:17...
phpunit	xml	1.2 KB	qwerty4044	600	05.05.2021 15:29...
pricing	css	155 ...	qwerty4044	600	23.03.2021 19:20...
server	php	584 ...	qwerty4044	600	05.05.2021 15:29...
test	txt	0 byt...	qwerty4044	600	05.05.2021 21:03...
webpack.mix	js	565 ...	qwerty4044	600	08.05.2021 13:15...

Рисунок 4.4 – Завантажений сайт на хостингу

Після всіх дій перевіряємо роботу сайту. На рисунку 4.5 зображено результати запуску інтернет ресурсу на хостингу.

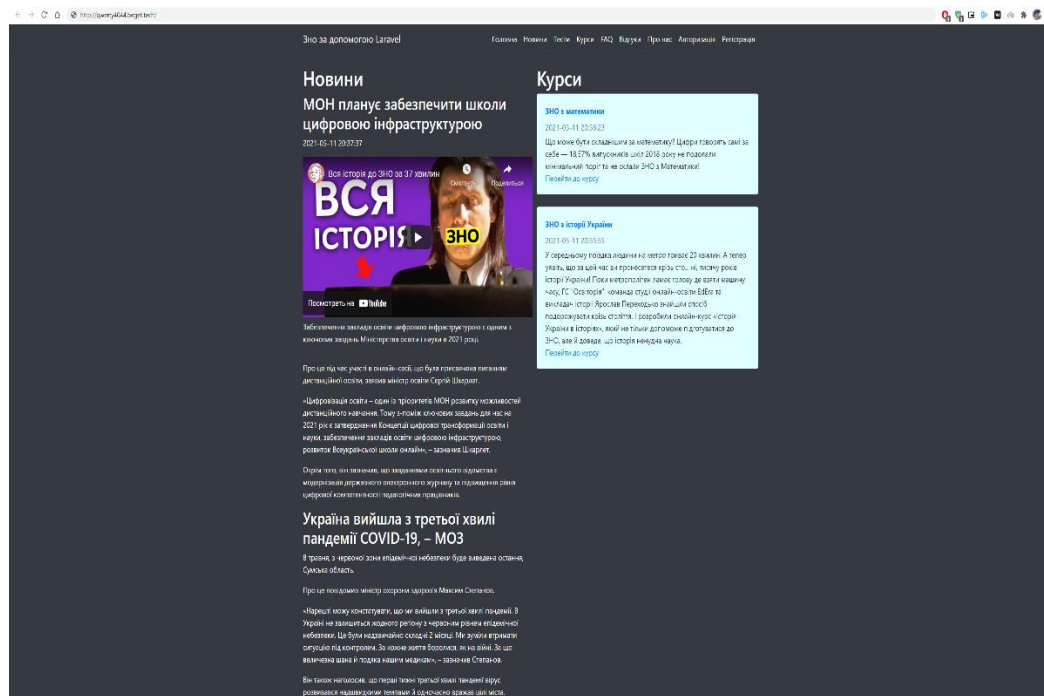


Рисунок 4.5 – Результат запуску інтернет-сервісу на хостингу

4.3 Робота із застосунком

Для роботи із застосунком потрібно мати пристрій, який має підтримку інтернету, браузер та мінімальні характеристики свого браузера.

Щоб почати роботу із сайтом, потрібно в адресному рядку ввести url сайту з хостингу, після чого користувач потрапляє на головну сторінку (рис 4.6).

На головній сторінці користувач може прочитати останні новини та курси які запропоновано на даному ресурсі. Для переходу на сторінку, що цікавить, користувача можна обрати її на панелі навігації.



Рисунок 4.6 – Головна сторінка сайту

Щоб залишити відгук про сайт, потрібно авторизуватися. Для цього необхідно перейти на сторінку авторизації або зареєструватися якщо це новий користувач. Для цього в шапці сайту є кнопки «Авторизація» та «Регістрація», що переадресовують на ці сторінки (рис. 4.7) або (рис. 4.8).

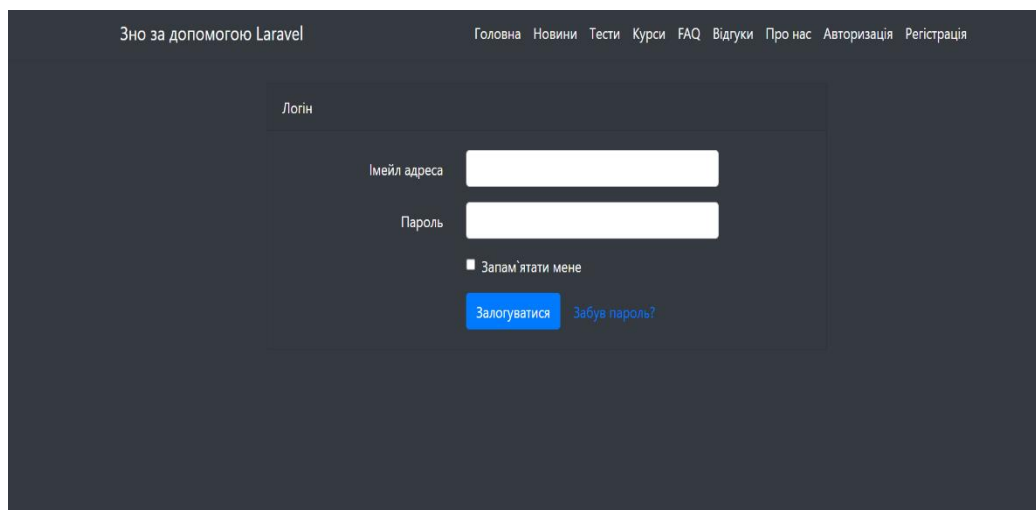


Рисунок 4.7 – Сторінка авторизація

Рисунок 4.8 – Сторінка реєстрація

Далі для створення відгуку користувач переходить на вкладку відгуків, й заповнює форми для електронної пошти, теми та самого тексту відгуку й натискає «відправити» (рис. 4.9).

Рисунок 4.9 – Сторінка зворотного зв'язку

Якщо користувача цікавлять тести по ЗНО, то він переходить на вкладку **ТЕСТИ** й може скачати матеріал, що зацікавив (рис. 4.10).

Якщо користувач зацікавлений у курсах, то він переходить на сторінку **КУРСИ** (рис. 4.11).

Якщо у користувача з'явилися якісь питання він може перейти у вкладку **FAQ**, або **Про нас** для отримання нашої контактної інформації, де знаходяться й робочі години нашої гарячої лінії (рис. 4.12) та (рис. 4.13).

