

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Платформа для розміщення інформаційних
продуктів»**

Виконав студент групи ПЗ-19-3
СІРАЧ Анна Сергіївна

Керівник роботи:
ТОЛСТОЙ Ігор Анатолійович

Рецензент:
ЧИЖМОТРЯ Олена Геннадіївна

Житомир – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

В.о. зав. кафедри інженерії
програмного забезпечення

Андрій МОРОЗОВ

«14» лютого 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **СІРАЧ Анна Сергіївна**

Керівник роботи: **ТОЛСТОЙ Ігор Анатолійович**

Тема роботи: **«Платформа для розміщення інформаційних продуктів»**,
затверджена Наказом закладу вищої освіти від «14» лютого 2023 р., №70с

Вихідні дані для роботи: **Предмет дослідження: методи та засоби проектування платформ. Об'єкт дослідження: комплексна реалізація програмного продукту.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Толстой І.А.,	14.01.2023р.	14.01.2023р.
2	Толстой І.А.,	14.02.2023р.	14.02.2023р.
3	Толстой І.А.,	21.04.2023р.	21.04.2023р.

РЕФЕРАТ

Випускна робота бакалавра представляє собою розроблений програмний продукт веб-платформи для розміщення інформаційних продуктів та пояснювальної записки.

Пояснювальна записка до кваліфікованої роботи складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 51 сторінках друкованого тексту. Пояснювальна записка має 5 сторінок додатків. Список використаних джерел містить 19 найменувань і займає 2 сторінки. В роботі наведено 23 рисунків та 13 таблиць. Загальний обсяг роботи - 56 сторінок.

В роботі визначені головні завдання для розробки платформи, проаналізовано аналогів продукту та методи вирішення поставленої задачі, обґрунтовано вибір архітектури системи. Представлено алгоритм роботи програмного комплексу, структуру та взаємодію його компонентів та описана програмна реалізація платформи.

Ключові слова: PHP, MYSQL, JAVASCRIPT, Yii2, MVC, СЕРВЕР, БД.

					ІПЗ.КР.Б – 121 – 23 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Платформа для розміщення інформаційних продуктів	Літ.	Арк.	Аркушів
Розроб.		Сірач А.С.						
Керівник		Толстой І.А.					3	
Н. контр.		Тичина О.П.				ФІКТ Гр. ІПЗ-19-3[2]		
Зав. каф.		Морозов А.В.						

ABSTRACT

Bachelor's graduation work is a developed software product of a web platform for posting information products and an explanatory note.

The explanatory note to the diploma project consists of a list of conditional abbreviations, introduction, three sections, conclusions, a list of used literature and an appendix.

The text part is laid out on 51 pages of printed text. The explanatory note has 5 pages of appendices. The list of used sources contains 19 names and occupies 2 pages. The paper contains 23 pictures and 13 tables. The total amount of work is 56 pages.

The main tasks for the development of the platform are defined in the work, analogues of the product and methods for solving the task are analyzed, the choice of the system architecture is justified. The algorithm of the software complex, the structure and interaction of its components are presented and the software implementation of the platform is described.

Keywords: PHP, MYSQL, JAVASCRIPT, YII2, MVC, SERVER, DB.

					ИПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						4

ЗМІСТ

РЕФЕРАТ	3
ВСТУП	6
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПЛАТФОРМИ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ	7
1.1. Постановка задачі.....	7
1.2. Аналіз аналогів програмного продукту	8
1.3. Вибір архітектури веб-орієнтованої системи навчальної платформи.....	13
1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	14
Висновки до розділу 1	15
РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАТФОРМИ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ	17
2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи.....	17
2.2. Розробка бази даних системи	20
2.3. Проектування та реалізація алгоритмів роботи системи.....	26
2.4. Реалізація додатку платформи для розміщення інформаційних продуктів	31
Висновки до розділу 2	38
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ПЛАТФОРМОЮ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ.....	39
3.1. Порядок встановлення та налаштування параметрів системи	39
3.2. Структура інтерфейсу платформи для розміщення інформаційних продуктів	40
3.3. Тестування роботи програмного комплексу.....	45
Висновки до розділу 3	46
ВИСНОВКИ.....	48
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	49
ДОДАТКИ	51

ВСТУП

Актуальність теми: На сьогоднішній день все більше і більше людей зацікавлюються в отриманні нових знань. Враховуючи сьогоднішні умови, для людей є дуже актуальною можливість отримати нові знання на сучасних онлайн курсах, щоб мати змогу в зручний час навчатися, освіжити інформацію в голові та взагалі перекваліфікуватися. Тому розробка платформи для розміщення інформаційних продуктів є над актуальною темою.

Метою випускної кваліфікованої роботи є закріплення та вдосконалення отриманих теоретичних та практичних знань з HTML, CSS, JS, PHP, які будуть використанні при розробці програмного продукту «Платформа для розміщення інформаційних продуктів».

Результатом виконання випускної роботи є реалізоване програмне забезпечення для розміщення навчальних інформаційних матеріалів на спеціалізованій платформі. Для досягнення даної мети та результатів необхідно виконати низку поставлених задач:

- аналіз аналогів продукту та способів їхньої реалізації;
- аналіз теоретичних засад моделювання програмного забезпечення;
- визначення алгоритму та архітектури для реалізації програмного додатку;
- реалізація програмної системи;
- оптимізація додатку для кращої та результативнішої взаємодії з ним;
- тестування та усунення помилок реалізованої програми;

Предмет дослідження є методи та засоби проектування «Платформи для розміщення інформаційних продуктів».

Об'єктом дослідження є комплексна реалізація програмного продукту «Платформа для розміщення інформаційних продуктів».

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПЛАТФОРМИ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ

1.1. Постановка задачі

Основною метою створення освітньої платформи з навчальними курсами є надання доступних і гнучких можливостей навчання для людей у всьому світі. Усуваючи географічні бар'єри, пропонуючи різноманітні варіанти курсу та сприяючи персоналізованому навчанню, платформа має на меті надати учням можливість здобувати нові знання, розвивати навички та сприяти особистому та професійному зростанню. Крім того, платформа створює інтерактивне та привабливе навчальне середовище, сприяє розвитку глобальної навчальної спільноти та пропонує масштабованість і економічну ефективність порівняно з традиційними моделями освіти.

1. Розробити дизайн користувацького інтерфейсу.

- 1.1. Створити дизайн-концепт графічного дизайну за допомогою графічного редактора Figma
- 1.2. За допомогою Figma створити графічні зображення основних елементів керування (кнопок, стрілок та ін.)
- 1.3. Зверстати та застосувати елементи інтерфейсу:
 - 1.3.1. Головна сторінка
 - 1.3.2. Вхід та реєстрація
 - 1.3.3. Особистий кабінет
 - 1.3.4. Мої курси
 - 1.3.5. Уроки
 - 1.3.6. Учні та доступ
 - 1.3.7. Модерація коментарів
 - 1.3.8. Успішність на курсах
 - 1.3.9. Користувачі
 - 1.3.10. Курси

- 1.3.11. Учні
- 1.3.12. Статистика для адміністратора
- 1.3.13. Експорт/імпорт SQL або CSV

2. Проектування та розробка основних контролерів

- 2.1. Контролер для основного сайту
- 2.2. Контролер для користувачів
- 2.3. Контролер для курсів
- 2.4. Контролер для уроків
- 2.5. Контролер для модерації
- 2.6. Контролер для БД
- 2.7. Контролер для студентів

3. Проектування та розробка основних модулів

- 3.1. Модуль для коментарів
- 3.2. Модуль для курсу
- 3.3. Модуль для уроків
- 3.4. Модуль для модерації
- 3.5. Модуль для відслідковування прогресу та успішності
- 3.6. Модуль для реєстрації
- 3.7. Модуль для студентів
- 3.8. Модуль для користувачів

Результатом реалізації поставленого завдання є функціональна платформа для розміщення навчальних курсів.

1.2. Аналіз аналогів програмного продукту

У цьому аналізі ми вивчимо різні освітні платформи, які пропонують можливості онлайн-навчання. Ці платформи служать цифровими хабами для людей, які прагнуть отримати нові знання, розвивати навички та продовжувати особисте та професійне зростання. Вивчаючи ці платформи, ми прагнемо зрозуміти

загальні особливості та характеристики, які роблять їх успішними в наданні онлайн-освіти.

Завдяки цьому аналізу ми будемо класифікувати освітні платформи на основі їх первинної спрямованості та підходу. Сюди входять:

- Платформи онлайн-навчання: ці платформи, такі як Coursera, Udey та edX, пропонують широкий спектр курсів з різних предметів. Вони співпрацюють з університетами, експертами галузі та інструкторами, щоб надати учням доступ до високоякісного освітнього контенту.
- Платформи на основі навичок: Платформи, такі як Skillshare та LinkedIn Learning, зосереджуються на розробці конкретних навичок, таких як дизайн, кодування, маркетинг або фотографія. Вони пропонують колекцію курсів, навчальних посібників та проектного навчання для підвищення практичних навичок.
- Платформи для вивчення мови: Duolingo, Babbel та Rosetta Stone потрапляють до цієї категорії. Ці платформи пропонують інтерактивні уроки та вправи, щоб допомогти учням придбати знання іноземної мови у власному темпі.
- Спеціалізовані платформи: деякі платформи спеціалізуються в певних областях або галузях. Наприклад, Codecademy фокусується на кодуванні та програмуванні, тоді як MasterClass пропонує курси під керівництвом експертів, які викладаються відомими професіоналами в таких галузях, як кіно, музика та письмо.

Усі ці платформи мають низку спільних функцій та переваг, такі як доступність, різноманітність курсів, інтерактивні інструменти навчання, відстеження прогресу, варіанти сертифікації та підтримка учнів. Аби краще глибше проаналізувати дані програми, було обрано для аналізу три популярних навчальних платформ в Україні – Udey, Prometheus, Softbook.

Udemy є популярною платформою онлайн-навчання, яка пропонує величезну колекцію курсів з широкого кола предметів. Доступ до платформи: <https://www.udemy.com/>

Особливості даної платформи:

- **Різноманітність курсу:** Udemy надає різноманітний вибір курсів, включаючи бізнес, технології, мистецтво, особистий розвиток та багато іншого.
- **Експертний спікер:** Курси по Udemy створюються і викладаються експертами і професіоналами в своїх областях.
- **Самостійне навчання:** учні можуть отримати доступ до матеріалів курсу в будь-який час і прогресувати через курси в своєму власному темпі.
- **Довічний доступ:** Після зарахування на курс, учні мають довічний доступ до матеріалів курсу та майбутніх оновлень.
- **Рейтинги та відгуки користувачів:** Учні можуть читати відгуки та рейтинги від попередніх студентів, щоб оцінити якість та актуальність курсу

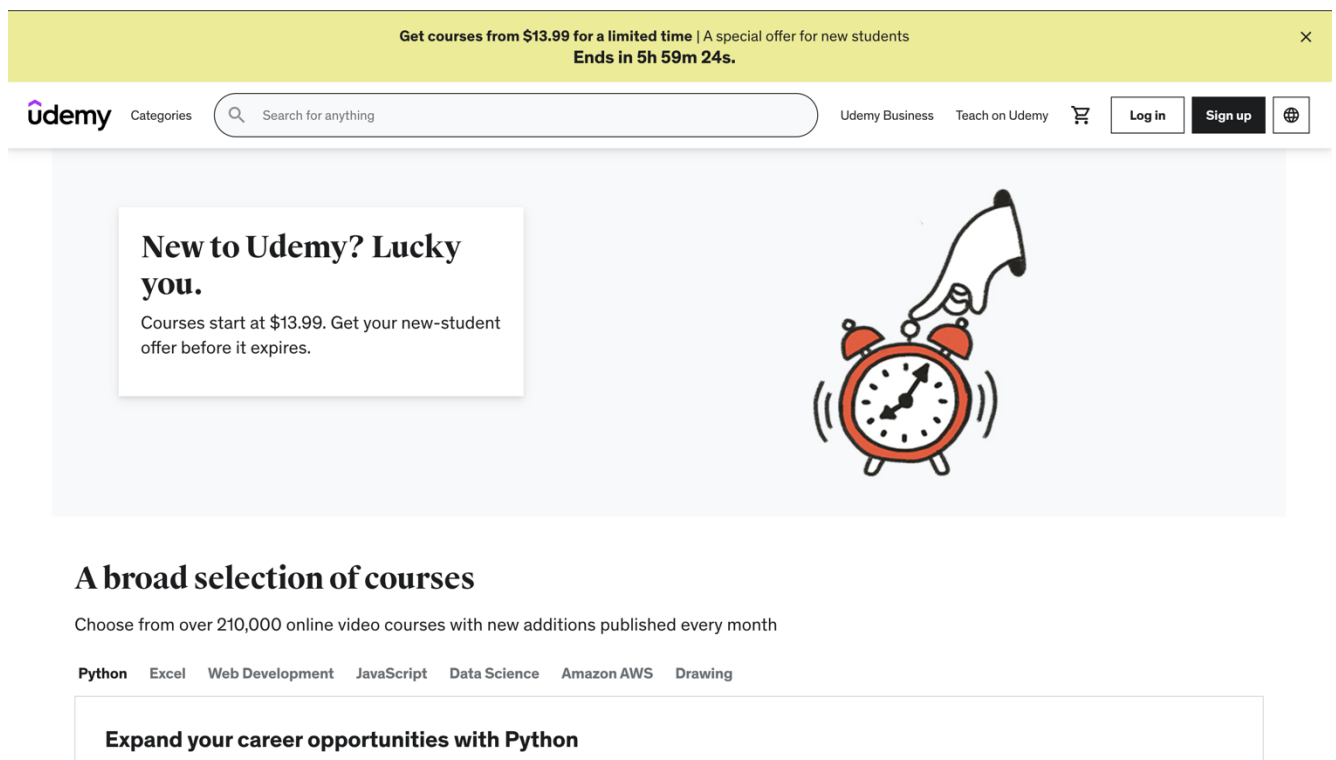


Рис. 1.1. Інтерфейс платформи Udemy

					ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						10

Prometheus - це українська платформа електронного навчання, яка пропонує широкий спектр курсів з різних предметів.

