

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 126 «Інформаційні системи та технології»
(освітня програма «Системи бізнес-аналітики»)

на тему:

**«Розробка інформаційної системи «Помічник
викладача»»»**

Виконав студент групи ІСТ-20-1
ВАЛЕЦЬКИЙ Максим Сергійович

Керівник роботи:
ГРАФ Марина Сергіївна

Рецензент:
СВІНЦИЦЬКА Олександра Миколаївна

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

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
Марина ГРАФ
«25» січня 2024 р.

ЗАВДАННЯ

на кваліфікаційну роботу освітнього ступеня «бакалавр»
за спеціальністю 126 «Інформаційні системи та технології»
(освітня програма «Системи бізнес-аналітики»)

Здобувач вищої освіти: **ВАЛЕЦЬКИЙ Максим Сергійович**

Керівник роботи: **ГРАФ Марина Сергіївна**

Тема роботи: **Розробка інформаційної системи «Помічник викладача»**

затверджена наказом закладу вищої освіти від **«25» січня 2024 р., №24/с**

Термін здачі закінченої роботи: **«05» червня 2024 р.**

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

Об'єктом дослідження є процес автоматизації робочих процесів у вищих навчальних закладах.

Предметом дослідження є методи та засоби автоматизації робочих процесів процесів у вищих навчальних закладах.

Консультанти випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Дата	
		Завдання видав	Завдання прийняв
1	ГРАФ М.С.	05.02.2024	07.02.2024
2	ГРАФ М.С.	12.03.2024	15.03.2024
3	ГРАФ М.С.	08.04.2024	30.04.2024

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Пошук, огляд та аналіз аналогічних розробок. Формування технічного завдання. Опрацювання літературних джерел.	2 лютого 2024 – 5 березня 2024	Виконано
2	Проектування структури системи	6 березня 2024 – 25 березня 2024	Виконано
3	Написання програмного коду	26 березня 2024 – 26 квітня 2024	Виконано
4	Тестування системи	27 квітня 2024 – 14 травня 2024	Виконано
5	Оформлення пояснювальної записки	15 травня 2024 – 1 червня 2024	Виконано
6	Попередній захист	6 червня 2024	Виконано

Здобувач вищої освіти **Максим ВАЛЕЦЬКИЙ**
Керівник **Марина ГРАФ**

РЕФЕРАТ

Кваліфікаційна робота бакалавра складається з інформаційної системи та пояснювальної записки. Пояснювальна записка до кваліфікаційної роботи містить 79 сторінку, 57 ілюстрацій, 5 таблиць, 4 додатків та 17 використаних джерел та складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатків.

Текстова частина викладена на 68 сторінках друкованого тексту.

Метою роботи є розробка веб-додатку який буде допомагати викладачам шляхом автоматизації та цифровізації деяких робочих процесів.

В роботі визначено основні завдання на розробку веб-додатку, продемонстровано основний функціонал, сформовано вимоги до системи. Спроековано архітектуру додатку, розроблено сучасний та зручний інтерфейс.

КЛЮЧОВІ СЛОВА: ВЕБ-ДОДАТОК, АВТОМАТИЗАЦІЯ, JET BRAINS, ПОМІЧНИК ВИКЛАДАЧА, ІНФОРМАЦІЙНА СИСТЕМА, PHP, BOOTSTRAP, MVC, HTML, CSS, JS.

					ІСТ.КР.Б – 126 – 24 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.	М.С. ВАЛЕЦЬКИЙ				Розробка інформаційної системи «Помічник викладача»	Літ.	Арк.	Аркушів
Керівник	М.С. ГРАФ						3	79
Рецензент	О.М. СВИНЦІЦЬКА				Пояснювальна записка	Житомирська політехніка, група ІСТ-20-1		
Н. контр.								
Зав. каф.	М.С. ГРАФ							

ABSTRACT

The bachelor's thesis consists of an information system and an explanatory note. The explanatory note to the qualification work contains 79 pages, 57 illustrations, 5 tables, 4 appendices and 17 references and consists of a list of abbreviations, introduction, three chapters, conclusions, a list of references and appendices. The text part is presented on 68 pages of printed text.

The aim of the work is to develop a web application that will assist teachers by automating and digitizing some work processes.

The work defines the main tasks for the development of the web application, demonstrates the main functionality, and forms requirements for the system. The architecture of the application has been designed, and a modern and convenient interface has been developed.