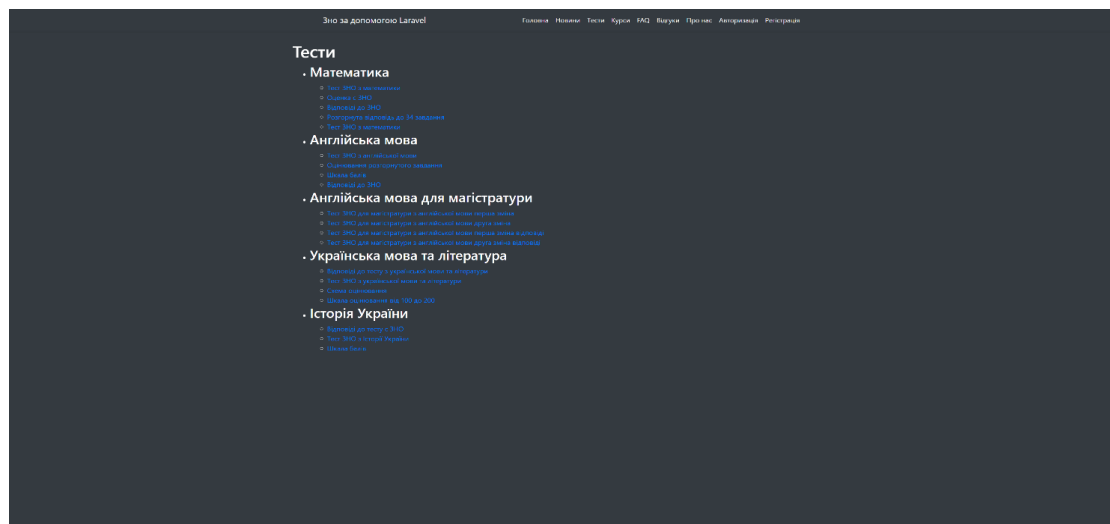


Рисунок 4.10 – Сторінка тести

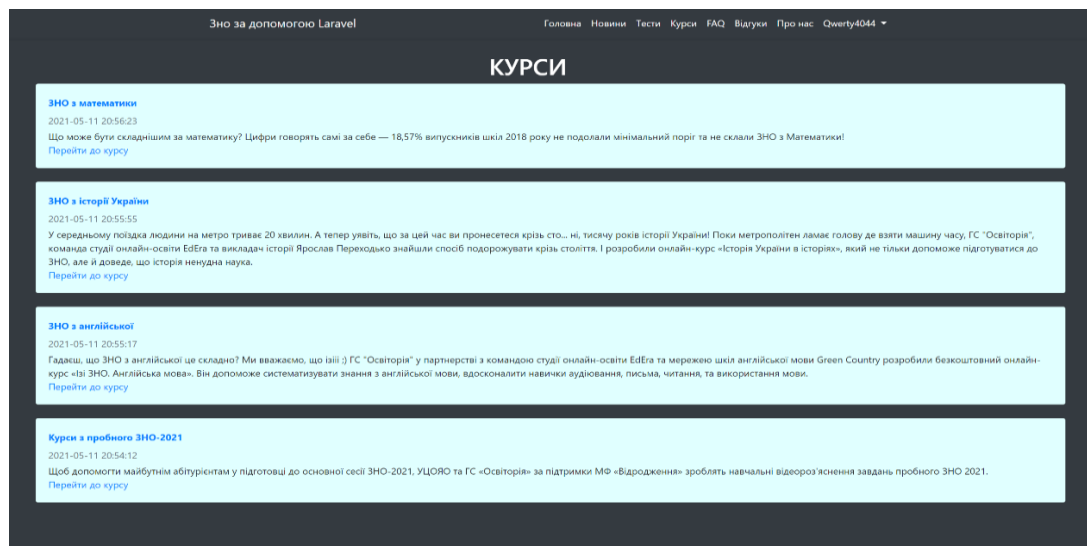


Рисунок 4.11 – Сторінка курси

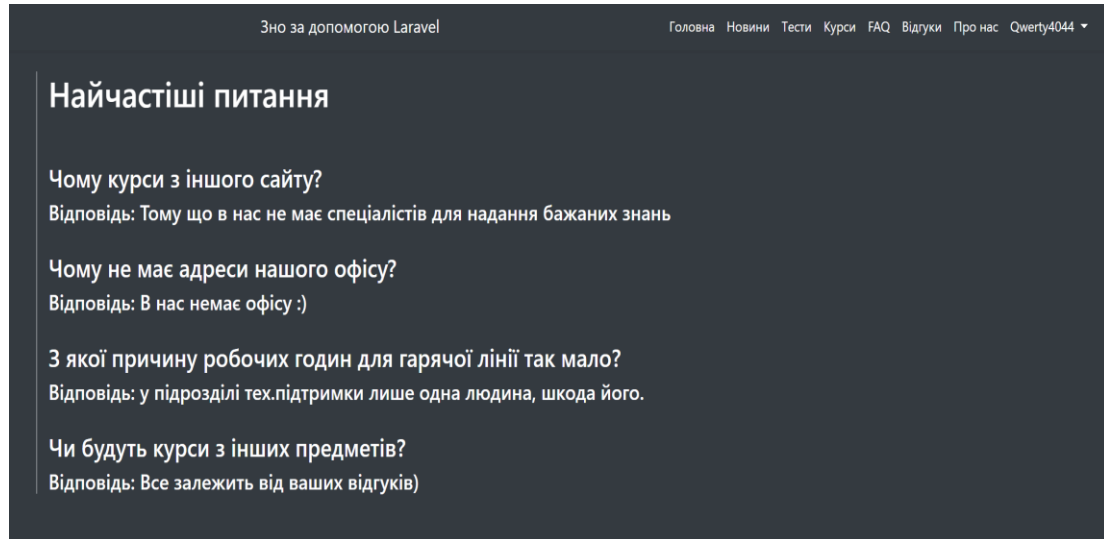


Рисунок 4.12 – Сторінка FAQ

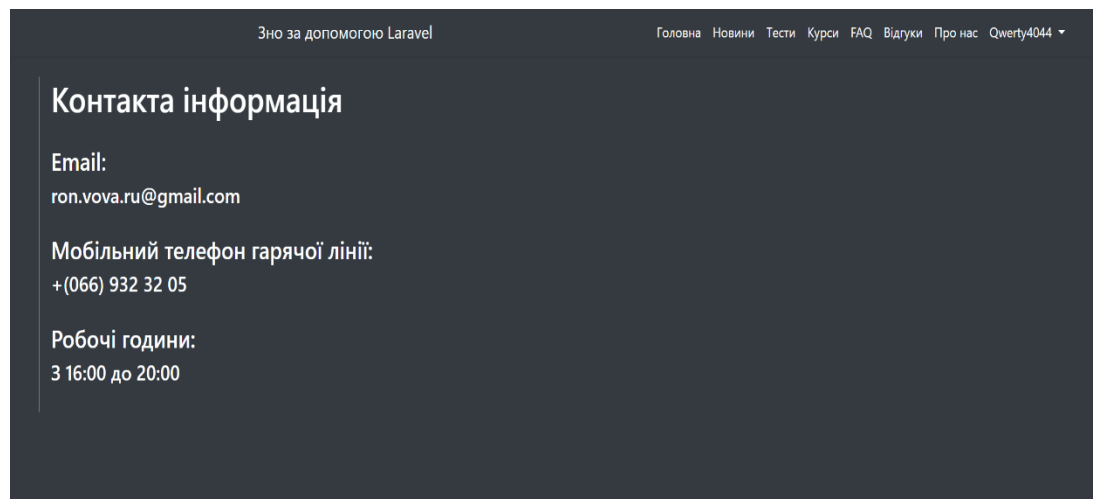


Рисунок 4.13 – Сторінка про нас

4.4 Можливі модифікації

Для покращення роботи з застосунком й прибирання його вразливостей необхідно:

- захистити код від модифікацій;
- додати онлайн варіант тестів;

– додати форум для можливості користувачів спілкуватися один з одним.

4.5 Вхідні й вихідні дані

Вхідними даними є HTTP та HTTPS запити, які користувачі відправляють через браузер на сервер.

Вихідні дані є згенеровані HTML сторінки та PDF файли з курсами по запиту користувачів.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи було вдало вирішено поставлену задачу. Було проведено аналіз предметної області та програмних продуктів для створення продукту який би відповідав певним стандартам. Після було проведено дослідження для вирішення мінімального інструментарію ресурсу й функціональним вимогам.

Після було проведено порівняння CMS систем та фреймворків, які були необхідні для вирішення цієї задачі й визначено Laravel як підходяще вирішення для розробки інтернет порталу.

Далі після ознайомлення з фреймворком й створенням бази сайту, було проведено розробку дизайну, проведено аналіз взаємодії користувача з майбутнім інтерфейсом, після чого було розпочато написання основного коду.

Далі було проведено тестування продукту й закінчення кінцевої версії.

Як результат біло створено інтернет ресурс для підготовки до ЗНО завдяки якому користувач може:

- дивитися останні новини й курси для ЗНО;
- скачувати тести з предмета який його цікавить у ЗНО;
- залишати відгук та отримувати підтримку по гарячій лінії;
- користуватися сайтом з будь якого пристрою з браузером бо він є

адаптивним.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. HTML hypertext Markup Language – [Електрон. ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/html>.
2. CSS что это [Електрон. ресурс]. – Режим доступа: <https://wiki.rookee.ru/css/>.
3. Що таке Java Script як функціонує Java Script – [Електрон. ресурс]. – Режим доступу: <http://ruszura.in.ua/uncategorized/scho-take-javascript-yak-funktsionuyu-javascript.html>
4. Java Script basics [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics?retiredLocale=uk
5. Что такое PHP [Електрон. ресурс]. – Режим доступа: <https://www.php.net/manual/ru/intro-what-is.php>
6. Динамічні вебсайти [Електрон. ресурс]. – Режим доступу: <https://dystosvita.gnomio.com/mod/page/view.php?id=2716>
7. Системи керування вмістом [Електрон. ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BC%D1%96%D1%81%D1%82%D0%BE%D0%BC

ДОДАТОК А
Технічне завдання

Вступ

Метою проєкту є програмна реалізація вебсайта для підготовки до ЗНО користувачами.

А.1 Підстави для розробки

Підставою до розробки є завдання на дипломну кваліфікаційну роботу на тему “програмна реалізація вебсайту для підготовки до ЗНО”, затверджене наказом № 103 від 30 березня 2021 року по Національному університету «Запорізька політехніка».

А.2 Призначення розробки

Програмний продукт призначений для надання користувачам можливість підготовлюватися до ЗНО в мережі інтернет.

А.3.1 Вимоги до функціональних характеристик

Вебзастосунок повинен мати наступні функціональні можливості:

- реєстрація та авторизація користувачів;
- можливість читати останні новини про ЗНО;
- можливість переходити до цікавлячих їх курсів по ЗНО;
- скачувати тести по необхідному для них ЗНО;
- можливість залишати відгук на сайті для його покращення;
- інформація про сайт та часті питання.

Інтерфейс повинен задовольняти такі вимоги:

- простота та зручність;
- просту й приємну для очей ввечері\вночі кольорову палітру;
- швидкий доступ до всіх функцій інтерфейсу.

A.3.2 Вимоги до надійності

Програмне забезпечення повинно забезпечувати оброблення наступних ситуацій:

- не вдалося авторизуватися, або забув пароль;
- задано некоректні дані під час написання відгуку.

A.3.3 Умови експлуатації

Для забезпечення експлуатації програмного забезпечення необхідно встановити наступні програмні бібліотеки:

- хостинг з передумовленою мовою PHP;
- база даних MySQL.

A.3.4 Вимоги до складу та параметрів технічних засобів

Доступ до вебзастосунку може бути отриманий з будь-якого засобу (персональних комп'ютер або мобільний пристрій) з встановленим браузером та доступом до мережі інтернет.

