**ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Кваліфікаційна робота

на тему: «**СИСТЕМА УПРАВЛІННЯ БАЗОЮ ДАНИХ ДЛЯ СИСТЕМИ ОБЛІКУ НАУКОВО-ПЕДАГОГІЧНИХ ПРАЦІВНИКІВ»**

на здобуття освітнього ступеня магістра
зі спеціальності __122 Комп'ютерні науки_____
(код, найменування спеціальності)
освітньо-професійної програми __Комп'ютерні науки _____
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____Михайло КРЕЩАНОВ_____
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КНДМ-61__

_____Михайло КРЕЩАНОВ_____
(Ім'я, ПРІЗВИЩЕ)

Керівник: д.т.н., професор Віктор ВИШНІВСЬКИЙ
Науковий ступінь, вчене звання *(Ім'я, ПРІЗВИЩЕ)*

Рецензент: _____
Науковий ступінь, вчене звання *(Ім'я, ПРІЗВИЩЕ)*

Київ – 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально–науковий інститут інформаційних технологій

Кафедра Комп'ютерних наук
Ступінь вищої освіти − «Магістр»
Спеціальність підготовки − «122 Комп'ютерні науки»
Освітньо-професійна програма − «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Комп'ютерних наук
_____Віктор Вишнівський
"____" _____2024 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Крещанов Михайло Олександрович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: «Система управління базою даних для системи обліку науково-педагогічних працівників»
Керівник кваліфікаційної роботи _Віктор ВИШНІВСЬКИЙ, д.т.н., професор ,
(ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу № 320 від 15.10.2024р.

2. Строк подання студентом роботи 15.12.2024 р.
3. Вихідні дані до роботи: Теоретичні та статистичні дані щодо електронних систем обліку в галузі освіти. Типологія СУБД, архітектура та технологічні можливості MySQL. Програмні технології PHP, JavaScript, Microsoft Visual Studio Code, HTTP-сервер Apache. Науково-технічна література з питань, пов'язаних з дослідженням ефективності моделей систем управління базами даних для розробки системи обліку педагогічних та науково-педагогічних працівників.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Сучасні рішення щодо систем електронного керування даними
2. Дослідження електронних систем обліку в галузі освіти
3. Дослідження ефективних рішень систем управління базами даних
4. Дослідження технологій для проектування та розробки системи обліку науково-педагогічних працівників

5. Перелік ілюстративного матеріалу: *презентація*

1. Мета роботи, об`єкт та предмет дослідження. Апробація результатів та публікації
2. Системи електронного керування даними
3. Бази даних у сфері освіти
4. Дослідження електронних систем обліку, цифровізації та уніфікації освітніх даних
5. Дослідження функціональних можливостей та класифікації систем управління базами даних
6. Аналіз ієрархічної моделі СУБД
7. Структура мережевої моделі СУБД
8. Дослідження функціональних і технологічних рішень реляційних баз даних
9. Сутність об'єктно-орієнтованої моделі БД
10. Технології об'єктно-реляційних баз даних
11. Методологія дослідження SQL і NoSQL баз даних
12. Дослідження популярних рішень СУБД для розробки програмного продукту
13. Інструментальні засоби розробки програмного забезпечення
14. Створення бази даних для системи обліку науково-педагогічних працівників
15. Реалізація розробки системи обліку науково-педагогічних працівників
16. Висновки

6. Дата видачі завдання 05.09.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	07.10 – 15.10	вик.
2	Сучасні рішення щодо систем електронного керування даними	16.10 – 21.10	вик.
3	Варіанти та технологічні вимоги до електронних систем обліку в галузі освіти	22.10 – 27.10	вик.
4	Дослідження показників продуктивності СУБД	28.10 – 10.11	вик.
5	Вибір технологій для проектування та розробка системи обліку відповідно до критеріїв	11.11 – 03.12	вик.
6	Вступ, висновки, реферат	04.12 – 10.12	вик.
7	Розробка демонстраційних матеріалів	11.12 – 13.12	вик.

Здобувач вищої освіти _____ Михайло КРЕЩАНОВ
(підпис) (Ім'я, ПРІЗВИЩЕ)

Керівник кваліфікаційної роботи _____ Віктор ВИШНІВСЬКИЙ
(підпис) (Ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 87 стор., 15 табл., 29 рис., 46 джерел.

Мета роботи – дослідити методи прийняття ефективних рішень та надати рекомендації щодо застосування сучасних моделей систем керування базами даних в процесі розробки та оптимізації функціональних можливостей системи обліку педагогічних та науково-педагогічних працівників.

Об`єкт дослідження – уніфікована система обліку педагогічних та науково-педагогічних працівників як важливий інтеграційний елемент цифрової трансформації процесів управління та моніторингу в галузі освіти.

Предмет дослідження – моделі систем керування базами даних та засоби підвищення продуктивності системи обліку педагогічних та науково-педагогічних працівників.

Короткий зміст роботи:

Створення спеціальних програмних додатків та інтегрованих систем для корпоративної взаємодії та обліку є ефективним засобом організації електронного середовища в закладах освіти. Наразі існує декілька інноваційних проєктів на загальнодержавному рівні щодо уніфікації даних в освітній галузі. Деякі з них досі перебувають на стадії тестування. Функціональні можливості і доступ до наявних автоматизованих систем мають певні обмеження, а уніфікований облік педагогічних та науково-педагогічних працівників відсутній. Також існує проблема продуктивності подібних інформаційних систем, яка найчастіше пов’язана з базами даних. Для успішної розробки програмного продукту необхідно обрати відповідну систему керування базами даних.

У зв’язку з цим було проведено ґрунтовне дослідження щодо застосування ефективної моделі системи керування базами даних в процесі розробки і оптимізації функціональних можливостей системи обліку науково-педагогічних працівників для закладів освіти різних рівнів акредитації.

КЛЮЧОВІ СЛОВА: СУБД, ECM, SQL/NoSQL БД, MySQL, PHP, JavaScript, Microsoft Visual Studio Code, HTTP-сервер Apache.

ABSTRACT

Text part of the Master's qualification work: 87 pages, 15 tables, 29 pictures, 46 sources.

The purpose of the work – to investigate methods of making effective decisions and provide recommendations for the application of modern models of database management systems in the process of developing and optimizing the functionality of the accounting system of scientific and pedagogical employees.

Object of research – a unified system for accounting of scientific and pedagogical employees as an important integration element of the digital transformation of management and monitoring processes in the field of education.

Subject of research – models of database management systems and means of increasing the productivity of the accounting system of scientific and pedagogical employees.

Summary of the work:

The creation of special software applications and integrated systems for corporate interaction and accounting is an effective means of organizing an electronic environment in educational institutions. Currently, there are several innovative projects at the national level regarding the unification of data in the educational sector. Some of them are still at the testing stage. Functionality and access to existing automated systems have certain limitations, and there is no unified accounting of teaching and scientific and pedagogical employees. There is also a problem of productivity of such information systems, which is most often associated with databases. For successful development of a software product, it is necessary to choose an appropriate database management system.

In this regard, a thorough study was conducted for the application of an effective database management system model in the process of developing and optimizing the functionality of the system for accounting of scientific and pedagogical employees for educational institutions of different accreditation levels.

KEYWORDS: DBMS, ECM, SQL/NoSQL DB, MySQL, PHP, JavaScript, Microsoft Visual Studio Code, Apache HTTP Server.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД	база даних
СУБД	система управління базами даних
ECM	Enterprise Content Management –система управління корпоративним контентом (УКК)
DSM	Document Management System – система електронного документообігу (СЕД)
IIM	Intelligent Information Management – інтелектуальне управління інформацією
ERIC	Education Resources Information Center – інформаційний центр освітніх ресурсів
SQL	Structured Query Language – мова структурованих запитів для взаємодії користувача з базами даних
T-SQL	Transact-SQL – процедурне розширення мови SQL для Microsoft SQL Server
SQL DB	реляційна модель бази даних
NoSQL	Not Only SQL / non SQL / non relational – нереляційна модель баз даних
ObQL	Object Query Language – мова запитів для об’єктних даних
ACID	Atomicity, Consistency, Isolation, Durability (Атомарність, Узгодженість, Ізольованість, Довговічність) – стандарт властивостей для надійної роботи транзакцій бази даних
MVCC	Multiversion Concurrency Control – відповідність ACID
API	Application Programming Interface – інтерфейс програмування додатків
BSON	Binary JSON (Binary JavaScript Object Notation) – комп'ютерний формат обміну даними
HTML	HyperText Markup Language – стандартизована мова розмітки документів

ВСТУП

В інформаційному суспільстві надзвичайно зросла роль віртуальних комунікацій, змінилася й сама технологія отримання інформації та роботи з нею. Інформаційні ресурси, які є основою майже всіх управлінських, економічних та соціальних процесів, набувають домінуючого значення.

Інформаційні ресурси визначають як сукупність документів і масивів документів в інформаційних середовищах (бібліотеках, архівах, фондах, банках даних, депозитаріях, музейних сховищах тощо) чи інформаційних продуктів певного призначення, що необхідні для забезпечення інформаційних потреб споживачів у визначеній сфері діяльності [23, с. 140].

Світовий інформаційний простір розширюється, тож освітня галузь має адаптуватися до вимог сьогодення. Окрім забезпечення потреб державного і бізнес рівня, створення спеціальних програмних додатків та інтегрованих систем для корпоративної взаємодії та обліку є ефективним засобом організації електронного середовища в закладах освіти. Це має позитивно вплинути на інтеграційні процеси та забезпечити якість освітніх послуг в цілому.

Відповідно до Плану відновлення Освіта і наука передбачено практичне вирішення питань, що стосуються «Цифрової трансформації процесів управління, регулювання та моніторингу в ЗВО та ефективного використання цифрових (дистанційних) технологій в освітньому процесі» (термін: січень 2023 року – грудень 2025 року) [10].

Одним з важливих аспектів реалізації цих завдань є створення систем електронного обліку педагогічних та науково-педагогічних працівників навчальних закладів різних рівнів акредитації (середньої, фахової передвищої та вищої освіти). Уніфікація відповідних даних, технологічне забезпечення доступу до інформаційних ресурсів, запровадження міжнародних стандартів для забезпечення пошуку, збору, збереження і використання інформації має сприяти

інтеграційним процесам і збільшити національну присутність у світовому інформаційному просторі [25, с. 44-45].

Але будь-який програмний продукт, система обліку, інформаційна мережа, Інтернет-ресурс потребують ефективного управління інформацією, що передбачає зберігання, пошук та електронне керування даними. Дані є одним із найважливіших ресурсів, які потрібно відстежувати, щоб забезпечити продуктивність програмного продукту. Кількісне збільшення інформаційних потоків й зростання вимог щодо швидкості обробки даних спонукає шукати ефективні рішення керування базами даних в роботі з різними інформаційними системами, наприклад, для реалізації проєкту розробки та запровадження уніфікованої системами обліку педагогічних та науково-педагогічних працівників.

Актуальність теми дипломної роботи ґрунтується перш за все на вирішенні завдань, що стосуються проблем забезпечення продуктивності облікової системи науково-педагогічних працівників в процесі її розробки та дослідження можливих алгоритмів застосування систем керування базами даних. Деталізований аналіз моделей і технології керування базами даних дозволяє у повній мірі реалізувати визначені програмні завдання.

Питання щодо використання інформаційних ресурсів та електронних систем відкритого доступу в освітній галузі розглядали Спірін О. М., Яцишин А. В., Іванова С. М., Кільченко А. В., Лупаренко Л. А., Швець С. М., Пітерс М. А., Брітез Р. Г. На сьогодні ці дослідження не втратили своєї актуальності і потребують ґрунтовного вивчення.

Основні теоретичні та практичні аспекти, що стосуються сфери ефективного керування базами даних висвітлено в наукових публікаціях Давиденка Є. О., Ковтуна Б. В., Манича А. М., Романюка О. В., Мазурової О. О., Набокої А. О., Широкопетлєвої М. С., Фісуна М. Т., Хердера Т; Ройтера А., Кемпбелла Л., Мейджорса Ч., Харрінгтона Я. Л. та інших. Проблематика здійснених досліджень важлива з точки зору практичного застосування в сфері Data science.

Метою даної роботи є розгляд оптимальних рішень щодо застосування ефективної моделі системи керування базами даних в процесі розробки та оптимізації функціональних можливостей системи обліку педагогічних та науково-педагогічних працівників.

Визначені наступні основні *завдання дослідження*:

– проаналізувати нормативно-правові засади та окреслити функції наявних уніфікованих систем обліку в галузі освіти, інших інформаційних систем відповідно до завдань і напрямів освітньої діяльності;

– провести порівняльний аналіз, вивчити структуру і зміст систем обліку педагогічних та науково-педагогічних працівників, визначити комплекс необхідних даних, інформаційних ресурсів, електронних документів, інших складових зазначених систем з метою визначення їх оптимальних функціональних можливостей;

– розглянути алгоритми застосування СУБД для підвищення продуктивності програмного продукту на основі детального аналізу різних моделей систем керування базами даних;

– визначити програмні засоби та алгоритм дій для успішної реалізації проєкту.

Об'єктом дослідження є уніфікована система обліку педагогічних та науково-педагогічних працівників як важливий інтеграційний елемент цифрової трансформації процесів управління та моніторингу в галузі освіти.

Предмет дослідження – вибір моделі системи керування базами даних та засобів підвищення продуктивності системи обліку педагогічних та науково-педагогічних працівників для оптимізація її функціональних можливостей.

Для розв'язання визначених завдань дипломної роботи були застосовані наступні *методи дослідження*:

– загальнонаукові для наукового пізнання (системний, структурний, функціональний, порівняльний);

– інформаційного аналізу для дослідження структурних елементів та функціональних можливостей уніфікованих електронних систем обліку з метою

їх порівняння, визначення існуючих проблем і методів прийняття ефективних рішень в процесі розробки нового програмного продукту;

– вимірювання і моделювання для комплексної оцінки якісних характеристик програмних технологій та створення оптимальних сценаріїв реалізації завдань.

Наукова новизна проведених досліджень пов'язана з вирішенням проблеми щодо систематизації, стандартизації та подальшої уніфікації даних педагогічних та науково-педагогічних працівників. Створення спеціальних програмних додатків та інтегрованих систем для корпоративної взаємодії та обліку є ефективним засобом організації електронного середовища в закладах освіти. Хоча в Україні на загальнодержавному рівні вже існують декілька тестових проєктів щодо уніфікації даних в системі освіти, наразі немає універсальної (стандартизованої) системи обліку педагогічних та науково-педагогічних працівників для широкого запровадження в освітніх закладах. В кваліфікаційній роботі пропонуються ефективні рішення щодо надійної системи керування базами даних для розробки універсальної системи обліку педагогічних та науково-педагогічних працівників. Вирішення проблем оптимізації функціональних можливостей електронної системи обліку доводить практичну значущість отриманих результатів.

Апробація результатів та публікації. За результатами проведених досліджень було підготовлено тези доповіді на тему «Дослідження рішень систем керування базами даних в процесі розробки системи обліку науково-педагогічних працівників» / «Research of solutions of database management systems in the process of developing the accounting system of scientific and pedagogical employees» для участі у II Всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу» (ДУІКТ), а також наукову статтю «Сучасні тенденції ефективного керування даними в умовах постійного розвитку інформаційного середовища» для друку у фаховому науковому виданні SCIENTIA (матеріали VII Міжнародної науково-теоретичної конференції

«Advanced discoveries of modern science: experience, approaches and innovations», 29 листопада 2024 р., Амстердам, Нідерланди).

Кваліфікаційна робота має як *прикладне*, так і *теоретико-методичне значення*, оскільки, по-перше, реалізує практичне завдання, що стосується запровадження систем обліку педагогічних та науково-педагогічних працівників для збору, зберігання та використання даних (формування загальних облікових даних, особових карток працівників, електронних трудових книжок, звітної документації тощо); по-друге, на основі проведеного аналізу ефективності СУБД, узагальнює методи вирішення проблем продуктивності програмних продуктів та роботи з базами даних.

В першому розділі кваліфікаційної роботи розглядаються нормативно-правові засади створення уніфікованих електронних систем обліку даних та систем електронного документообігу у сфері освіти. Також досліджуються наявні в Україні системи обліку даних з питань освіти, які мають сприяти формуванню інтегрованого інформаційного середовища.

У другому розділі аналізуються функціональні можливості та визначається типологія СУБД, розглядаються сучасні моделі систем керування базами даних та окреслюються проблеми продуктивності програмних продуктів, досліджуються різноманітні технології для ефективного керування базами даних.

У третьому розділі обґрунтовані запропоновані рішення щодо реалізації проєкту, представлені програмні технології (стеки), які були використані в процесі розробки системи обліку науково-педагогічних працівників, на основі отриманих результатів доводиться ефективність використаної моделі СУБД.

Аналіз сучасних технологій керування базами даних показав необхідність подальших досліджень у цій галузі для вирішення проблем продуктивності, масштабування, зберігання та обробки даних. Тож, вибір найкращого рішення для застосування системи керування базами даних є першочерговим в процесі розробки програмного продукту, а саме уніфікованої системи обліку науково-педагогічних працівників з подальшим тестуванням і запровадженням.

1 ПРЕДМЕТНА ОБЛАСТЬ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Системи електронного керування даними. Управління корпоративним контентом (ECM)

Серед основних термінів та понять, які визначені в «Методичних рекомендації щодо підвищення рівня кіберзахисту систем електронного документообігу», є системи електронного документообігу (СЕД) (Document management system, DSM) і системи управління корпоративним контентом (УКК) (Enterprise Content Management, ECM) [6]. Останні є інтегрованими системами управління інформацією на рівні підприємства (організації), або системами електронного керування даними та вебконтентом.

Систему ECM розглядають як програмне рішення для ефективної групової роботи та управління документами, що забезпечує організоване використання наявної інформації. Однією з важливих особливостей системи ECM є те, що традиційні функції керування даними (робота з архівами, керування документами, документообіг) поєднуються з веб-компонентами. Також система ECM дозволяє керувати інформацією різного змісту і типу, структурованою та неструктурованою. ECM використовує різноманітні алгоритми підтримки керування даними, забезпечує автоматичну класифікацію інформації, профілювання та запис веб-операцій [27].

ECM система надає можливості щодо:

- інтеграції з іншими системами і додатками;
- підтримки функціонування багатокористувацьких систем;
- використання незалежних сервісів на рівні будь-якої корпоративної програми, що спрощує її технічну підтримку;
- функціонування як додаткової окремої платформи;
- зберігання всіх типів інформації в центральному сховищі з єдиною структурою.

Функціональність ECM системи визначається її базовими модулями, а саме:

- DM (Document Management) – додаток для управління документами;
- Collaboration – інструменти групової співпраці;
- WCM (Web Content Management) – програмний засіб управління вмістом сайту;
- RM (Records Management) – додаток, що забезпечує архівацію даних;
- Workflow / BPM – електронне керування документообігом та бізнес-процесами [27].

Системи управління корпоративним контентом (ECM) адаптовані для складних багаторівневих корпоративних бізнес-структур.

Останнім часом з'явився новий термін IIM (Intelligent information management), або «інтелектуальне управління інформацією». IIM системи – це покращені ECM системи, що пропонують якісний інформаційний інструментарій, високий рівень цифровізації та автоматизації основних організаційних процесів, використання аналітичних програм і машинного навчання для забезпечення ефективного управління.

У сучасному суспільстві системи електронного керування даними та управління корпоративним контентом завдяки своїм можливостям все частіше розглядаються не лише як необхідні складові елементи бізнес-структур, але й як альтернативні рішення для покращення корпоративної взаємодії в будь-яких установах, організаціях, навіть некомерційних.

Подібні IT-системи потрібні також в освітній галузі для цифровізації наявної інформації і керування даними. Електронні автоматизовані системи управління освітою мають забезпечити вирішення багатьох питань, що стосуються не лише процесів ефективного електронного документообігу для покращення взаємодії та звітності, але й системного керування ключовими процесами і даними у сфері освіти.

1.2 Бази даних у сфері освіти

Бази даних в освітній галузі є необхідними та перспективними інструментами для ефективного управління і використання величезних обсягів інформації. Вони формують майбутні перспективи і мають ключове значення в трансформаційних процесах у цій галузі.

Бази даних є основою сучасних навчальних закладів. Ефективне керування базами даних сприяє якісній взаємодії, дозволяє викладачам, адміністраторам і здобувачам освіти отримувати доступ до необхідної інформації у потрібний час, покращує організацію освітнього процесу, сприяє прийняттю обґрунтованих рішень.

Існує ряд важливих завдань в сфері освіти, які вирішуються за умови ефективного керування даними:

- освітні бази даних забезпечують адміністративну ефективність та спрощують процеси звітності, оцінювання та планування;
- бази даних є основним інструментом формування звітності та аналітичних досліджень, допомагають оцінити результати та визначити освітні траєкторії;
- бази даних забезпечують вільний доступ до освітніх матеріалів (за необхідності керований – за умови авторизації користувачів), інших важливих інформаційних ресурсів;
- підтримують персоналізоване навчання враховуючи індивідуальні потреби здобувачів освіти, сприяють адаптації до навчального процесу, покращують його ефективність;
- бази даних сприяють розширеній співпраці між освітніми закладами, викладачами, учнями, студентами, батьками, стейкхолдерами, формують єдиний освітній простір [30].

Тож, бази даних відіграють важливу роль у формуванні сучасного ефективного освітнього середовища, сприяють впровадженню інновацій і забезпечують успішність навчання.

Наразі існує визначена класифікація освітніх баз даних різного цільового призначення:

- Інформаційні системи для студентів (SIS) – комплексні освітні бази даних для управління та організації інформації, яка пов’язана зі студентами. Використовуються для збору, зберігання та пошуку даних про студентів (особисті дані, інформація про зарахування, академічні записи, відвідуваність тощо).
- Системи управління навчанням (LMS) – онлайн-платформи для надання, керування та відстеження освітнього контенту і діяльності. Використовуються для організації змішаного та дистанційного навчання, надають доступ до навчальних матеріалів, завдань, мультимедійних ресурсів, забезпечують оцінювання та взаємодію студентів, формують структуроване та інтерактивне середовище.
- Бази даних керування бібліотекою – універсальні системи спеціального призначення, які використовуються бібліотеками та навчальними закладами для каталогізації друкованих видань, цифрових ресурсів та інших матеріалів.
- Адміністративні бази даних – бази даних для реалізації адміністративних функцій (системи реєстрації студентів, системи управління фінансами, бази даних кадрів тощо).
- Дослідницькі бази даних – колекції наукових видань, журналів, статей, статистика освітньої галузі, інші наукові публікації, що стосуються різноманітних галузевих досліджень.
- Спеціалізовані освітні бази даних – стосуються конкретних навчальних дисциплін, освітніх спеціальностей та предметних галузей (тематичні статті, дисертації, реферати, цифрові архіви, освітні інструменти для різних спеціалізованих областей).

- Довідкові центри для викладачів – електронні ресурси, що забезпечують доступ до навчально-методичних матеріалів та спеціальної літератури для педагогічних працівників (плани занять, дослідницькі роботи, матеріали для навчання та професійного розвитку).
- Інформаційний центр освітніх ресурсів (ERIC) – системна база даних, яка спеціалізується на дослідженнях і ресурсах, пов'язаних з освітою. Пропонує матеріали (наукові статті, доповіді, навчальні посібники тощо) різної тематики та різного цільового призначення [30].

Ці освітні бази даних сприяють формуванню сучасного інтегрованого освітнього середовища, допомагають комплексно вирішувати проблеми освітньої галузі, забезпечують оптимізацію адміністративних процесів і визначають вектор академічних досліджень та інновацій.

Огляд освітніх баз даних визначає певну прогалину, що стосується системного обліку та ідентифікації педагогічних і науково-педагогічних працівників, збору та узагальнення інформації щодо дотримання кваліфікаційних вимог, атестаційних процесів, наявності даних про попит і пропозицію педагогічних працівників.

Важливість розробок у цій площині доводиться публікаціями зарубіжних науковців. Так Майкл Б. Аллен у своєму огляді "Guide to Teacher Data" згадує звіт Річарда Вурхіза та Гарі Барнсома "Data Systems to Enhance Teacher Quality", опублікований Управлінням з питань вищої освіти США (SHEEO), в якому досліджуються проблеми та пропонуються алгоритми рішень щодо створення і запровадження комплексної системи даних вчителів, акцентується увага на необхідності міжнародної інтеграції та уніфікації такої системи [40]. Також обґрунтовується недостатність проведених досліджень та ключове значення подібних проєктів в освітній галузі.

Підсумовуючи, бази даних є трансформаційними інструментами в освіті, що покращують доступність даних, адміністративну ефективність, персоналізоване навчання, процеси прийняття рішень, співпрацю та дослідження.

1.3 Нормативно-правова база інформатизації суб'єктів освітньої діяльності

Створення облікових баз педагогічних та науково-педагогічних працівників навчальних закладів різних рівнів акредитації є ключовим завданням в процесі систематизації даних та формування інтегрованого інформаційного освітнього простору.

У статті «Open Education and Education for Openness» (Peters, M. A., Britez, R. G.) зазначено, що цифровізація освіти не обмежується залученням відкритих освітніх ресурсів, а є інтегральною складовою цифрової економіки та відкритого суспільства, що передбачає створення нормативно-правової бази інформатизації різних суб'єктів та упровадження міжнародних стандартів для забезпечення пошуку, збору, збереження і використання інформації [43].

Стаття 3 Закону України «Про Національну програму інформатизації» серед основних пріоритетних завдань визначає наступні, а саме: формування систем національних інформаційних ресурсів, впровадження та застосування інформаційно-комунікаційних технологій в усі сфери суспільного життя, забезпечення рівного доступу до інформаційно-комунікаційних технологій та підвищення рівня цифрової освіти, підтримку сфери інформаційних продуктів і послуг, що має забезпечити активну інтеграцію України у світовий інформаційний простір [8].

Для успішної інтеграції України в міжнародну освітню систему необхідно дотримуватись всіх вимог щодо забезпечення якості освіти. Важливим аспектом є розробка засобів, які здатні допомогти в оцінці кваліфікацій та полегшити визнання дипломів, що має в свою чергу сприяти мобільності та підвищенню інтернаціоналізації вищої освіти.

Так у загальному переліку ключових питань та напрямків роботи міжнародних освітніх мереж, наприклад ENIC-NARIC (ENIC Network / European Network of National Information Centres), яка була створена ще у 1994 році Радою Європи та ЮНЕСКО з метою уніфікації та застосування єдиної для всіх

європейських країн політики та практики визнання дипломів і кваліфікацій [35], зазначені важливі інструменти визнання вищих навчальних закладів вищої освіти, які стосуються процедури підтвердження кваліфікацій та забезпечення якості у вищій освіті.

Засади забезпечення якості освіти безпосередньо пов'язані з професійним розвитком та підвищенням кваліфікації педагогічних і науково-педагогічних працівників, що прописано у пункті 3 Порядку підвищення кваліфікації педагогічних і науково-педагогічних працівників, затвердженим постановою Кабінету Міністрів України від 21 серпня 2019 року № 800 («Деякі питання підвищення кваліфікації педагогічних і науково-педагогічних працівників»), а також визначено статтею 59 Закону України «Про освіту», статтею 60 Закону України «Про вищу освіту», статтею 24 Закону України «Про фахову передвищу освіту» та Наказом Міністерства освіти і науки України «Про затвердження Положення про атестацію педагогічних працівників» від 9 вересня 2022 року № 805 [1; 5; 7; 9; 11].

Систематизація даних, що стосуються підвищення кваліфікації педагогічних і науково-педагогічних працівників, загальних облікових даних, які можуть бути використані для формування особових карток працівників, електронних трудових книжок, іншої звітної документації, є важливою передумовою створення уніфікованих систем електронного обліку для ефективної взаємодії у глобальному інформаційному просторі.

1.4 Приклади цифровізації та уніфікації освітніх даних

1.4.1 Уніфікована електронна система обліку ЄДЕБО

Першим етапом впровадження уніфікованої електронної системи обліку стало створення 13 липня 2011 року постановою Кабінету Міністрів України №752 Єдиної державної електронної бази з питань освіти. Система надає інформацію щодо:

– електронного ліцензування освітньої діяльності;

– акредитації освітніх програм та спеціальностей;

– електронних кабінетів вступників (абітурієнтів);

– перебігу вступної кампанії (конкурсних пропозицій, рейтингових списків вступників, рекомендацій до зарахування та зарахування до закладів вищої та фахової передвищої освіти);

– перевірки достовірності документів про освіту;

– статистичних звітів у сферах фахової передвищої та вищої освіти тощо [15].

Серед основних реєстрів та сервісів, які представлені на сайті ЄДЕБО, є Реєстр сертифікатів педагогічних працівників (рис. 1.1), що надає можливість перевірки достовірності сертифікатів педагогічних працівників, виданих за результатами успішного проходження сертифікації [4; 15].

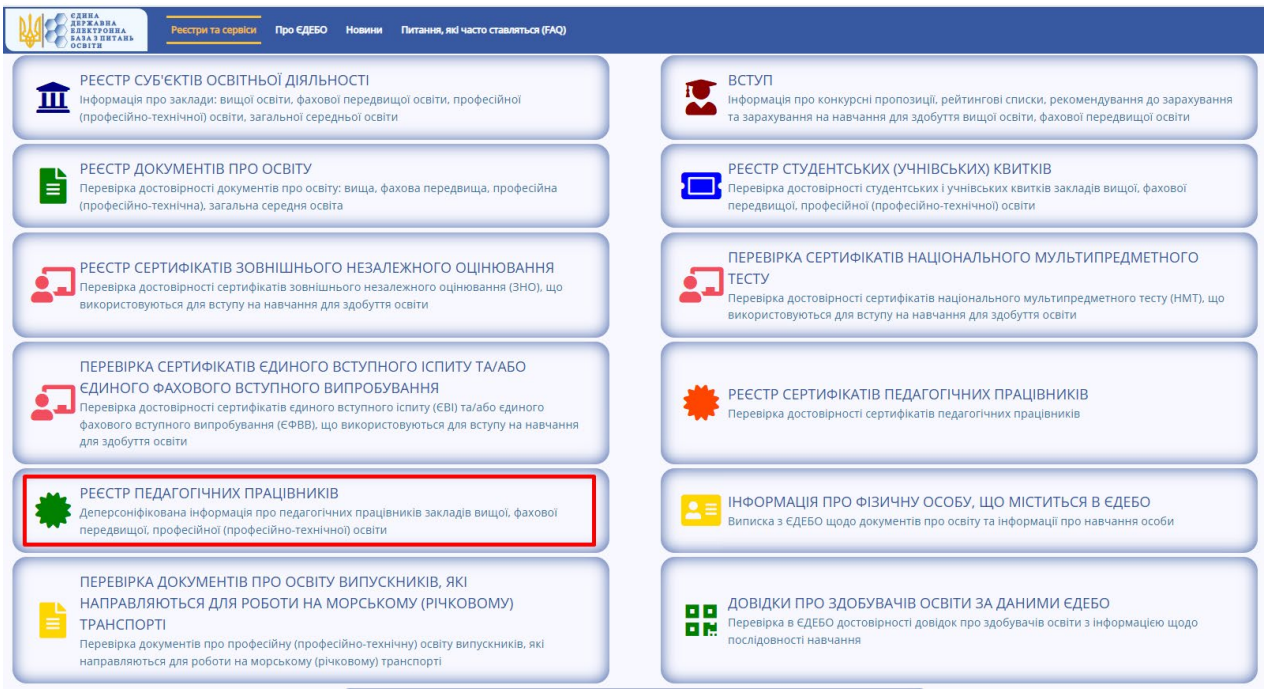

Рисунок 1.1 – Головна сторінка сайту ЄДЕБО

Для пошуку сертифіката в Реєстрі зазначаються необхідні дані: прізвище, ім’я, по-батькові педагогічного працівника, номер відповідного сертифіката (рис. 1.2). Дані заповнюються згідно з їх написанням у документі та передаються до

пошукової системи в захищеному вигляді.