Особливості даної платформи:

- **Різноманітність курсу:** Prometheus проводить курси в різних областях, включаючи навчальні предмети, професійні навички та особистісний розвиток.
- **Інтерактивне навчання:** Платформа включає інтерактивні елементи, вікторини та мультимедіа для підвищення досвіду навчання.
- **Співпраця:** Prometheus заохочує співпрацю між учнями через дискусійні форуми та групові проекти.
- **Сертифікація:** Учні можуть заробляти сертифікати після успішного завершення курсів, перевіряючи свої досягнення.
- **Доступність:** Курси на Prometheus доступні в Інтернеті, що дозволяє учням навчатися у власній зручності.

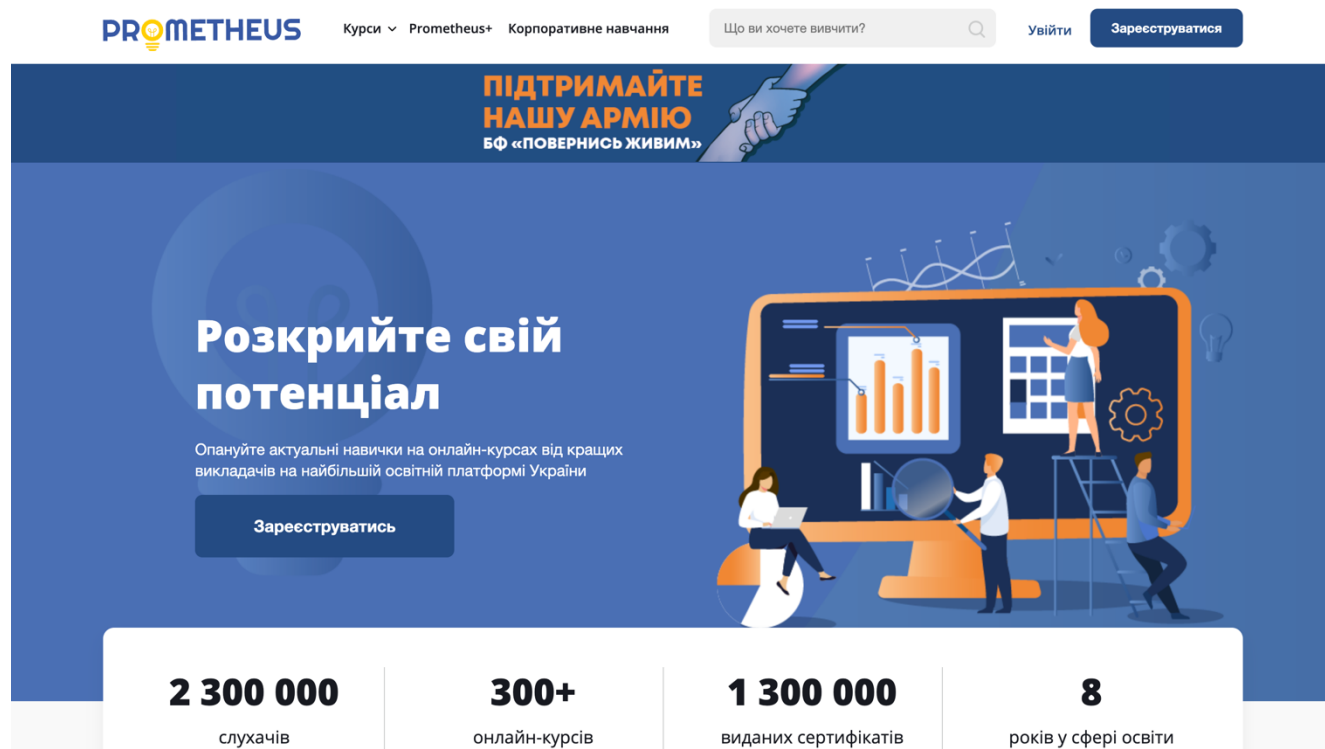


Рис. 1.2. . Інтерфейс платформи Prometheus

Softbook - цифрова навчальна платформа, яка спеціалізується на наданні навчальних матеріалів та ресурсів для навчальних предметів.

Особливості даної платформи:

- Softbook пропонує навчальні матеріали, які відповідають національним стандартам навчальних програм.
- Мультимедійні ресурси: Платформа надає інтерактивні електронні книги, відео, вікторини та інші мультимедійні ресурси для залучення учнів.
- Підтримка вчителів: Softbook пропонує ресурси та інструменти для вчителів, включаючи плани уроків, оцінки та відстеження прогресу.
- Персоналізація: Платформа дозволяє вчителям налаштовувати шляхи навчання та призначати конкретні ресурси на основі індивідуальних потреб учнів.
- Автономний доступ: Softbook пропонує автономний доступ до вибраних ресурсів, що дозволяє навчатися в середовищах з обмеженим підключенням до Інтернету.

Переїжджайте на нашу платформу та отримайте 2 місяці користування у подарунок! [Докладніше](#)



[Головна](#) [Можливості](#) [Ціни](#) [Допомога](#) [Про нас](#)

UA ▾

[Вхід](#)

Заробляйте на своїх знаннях та досвіді 24/7

Відкривайте власну онлайн-школу за лічені хвилини.
Створюйте та продавайте курси для необмеженої
кількості людей

Повна автоматизація

Отримати доступ на 30 днів безкоштовно

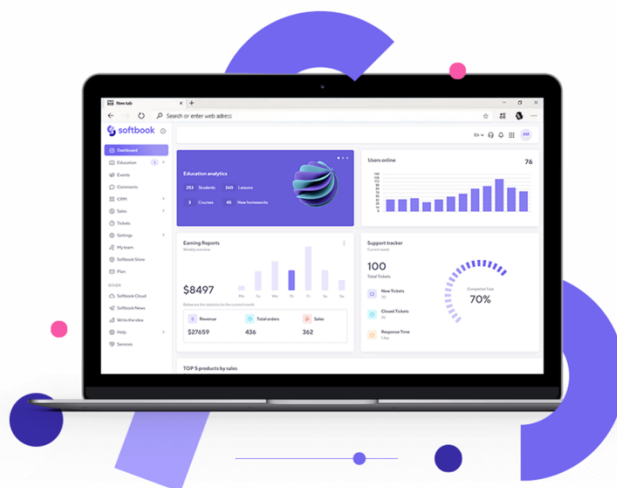


Рис. 1.3. Інтерфейс платформи Softbook

Таблиця 1.1 Порівняння платформ

Платформа	Udemy	Prometheus	Softbook
Різноманітність курсів	+	+	-
Інтерактивне навчання	-	+	+
Співпраця	-	+	-
Сертифікація	-	+	-
Доступність	+	+	+
Узгодження з навчальною програмою	-	-	+
Підтримка вчителів	-	-	+
Офлайн доступ	-	-	+

В результаті аналізу платформ-аналогів для власного веб-додатку було зіставлено наступний список вимог до функціоналу: створення, перегляд, редагування, видалення курсів/уроків, можливість відслідковування прогресу кожного учня, створення та модерація коментарів, пошук по учням, коментарям. Також потрібно передбачити чотири ролі для сайту: гість, учень, автор курсу та адмін.

1.3. Вибір архітектури веб-орієнтованої системи навчальної платформи

При проектуванні програмного комплексу необхідно застосувати архітектуру MVC (Model-View-Controller) для розмежування алгоритмічної частини від інтерфейсу та забезпечення спільного використання компонентів [3].

MVC - це патерн проектування веб-додатків, який включає в себе кілька дрібніших шаблонів. При використанні MVC на три окремих компонента розділені модель даних програми, призначений для користувача інтерфейс і логіка взаємодії користувача з системою, завдяки чому модифікація одного з цих компонентів надає мінімальний вплив на інші або не робить його зовсім.

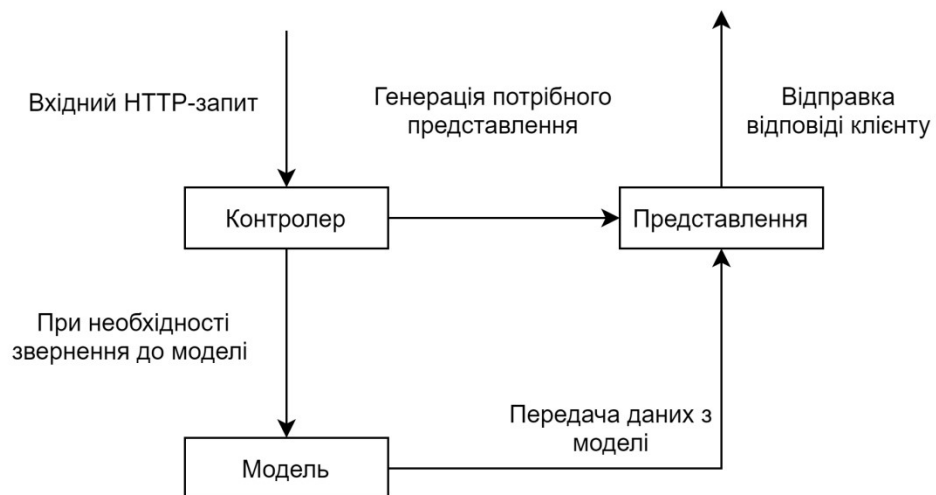


Рис. 1.4 Загальний принцип роботи MVC

На початку роботи додаток повинен підключитись до бази даних та оновити її до останньої версії за допомогою міграцій. Далі додаток підключає необхідні проміжні сервіси та чекає вхідні запити.

Клієнт матиме клієнт-серверну архітектуру, тому розглянемо нижче цикл запиту.

Після того, як сервер отримав запит від користувача, формується функція для виконання запитів до бази даних, оброблюються дані сесії.

Далі перевіряється чи має користувач доступ до сторінки, яку запитує. Якщо користувач не має доступу, то він перенаправляється на сторінку помилки. Якщо навпаки, то запит оброблюється. При необхідності сервер надішле запит до бази та даних та оброблює отримані дані.

Після обробки запиту сервер оброблює відповідне представлення та надсилає сформований html-код користувачу. В розробці передбачено, щоб користувач мав взаємодію з сервером лише за допомогою графічного інтерфейсу, що якраз і передбачає патерн MVC [6].

1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Програмний комплекс повинен бути побудований на клієнт-серверній архітектурі з використанням веб-технологій, що не вимагають додаткового ліцензування. Серверна частина програмного комплексу має базуватися на веб

сервері Open Server або Xampp та системі керування базами даних MySQL. Базовою мовою програмування пропонується обрати PHP. При розробці буде застосовуватися фреймворк Yii2 [5].

Клієнтська частина повинна бути побудована на стандартних веб-технологіях, що не вимагають встановлення додаткових компонентів на комп'ютер (не рекомендується використовувати Flash або Java-аплети). Сторінки інтерфейсу генеруються сервером за стандартом HTML5, оформлення здійснюється за допомогою таблиць стилів CSS (бібліотека Bootstrap 5 [12]), а інтерактивність сторінок забезпечується скриптами Javascript та асинхронними запитами через AJAX.

Клієнтська частина програмного комплексу повинна бути максимально сумісною з різними веб-браузерами, таких як Mozilla FireFox та Google Chrome, і працювати на різних платформах, забезпечуючи зручний та надійний доступ до основних функцій. Інтерфейс користувача повинен бути інтуїтивно зрозумілим та привабливим, з використанням сучасних дизайнерських підходів і технік. При розробці важливо забезпечити адаптивність до різних пристроїв, включаючи мобільні телефони та планшети, для зручного використання платформою на різних пристроях та забезпечення мобільності для користувачів.

Висновки до розділу 1

В даному розділі ми змогли глибше проаналізувати обрану тематику та провели аналіз різних освітніх платформ, які пропонують можливості онлайн-навчання. Завдяки цьому аналізу ми змогли визначати основні задачі для нашої майбутньої платформи та краще зрозуміти загальні особливості та характеристики, які роблять ці платформи успішними в наданні онлайн-освіти.

Крім того, ми обговорили вибір веб-орієнтованої архітектури для освітньої платформи, зокрема Model-View-Controller (MVC). Архітектура MVC дозволяє розділяти модель даних програми, інтерфейс користувача та логіку взаємодії, забезпечуючи мінімальний вплив або модифікацію на інші компоненти при зміні.

Цей архітектурний підхід сприяє модульній розробці, багаторазовому використанню коду та ремонтпридатності.

Був обґрунтований і вибір інструментів, і апаратних вимог для розробки програмного комплексу. Серверна сторона платформи основана на веб-сервері Open Server або Xampp і використовуватиме систему управління базами даних MySQL. Основною мовою програмування було обрано PHP, а для розробки запропоновано Yii2 фреймворк [10]. На стороні клієнта стандартні веб-технології, такі як HTML5, CSS та JavaScript, використовуються для створення сторінок інтерфейсу, забезпечення стилізації та підвищення інтерактивності з AJAX для асинхронних запитів. Було підкреслено сумісність з різними веб-браузерами та платформами, щоб забезпечити бездоганний користувацький досвід.

В результаті, цей розділ дав початок розвитку освітньої платформи, визначивши її цілі, провівши аналіз аналогічних платформ, вибравши архітектуру системи, інструментів і апаратних вимог.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАТФОРМИ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ

2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи

Даний документ описує високорівневі вимоги, якими повинна володіти online система “Платформа для розміщення інформаційних продуктів”, щоб мати можливість безперервно виконувати покладені на неї функції, тим самим відповідно задовольнивши всі зазначені стандарти, специфікації та інші формальні документи. Зазначені умови, яких повинен дотримуватись користувач для роботи з даною інформаційною системою та вказані основні ділові переваги даної ІС [13]. Приведені можливості системи, обмеження та показники якості.