KEYWORDS: WEB APPLICATION, AUTOMATION, JET BRAINS, TEACHER'S ASSISTANT, INFORMATION SYSTEM, PHP, BOOTSTRAP, MVC, HTML, CSS, JS.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ	7
ВСТУП.....	9
РОЗДІЛ 1. АНАЛІЗ ВИМОГ КОРИСТУВАЧА ТА КОНЦЕПТУАЛЬНЕ МОДЕЛЮВАННЯ	11
1.1 Технічне завдання на розробку системи.....	11
1.2 Вибір архітектури для побудови інформаційної системи	13
1.3 Інструментарій реалізації та вимоги до апаратних засобів для інформаційної системи	18
Висновки до першого розділу	22
РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ	24
2.1 Визначення варіантів використання системи.....	24
2.2 Проєктування архітектури та узагальненої структури системи.....	26
2.3 Проєктування структури та розробка бази даних системи.....	29
2.4 Розробка основних функціональних алгоритмів системи	41
Висновки до другого розділу	42
РОЗДІЛ 3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ, ОГЛЯД ІНТЕРФЕЙСУ ТА ЇЇ ТЕСТУВАННЯ	44
3.1 Розробка інформаційної системи	44
3.2 Огляд інтерфейсу інформаційної системи.....	48
3.3 Тестування інформаційної системи	62
Висновки до третього розділу	65
ВИСНОВОК	66
ДОДАТКИ	69
Додаток А.....	70
Додаток Б.....	71
Додаток В	72
Додаток Г	75

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

UI — візуальна частина, зовнішній вигляд сайту чи застосунку (User Interface)

UX — це користувацький досвід, те, наскільки продукт інтуїтивно зрозумілий і які наші потреби вирішує (User Experience)

Bootstrap — це безкоштовний набір інструментів з відкритим кодом, призначений для створення веб-сайтів та веб застосунків, який містить шаблони CSS та HTML

SQL — декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД

БД – база даних

MVC (Model-View-Controller) — архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення.

Frontend — це публічна частина web-додатків (веб-сайтів), з якою користувач може взаємодіяти і контактувати напряму.

Backend — це програмно-апаратна частина проєкту, все те, що відбувається на стороні сервера

HTML — стандартизована мова розмітки документів для перегляду веб сторінок у браузері (HyperText Markup Language)

CSS — це спеціальна мова стилю сторінок, що використовується для опису їхнього зовнішнього вигляду (Cascading Style Sheets)

JavaScript — динамічна, об'єктно-орієнтована прототипна мова програмування

PHP — скриптова мова програмування, була створена для генерації HTML-сторінок на стороні веб сервера

Python — інтерпретована об'єктно-орієнтована мова програмування високого рівня із суворою динамічною типізацією

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		7

IDE — Інтегроване середовище розробки — комплексне програмне рішення для розробки програмного забезпечення (Integrated Drive Electronics)

Xdebug — розширення PHP, яке надає можливості зневадження і профілювання.

СУБД — набір взаємопов'язаних даних і програм для доступу до цих даних. (Система управління базами даних)

MySQL — вільна система керування реляційними базами даних (My Structured Query Language)

HTTPS – схема URL, що використовується для доступу до ресурсів Інтернет (HyperText Transfer Protocol Secure)

Apache - відкритий веб сервер Інтернет

ІС – інформаційна система

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		8

ВСТУП

Актуальність теми. Активний розвиток нових технологій у інформаційній сфері дозволяють нам автоматизовувати та оцифровувати практично будь-який процес. Такий бурхливий розвиток призвів до появи тенденції оцифровування багатьох повсякденних речей, наприклад запису до лікаря, отримання документів та ін. З впевненістю можна сказати що сьогодні інформаційні системи оточують нас повсюди та торкаються людей усіх професій, в тому числі викладачів.

В сучасному світі освітній процес потребує ефективних рішень для оптимізації та полегшення роботи викладачів. Організація навчальних матеріалів, складання освітніх програм та багато інших задач потребують багато часу й зусиль. Використання інформаційної системи може значно спростити велику кількість робочих процесів, дозволяючи викладачам більше часу приділити безпосередньо навчанню та взаємодії зі студентами.

Загалом ми можемо спостерігати тенденцію впровадження інформаційних технологій в освітній процес не тільки в Україні, а й у всьому світі. Саме тому обрана тематика є дуже актуальною та перспективною.

Мета дослідження. Метою дослідження, є розробка інформаційної для автоматизації робочого процесу формування освітніх програм у закладах вищої освіти.

Об'єкт дослідження. Об'єктом дослідження є інформаційні системи та технології для автоматизації робочих процесів.

Предмет дослідження. Предметом дослідження є процес автоматизації робочих процесів у вищих навчальних закладах.

Методи дослідження. В кваліфікаційній роботі використано: результати порівняльного аналізу, прогностного аналізу, а також алгоритми оптимізації.

Практичне значення отриманих результатів. У результаті виконання кваліфікаційної роботи, буде розроблено повноцінний веб-додаток, який може бути використаний будь-яким закладом вищої освіти та допоможе оптимізувати навчальний процес.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		9

Апробація результатів. Матеріали роботи були представлені на всеукраїнській науково-технічній конференції, 29-30 листопада 2023 року та Всеукраїнській науково-практичній конференції студентів, аспірантів, молодих учених «Актуальні напрями досліджень молодих учених в іншомовному просторі», 25 квітня 2024 року.