A.3.5 Вимоги до маркування та упакування

Програма може бути записана на жорсткому чи оптичному диску або USB-носії. На упаковці має бути вказано назву роботи «Програмна реалізація вебсайту для підготовки до ЗНО».

A.3.6 Вимоги до транспортування і збереження

Вимоги до транспортування та зберігання повністю ідентичні вимогам, пропонованим до носія, на якому зберігається програма.

A.4 Стадії та етапи розробки

Для роботи над дипломною кваліфікаційною роботою бакалавра заплановано виконання наступних етапів:

- аналіз проблеми;
- проектування програмного забезпечення;
- розробка програмного забезпечення;
- тестування і відлагодження програмного забезпечення;
- оформлення документації.

A.5 Порядок контролю та приймання

Дипломна кваліфікаційна робота бакалавра має бути виконана у відповідності з календарним планом.

Дипломна кваліфікаційна робота бакалавра має бути прийнята консультантами відповідних розділів та керівником роботи загалом, після чого затверджена завідуючим кафедрою та захищена на засіданні екзаменаційної комісії.

ДОДАТОК Б
Текст програми

Б.1 Зміст файлу MainController.php

```
<?php
namespace App\Http\Controllers;
use App\Http\Controllers\Controller;
use App\Models>Contact;
use App\Models\Course;
use App\Models\News;
use Illuminate\Http\Request;
class MainController extends Controller
{
    public function home()
    {
        $news = News::query()
        ->orderBy('created_at', 'DESC')
        ->limit(2)
        ->get();
        $courses = Course::query()
        ->orderBy('created_at', 'DESC')
        ->limit(2)
        ->get();
        return view('home', compact(
        'news',
        'courses'
        ));
    }
    public function about()
    {
        return view('about');
    }
    public function news()
    {
        $news = News::query()
        ->orderBy('created_at', 'DESC')
        ->get();
        return view('news', compact('news'));
    }
    public function tests()
    {

```

```

return view('tests');
}
public function reviews()
{
$reviews = Contact::all();
return view('reviews', compact('reviews'));
}
public function reviews_check(Request $request)
{
$valid = $request->validate([
'email' => 'required|min:4|max:100',
'subject' => 'required|min:4|max:100',
'message' => 'required|min:15|max:500'
]);
$review = new Contact();
$review->email = $request->input('email');
$review->subject = $request->input('subject');
$review->message = $request->input('message');
$review->save();
return redirect('reviews');
}
public function ansquest()
{
return view('ansquest');
}
public function cources()
{
$courses = Course::query()
->orderBy('created_at', 'DESC')
->get();
return view('cources', compact(
'courses'
));
}
}
}

```

Б.2 Зміст файлу web.php

```

<?php
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\MainController;

```

```

Auth::routes();
Route::get('/', [MainController::class, 'home'])->name('home');
Route::get('/about', [MainController::class, 'about'])->name('about');
Route::get('/pdf', [MainController::class, 'tests'])->name('tests');
Route::get('/ansquest', [MainController::class, 'ansquest'])->name('ansquest');
Route::get('/cources', [MainController::class, 'cources'])->name('cources');
Route::get('/reviews', [MainController::class, 'reviews'])->name('reviews');
Route::post('/reviews/check', [MainController::class, 'reviews_check'])->
>name('reviews_check');
Route::get('/news', [MainController::class, 'news'])->name('news');

```

Б.3 Зміст файлу app.blade.php

```

<!doctype html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- CSRF Token -->
    <meta name="csrf-token" content="{{ csrf_token() }}">
    <title>{{ config('app.name', 'Laravel') }}</title>
    <!-- Scripts -->
    <script src="{{ asset('js/app.js') }}" defer></script>
    <!-- Fonts -->
    <link rel="dns-prefetch" href="//fonts.gstatic.com">
    <link
        href="https://fonts.googleapis.com/css?family=Nunito"
rel="stylesheet">
    <!-- Styles -->
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/css/bootstrap.min.css"
rel="stylesheet"
>
</head>
<body class="bg-dark">
<div id="app" class="bg-dark">
<nav class="navbar navbar-expand-md navbar-light bg-dark shadow-sm text-
white">
    <div class="container text-white">
    <a class="navbar-brand text-white" href="{{ url('/') }}">Зно за допомогою
    {{ config('app.name', 'Laravel') }}
    </a>

```

```

<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="{{ __('Toggle navigation') }}">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<!-- Left Side Of Navbar -->
<ul class="navbar-nav mr-auto">
</ul>
<!-- Right Side Of Navbar -->
<ul class="navbar-nav ml-auto">
<!-- Authentication Links -->
@if (Route::has('home'))
<li class="nav-item">
<a class="nav-link text-white" href="{{ route('home') }}">{{ __('Головна')
}}</a>
</li>
@endif
@if (Route::has('news'))
<li class="nav-item">
<a class="nav-link text-white" href="{{ route('news') }}">{{ __('Новини')
}}</a>
</li>
@endif
@if (Route::has('tests'))
<li class="nav-item">
<a class="nav-link text-white" href="{{ route('tests') }}">{{ __('Тести')
}}</a>
</li>
@endif
@if (Route::has('courses'))
<li class="nav-item">
<a class="nav-link text-white" href="{{ route('courses') }}">{{ __('Курси')
}}</a>
</li>
@endif
@if (Route::has('ansquest'))
<li class="nav-item">
<a class="nav-link text-white" href="{{ route('ansquest') }}">{{ __('FAQ')
}}</a>
</li>

```

```

    @endif
    @if (Route::has('reviews'))
    <li class="nav-item">
    <a class="nav-link text-white" href="{{ route('reviews') }}">{{ __('Відгуки')
}}</a>
    </li>
    @endif
    @if (Route::has('about'))
    <li class="nav-item">
    <a class="nav-link text-white" href="{{ route('about') }}">{{ __('Про нас')
}}</a>
    </li>
    @endif
    @guest
    @if (Route::has('login'))
    <li class="nav-item">
    <a class="nav-link text-white" href="{{ route('login') }}">{{
__('Авторизація') }}</a>
    </li>
    @endif
    @if (Route::has('register'))
    <li class="nav-item">
    <a class="nav-link text-white" href="{{ route('register') }}">{{
__('Регістрація') }}</a>
    </li>
    @endif
    @endguest
    @auth
    <li class="nav-item dropdown">
    <a id="navbarDropdown" class="nav-link dropdown-toggle text-white"
href="#" role="button"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
{{ Auth::user()->name }}
</a>
    <div class="dropdown-menu dropdown-menu-right bg-dark" aria-
labelledby="navbarDropdown">
    <a class="dropdown-item text-white" href="{{ route('logout') }}"
onclick="event.preventDefault();
document.getElementById('logout-form').submit();">
{{ __('Вийти') }}
    </a>

```

```

        <form id="logout-form" action="{{ route('logout') }}" method="POST"
class="d-none ">
        @csrf
    </form>
</div>
</li>
@endauth
</ul>
</div>
</div>
</nav>
<main class="py-4">
@yield('content')
</main>
</div>
</body>
</html>

```

Б.4 Зміст файлу **home.blade.php**

```

@extends('layouts.app')
@section('content')
<div class="container">
<div class="row justify-content-center">
<div class="col-md-8">
</div>
<div class="col-md-6">
<h1 class="text-white mx-auto">Новини</h1>
@if($news)
@foreach($news as $newsItem)
<article class="blog-post text-white">
<h2 class="heading">{{ $newsItem->title }}</h2>
<p class="blog-post-meta">{{ $newsItem->created_at }}</p>
<div>
{!! $newsItem->body !!}
</div>
</article>
@endforeach
@else
<p>Новости отсутствуют</p>
@endif

```

```

</div>
<div class="col-md-6">
<h1 class="text-white mx-auto">Курси</h1>
@if($courses)
@foreach($courses as $course)
<div>
<div class="card flex-md-row mb-4 box-shadow h-md-250
"style="background-color: rgba(224,255,255);">
<div class="card-body d-flex flex-column align-items-start">
<strong class="d-inline-block mb-2 text-primary">{{ $course->title
}}</strong>
<div class="mb-1 text-muted">{{ $course->created_at }}</div>
<p class="card-text mb-auto">{{ $course->description }}</p>
<a href="{{ $course->link }}">Перейти до курсу</a>
</div>
</div>
</div>
@endforeach
@else
<p>Новости отсутствуют</p>
@endif
</div>
</div>
</div>
@endsection

```

Б.5 Зміст файлу login.blade.php

```

@extends('layouts.app')
@section('content')
<div class="container">
<div class="row justify-content-center">
<div class="col-md-8">
<div class="card bg-dark text-white">
<div class="card-header">{{ __('Логін') }}</div>
<div class="card-body">
<form method="POST" action="{{ route('login') }}">
@csrf
<div class="form-group row">
<label for="email" class="col-md-4 col-form-label text-md-right">{{
__('Імейл адреса') }}</label>