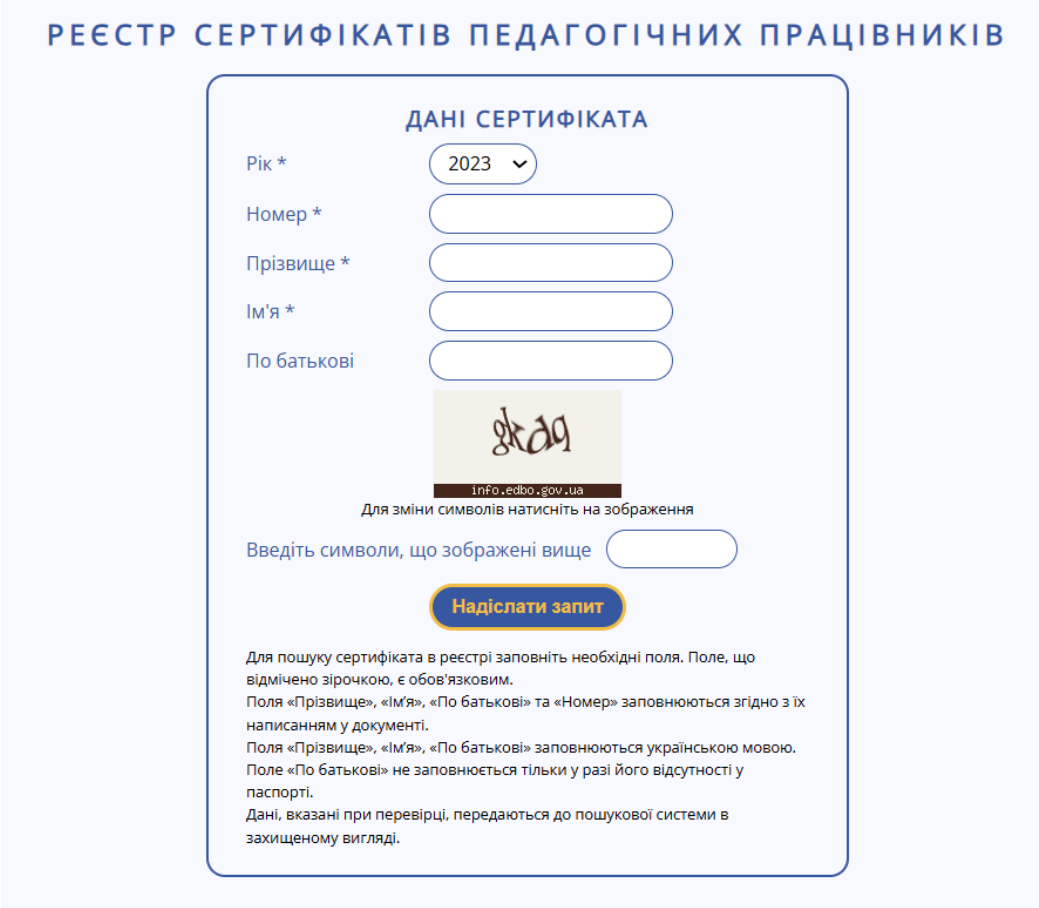

Рисунок 1.2 – Сторінка Реєстру сертифікатів педагогічних працівників на сайті ЄДЕБО

Для захисту конфіденційних даних порядок доступу до ЄДЕБО максимально детально прописаний. Доступ до інформації у форматі відкритих даних забезпечується технічним адміністратором шляхом її розміщення на веб-сайті ЄДЕБО за електронною адресою https://info.edbo.gov.ua/, зокрема через офіційний веб-сайт розпорядника ЄДЕБО. Технічним адміністратором ЄДЕБО є державне підприємство «Інфоресурс». В той же час, фізична особа формує запит до всієї інформації про себе, що міститься в ЄДЕБО, з накладенням електронного цифрового підпису, у якому зазначаються прізвище, ім’я, по батькові, місце проживання і реквізити документа, що посвідчують фізичну особу, яка подає

запит. Надання персональних даних, що містяться в ЄДЕБО, третім особам здійснюється уповноваженими суб’єктами виключно у випадках, передбачених законами, або за згодою суб’єкта персональних даних [13].

1.4.2 Електронна платформа ЄАС для цифровізації атестаційного процесу

З 15 жовтня 2022 року навчальним закладам України надається доступ до атестаційних кабінетів педагогічних і науково-педагогічних працівників в Єдиній атестаційній системі (ЄАС), яка була створена за підтримки Міністерства освіти і науки України. Єдина атестаційна система (ЄАС) є електронною платформою для цифровізації атестаційного процесу. Це тестовий проєкт, який має об'єднати всю ланку атестаційного процесу – заклади освіти, атестаційні комісії, ІППО (Інститути післядипломної педагогічної освіти) та суб'єктів підвищення кваліфікації. Можливості системи передбачають доступ адміністрацією освітнього закладу до атестаційної інформації, цифрових сертифікатів, свідоцтв та інших документів необхідних для атестування педагогічних і науково-педагогічних працівників, а також завантаження та доступ до сертифікатів курсів підвищення кваліфікації в особистому кабінеті науково-педагогічного працівника, доступ до списків закладів освіти та педагогів, які атестуються, захист даних від несанкціонованого доступу, що підтверджується атестатом відповідності комплексної системи захисту інформації (КСЗІ) [14].

ЄАС також забезпечує автоматизацію всіх етапів атестаційного процесу:

– реєстрацію педагогічних та науково-педагогічних працівників, що можуть зареєструватися в ЄАС самостійно або за допомогою відповідальної особи в освітньому закладі (рис. 1.3);

– формування атестаційних листів (на підставі даних, внесених педагогами та атестаційними комісіями, відповідно до наказу МОН від 09.09.2022р. № 805 “Про затвердження Положення про атестацію педагогічних працівників”);

– оцінювання педагогічних працівників атестаційними комісіями на підставі атестаційних листів та інших документів;

– прийняття рішень про атестацію педагогів атестаційними комісіями на підставі їхніх оцінок [14].

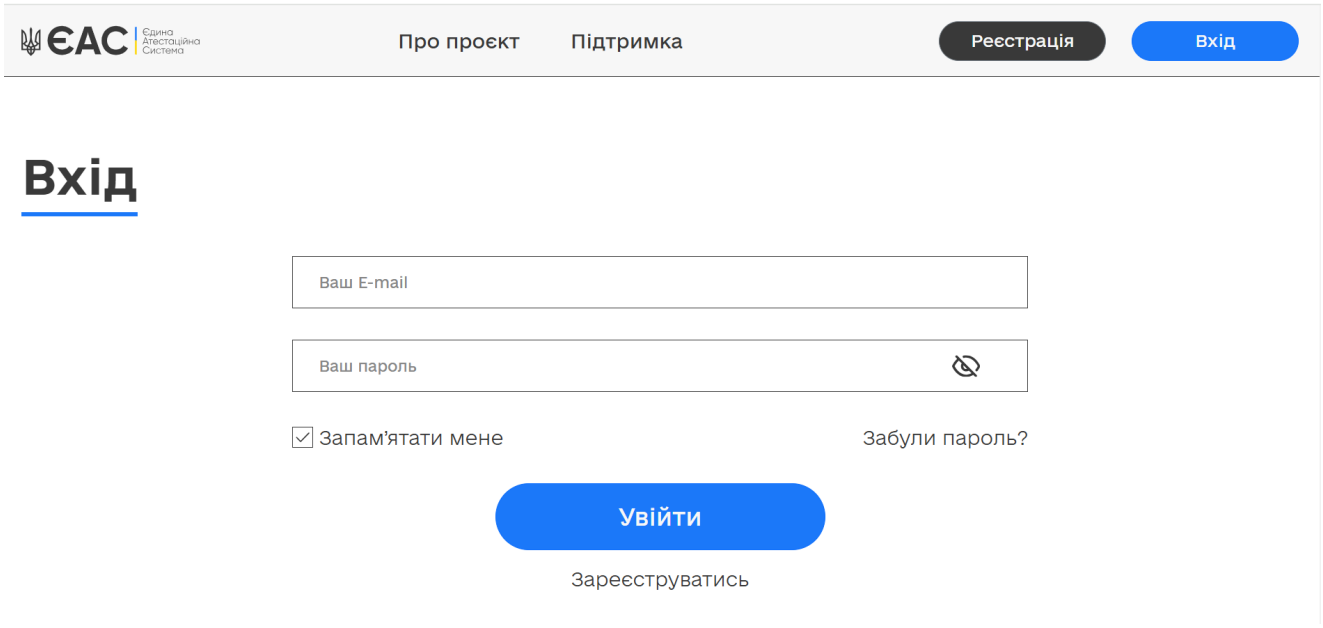

Рисунок 1.3 – Сторінка реєстрації на сайті ЄАС

В особистому кабінеті науково-педагогічного працівника (рис. 1.4) зазначені дані, що стосуються назви закладу освіти, де працює педагог, особисті (анкетні) дані, які за потреби можна редагувати (рис. 1.5), документи ЄАС і документи від установ, які не інтегровані з ЄАС, а також додається облік учнівських досягнень (документи отримані учнями / студентами, які можуть бути враховані під час атестації).

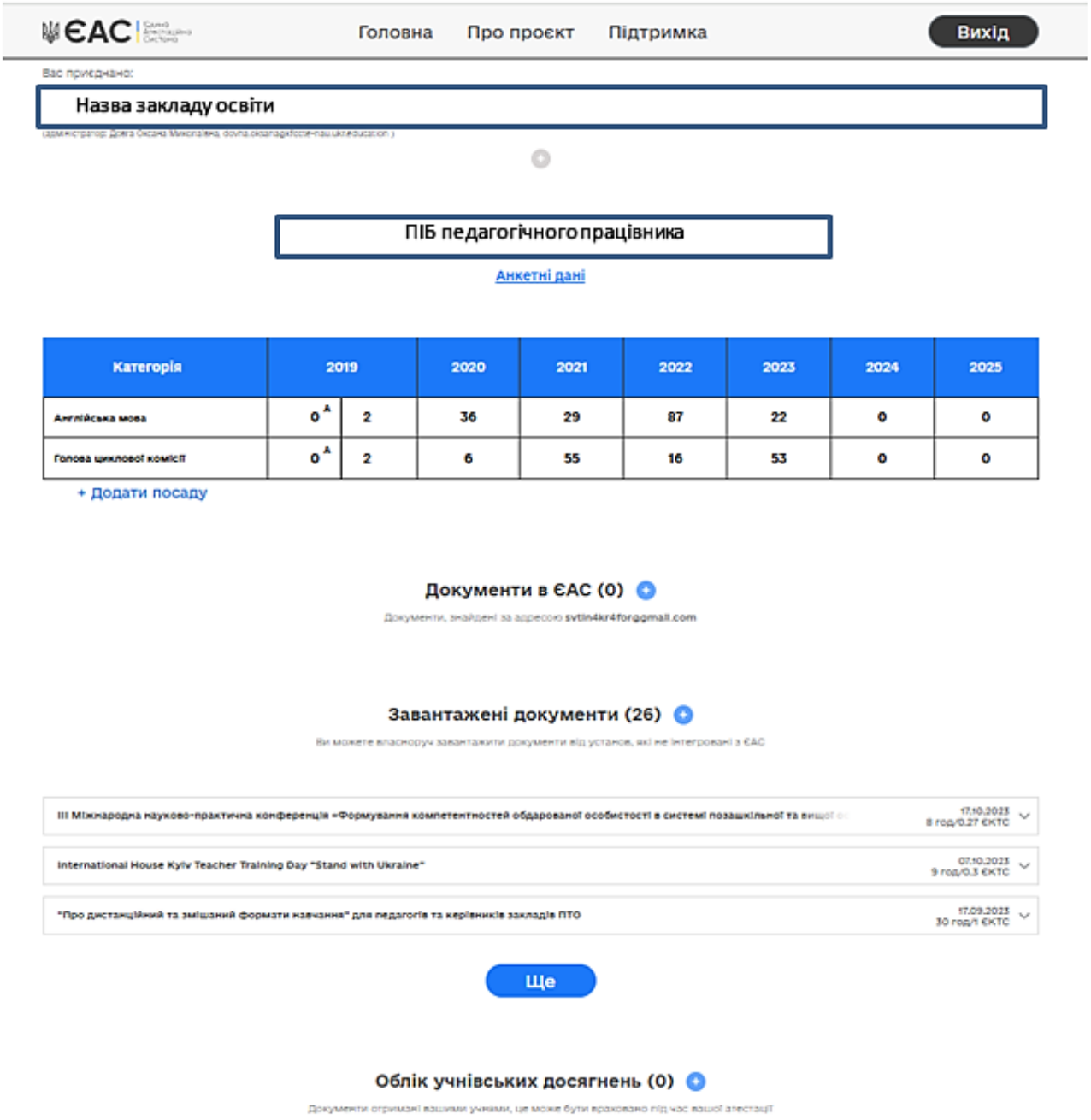

Рисунок 1.4 – Особистий кабінет педагогічного працівника на сайті ЄАС

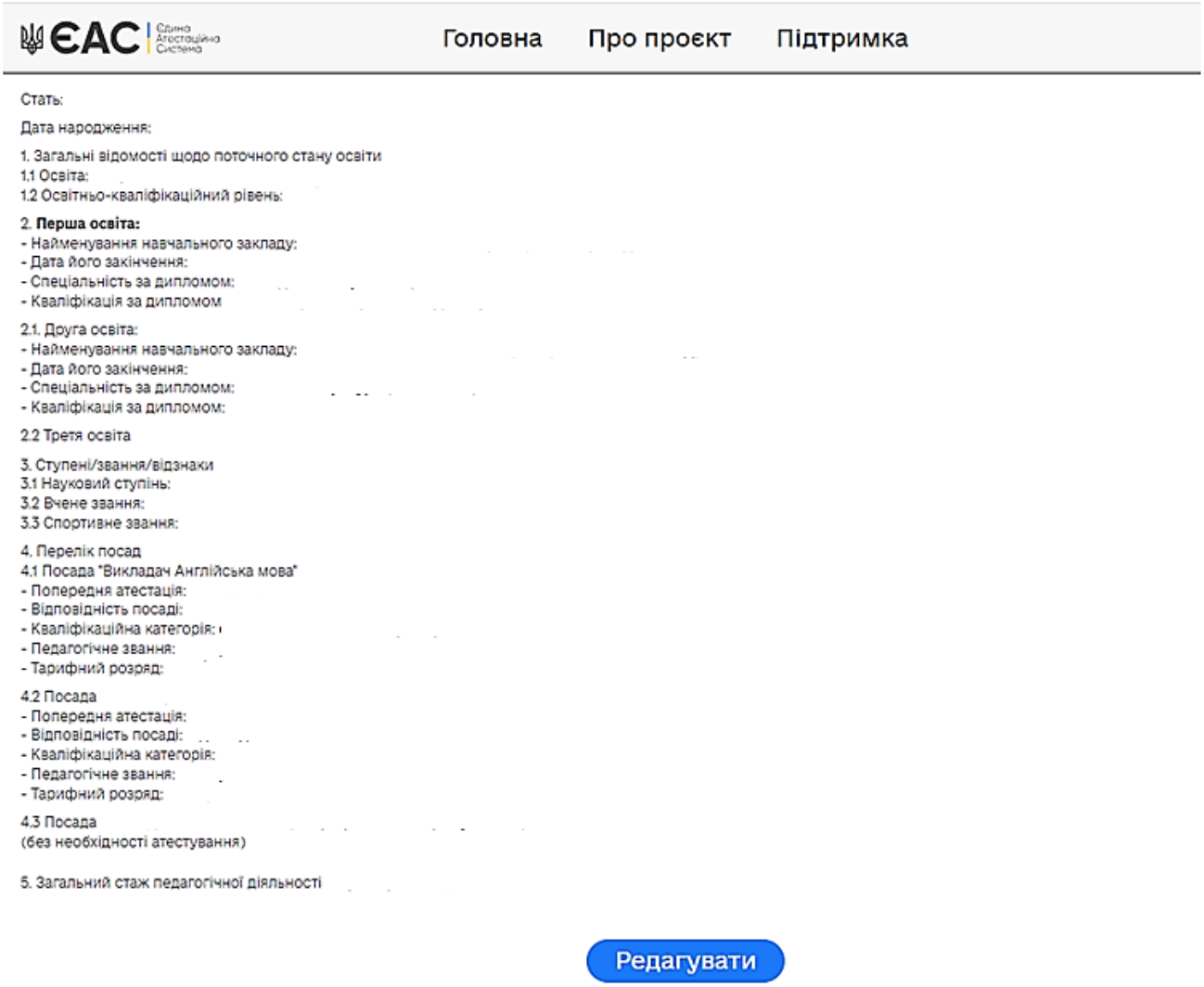

Рисунок 1.5 – Сторінка «Анкетні дані» на сайті ЄАС

Серед закладів освіти, які використовують платформу ЄАС, Національний педагогічний університет імені М.П. Драгоманова, Київський університет імені Б. Грінченка, Інститут цифровізації освіти Національної академії педагогічних наук України, Тернопільський національний педагогічний університет імені В. Гнатюка та інші [14].

1.4.3 Електронний застосунок єДокумент про освіту

Важливим кроком у сфері цифрової трансформації процесів управління, регулювання та моніторингу в ЗВО стала реалізація проєкту щодо використання

застосунку єДокумент про освіту, який був запущений у використання на Єдиному порталі державних послуг «Дія» 22 березня 2024 року відповідно до постанови Кабінету Міністрів України від 4 листопада 2022 р. № 1242 (спільний проєкт Міністерства освіти і науки України та Міністерства цифрової трансформації України).

Застосунок надає доступ до документів про освіту, які містяться в Реєстрі документів про освіту Єдиної державної електронної бази з питань освіти, а саме: свідоцтво про базову середню освіту, атестат/свідоцтво про повну загальну середню освіту, свідоцтво про присвоєння (підвищення) робітничої кваліфікації, диплом кваліфікованого робітника, диплом фахового молодшого бакалавра, диплом молодшого спеціаліста, диплом молодшого бакалавра, диплом бакалавра, диплом спеціаліста, диплом магістра, диплом доктора філософії та диплом доктора мистецтва [2].

Для відображення документа про освіту необхідно, щоб РНОКПП (Реєстраційний номер облікової картки платника податків) користувача в Дії збігався з РНОКПП у даних документа про освіту в ЄДЕБО.

Крім цього, у застосунку відображаються документи про освіту, видані з 2000 року, крім деяких винятків, а саме: інформації про випускників військових закладів вищої (фахової передвищої) освіти та військових навчальних підрозділів закладів вищої (фахової передвищої) освіти, а також щодо дипломів про перепідготовку (молодших спеціалістів, спеціалістів), що видавались до 2016 року, та окремих документів про освітньо-науковий ступінь доктора філософії (доктора мистецтва), виданих до 2022 року [3].

1.4.4 Автоматизований інформаційний комплекс освітнього менеджменту

Ще однією спробою щодо створення сучасного інформаційного середовища зі збору, обробки та верифікації освітніх даних став проєкт АІКОМ – Автоматизований інформаційний комплекс освітнього менеджменту, який

з’явився у липні 2019 року. Це електронна система управління освітою зі збору, зберігання, управління та використання даних (рис. 1.6). АІКОМ здатен генерувати статистику та здійснювати моніторинг в межах освітньої системи [12].

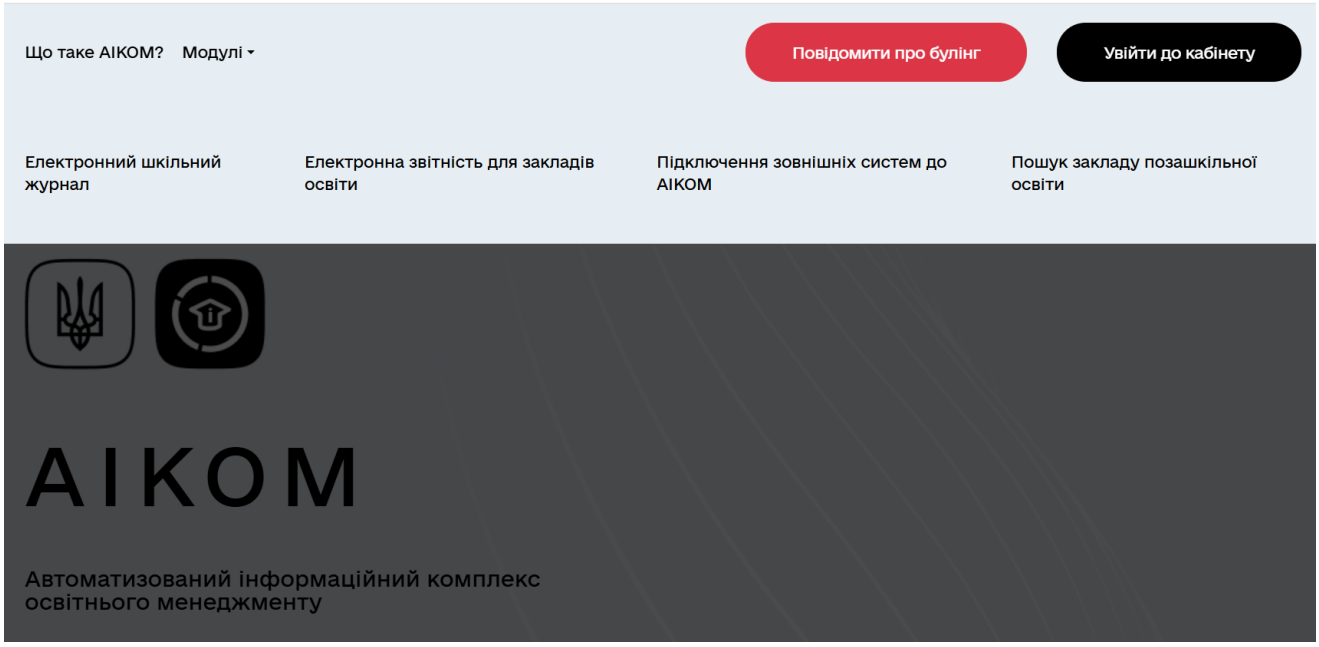

Рисунок 1.6 – Сторінка «Модулі» на сайті ЄАС

Проєкт об’єднує заклади дошкільної, загальної середньої, позашкільної та професійної (професійно-технічної) освіти. Електронна система дає змогу суттєво підвищити достовірність освітньої статистичної та адміністративної інформації та покращити якість управлінських рішень.

Серед основних цілей проєкту:

– збір актуальних даних та автоматична генерація звітності;

– підвищення кваліфікації педагогічних працівників через електронну систему на базі АІКОМ;

– реалізація можливості взаємодії АІКОМ з різними освітніми системами, базами даних та державними реєстрами;

– автоматичне генерування документів затверджених форм на основі внесених даних в електронний журнал;

– облік дітей дошкільного, шкільного віку та закладів дошкільної освіти;

– запровадження електронної системи управління професійною (професійно-технічною) освітою та інше [12].

Хоча ще триває пілотне тестування проєкту, 7 травня 2024 року в рамках співпраці, покращення взаємодії та інтеграції освітніх систем до центральної бази даних АІКОМ приєдналася освітня інформаційна система (ОІС) «Всеосвіта».

1.5 Аналіз недоліків автоматизованих систем та постановка задачі

Як показують дослідження, на загальнодержавному рівні є певні спроби систематизації даних, що стосуються сфери освіти. Багато кроків було зроблено щодо запровадження цифрових технологій в освітньому процесі, але практично немає ефективної моделі створення загальнодоступної уніфікованої системи обліку даних педагогічних та науково-педагогічних працівників навчальних закладів різних рівнів акредитації. Також немає єдиного стандарту створення подібних баз даних.

Треба зазначити, що функціональні можливості і доступ до наявних автоматизованих систем, які досліджувалися, мають певні обмеження, що суттєво впливає на ефективність електронної комунікації між суб’єктами, закладами та установами різного рівня. Так перелік навчальних закладів, які користуються Єдиною атестаційною системою (ЄАС), нажаль, поки досить обмежений, а особова інформація в кабінетах педагогічних і науково-педагогічних працівників не передбачає внесення додаткових даних, окрім тих, що стосуються проходження підвищення кваліфікації та стажування.

В системі ЄДЕБО та у застосунку єДокумент про освіту на Порталі «Дія» не відображаються певні документи про освіту, особливо видані до 2000 року. Також документи про наукові ступені доктора наук, кандидата наук та про вчені звання професора, доцента, старшого дослідника не є документами про освіту, тому відсутні в Єдиній державній електронній базі з питань освіти.

Проведені дослідження спонукають до пошуку альтернативних рішень в процесі розробки і запровадження електронних систем обліку даних в українських освітніх закладах. Досвід таких проєктів як ЄДЕБО, Єдина атестаційна система (ЄАС), Єдиний портал державних послуг «Дія», автоматизований інформаційний комплекс АІКОМ доводить необхідність подальших досліджень для вирішення актуальної задачі щодо розробки і впровадження програмного засобу – системи обліку педагогічних і науково-педагогічних працівників. Уніфікація відповідної інформації, що стосується загальних облікових даних науково-педагогічних працівників для формування особових карток, підвищення їх кваліфікації для атестаційних звітів, іншої звітної документації, має забезпечити як внутрішні потреби окремих навчальних закладів (організацій), так і для зовнішньої електронної комунікації, враховуючи потреби України щодо інтеграції в міжнародний освітній простір.

Практичний досвід запровадження та використання систем обліку викладачів розглядається в статті «Teachers Record Management System Project» (Kamal Acharya), в якій представлено аргументований аналіз сучасних методів і технологій для розробки подібних програмних продуктів, та їх інтеграцію в багаторівневий освітній простір [39].

Таким чином, результатом даної кваліфікаційної роботи є подібна система обліку, що дозволить систематизувати облікові дані науково-педагогічних працівників та синхронізувати роботу освітніх закладів різних рівнів акредитації.

2 АНАЛІЗ ТА ДОСЛІДЖЕННЯ ЕФЕКТИВНИХ РІШЕНЬ СИСТЕМ КЕРУВАННЯ БАЗАМИ ДАНИХ

2.1 Огляд функціональних можливостей та класифікації СУБД

Керування базами даних передбачає використання відповідних інструментів систем управління базами даних (СУБД) із таким набором функцій, як зберігання даних, пошук, обробка запитів, захист, контроль доступу, резервне копіювання та відновлення, аналітика.

Ці процеси стосуються функціональних і технологічних рішень щодо набору структурованих або неструктурованих даних, що зберігаються в різних форматах, таких як таблиці, документи та пари «ключ-значення».

В загальному розумінні, СУБД (система управління базами даних) – це спеціальне програмне забезпечення, що дозволяє ефективно взаємодіяти з базою даних [31].

Наскільки ця взаємодія буде ефективною залежить від правильного вибору СУБД. На сьогодні на ІТ ринку існує понад 300 систем керування базами даних, рейтинг популярності яких щомісяця оновлюється на сторінці DB-Engines [32; 42].

Тож, вибір найкращого рішення для застосування системи керування базами даних є першочерговим в процесі розробки програмного продукту, а саме системи обліку науково-педагогічних працівників з подальшим тестуванням і запровадженням.

В процесі дослідження аналіз СУБД стосувався їх функціональних можливостей, потрібних для реалізації програмного завдання. Класифікація СУБД передбачає розподіл за певними критеріями, а саме: за моделлю даних (ієрархічні, реляційні, мережеві, об’єктно-орієнтовані, об’єктно-реляційні), за функціональним призначенням (оперативні та аналітичні), розподілом (централізовані та розподілені), типом використання (SQL-орієнтовані та NoSQL) і ліцензією (відкриті (Open Source), пропрієтарні (Commercial) [29].

Якщо ієрархічні СУБД надають ефективний доступ до деревоподібних структур даних (кожен запис має одного «предка / батька» і багато «нащадків») та демонструють простоту в моделюванні зв’язків, то реляційні СУБД забезпечують велику підтримку SQL (Structured query language, мова структурованих запитів для взаємодії користувача з базами даних) для запитів і маніпуляції даними, а також незалежність даних від фізичної структури зберігання.

Мережеві СУБД використовуються для підтримки складних і взаємопов’язаних даних, але складні в управлінні структурою та забезпечення запитів. Об’єктно-реляційні, які поєднують переваги реляційних і об’єктно-орієнтованих моделей та застосовуються для підтримки більш складних структур даних, складні у проєктуванні [29].

Враховуючи технологічні вимоги та перспективи реалізації програмного продукту, функціональні можливості наявних СУБД за моделями даних вимагають детального дослідження.

2.2 Прикладне дослідження основних моделей систем керування базами даних

2.2.1 Ієрархічна модель даних

Ієрархічна модель даних (Hierarchical database) представляє базу даних у вигляді деревоподібної (ієрархічної) структури, що складається з об’єктів (даних) різних рівнів, які мають між собою певні зв’язки. Структура бази даних схожа на дерево з вузлами, що представляють записи, і гілками, що представляють поля. В одному об’єкті може міститися кілька окремих об’єктів більш низького рівня («нащадків»). Ці структурні зв’язки відомі як «батько-нащадок», де об’єкт-батько або не має «нащадків», або має декілька об’єктів більш низького рівня. У той же час об’єкт-нащадок має лише одного «батька» (рис. 2.1). Об’єкти, що мають спільного «батька», називаються «братами / близнюками».

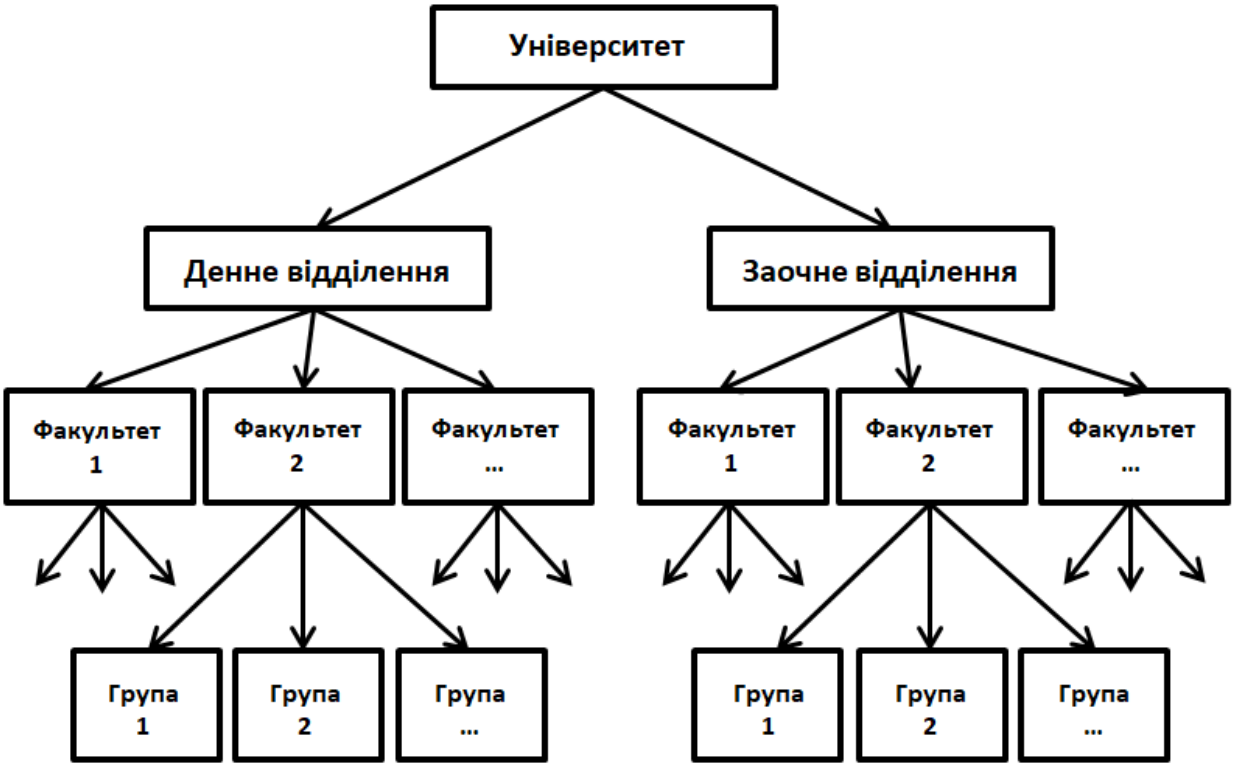

Рисунок 2.1 – Приклад ієрархічної структури баз даних

Основними структурними елементами ієрархічної модель даних є рівень, елемент (або вузол) та зв’язок.

Вузол - це сукупність атрибутів даних, що описують деякий об’єкт. На схемі ієрархічного дерева вузли представляються вершинами графа. Кожен вузол на більш низькому рівні пов’язаний лише з одним вузлом, що знаходиться на більш високому рівні. Схематично вузли ієрархічного дерева представлені як вершини графа (рис. 2.2).