Бізнес вимоги

1. Основні цілі: проект створюється з метою спрощення процесу створення інформаційного продукту та його аналізу;
2. Представлення проекту: проект буде реалізовано у вигляді сайту, що містить структуровану інформацію по даній тематичній області;

Вимоги зовнішніх користувачів (Студент)

1. Можливість права повного доступу (створення, редагування та видалення) до власного профілю;
2. Можливість перегляду доступних курсів, на які Студент зарахований;
3. Можливість перегляду доступних уроків;
4. Можливість створення коментарів та перегляду власних та доступних коментарів під уроками;

Вимоги зовнішніх користувачів (Автори)

1. Можливість права повного доступу (створення, редагування та видалення) до власного профілю;
2. Можливість права повного доступу до власного курсу;
3. Можливість права повного доступу до уроків;

4. Можливість додавання студентів до курсу, переглядання списків зарахованих студентів та вилучання студентів з курсу;
5. Можливість створення та видалення власних коментарів;
6. Можливість переглядання наявних коментарів від учнів та їх модерації (активування/деактивування, видалення);
7. Можливість збору статистичних даних з успішності на усіх курсах;

Вимоги внутрішніх користувачів (Адміністратор):

1. Можливість надання та видалення користувачам права Автор;
2. Можливість перегляду та редагування курсів;
3. Можливість перегляду та додавання студентів на курси;
4. Можливість права повного доступу до особистої інформації в профілі користувачів;
5. Можливість перегляду статистичних даних про курси, авторів, успішності студентів;
6. Можливість архівування БД чи окремих таблиць;

Характеристика об'єкта комп'ютеризації

Користувач додатку має достатньо можливостей та зручно розроблений інтерфейс для взаємодії з авторами чи модераторами курсів та адміністрацією платформи.

Функціональні вимоги:

1. Авторизація в системі: В системі повинна бути представлена можливість реєстрації користувача та присвоєння йому відповідної ролі (студент, автор, адміністратор);
2. Можливість збереження інформації: Система повинна зберігати інформацію і надавати можливість керувати нею;

Нефункціональні вимоги:

1. Сприйняття

- Час, потрібний для навчання інструментами роботи з інформаційною системою для звичайних користувачів – 2-3 години, а для досвідчених – 1 година;
- Час відповіді системи для звичайних запитів не повинна перевищувати 5 секунд, а для більш складних запитів – 10 сек;
- Інтерфейс представлення ІС повинен бути інтуїтивно зручним для користувача та не вимагати від нього додаткової підготовки;

2. Надійність

- Доступність – час, потрібний для обслуговування системи не повинен перевищувати 10% від загального часу роботи;
- Середній час безперервної роботи – 20 робочих днів;
- Максимальна норма помилок та дефектів в роботі системи – 1 помилка на 1000 запитів користувача;

3. Продуктивність

- Система повинна підтримувати мінімум 100 одночасно працюючих користувачів, пов'язаних з спільною базою даних;

4. Можливість експлуатації

- Масштабування – система повинна мати можливість збільшувати;
- потужності (продуктивність), зі збільшенням користувачів таким чином, щоб це аж ні як негативно не відобразилося на її роботі;

Системні вимоги:

1. Вимоги до середовища виконання:

Система повинна задовольняти вимогам на комп'ютері, що знаходиться в наступній мінімальній комплектації:

- операційна система Windows 7 або вище, Linux 8.10 та вище, MacOS 10.6 Snow Leopard та вище;
- 32-розрядний (x86) або 64-розрядний (x64) процесор із тактовою частотою 1 ГГц або вище;

- 1 ГБ (для 32-розрядних процесорів) або 2 ГБ (для 64-розрядних процесорів) ОЗУ;
- 16 ГБ (для 32-розрядної системи) або 20 ГБ (для 64-розрядної системи) вільного місця на жорсткому диску;

2. Вимоги до СУБД та доступу до даних:

- У ядрі системи повинна бути представлена СУБД реляційного доступу;

Аналіз вимог користувачів дав підстави для визначення варіантів використання сайту-платформи для інформаційних курсів. На рис. 1 (Додаток А) наведено діаграму використання програмної системи [16].

2.2. Розробка бази даних системи

На цьому етапі розробки курсового проекту маємо чітко сформовані задачі. Обравши засоби реалізації, перейдемо до проектування структури бази даних платформи для розміщення інформаційних продуктів [1].

База даних в наслідок проектування буде складатись з 12 таблиць:

- Auth_assignment
- Auth_item
- Auth_item_child
- Auth_role
- Comment
- Course
- Lesson
- Migration
- Progress
- Status
- Student
- User

```
CREATE VIEW `lessonpassed` AS
SELECT
    `course`.`course_id` AS `course_id`,
    COUNT(`lesson`.`lesson_id`) AS `passed`
FROM
    ((`course`
    JOIN `lesson` ON (`lesson`.`course_id` = `course`.`course_id`))
    JOIN `progress` ON (`progress`.`lesson_id` = `lesson`.`lesson_id`))
WHERE
    `progress`.`status_id` > 2
GROUP BY
    `course`.`course_id`;
```

id	course_name	student_course
20	СММ та веб-дизайн	3
20	Таргет	2
20	База даних	2
24	Тест	1

На основі створених таблиць була спроектована фізична (Рис. 2.3) та логічна (Рис. 2.4) ERD діаграми.

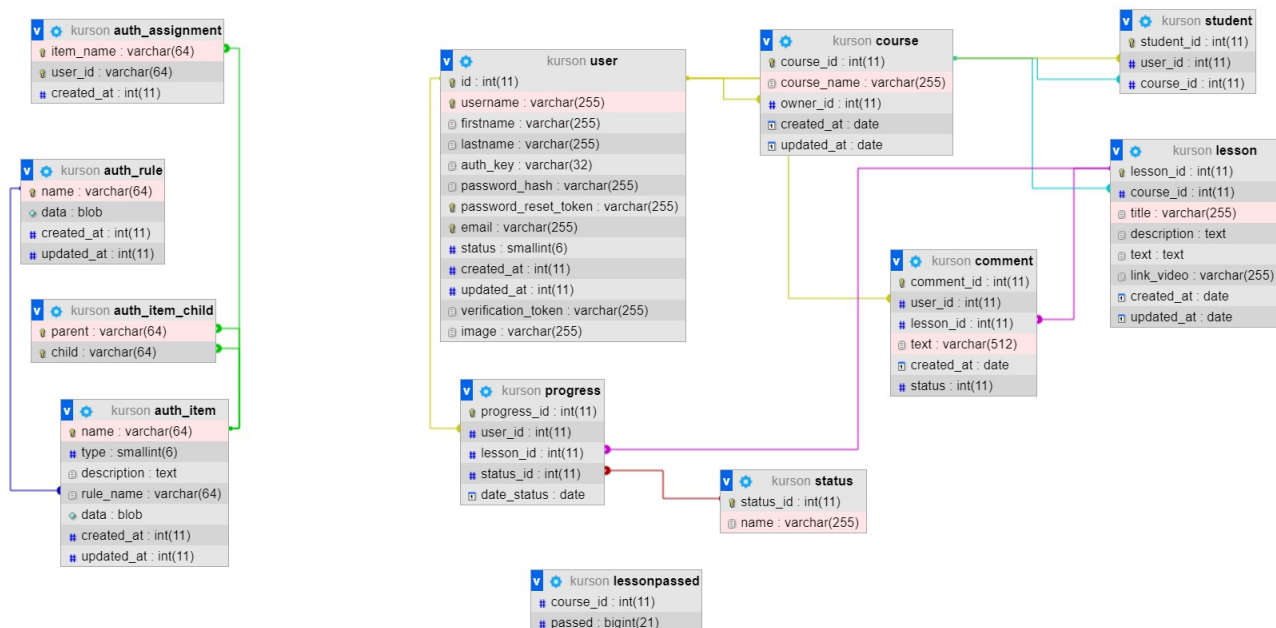


Рис. 2.2 Фізична ERD діаграма

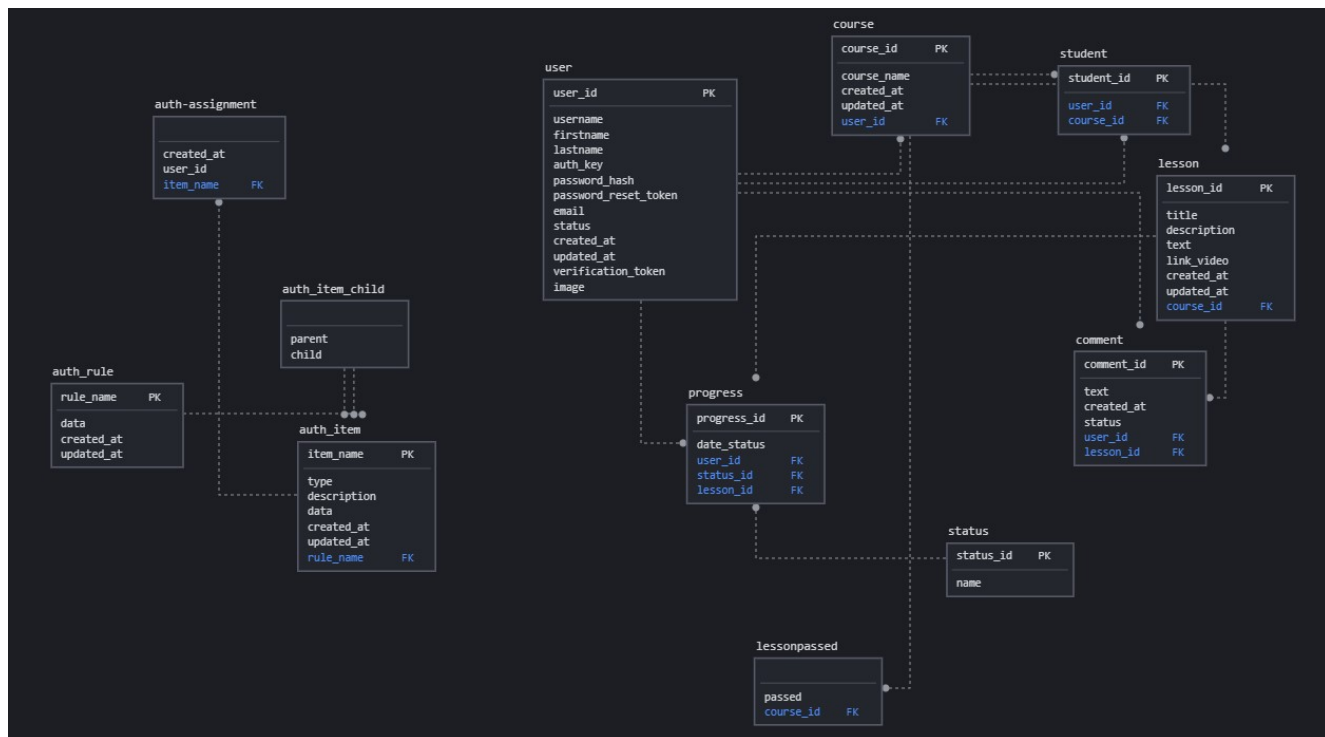


Рис. 2.3 Логічна ERD діаграма

Структура таблиць бази даних:

Таблиця 2.1

Таблиця auth_assignment

Поле	Ключ	Тип	Призначення
item_name	PK	Varchar(64)	Назва ролі
user_id	FK	Varchar(64)	Код користувача
created_at		Int(11)	Дата привласнення ролі користувачу

Таблиця 2.2

Таблиця auth_item

Поле	Ключ	Тип	Призначення
Name	RK	Varchar(64)	Ім'я ролі
Type		Smallint(6)	Тип. Якщо 1 – це роль, якщо 2 0 дозвіл

description		text	Опис ролі чи дозволу
Rule_name	FK	Varchar(64)	Ім'я правила, яке застосовується до даного дозволу чи ролі
Data		blob	Додаткові дані в серіалізованому вигляді
Created_at		Int(11)	Дата створення
Updated_at		Int(11)	Дата оновлення запису

Таблиця 2.3

Таблиця auth_item_child

Поле	Ключ	Тип	Призначення
Parent	FK	Varchar(64)	Ім'я батьківського дозволу чи ролі
Child	FK	Varchar(64)	Ім'я дочірнього дозволу чи ролі

Таблиця 2.4

Таблиця auth_rule

Поле	Ключ	Тип	Призначення
name	PK	Varchar(64)	Ім'я правила
data		blob	Серіалізований об'єкт yii\rbac\Rule
created_at		Int(11)	Дата створення запису
updated_at		Int(11)	Дата оновлення запису

Таблиця 2.5

Таблиця comment

Поле	Ключ	Тип	Призначення
comment_id	PK	Int(11)	Код коментаря
user_id	FK	Int(11)	Код користувача
lesson_id	FK	Int(11)	Код уроку
text		Varchar(512)	Текст коментаря
created_at		date	Дата створення

status		Int(11)	Статус, 1 – активовано, 0 не активний
--------	--	---------	---------------------------------------

Таблиця 2.6

Таблиця course

Поле	Ключ	Тип	Призначення
course_id	PK	Int(11)	Код курсу
course_name		Varchar(255)	Назва курсу
owner_id	FK	Int(11)	Код власника курсу
created_at		date	Дата створення
updated_at		date	Дата останнього оновлення

Таблиця 2.7

Таблиця lesson

Поле	Ключ	Тип	Призначення
lesson_id	PK	Int(11)	Код урок
course_id	FK	Int(11)	Код курсу
title		Varchar(255)	Назва уроку
description		Text	Опис до уроку
text		Text	Текст до уроку
link_video		Varchar(255)	Посилання на відео
created_at		Date	Дата створення
updated_at		Date	Дата оновлення

Таблиця 2.8

Таблиця migration

Поле	Ключ	Тип	Призначення
Version		Varchar(180)	Версія міграції
Apply_time		Int(11)	Дата створення

Таблиця 2.9

Таблиця progress

Поле	Ключ	Тип	Призначення
progress_id	PK	Int(11)	Код прогресу
user_id	FK	Int(11)	Код користувача
lesson_id	FK	Int(11)	Код уроку
status_id	FK	Int(11)	Код статусу
date_status		date	Дата оновлення

Таблиця 2.10

Таблиця status

Поле	Ключ	Тип	Призначення
status_id	PK	Int(11)	Код статусу
name		Varchar(255)	Назва статусу

Таблиця 2.11

Таблиця student

Поле	Ключ	Тип	Призначення
student_id	PK	Int(11)	Код студента
user_id	FK	Int(11)	Код користувача
course_id	FK	Int(11)	Код курсу

Таблиця 2.12

Таблиця user

Поле	Ключ	Тип	Призначення
id	PK	Int(11)	Код користувача
username		Varchar(255)	Нікнейм
firstname		Varchar(255)	Ім'я

lastname		Varchar(255)	Прізвище
auth_key		Varchar(32)	Автентифікаційний код
password_hash		Varchar(255)	Захешований пароль
password_reset_token		Varchar(255)	Пароль для відновлення
email		Varchar(255)	Електронна пошта
status		Smallint(6)	Статус
created_at		Int(11)	Дата створення
updated_at		Int(11)	Дата оновлення
verification_token		Varchar(255)	Верифікаційний токен
image		Varchar(255)	Ім'я картинки з профіля

2.3. Проектування та реалізація алгоритмів роботи системи

У цьому розділі ми вникаємо в розробку і реалізацію алгоритмів роботи системи. Одним з найважливіших візуальних представлень в уніфікованій мові моделювання (UML) є діаграма активності. Діаграма діяльності забезпечує візуальне представлення графіки діяльності. Граф діяльності є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій [14].