```

```

<div class="col-md-6">
  <input id="email" type="email" class="form-control @error('email') is-invalid
  @enderror" name="email" value="{{ old('email') }}" required autocomplete="email"
  autofocus>
  @error('email')
  <span class="invalid-feedback" role="alert">
    <strong>{{ $message }}</strong>
  </span>
  @enderror
</div>
</div>
<div class="form-group row">
  <label for="password" class="col-md-4 col-form-label text-md-right">{{
  __('Пароль') }}</label>
  <div class="col-md-6">
    <input id="password" type="password" class="form-control
    @error('password') is-invalid @enderror" name="password" required
    autocomplete="current-password">
    @error('password')
    <span class="invalid-feedback" role="alert">
      <strong>{{ $message }}</strong>
    </span>
    @enderror
  </div>
</div>
<div class="form-group row">
  <div class="col-md-6 offset-md-4">
    <div class="form-check">
      <input class="form-check-input" type="checkbox" name="remember"
      id="remember" {{ old('remember') ? 'checked' : '' }}>
      <label class="form-check-label" for="remember">
        {{ __('Запам`ятати мене') }}
      </label>
    </div>
  </div>
</div>
<div class="form-group row mb-0">
  <div class="col-md-8 offset-md-4">
    <button type="submit" class="btn btn-primary">
      {{ __('Залогуватися') }}
    </button>
    @if (Route::has('password.request'))

```



```

        <label for="email" class="col-md-4 col-form-label text-md-right">{{
__('Імейл адреса') }}</label>
        <div class="col-md-6">
            <input id="email" type="email" class="form-control @error('email') is-invalid
@enderror" name="email" value="{{ old('email') }}" required
autocomplete="email">
            @error('email')
            <span class="invalid-feedback" role="alert">
            <strong>{{ $message }}</strong>
            </span>
            @enderror
        </div>
    </div>
    <div class="form-group row">
        <label for="password" class="col-md-4 col-form-label text-md-right">{{
__('Пароль') }}</label>
        <div class="col-md-6">
            <input id="password" type="password" class="form-control
@enderror" name="password" required
autocomplete="new-password">
            @error('password')
            <span class="invalid-feedback" role="alert">
            <strong>{{ $message }}</strong>
            </span>
            @enderror
        </div>
    </div>
    <div class="form-group row">
        <label for="password-confirm" class="col-md-4 col-form-label text-md-
right">{{ __('Підтвердіть пароль') }}</label>
        <div class="col-md-6">
            <input id="password-confirm" type="password" class="form-control"
name="password_confirmation" required autocomplete="new-password">
        </div>
    </div>
    <div class="form-group row mb-0">
        <div class="col-md-6 offset-md-4">
            <button type="submit" class="btn btn-primary">
            {{ __('Зареєструватися') }}
            </button>
        </div>
    </div>

```

```

</form>
</div>
</div>
</div>
</div>
</div>
@endsection

```

Б.7 Зміст файлу about.blade.php

```

@extends('layouts.app')
@section('yiyle')О нас @endsection
@section('content')
<div class="col-md-8 text-white mx-auto border-left">
<h1 class="text-white">Контакта інформація</h1> <br>
<h3 class="text-white">Email:</h3>
<h4 class="text-white">ron.vova.ru@gmail.com</h4> <br>
<h3 class="text-white">Мобільний телефон гарячої лінії:</h3>
<h4 class="text-white">+(066) 932 32 05</h4> <br>
<h3 class="text-white">Робочі години:</h3>
<h4 class="text-white">3 16:00 до 20:00</h4> <br>
</div>
@endsection

```

Б.8 Зміст файлу ansquest.blade.php

```

@extends('layouts.app')
@section('yiyle')FAQ @endsection
@section('content')
<div class="col-md-8 text-white mx-auto border-left">
<h1 class="heading">Найчастіші питання</h1>
<br><br>
<article class="blog-post">
<h3>Чому курси з іншого сайту?</h3>
<h4>Відповідь: Тому що в нас не має спеціалістів для надання бажаних
знань</h4> <br>
<h3>Чому не має адреси нашого офісу?</h3>
<h4>Відповідь: В нас немає офісу :)</h4> <br>
<h3>З якої причину робочих годин для гарячої лінії так мало?</h3>
<h4>Відповідь: у підрозділі тех.підтримки лише одна людина, шкода
його.</h4> <br>

```

```

<h3>Чи будуть курси з інших предметів?</h3>
<h4>Відповідь: Все залежить від ваших відгуків</h4>
</article><!-- /.blog-post -->
</nav>
</div>
</section>@endsection

```

Б.9 Зміст файлу `courses.blade.php`

```

@extends('layouts.app')
@section('yiyile')Курси @endsection
@section('content')
<h1 class="text-white mx-auto" style="width: 200px;">КУРСИ</h1>
<div class="col-md-8 mx-auto">
@foreach($courses as $course)
<div>
<div class="card flex-md-row mb-4 box-shadow h-md-250
"style="background-color: rgba(224,255,255);">
<div class="card-body d-flex flex-column align-items-start">
<strong class="d-inline-block mb-2 text-primary">{{ $course->title
}}</strong>
<div class="mb-1 text-muted">{{ $course->created_at }}</div>
<p class="card-text mb-auto">{{ $course->description }}</p>
<a href="{{ $course->link }}">Перейти до курсу</a>
</div>
</div>
</div>
@endforeach
</div>
@endsection

```

Б.10 Зміст файлу `news.blade.php`

```

@extends('layouts.app')
@section('yiyile')Новини @endsection
@section('content')
<div class="col-md-8 text-white mx-auto border-left">
<h1 class="text-white mx-auto">Новини</h1>
@foreach($news as $newsItem)
<article class="blog-post">
<h2 class="heading">{{ $newsItem->title }}</h2>

```

```

<p class="blog-post-meta">{{ $newsItem->created_at }}</p>
<div>
  {!! $newsItem->body !!}
</div>
</article>
@endforeach
</div>
@endsection

```

Б.11 Зміст файлу reviews.blade.php

```

@extends('layouts.app')
@section('yiyile')Отзывы @endsection
@section('content')
@if($errors->any())
<div class="alert alert-danger">
<ul>
@foreach($errors->all() as $error)
<li>{{ $error }}</li>
@endforeach
</ul>
</div>
@endif
<div class="col-md-6 text-white mx-auto">
<h1>Усі відгуки</h1>
@foreach($reviews as $review)
<div class="alert alert-warning">
<h3>{{ $review->subject }}</h3>
<b>{{ $review->email }}</b>
<p>{{ $review->message }}</p>
</div>
@endforeach
</div>
@auth
<div class="col-md-6 text-white mx-auto">
<form      class="text-white      w-50      mx-auto"      method="post"
action="/reviews/check">
  @csrf
  <input type="email" name="email" id="email" placeholder="Введіть email"
class="form-control"><br>

```

```

        <input type="text" name="subject" id="subject" placeholder="Введіть тему
відгук" class="form-control"><br>
        <textarea name="message" id="message" class="form-control"
placeholder="Введіть відгук "></textarea><br>
        <button type="submit" class="btn btn-success">Відправити</button>
        <br><br><br>
    </form>
</div>
@endauth
@guest
<div class="col-md-6 text-white mx-auto">
    <p>Щоб додати відгук авторизуйтеся за наступним посиланням <a
href="{{ route('login') }}">авторизація</a></p>
</div>
@endguest
@endsection

```

Б.12 Зміст файлу tests.blade.php

```

@extends('layouts.app')
@section('yiyile')Тесты @endsection
@section('content')
<div class="col-md-6 text-white mx-auto">
<h1>Тесты</h1>
<ul>
<li>
<h2>Математика</h2>
<ul>
<li><a href="{{ asset('tests/math/mathtestone.pdf') }}"
download="mathtestone.pdf">Тест ЗНО з математики</a></li>
<li><a href="{{ asset('tests/math/ocinka.pdf') }}"
download="mathtestone.pdf">Оценка с ЗНО</a></li>
<li><a href="{{ asset('tests/math/otvet.pdf') }}"
download="mathtestone.pdf">Відповіді до ЗНО</a></li>
<li><a href="{{ asset('tests/math/rozgorn.pdf') }}"
download="mathtestone.pdf">Розгорнута відповідь до 34 завдання</a></li>
<li><a href="{{ asset('tests/math/shkala100-200.pdf') }}"
download="mathtestone.pdf">Тест ЗНО з математики</a></li>
</ul>
</li>
<li>