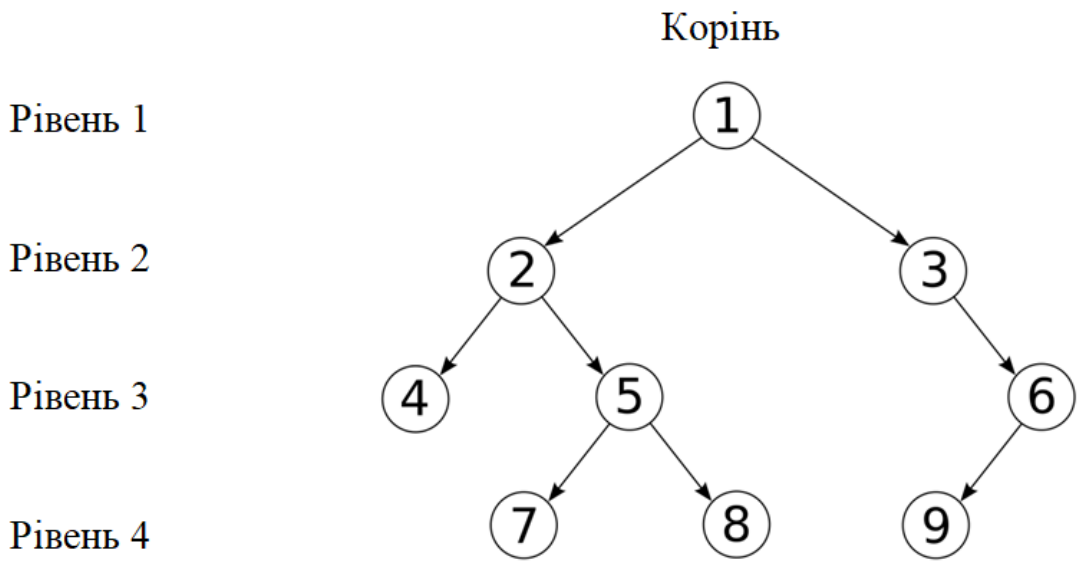

Рисунок 2.2 – Ієрархічна структура даних

Верхній (перший) рівень ієрархічного дерева – це його вершина. Кількість рівнів визначається кількістю зв’язків, отже залежні вузли знаходяться на більш низьких рівнях (другому, третьому і т.д. відповідно). Водночас кількість дерев в ієрархічній моделі даних визначається числом кореневих записів.

Основними одиницями ієрархічній моделі даних є поле (найменша неподільна одиниця даних) і сегмент (конкретні значення полів даних). Всі об’єкти ієрархічній моделі («батьки» і «нащадки») мають певні елементами:

– атрибут – визначає функціональну сутність об’єкта (подається найменшою одиницею елемента даних);
– запис – визначає конкретний екземпляр певного об’єкта (представлений групою атрибутів);
– групове відношення – зв'язок між записами різних типів, що характеризує взаємодію записів «батьки-нащадки»;
– ключовий елемент – подається атрибутом, що містить лише унікальні значення для кожного запису.

Ієрархічні бази даних мають просту модель зв’язків і є найстарішими. Серед основних недоліків ієрархічних моделей є:

- досить повільний доступ до сегментів даних нижніх рівнів ієрархії;
- не передбачений зв’язок «багато-до-багатьох», що суттєво обмежує систему зберігання даних;
- необхідна чітка орієнтація на певні типи запитів.

Основні приклади застосування – організація файлових систем, DNS та LDAP-з’єднання [24; 31]. Не зважаючи на обмежену гнучкість в обробці складних зв’язків, активно використовується в логістиці та фінансах, для відстеження відправлень і керування фінансовими рахунками [31].

Серед найвідоміших ієрархічних моделей баз даних визначають Information Management System (IMS), розробник – IBM (1966 - 1968 р.); Time-Shared Data Management System (TDMS), розробник – System Development Corporation; Mark IV MultiAccess Retrieval System від компанії Control Data Corporation; System 2000 – SAS Institute; InterSystems Caché.

2.2.2 Структура мережевої моделі баз даних

На відміну від ієрархічних моделей баз даних мережеві СУБД (Network model database) здатні підтримувати відношення «багато до багатьох». Фактично це розширення ієрархічного підходу. В ієрархічних структурах запис «нащадок» повинен мати в тільки одного «предка» («батька»). В мережевій структурі даних у «нащадка» може бути будь-яке число «предків» (рис. 2.3).

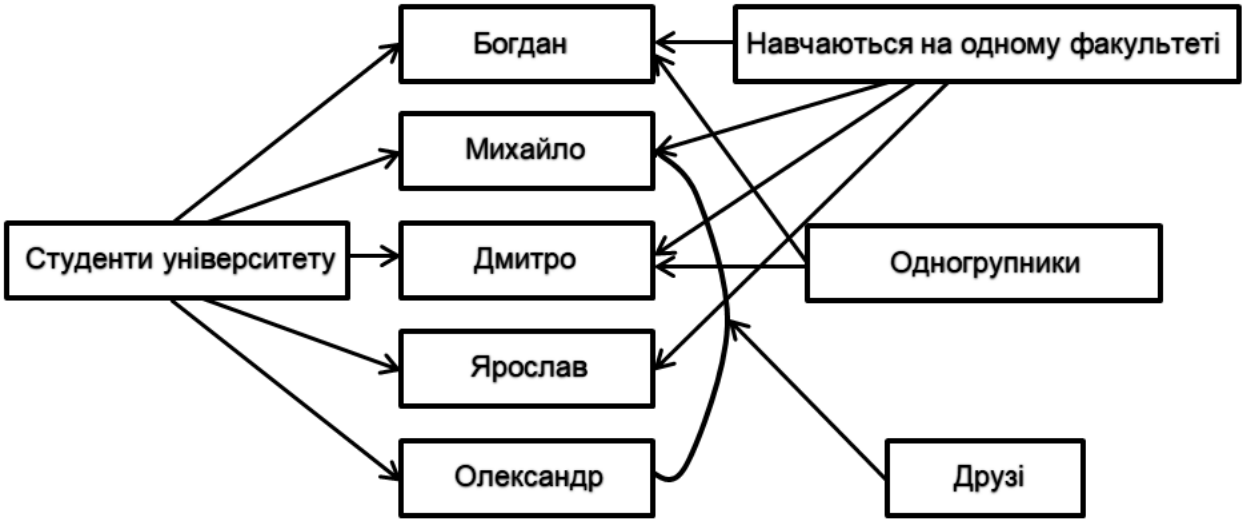

Рисунок 2.3 – Приклад мережевої структури даних

Мережева база даних подається загальним графом (сукупністю двох кінцевих множин). Вона складається з набору екземплярів певного типу запису і набору екземплярів певного типу зв’язків між цими записами. Структуру мережевої моделі визначають: елемент даних (найменша одиниця даних), агрегат даних (сукупність елементів даних усередині запису) і запис (сукупність елементів даних і / або агрегатів). Всі визначені структурні одиниці мають свої імена.

За допомогою елемента даних здійснюється побудова всіх інших структур. Агрегат даних, в свою чергу, може бути простим (складається тільки з елементів даних) і складовим (якщо включає до свого складу інші агрегати). Запис може мати складну ієрархічну структуру та не входить до складу жодного іншого агрегату.

Набір записів утворює дворівневу ієрархічну структуру підпорядкування. Кожен тип набору представляє певний зв'язок між двома або кількома типами записів.

Така модель призводить до складних структур бази даних. Не зважаючи на це, мережева модель даних досить ефективна за показниками витрат пам'яті і оперативності [16]. Основні сфери використання – телекомунікації, соціальні

мережі, системи бронювання авіаквитків, транспорт для маршрутизації викликів, топології мережі та відстеження відправлень [31].

Прикладами найвідоміших мережевих СУБД є: IDS (Integrated Data Store) – компанія General Electric; IDS/2 – компанія Honeywell; IMAGE/3000 – компанія Hewlett-Packard; dbVista; Universal Datenbank System (UDS) від Siemens; Caché.

2.2.3 Функціональні і технологічні рішення реляційних баз даних. SQL запити

Реляційні бази даних (Relational database) представляють дані у формі таблиць зі стовпцями та рядками, між якими є попередньо визначені зв'язки. У рядках наводяться відомості про об'єкти (значення властивостей), а стовпці є властивостями об'єктів (поля). Рядки таблиці, які називаються «кортежами», мають однакові атрибути – стовпці (рис. 2.4).

Рисунок 2.4 – Таблиця реляційної бази даних

Складні відносини об'єктів у реляційних баз даних узгоджуються за допомогою ключів. Так основний ключ (Primary Key) надає можливість унікальної ідентифікації кожного окремого кортежу (рядка) таблиці за

значеннями одного чи декількох атрибутів. Зовнішній ключ (Foreign Key) створює зв'язки між таблицями за допомогою посилань. Це спрощує проєктування баз даних та мінімізує надмірності при описі властивостей об'єктів [29].

Для ефективного вирішення завдань та використання SQL моделі необхідно враховувати основні елементи реляційних баз даних, які і утворюють їхню структуру (табл. 2.1).

Таблиця 2.1 – Структура реляційних баз даних

Структурні елементи SQL баз даних	**Опис та функції**
Реляційні таблиці	мають ім'я та структуру (рядки та стовпці), визначену схемою даних
Рядки (кортежі)	окремі записи, що містять інформацію про об'єкт, кожен запис є унікальним та ідентифікується за допомогою ключа
Стовпці (атрибути)	мають ім'я і тип даних (визначають інформацію про характеристики об'єктів, що описуються в таблиці і зберігаються в цьому стовпчику - текст, числа, дати тощо)
Ключі	використовуються для унікальної ідентифікації рядків у таблиці. Види ключів: - первинний (PRIMARY KEY) – унікальний, єдиний, найголовніший ключ; атрибут (набір атрибутів), що ідентифікує кортеж даного відношення та задається обмеженням; - вторинний (FOREIGN KEY) – зовнішній ключ, що визначає спосіб об'єднання таблиць; одне або кілька полів (стовпців) у таблиці, що містять посилання на певне поле або поля первинного ключа в іншій таблиці.
Зв'язки	дають змогу об'єднувати дані з різних таблиць для виконання складних запитів
SQL (Structured Query Language)	використовує стандарт ACID (Atomicity, Consistency, Isolation, Durability) для успішного виконання транзакцій; дає змогу створювати, змінювати, видаляти та витягувати дані з таблиць, а також визначати правила для цілісності даних
Нормалізація	зменшення надмірності даних і забезпечення їхньої цілісності
Транзакції	забезпечуються стандартом ACID, що є ефективним під час одночасного доступу кількох користувачів до бази даних

Дані сформовано з [29]

Щоб підтримувати та надсилати запити до реляційної бази даних, система керування базами даних використовує мову структурованих запитів (SQL, Structured Query Language), яка забезпечує простий інтерфейс програмування для взаємодії з базою даних [19].

Реляційна модель є найпопулярнішим типом СУБД, оскільки характеризується високим рівнем надійності та безпеки. Серед її основних переваг:

- універсальність (підходить майже для будь-яких завдань, що потребують обробки великої кількості запитів та зберігання значних обсягів даних);
- структурована мова запитів;
- добре задокументована (велика підтримка SQL для запитів і маніпуляції даними);
- доступність, зрозумілість для користувачів;
- нормалізація даних (мінімізації надлишковості даних, поліпшення їх цілісності, стійкість до відмов);
- поширення (основний інструмент розробки різних застосунків і сервісів) [16; 24; 29].

Реляційна база даних ідеально підходить для зберігання структурованих даних (поштових індексів, номерів кредитних карток, дат, ідентифікаційних номерів тощо).

Основні недоліки реляційних баз даних – це жорстка фіксована структура і зниження продуктивності при роботі з великими обсягами даних.

Прикладами популярних реляційних моделей баз даних є MySQL, MariaDB, Microsoft SQL Server, PostgreSQL, SQLite [24].

Реляційні бази даних широко використовуються для зберігання та управління корпоративними даними (корпоративні системи управління базами даних), у веб-додатках (інформація про користувачів, замовлення, контент тощо), в системах електронної комерції.

2.2.4 Сутність об’єктно-орієнтованої моделі БД

Об’єктно-орієнтована модель бази даних (Object database) – це система керування базами даних, в якій дані зберігаються як абстрактні об’єкти з певними властивостями, та визначаються або створюються за допомогою методів об’єктно-орієнтованого програмування. Ця модель БД використовується зі складними структурами даних (об’єктами) для підвищення продуктивності програмних продуктів та IT-систем [29; 31].

Сутність об’єктно-орієнтованої моделі в тому, що дані та методи, що їх обробляють, об’єднуються в структури, які називаються об’єктами (рис. 2.5). Типи об’єктів називаються класами. Числа, символи, символьні рядки довільної довжини – це найпростіші об’єкти.

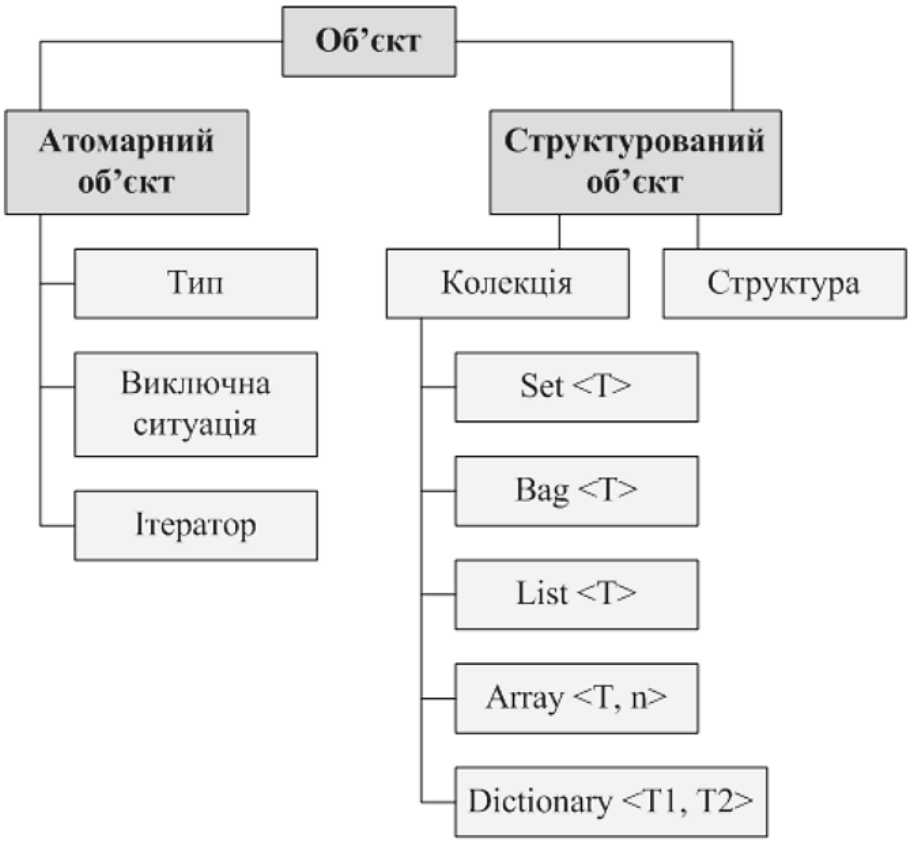

Рисунок 2.5 – Класифікація об’єктів об’єктно-орієнтованої бази даних

Щоб побудувати з простих об’єктів складні (кортежі, множини, мультимножини, списки, масиви), використовуються конструктори (для структур – Struct, для типів колекцій – Set, Bag, List, Array). Конструктори застосовуються для роботи з будь-якими об’єктами. Певні операції забезпечують маніпулювання складними об’єктами та розповсюджуються на всі їх компоненти.

Об’єктно-орієнтована модель бази даних має ряд важливих особливостей, а саме:

- забезпечує підтримку структур даних, що мають довільний рівень складності;
- ідентифікує, визначає унікальність об’єктів;
- визначає належність об’єктів класам (узагальнює спільні риси об’єктів, що мають однакові властивості, встановлюючи їхню структуру, поведінку, інформацію про їхні стани);
- використовує інкапсуляцію (приховує інформацію про дані та програмні коди для маніпулювання даними);
- забезпечує успадкування та ієрархію класів (надає можливість створювати нові класи з використанням даних і методів інших класів);
- застосовує принцип поліморфізму (дає змогу змінювати методи в успадкованих класах) [29].

Об’єктно-орієнтовані бази даних (Versant Object Database, Poet, ZODB, db4o та інші) використовуються з такими мовами програмування, як Java, C, C++, Kotlin [31].

Основні галузі застосування об’єктно-орієнтованих СУБД це сфера фінансів та мультимедійні бази даних.

2.2.5 Технології об’єктно-реляційних баз даних

Об’єктно-реляційна модель бази даних (Object-Relational Database / ORD) подібна до реляційної, але має структурні елементи об’єктно-орієнтованої моделі (об’єкти, класи, наслідування, що підтримуються в схемі даних та мові запитів).

Об’єктно-реляційна база даних підтримує розширення моделі даних новими типами даних та методами. Така технологія застосовується і в реляційних базах даних.

Об’єктно-реляційна СУБД дозволяє завантажувати код, призначений для обробки «нетипових» даних. Таким чином, база даних зберігає свою табличну структуру, але спосіб обробки деяких полів таблиць визначається ззовні.

Оскільки об’єктно-реляційна база даних об’єднує технології об’єктно-орієнтованої та реляційної моделей, вона потребує певного узгодження. Структура мови запитів для об’єктних даних ObQL (Object Query Language) схожа на мову запитів SQL (Structured Query Language), але на сьогодні вже існує декілька мов запитів і програмних інтерфейсів для об’єктно-реляційних баз даних (OQL, JDOQL, db4o SODA) (табл. 2.2).

Таблиця 2.2 – Мови запитів ORD

Мова запитів ORD	Програмна мова	Приклад запиту

OQL	JavaScript	`SELECT pc.cpuspeed`
		`FROM PCs pc`
		`WHERE pc.ram > 64;`

JDOQL	Java	`Employee.class, “salary>boss.salary”`
db4o SODA	Java .NET	`objectContainer . store ( new SomeClass ());`

Основні мови програмування: Java, C++, Visual Basic .NET або C#.

Серед недоліків об’єктно-реляційної моделі відзначають додаткову складність у проєктуванні та виникнення можливих проблеми з продуктивністю через додаткові шари абстракції [29].

2.2.6 Варіативні моделі хмарних баз даних

Стрімкий розвиток технологій штучного інтелекту (AI) та Інтернету речей (IoT) вимагає шукати зручні рішення для зберігання та обробки даних на віддалених серверах.

Хмарні бази (Cloud Database) даних все частіше залучаються як ефективний інструмент підтримки різних механізмів баз даних. Вони пропонують широкий вибір інструментів для обробки великих обсягів даних, що вимагають продуктивності, масштабованості та залучення додаткових серверів [22].

Найвідомішою моделлю розгортання Cloud Database є інфраструктура як послуга (Infrastructure-as-a-Service, IaaS). Модель IaaS надає доступ до віртуальної платформи з можливістю встановлення та експлуатації своєї бази даних. Вдалим прикладом цієї моделі є сервіс SQL Server on Azure Virtual Machines від Microsoft Azure.

Наступна модель – платформа як послуга (Platform as a Service, PaaS). PaaS дозволяє створювати хмарні бази даних із використанням базової інфраструктури (апаратне забезпечення, операційна система, СУБД), що надається як послуга хмарними провайдерами. PaaS модель характеризується економічною ефективністю, масштабованістю, високою продуктивністю та повною керованістю, що відповідає сучасним вимогам великих компаній. Amazon RDS Service від AWS є гарним прикладом такої моделі Cloud Database.

Нове покоління хмарних баз даних – це безсерверні бази даних (Serverless), які є новітнім трендом в цій галузі. Модель Serverless, яка здатна автоматично запускатися, вимикатися та масштабуватися відповідно до потреб застосунку користувача, пропонує сценарій хмарних обчислень (англ. Cloud Computing). Ця модель є високопродуктивною, масштабованою та з мінімальною затримкою. Найбільш популярними прикладами Serverless є AWS Lambda та GCP Cloud Functions.

Модель програмне забезпечення як послуга (SaaS) використовує принцип Fully-Managed Database Services, коли хмарний провайдер повністю керує

відповідною СУБД, що звільняє користувачів від багатьох трудомістких завдань. SaaS бази даних, такі як Firestore від Google Cloud Provider та Amazon DynamoDB від AWS, характеризуються надійністю, доступністю, гнучкістю та масштабуванням [22].

Для прийняття обґрунтованого рішення щодо вибору структури бази даних необхідно враховувати також їхні характеристики з точки зору складності даних, масштабованості, гнучкості, потреб у запитах і продуктивності, а також галузевих випадків використання [31].

2.3 Функціональні та технологічні вимоги до сучасних систем керування базами даних

Сучасна сфера Data science розглядає різноманітні технології для ефективного керування базами даних, які враховують наступні ключові аспекти, а саме, масштабування, глобальну реплікацію і підвищену консистентність [29].

Масштабування баз даних (Database Scalability) відіграє важливу роль у підтримці високої продуктивності програмних продуктів. Горизонтальне масштабування (Horizontal Database Scalability), на відміну від вертикального масштабування (Vertical Database Scalability), яке зосереджується лише на додаванні оперативної пам'яті (більш потужних компонентів обладнання) до існуючого сервера, передбачає залучення нових ресурсів (нових вузлів / серверів) для забезпечення достатньої потужності IT-системи [33]. Це дозволяє ефективно розподілити робоче навантаження, значно скоротивши час обробки великого обсягу даних при виконанні складних завдань. Але для реалізації технологій горизонтального масштабування слід враховувати необхідність значних початкових інвестицій у додаткові сервери.

Реплікація (розподіл) баз даних (Data Replication) дозволяє зберігати копії необхідних даних на різних вузлах мережі, що забезпечує високу доступність, швидкий пошук даних та підвищену стійкість до відмов. Доступ у будь-який час мінімізує простої системи.

В свою чергу, консистентність (узгодженість) даних (Data Consistency) передбачає їх цілісність, а також внутрішню несуперечність, тобто, визначення певних атрибутів, зв'язків між вузлами. Захист цілісності даних забезпечує їх актуальність і точність. Це є одним з важливих показників ефективності управління базами даних.

Будь-яка сучасна СУБД має виконувати обов'язкову низку вимог:

– забезпечувати несуперечність даних, їх цілісність (пов'язані дані мають адекватно змінюватися);

– гарантувати актуальність даних та їх доступність;

– уникати надмірності даних (дані мають бути захищені від несанкціонованого доступу, зміни чи видалення);

– надавати можливість відновлення та резервного копіювання [16].

Також існують певні правила взаємодії, які називаються моделлю узгодженості ACID [37]. Стандарт ACID (Atomicity, Consistency, Isolation, Durability), з яким сумісні реляційні (SQL) та деякі NoSQL бази даних, наприклад, MongoDB, забезпечує високий рівень атомарності (нерозривності), узгодженості, ізольованості та довговічності транзакцій [18].

Атомарність (Atomicity, нерозривність) пов'язана з неподільним, послідовним, обов'язковим виконанням серії операцій в одному повному циклі (одна логічна операція над даними). Лише такі умови можуть гарантувати забезпечення цілісності бази даних, ефективність транзакції та стабільність роботи системи.

Узгодженість (Consistency) – вимога, за якої система має перебувати в несуперечливому стані з моменту початку дії транзакції і до її завершення.

Ізольованість (Isolation) забезпечує недоступність проміжних результатів (змінних) до остаточного завершення транзакції.

Довговічність (Durability, стійкість) гарантує у будь-якому випадку збереження результатів завершених транзакцій (дані зберігаються та відновлюються у випадку збою системи) [21; 42].

Для реалізації програмних завдань використовують різноманітні технології ефективного керування базами даних. Кожна технологія має свої функції, набір інструментів та умови успішного застосування.

2.4 Методологія дослідження SQL і NoSQL баз даних

Більшість прикладних досліджень стосуються вибору найкращого рішення для бази даних між структурами даних SQL і NoSQL. Вирішення проблем масштабування, узгодженості, захисту цілісності даних є важливими технологічними аспектами, що заохочують до пошуку нових сценаріїв вирішення в системі.

Бази даних SQL найбільш бажані в роботі зі структурованими даними за попередньо визначеною схемою. Ця система керування базами даних використовує мову структурованих запитів (SQL), яка підтримує стандарт ACID, що забезпечує високу надійність керування транзакціями. Реляційні (SQL) бази даних Oracle, MySQL, PostgreSQL і Microsoft SQL Server визнані найбільш привабливими, оскільки дозволяють зменшити ймовірність неправильної поведінки системи та забезпечити цілісність бази даних, але за умови вибору роботи за стандартом ACID (Atomicity, Consistency, Isolation, Durability) [17]. Відповідність вимогам ACID потрібна для обробки конфіденційних фінансових, медичних та особистих даних. Дотримання вимог стандарту ACID автоматично забезпечує конфіденційність користувачів. Тому SQL (реляційні) бази даних є найкращим рішенням для розробки і підтримки фінансових, медичних проектів, та для роботи з даними де потрібні права доступу [42].

Робота з реляційними базами даних передбачає виконання певних запитів і операцій з даними (табл. 2.3).

Таблиця 2.3 – Види SQL запитів

SQL запити		Призначення	Приклади
DDL (Data Definition Language)	мова визначення даних	створення БД та опис її структури (в якому вигляді різні дані будуть розміщуватися в БД)	ALTER COLLATE CREATE DROP DISABLE TRIGGER ENABLE TRIGGER RENAME UPDATE STATISTICS
DML (Data Manipulation Language)	мова маніпулювання даними	потрібні для додавання змін до вже внесених даних, для отримання даних з БД, для їх збереження, для оновлення різних записів і для їх видалення з БД	BULK INSERT SELECT DELETE UPDATE INSERT UPDATETEXT MERGE WRITETEXT READTEXT
DCL (Data Control Language)	мова управління даними	запити та команди, що стосуються дозволів, прав та інших налаштувань СУБД	GRANT REVOKE DENY
TCL (Transaction Control Language)	мова управління транзакціями	застосовують для керування змінами, які здійснюються з використанням DML-запитів (дозволяють проводити об'єднання DML запитів у набори транзакцій)	BEGIN COMMIT ROLLBACK

Дані сформовано з [19]

Окрім цього під час роботи з реляційними базами даних потрібно зважати на те, що будь-які зміни в об’єктах необхідно відображати у структурі таблиць, якщо фізична структура даних не відповідає об’єктній моделі додатка.

Реляційні БД ідеально підходять для роботи зі структурованими даними, структура яких не вимагає частих змін.

Що стосується NoSQL СУБД, зазвичай до них відносять бази даних, які не використовують реляційну модель. NoSQL (нереляційні) бази даних є гнучкими і динамічними, тобто можуть постійно змінюватись. Це, в певній мірі, ускладнює доступ до даних. Але нереляційні СУБД відрізняються значною швидкістю і продуктивністю, та використовуються для зберігання великих обсягів неструктурованої інформації, такої як дані з соціальних мереж, фотографії, MP3-файли тощо [42].

NoSQL бази даних дозволяють зберігати дані будь-яких типів та при необхідності додавати нові. Фізичні об’єкти зберігаються у тому вигляді, в якому з ними потім працює додаток.

Існують різні операційні моделі даних, які пов’язані з концепцією NoSQL. Залежно від типу і способу збереження даних, нереляційні бази даних поділяються на такі групи (табл. 2.4):

Таблиця 2.4 – Типи та приклади NoSQL баз даних

Типи NoSQL баз даних	**Функції та характеристики**	**Способи та сфери використання**	**Приклади**
Бази даних «ключ-значення» (Key-Value Database)	пропонують базові функції для отримання значення, пов’язаного з ключем	для швидкого пошуку інформації за допомогою ключа, для кешування даних (для зберігання коментарів у блогах, оглядів продуктів, профілів користувачів і налаштувань)	Apache Ignite ArangoDB Couchbase FoundationDB InfinityDB MemcacheDB Oracle NoSQL Database OrientDB Redis Velocity
Документно-орієнтовані бази даних (Document-Oriented Database)	зберігають всю інформацію, пов’язану з об’єктом, в одному файлі XML, YAML, JSON, BSON	для систем керування вмістом, швидкого створення прототипів та аналізу даних (у видавничій справі, документальному пошуку тощо)	Apache CouchDB ArangoDB BaseX Couchbase IBM Notes MarkLogic MongoDB OrientDB RethinkDB XML-databases
Стовпчасті бази даних (Column-Oriented Database / Wide-Column Store)	орієнтовані на стовпці, зберігають кожен стовпець як логічний масив значень, характеризуються високою масштабованістю і можуть легко дублюватися, працюють зі структурованими та неструктурованими даними	для обробки аналітичних операцій	BigTable Cassandra Druid HBase Hypertable Qbase ScyllaDB Vertica

Продовження таблиці 2.4

Типи NoSQL баз даних	Функції та характеристики	Способи та сфери використання	Приклади
Графові бази даних / (Graph Database)	використовують структури графів для семантичних запитів, кожен вузол (вершина) є ізольованим документом із даними у довільній формі, вузли з'єднані ребрами, які визначають їхні відношення (властивості)	для проектів із графовими структурами даних (соціальні мережі семантична мережа)	AllegroGraph ArangoDB Datastax Enterprise InfiniteGraph Apache Giraph MarkLogic Neo4j OrientDB Virtuoso Universal Server

Дані сформовано з [36; 42]

Окремим прикладом NoSQL баз даних є мульти-модельні СУБД, які можуть одночасно поєднувати декілька підходів до організації даних. Це в певній мірі розширює алгоритми дій та функціональні можливості під час розробки програмних продуктів з їх використанням.

Мульти-модельні СУБД спроможні в одному запиті працювати з даними із різних типів баз, не порушуючи при цьому узгодженості. Також за рахунок легкої інтеграції нових моделей баз даних в інфраструктуру конкретного програмного продукту надають широкі можливості масштабування [24].

Серед баз даних такого типу визначають Apache Ignite, ArangoDB, Cosmos DB, Couchbase, FoundationDB, InfinityDB, MarkLogic, OrientDB.

Основною перевагою NoSQL СУБД є їх масштабованість. Вони мають розподілену архітектуру і використовують технологію горизонтального масштабування, що забезпечує високу продуктивність та дозволяє автоматично розподіляти дані на різних серверах.

На відміну від реляційних баз даних NoSQL СУБД не мають стандартизованої мови запитів, що може спричиняти певну неузгодженість. Це обмежує підтримку складних запитів та є проблемою, особливо для додатків, які залежать від комплексних служб аналізу даних і звітності.

Порівняльний аналіз реляційних (SQL) та нереляційних (NoSQL) баз даних за певними критеріями дає чітке розуміння алгоритмів реалізації програмних проєктів та сфер їх використання (табл. 2.5).

Таблиця 2.5 – Порівняння реляційних та нереляційних баз даних

Характеристики СКБД	**SQL**	**NoSQL**
Структура	Формат таблиць: складаються з рядків (кортежів) і стовпців (атрибутів)	Формат документу, ключ-значення, графік або сховище у широких стовпцях
Підтримка запитів	Використовує SQL (Structured query language), мову структурованих запитів для взаємодії користувача з базами даних Підтримує JOIN (операцію з'єднання таблиць в SQL) і складні запити	Використовує «віртуальну об'єктну базу даних» ORM (Object-Relational Mapping), об’єктно-реляційне зіставлення Не підтримує JOIN і складні запити
Властивості	Сумісні зі стандартом ACID для забезпечення високої надійності керування транзакціями	Більшість не підтримують стандарт ACID (виключення, наприклад, MongoDB)
Масштабованість	Вертикальне масштабування (Vertical Database Scalability) із розширенням за допомогою більшого сервера	Горизонтальне масштабування (Horizontal Database Scalability) на кількох серверах
Цілі використання	Для роботи зі структурованими даними за попередньо визначеною схемою	Для роботи з неструктурованими (або напівструктурованими) даними з динамічною схемою
Переваги	– просте використання та моделювання даних; – незалежність даних від фізичної структури зберігання; – велика підтримка SQL для запитів і маніпуляції даними; – добре працює з більшістю фреймворків і бібліотек; – стабільність, безпечність, безкоштовність.	– масштабованість (пристосовані до зростаючих робочих навантажень); – гнучкість (дозволяють обробляти дані з різними структурами без обмежень фіксованої схеми); – висока відмовостійкість; – можливість додати більше серверів до кластера.
Недоліки	– неефективно працюють із напівструктурованими або неструктурованими даними; – зменшення продуктивності під час роботи з великими обсягами даних; – працюють лише на одному сервері.	– обмежені транзакції ACID (не підходять для програм, які вимагають суворої цілісності даних); – відсутність стандартизованої мови запитів; – складно організувати зберігання та обробку погано структурованих або зовсім не структурованих даних.