Діаграми активності та їхній опис наведені нижче. Необхідно зазначити, що кожна діаграма представлена для вже авторизованих та автентифікованих користувачів. Опис діаграми зображено на рисунку 2.4:

- Для користувача з роллю Автор:
 - Можливість переглядати списки студентів та здійснювати пошук студенту
 - Можливість додавання чи вилучення студенту з курсу
 - Перегляд персональних даних студента
 - Можливість скорегувати інформацію в обліковому записі студента

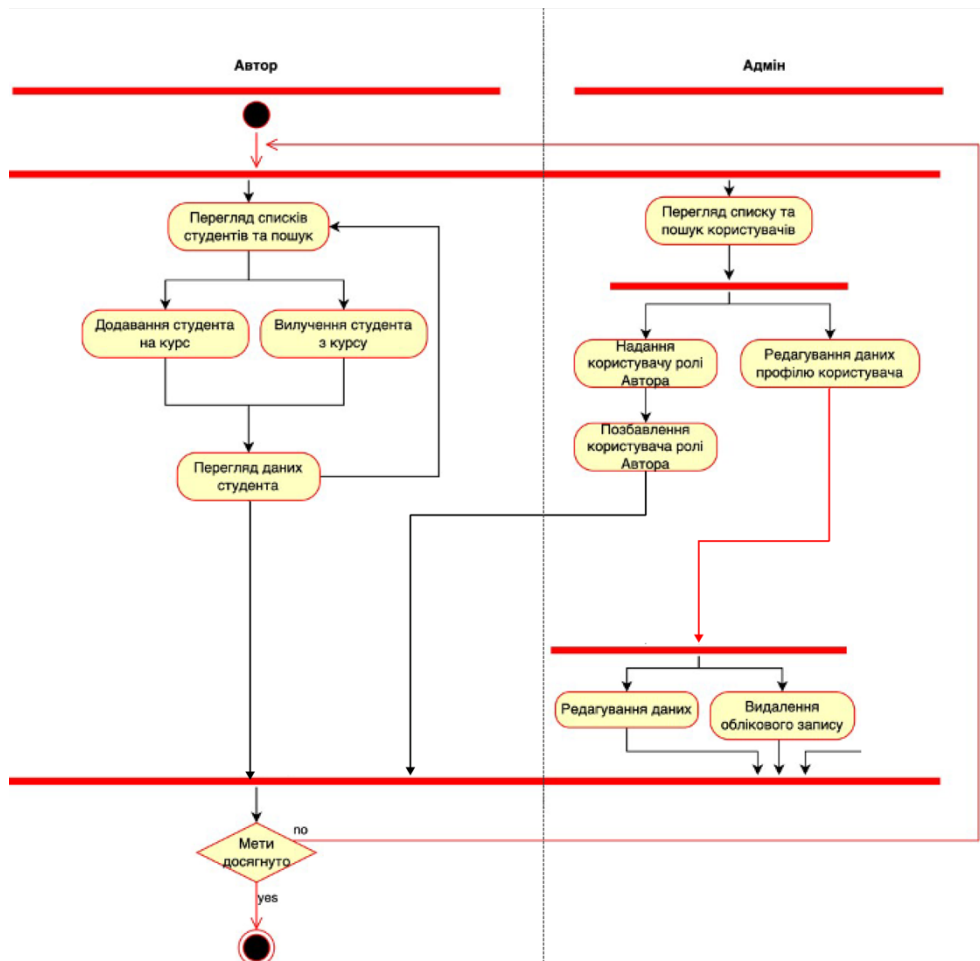


Рис. 2.4 Діаграма дій процесу присвоєння ролей та додавання студенту на курс

Опис діаграми на рис. 2.4 для користувачів з роллю системного Адміністратора:

- Можливість переглядати списки користувачів та здійснювати пошук серед них
- Можливість зміни ролі для користувача (з ролі Студент, яка йде за замовчуванням, змінити на роль Автор, або навпаки)
- Можливість скорегувати інформацію в обліковому записі користувача
- Завершення роботи у разі досягнення мети

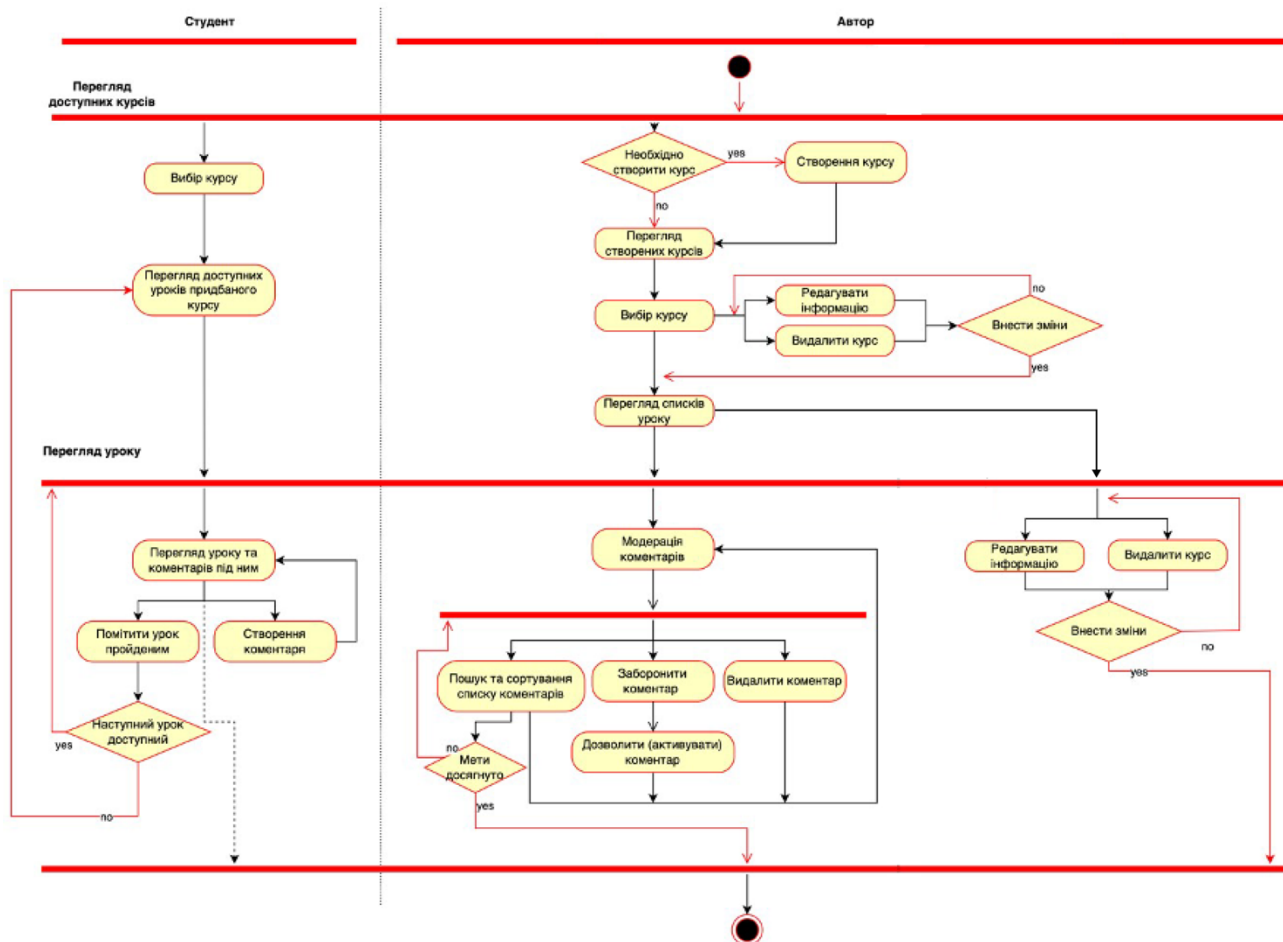


Рис. 2.5 Діаграма дій взаємодії з курсами та їхніми уроками для 2 ролей

Опис діаграми взаємодії з курсами (Рис. 2.5). В момент відкриття сторінки для перегляду доступних курсів:

- Для користувачів з роллю Студент:
 - Обирання курсу
 - Перегляд до нього доступних уроків
- Для користувача з роллю Автор
 - Можливість створення курсу
 - Можливість редагування та видалення курсу після вибору певного курсу
 - Перегляд списку уроків обраного курсу
 - Можливість створення уроку до курсу

В момент відкриття сторінки вибраного уроку:

- Для користувачів з роллю Студент
 - Можливість перегляду наявних коментарів
 - Можливість створення коментаря
 - Можливість відмітити урок як пройденим
 - Перехід до наступного уроку, якщо той наявний
 - Вихід до списку курсів, якщо відсутній наступний урок
- Для користувачів з роллю Модератора
 - Можливість входження, з перегляду уроку, в модерацію коментарів
 - Пошук та сортування усіх наявних коментарів
 - Можливість заборонити чи активувати коментар
 - Можливість видалення коментаря
 - Можливість редагування інформації
 - Можливість видалення уроку
 - Завершення роботи у разі досягнення мети

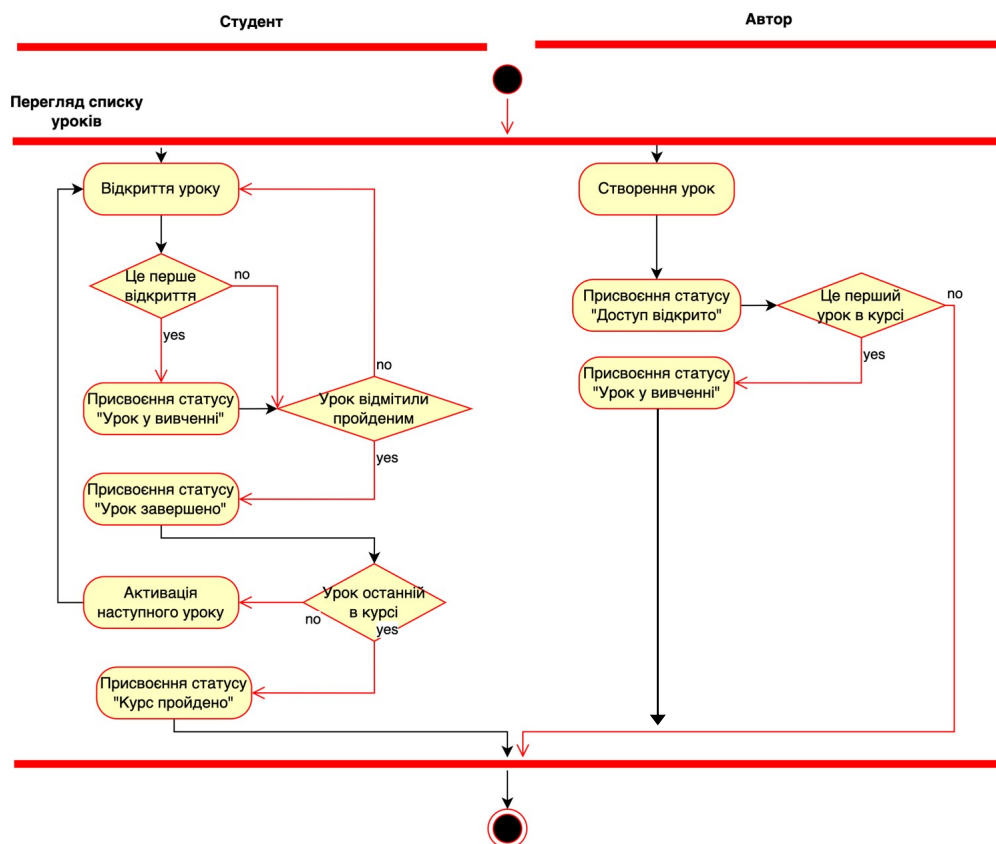


Рис. 2.6 - Діаграма дій зміни статусу уроків в процесі проходження курсу

Опис діаграми дій змін статусу уроків в процесі проходження курсу, яка зображена рисунку 2.6:

- Для користувачів з роллю Студент
 - o Відкриття уроку. Якщо перший раз – присвоєння статусу «Урок у вивченні».
 - o Можливість відмітити урок пройденим – присвоєння статусу «Урок завершено»
 - o Активація наступного уроку, якщо він наявний
 - o Присвоєння статусу «Курс пройдено», якщо уроків вже немає в курсі
- Для користувачів з роллю Автор:
 - o Створення уроку
 - o Автоматичне присвоєння статусу «Доступ відкрито»
 - o Автоматичне присвоєння статусу «Урок у вивченні», якщо це найперший урок в курсі
 - o Завершення роботи у разі досягнення мети

Останній, та не менш важливий, крок для того аби приступити до реалізації спроектованої системи – це розробити діаграму послідовності системи [19]. На даній діаграмі можна побачити, що відбувається в системі поза очима користувача. Коли користувач відкриває вікно з даними, відображення надсилає запит до контролера для отримання даних, контролер в свою чергу надсилає запит до моделі, яка отримує дані з бази даних. Ці дані з бази надходять до відображення пройшовши обробку і відображаються користувачу.