```

```

<h2>Англійська мова</h2>
<ul>
  <li><a href="{{ asset('tests/english/Ang_mova-ZNO_2020-Klyuchi.pdf') }}"
download="Ang_mova-ZNO_2020-Klyuchi.pdf">Тест ЗНО з англійської
мови</a></li>
  <li><a href="{{ asset('tests/english/ocenka.pdf') }}"
download="ocenka.pdf">Оцінювання розгорнутого завдання</a></li>
  <li><a href="{{ asset('tests/english/shkala.pdf') }}"
download="shkala.pdf">Шкала балів</a></li>
  <li><a href="{{ asset('tests/english/zoshut.pdf') }}"
download="zoshut.pdf">Відповіді до ЗНО</a></li>
</ul>
</li>
<li>
<h2>Англійська мова для магістратури</h2>
<ul>
  <li><a href="{{ asset('tests/english/EVI_2020-Angl_mova-1_-Zoshyt_1.pdf') }}"
download="EVI_2020-Angl_mova-1_-Zoshyt_1.pdf">Тест ЗНО для
магістратури з англійської мови перша зміна</a></li>
  <li><a href="{{ asset('tests/english/EVI_2020-Angl_mova-2_-Zoshyt_1.pdf') }}"
download="EVI_2020-Angl_mova-2_-Zoshyt_1.pdf">Тест ЗНО для
магістратури з англійської мови друга зміна</a></li>
  <li><a href="{{ asset('tests/english/Angl_mova-YEVI_2020-Klyuchi-
I_zmina-osn_.pdf') }}" download="Angl_mova-YEVI_2020-Klyuchi-I_zmina-
osn_.pdf">Тест ЗНО для магістратури з англійської мови перша зміна
відповіді</a></li>
  <li><a href="{{ asset('tests/english/Angl_mova-YEVI_2020-Klyuchi-
II_zmina-osn.pdf') }}" download="Angl_mova-YEVI_2020-Klyuchi-II_zmina-
osn.pdf">Тест ЗНО для магістратури з англійської мови друга зміна
відповіді</a></li>
</ul>
</li>
<li>
<h2>Українська мова та література</h2>
<ul>
  <li><a href="{{ asset('tests/english/Ukr-mova_lit-ZNO_2020-Klyuchi.pdf') }}"
download="Ukr-mova_lit-ZNO_2020-Klyuchi.pdf">Відповіді до тесту з
української мови та літератури</a></li>
  <li><a href="{{ asset('tests/english/Ukr-mova_lit-ZNO_2020-Zoshyt_1.pdf') }}"
download="Ukr-mova_lit-ZNO_2020-Zoshyt_1.pdf">Тест ЗНО з української
мови та літератури</a></li>

```

```

    <li><a href="{ { asset('tests/english/Ukr_mova-ZNO_2020-VV-Shema-
otsinyuvannya-na-sajt.pdf') } }" download="Ukr_mova-ZNO_2020-VV-Shema-
otsinyuvannya-na-sajt.pdf">Схема оцінювання</a></li>
    <li><a href="{ { asset('tests/english/Ukrayinska-mova-i-literatura-
SHkala_100-200_ZNO_2020.pdf') } }" download="Ukrayinska-mova-i-literatura-
SHkala_100-200_ZNO_2020.pdf">Шкала оцінювання від 100 до 200</a></li>
</ul>
</li>
<li>
<h2>Історія України</h2>
<ul>
    <li><a href="{ { asset('tests/english/Ist_Ukr-ZNO_2020-Klyuchi-osnovna-
sesiya.pdf') } }" download="Ist_Ukr-ZNO_2020-Klyuchi-osnovna-
sesiya.pdf">Відповіді до тесту с ЗНО</a></li>
    <li><a href="{ { asset('tests/english/Ist_Ukr-ZNO_2020-Zoshyt_1_osnovna-
sesiya.pdf') } }" download="Ist_Ukr-ZNO_2020-Zoshyt_1_osnovna-sesiya.pdf">Тест
ЗНО з Історії України</a></li>
    <li><a href="{ { asset('tests/english/Istoriya-Ukrayiny-SHkala-100-
200_ZNO_2020.pdf') } }" download="Istoriya-Ukrayiny-SHkala-100-
200_ZNO_2020.pdf">Шкала балів</a></li>
</ul>
</li>
</ul>
</div>
@endsection

```

ДОДАТОК В
Слайди презентації

Національний університет "Запорізька політехніка"
Кафедра програмних засобів

1

Програмна реалізація вебсайту для підготовки до ЗНО

Виконав	Керівник
ст.гр. КНТ-227	к.ф.м.н., доцент
М.О. Вавілов	Е.В. Терещенко

Рисунок В.1 – Слайд №1

Об'єкт, предмет та мета роботи

2

Об'єкт дослідження – реалізація інтернет-порталу підготовки до ЗНО

Предмет дослідження – сучасні інформаційні портали

Мета роботи – розробко інтернет порталу для підготовки до ЗНО та надання актуальної інформації в цій темі

Рисунок В.2 – Слайд №2

Завдання роботи

3

- ❖ Аналіз предметної області
- ❖ Розробка архітектури програми
- ❖ Вибір мови програмування
- ❖ Вибір фреймворку
- ❖ Розробка діаграм дій клієнта та сервера
- ❖ Розробка дизайну
- ❖ Написання сайту й його бази даних
- ❖ Завантаження сайту на хостинг та тестування
- ❖ Результати та висновки

Рисунок В.3 – Слайд №3

Існуючі інформаційні портали

4



Рисунок В.4 – Слайд №4

Технології необхідні для розробки

5

Основні технології розробки для створення веб сервісу були:

❖ Мова гіпертекстової розмітки HTML 

❖ Мови опису стилів CSS 

❖ Мова програмування PHP 

Рисунок В.5 – Слайд №5

Програмні засоби

6

У якості фреймворку для розробки було обрано Laravel – open-source фреймворк, на архітектурі MVC заснований на Symfony. Головними перевагами якого є безкоштовне розповсюдження, багате ком'юніті, та гарно написана документація.

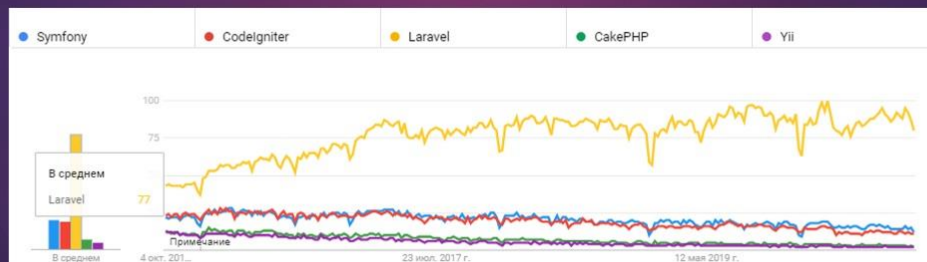


Рисунок В.6 – Слайд №6

Редактор коду IDE

7

Редактором коду було обрано PhpStorm – це платна IDE, з безкоштовним випробним терміном в 30 днів. Було обрану цю середу розробки через:

- ❖ Можливість отримати безкоштовну ліцензію як студент або викладач.
- ❖ Простий й зрозумілий інтерфейс.
- ❖ Автопідстроювання й виправлення коду.
- ❖ Має можливість працювати з використанням PHP,SQL,XML,RUBY.
- ❖ Повна підтримка фронтенд технологій HTML, CSS, Stylus тощо.



Рисунок В.7 – Слайд №7

Архітектура шаблону MVC

8

MVC складається з таких частин:

- ❖ Модель (Model) - являє собі дані які змінюються контролером залежно від запиту.
- ❖ Контролер – інтерпретує запит користувача й передає в модель дані для зміни, на виході дані зміненої моделі повертаються до контролера .
- ❖ Вигляд (View) – графічний інтерфейс в який попадає вже оброблений візуалізований запит.

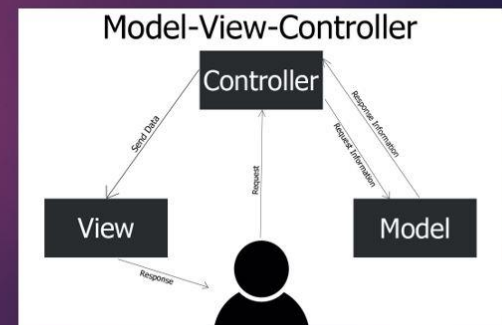


Рисунок В.8 – Слайд №8

Опис взаємодії клієнта та сервера

9

- ❖ Клієнт – веббраузер, інтерфейс користувача для відправки запитів.
- ❖ Сервер – Laravel проєкт на сервері.
- ❖ База даних – місце для збереження інформації.

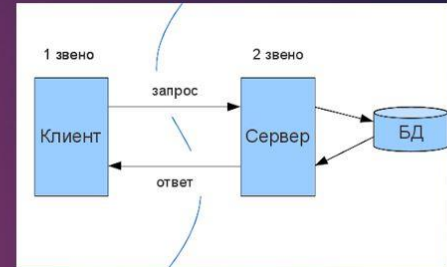


Рисунок В.9 – Слайд №9

Створення логіки HTTP-запитів

10

Вся логіка запитів організовується за допомогою класів контролерів. Контролери групують зв'язну з обробкою HTTP-запитів в окремий клас

Для опису логіки запитів створено клас MainController

Зберігається клас в папці {папка проєкту}/app/http/Controller

Функції в контролері
MainController

```

function home(){...}
function about(){...}
function news(){...}
function tests(){...}
function reviews(){...}
function reviews_check(Request $request){...}
function ansquest(){...}
function cources(){...}
  
```

Рисунок В.10 – Слайд №10

Шаблони проєкту

11

Шаблонізатор Blade який входить до складу Laravel має можливості успадкування шаблонів. Всі шаблони мають розширення .blade.php.