Продовження таблиці 2.5

Характеристики СКБД	SQL	NoSQL
Використання	Фінансові системи, CRM, програмне забезпечення для планування ресурсів підприємства (ERP)	Веб-програми, аналітика в реальному часі, програми IoT, CMS
Приклади	MySQL, Oracle, Microsoft SQL Server, IBM DB2	MongoDB, Cassandra, CouchDB, Redis

Дані сформовано з [29; 36; 42]

При виборі СУБД слід враховувати важливі фактори, які суттєво можуть впливати на ефективність роботи з певним програмним проєктом: типи даних, які потрібно збирати та обробляти, інтеграцію з іншими інструментами, дієвий та доступний спосіб масштабування. Кожна СУБД має свої унікальні особливості та обмеження, застосовується для конкретних сценаріїв. База даних має відповідати ключовим вимогам – бути надійною, безпечною, адаптованою, щоб задовольняти потреби сучасних програмних продуктів і технологій.

На сьогодні завдяки своїй стабільності та безпечності реляційні бази даних залишаються найбільш популярними серед розробників програмних продуктів. Згідно рейтингу DB-Engines реляційні бази даних Oracle, MySQL, Microsoft SQL Server і PostgreSQL займають найвищі позиції [32], нереляційна СКБД MongoDB посіла п’яте місце (рис. 2.6).

DB-Engines Ranking

The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.

Read more about the method of calculating the scores.

423 systems in ranking, December 2024

Rank Dec 2024	Rank Nov 2024	Rank Dec 2023	DBMS	Database Model	Score Dec 2024	Score Nov 2024	Score Dec 2023
1.	1.	1.	Oracle	Relational, Multi-model	1263.79	-53.22	+6.38
2.	2.	2.	MySQL	Relational, Multi-model	1003.76	-14.04	-122.88
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	805.69	+5.88	-98.14
4.	4.	4.	PostgreSQL	Relational, Multi-model	666.37	+12.04	+15.47
5.	5.	5.	MongoDB	Document, Multi-model	400.39	-0.54	-18.76
6.	6.	6.	Redis	Key-value, Multi-model	150.27	+1.63	-8.08
7.	7.	↑ 10.	Snowflake	Relational	147.36	+4.87	+27.48
8.	8.	↓ 7.	Elasticsearch	Multi-model	132.32	+0.68	-5.43
9.	9.	↓ 8.	IBM Db2	Relational, Multi-model	122.78	+1.04	-11.81
10.	10.	↑ 11.	SQLite	Relational	101.72	+2.24	-16.23
11.	11.	↑ 12.	Apache Cassandra	Wide column, Multi-model	97.94	+0.22	-14.26
12.	12.	↓ 9.	Microsoft Access	Relational	90.82	-0.49	-30.93
13.	↑ 14.	↑ 17.	Databricks	Multi-model	87.68	+1.22	+7.37
14.	↓ 13.	14.	Splunk	Search engine	85.36	-3.11	-10.94
15.	15.	↓ 13.	MariaDB	Relational, Multi-model	83.77	+1.07	-16.66
16.	16.	↓ 15.	Microsoft Azure SQL Database	Relational, Multi-model	76.37	-0.16	-6.67
17.	17.	↓ 16.	Amazon DynamoDB	Multi-model	72.73	+0.33	-9.68
18.	18.	18.	Apache Hive	Relational	53.09	+1.59	-16.31
19.	19.	19.	Google BigQuery	Relational	52.29	+1.71	-9.88
20.	↑ 21.	↑ 22.	Neo4j	Graph	43.07	+0.37	-6.92

Рисунок 2.6 – Рейтинг DB-Engines систем керування базами даних за їх популярністю (станом на грудень 2024 року)

Тож, ефективність рішень щодо зберігання та обробки даних безпосередньо залежить від доцільності використання певних систем керування базами даних, які мають вирішальне значення в процесі розробки та обслуговування сучасних програмних продуктів та IT-систем.

2.5 Дослідження популярних рішень СУБД для розробки програмного продукту

2.5.1 Об'єктно-реляційна система управління базами даних Oracle Database

Інновації у сфері технологій систем керування базами даних значно покращили їх функціональні характеристики. Це стосуються таких аспектів, як різні допоміжні структури даних, алгоритми сортування і об’єднання. Компанії-постачальники СУБД підкреслюють переваги продуктивності однієї СУБД над іншою, враховуючи насамперед маркетингову доцільність. Проблема підвищення ефективності керування даними в умовах постійного розвитку ринку IT та запровадження новітніх технологій у різні сфери людської діяльності визначає актуальність дослідження різних моделей СУБД.

Серед розробників програмного забезпечення немає єдиного рішення щодо розробки сценаріїв застосування програмних стеків (компонентів) для досягнення кінцевої мети – створення застосунку високої продуктивності. Дослідження ефективних рішень СУБД для розробки визначеного програмного продукту стосується найбільш популярних технологій, які використовуються у сфері IT.

Список DB-Engines очолює об’єктно-реляційна система управління базами даних Oracle (Oracle Database) від компанії Oracle Corporation як найбільш популярна серед підприємств і державних установ [29; 32; 38].

Перевагами СУБД Oracle є надзвичайна висока продуктивність, масштабованість, безпека, надійність та легкість управління. Це комерційна система керування базами даних, яка зосереджена на системній обробці онлайн-транзакцій (OLTP) та сховищі даних (рис. 2.7).

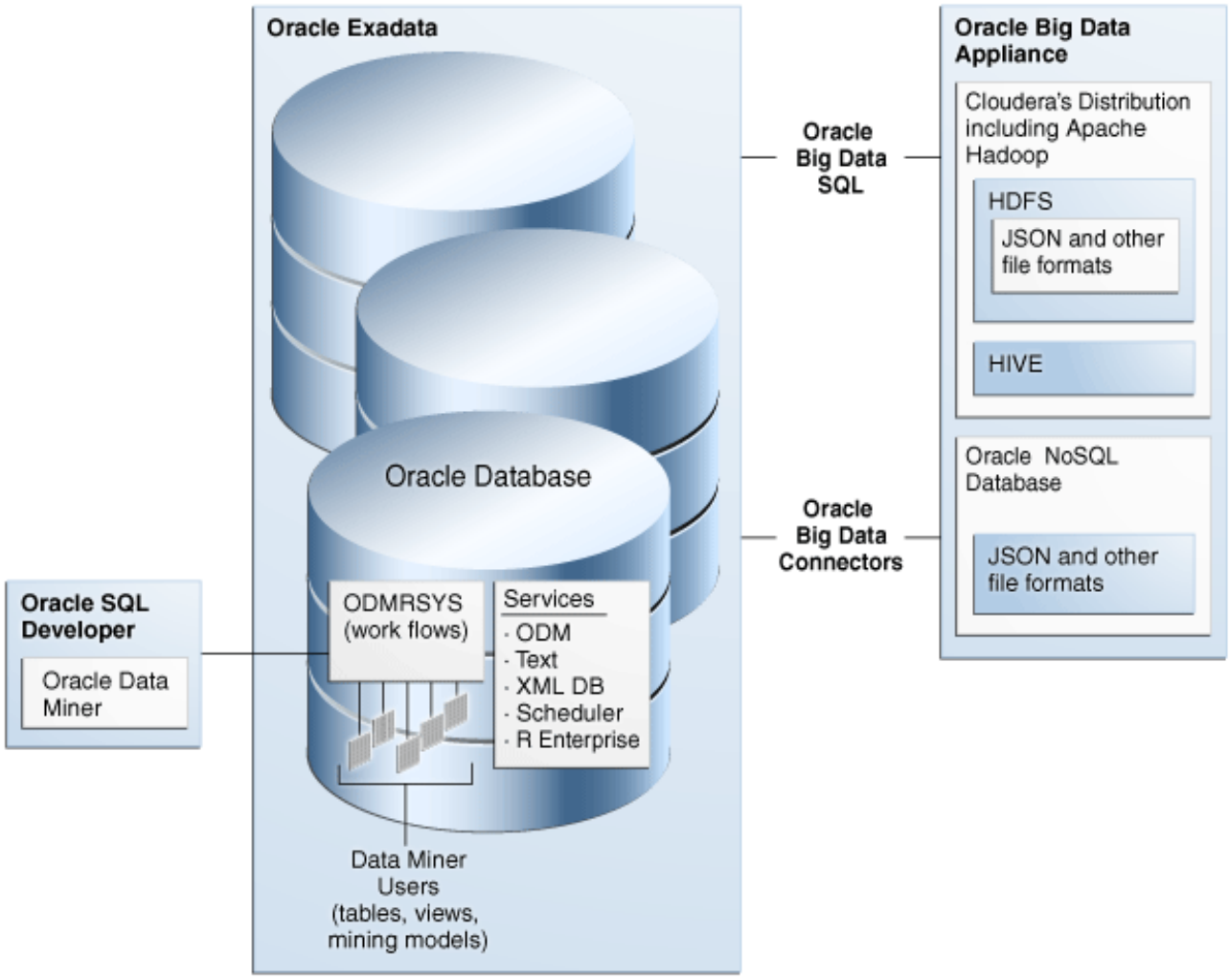

Рисунок 2.7 – Схема архітектури Oracle Data Mining для великих даних

База даних Oracle використовує вдосконалення кількох робочих навантажень, пропонує мульти-модельні зв’язки і покращену обробку даних завдяки інтеграції AutoML для ефективного прийняття рішення. Додатковою перевагою для користувачів є широкий спектр послуг і технічної підтримки, просте масштабування, легка доступність даних, резервування та гнучкість для покращення роботи системи [29; 38].

Oracle Autonomous Database дозволяє спростити середовища реляційних баз даних і зменшити навантаження на керування завдяки хмарній технології Oracle Cloud Infrastructure або локально через Oracle Cloud@Customer та Dedicated Region [34]. Працює з мовами програмування C, C++.

У 2022 році компанію Oracle було визнано лідером Magic Quadrant™ у номінації «Системи керування хмарними базами даних». Також вона набрала найвищий бал у номінації Gartner «Критичні можливості для систем керування хмарними базами даних для операційних прикладів використання» [34].

З 2024 року Oracle пропонує додаткові функціональні розширення, а саме вбудований менеджер відновлення (RMAN) для відновлення даних у разі відмов та збоїв, розширення PL/SQL для процедурного програмування, налаштування кілька екземплярів Oracle на одному сервері [38].

Нещодавно з’явилась нова версія Oracle APEX 23.2, за допомогою якої можна швидко та ефективно розробляти низькокодові застосування корпоративного рівня для всіх відомих платформ і IT-систем [34].

Хоча Oracle доступний, але, перш за все, комерційний проєкт, призначений для великих підприємств і бізнес компаній, та з обмеженим використанням у некомерційних структурах. Безкоштовна версія Express Edition (XE) має оперативну пам’ять лише 1 ГБ і максимальний об’єм бази даних 11 ГБ [16].

2.5.2 Багатофункціональна реляційна база даних MySQL

За рейтингом DB-Engines Ranking (грудень, 2024 року), MySQL посідає друге місце за популярністю, але є найпоширенішою системою управління реляційною базою даних (RDBMS) з відкритим вихідним кодом [32]. Це проста та доступна СУБД, яка має вичерпну документацію та значну підтримку завдяки великій кількості Інтернет ресурсів і спільноті розробників, які працюють з MySQL.

База даних MySQL використовує мову структурованих запитів, що підтримує сумісність з будь-якою Web- та мобільною програмою. Також її можна використовувати для надання послуг PaaS (платформа як послуга), SaaS (програмне забезпечення як послуга) і DBaaS (база даних як послуга) [38; 41].

MySQL має трикомпонентну архітектуру «клієнт-сервер». Архітектура MySQL містить такі рівні (рис. 2.8):

– клієнтський рівень – має важливі служби: обробка підключення; аутентифікація (за допомогою імені користувача та пароля); безпека (перевірка доступу та привілей для виконання певних запитів);
– серверний рівень – відповідає за всі логічні функції системи керування реляційною базою даних;
– рівень зберігання – використовує різні механізми зберігання (InnoDB, MyISAM, NDB, Memory тощо), вважається кращим для розробників.

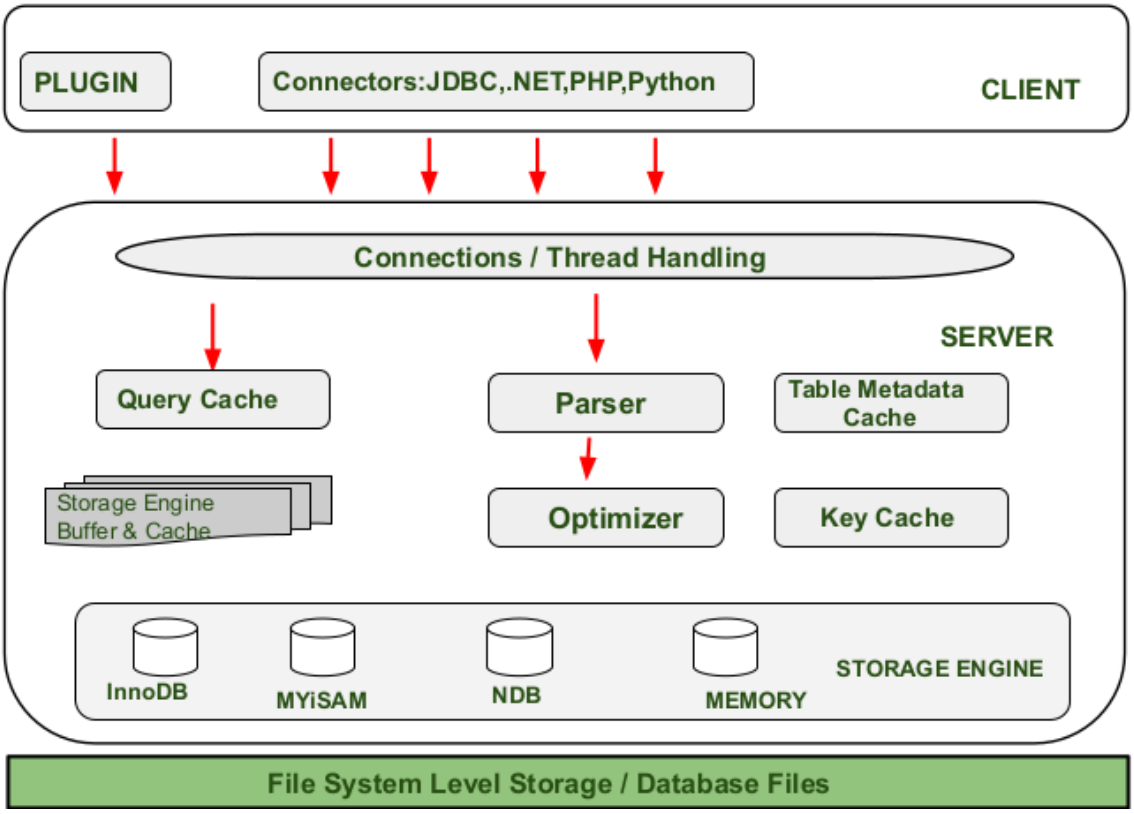

Рисунок 2.8 – Архітектура MySQL

MySQL працює на всіх основних платформах (Red Hat, Fedora, Ubuntu, Debian, Solaris, Microsoft Windows та Apple MacOS) та забезпечує підтримку кількох процесорів та багатопоточність. Через API (Application Programming Interface, інтерфейс програмування додатків) підтримується такими популярними мовами програмування як C, C++, C#, PHP, Java, Ruby, Python та Perl, також

використовується з Novell Cluster для спрощеного управління мережевими ресурсами та пристроями пам'яті [14; 23; 31].

Досвід використання цієї СУБД став підґрунтям для створення певних програм та інструментів, які розширили функціональні можливості MySQL (phpMyAdmin , DBeaver і HeidiSQL) [45].

Основні функції MySQL:

- підтримує стандарт ACID, що забезпечує надійність виконання транзакцій;
- забезпечує продуктивність і швидкість завантаження;
- можна використовувати з будь-якою технологією розробки веб-додатків (.NET , PHP , JavaScript тощо);
- пропонує різні версії відповідно до програмних вимог та потреб користувача;
- забезпечує безпеку даних, збереження їх цілісності та конфіденційності [38; 41].

MySQL має як певні переваги, так і недоліки (табл. 2.6).

Таблиця 2.6 – Порівняльний аналіз переваг і недоліків MySQL

Переваги	Недоліки
Популярність і простота використання Велика кількість документації, технічна підтримка, використання програмних продуктів інших компаній-розробників phpMyAdmin, DBeaver і HeidiSQL для спрощення роботи і розширення функціональних можливостей.	**Ліцензування та пропрієтарні функції** Деякі функції та плагіни доступні лише для пропрієтарних (з обмеженим правом користування) версій.
Безпека та надійність Можливість аутентифікації з паролем та обмеження доступу до даних, визначення пароля для користувача «root», видалення анонімних облікових записів і видалення тестових баз даних, підтримує керування користувачами.	**Функціональні обмеження** Відсутня підтримка FULL JOIN пунктів, не повністю реалізує стандарт SQL.

Продовження таблиці 2.6

Переваги	Недоліки
Швидкість Оптимальна швидкість для обробки великої аналітики та ведення електронної комерції.	**Не часто оновлюється** Процес розробки MySQL значно сповільнився з моменту її придбання корпорацією Oracle у 2009 році.
Реплікація Резервне копіювання бази даних за рахунок обміну інформацією між двома чи більше хостами (для покращення надійності, доступності та відмовостійкості).	**Проблеми з паралельністю** Втрата швидкості програми, якщо багато користувачів записуватимуть дані одночасно.

Дані сформовано з [38; 45]

MySQL найчастіше використовується для розподілених операцій та у роботі з веб-додатками. Ця СУБД успішно працює на багатьох найбільших Web-сайтах, платформах і програмах (Google, Facebook, YouTube, Twitter, Yahoo!, Netflix і Spotify) [45].

Реляційна база даних MySQL є також гарним вибором для навчальних закладів завдяки її структурованому та організованому підходу, узгодженості даних і зв’язками між таблицями.

2.5.3 Microsoft SQL Server

Microsoft SQL Server, програмне забезпечення, що застосовується для керування реляційними базами даних, посідає почесне третє місце у рейтингу DB-Engines Ranking [32]. Окрім стандартних технологій, таких як .NET, ASP.NET, .NET Core, розроблених корпорацією Microsoft, він дозволяє залучати інші технологічні стеки для роботи зі структурованими і неструктурованими даними [38]. Використовуючи середовища Azure SQL Database, Azure Cosmos DB, MySQL та інші інструменти для роботи з даними, Microsoft SQL Server дозволяє відстежувати продуктивність баз даних та керувати сховищем даних на єдиній платформі.

Архітектура SQL Server складає три основні компоненти (рис. 2.9):

– рівень протоколу (підтримує 3 типи протоколу «клієнт-сервер» – протокол загальної пам'яті («клієнт» та «сервер» працюють на одній машині); протокол TCP / IP («клієнт» та «сервер» віддалені один від одного і працюють на окремих машинах); протокол LAN (Local Area Network) – комп'ютерна мережа для обмеженого кола користувачів, що об'єднує комп'ютери в одному приміщенні або в рамках одного підприємства);
– реляційний механізм / процесор запитів (CMD-парсер – отримання даних запиту, оптимізатор – план виконання запитів користувача, виконання запитів);
– система зберігання (визначає типи файлів, метод доступу тощо) [38].

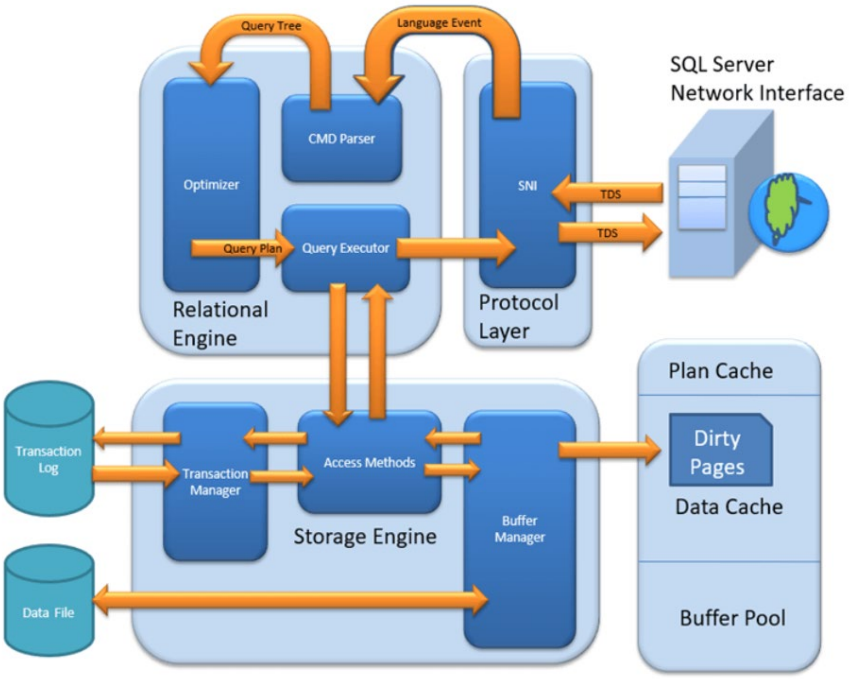

Рисунок 2.9 – Схема архітектури SQL Server

Система управління базами даних Microsoft SQL Server пропонує широкий функціонал для розробки Web-додатків та використовує свою унікальну мову для запитів Transact-SQL, яка була розроблена Microsoft та Sybase. Мова Transact-SQL

реалізує стандарт ANSI / ISO щодо структурованої мови запитів SQL, але має додаткові розширення. Microsoft SQL Server легко масштабується (може розгортатися в настільних системах, у центрах обробки даних, а також у «хмарі»), гарантує високий рівень безпеки і низьку вартість обслуговування [16; 38].

Популярність Microsoft SQL Server безпосередньо пов'язана з його основними і розширеними функціями, які пропонуються у різних версіях SQL Server.

Серед основних функцій:

- онлайн-аналітична обробка, наявність вбудованих служб аналізу для створення і керування OLAP;
- пропонує багатомовну підтримку, підтримує Open Database Connectivity (ODBC) – інтерфейс взаємодії застосунків з СУБД;
- можливість переміщувати, копіювати та перетворювати дані;
- підтримує класифікацію та кластеризацію баз даних (розподіляє робоче навантаження між декількома серверами);
- підтримує автоматичні сповіщення та надлишкове дублювання даних за сценаріями «знімок» бази даних, безперервне формування «історії змін», синхронізація з іншими серверами в процесі звірки даних;
- розширення служб Microsoft Azure для оптимізації продуктивності та зниження витрат на розробку та обслуговування [38].

Корпорація Microsoft пропонує різноманітні версії SQL Server, орієнтуючись на технологічні вимоги щодо розробки сучасних програмних застосунків та розширення цільової аудиторії користувачів:

SQL Server Compact Edition (SQL CE) / Компактне видання – невеликий за обсягом вкладений механізм бази даних (2 Мб для DLL, Dynamic-link library – динамічно приєднуваної бібліотеки).

SQL Server Express Edition – вільно поширювана версія SQL Server, що має деякі технічні обмеження і відсутність графічних інструментів адміністрування. Використовується при проектуванні застосунків та для самостійного вивчення.

Розмір бази даних – 4 Гб, розмір пам'яті (адресованої) – 1 Гб, підтримує лише один процесор.

SQL Server Workgroup Edition – зосереджений на функціональності ядра бази даних без пропозиції додаткових сервісів.

SQL Server Standard Edition – включає основний функціонал баз даних та автономні сервіси без окремих функцій збільшення продуктивності.

SQL Server Enterprise Edition – пропонує розширену багатофункціональну версію SQL Server.

SQL Server Developer Edition – подібний до SQL Server Enterprise Edition, але використовується лише для розробки та тестування.

Серед основних переваг Microsoft SQL Server:

- масштабування системи, що дозволяє при необхідності обробляти великий обсяг запитів;
- автоматизація адміністративних завдань (управління блокуванням, пам'яттю, редагування розмірів файлів, створення профілів користувачів);
- зручний пошук за допомогою ключових фраз, слів, текстів, індексів;
- підтримка роботи з іншими додатками Microsoft (Excel, Access тощо).

Мови програмування, що використовуються з Microsoft SQL Server – C++, C і C Sharp.

Обмеження і недоліки СУБД пов’язані з її повною залежністю від операційної системи (працює лише з Windows) та високою ціною [16; 38].

2.5.4 Об’єктно-реляційна база даних PostgreSQL

PostgreSQL – досить популярна об’єктно-реляційна база даних, яка є переважно реляційною, але містить окремі функції об’єктних баз даних (успадкування таблиць та перевантаження функцій) [45]. Хоча вона повністю безкоштовна, але окремі налаштування, впровадження, модифікація та підтримка все ж платні.

PostgreSQL працює з базами даних будь-якого розміру, також ефективна в роботі з великою кількістю записів та індексів у таблиці, здатна реалізувати складні та надійні механізми транзакцій та реплікації, успішно може обробляти декілька завдань одночасно завдяки Multiversion Concurrency Control (MVCC). Multiversion Concurrency Control є відповідністю ACID. Також PostgreSQL підтримує бітові рядки, мережеві адреси, масиви даних, у тому числі багатовимірні, композитні типи та інші складні структури. В СУБД реалізована підтримка XML, JSON і NoSQL баз даних [16; 45].

PostgreSQL багатофункціональна СУБД, її можна встановлювати на операційних системах Windows, Linux і macOS.

Архітектура PostgreSQL – це модель «клієнт/сервер», яка передбачає отримання запиту від кінцевого користувача, обробку запиту та повернення його клієнту. Для кожного запиту на підключення клієнта запускається новий процес, а запит клієнта обробляється процесом PostgreSQL Server (рис. 2.10).

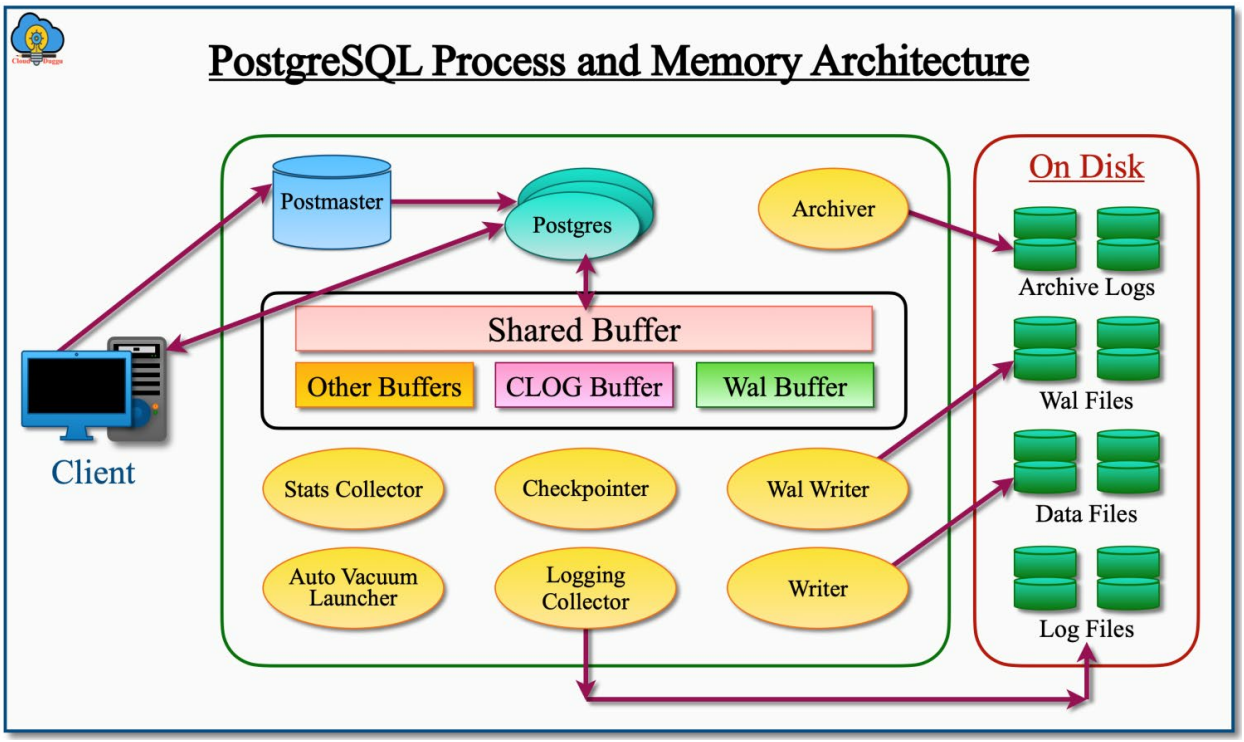

Рисунок 2.10 – Схема процесу «клієнт/сервер» і архітектура пам'яті PostgreSQL

Для спрощення роботи з PostgreSQL використовуються інструменти pgAdmin і Postbird. Сумісні мови програмування – SQL, PL/pgSQL, Python, Perl, Tcl, Java, C, C++, Ruby [29].

PostgreSQL підтримує різні типи даних: числові, рядкові, дати, типи даних часу, геометричних фігур, мережевих адрес, бітових рядків, текстового пошуку та записів JSON, а також окремі специфічні дані [29; 45].

Основні переваги PostgreSQL:

- чітке дотримання стандартів SQL;
- відкритий вихідний код та активна підтримка спільноти розробників PostgreSQL (численні онлайн-ресурси, офіційна документація, різноманітні онлайн-форуми);
- використання кількох мов програмування (без повторної компіляції);
- можливість створювати власні функції та типи даних відповідно до вимог користувача;
- підтримка міжнародних символів, повнотекстовий пошук;
- вбудовані функції аварійного відновлення (PITR – відновлення на певний момент часу);
- за замовчуванням підтримує доступні механізми автентифікації (LDAP, SSPI, SCRAM-SHA-256, GSSAPI тощо);
- використання каталогів та динамічного завантаження для швидкого розширення [38; 45].

Не зважаючи на позитивні сторони, PostgreSQL за останні роки втратив свою популярність. Одною з причин є проблема продуктивності пам’яті. Кожному новому процесу виділяється близько 10 МБ пам’яті, яка може швидко збільшуватися для баз даних із великою кількістю з’єднань, що суттєво впливає на швидкість. Проблеми з продуктивністю та ресурсозатратністю залишаються вирішальними при виборі СУБД. Також PostgreSQL не підходить для важких операцій читання і реалізує функцію реплікації лише за допомогою розширень. У цьому випадку MySQL є більш вдалим вибором. Крім того, на сьогодні існує

небагато адміністраторів баз даних з досвідом керування базою даних PostgreSQL порівняно з іншими СУБД, наприклад MySQL [45].

Не дивлячись на певні недоліки, PostgreSQL активно використовується в таких сферах як інтернет речей (IoT), геймінг та ШІ. Ця база даних є основою для динамічних сайтів та великомасштабних додатків, таких як соціальні мережі. Також забезпечує високу продуктивність при обробці великих обсягів інформації для аналізу даних.

2.5.5 Документо-орієнтована система управління базами даних MongoDB

MongoDB – нереляційна, документо-орієнтована СУБД з відкритим вихідним кодом типу NoSQL. Це система управління базами даних, яка зберігає дані у вигляді колекцій та документів у форматі BSON (JSON) і не потребує опису схеми таблиць типу JOIN [16; 29].

MongoDB використовується для зберігання та управління великими обсягами неструктурованих даних, таких як текстові файли, зображення та відео. Дозволяє масштабувати базу даних в залежності від зростання потреб користувачів і має високу швидкість обробки запитів [29]. Є цікавим та ефективним рішенням для автоматичного масштабування ресурсів оскільки використовує хмарні платформи AWS SDK і Microsoft Azure, а також має власну хмарну версію. Також надає доступ до наскрізного шифрування, і сумісна практично з усіма мовами програмування (Java, C++, C#, PHP, Python, Perl, Ruby) [38]. MongoDB забезпечує розподіл набору даних на декілька частин (шардів) по різних серверах на основі певного ключа (функція шардингу) (рис. 2.11); дозволяє будувати горизонтально масштабований кластер зберігання без точки відмови, щоб забезпечити ефективність роботи бази даних та уникнути наслідки збоїв. В той же час, розширення кластера забезпечується додаванням нових машин.