Це загальна схема послідовності роботи будь-якого користувача з системою. Так, наприклад, така послідовність відбувається при редагуванні адміністратором даних автора чи студента. Якщо користувач просто відкриває сторінку для перегляду, то виконується перша частина діаграми послідовностей.

Для даного проекту було розроблено одну загальну діаграму послідовностей, яка представлена на рисунку 2.7.

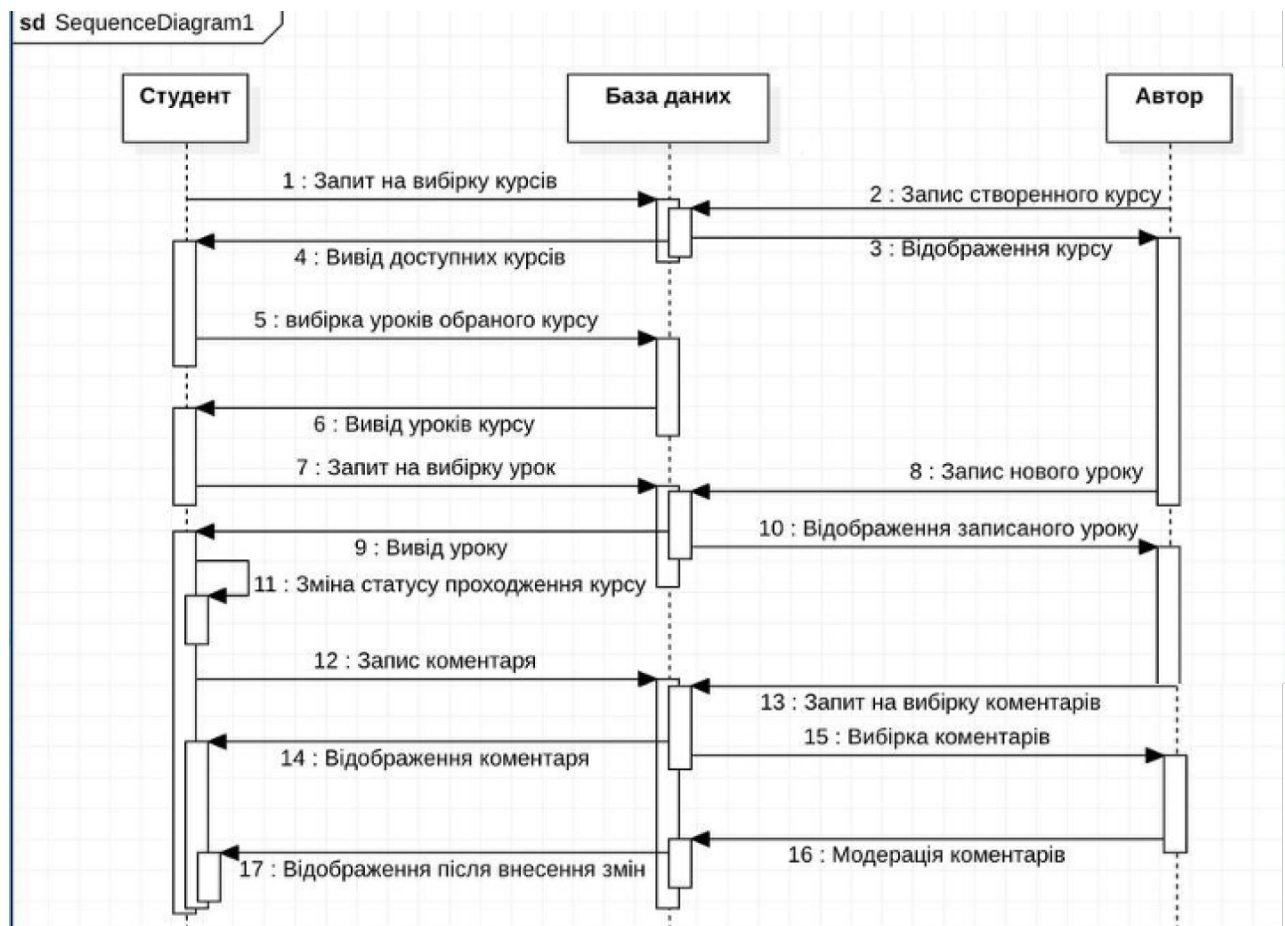


Рис. 2.7 – Діаграма послідовності системи

2.4. Реалізація додатку платформи для розміщення інформаційних продуктів

В даному розділі будуть розглянуті діаграми компонентів, які відображають взаємозв'язки між різними складовими програмного забезпечення, такими як вихідний код, бінарні компоненти та виконувані компоненти. Кожен модуль програмного забезпечення буде представлений у вигляді окремої компоненти.

Деякі компоненти існують під час компіляції, деякі під час компонування, а деякі під час роботи програми. Важливо зазначити, що діаграма компонентів відображає лише структурні характеристики, а для відображення окремих екземплярів компонентів слід використовувати діаграму розгортання.

Компоненти представляють собою фізично існуючі частини системи, які відповідають за реалізацію класів, відносин та функціональної поведінки програмної системи. У даному випадку (див. Рисунок 2.8), компоненти включають:

- MySQL - базу даних

- Site Context - компонент, який відповідає за відображення контексту на веб-сторінці. Контекст залежить від ролей, які будуть наведені нижче.
- Student – тип користувача у системі(студент);
- Moderator – тип користувача у системі(модератор);
- Author – тип користувача у системі(автор);
- Admin – тип користувача у системі(адміністратор);
- Models – компоненти, які є модулями в програмному коді
 - Comment – модель для CRUD коментарів
 - Progress – модель, що створює та відображує статистичні дані успішності;
 - Course - модель для CRUD курсів
 - CourseSearch – модель для пошуку курсів
 - Lesson - модель для CRUD уроку
 - LessonSearch – модель для пошуку уроку
 - User – загальна модель для користувача системи
 - UserSearch – модель для пошуку користувачів
 - Student – модель для створення, редагування та додавання студенту до курсу
 - StudentSearch – модель для пошуку студентів
 - Moderation – модель для модерації коментарів
- Залежності

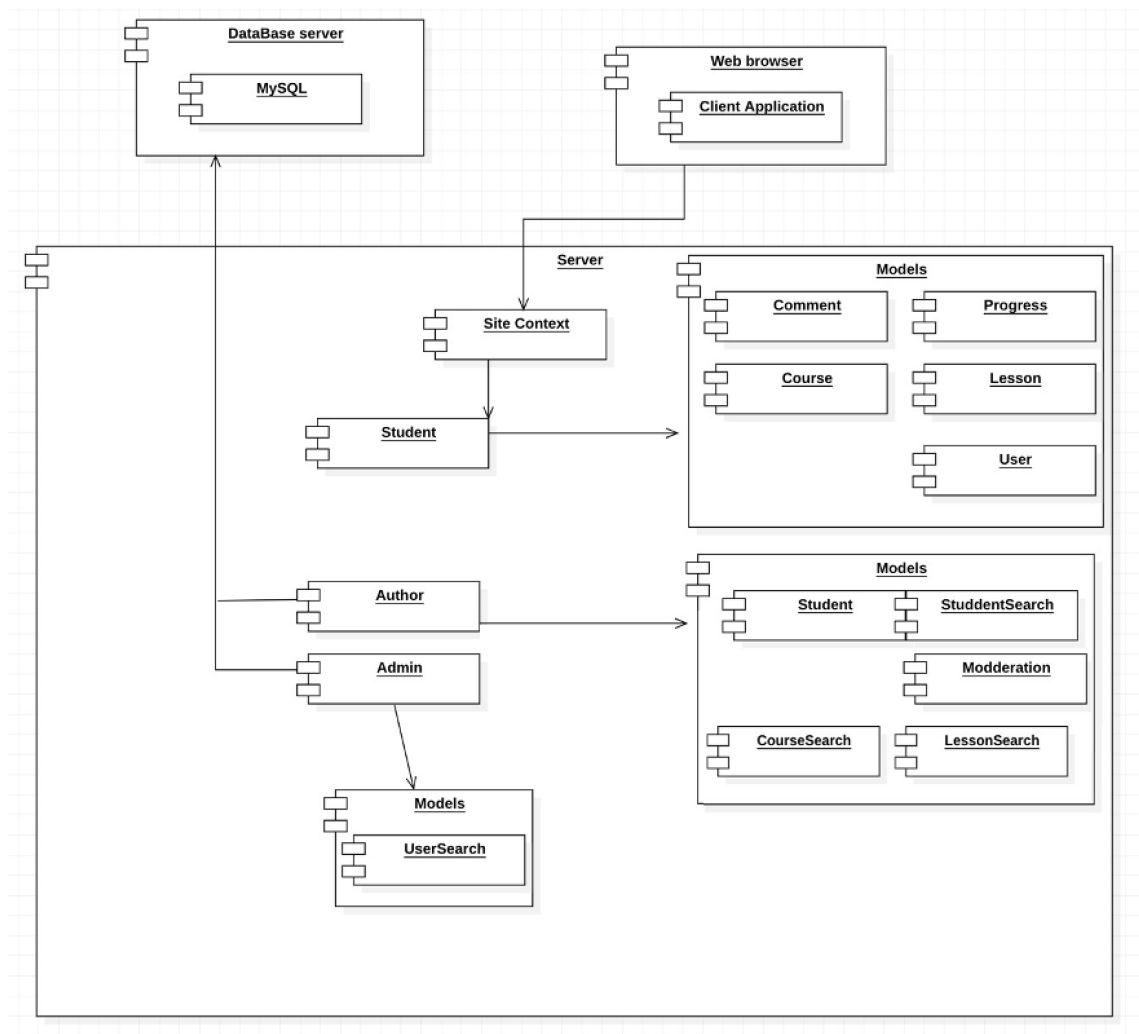


Рис. 2.8 Діаграма компонентів

Для того, щоб додаток функціонував - найголовніше, що потрібно зробити на початку роботи, це підключити програмного додатку до БД [7]. У фреймворку Yii2 підключення відбувається у файлі config -> main-local.php [4].

Лістинг підключення до БД:

```

return [
    'components' => [
        'db' => [
            'class' => 'yii\db\Connection',
            'dsn' => 'mysql:host=localhost;dbname=kurson',
            'username' => 'root',
            'password' => 'root',
            'charset' => 'utf8',
        ],
        'mailer' => [
            'class' => 'yii\swiftmailer\Mailer',
            'viewPath' => '@common/mail',
        ],
    ],
];

```

Реєстрація користувача відбувається в Yii2 за допомогою декількох системних функцій [11]. Спочатку проходить валідація користувача функцією `validate()`, яка в свою чергу визиває ряд системних функцій для перевірок даних. Далі відбувається збереження даних методом `save()` [17] і занесення їх до таблиці `user`. Але для даної моделі користувача, ще визивається автоматично функція `afterSave()`, в якій при реєстрації користувача йому надається роль Студент.

```
public function afterSave($insert, $changedAttributes)
{
    if ($insert) {
        $auth = Yii::$app->authManager;
        $student = $auth->getRole("Student");
        $auth->assign($student, $this->getId());
    }

    parent::afterSave($insert, $changedAttributes); // TODO: Change the
    autogenerated stub
}
```

Щоб відобразити усі списки курсів, на яких навчається учень, було створено наступну функцію, де саме змінна `studyCourse` відповідає за курси, на яких навчається користувач [18]:

```
public function actionIndex()
{
    $searchModel = new CourseSearch();
    $user_id = Yii::$app->user->getId();
    $studentsCount = Student::getStudentsOnCourse($user_id);

    $courses = Course::find()->where(['owner_id' => $user_id])->asArray()->all();
    $studyCourse = Yii::$app->db->createCommand("SELECT * FROM course join student
on student.course_id = course.course_id WHERE student.user_id =:id")-
>bindParam(':id', $user_id)->queryAll();

    $dataProvider = $searchModel->search(Yii::$app->request->queryParams);

    if ($this->request->isPost)
    {
        $sort = $_REQUEST['sort'];
        $model = new Course();
        $courses = $model->sortCourse($sort);
    }

    return $this->render('index', [
        'searchModel' => $searchModel,
        'dataProvider' => $dataProvider,
        'courses' => $courses,
        'studyCourse' => $studyCourse,
    ]);
}
```

Автор курсів має можливість додавати нових студентів на свої курси. Він може додавати як вже зареєстрованих користувачів, так і тих, хто ще не

zareestrovaniy na daniy platformi. Nizhe navedena funkciya z modeli Course, de vidbuvaetsya reestraciya novogo koristuvacha ta zarahuvannya studenta na kurs [17].

```
public function addStudentOnCourse($id_course, $model, $id_user=null)
{
    $date = date("y-m-d");
    $id_user = $model->id;
    $newOne = false;

    if($model->id==null)
    {
        $security = new Security();
        $passwordRandom = $security->generateRandomString(8); // Generate an 8-
character random password
        // zareestrovuvati koristuvacha
        $signupForm = new SignupForm();
        $signupForm->signup($model, $passwordRandom);
        $check_uniq = null;
        $newOne = true;
        $id_user = $this->getStudentId($model->email);
        $sendEmail = $this->emailToNewStudent($model, $passwordRandom);
    }
    else{
        $check_uniq = Yii::$app->db->createCommand("SELECT student_id FROM student
WHERE student.user_id =:id_user AND student.course_id =:id_course")
        ->bindValue(':id_user', $id_user)
        ->bindValue('id_course', $id_course)
        ->queryScalar();
    }
    // додаємо користувача як студента на курс (нового і вже зареєстрованого)

    if (!$check_uniq) {
        $createStudent = Yii::$app->db->createCommand("INSERT INTO `student`
(`student_id`, `user_id`, `course_id`) VALUES (NULL, '{$id_user}',
 '{$id_course}')")->execute();
        $lesson_id = Lesson::getLessons($id_course);
        foreach ($lesson_id as $item){
            $createProgress = Yii::$app->db->createCommand("INSERT INTO `progress`
(`progress_id`, `user_id`, `lesson_id`, `status_id`, `date_status`) VALUES (NULL,
 '{$id_user}', '{$item['lesson_id']}', '1', '{$date}')")->execute();
        }
        if ($lesson_id[0]['lesson_id']) {
            $createProgress = Yii::$app->db->createCommand("INSERT INTO `progress`
(`progress_id`, `user_id`, `lesson_id`, `status_id`, `date_status`) VALUES (NULL,
 '{$id_user}', '{$lesson_id[0]['lesson_id']}', '1', '{$date}')")->execute();
        }
        return true;
    }
    else
    {
        Yii::$app->session->setFlash('danger', "Студент вже зарахований на цей
курс!");
        return false;
    }
}
```

В даному додатку закладена логіка для відстежування успішності учнів. Щоб було зручніше та краще це робити, було створено 4 етапи для відмітки прогресу учня: 1 – доступ відкрито, 2 – урок у вивченні, 3- урок завершено, 4 – курс завершено. Вся інформація про прогрес вивчення фіксується у таблиці progress, де фіксується статус прогресу для кожного з уроків. Учень, в свою чергу, під наданим для нього уроком, може відмічати урок пройденим, а коли це останній урок курсу – як курс вивчений. Перший статус додається до таблички автоматично через код, а 2-4 після взаємодії користувача з інтерфейсом.