Перелік шаблонів

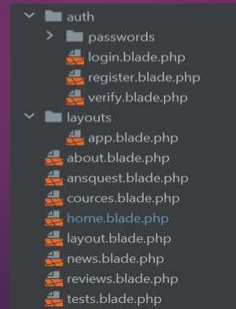


Рисунок В.11 – Слайд №11

Можливості порталу

12

Розроблений інтернет портал дає змогу:

- ❖ Переглядати останні новини й курси для ЗНО
- ❖ Скачувати Тести ЗНО
- ❖ Реєстрації та авторизації на сайті
- ❖ Подивитися на відгуки інших користувачів та залишити свій відгук
- ❖ Подивитися контактну інформацію та часті запитання

Рисунок В.12 – Слайд №12

Діаграми взаємодії клієнта і сервера та діаграма дій користувача у системі

13

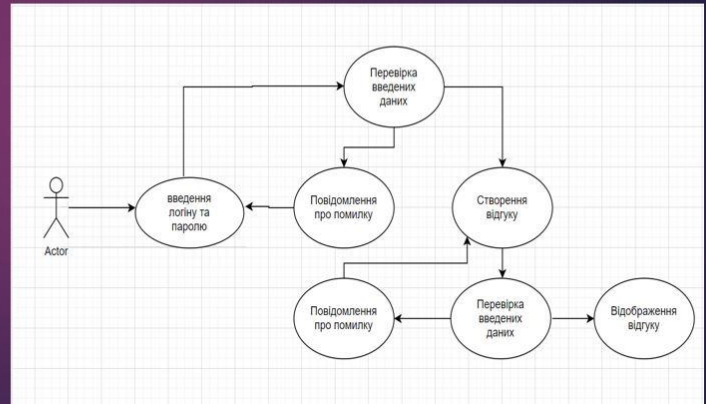
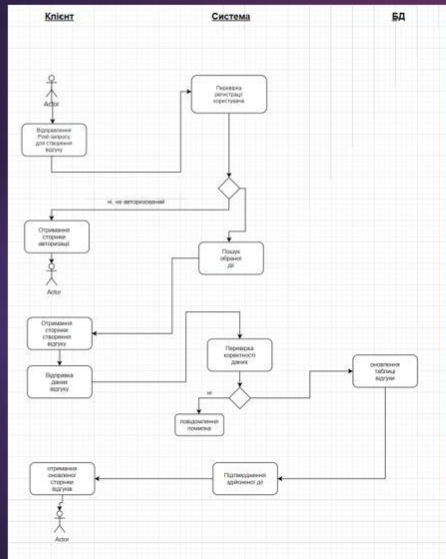