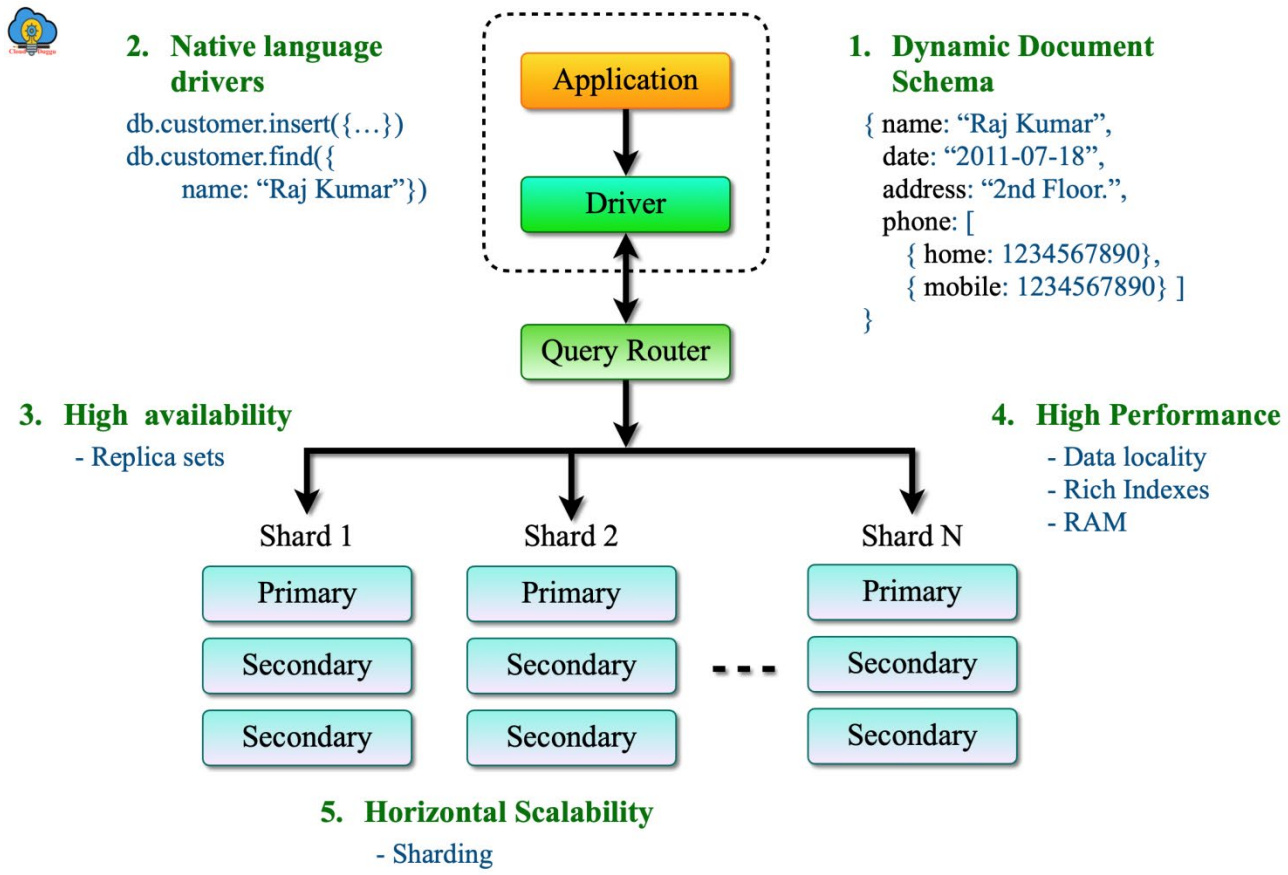

Рисунок 2.11 – Горизонтальне масштабування MongoDB за допомогою шардингу

При виборі MongoDB враховують її переваги:

- забезпечує максимальну доступність даних для користувачів;
- демонструє високу продуктивність і швидкість;
- можливість запуску на кількох серверах одночасно (для резервування під час збоїв);
- зберігання бінарних даних великих обсягів (фото / відео);
- ефективно реалізує ієрархічні зв’язки, демонструє якісну роботу з масивами та іншими структурами даних (підтримка індексів, робота з кількома репліками на різних вузлах, автоматична фрагментація) [38].

Значним недоліком MongoDB та деяких інших NoSQL є проблеми з безпекою і втратою даних (окремі випадки втрати контролю над серверами, хакерські атаки).

Зазвичай MongoDB обирають, коли є необхідність у базі даних, що масштабується. Вона успішно застосовується у сфері електронної комерції, для систем управління контентом, з різноманітними мобільними додатками і програмами тощо.

Систему управління базами даних MongoDB використовують такі великі корпорації, як Forbes, Facebook, Google, IBM, Twitter, Zendesk та інші [16].

2.6 Обґрунтування програмних технологій для розробки програмного засобу

Ефективність рішень щодо зберігання та обробки даних безпосередньо залежить від доцільності використання певних систем керування базами даних, які мають вирішальне значення в процесі розробки та обслуговування сучасних програмних продуктів та IT-систем.

В процесі проведених досліджень визначено сценарій застосування програмних технологій для реалізації проєкту – розробки системи обліку педагогічних та науково-педагогічних працівників. Вирішення проблем оптимізації функціональних можливостей подібної системи обліку доводить практичну значущість отриманих результатів.

Попередній аналіз СУБД демонструє надійність та продуктивність саме реляційних баз даних у порівнянні з NoSQL моделями. Серед популярних SQL СУБД на основі їх ґрунтовного дослідження була обрана система управління базами даних MySQL, оскільки вона має багато переваг: забезпечує надійність виконання транзакцій; продуктивність і швидкість завантаження; наявність змістовної документації та підтримки; має просту і зрозумілу структуру, забезпечує підтримку складних запитів, пропонує різноманітні технологічні версії (успішно працює з мовами програмування C, C++, C#, PHP, Java, Ruby, Python; розширює функціонал з інструментами phpMyAdmin, DBeaver і HeidiSQL; застосовує Novell Cluster для спрощеного управління мережевими ресурсами та пристроями пам'яті тощо). Головна перевага MySQL – гарантія безпеки даних,

збереження їх цілісності та конфіденційності. NoSQL СУБД мають обмежену підтримку складних запитів та менш надійні, хоча здатні працювати з неструктурованими даними з динамічною схемою і пропонують горизонтальне масштабування на кількох серверах.

Бази даних освітньої галузі, які можуть використовуватись для обліку педагогічних та науково-педагогічних працівників, або здобувачів освіти, у більшості випадків структуровані, зручні для табличного формату, і часто потребують конфіденційності. Тож реляційні моделі систем управління базами даних є ефективним рішенням для розробки освітніх програмних продуктів, платформ для організації дистанційного навчання та систем управління контентом.

MySQL пропонує широкий спектр технологічних рішень щодо використання програмного забезпечення. Серед популярних програмних стеків, які здатні успішно реалізувати програмний проєкт – Apache, MySQL / MariaDB, Perl / PHP / Python. Тож, для розробки програмного засобу – системи обліку педагогічних та науково-педагогічних працівників, було обрано такі технології, як кросплатформений HTTP-сервер Apache, систему управління базами даних MySQL, скриптову програмну мову PHP, динамічну мову JavaScript для створення сценаріїв Web-сторінок та редактор початкового коду Microsoft Visual Studio Code.

3 ОГЛЯД ТЕХНОЛОГІЙ ДЛЯ ПРОЕКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ ОБЛІКУ НАУКОВО-ПЕДАГОГІЧНИХ ПРАЦІВНИКІВ

3.1 Інструментальні засоби розробки програмного забезпечення

3.1.1 Редактор початкового коду Microsoft Visual Studio Code

Для розробки системи обліку було обрано середовище розробки Microsoft Visual Studio Code (VS Code). Visual Studio Code (VS Code) – спрощений редактор вихідного коду, який доступний для операційних систем Windows, macOS і Linux, що містить базову підтримку для більшості поширених мов програмування, здатен підтримувати JavaScript, TypeScript і Node.js і має додаткові розширення для C++, C#, Java, Python, PHP і середовищ.NET та Unity.

Серед основних функціональних можливостей Visual Studio Code:

- підсвічування синтаксису – основні структурні елементи коду виділяються різними кольорам, такі як функції, класи, змінні тощо;
- автоматичне доповнення – пропонуються варіанти завершення рядка;
- контроль версій – інтеграції з GitHub, GitLens, іншими сервісами;
- відлагодження (debugging) – виявлення помилок та варіанти виправлення, також підтримується режим відлагодження коду, наприклад, для JavaScript відкриється нове вікно браузера;
- рефакторинг – рекомендації щодо оптимізації коду та його ефективності;
- швидкість – стабільність роботи на пристроях з оперативною пам'яттю 1 ГБ та частотою процесора від 1,6 ГГц [20].

Visual Studio Code пропонує ефективний інструментарій IntelliSense для JavaScript, TypeScript, JSON, CSS і HTML. Також є можливість використовувати стандартні комбінації клавіш, змінювати теми, встановлювати параметри розширень.

Тож Visual Studio Code є універсальним інструментом, який використовується для розробки системних програмних продуктів.

Інтерфейс «вітального вікна» (рис. 3.1) Visual Studio Code містить посилання на певні розділи і функції редактора, серед яких є розділ «Start» (рис. 3.2) для створення нових файлів, папок, перегляду їх вмісту, «Recent», що містить посилання на попередньо створені файли та папки, «Help», в якому є корисні посилання та інформація про стандартні комбінації клавіш Visual Studio Code тощо.

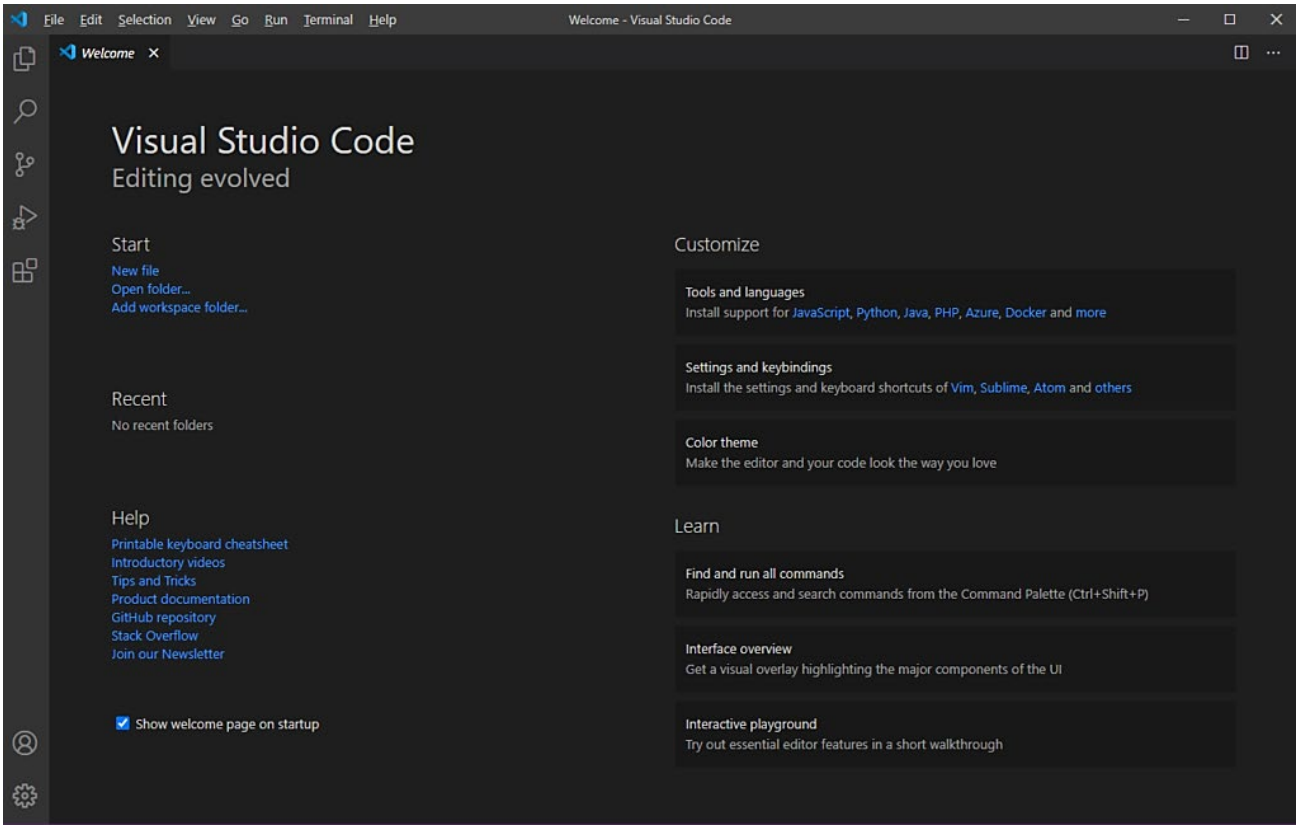

Рисунок 3.1 – Інтерфейс вітального вікна Visual Studio Code

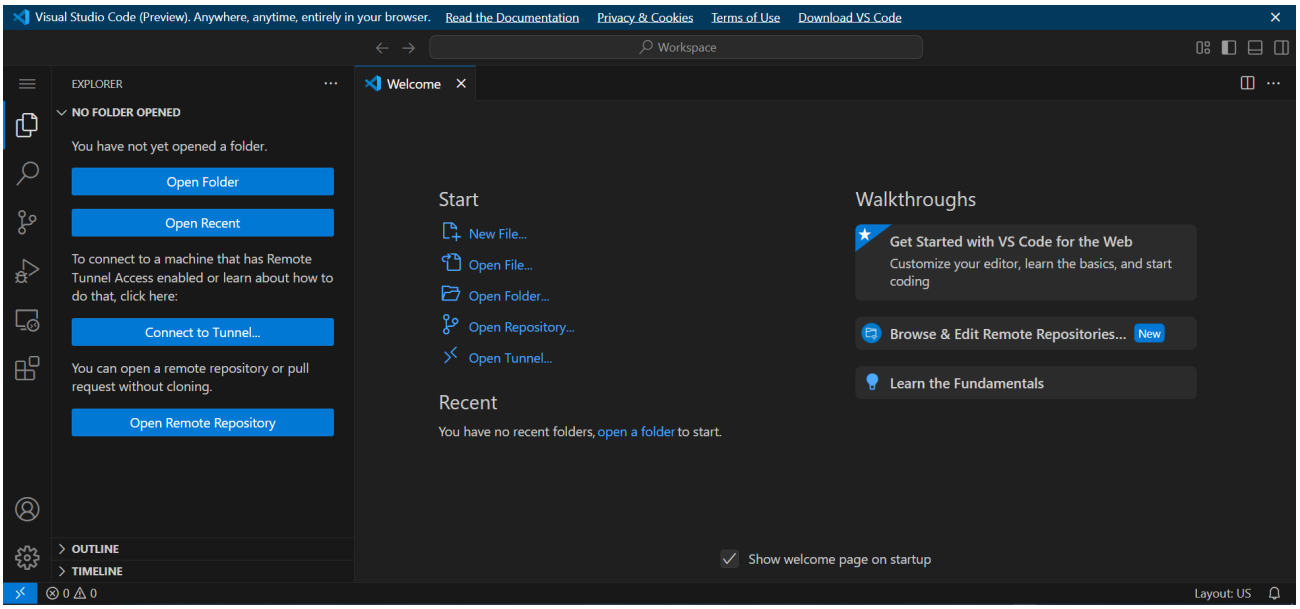

Рисунок 3.2 – Розділ «Start» Visual Studio Code

Серед базових (основних) комбінацій клавіш для організації ефективної роботи з редактором Visual Studio Code є наступні (табл. 3.1):

Таблиця 3.1 – Список основних комбінацій клавіш Visual Studio Code

Комбінації клавіш	**Функції**
[Ctrl]+[Shift]+[P], [F1]	надає можливість відкрити панель команд
[Ctrl]+[P]	дозволяє перейти до файлу
[Ctrl]+[Shift]+[N]	відкриває нове вікно
[Ctrl]+[Shift]+[W]	дозволяє закрити вікно
[Ctrl]+[,]	користувацькі налаштування
[Ctrl]+[K]+[Ctrl]+[S]	доступ до списку усіх комбінацій клавіш

Дані сформовано з [20]

Вибір редактора Visual Studio Code обумовлений такими важливими перевагами:

– простою та зрозумілою конфігурацією;

– функцією відкритого вихідного коду;

- розширенням для FTP (безкоштовна альтернатива для веб-розробки);
- можливістю генерації коду під платформу .NET Framework;
- наявністю необхідних стандартних компонентів та додаткових бібліотек для розширення функціоналу;
- достатньо високою швидкістю та надійністю роботи;
- популярністю [20, 26].

3.1.2 XAMPP Control Panel – кросплатформений пакет рішень веб-сервера

Для реалізації розробки програмного засобу – системи обліку педагогічних та науково-педагогічних працівників, необхідний надійний HTTP веб-сервер. Кросплатформений HTTP-сервер Apache з відкритим вихідним кодом – є найпопулярнішим рішенням. Він пропонує безліч функціональних можливостей та надає підтримку таким мовам програмування як Perl, Python, Tcl та PHP.

Оскільки в процесі досліджень одним з технологічних рішень була обрана система управління базами даних MySQL, здійснено пошук інструментів здатних узгодити обрані програмні технології (PHP, MySQL, HTTP-сервер Apache).

Кросплатформений пакет XAMPP Control Panel дозволяє успішно встановити HTTP-сервер Apache, СУБД MySQL та PHP для розробки програмного засобу (рис. 3.3). Адреса офіційного сайту XAMPP – https://www.apachefriends.org/.

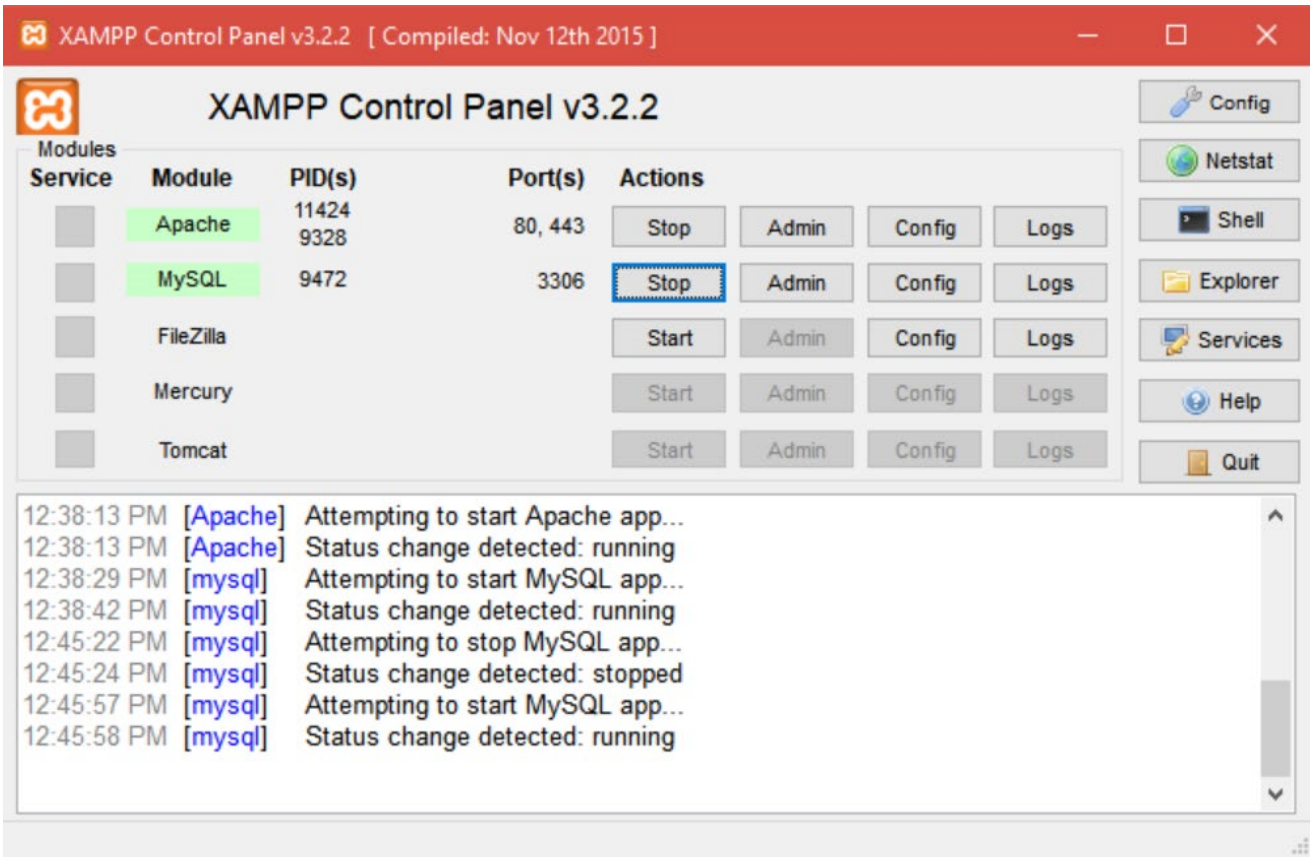

Рисунок 3.3 – Вікно налаштувань XAMPP Control Panel

XAMPP – є безкоштовним кросплатформним пакетом рішень веб-сервера з відкритим вихідним кодом, який був розроблений Apache Friends [46]. Він приваблює простим, зрозумілим для користувача інтерфейсом та дозволяє суттєво скоротити час для встановлення та налаштування окремих програмних компонентів (Apache, MySQL, PHP). XAMPP однаково добре працює на популярних платформах (Windows, Linux, Mac OS), дозволяє встановити такі інструменти, як PHPmyAdmin, OpenSSL, PERL або Webalizer, і вести розробку на окремому комп'ютері без підключення до мережі без потреби в хостингу.

Сама назва XAMPP, яка є абревіатурою (X – кросплатформений, працює на всіх операційних системах, Apache, MySQL, PHP та Perl), розкриває можливості цього проєкту [46]. Цей пакет програмних інструментів був обраний не випадково. Apache – надійний веб-сервер, що підтримує основні версії ОС. MySQL – найвідоміша, популярна та надійна система керування базами даних.

PHP – скриптова мова програмування для створення Web-сайтів (найвідоміші – Facebook та Yahoo!), яка підтримується майже всіма хостинг-провайдерами. Perl – легка та доступна мова програмування, яка підтримує модулі інших сторонніх розробників.

XAMPP не бажано використовувати як робочий сервер через проблеми з надійністю. Але якщо є потреба зробити сервер доступним в мережі, для часткового блокування розробниками Apache Friends пропонуються відповідні рекомендації та налаштування.

3.1.3 Функціональні можливості скриптової мови PHP

Важливим аспектом технологічних рішень при розробці програмного продукту є мова програмування. Серед переліку базових програмних компонентів для розробки системи обліку була обрана скриптова мова програмування PHP. Її основна перевага – це практичність, гнучкість та ефективність для вирішення різноманітних програмних завдань.

PHP – це скриптова мова загального призначення, яка широко застосовується для розробки веб-додатків. Хоча на сьогодні існує багато інших програмних мов, що можуть успішно конкурувати з нею, PHP є одним з лідерів, що застосовуються для створення динамічних веб-сайтів. Вона має відкритий вихідний код, простий зрозумілий синтаксис, який частково схожий на Java та C++, і вбудовані бібліотеки для роботи з популярними базами даних (MySQL, PostgreSQL, SQLite, mSQL, Oracle, Hyperware, Informix, InterBase та інші).

Традиційно PHP використовується у поєднанні з HTML-кодом, також може інтегруватися в JavaScript, WML, XML та інші мови Інтернет-програмування.

Серед важливих функціональних характеристик PHP є наступні:

- висока продуктивність;
- підтримка кількох баз даних (MySQL, Oracle, PostgreSQL тощо);

- сумісність між платформами (Windows, Mac, Linux і Unix);
- доступність на будь-яких пристроях;
- гнучкість, можливості інтеграції з JavaScript, HTML, XML;
- сумісність з різними серверами (IIS, Apache тощо);
- доступ до даних про використання процесора та пам’яті;
- трирівнева архітектура, яка лінійно працює на сервері, системі баз даних і браузері (для безпечних онлайн-транзакції);
- розширена підтримка різних баз даних та взаємодія з різними службами через протоколи (IMAP, POP3, HTTP, COM, SNMP тощо);
- інтеграція з багатьма бібліотеками (ImageMagick, бібліотека GD, Imagine та іншими додатками на основі PHP);
- розширені додаткові можливості (пропонує магічні константи, регулярні вирази, клас PDO, підтримує файли cookie, виконання командного рядка оболонки тощо) [44].

Також PHP надійно керує текстовим вмістом, графічним дизайном і обробкою зображень.

Програмна мова PHP постійно вдосконалюється розробниками. Нещодавно з’явилась нова розширена версія PHP 8.4.1., яка пропонує додаткові функціональні налаштування (адреса офіційного сайту PHP – https://www.php.net/) (рис. 3.4).

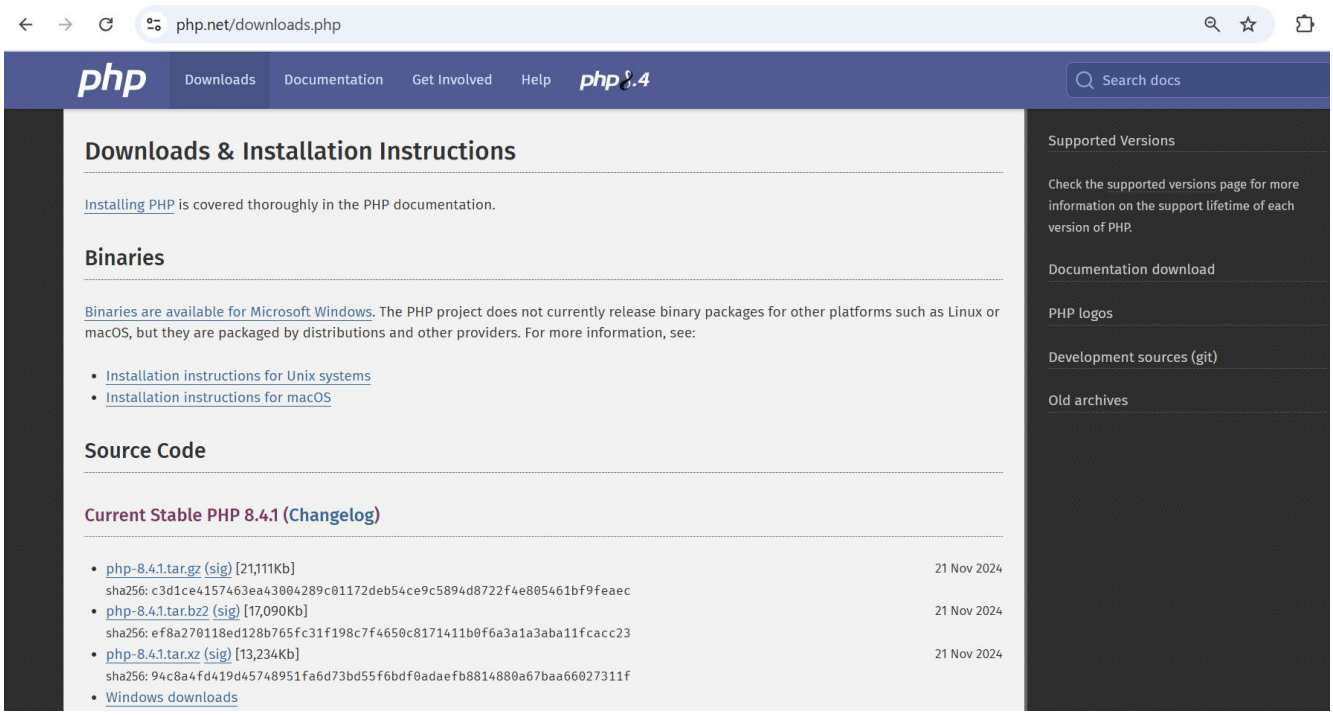

Рисунок 3.4 – Сайт завантаження PHP

PHP широко застосовується для розробки систем керування Web-контентом та освітніх платформ (WordPress, Drupal, Joomla, Moodle та інші).

Серед популярних Web-сайтів, які використовують PHP, WordPress.com, Facebook.com, Wikipedia.org, Zoom.us, Microsoft.com, Canva.com, Salesforce.com та багато інших [44].

Тож, представлені переваги і функціональні характеристики роблять програмну мову PHP незамінною для розробки програмних додатків у сфері освіти, наприклад, систем обліку науково-педагогічних працівників.

3.1.4 Програмна мова JavaScript для створення сценаріїв Web-сторінок

Для створення сценаріїв Web-сторінок потрібна програмна мова, що підтримується сучасними браузерами. JavaScript – це динамічна, об'єктно-орієнтована прототипна мова програмування, яка повністю інтегрована з HTML/CSS. Напрямки використання JavaScript різноманітні:

- забезпечує інтерактивність Web-сторінок;
- застосовується для програмування на боці сервера (Node.js (Express.js));
- використовується для стаціонарних застосунків (Electron, NW.js) та мобільних застосунків (React Native, Cordova)
- поширена в прикладних програмах (Adobe Creative Suite, Apache JMeter);
- використовується в PDF-документах.

В процесі роботи з Web-сторінками надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд Web-сторінки [28].

JavaScript може виконуватися не лише в браузері, але й на сервері чи іншому пристрої за допомогою спеціальної програми «рушій JavaScript» [28].

Функціонал JavaScript може змінюватись залежно від середовища. Платформа з відкритим кодом Node.js дозволяє JavaScript читати та записувати довільні файли, здійснювати мережеві запити та виконувати інші функції [28].

Серед основних функцій об'єктно-орієнтованої прототипної мови JavaScript наступні:

- можливість додавати новий HTML-код на сторінку, змінювати наявний вміст та стилі;
- надсилати запити через мережу до віддалених серверів, завантажувати та відвантажувати файли (технології AJAX і COMET);
- функції керування діями користувача (опрацьовує натискання миші, переміщення вказівника, натискання на клавіші клавіатури);
- робота з Cookie, запитання відвідувачам, показ повідомлень;
- можливість запам’ятовувати дані на стороні клієнта, які будуть доступні в майбутніх сесіях на цьому Web-сайті [28].

Мову JavaScript вважають багатофункціональною, оскільки крім браузера вона також може використовуватись в інших середовищах. На сьогодні це найбільш популярна мова програмування на IT-ринку.

3.2 Створення бази даних для системи обліку науково-педагогічних працівників

Для узгодження і візуалізації процесів розробки програмного засобу системи обліку науково-педагогічних працівників була спроєктована база даних. Структура відповідної бази даних складається із семи основних таблиць (рис. 3.5):

- «вчителі» / «Teachers» (таблиця 3.2);
- «кафедри» / «Departments» (таблиця 3.3);
- «факультети» / «Faculties» (таблиця 3.4);
- «дипломи» / «Diplomas» (таблиця 3.5);
- «сертифікати» / «Certificates» (таблиця 3.6);
- «курси» / «Courses» (таблиця 3.7);
- «організації» / «Organizations» (таблиця 3.8).