Алгоритм надання статусу та зміни в таблиці з прогресом представленні нижче.

Після створення нового уроку до курсу викликається метод, який заносить усіх учнів до таблиці зі статусом 1 для даного уроку:

```
public function setStatusBegin()
{
    $students = Student::getStudentsOnCourse($_SESSION['courseID']);
    $date = date("y-m-d");
    foreach ($students as $student) {
        Yii::$app->db->
            createCommand("INSERT INTO `progress` (`progress_id`, `user_id`,
`lesson_id`, `status_id`, `date_status`) VALUES (NULL, {$student['user_id']},
{$this->lesson_id}, '1', '{$date}');"
            ->execute();
    }
}
```

При відкритті уроку учнем, статус для даного уроку автоматично оновлюється до статусам 2 – урок розпочато. В контролері викликається метод, який вертає статус для даного уроку

Лістинг з контролера Lesson:

```
$progress = $model->isLessonLearned();
if ($progress == 1) {
    $model->setNewStatus(2);
}
```

Лістинг методу isLessonLearned():

```
public function isLessonLearned()
{
    $user_id = Yii::$app->user->getId();
    $result = Yii::$app->db->createCommand("SELECT status.status_id FROM status
JOIN progress on progress.status_id = status.status_id JOIN lesson on
lesson.lesson_id = progress.lesson_id WHERE progress.user_id =:user_id AND
lesson.lesson_id =:lesson_id")
        ->bindValue(':user_id', $user_id)
```

```

        ->bindValue(':lesson_id', $this->lesson_id)
        ->queryScalar();
    return $result;
}

```

Лістинг методу setNewStatus ():

```

public function setNewStatus($status)
{
    $user_id = Yii::$app->user->getId();
    $process_record = Yii::$app->db->createCommand("SELECT * FROM progress JOIN
lesson ON lesson.lesson_id = progress.lesson_id JOIN user ON user.id =
progress.user_id WHERE progress.user_id =:user_id AND progress.lesson_id
=:lesson_id")
        ->bindValue(':user_id', $user_id)
        ->bindValue(':lesson_id', $this->lesson_id)
        ->queryOne();

    if ($process_record){
        $progress = Yii::$app->db->createCommand("UPDATE progress SET
status_id={$status} WHERE progress_id={$process_record['progress_id']}") -
>execute();
    }
}

```

Коли користувач під уроком натискає кнопку «Урок вивчено», в контролері спрацьовує блок обробки кнопки через метод Ajax. В якому викликається попередній метод setNewStatus(), але вже передається статус 3 [8].

Лістинг з контролера Lesson:

```

if (Yii::$app->request->isPjax){
    $progress = 3;
    $model->setNewStatus($progress);
}

```

Коли користувач натискає кнопку «Курс завершено», яка доступна тільки на останньому уроці курсу, та після відмітки, що урок пройдено, спрацьовує наступний код в контролері:

```

public function actionCoursedone($id)
{
    $model = $this->findModel($id);
    $model->setNewStatus(4);
    return $this->redirect(['index', 'id' => $_SESSION['courseID']]);
}

```

Для забезпечення збереження даних з бази даних було реалізовано імпорт та експорт усієї бази даних у файл .sql [2]. Дана можливість функціоналу є у адміністратора. Він у будь який момент може створити дамп БД. Скачаний файл зберігається в проекті в папці common->db [9]. Адміністратор може напряму через інтерфейс імпортувати дамп в БД.

База даних

Створити дамп БД (експорт)

Showing 1-6 of 6 items.

#	Шлях до дампу БД		
1	W:\domains\kursova\common\db\dump_08-06-2021_01-48-06.sql	Імпорт в БД	Видалити
2	W:\domains\kursova\common\db\dump_09-06-2021_21-43-53.sql	Імпорт в БД	Видалити

Рис. 2.9. Вигляд сторінки з імпортом/експортом БД

Також було реалізовано метод для експорту таблиць з БД в Excel файл [16]. Для того, щоб у зручний спосіб можна було переглядати дані та друкувати їх, при необхідності для звітності.

Лістинг контролеру та моделі, що відповідає за наведені вище функції представлені у додатку Б.

Висновки до розділу 2

Платформа для розміщення інформаційних продуктів реалізує повний функціонал для розміщення навчальних курсів, відслідковування прогресу навчання зарахованих на них студентів та здійснення модерації.

Для забезпечення збереження даних з бази даних, для адміністратора, було реалізовано імпорт та експорт усієї бази даних у файл .sql. Також було реалізовано метод для експорту таблиць з БД в Excel файл. Для того, щоб у зручний спосіб можна було переглядати дані та друкувати їх, при необхідності для звітності

Було розглянуто та спроектовано алгоритми взаємодії між ролями, алгоритми дії процесів системи та розроблено діаграму послідовності системи.

В останньому пункті було реалізовано весь запланований та спроектований функціонал в повноцінний працюючий програмний продукт. Представлено методи реалізації основних та важливих функцій системи.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ПЛАТФОРМОЮ ДЛЯ РОЗМІЩЕННЯ ІНФОРМАЦІЙНИХ ПРОДУКТІВ

3.1. Порядок встановлення та налаштування параметрів системи

Для повної взаємодії з платформою для розміщення курсів, система користувача повинна задовольняти вимогам на комп'ютері, що знаходиться в наступній мінімальній комплектації:

- операційна система Windows 7 або вище, Linux 8.10 та вище, MacOS 10.6 Snow Leopard та вище;
- 32-розрядний (x86) або 64-розрядний (x64) процесор із тактовою частотою 1 ГГц або вище;
- 1 ГБ (для 32-розрядних процесорів) або 2 ГБ (для 64-розрядних процесорів) ОЗУ;
- 16 ГБ (для 32-розрядної системи) або 20 ГБ (для 64-розрядної системи) вільного місця на жорсткому диску;

Діаграма розгортання – діаграма, на якій відображаються обчислювальні вузли під час роботи програми, компоненти, та об'єкти, що виконуються на цих вузлах.

На рисунку 3.1. представлена діаграма розгортання системи. Діаграма має наступну структуру:

- User PC – веб-інтерфейс в браузері;
- DataBase server – інтерфейс для взаємодії з базою даних;
- Web Server – веб-сервер, на якому працює програмне забезпечення;
- PHP – проєктований програмний продукт;

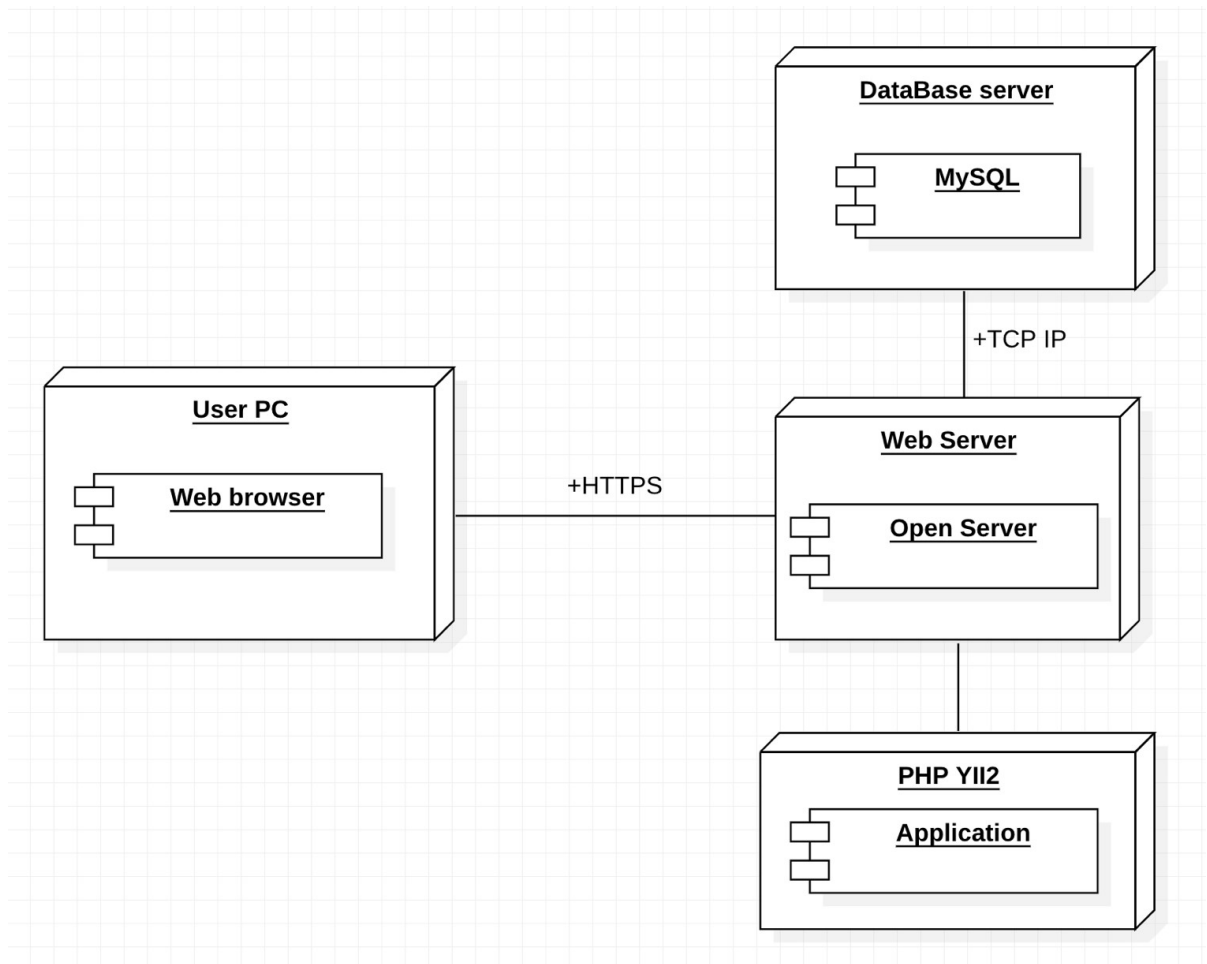


Рис. 3.1. Діаграма розгортання

3.2. Структура інтерфейсу платформи для розміщення інформаційних продуктів

Наш користувач розпочне взаємодію з платформою як Гість системи, через що зможе бачити лише Головну сторінку нашої платформи.



Рис. 3.2. Головна сторінка платформи

Для аутентифікації користувачу необхідно зареєструватися або ж увійти за допомогою свого логіну/електронної пошти та пароллю.

Рис. 3.3. Реєстрація користувача

Якщо наш користувач – Студент, тоді він може переглядати списки курсів, на яких він навчається, відкривати список уроків курсів та для доступного йому уроку детально вивчати матеріал (див. рис. 3.4). У відкритому уроці, під навчальним матеріалом Студент, після вивчення уроку, може відмітити урок пройденим за допомогою відповідної кнопки та залишити коментар до уроку.

Рис. 3.4. Вигляд змісту уроку

Якщо ж наш користувач – Автор, тоді перед ним буде представлено сторінки «Мої курси» та «Уроки до курсу» у схожому вигляді: перелік його курсів/уроків та повний функціонал для їхнього створення/редагування/перегляду/видалення. На рис. 3.5. представлено як приклад вигляд списку курсів.

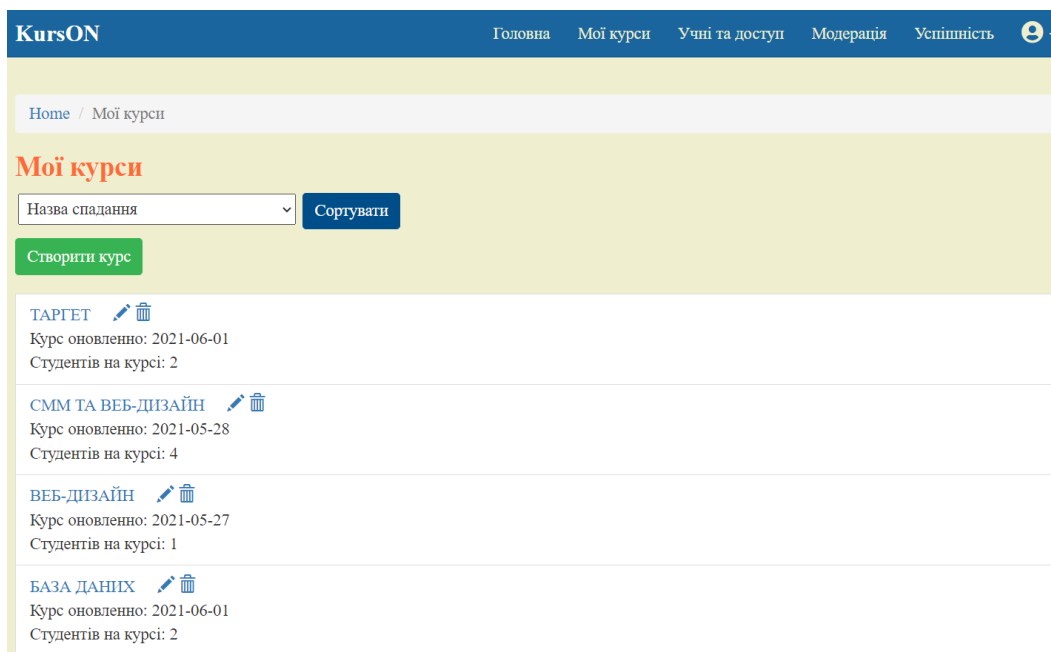


Рис. 3.5. Інтерфейс сторінки з курсами Автора

На сторінці «Учні та доступ» (рис. 3.6.) Автор може переглядати список учнів, які поточно навчаються на його курсах, може шукати необхідного студента за параметрами пошта та курс, а також перейти до додавання нового учня на свій курс. Автор також має можливість вилучати будь-якого студенту з обраного курсу.