Рисунок В.13 – Слайд №13

Результати роботи. Головна сторінка сайту

14

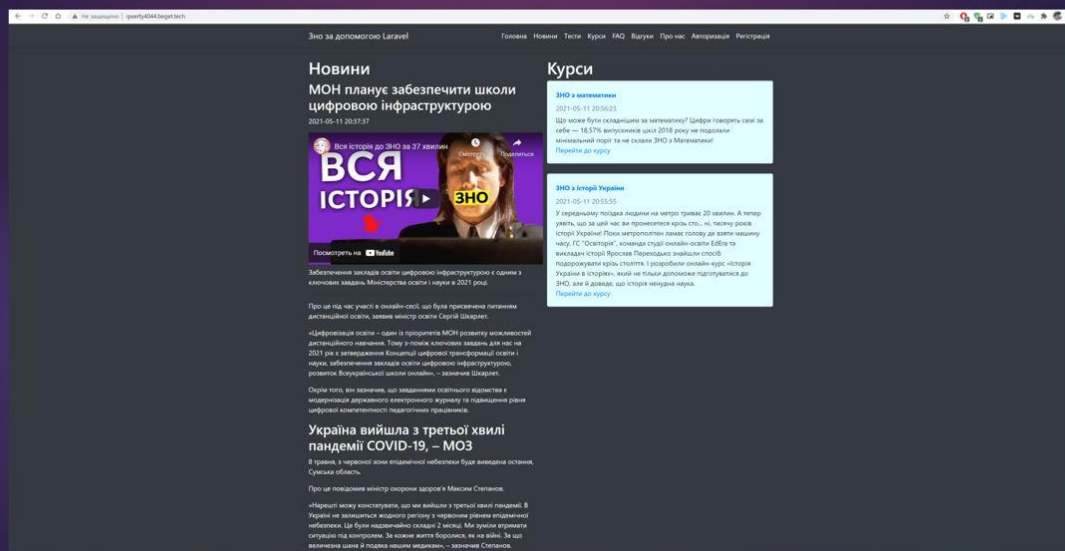


Рисунок В.14 – Слайд №14

Результати роботи. Сторінка Новин

15

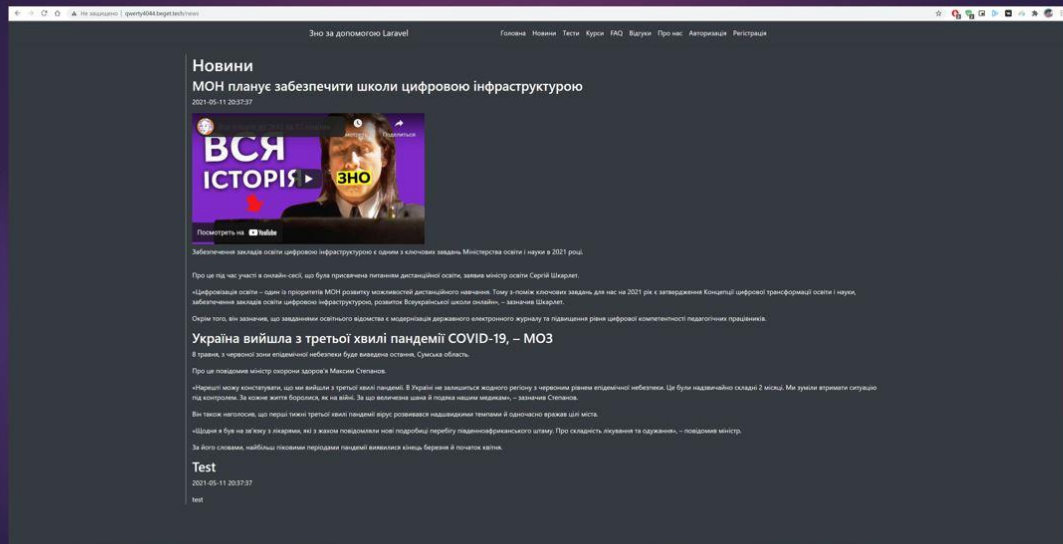


Рисунок В.15 – Слайд №15

Результати роботи. Сторінка тестів

16

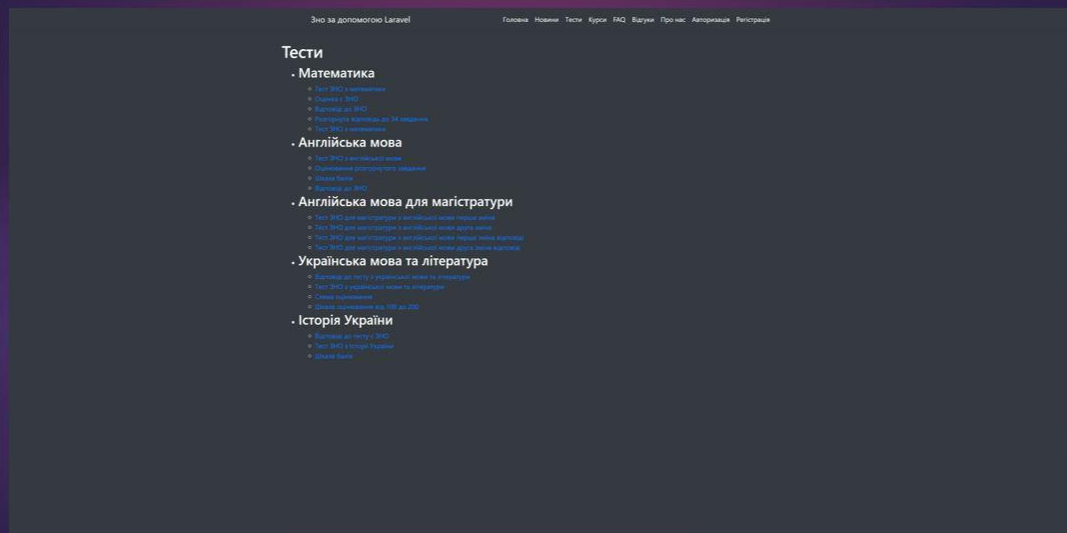


Рисунок В.16 – Слайд №16

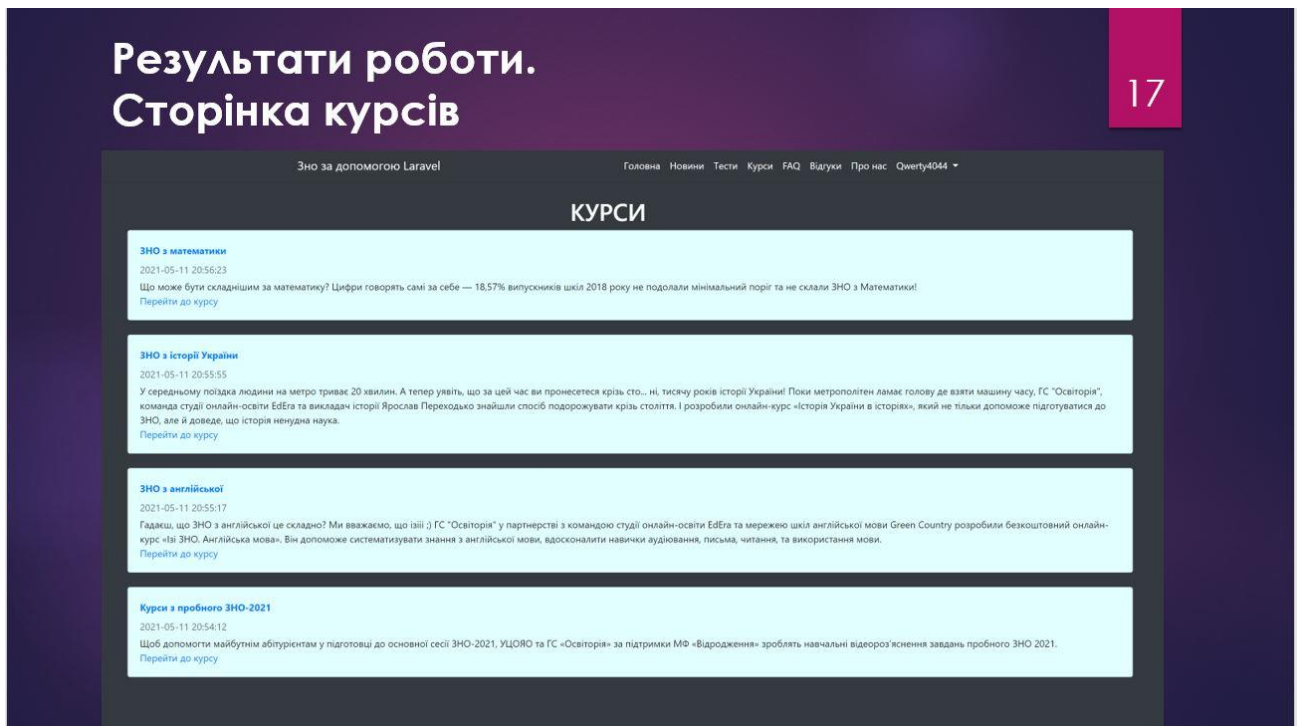


Рисунок В.17 – Слайд №17

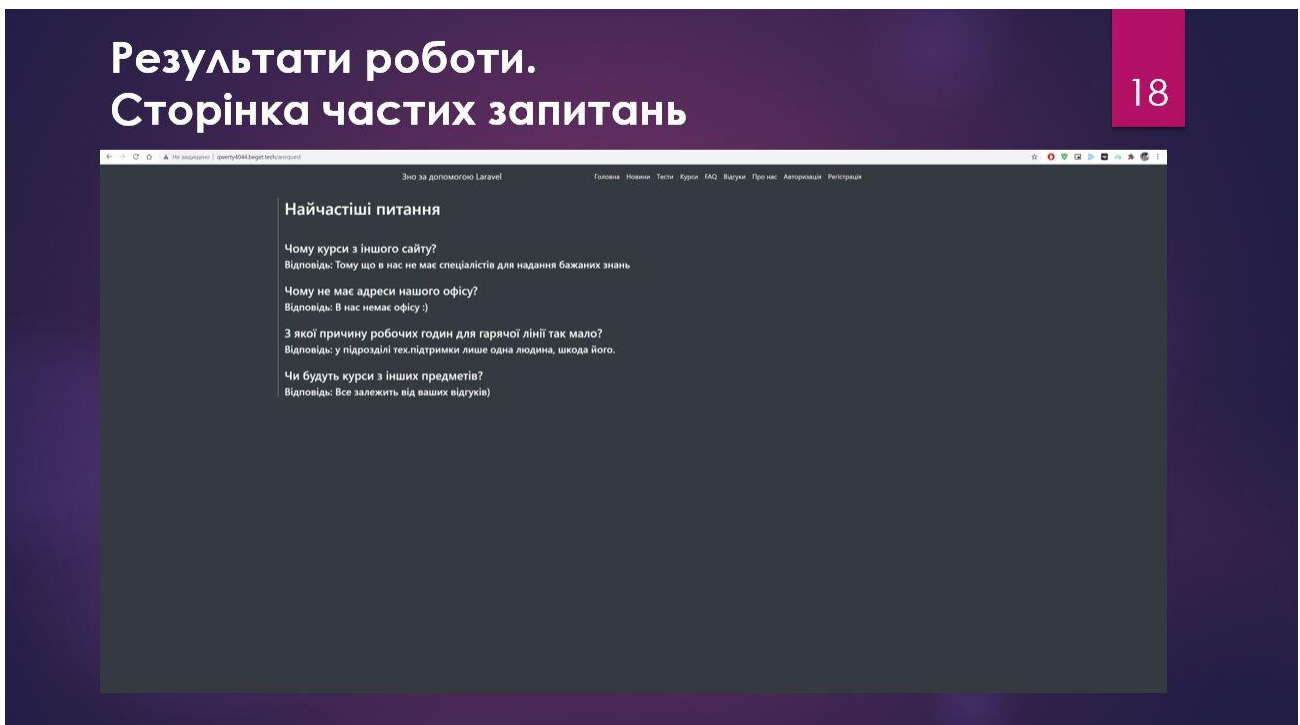


Рисунок В.18 – Слайд №18

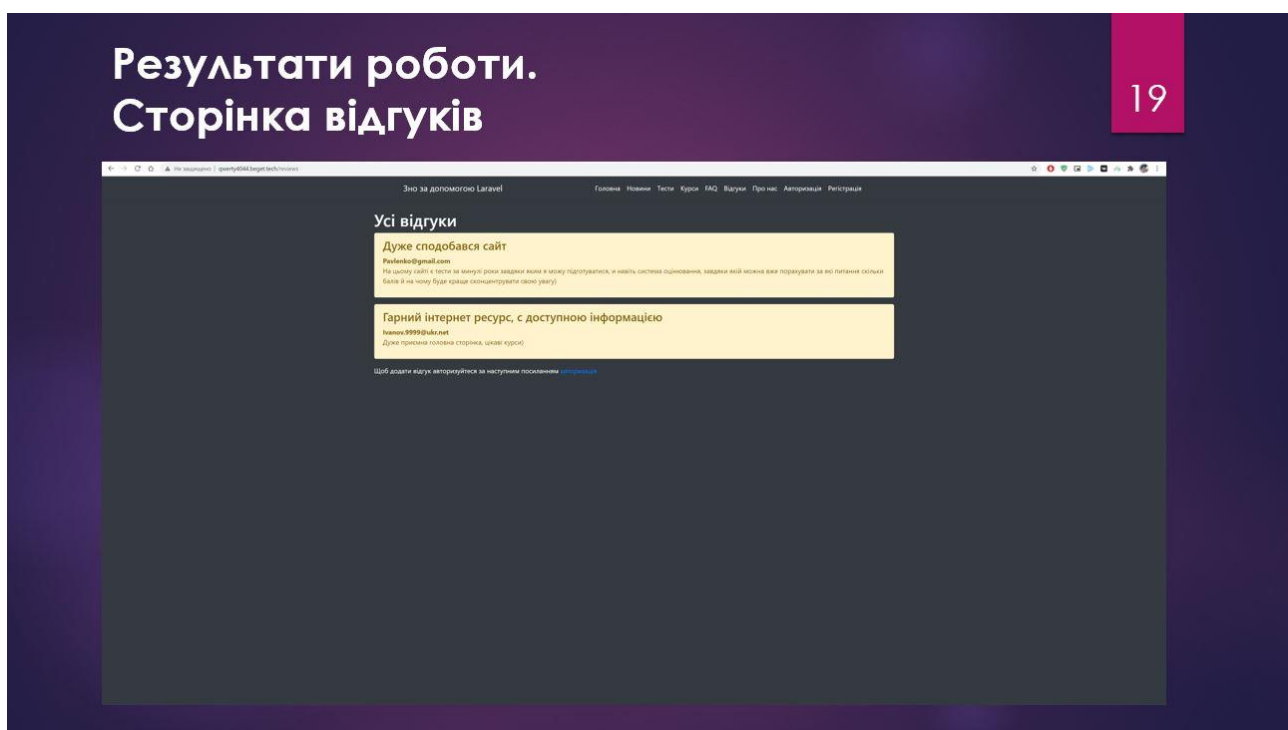


Рисунок В.19 – Слайд №19

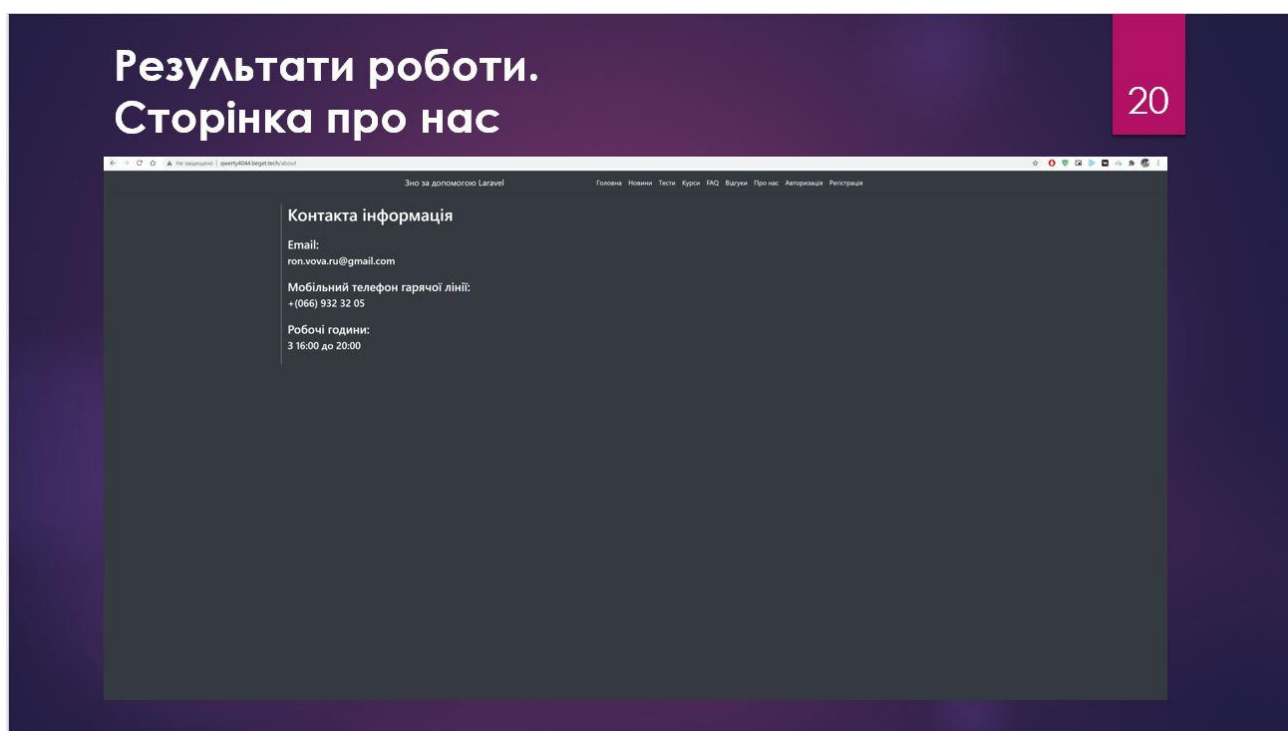


Рисунок В.20 – Слайд №20

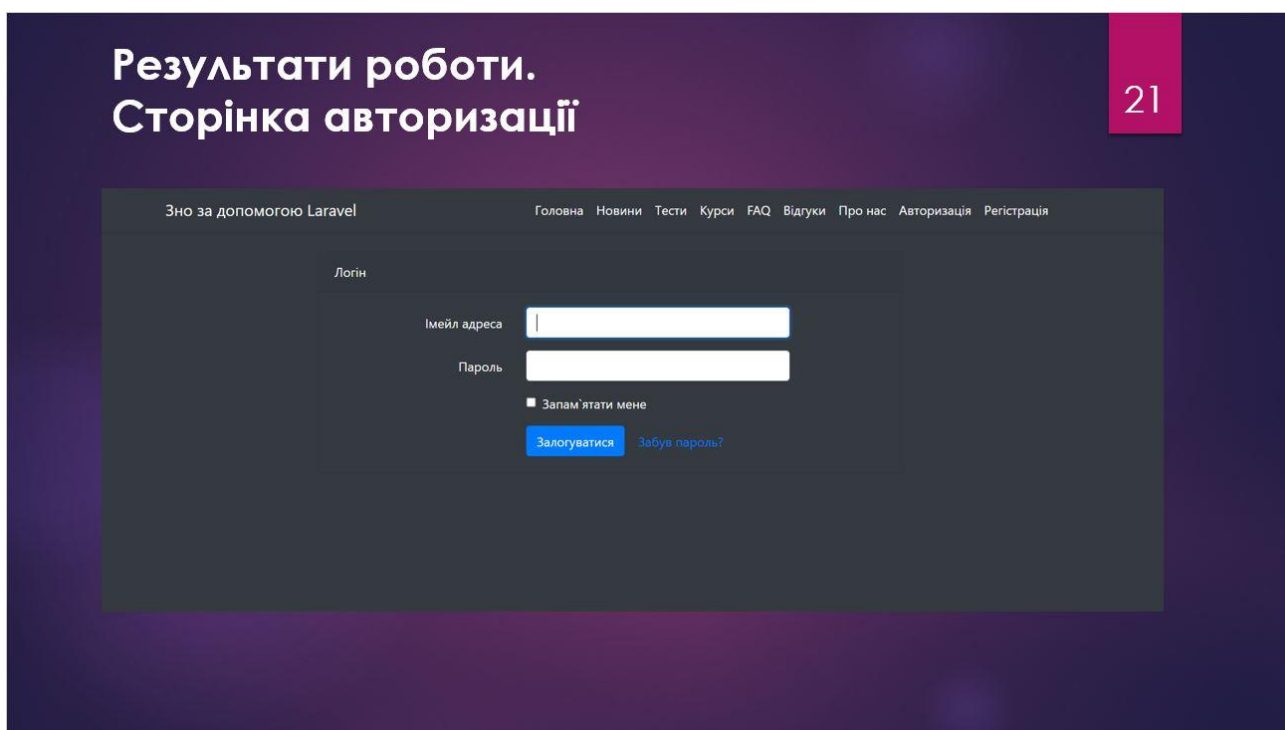


Рисунок В.21 – Слайд №21

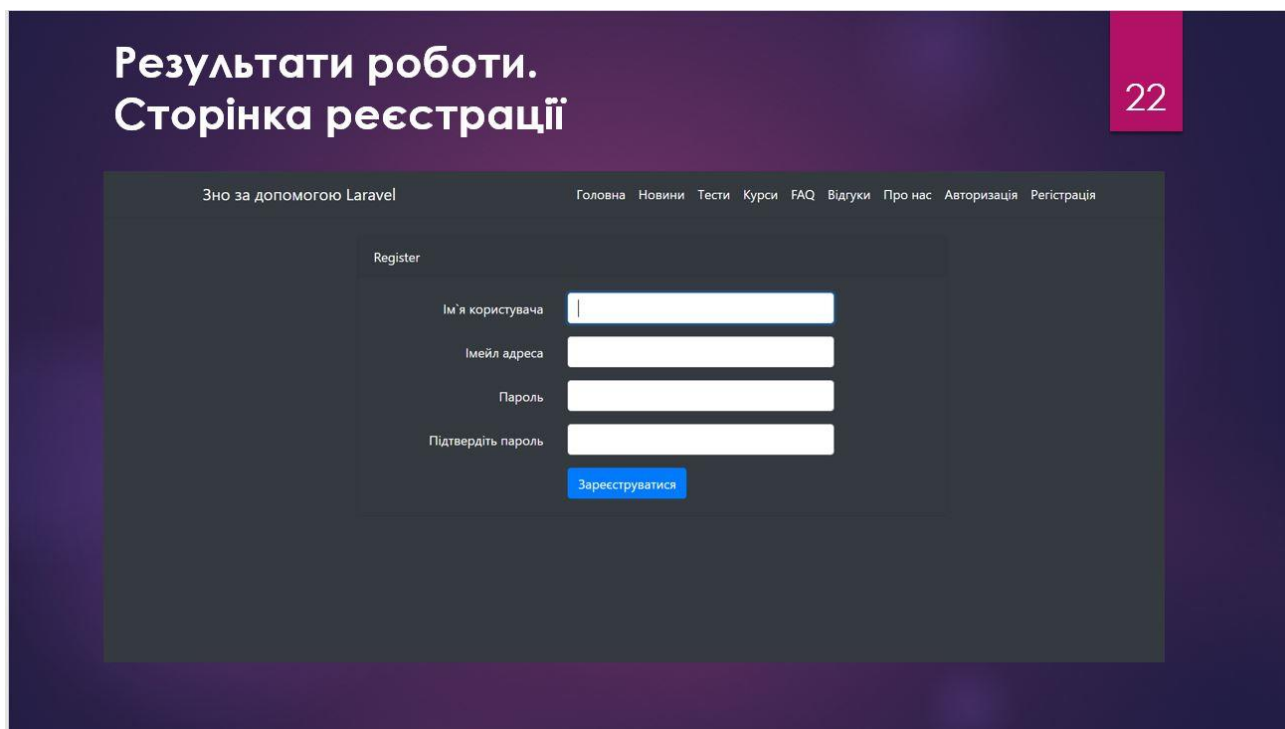


Рисунок В.22 – Слайд №22

Висновки

23

У результаті проведеної роботи, за допомогою сучасного фреймворка Laravel було створено інтернет-портал для підготовки до ЗНО, який надає можливість переглядати останні новини та курси з ЗНО, скачувати тести, ділитися відгуками з іншими користувачами.

Через відкритий серцевий код, кожен може його модифікувати, ще залишилися місця для модифікацій сайту:

- ❖ Захист коду від модифікацій
- ❖ Додати онлайн варіант тестів
- ❖ Додати форум для можливості користувачів спілкуватися один з одним

Джерело посилання на зображення з 6 слайду:
<https://proglife.ru/news/Model-Predstavlenie-Kontroller>

Рисунок В.23 – Слайд №23

24

Дякую за увагу!

Рисунок В.24 – Слайд №24