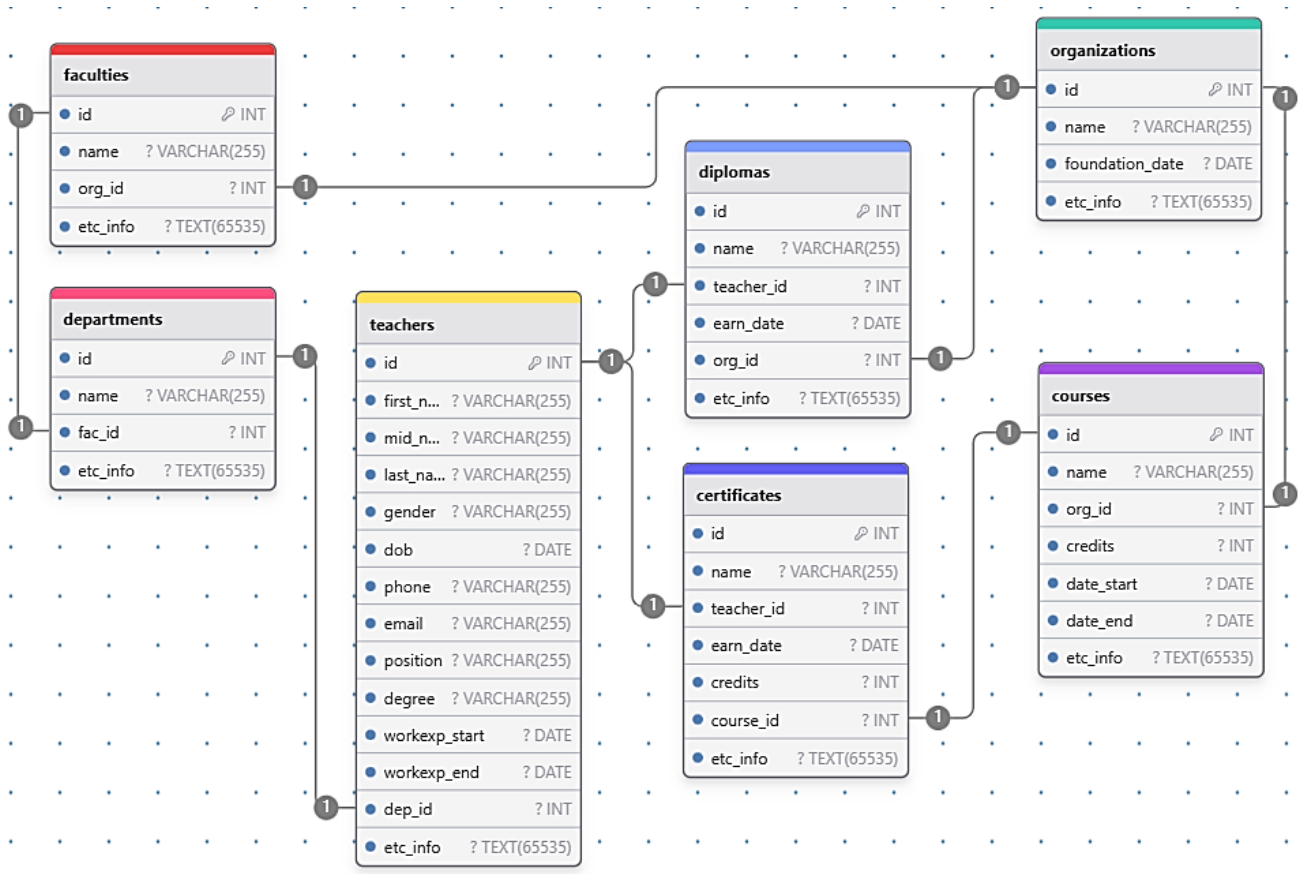

Рисунок 3.5 – Схема бази даних

Таблиця під назвою «Teachers» (таблиця 3.2) містить усю необхідну інформацію щодо науково-педагогічних працівників, які зареєстровані в даній системі та задіяні в проходженні курсів і отриманні дипломів та сертифікатів.

Таблиця 3.2 – Структура таблиці «Teachers»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
id	int	255	Унікальний ID працівника	Первинний ключ
first_name	varchar	255	Ім’я	Обов’язковий
mid_name	varchar	255	По батькові	Обов’язковий
last_name	varchar	255	Прізвище	Обов’язковий
gender	varchar	255	Стать	Обов’язковий

Продовження таблиці 3.2

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
dob	date	255	Дата народження	Обов’язковий
phone	varchar	255	Номер телефону	Обов’язковий
email	varchar	255	Електронна пошта	Обов’язковий
position	varchar	255	Посада	Обов’язковий
degree	varchar	255	Вчене звання	Обов’язковий
workexp_start	date	унікальна	Час початку нарахування педагогічного стажу	Обов’язковий
workexp_end	date	унікальна	Час останнього нарахування педагогічного стажу	Обов’язковий
dep_id	int	255	Унікальний ID кафедри	Зовнішній ключ, що посилається на первинний ключ відношення Departments. Обов’язковий
etc_info	text	65535	Додаткова інформація	Факультативний

У наступній таблиці «Departments» (таблиця 3.3) міститься загальна інформація про кафедри університету (підрозділи освітнього закладу), до яких належать науково-педагогічні працівники.

Таблиця 3.3 – Структура таблиці «Departments»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Id	int	255	Унікальний ID кафедри	Первинний ключ
Name	varchar	255	Назва кафедри	Обов’язковий

Продовження таблиці 3.3

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
fac_id	int	255	Унікальний ID факультету	Зовнішній ключ, що посилається на первинний ключ відношення Faculties. Обов’язковий
etc_info	varchar	255	Додаткова інформація	Факультативний

Таблиця «Faculties» (таблиця 3.4) містить відповідну інформація про факультети, до яких належать кафедри й відповідні науково-педагогічні працівники.

Таблиця 3.4 – Структура таблиці «Faculties»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Id	int	255	Унікальний ID факультету	Первинний ключ
Name	varchar	255	Назва факультету	Обов’язковий
org_id	int	255	Унікальний ID організації (університету в цьому контексті)	Зовнішній ключ, що посилається на первинний ключ відношення Organizations. Обов’язковий
etc_info	varchar	255	Додаткова інформація	Факультативний

Інформація щодо наявності відповідних дипломів, які засвідчують здобуття освітнього (кваліфікаційного) ступеню педагогічними та науково-педагогічнми працівниками, міститься у таблиці «Diplomas» (таблиця 3.5).

Таблиця 3.5 – Опис функціональності класів системи

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Id	int	255	Унікальний ID диплому	Первинний ключ
Name	varchar	255	Назва диплому	Обов’язковий
teacher_id	varchar	255	Унікальний ID працівника	Зовнішній ключ, що посилається на первинний ключ відношення Teachers. Обов’язковий
earn_date	date	унікальна	Час отримання диплому	Обов’язковий
org_id	int	255	Унікальний ID організації (університету в цьому контексті)	Зовнішній ключ, що посилається на первинний ключ відношення Organizations. Обов’язковий
etc_info	text	65535	Додаткова інформація	Факультативний

У таблиці «Certificates» (таблиця 3.6) міститься вся необхідна інформація про сертифікати педагогічних та науково-педагогічних працівників (реєстраційний номер, кількість кредитів тощо).

Таблиця 3.6 – Структура таблиці «Certificates»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Id	int	255	Унікальний ID сертифікату	Первинний ключ
Name	varchar	255	Назва сертифікату	Обов’язковий

Продовження таблиці 3.6

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
teacher_id	int	255	Унікальний ID працівника	Зовнішній ключ, що посилається на первинний ключ відношення Teachers. Обов’язковий
earn_date	date	унікальна	Час отримання сертифікату	Обов’язковий
Credits	int	255	Кількість кредитів	Обов’язковий
course_id	int	255	Унікальний ID курсу	Зовнішній ключ, що посилається на первинний ключ відношення Courses. Обов’язковий
etc_info	varchar	255	Додаткова інформація	Факультативний

Таблиця «Courses» (таблиця 3.7) надає детальну інформацію про курси, які пройшли науково-педагогічні працівники.

Таблиця 3.7 – Структура таблиці «Courses»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Id	int	255	Унікальний ID курсу	Первинний ключ
Name	varchar	255	Назва курсу	Обов’язковий
teacher_id	int	255	Унікальний ID організації	Зовнішній ключ, що посилається на первинний ключ відношення Organizations. Обов’язковий

Продовження таблиці 3.7

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
Credits	int	255	Кількість кредитів	Обов’язковий
date_start	date	унікальна	Час початку проходження курсу	Обов’язковий
date_end	date	унікальна	Час закінчення проходження курсу	Обов’язковий
etc_info	varchar	255	Додаткова інформація	Факультативний

В таблиці «Organizations» (таблиця 3.8) є наявна інформація щодо організацій, які відповідальні за проведення курсів, видачу дипломів та включають навчальні заклади, до складу яких належать науково-педагогічні працівники.

Таблиця 3.8 – Структура таблиці «Organizations»

Ім’я стовпця	Тип	Довжина	Призначення	Обмеження цілісності стовпців
id	int	255	Унікальний ID організації	Первинний ключ
name	varchar	255	Назва організації	Обов’язковий
foundation_date	date	унікальна	Дата заснування організації	Обов’язковий
etc_info	varchar	255	Додаткова інформація	Факультативний

3.3 Реалізація розробки системи обліку науково-педагогічних працівників

3.3.1 Розробка програмного забезпечення

В процесі розробки системи обліку науково-педагогічних працівників СОД були спроєктовані та розроблені наступні класи (рис. 3.6):

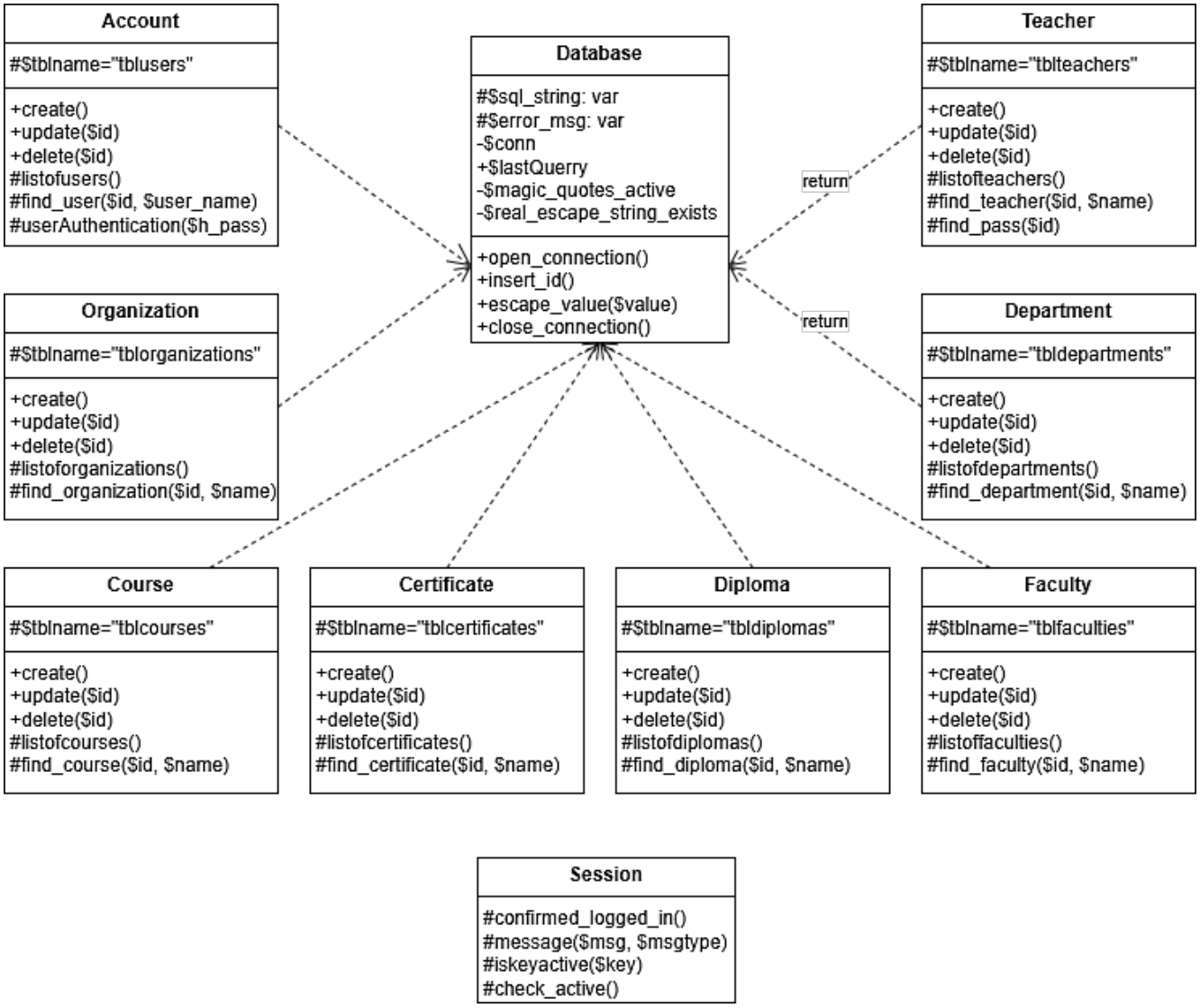

Рисунок 3.6 – Діаграма класів

- *Database* – слугує для забезпечення взаємодії з базою даних для керування з’єднанням з нею;
- *Session* – забезпечує взаємодію з базою даних для керування онлайн-сесіями користувачів в системі;
- *Account* – забезпечує створення, редагування, оновлення та пошук інформації щодо акаунтів користувачів, їх активності та планування проходження курсів;
- *Faculty* – призначений для створення, редагування, оновлення та пошуку інформації щодо факультетів (підрозділів);
- *Department* – надає можливість створення, редагування, оновлення та пошуку інформації щодо кафедр;
- *Teacher* – слугує для створення, редагування, оновлення та пошуку інформації щодо педагогічних та науково-педагогічних працівників, їх кваліфікації;
- *Diploma* – надає можливість створення, редагування, оновлення та пошуку інформації щодо дипломів, які належать певним науково-педагогічним працівникам;
- *Certificate* – призначений для створення, редагування, оновлення та пошуку інформації щодо сертифікатів педагогічних та науково-педагогічних працівників;
- *Course* – призначений для створення, редагування, оновлення та пошуку інформації щодо курсів;
- *Organization* – призначений для створення, редагування, оновлення та пошуку інформації щодо організацій, які забезпечують проходження курсів.

Опис функціональності даних класів наведено в таблиці 3.9.

Таблиця 3.9 – Опис функціональності класів розробленої системи обліку

Класи	Назва функції	Короткий опис
Database		Взаємодія з БД у контексті управління з'єднанням
	public open_connection()	Відкриття поточного підключення
	public insert_id()	Отримання останнього ідентифікатора, вставленого через поточне підключення до бази даних
	public escape_value($value)	Екранування спеціальних символів
	public close_connection()	Закриття поточного підключення
Session		Взаємодія з БД у контексті управління сесіями користувачів
	public confirm_logged_in()	Підтвердження встановленого користувача сесії
	public message($msg, $msgtype)	Отримання повідомлення в сесії користувача
	public iskeyactive($key)	Отримання інформації про активний ключ
	public check_active()	Перевірка стану активності сесії користувача
Account		Взаємодія з БД у контексті управління акаунтами користувачів
	public create()	Створення нових акаунтів користувачів
	public update($id)	Оновлення даних про акаунти в базі даних
	public delete($id)	Видалення існуючих акаунтів з бази даних

Продовження таблиці 3.9

Класи	Назва функції	Короткий опис
	listofusers()	Отримання списку користувачів
	find_user($id, $user_name)	Пошук користувачів зі списку
	userAuthentication($h_pass)	Автентифікація користувача
Faculty		Взаємодія з БД у контексті управління факультетами
	public create()	Створення нових факультетів в базі даних
	public update($id)	Оновлення даних про факультети в базі даних
	public delete($id)	Видалення існуючого факультету з бази даних
	listoffaculties()	Отримання списку факультетів
	find_faculty($id, $name)	Пошук факультету зі списку
Department		Взаємодія з БД у контексті управління кафедрами
	public create()	Створення нових кафедр в базі даних
	public update($id)	Оновлення даних про кафедри в базі даних
	public delete($id)	Видалення існуючих кафедр з бази даних
	listofdepartments()	Отримання списку кафедр
	find_department($id, $name)	Пошук кафедр зі списку
Teacher		Взаємодія з БД у контексті роботи зі науково-педагогічними працівниками
	public create()	Створення нових працівників
	public update($id)	Оновлення даних про працівників
	public delete($id)	Видалення існуючих працівників з бази даних

Продовження таблиці 3.9

Класи	Назва функції	Короткий опис
	listofteachers ()	Отримання списку працівників
	find_teacher($id, $name)	Пошук працівників в базі даних
	find_pass($id)	Пошук пароля працівника в базі даних
Diploma		Взаємодія з БД у контексті роботи з дипломами працівників
	public create()	Створення нових дипломів працівників
	public update($id)	Оновлення даних про дипломи працівників
	public delete($id)	Видалення існуючих дипломів з бази даних
	listofdiplomas ()	Отримання списку дипломів працівників
	find_diploma($id, $name)	Пошук дипломів в базі даних
Certificate		Взаємодія з БД у контексті роботи з сертифікатами працівників
	public create()	Створення нових сертифікатів працівників
	public update($id)	Оновлення даних про сертифікати працівників
	public delete($id)	Видалення існуючих сертифікатів з бази даних
	listofcertificates ()	Отримання списку сертифікатів працівників
	find_certificate($id, $name)	Пошук сертифікатів в базі даних
Course		Взаємодія з БД у контексті роботи з курсами від організацій
	public create()	Створення нових курсів від організацій

Продовження таблиці 3.9

Класи	Назва функції	Короткий опис
	public update($id)	Оновлення даних про курси від організацій
	public delete($id)	Видалення існуючих курсів з бази даних
	listofcourses ()	Отримання списку курсів від організацій
	find_course($id, $name)	Пошук курсів в базі даних
Organization		Взаємодія з БД у контексті роботи з організаціями
	public create()	Створення нових організацій
	public update($id)	Оновлення даних організацій
	public delete($id)	Видалення існуючих організацій з бази даних
	listofcourses ()	Отримання списку організацій
	find_course($id, $name)	Пошук організацій в базі даних

Програмний код представлений в Додатку А.

3.3.2 Огляд функціональних можливостей програмного продукту

Розроблена система обліку педагогічних та науково-педагогічних працівників має реалізувати виконання певних завдань, що стосуються збору і систематизації особових даних працівників освітнього закладу шляхом внесення інформації в особистому кабінеті користувача; формування даних щодо проходження підвищення кваліфікації, стажування, відповідних курсів, сертифікації тощо; участі в конференціях та наукових дослідженнях (публікація тез, наукових статей, наукове керівництво та інше).

Система обліку СОД (тестова назва системи) надає можливість реєстрації педагогічних та науково-педагогічних працівників через електронні кабінети. Для доступу до осноного функціоналу системи СОД користувачеві пропонується

спочатку увійти в систему використавши логін і пароль. Вигляд вікна авторизації в системі наведено на рисунку 3.7.

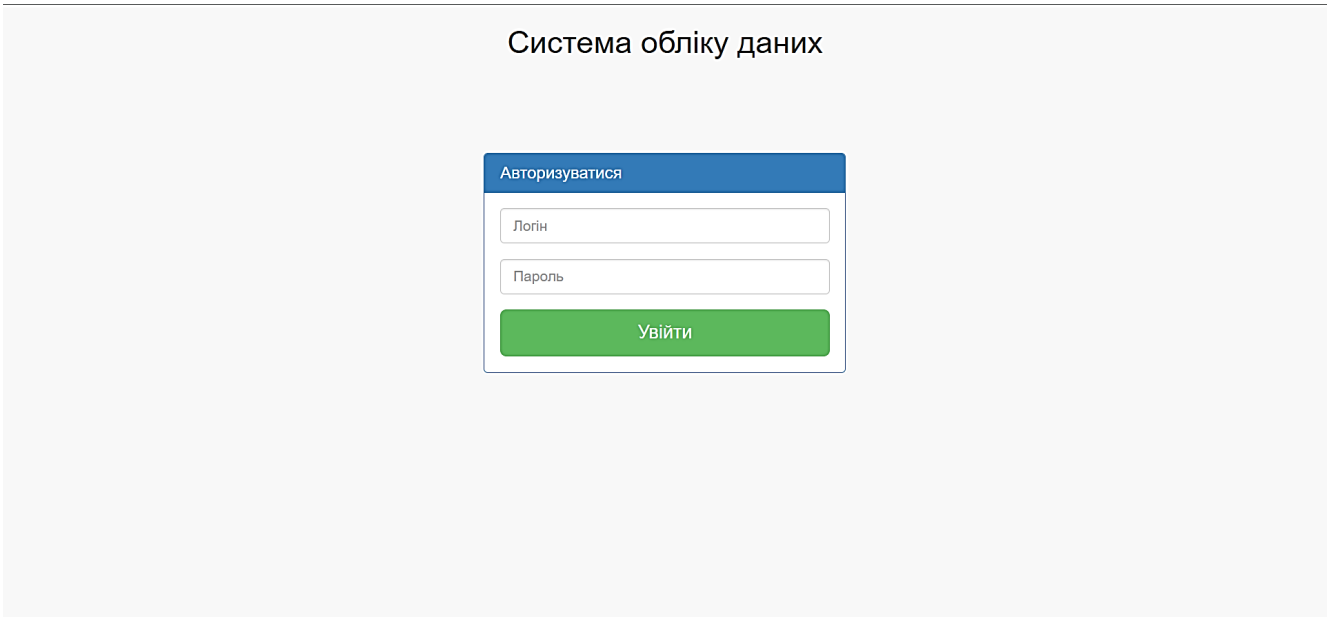

Рисунок 3.7 – Вікно авторизації

Після входу в систему обліку СОД користувача зустрічає домашня сторінка системи (рис. Х.Х), яка має певну структуру. Серед основних розділів:

– «Головна сторінка»;
– «Структура університету»;
– «Дипломи та сертифікати»;
– «Установи та організації»;
– «Вихід з системи».

Також є певні важливі підрозділи, які містять конкретизуючу інформацію:

– «Факультети»;
– «Кафедри»;
– «Науково-педагогічний склад»;
– «Дипломи»;
– «Сертифікати»;
– «Курси».

Окремі підрозділи можна додавати з метою покращення функціоналу даної системи. Головна сторінка з прикладом відповідної структури системи СОД наведена на рисунку 3.8.

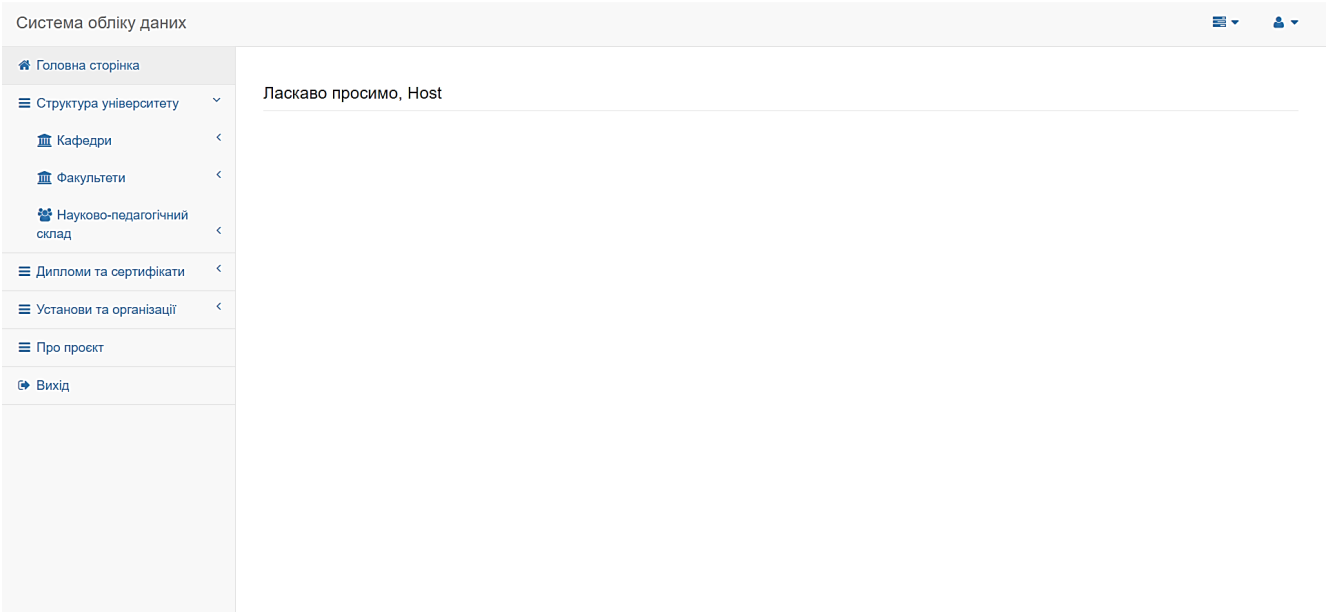

Рисунок 3.8 – Головна сторінка системи

Перейшовши у відповідний підрозділ (наприклад, «Науково-педагогічний склад»), користувачеві пропонується додати нову інформацію (особові дані; дані щодо підвищення кваліфікації, проходження стажування; отримання свідоцтв, сертифікатів тощо) або переглянути таблицю наявних даних. Відповідний функціонал відображений на рисунках 3.9 та 3.10.

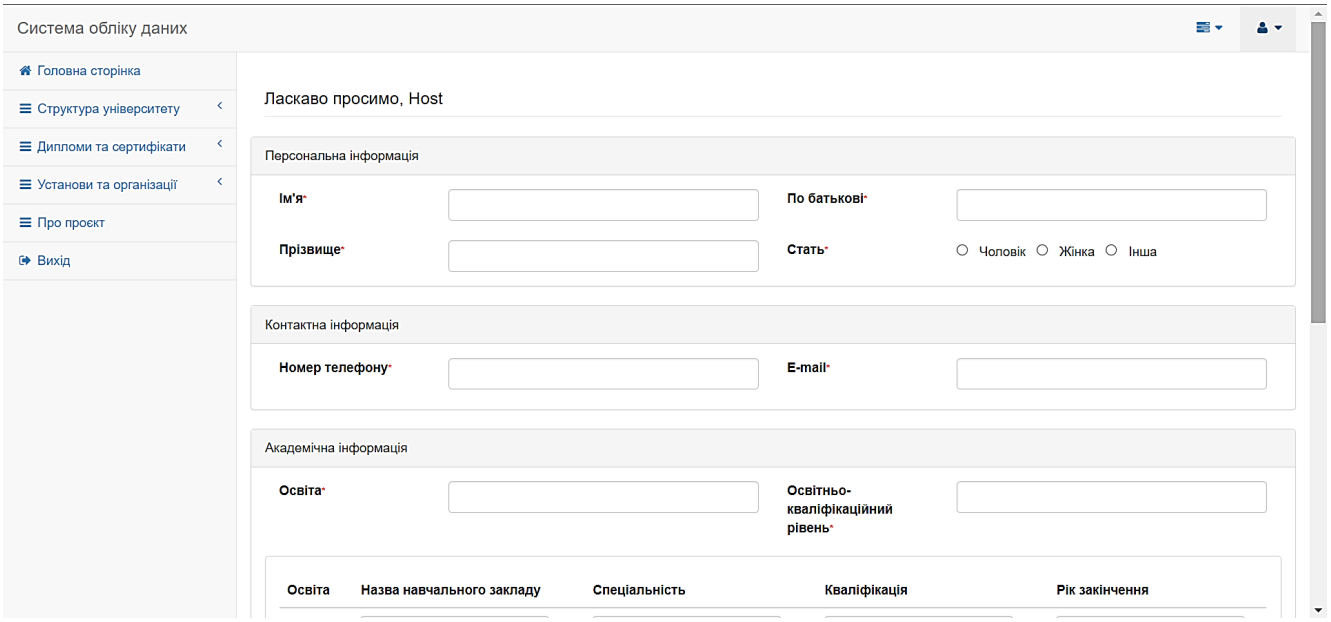

Рисунок 3.9 – Вікно внесення даних про викладача (початок сторінки)

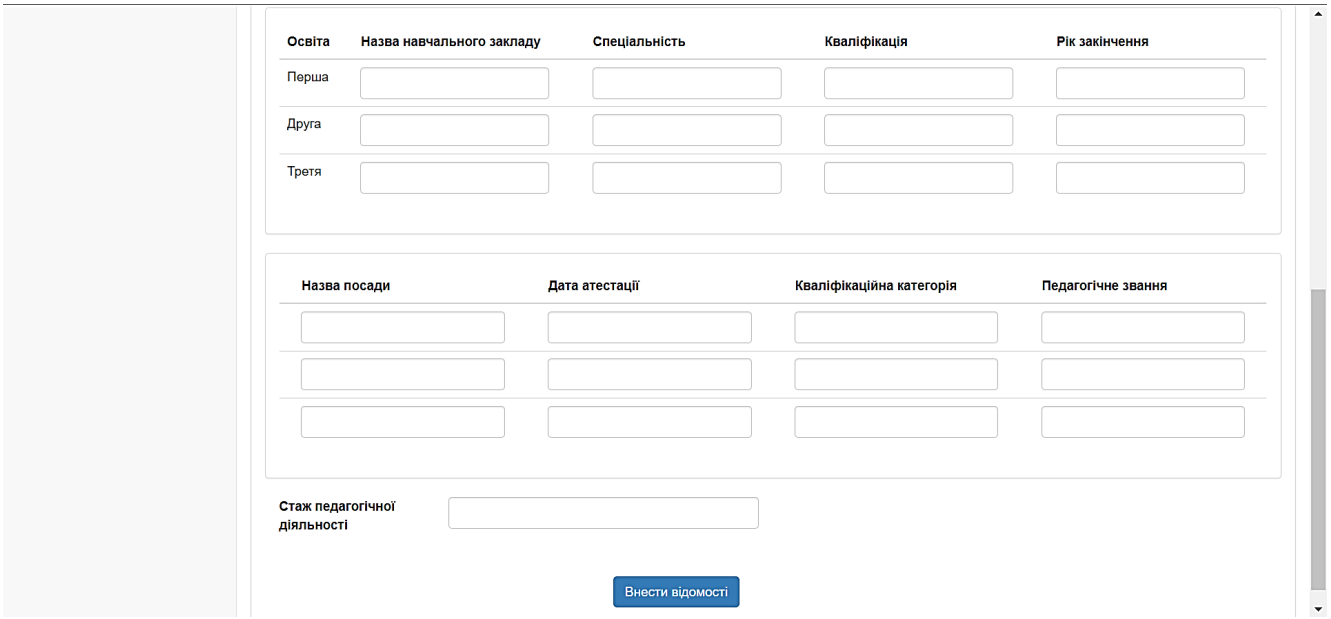

Рисунок 3.10 – Вікно внесення даних про викладача (продовження сторінки)

Також є можлівість перегляду і редагування інформації у відповідних підрозділах системи (наприклад, «Кафедри»), при необхідності можна змінити або вилучити дані (рис. 3.11 – 3.12).