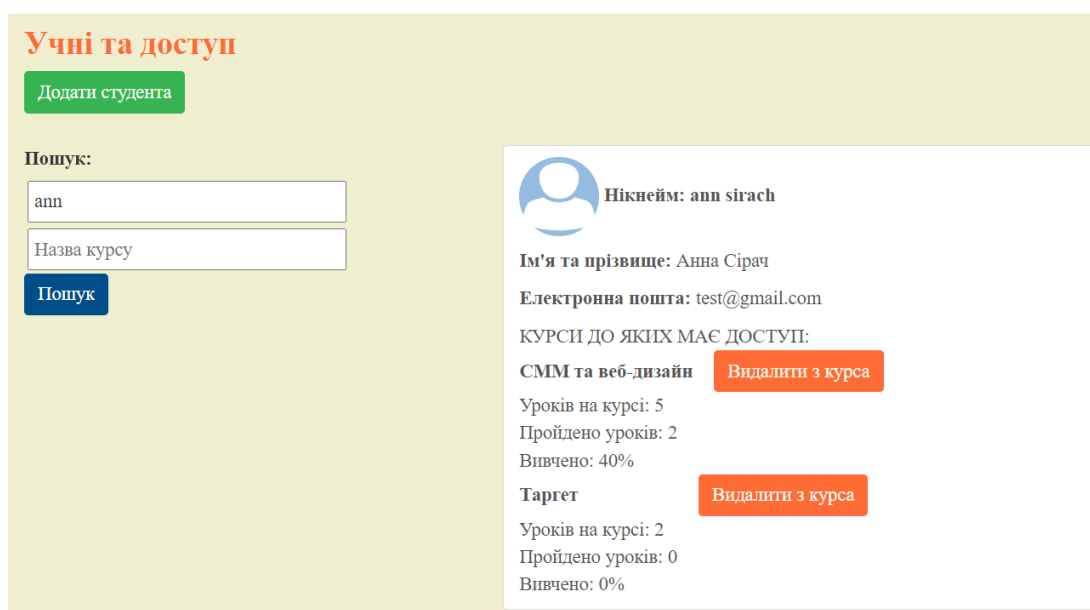


Рис. 3.6. Інтерфейс сторінки «Учні та доступ»

При додаванні студента на курс, Автору буде представлену сторінку з відповідною формою (див. рис. 3.7), де необхідно ввести дані нового студента: ім'я, прізвище, електронна пошта та курс, на який потрібно зарахувати.

Додати користувача

Ім'я

Владислав

Прізвище

Петренко

Електронна пошта

ganskiysl@gmail.com

Course Name

База даних

Відправити

Рис. 3.7. Форма для заповнення для додавання студента на курс

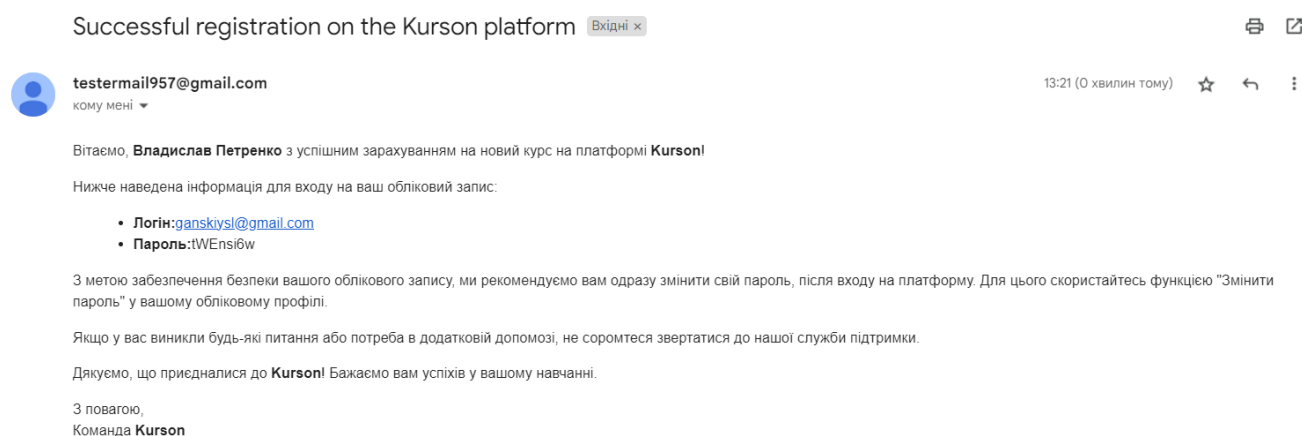


Рис. 3.8. Лист з даними доступу для ново зареєстрованого користувача
Автором

На сторінці «Модерація» Автор може переглядати усі наявні коментарі та здійснювати фільтрацію по ним. Для їхньої модерації – Автор може видаляти обраний коментар, або забороняти його (сховати з загального доступу для студентів).

Модерація						
Урок 1		дд.мм.рррр		дд.мм.рррр		Пошук
Нікнейм		Сортування за датою				
#	Урок	Автор	Текст	Дата створення:	Дії	
28	Урок 1 - Знайомство	ann sirach	Hello my friend	2023-06-02	Заборонити	Видалити
25	Урок 1 - Знайомство	ann sirach	casdc	2023-05-31	Заборонити	Видалити
26	Урок 1 - Знайомство	ann sirach	s fgbxgfs	2023-05-31	Заборонити	Видалити
27	Урок 1 - Знайомство	ann sirach	hibivgiouhihyugyuftrxfvbjnk	2023-05-31	Заборонити	Видалити
15	Урок 1 - Знайомство	crushka12	"Привіт! Дуже цікавий урок. Я зміг легко зрозуміти та застосувати навчальний матеріал. Дякую за інформативну та змістовну лекцію!"	2023-05-25	Заборонити	Видалити
24	Урок 1 - Знайомство	12356	And what we can do now&)	2023-05-25	Заборонити	Видалити
13	Урок 1 - Знайомство	12356	Що робити	2023-05-24	Дозволити	Видалити
14	Урок 1 - Знайомство	malinka_zt	Hello, I've done with this task	2023-05-24	Заборонити	Видалити

Рис. 3.9. Інтерфейс сторінки «Модерація»

Автор також має можливість переглядати статистичні дані по своїм курсам у зручному вигляді діаграм. Для цього йому достатньо перейти до сторінки «Успішність» та проаналізувати представлені йому діаграми.

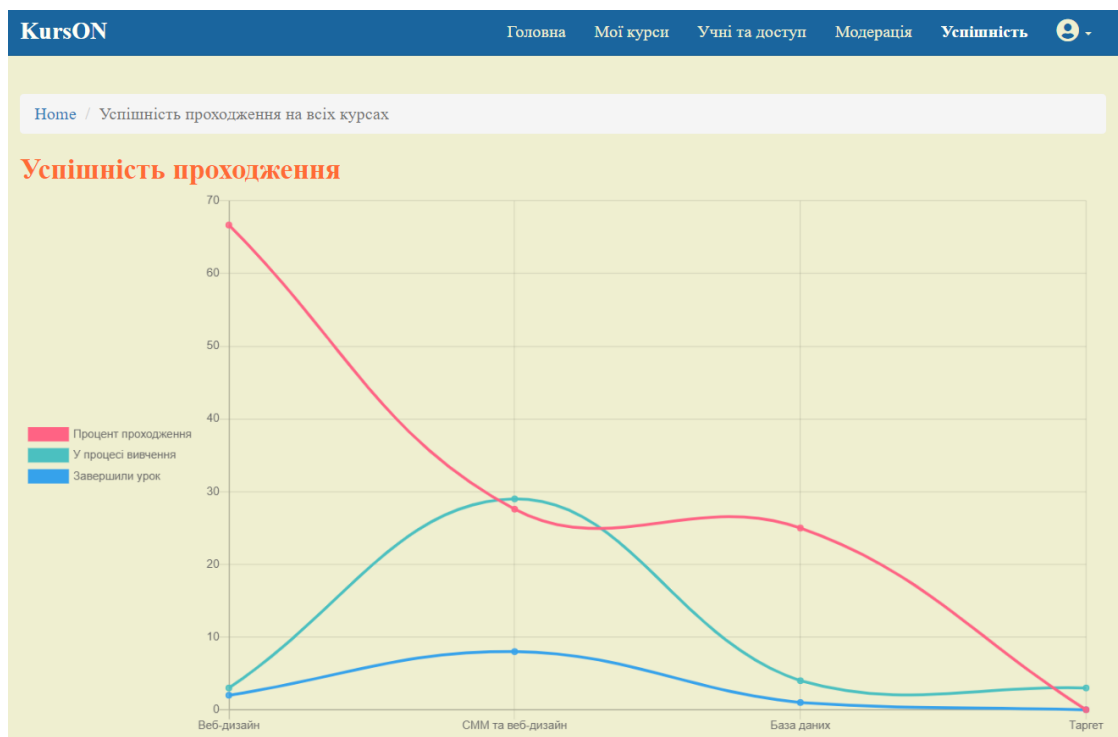


Рис. 3.10. Інтерфейс сторінки «Успішність»

3.3. Тестування роботи програмного комплексу

Успішне тестування програмного продукту є ключовим етапом у процесі його розробки та впровадження. Воно спрямоване на виявлення помилок, перевірку функціональності та визначення його загальної якості та ефективності. В даному розділі буде розглянуто тест-кейси для основних функцій та методів системи, а також результат загального тестування функціоналу.

Зміни статусу уроку з «Урок вивчається» на «Урок вивчено»

Передумови: користувач повинен знаходитись останньому відкритому уроці, який містить статус «Урок вивчається»;

Кроки: натискання кнопки «Відмітити урок пройденим»;

Очікуваний результат: натиснути кнопка зникає з інтерфейсу та статус для уроку змінюється на «Урок вивчено»;

Фактичний результат: натиснути кнопка зникає з інтерфейсу та статус для уроку змінюється на «Урок вивчено».

Заборона коментаря в модерації

Передумови: користувач повинен перебувати на сторінці «Модерація» ;

Кроки: натискання для будь-якого коментаря кнопки «Заборонити»;

Очікуваний результат: зміна натиснутої кнопки на кнопку «Дозволити» та зникнення коментаря у відображенні під уроком, на якому він був залишений;

Фактичний результат: змінилася кнопка на кнопку «Дозволити» та зник коментар під уроком.

Видалення студента з курсу

Передумови: користувач повинен перебувати на сторінці «Учні та доступ»;

Кроки: натискання кнопки «Видалити з курсу» для будь-якого користувача;

Очікуваний результат: курс, з якого було відраховано, повинен зникнути з переліку доступних курсів студента;

Фактичний результат: з переліку доступних курсів, на яких вивчається студент, курс зникнув.

Додавання нового студента, який ще не зареєстрований на платформі

Передумови: перебування користувача на сторінці «Додати студента»;

					ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						45

Кроки: відправити заповнену форму відповідними даними:

- Ім'я: Владислав
- Прізвище: Петренко
- Пошта: ganskiysl@gmail.com
- Зарахувати на курс: Таргет

Очікуваний результат: Студент відобразиться в списку студентів з відповідним курсом, а на його пошту прийде лист з даними для входу;

Фактичний результат: студент відобразився на сторінці зі списком студентів та отримав електронний лист з даними для входу на платформу.

Експорт дампу БД

Передумови: користувач з роллю Адміністратор має перебувати на сторінці «Архівування БД»;

Кроки: натискання кнопки «Створити дамп БД (експорт)»;

Очікуваний результат: створений файл з дампом завантажено в папку для архіву БД та відображено на сторінці новий запис з дампом;

Фактичний результат: створений файл з дампом завантажено в папку для архіву БД та відображено на сторінці новий запис з дампом.

Було також здійснено тестування усього функціоналу системи платформи. Під час даного тестування помилок та чи недоліків в роботі системи не було виявлено. Додаток повністю готовий до користування.

Висновки до розділу 3

В даному розділі було зафіксовано вимоги до комп'ютерної системи для повної взаємодії з платформою для розміщення курсів, а також було спроектовано діаграму розгортання системи, на якій зображені компоненти та об'єкти, що використовуються під час роботи програми.

Після чого було представлено структуру інтерфейсу платформи для розміщення інформаційних продуктів. Для кожної з наявних ролей, було продемонстровано доступний вигляд інтерфейсу та описано можливості функціоналу системи. І не менш важливо – було проведено тестування

програмного комплексу, який дав позитивний результат та розуміння, що програмний продукт готовий до повного користування.

В результаті дій – отримано готовий продукт «Платформа для розміщення інформаційних продуктів», який відповідає усьому раніше зазначеному функціоналу системи.

					ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						47

ВИСНОВКИ

Під час роботи над випускною роботою бакалавра було спроектовано та реалізовано систему платформи для розміщення інформаційних продуктів. У процесі виконання роботи були проведені детальний аналіз обраної тематики, аналіз різних освітніх платформ, що пропонують онлайн-навчання, та визначені основні задачі для майбутньої платформи.

Основним результатом дослідження є вибір веб-орієнтованої архітектури Model-View-Controller (MVC) для розробки платформи. Цей вибір дозволяє ефективно розділити модель даних, інтерфейс користувача та логіку взаємодії, що забезпечує гнучкість, модульність та ремонтпридатність системи.

Також були обґрунтовані вибір інструментів та апаратних вимог для розробки програмного комплексу. Використано Open Server або Xampp, MySQL, PHP та Yii2 фреймворк на сервері, HTML5, CSS та JavaScript на клієнтській стороні.