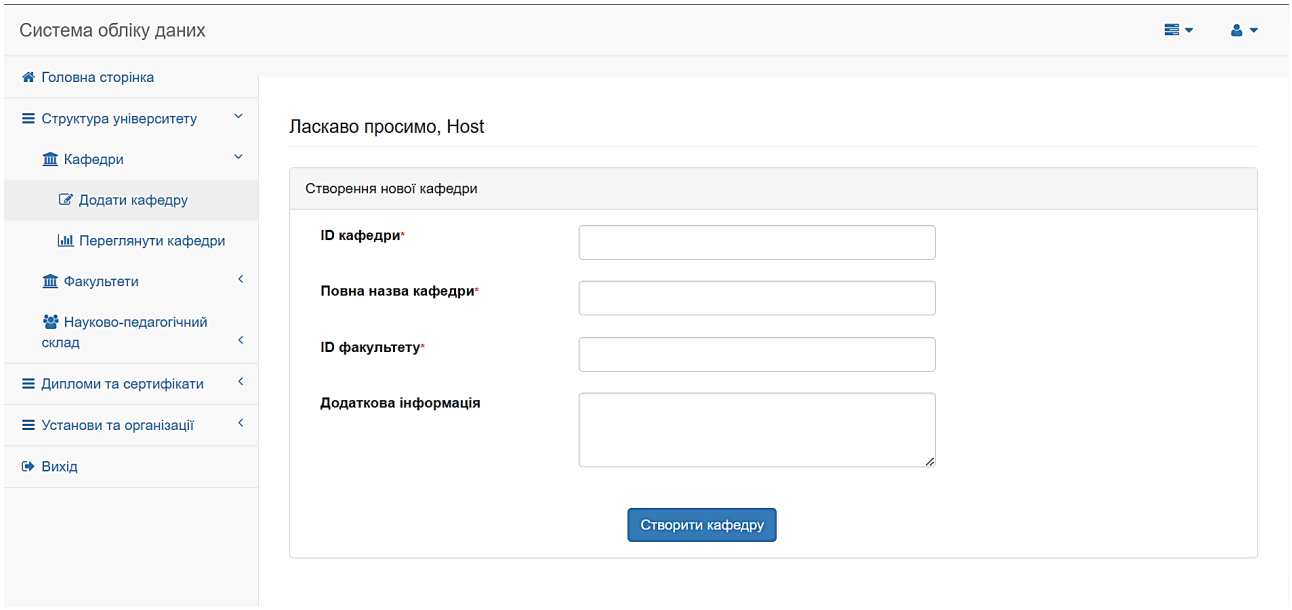

Рисунок 3.11 – Вікно створення нової кафедри

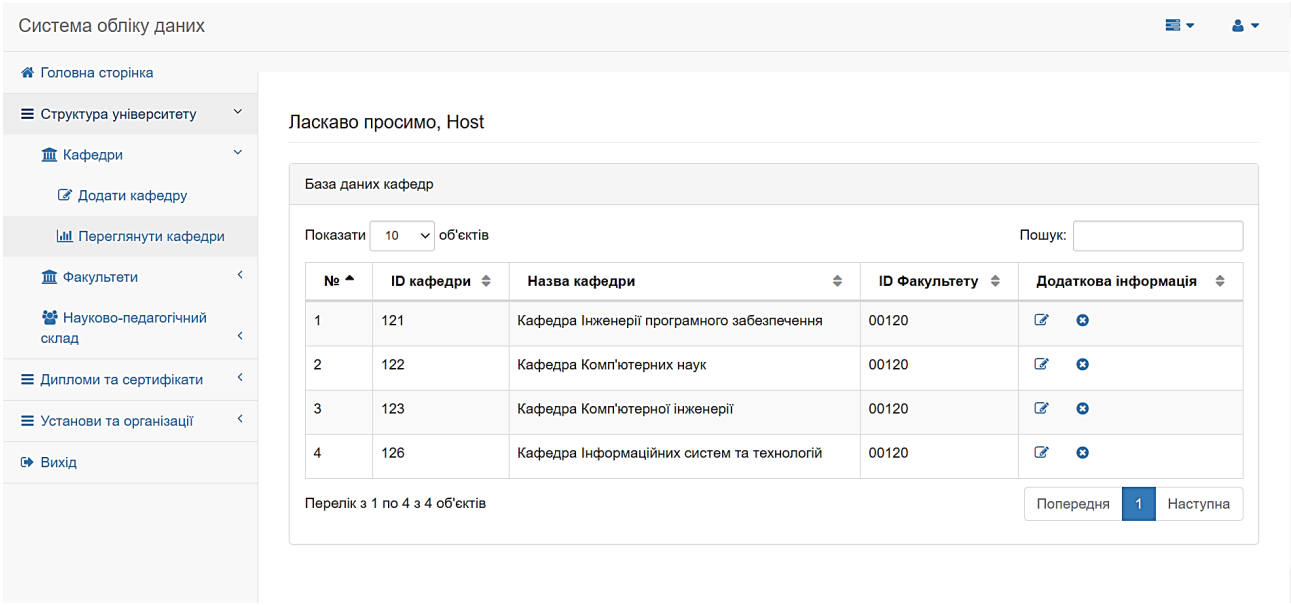

Рисунок 3.12 – Вікно перегляду таблиці кафедр

Дана система обліку педагогічних та науково-педагогічних працівників надає можливість систематизувати облікові дані працівників освітнього закладу, спростити процеси формування звітньої документації та прийняття рішень щодо атестації науково-педагогічних працівників атестаційними комісіями на підставі їхніх результатів проходження підвищення кваліфікації та науково-дослідницької роботи.

ВИСНОВКИ

Розробка і впровадження нових якісних систем ECM та електронних систем обліку даних в освітній галузі для цифровізації наявної інформації – є логічним кроком, що має вирішити багато питань, які стосуються ефективної взаємодії, систематизації та уніфікації даних, дотримання європейських стандартів щодо забезпечення якості освіти.

Під час виконання даної кваліфікаційної роботи були здійснені прикладні дослідження, що стосувались наявних освітніх електронних систем обліку (ЄДЕБО, Єдина атестаційна система (ЄАС), Єдиний портал державних послуг «Дія», автоматизований інформаційний комплекс АІКОМ). Дослідження доворять, що наявні облікові системи освітніх даних не враховують у повній мірі потреби навчальних закладів щодо узагальнення особових даних педагогічних та науково-педагогічних працівників, які стосуються атестації, підвищення кваліфікації, сертифікації та автоматизації процесів управління і звітності. Зокрема, не розроблена уніфікована система обліку науково-педагогічних працівників, немає стандартизації облікових даних, не допрацьовані інструменти організації багаторівневої комунікації.

Також проведені ґрунтовні дослідження систем управління базами даних: порівняльний аналіз реляційних та нереляційних моделей баз даних, технологічних вимог щодо їх застосування та можливих варіантів програмних стеків. Серед розглянутих моделей СУБД була обрана саме реляційна модель баз даних (MySQL) як один з найкращих інструментів для розробки систем обліку освітніх даних, враховуючи їх структурованість.

Також важливими технологіями, які дозволили розробити програмний засіб (система обліку педагогічних та науково-педагогічних працівників) є редактор вихідного коду Microsoft Visual Studio Code (VS Code), скриптова мова програмування PHP, реляційна система управління базами даних MySQL та об'єктно-орієнтована прототипна мова програмування JavaScript.

Результатом кваліфікаційної роботи стала така система обліку педагогічних та науково-педагогічних працівників, яка дозволяє частково вирішити визначенні проблемні завдання, а саме сформувати електронну базу даних працівників освітнього закладу для систематизації загальних особових даних, інформації щодо підвищення кваліфікації, сертифікації та атестації.

Система планується покращуватися шляхом додавання нового функціоналу. Можливі варіанти використання такої системи обліку – це автоматизоване формування звітньої документації, узагальнення інформації про дотримання кваліфікаційних вимог, атестаційних процесів, систематизація даних щодо кадрового забезпечення.

Подальші дослідження в цій площині допоможуть створити уніфіковану модель системи обліку педагогічних і науково-педагогічних працівників, яка здатна забезпечити дотримання європейських стандартів щодо сертифікації та якості освіти.

ПЕРЕЛІК ПОСИЛАНЬ

1. Деякі питання підвищення кваліфікації педагогічних і науково-педагогічних працівників : Постанова Кабінету Міністрів України від 21.08.2019 року № 800. URL: https://zakon.rada.gov.ua/laws/show/800-2019-%D0%BF#Text (дата звернення: 03.12.2024).

2. Освітні документи вже в застосунку Дія. Сайт Дія. URL: https://diia.gov.ua/news/osvitni-dokumenti-vzhe-v-zastosunku-diya (дата звернення: 02.12.2024).

3. Основні запитання-відповіді щодо єДокумента про освіту. Сайт ЄДЕБО. URL: https://info.edbo.gov.ua/news/09-04-2024-osnovni-zapytannia-vidpovidi-shchodo-iedokumenta-pro-osvitu/ (дата звернення: 02.12.2024).

4. Офіційний сайт Міністерства освіти і науки України: Реєстри та сервіси ЄДЕБО. URL: https://mon.gov.ua/reestri-ta-servisi-edebo (дата звернення: 02.12.2024).

5. Про вищу освіту : Закон України від 01.07.2014 року № 1556-VII (Редакція від 16.08.2024). URL: https://zakon.rada.gov.ua/laws/show/1556-18#Text (дата звернення: 03.12.2024).

6. Про затвердження Методичних рекомендацій щодо підвищення рівня кіберзахисту систем електронного документообігу : Наказ Адміністрація Державної служби спеціального зв'язку та захисту інформації України від 30.08.2023 року № 773. URL: https://zakon.rada.gov.ua/rada/show/v0773519-23#Text (дата звернення: 14.12.2024).

7. Про затвердження Положення про атестацію педагогічних працівників : Наказ Міністерства освіти і науки України від 09.09.2022 року № 805 (Редакція від 05.11.2024). URL: https://zakon.rada.gov.ua/laws/show/z1649-22#Text (дата звернення: 03.12.2024).

8. Про Національну програму інформатизації : Закон України від 01.12.2022 року № 2807-IX. URL: https://zakon.rada.gov.ua/laws/show/2807-20#Text (дата звернення: 14.12.2024).

9. Про освіту : Закон України від 05.09.2017 року № 2145-VIII (Редакція від 15.11.2024). URL: https://zakon.rada.gov.ua/laws/show/2145-19#Text (дата звернення: 03.12.2024).

10. Проект Плану відновлення України. Матеріали робочої групи «Освіта і наука» : Національна рада з відновлення України від наслідків війни. 2022. URL: https://mon.gov.ua/static-objects/mon/sites/1/gromadske-obgovorennya/2022/08/19/HO.proyekt.Planu.vidnovl.Osv.i.nauky-19.08.2022.pdf (дата звернення: 01.12.2024).

11. Про фахову передвищу освіту : Закон України від 06.06.2019 року № 2745-VIII (Редакція від 17.11.2024). URL: https://zakon.rada.gov.ua/laws/show/2745-19#Text (дата звернення: 03.12.2024).

12. Сайт AIKOM. URL: https://aikom.iea.gov.ua/site/about# (дата звернення: 04.12.2024).

13. Сайт Державного підприємства «Інфоресурс»: Інструкції по роботі з ЄДЕБО. URL: https://www.inforesurs.gov.ua/edebo/instructions/ (дата звернення: 02.12.2024).

14. Сайт Єдиної атестаційної системи. URL: https://eas.ua/ (дата звернення: 02.12.2024).

15. Сайт ЄДЕБО. URL: https://info.edbo.gov.ua/ (дата звернення: 02.12.2024).

16. Бондаренко С. СУБД: які бувають, як вибрати. *Highload.* https://highload.today/uk/subd-yaki-buvayut-yak-vibrati/ (дата звернення: 03.12.2024).

17. Ковтун Б. В., Манич А. М., Романюк О. В. Порівняльна характеристика реляційних та NoSQL баз даних. *Матеріали XLIX науково-технічної конференції підрозділів ВНТУ, Вінниця, 27-28 квітня 2020 р.* 2020. URL: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/29461/9907.pdf?sequence=3&isAllowed=y (дата звернення: 30.10.2024).

18. Мазурова О. О., Набока А. О., Широкопетлєва М. С. Дослідження методів реалізації розподілених ACID транзакцій за технологією реплікації. *Сучасний стан наукових досліджень та технологій в промисловості.* 2021. № 2 (16). С. 19–31. DOI: https://doi.org/10.30837/ITSSI.2021.16.019

19. Маїлян А. Огляд основних SQL запитів. URL: https://itvdn.com/ua/blog/article/m-sql (дата звернення: 03.12.2024).

20. Петров М. VS Code (Visual Studio Code): налаштування, розширення, комбінації клавіш. URL: https://petrov.net.ua/visual-studio-code-settings-extensions-hotkeys/ (дата звернення: 14.12.2024).

21. Пєтух А., Романюк О., Романюк О. Бази даних. Мови запитів, управління транзакціями, розподілена обробка даних : Електрон. навч. посіб. ВНТУ, 2016. URL: https://web.posibnyky.vntu.edu.ua/fitki/11petuh_bazdanyh_movy_zalitiv/ (дата звернення: 30.10.2024).

22. Резников І. Хмарні бази даних. Детальна інструкція, як застосовувати сучасний IT-підхід. *DOU*. URL: https://dou.ua/forums/topic/43820/ (дата звернення: 18.11.2024).

23. Спірін О. М. Використання електронних систем відкритого доступу для інформаційно-аналітичної підтримки педагогічних досліджень / О. М. Спірін, А. В. Яцишин, С. М. Іванова, А. В. Кільченко, Л. А. Лупаренко // Інформаційні технології і засоби навчання. - 2016. - Т. 55, вип. 5. - С. 136-174. DOI:10.33407/itlt.v55i5.1501 URL: https://www.researchgate.net/publication/331470545_VIKORISTANNA_ELEKTRONNIH_SISTEM_VIDKRITOGO_DOSTUPU_DLA_INFORMACIJNO-ANALITICNOI_PIDTRIMKI_PEDAGOGICNIH_DOSLIDZEN (дата звернення: 01.12.2024).

24. Ткачук І. Типи баз даних: особливості, відмінності та приклади. *DOU.* URL: https://dou.ua/lenta/articles/types-of-databases/ (дата звернення: 14.12.2024).

25. Швець С. М. Питання інформаційних ресурсів: зв'язок національних зі світовими. Правова інформатика: №3(23). 2009. С. 42-51. URL: https://ippi.org.ua/sites/default/files/09ssmzns.pdf (дата звернення: 01.12.2024).

26. Сайт Visual Studio Code. URL: https://code.visualstudio.com/ (дата звернення: 14.12.2024).

27. Система ECM (Enterprise Content Management). *EVolpe.* URL: https://evolpe.com.ua/%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0-ecm-enterprise-content-management/ (дата звернення: 14.12.2024).

28. Сучасний підручник з JavaScript. *JavaScript.Info* URL: https://uk.javascript.info/ (дата звернення: 14.12.2024).

29. Що таке СУБД і для чого вони потрібні. *FoxmindEd.* URL: https://foxminded.ua/systema-upravlinnia-bazamy-danykh/ (дата звернення: 18.11.2024).

30. Cheyenne Kolosky. A Guide to Building a Free Database for Education in 11 Simple Steps. *Knack.* URL: https://www.knack.com/blog/database-for-education/ (дата звернення: 14.12.2024).

31. Cuello C. Understanding the different types of databases & When to use them. *Rivery*. URL: https://rivery.io/data-learning-center/database-types-guide/ (date of access: 30.10.2024).

32. DB-Engines Ranking. *DB-Engines*. URL: https://db-engines.com/en/ranking (date of access: 20.11.2024).

33. Database Scalability Definition. *ScyllaDB.* URL: https://www.scylladb.com/glossary/database-scalability/ (date of access: 18.11.2024).

34. Database Technologies. *Oracle Україна*. URL: https://www.oracle.com/ua/database/technologies/ (дата звернення: 14.12.2024).

35. ENIC-NARIC. URL: https://www.enic-naric.net/ (date of access: 02.12.2024).

36. Grishina A. The 5 Best Databases for Web Apps in 2023: How to Make the Right Choice. *Softteco.* URL: https://softteco.com/blog/best-databases-for-web-apps-2023 (date of access: 18.11.2024).

37. Haerder, T., Reuter, A. Principles of transaction-oriented database recovery. *ACM Computing Surveys.* 1983. No. 15 (4). P. 287–317. URL: https://www.researchgate.net/publication/200030219_Principles_of_Transaction-Oriented_Database_Recovery (date of access: 18.11.2024).

38. Jemin Desai. Top 12 Best Databases for Web Application Development in 2024. *Positiwise.* URL: https://positiwise.com/blog/best-databases-for-web-application-development (дата звернення: 14.12.2024).

39. Kamal Acharya. (2024) Teachers Record Management System Project. SSRN Electronic Journal. URL: https://www.researchgate.net/publication/380740859_Teachers_Record_Management_System_Project_Report (date of access: 18.11.2024).

40. Michael B. Allen. Guide to Teacher Data. *The Association of Public and Land-grant Universities (APLU).* URL: https://www.aplu.org/our-work/5-archived-projects/stem-education/science-and-mathematics-teaching-imperative/smti-projects/improving-state-need-assessment/guide-to-teacher-data/ (дата звернення: 14.12.2024).

41. MySQL. *Hosting Ukraine.* URL: https://www.ukraine.com.ua/uk/wiki/hosting/databases/mysql/ (date of access: 13.12.2024).

42. Panasenko Y., Omelchenko D., Shcherban N. A guide to choosing a reliable and scalable database solution for your product. *Yalantis*. URL: https://yalantis.com/blog/how-to-choose-a-database/ (date of access: 20.11.2024).

43. Peters, M. A., Britez, R. G. (Eds.). (2008). Open Education and Education for Openness. Rotterdam, the Netherlands: Sense Publishers. URL: https://www.researchgate.net/publication/255525908_Open_Education_and_Education_for_Openness (date of access: 18.11.2024).

44. PHP vs Python: A Detailed Comparison Between the Two Languages. *Kinsta*. URL: https://kinsta.com/blog/php-vs-python/ (date of access: 14.12.2024).

45. SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems. *DigitalOcean.* URL:

https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems (дата звернення: 14.12.2024).

46. XAMPP for Windows. *PCWorld* URL: https://www.pcworld.com/article/486634/xampp_for_windows.html (дата звернення: 14.12.2024).

ДОДАТОК А

ПРОГРАМНИЙ КОД

Database.php

```
<?php
require_once(LIB_PATH.DS."config.php");
class Database {
    var $sql_string = '';
    var $error_no = 0;
    var $error_msg = '';
    private $conn;
    public $last_query;
    private $magic_quotes_active;
    private $real_escape_string_exists;

    function __construct() {
        $this->open_connection();
        $this->magic_quotes_active = get_magic_quotes_gpc();
        $this->real_escape_string_exists =
function_exists("mysqli_real_escape_string");
    }

    public function open_connection() {
        $this->conn = mysqli_connect(server,user,pass);
        if(!$this->conn){
            echo "Problem in database connection! Contact administrator!";
            exit();

        }else{

            $db_select = mysqli_select_db($this->conn,database_name);
            if (!$db_select) {
                echo "Problem in selecting database! Contact administrator!";
```

```
            exit();
        }
    }

}

function setQuery($sql='') {
    $this->sql_string=$sql;
}

function executeQuery() {
    $result = mysqli_query($this->conn,$this->sql_string);
    $this->confirm_query($result);
    return $result;
}

private function confirm_query($result) {
    if(!$result){
        $this->error_no = mysqli_errno($this->conn);
        $this->error_msg = mysqli_error($this->conn);
        return false;
    }
    return $result;
}

function loadResultList( $key='' ) {
    $cur = $this->executeQuery();

    $array = array();
    while ($row = mysqli_fetch_object($cur)) {
        if ($key) {
            $array[$row->$key] = $row;
        } else {
            $array[] = $row;
        }
    }
    mysqli_free_result( $cur );
```

```
    return $array;
}

function loadSingleResult() {
    $cur = $this->executeQuery();

    while ($row = mysqli_fetch_object($cur)) {
    return $data = $row;
    }
    mysqli_free_result($cur);
}

function getFieldsOnOneTable($tbl_name) {

    $this->setQuery("DESC ".$tbl_name);
    $rows = $this->loadResultList();

    $f = array();
    for ( $x=0; $x<count($rows); $x++ ) {
        $f[] = $rows[$x]->Field;
    }

    return $f;
}

public function fetch_array($result) {
    return mysqli_fetch_array($result);
}
public function num_rows($result_set) {
    return mysqli_num_rows($result_set);
}

public function insert_id() {
    return mysqli_insert_id($this->conn);
}

public function affected_rows() {
```

```
        return mysqli_affected_rows($this->conn);
    }

     public function escape_value( $value ) {
        if( $this->real_escape_string_exists ) {
            if($this->magic_quotes_active) { $value = stripslashes($value); }
            $value = mysqli_real_escape_string($this->conn,$value);
        } else {
            if( !$this->magic_quotes_active ) { $value = addslashes($value);
}
        }
        return $value;
    }

    public function close_connection() {
        if(isset($this->conn)) {
            mysqli_close($this->conn);
            unset($this->conn);
        }
    }

}
$mydb = new Database();

?>
```

Session.php

```
<?php
session_start();

    function logged_in() {
        return isset($_SESSION['USERID']);

    }
    function confirm_logged_in() {
        if (!logged_in()) {?>
```

```
            <script type="text/javascript">
                window.location = "login.php";
            </script>

        <?php
        }
    }
function admin_confirm_logged_in() {
        if (@!$_SESSION['USERID']) {?>
            <script type="text/javascript">
                window.location ="login.php";
            </script>

        <?php
        }
    }

    function studlogged_in() {
        return isset($_SESSION['IDNO']);

    }
    function studconfirm_logged_in() {
        if (!studlogged_in()) {?>
            <script type="text/javascript">
                window.location = "index.php";
            </script>

        <?php
        }
    }

    function message($msg="", $msgtype="") {
      if(!empty($msg)) {
        $_SESSION['message'] = $msg;
        $_SESSION['msgtype'] = $msgtype;
      } else {
```

```
            return $message;
      }
    }
    function check_message(){

        if(isset($_SESSION['message'])){
            if(isset($_SESSION['msgtype'])){
                if ($_SESSION['msgtype']=="info"){
                    echo  '<div class="alert alert-info">'.
$_SESSION['message'] . '</div>';

                }elseif($_SESSION['msgtype']=="error"){
                    echo  '<div class="alert alert-danger">' .
$_SESSION['message'] . '</div>';

                }elseif($_SESSION['msgtype']=="success"){
                    echo  '<div class="alert alert-success">' .
$_SESSION['message'] . '</div>';
                }
                unset($_SESSION['message']);
                unset($_SESSION['msgtype']);
            }

        }

    }
 function keyactive($key=""){
     if(!empty($key)) {
        $_SESSION['active'] = $key;
      } else {
            return $keyactive;
      }

 }

 function check_active(){
     if(isset($_SESSION['active'])){
```

```
         switch ($_SESSION['active']) {

        case 'basicInfo' :
        $_SESSION['basicInfo']   = "active";
        break;
        case 'otherInfo' :
        $_SESSION['otherInfo']= 'active';
        break;

        case 'work' :
        $_SESSION['work'] = 'active' ;
        break;
      }
      }else{

          $active = (isset($_GET['active']) && $_GET['active'] != '') ?
$_GET['active'] : '';
                switch ($active) {

                 case 'otherInfo' :
                  $_SESSION['otherInfo']= 'active';
                    break;

                 case 'work' :
                  $_SESSION['work'] = 'active' ;
                     break;

                 default :

                   $_SESSION['basicInfo']   = "active";
                break;

      }
 }
}

?>
```

Account.php

```
<?php
require_once(LIB_PATH.DS.'database.php');
class User {
    protected static  $tblname = "tblusers";

    function dbfields () {
        global $mydb;
        return $mydb->getfieldsononetable(self::$tblname);
    }
    function listofuser(){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname);
        return $cur;
    }
    function find_user($id="",$user_name=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            WHERE USERID = {$id} OR UEMAIL = '{$user_name}'");
        $cur = $mydb->executeQuery();
        $row_count = $mydb->num_rows($cur);
        return $row_count;
    }
    static function userAuthentication($email,$h_pass){
        global $mydb;
        $mydb->setQuery("SELECT * FROM `tblusers` WHERE `UEMAIL` = '". $email
."' and `PASS` = '". $h_pass ."'");
        $cur = $mydb->executeQuery();
        if($cur==false){
            die(mysql_error());
        }
        $row_count = $mydb->num_rows($cur);
         if ($row_count == 1){
         $user_found = $mydb->loadSingleResult();
            $_SESSION['USERID']         = $user_found->USERID;
            $_SESSION['NAME']           = $user_found->NAME;
```

```
        $_SESSION['UEMAIL']         = $user_found->UEMAIL;
        $_SESSION['PASS']           = $user_found->PASS;
        $_SESSION['TYPE']           = $user_found->TYPE;
       return true;
     }else{
        return false;
     }
}
function single_user($id=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            Where USERID= '{$id}' LIMIT 1");
        $cur = $mydb->loadSingleResult();
        return $cur;
}
static function instantiate($record) {
    $object = new self;

    foreach($record as $attribute=>$value){
      if($object->has_attribute($attribute)) {
        $object->$attribute = $value;
      }
    }
    return $object;
}

private function has_attribute($attribute) {
  return array_key_exists($attribute, $this->attributes());
}

protected function attributes() {
  global $mydb;
  $attributes = array();
  foreach($this->dbfields() as $field) {
    if(property_exists($this, $field)) {
        $attributes[$field] = $this->$field;
    }
```

```
  }
  return $attributes;
}

protected function sanitized_attributes() {
  global $mydb;
  $clean_attributes = array();
  foreach($this->attributes() as $key => $value){
    $clean_attributes[$key] = $mydb->escape_value($value);
  }
  return $clean_attributes;
}



public function save() {
  return isset($this->id) ? $this->update() : $this->create();
}

public function create() {
    global $mydb;
    $attributes = $this->sanitized_attributes();
    $sql = "INSERT INTO ".self::$tblname." (";
    $sql .= join(", ", array_keys($attributes));
    $sql .= ") VALUES ('";
    $sql .= join("', '", array_values($attributes));
    $sql .= "')";
echo $mydb->setQuery($sql);

 if($mydb->executeQuery()) {
    $this->id = $mydb->insert_id();
    return true;
  } else {
    return false;
  }
}

public function update($id=0) {
```

```
      global $mydb;
        $attributes = $this->sanitized_attributes();
        $attribute_pairs = array();
        foreach($attributes as $key => $value) {
          $attribute_pairs[] = "{$key}='{$value}'";
        }
        $sql = "UPDATE ".self::$tblname." SET ";
        $sql .= join(", ", $attribute_pairs);
        $sql .= " WHERE USERID=". $id;
      $mydb->setQuery($sql);
        if(!$mydb->executeQuery()) return false;

    }

    public function delete($id=0) {
        global $mydb;
          $sql = "DELETE FROM ".self::$tblname;
          $sql .= " WHERE USERID=". $id;
          $sql .= " LIMIT 1 ";
          $mydb->setQuery($sql);

            if(!$mydb->executeQuery()) return false;

    }

}
?>
```

Teacher.php

```
<?php
require_once(LIB_PATH.DS.'database.php');
class Teacher {
    protected static  $tblname = "tblteacher";

    function dbfields () {
        global $mydb;
```

```
        return $mydb->getfieldsononetable(self::$tblname);

    }
    function listofteachers(){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname);
        return $cur;
    }
    function find_teacher($id="",$name=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            WHERE TeacherID = {$id} OR LNAME = '{$name}'");
        $cur = $mydb->executeQuery();
        $row_count = $mydb->num_rows($cur);
        return $row_count;
    }

    function find_all_teachers($name=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            WHERE LNAME = '{$name}'");
        $cur = $mydb->executeQuery();
        $row_count = $mydb->num_rows($cur);
        return $row_count;
    }
        static function teacherAuthentication($email,$h_pass){
        global $mydb;
        $mydb->setQuery("SELECT * FROM `tblteacher` WHERE `TEACHERUSERNAME` =
'". $email ."' and `TEACHERPASS` = '". $h_pass ."'");
        $cur = $mydb->executeQuery();
        if($cur==false){
            die(mysqli_error());
        }
        $row_count = $mydb->num_rows($cur);
         if ($row_count == 1){
         $user_found = $mydb->loadSingleResult();
            $_SESSION['TeacherID']      = $user_found->TeacherID;
```

```
        $_SESSION['Fname']          = $user_found->Fname;
        $_SESSION['Lname']          = $user_found->Lname;
        $_SESSION['USERNAME']       = $user_found->TEACHERUSERNAME;
        $_SESSION['TEACHERPASS']       = $user_found->TEACHERPASS;
       return true;
     }else{
        return false;
     }
}
function find_pass($id="",$pass=""){
    global $mydb;
    $mydb->setQuery("SELECT * FROM ".self::$tblname."
        WHERE TeacherID = {$id} and TEACHERPASS = '{$pass}'");
    $cur = $mydb->executeQuery();
    $row_count = $mydb->num_rows($cur);
    return $row_count;
}


function single_teacher($id=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            Where TeacherID= '{$id}' LIMIT 1");
        $cur = $mydb->loadSingleResult();
        return $cur;
}

static function instantiate($record) {
    $object = new self;

    foreach($record as $attribute=>$value){
      if($object->has_attribute($attribute)) {
        $object->$attribute = $value;
      }
    }
    return $object;
}
```

```
private function has_attribute($attribute) {
  return array_key_exists($attribute, $this->attributes());
}

protected function attributes() {
  global $mydb;
  $attributes = array();
  foreach($this->dbfields() as $field) {
    if(property_exists($this, $field)) {
        $attributes[$field] = $this->$field;
    }
  }
  return $attributes;
}

protected function sanitized_attributes() {
  global $mydb;
  $clean_attributes = array();
  foreach($this->attributes() as $key => $value){
    $clean_attributes[$key] = $mydb->escape_value($value);
  }
  return $clean_attributes;
}

public function save() {
  return isset($this->id) ? $this->update() : $this->create();
}

public function create() {
    global $mydb;
    $attributes = $this->sanitized_attributes();
    $sql = "INSERT INTO ".self::$tblname." (";
    $sql .= join(", ", array_keys($attributes));
    $sql .= ") VALUES ('";
    $sql .= join("', '", array_values($attributes));
    $sql .= "')";
```

```
    echo $mydb->setQuery($sql);

     if($mydb->executeQuery()) {
        $this->id = $mydb->insert_id();
        return true;
      } else {
        return false;
      }
    }

    public function update($id=0) {
      global $mydb;
        $attributes = $this->sanitized_attributes();
        $attribute_pairs = array();
        foreach($attributes as $key => $value) {
          $attribute_pairs[] = "{$key}='{$value}'";
        }
        $sql = "UPDATE ".self::$tblname." SET ";
        $sql .= join(", ", $attribute_pairs);
        $sql .= " WHERE TeacherID=". $id;
      $mydb->setQuery($sql);
        if(!$mydb->executeQuery()) return false;

    }

    public function delete($id=0) {
        global $mydb;
          $sql = "DELETE FROM ".self::$tblname;
          $sql .= " WHERE TeacherID=". $id;
          $sql .= " LIMIT 1 ";
          $mydb->setQuery($sql);

            if(!$mydb->executeQuery()) return false;

    }

}
```

```
?>
```

Department.php

```
<?php
require_once(LIB_PATH.DS.'database.php');

 class Department {
    protected static  $tblname = "tbldepartment";

    function dbfields () {
        global $mydb;
        return $mydb->getfieldsononetable(self::$tblname);

    }
    function listofdepartments(){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname);
        return $cur;
    }
    function find_course($id="",$faculty=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            WHERE DepartmentID = {$id} OR FacultyID = '{$faculty}'");
        $cur = $mydb->executeQuery();
        $row_count = $mydb->num_rows($cur);
        return $row_count;
    }

    function find_all_course($faculty=""){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            WHERE FacultyID = '{$faculty}'");
        $cur = $mydb->executeQuery();
        $row_count = $mydb->num_rows($cur);
        return $row_count;
    }
```

```
function single_department($id=0){
        global $mydb;
        $mydb->setQuery("SELECT * FROM ".self::$tblname."
            Where DepartmentID= '{$id}' LIMIT 1");
        $cur = $mydb->loadSingleResult();
        return $cur;

static function instantiate($record) {
    $object = new self;

    foreach($record as $attribute=>$value){
      if($object->has_attribute($attribute)) {
        $object->$attribute = $value;
      }
    }
    return $object;
}

private function has_attribute($attribute) {
  return array_key_exists($attribute, $this->attributes());
}

protected function attributes() {
  global $mydb;
  $attributes = array();
  foreach($this->dbfields() as $field) {
    if(property_exists($this, $field)) {
        $attributes[$field] = $this->$field;
    }
  }
  return $attributes;
}

protected function sanitized_attributes() {
  global $mydb;
  $clean_attributes = array();
```

```
  foreach($this->attributes() as $key => $value){
    $clean_attributes[$key] = $mydb->escape_value($value);
  }
  return $clean_attributes;
}

public function save() {
  return isset($this->id) ? $this->update() : $this->create();
}

public function create() {
    global $mydb;
    $attributes = $this->sanitized_attributes();
    $sql = "INSERT INTO ".self::$tblname." (";
    $sql .= join(", ", array_keys($attributes));
    $sql .= ") VALUES ('";
    $sql .= join("', '", array_values($attributes));
    $sql .= "')";
echo $mydb->setQuery($sql);

 if($mydb->executeQuery()) {
    $this->id = $mydb->insert_id();
    return true;
  } else {
    return false;
  }
}

public function update($id=0) {
  global $mydb;
    $attributes = $this->sanitized_attributes();
    $attribute_pairs = array();
    foreach($attributes as $key => $value) {
      $attribute_pairs[] = "{$key}='{$value}'";
    }
    $sql = "UPDATE ".self::$tblname." SET ";
    $sql .= join(", ", $attribute_pairs);
```

```
        $sql .= " WHERE DepartmentID=". $id;
      $mydb->setQuery($sql);
        if(!$mydb->executeQuery()) return false;

    }

    public function delete($id=0) {
        global $mydb;
          $sql = "DELETE FROM ".self::$tblname;
          $sql .= " WHERE DepartmentID=". $id;
          $sql .= " LIMIT 1 ";
          $mydb->setQuery($sql);

            if(!$mydb->executeQuery()) return false;

    }

}
?>
```