Були розглянуті та реалізовані основні можливості, включаючи розміщення навчальних курсів, відслідковування прогресу студентів та модерацію. Також було розроблено алгоритми взаємодії між ролями та діаграму послідовності системи, що допомагають управляти процесами та забезпечувати ефективну роботу платформи.

Для забезпечення повної взаємодії з платформою, були встановлені вимоги до системи та розроблена діаграма розгортання системи. Було представлено структуру інтерфейсу платформи для кожної з ролей та проведено тестування програмного комплексу, що підтвердило його готовність до повного використання.

В результаті виконання кваліфікованої роботи був розроблений і реалізований повноцінний програмний продукт - "Платформа для розміщення інформаційних продуктів". Цей продукт відповідає всім визначеним вимогам та функціоналу, що були поставлені перед роботою.

Результати випускної роботи можуть бути використані як основа для подальшого розвитку та вдосконалення освітньої платформи, а також в якості допоміжного матеріалу для вивчення та розробки подібних систем у сфері онлайн-освіти.

					ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						48

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Веллінг Л., Томсон Л. PHP and MySQL Web Development. - Indianapolis: Addison-Wesley Professional, 2016. - 704 с.
2. В'єскас, Дж. Л., Гернандес, М. Дж. SQL Queries for Mere Mortals: A Hands-On Guide to Data Manipulation in SQL. - Indianapolis: Addison-Wesley Professional, 2014. - 792 с.
3. Зандстра М. PHP Objects, Patterns, and Practice. - Birmingham: Apress, 2016. - 704 с.
4. Калдареллі, Ф. Yii2 By Example. - Birmingham: Packt Publishing, 2016. - 332 с.
5. Кек Б. Yii2 for Beginners. - Сан-Франциско: Bill Keck, 2016. - 116 стор.
6. Локхарт Дж. Modern PHP: New Features and Good Practices. - Sebastopol: O'Reilly Media, 2015. - 270 с.
7. Макаров А., Yii2 Application Development Cookbook. - Birmingham: Packt Publishing, 2016. - 400 с.
8. Пауерс Д. PHP Solutions: Dynamic Web Design Made Easy. - Бермонт: Apress, 2014. - 528 стор.
9. Портвуд П, Ц.Р. Mastering Yii. - Birmingham: Packt Publishing, 2016. - 404 с.
10. Сафронов М. Yii Application Development Essentials. - Бірмінгем: Packt Publishing, 2014. - 164 стор.
11. Authorization: Authorization [Електронний ресурс] / yiiframework. – 2022-2023 – Режим доступу до ресурсу:
<https://www.yiiframework.com/doc/guide/2.0/en/security-authorization>
12. Bootstrap Documentation: Components [Електронний ресурс] - 2022-2023. – Режим доступу: <https://getbootstrap.com/docs/5.0/components/>
13. Umbrello. [Електронний ресурс] –
<https://docs.kde.org/stable/uk/kdesdk/umbrello/index.html>
14. UML Activity Diagram Tutorial. [Електронний ресурс]. –
<https://www.lucidchart.com/pages/uml-activity-diagram>

15. StarUML documentation: Introduction. [Електронний ресурс] – <https://docs.staruml.io>
16. SQLZoo: Interactive SQL Tutorial [Електронний ресурс] – Режим доступу до ресурсу: <https://sqlzoo.net/>
17. Робота з базами даних: Active Record [Електронний ресурс] / yiiframework. – 2022-2023 – Режим доступу до ресурсу: <https://www.yiiframework.com/doc/guide/2.0/en/db-active-record>
18. Робота з базами даних: Query Builder [Електронний ресурс] / yiiframework. – 2022-2023 – Режим доступу до ресурсу: <https://www.yiiframework.com/doc/guide/1.1/en/database.query-builder>
19. Уніфікована мова моделювання UML. Загальна характеристика. [Електронний ресурс] – <https://svitppt.com.ua/informatika/unifikovana-mova-modelyuvannya-umlzagalna-harakteristika.html>

ДОДАТКИ

					ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						51

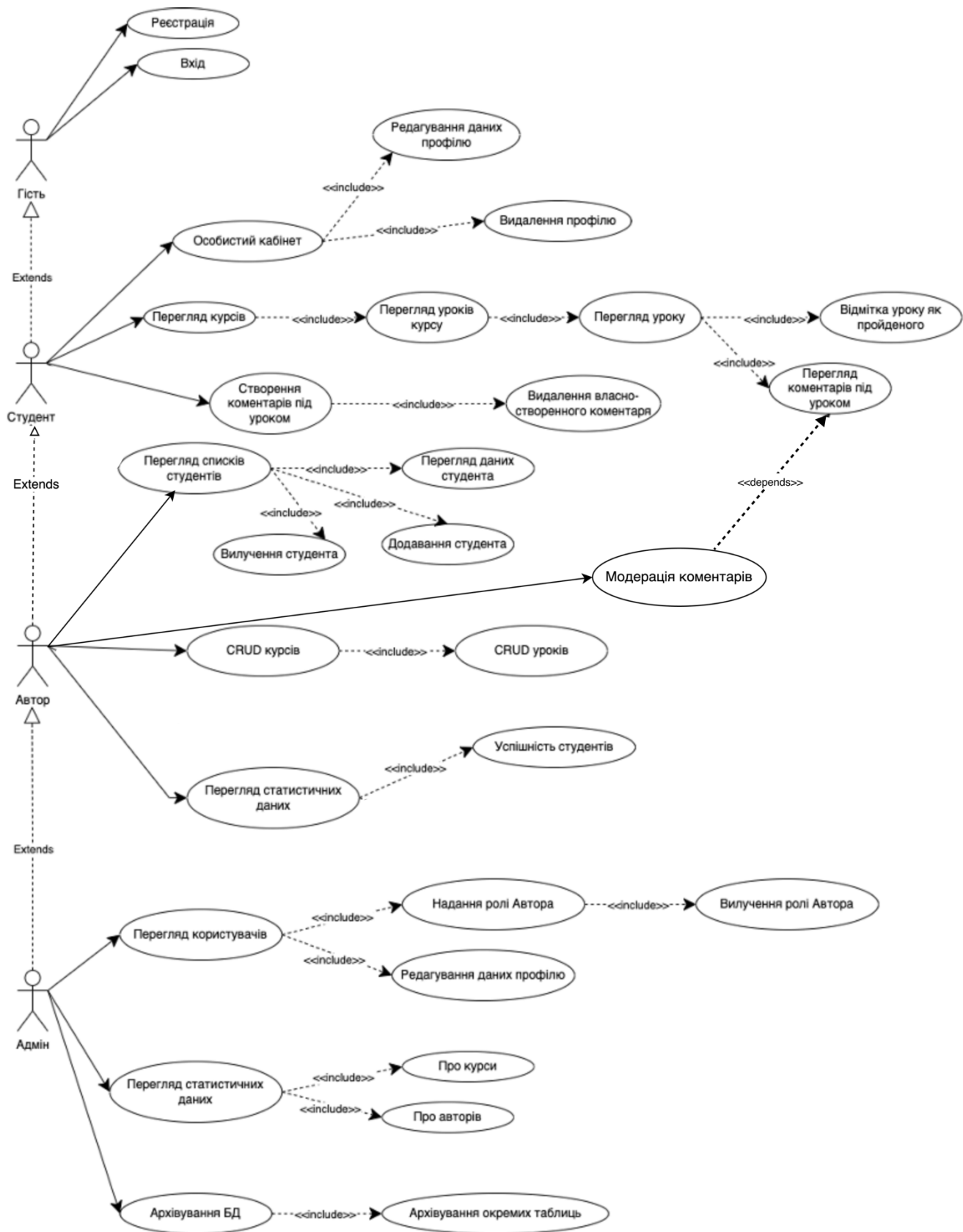


Рис. 1. Варіанти використання системи «Платформа для розміщення інформаційних продуктів»

Лістинг контролера DbController:

```

class DbController extends Controller{

    //Шлях до файлів БД по дефолту
    public $dumpPath = '@common/db/';
    public function actionIndex($path = null){
        //Отримати масив шляхів до файлів з дампом БД (.sql)
        $path = FileHelper::normalizePath(Yii::getAlias($this->dumpPath));
        //'recursive' => FALSE не шукати в підпапках
        $files = FileHelper::findFiles($path, ['only' => ['*.sql'], 'recursive' =>
FALSE]);
        $model = new Db();
        //Метод формує масив в потрібний для GridView формат з пагінацією
        $dataProvider = $model->getFiles($files);
        return $this->render('index', [
            'dataProvider' => $dataProvider,
        ]);
    }
    public function actionImport($path) {
        $model = new Db();
        //Метод видаляє імпорт дампа БД
        $model->import($path);
    }
    public function actionExport($path = null) {
        $path = $path ? : $this->dumpPath;
        $model = new Db();
        //Метод експортує дані з БД у вказану теку
        $model->export($path);
    }
    public function actionDelete($path) {
        $model = new Db();
        //Метод видаляє дампи БД
        $model->delete($path);
    }
    //Доступ лише для адміну
    public function beforeAction($action)
    {
        if (parent::beforeAction($action)) {
            if (!Yii::$app->user->can('Admin')) {
                throw new \yii\web\ForbiddenHttpException('Доступ заборонено.');
            }
            return true;
        } else {
            return false;
        }
    }
}

public function actionCsv()
{
    $tables = Yii::$app->db->schema->getTableNames();

    if (Yii::$app->request->post()) {
        $table = $_REQUEST['table'];
        Db::exportCSV($table);
        return $this->render('csv');
    }

    return $this->render('csv', ['tables'=> $tables]);
}
}

```

Лістинг моделі Db:

```
class Db extends Model{
    public function getFiles($files){
        //кількість файлів, які зберігаються для використання у віджеті
        Yii::$app->params['count_db'] = count($files);
        $arr = array();
        foreach($files as $key => $file){
            $arr[] = array('dump' =>$file);
        }
        $dataProvider = new \yii\data\ArrayDataProvider([
            'allModels' => $arr,
            'sort' => [
                'attributes' => ['dump'],
            ],
            'pagination' => [
                'pageSize' => 10,
            ],
        ]);
        return $dataProvider;
    }
    public function import($path) {
        if (file_exists($path)) {
            $path = \yii\helpers\Html::encode($path);
            $db = Yii::$app->getDb();
            if (!$db) {
                Yii::$app->session->setFlash('error', 'Немає підключення до бази
даних.');
```

```
            }
            //Екрануємо дужку яка є в паролі
            $db->password = str_replace("(", "\(", $db->password);
            exec('mysql --host=' . $this->getDsnAttribute('host', $db->dsn) . ' --
user=' . $db->username . ' --password=' . $db->password . ' ' . $this-
>getDsnAttribute('dbname', $db->dsn) . ' < ' . $path);
            Yii::$app->session->setFlash('success', 'Дамп ' . $path . ' успішно
імпортована.');
```

```
        } else {
            Yii::$app->session->setFlash('error', 'Вказаний шлях не існує.');
```

```
        }
        return Yii::$app->response->redirect(['db/index']);
    }

    public function export($path = null) {
        $path = FileHelper::normalizePath(Yii::getAlias($path));
        if (file_exists($path)) {
            if (is_dir($path)) {
                if (!is_writable($path)) {
                    Yii::$app->session->setFlash('error', 'Директорія не доступна
для запису.');
```

```
                    return Yii::$app->response->redirect(['db/index']);
                }
                $fileName = 'dump_' . date('d-m-Y_H-i-s') . '.sql';
                $filePath = $path . DIRECTORY_SEPARATOR . $fileName;
                $db = Yii::$app->getDb();
                if (!$db) {
                    Yii::$app->session->setFlash('error', 'Немає підключення до
бази даних.');
```

```
                    return Yii::$app->response->redirect(['db/index']);
                }
                $db->password = str_replace("(", "\(", $db->password);
                exec('mysqldump --host=' . $this->getDsnAttribute('host', $db-
>dsn) . ' --user=' . $db->username . ' --password=' . $db->password . ' ' . $this-
>getDsnAttribute('dbname', $db->dsn) . ' --skip-add-locks > ' . $filePath);
```

```

        Yii::$app->session->setFlash('success', 'Експорт успішно
завершено. Файл "'. $fileName.'" в папці ' . $path);
    } else {
        Yii::$app->session->setFlash('error', 'Шлях має бути папкою.');
```

```

    }
} else {
    Yii::$app->session->setFlash('error', 'Вказаний шлях не існує.');
```

```

}
return Yii::$app->response->redirect(['db/index']);
}

private function getDsnAttribute($name, $dsn) {
    if (preg_match('/' . $name . '=(^;*)/', $dsn, $match)) {
        return $match[1];
    } else {
        return null;
    }
}

public function delete($path) {
    if (file_exists($path)) {
        $path = \yii\helpers\Html::encode($path);
        unlink($path);
        Yii::$app->session->setFlash('success', 'Дамп БД видалено.');
```

```

    } else {
        Yii::$app->session->setFlash('error', 'Вказаний шлях не існує.');
```

```

    }
    return Yii::$app->response->redirect(['db/index']);
}

public static function exportCSV($tableName) {
    $connection = Yii::$app->db->pdo;

    $res2 = Yii::$app->db->createCommand("SELECT * FROM {$tableName}")-
>queryAll();

    $sql = "SHOW COLUMNS FROM " . $tableName;
    $stmt = $connection->prepare($sql);
    $stmt->execute([]);
    $columns_table = $stmt->fetchAll();

    $columns = array();
    foreach ($columns_table as $item){
        array_push($columns, $item['Field']);
    }

    $fileName = $tableName . '_' . date('d-m-Y_H-i-s') . '.csv';

    header("Content-type: text/csv");
    header("Content-Disposition: attachment; filename=" . $fileName);
    header("Pragma: no-cache");
    header("Expires: 0");
    $buffer = fopen('php://output', 'a');
    fputs($buffer, chr(0xEF) . chr(0xBB) . chr(0xBF));
    fputcsv($buffer, $columns, ';');
    foreach ($res2 as $item){
        fputcsv($buffer, $item, ';');
    }
    fclose($buffer);

    exit();
}
}

```