Home.php

```
<?php session_start();

if (!(isset($_SESSION['login']))) {

    header('location:../index.php');

}

?>
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">
```

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

    <link href="../bower_components/metisMenu/dist/metisMenu.min.css"
rel="stylesheet">

    <link href="../dist/css/sb-admin-2.css" rel="stylesheet">

    <link href="../bower_components/font-awesome/css/font-awesome.min.css"
rel="stylesheet" type="text/css">

</head>

<body>

    <div id="wrapper">
        <nav class="navbar navbar-default navbar-static-top"
role="navigation" style="margin-bottom: 0">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-
toggle="collapse" data-target=".navbar-collapse">
                    <span class="sr-only"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a class="navbar-brand" href="index.html">Система обліку
даних</a>
            </div>

            <ul class="nav navbar-top-links navbar-right">
                <li class="dropdown">
                    <a class="dropdown-toggle" data-toggle="dropdown"
href="#">
                        <i class="fa fa-user fa-fw"></i>  <i class="fa fa-
caret-down"></i>
                    </a>
                    <ul class="dropdown-menu dropdown-user">
                        <li><a href="profile.php"><i class="fa fa-user fa-
fw"></i> Профіль користувача</a>
                        </li>
                        <li><a href="options.php"><i class="fa fa-gear fa-
fw"></i> Налаштування</a>
                        </li>
                        <li class="divider"></li>
                        <li><a href="login.php"><i class="fa fa-sign-out fa-
fw"></i> Вихід</a>
                        </li>
                    </ul>
                </li>
```

```
            </ul>

            <div class="navbar-default sidebar" role="navigation">
                <div class="sidebar-nav navbar-collapse">
                    <ul class="nav" id="side-menu">

                        <li>
                       <a href="home.php"><i class="fa fa-home fa-fw"></i>
Головна сторінка</a>
                        </li>
                        <li>
                            <a href="#"><i class="fa fa-navicon fa-fw"></i>
Структура університету<span class="fa arrow"></span></a>
                            <ul class="nav nav-second-level">
                                <li>
                                    <a href="#"><i class="fa fa-university
fa-fw"></i> Кафедри<span class="fa arrow"></span></a>
                                    <ul class="nav nav-third-level">
                                        <li>
                                            <a href="add-department.php"><i
class="fa fa-edit fa-fw"></i> Додати кафедру</a>
                                        </li>
                                        <li>
                                            <a href="view-department.php"><i
class="fa fa-bar-chart fa-fw"></i> Переглянути кафедри</a>
                                        </li>
                                    </ul>
                                </li>
                                <li>
                                    <a href="#"><i class="fa fa-university
fa-fw"></i> Факультети<span class="fa arrow"></span></a>
                                    <ul class="nav nav-third-level">
                                        <li>
                                            <a href="add-faculty.php"><i
class="fa fa-edit fa-fw"></i> Додати факультет</a>
                                        </li>
                                        <li>
                                            <a href="view-faculty.php"><i
class="fa fa-bar-chart fa-fw"></i> Переглянути факультети</a>
                                        </li>
                                    </ul>
                                </li>
                                <li>
                                    <a href="#"><i class="fa fa-group fa-
fw"></i> Науково-педагогічний склад<span class="fa arrow"></span></a>
                                    <ul class="nav nav-third-level">
                                        <li>
                                            <a href="add-teacher.php"><i
class="fa fa-edit fa-fw"></i> Додати педпрацівника</a>
                                        </li>
```

```
                                        <li>
                                            <a href="view-teacher.php"><i
class="fa fa-bar-chart fa-fw"></i> Переглянути таблицю педпрацівників</a>
                                        </li>
                                    </ul>
                                </li>
                            </ul>
                        </li>
                        <li>
                            <a href="#"><i class="fa fa-navicon fa-fw"></i>
Дипломи та сертифікати<span class="fa arrow"></span></a>
                            <ul class="nav nav-second-level">
                                <li>
                                    <a href="view_diploma.php">Дипломи<span
class="fa arrow"></span></a>
                                </li>
                                <li>
                                    <a href="view-
sertificate.php">Сертифікати<span class="fa arrow"></span></a>
                                </li>
                            </ul>
                        </li>
                        <li>
                            <a href="#"><i class="fa fa-navicon fa-fw"></i>
Установи та організації<span class="fa arrow"></span></a>
                            <ul class="nav nav-second-level">
                                <li>
                                    <a href="add-organization.php">Додати
установу</a>
                                </li>
                                <li>
                                    <a href="view-course.php">Переглянути
установи</a>
                                </li>
                            </ul>
                        </li>
                        <li>
                            <a href="project_about.php"><i class="fa fa-
navicon fa-fw"></i> Про проєкт</a>
                        </li>
                        <li>
                            <a href="logout.php"><i class="fa fa-sign-out fa-
fw"></i> Вихід</a>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>

        <div id="page-wrapper">
```

```
            <div class="row">
                <div class="col-lg-12">
                    <h4 class="page-header"> <?php echo strtoupper("Ласкаво
просимо, "." ".htmlentities($_SESSION['login']));?></h4>
                </div>
            </div>
        </div>

    </div>

    <script src="../bower_components/jquery/dist/jquery.min.js"></script>
    <script
src="../bower_components/bootstrap/dist/js/bootstrap.min.js"></script>
    <script
src="../bower_components/metisMenu/dist/metisMenu.min.js"></script>
    <script src="../dist/js/sb-admin-2.js"></script>

</body>

</html>
```

ДОДАТОК Б

ПРЕЗЕНТАЦІЯ

Апробація результатів та публікації :

1. Крещанов М. О. Вишнівський В.В. Research of solutions of database management systems in the process of developing the accounting system of scientific and pedagogical employees. Матеріали II Всеукраїнської науково-технічної конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу» : Збірник тез, ДУІКТ, Київ, 18 листопада 2024р. 2024. С. 387–390. URL: https://duikt.edu.ua/uploads/p_2661_83654085.pdf
2. Kreshchanov M.O., Vyshnivskyi V.V. Modern trends in effective data management in the context of the constantly evolving information environment. Advanced discoveries of modern science: experience, approaches and innovations»: collection of scientific papers «SCIENTIA» with Proceedings of the VII International Scientific and Theoretical Conference, November 29, 2024. Amsterdam, The Netherlands: International Center of Scientific Research. 2024. P. 84–89. URL: https://previous.scientia.report/index.php/archive/issue/view/

СИСТЕМИ ЕЛЕКТРОННОГО КЕРУВАННЯ ДАНИМИ

Система управління корпоративним контентом (УКК) (Enterprise Content Management, ECM) – це програмне рішення для ефективної групової роботи та управління документами, що забезпечує організоване використання наявної інформації.

Базові модулі ECM :

- ✓ DM (Document Management) – додаток для управління документами;
- ✓ Collaboration – інструменти групової співпраці;
- ✓ WCM (Web Content Management) – програмний засіб управління вмістом сайту;
- ✓ RM (Records Management) – додаток, що забезпечує архівацію даних;
- ✓ Workflow / BPM – електронне керування документообігом та бізнес-процесами.

БАЗИ ДАНИХ У СФЕРІ ОСВІТИ

Інформаційні системи для студентів (SIS) – комплексні освітні бази даних для управління та організації інформації, яка пов'язана зі студентами.
Системи управління навчанням (LMS) – онлайн-платформи для надання, керування та відстеження освітнього контенту і діяльності.
Бази даних керування бібліотекою – універсальні системи спеціального призначення, які використовуються бібліотеками та навчальними закладами для каталогізації друкованих видань та цифрових ресурсів.
Адміністративні бази даних – бази даних для реалізації адміністративних функцій (системи реєстрації студентів, системи управління фінансами, бази даних кадрів тощо).
Дослідницькі бази даних – колекції наукових видань, журналів, статей, інші наукові публікації, що стосуються різноманітних галузевих досліджень.
Спеціалізовані освітні бази даних – стосуються конкретних навчальних дисциплін, освітніх спеціальностей та предметних галузей.
Довідкові центри для викладачів – електронні ресурси, що забезпечують доступ до навчально-методичних матеріалів.
Інформаційний центр освітніх ресурсів (ERIC) – системна база даних, яка спеціалізується на дослідженнях і ресурсах, пов'язаних з освітою.

БАЗИ ДАНИХ У СФЕРІ ОСВІТИ

Назва	Функції	Недоліки
ЄДИНА ДЕРЖАВНА ЕЛЕКТРОННА БАЗА З ПИТАНЬ ОСВІТИ **Уніфікована електронна система обліку ЄДЕБО**	✓ електронне ліцензування освітньої діяльності; ✓ акредитації освітніх програм та спеціальностей; ✓ електронні кабінети вступників (абітурієнтів); ✓ перебігу вступної кампанії (конкурсних пропозицій, рейтингових списків вступників тощо); ✓ перевірки достовірності документів про освіту; ✓ статистичні звіти у сферах фахової передвищої та вищої освіти тощо; ✓ реєстр сертифікатів педагогічних працівників.	✓ обмежений доступ до певних реєстрів; ✓ немає уніфікованого обліку науково-педагогічних працівників; ✓ відсутні документи про освіту до 2000 року, документи про наукові ступені доктора наук, кандидата наук та про вчені звання професора, доцента, старшого дослідника.

БАЗИ ДАНИХ У СФЕРІ ОСВІТИ

Назва	Функції	Недоліки
Єдина атестаційна система (ЄАС)	✓ реєстрація педагогічних та науково-педагогічних працівників; ✓ формування атестаційних листів (на підставі даних, внесених педагогами та атестаційними комісіями); ✓ оцінювання педагогічних працівників атестаційними комісіями на підставі атестаційних листів та інших документів; ✓ прийняття рішень про атестацію педагогів атестаційними комісіями на підставі їхніх оцінок.	✓ обмежені функціональні можливості; ✓ відсутність уніфікації даних; ✓ обмежений перелік навчальних закладів, які користуються системою.

БАЗИ ДАНИХ У СФЕРІ ОСВІТИ

Назва	Функції	Недоліки
Електронний застосунок єДокумент про освіту («Дія»)	✓ надає доступ до документів про освіту (свідоцтво про базову середню освіту, атестат/свідоцтво про повну загальну середню освіту, свідоцтво про присвоєння (підвищення) робітничої кваліфікації, диплом кваліфікованого робітника, диплом фахового молодшого бакалавра, диплом молодшого спеціаліста, диплом молодшого бакалавра, диплом бакалавра, диплом спеціаліста, диплом магістра, диплом доктора філософії).	✓ для відображення документа про освіту необхідно, щоб РНОКПП (Реєстраційний номер облікової картки платника податків) користувача в Дії збігався з РНОКПП у даних документа про освіту в ЄДЕБО; ✓ відсутні інші реєстраційні дані.

БАЗИ ДАНИХ У СФЕРІ ОСВІТИ

Назва	Функції	Недоліки
 **Автоматизований інформаційний комплекс освітнього менеджменту (АІКОМ)**	✓ збір актуальних даних та автоматична генерація звітності (автоматичне генерування документів); ✓ підвищення кваліфікації педагогічних працівників; ✓ реалізація можливості взаємодії АІКОМ з різними освітніми системами, базами даних; ✓ облік дітей дошкільного, шкільного віку та закладів дошкільної освіти; ✓ запровадження електронної системи управління професійно-технічною освітою.	✓ пілотне тестування проєкту; ✓ обмежені можливості для закладів вищої освіти; ✓ не реалізована взаємодія з державними реєстрами і базами даних в системі освіти; ✓ немає систематизації даних педагогічних та науково-педагогічних працівників.

ФУНКЦІОНАЛЬНІ МОЖЛИВОСТІ ТА КЛАСИФІКАЦІЯ СУБД

Система управління базами даних, СУБД (Database Management System, DBMS) – це спеціальне програмне забезпечення, що дозволяє ефективно взаємодіяти з базою даних.

Класифікація СУБД :

- ✓ за моделлю даних (ієрархічні, реляційні, мережеві, об'єктно-орієнтовані, об'єктно-реляційні);
- ✓ за функціональним призначенням (оперативні та аналітичні), розподілом (централізовані та розподілені);
- ✓ за типом використання (SQL-орієнтовані та NoSQL);
- ✓ за ліцензією (відкриті (Open Source), пропрієтарні (Commercial).

СУБД має виконувати обов'язкові вимоги :

- ✓ забезпечувати несуперечність даних, їх цілісність (пов'язані дані мають адекватно змінюватися);
- ✓ гарантувати актуальність даних та їх доступність;
- ✓ уникати надмірності даних (дані мають бути захищені від несанкціонованого доступу, зміни чи видалення);
- ✓ надавати можливість відновлення та резервного копіювання.

СТРУКТУРА РЕЛЯЦІЙНИХ СУБД

Структурні елементи SQL баз даних	Опис та функції
Реляційні таблиці	мають ім'я та структуру (рядки та стовпці), визначену схемою даних
Рядки (кортежі)	окремі записи, що містять інформацію про об'єкт, кожен запис є унікальним та ідентифікується за допомогою ключа
Стовпці (атрибути)	мають ім'я і тип даних (визначають інформацію про характеристики об'єктів, що описуються в таблиці і зберігаються в цьому стовпчику - текст, числа, дати тощо)
Ключі	використовуються для унікальної ідентифікації рядків у таблиці. Види ключів: - первинний (PRIMARY KEY) – унікальний, єдиний, найголовніший ключ; атрибут (набір атрибутів), що ідентифікує кортеж даного відношення та задається обмеженням; - вторинний (FOREIGN KEY) – зовнішній ключ, що визначає спосіб об'єднання таблиць; одне або кілька полів (стовпців) у таблиці, що містять посилання на певне поле або поля первинного ключа в іншій таблиці.

СТРУКТУРА РЕЛЯЦІЙНИХ СУБД

Структурні елементи SQL баз даних	Опис та функції
Зв'язки	дають змогу об'єднувати дані з різних таблиць для виконання складних запитів
SQL (Structured Query Language)	використовує стандарт ACID (Atomicity, Consistency, Isolation, Durability) для успішного виконання транзакцій; дає змогу створювати, змінювати, видаляти та витягувати дані з таблиць, а також визначати правила для цілісності даних
Нормалізація	зменшення надмірності даних і забезпечення їхньої цілісності
Транзакції	забезпечуються стандартом ACID, що є ефективним під час одночасного доступу кількох користувачів до бази даних

Приклади реляційних моделей баз даних: MySQL, MariaDB, Microsoft SQL Server, PostgreSQL, SQLite.
Сфери використання: корпоративні системи управління базами даних, веб-додатки, системи електронної комерції.

ВИДИ SQL ЗАПИТІВ

SQL запити		Призначення	Приклади
DDL (Data Definition Language)	мова визначення даних	створення БД та опис її структури (в якому вигляді різні дані будуть розміщуватися в БД)	ALTER COLLATE CREATE DROP DISABLE TRIGGER ENABLE TRIGGER RENAME UPDATE STATISTICS
DML (Data Manipulation Language)	мова маніпулювання даними	потрібні для додавання змін до вже внесених даних, для отримання даних з БД, для їх збереження, для оновлення різних записів і для їх видалення з БД	BULK INSERT SELECT DELETE UPDATE INSERT UPDATETEXT MERGE WRITETEXT READTEXT
DCL (Data Control Language)	мова управління даними	запити та команди, що стосуються дозволів, прав та інших налаштувань СУБД	GRANT REVOKE DENY
TCL (Transaction Control Language)	мова управління транзакціями	застосовують для керування змінами, які здійснюються з використанням DML-запитів (дозволяють проводити об'єднання DML запитів у набори транзакцій)	BEGIN COMMIT ROLLBACK

ТИПОЛОГІЯ ТА ПРИКЛАДИ NoSQL БАЗ ДАНИХ

Типи NoSQL баз даних	Функції та характеристики	Способи та сфери використання	Приклади
Бази даних «ключ-значення» (Key-Value Database)	пропонують базові функції для отримання значення, пов'язаного з ключем	для швидкого пошуку інформації за допомогою ключа, для кешування даних (для зберігання коментарів у блогах, оглядів продуктів, профілів користувачів і налаштувань)	Apache Ignite ArangoDB Couchbase FoundationDB InfinityDB MemcacheDB Oracle NoSQL Database OrientDB Redis Velocity
Документно-орієнтовані бази даних (Document-Oriented Database)	зберігають всю інформацію, пов'язану з об'єктом, в одному файлі XML, YAML, JSON, BSON	для систем керування вмістом, швидкого створення прототипів та аналізу даних (у видавничій справі, документальному пошуку тощо)	Apache CouchDB ArangoDB BaseX Couchbase IBM Notes MarkLogic MongoDB OrientDB RethinkDB XML-databases

ТИПОЛОГІЯ ТА ПРИКЛАДИ NoSQL БАЗ ДАНИХ

Типи NoSQL баз даних	Функції та характеристики	Способи та сфери використання	Приклади
Стовпчасті бази даних (Column-Oriented Database / Wide-Column Store)	орієнтовані на стовпці, зберігають кожен стовпець як логічний масив значень, характеризуються високою масштабованістю і можуть легко дублюватися, працюють зі структурованими та неструктурованими даними	для обробки аналітичних операцій	BigTable Cassandra Druid HBase Hypertable Qbase ScyllaDB Vertica
Графові бази даних / (Graph Database)	використовують структури графів для семантичних запитів, кожен вузол (вершина) є ізольованим документом із даними у довільній формі, вузли з'єднані ребрами, які визначають їхні відношення (властивості)	для проектів із графовими структурами даних (соціальні мережі семантична мережа)	AllegroGraph ArangoDB Datastax Enterprise InfiniteGraph Apache Giraph MarkLogic Neo4j OrientDB Virtuoso Universal Server

ПОРІВНЯННЯ РЕЛЯЦІЙНИХ ТА НЕРЕЛЯЦІЙНИХ БАЗ ДАНИХ

Характеристики СКБД	SQL	NoSQL
Структура	Формат таблиць: складаються з рядків (кортежів) і стовпців (атрибутів)	Формат документу, ключ-значення, графік або сховище у широких стовпцях
Підтримка запитів	Використовує SQL (Structured query language), мову структурованих запитів для взаємодії користувача з базами даних Підтримує JOIN (операцію з'єднання таблиць в SQL) і складні запити	Використовує «віртуальну об'єктну базу даних» ORM (Object-Relational Mapping), об'єктно-реляційне зіставлення Не підтримує JOIN і складні запити
Властивості	Сумісні зі стандартом ACID для забезпечення високої надійності керування транзакціями	Більшість не підтримують стандарт ACID (виключення, наприклад, MongoDB)
Масштабованість	Вертикальне масштабування (Vertical Database Scalability) із розширенням за допомогою більшого сервера	Горизонтальне масштабування (Horizontal Database Scalability) на кількох серверах
Цілі використання	Для роботи зі структурованими даними за попередньо визначеною схемою	Для роботи з неструктурованими (або напівструктурованими) даними з динамічною схемою

ПОРІВНЯННЯ РЕЛЯЦІЙНИХ ТА НЕРЕЛЯЦІЙНИХ БАЗ ДАНИХ

Характеристики СКБД	SQL	NoSQL
Переваги	– просте використання та моделювання даних; – незалежність даних від фізичної структури зберігання; – велика підтримка SQL для запитів і маніпуляції даними; – добре працює з більшістю фреймворків і бібліотек; – стабільність, безпечність, безкоштовність.	– масштабованість (пристосовані до зростаючих робочих навантажень); – гнучкість (дозволяють обробляти дані з різними структурами без обмежень фіксованої схеми); – висока відмовостійкість; – можливість додати більше серверів до кластера.
Недоліки	– неефективно працюють із напівструктурованими або неструктурованими даними; – зменшення продуктивності під час роботи з великими обсягами даних; – працюють лише на одному сервері.	– обмежені транзакції ACID (не підходять для програм, які вимагають суворої цілісності даних); – відсутність стандартизованої мови запитів; – складно організувати зберігання та обробку не структурованих даних.
Використання	Фінансові системи, CRM, програмне забезпечення для планування ресурсів підприємства (ERP)	Веб-програми, аналітика в реальному часі, програми IoT, CMS
Приклади	MySQL, Oracle, Microsoft SQL Server	MongoDB, Cassandra, CouchDB, Redis

ДОСЛІДЖЕННЯ ПОПУЛЯРНИХ РІШЕНЬ СУБД

СУБД	Переваги	Недоліки
Oracle	– системна обробка онлайн-транзакцій (OLTP) та сховище даних; – надзвичайна висока продуктивність; – масштабованість; – безпека, надійність; – легкість управління; – мульти-модельні зв'язки і покращена обробку даних (інтеграція AutoML); – гарна технічна підтримка.	– комерційний проєкт, призначений для великих підприємств і бізнес компаній, з обмеженим використанням у некомерційних структурах; – безкоштовна версія Express Edition (XE) має оперативну пам'ять лише 1 ГБ і максимальний об'єм бази даних 11 ГБ [16].
MySQL	– відкритий вихідний код; – забезпечує продуктивність і швидкість завантаження; – підтримує стандарт ACID; – проста та доступна; – має вичерпну документацію та значну підтримку; – підтримує сумісність з будь-якою Web- та мобільною програмою; – підтримується C, C++, C#, PHP, Java, Ruby, Python, Perl; – забезпечує безпеку даних, збереження їх цілісності та конфіденційності.	– деякі функції та плагіни доступні лише для пропрієтарних версій); – відсутня підтримка FULL JOIN пунктів, не повністю реалізує стандарт SQL; – втрата швидкості програми, якщо багато користувачів записуватимуть дані одночасно.

ДОСЛІДЖЕННЯ ПОПУЛЯРНИХ РІШЕНЬ СУБД

СУБД	Переваги	Недоліки
Microsoft SQL Server	– масштабування; – автоматизація адміністративних завдань; – зручний пошук за допомогою ключових фраз, слів, текстів, індексів; – підтримка роботи з іншими додатками Microsoft (Excel, Access тощо); – багатомовну підтримка, підтримує Open Database Connectivity (ODBC); – може залучати .NET, ASP.NET, .NET Core, для роботи зі структурованими і неструктурованими даними; – використовує середовища Azure SQL Database, Azure Cosmos DB, MySQL; – онлайн-аналітична обробка, наявність вбудованих служб аналізу для створення і керування OLAP; – підтримує класифікацію та кластеризацію баз даних; – підтримує автоматичні сповіщення та надлишкове дублювання даних.	– повна залежністю від операційної системи (працює лише з Windows); – висока ціна.

ДОСЛІДЖЕННЯ ПОПУЛЯРНИХ РІШЕНЬ СУБД

СУБД	Переваги	Недоліки
PostgreSQL	– відкритий вихідний код; – ефективна в роботі з великою кількістю записів та індексів у таблиці; – здатна реалізувати складні та надійні механізми транзакцій та реплікації; – чітке дотримання стандартів SQL (ACID); – підтримує складні структури; – реалізована підтримка XML, JSON і NoSQL баз дани; – активна підтримка; – використання каталогів для швидкого розширення.	– проблема продуктивності пам'яті (кожному новому процесу виділяється близько 10 МБ пам'яті); – не підходить для важких операцій читання і реалізує функцію реплікації лише за допомогою розширень; – окремі налаштування, впровадження, модифікація та підтримка платні.
MongoDB	– відкритий вихідний код типу NoSQL; – забезпечує максимальну доступність даних для користувачів; – висока продуктивність і швидкість; – можливість запуску на кількох серверах одночасно; – зберігання бінарних даних великих обсягів (фото / відео); – ефективно реалізує ієрархічні зв'язки, демонструє якісну роботу з масивами.	– проблеми з безпекою; – випадки втрати контролю над серверами; – проблеми з втратою даних.

ТЕХНОЛОГІЇ ДЛЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ НАУКОВО-ПЕДАГОГІЧНИХ ПРАЦІВНИКІВ

Інструментальні засоби для розробки системи

- ✓ Microsoft Visual Studio Code (VS Code)
- ✓ кросплатформний пакет XAMPP
- ✓ скриптова мова програмування PHP
- ✓ об'єктно-орієнтована прототипна мова JavaScript (JS)
- ✓ реляційна система управління базами даних MySQL

РЕАЛІЗАЦІЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ

Створення бази даних

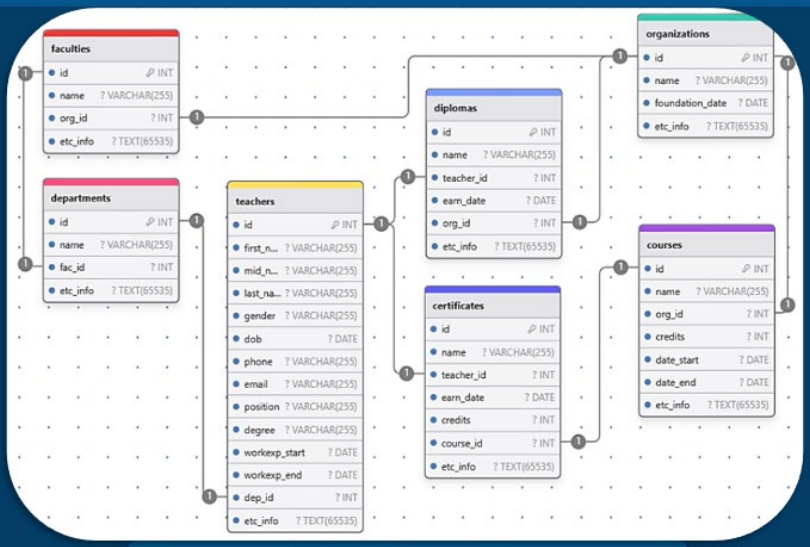

Рисунок 22.1 – Схема бази даних

РЕАЛІЗАЦІЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ

Розробка програмного забезпечення

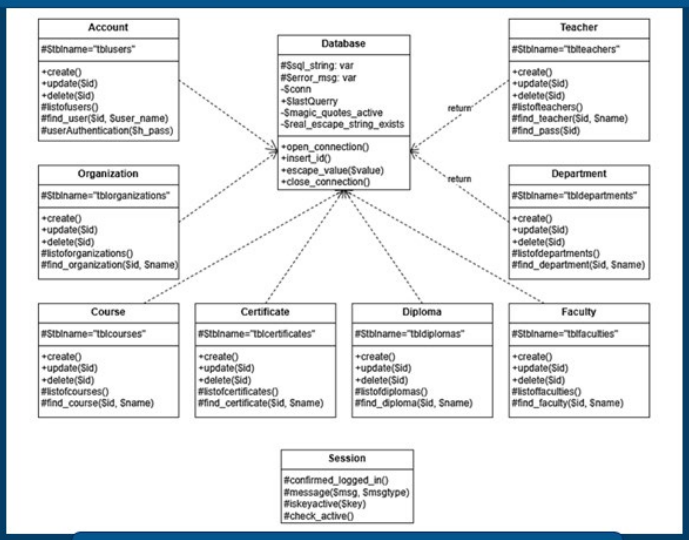

Рисунок 23.1 – Діаграма класів

РЕАЛІЗАЦІЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ

Розробка програмного забезпечення

Спроектовані та розроблені класи:

Назва класу	Призначення та функції
Database	забезпечує взаємодію з БД для керування з'єднанням з нею
Session	слугує для забезпечення взаємодії з БД для керування онлайн-сесіями користувачів в системі
Account	забезпечує створення, редагування, оновлення та пошук інформації щодо акаунтів користувачів, їх активності та планування проходження курсів
Faculty	призначений для створення, редагування, оновлення та пошуку інформації щодо факультетів
Department	надає можливість створення, редагування, оновлення та пошуку інформації щодо кафедр

РЕАЛІЗАЦІЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ

Розробка програмного забезпечення

Спроектовані та розроблені класи:

Diploma	Призначення та функції
Teacher	слугує для створення, редагування, оновлення та пошуку інформації щодо педагогічних та науково-педагогічних працівників, їх кваліфікації
Diploma	надає можливість створення, редагування, оновлення та пошуку інформації щодо дипломів, які належать певним науково-педагогічним працівникам
Certificate	призначений для створення, редагування, оновлення та пошуку інформації щодо сертифікатів педагогічних та науково-педагогічних працівників
Course	призначений для створення, редагування, оновлення та пошуку інформації щодо курсів
Organization	призначений для створення, редагування, оновлення та пошуку інформації щодо організацій, які забезпечують проходження курсів

РЕАЛІЗАЦІЯ РОЗРОБКИ СИСТЕМИ СИСТЕМИ ОБЛІКУ СОД

Інструкція по використанню системи

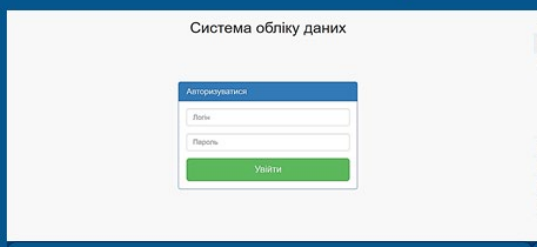

Рисунок 12.1 – Вікно авторизації

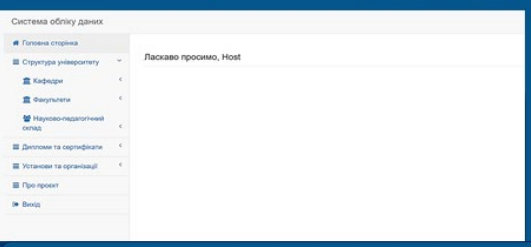

Рисунок 12.3– Головна сторінка

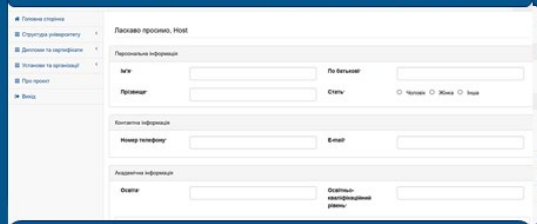

Рисунок 12.2 – Вікно внесення даних про викладача

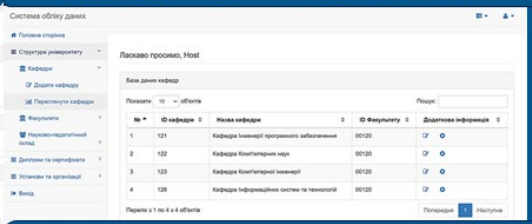

Рисунок 12.4 – Вікно перегляду таблиці кафедр

ВИСНОВКИ

1. У кваліфікаційній роботі здійснені прикладні дослідження, що стосувались наявних освітніх електронних систем обліку (ЄДЕБО, Єдина атестаційна система (ЄАС), Єдиний портал державних послуг «Дія», автоматизований інформаційний комплекс АІКОМ).

2. В процесі досліджень виявлено, що наявні облікові системи освітніх даних не враховують у повній мірі потреби навчальних закладів щодо узагальнення особових даних педагогічних та науково-педагогічних працівників, які стосуються атестації, підвищення кваліфікації, сертифікації та автоматизації процесів управління і звітності.

3. Проведені ґрунтовні дослідження систем управління базами даних: порівняльний аналіз реляційних та нереляційних моделей баз даних, технологічних вимог щодо їх застосування та можливих варіантів програмних стеків.

4. Були розглянуті та проаналізовані технології, які дозволили розробити програмний засіб (система обліку педагогічних та науково-педагогічних працівників), а саме: редактор вихідного коду Microsoft Visual Studio Code (VS Code), кросплатформний пакет XAMPP для роботи з HTTP веб-сервером Apache, скриптова мова програмування PHP, реляційна система управління базами даних MySQL та об'єктно-орієнтована прототипна мова програмування JavaScript.

5. Також були описані основні процеси щодо розробки ПЗ, тестування, складання інструкції користувача по використанню системи.

6. Результатом кваліфікаційної роботи стала така система обліку педагогічних та науково-педагогічних працівників, яка дозволяє сформувати електронну базу даних працівників освітнього закладу для систематизації особових даних, інформації щодо підвищення кваліфікації, сертифікації та атестації. Система планується покращуватися шляхом додавання нового функціоналу. Подальші дослідження в цій площині допоможуть створити уніфіковану модель системи обліку педагогічних і науково-педагогічних працівників, яка здатна забезпечити дотримання європейських стандартів щодо сертифікації та якості освіти.