Публікації. За темою кваліфікаційної роботи було опубліковано 2 тези:

1. Граф М.С., Валецький М.С., Аналіз сучасних тенденцій використання штучного інтелекту при розробці інформаційних систем. Комп'ютерні технології: інновації, проблеми, рішення: тези VI Всеукраїнської науково-технічної конференції, 29-30 листопада 2023 року. Житомир : «Житомирська політехніка», 2023. С.259-261
2. Граф М.С., Валецький М.С., Шадур В.А., The Importance of Automation Process. Current Trends in Young Scientists' Research: XI All Ukrainian Scientific and Practical Conference, 25 квітня 2024 року. Житомир : «Житомирська політехніка», 2023. С.127-129

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		10

РОЗДІЛ 1. АНАЛІЗ ВИМОГ КОРИСТУВАЧА ТА КОНЦЕПТУАЛЬНЕ МОДЕЛЮВАННЯ

1.1 Технічне завдання на розробку системи

1. Назва системи:

Повне найменування: Інформаційна система «Помічник викладача»

Коротке найменування: «Помічник викладача»

2. Призначення системи:

Інформаційна система «Помічник викладача» може значно підвищити продуктивність та ефективність навчального процесу в навчальних закладах вищої освіти. Система може зменшити обсяг роботи викладачів, що дозволить їм сфокусуватись на інших, не менш важливих завданнях.

3. Загальні вимоги до системи:

- Додаток, що відображає роботу ІС має бути веб-орієнтованим. Користувачі повинні мати доступ до системи з будь-якого пристрою, в якого є доступ до мережі Інтернет.
- Дизайн та інтерфейс ІС повинні бути сучасними та водночас простими та інтуїтивно зрозумілими.
- Система повинна реалізовувати систему ролей, щоб чітко розподілити обов'язки та привілеї користувачів. Обов'язково має бути передбачено роль для викладача, адміністратора навчального закладу та гостя (ним можуть бути студенти, батьки студентів і т.д.)
- Система має бути стресостійкою та має стабільно працювати навіть при великій кількості користувачів, які одночасно нею користуються.

4. Вимоги до інтерфейсу

Основна вимога до інтерфейсу – компактність та зрозумілість, ці дві умови необхідні для того щоб користувачі не витрачали надто багато часу на опанування додатком. Також інтерфейс повинен бути розроблений за всіма сучасними правилами UI/UX дизайну, а також дотримувався стандартів правильного розміщення елементів [3]. Додатково необхідно забезпечити

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		11

адаптивність додатку, адже деяким користувачам простіше користуватись додатком з телефону чи планшету, і саме тому необхідно забезпечити коректне відображення усіх компонентів на екрані будь-якого розміру. Досягти це можна за допомогою адаптивної верстки з використанням спеціальних стилів фреймворку bootstrap [6] або з використанням media-запитів [5].

5. Функціональні вимоги

Для зручності, умовно поділимо інформаційну систему на модулі, та складемо вимоги для кожного з них:

Модуль «Користувачі»:

- Авторизація користувачів
- Реєстрація користувачів
- Видалення користувачів

Модуль «Факультети»:

- Перегляд списку всіх факультетів
- Редагування інформації про факультет
- Видалення факультету
- Додавання нових факультетів

Модуль «Кафедри»:

- Перегляд всіх кафедр певного факультету
- Додавання кафедри до факультету
- Редагування інформації про кафедри
- Видалення кафедри
- Відображення кількості викладачів кафедри
- Перегляд списку викладачів кафедри

Модуль «Адміністративна панель»:

- Додавання нового викладача
- Додавання нового адміністратора

Модуль «Спеціальності»:

- Перегляд списку спеціальностей певної кафедри

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				12
Змн.	Арк.	№ докум.	Підпис	Дата		

- Перегляд інформації про певну спеціальність
- Редагування інформації про певну спеціальність
- Видалення певної спеціальності

Модуль «Профіль»:

- Перегляд особистої інформації
- Редагування особистої інформації

Модуль «Освітні програми»:

- Перегляд компонентів освітньої програми
- Редагування компонентів освітньої програми
- Видалення компонентів освітньої програми
- Перегляд списку освітніх програм

6. Нефункціональні вимоги:

- Час, необхідний для навчання роботи з системою, повинен складати не більше 3 годин
- Система має забезпечувати стабільний доступ понад 90% часу
- Система має підтримувати стабільну роботу при великій кількості одночасно працюючих користувачів
- Система має бути легко масштабованою та відкритою до додавання нового функціоналу, без потреби переписування старого

Отож, в межах даного пункту було сформовано технічне завдання яке містить різноманітні вимоги до майбутньої інформаційної системи.

7. Додаткові вимоги:

В результаті проведеного прогнозного аналізу в рамках дослідження [7] потрібно врахувати висвітлену тенденцію у використанні штучного інтелекту у розробці інформаційної системи та забезпечити плацдарм для використання моделі штучного інтелект. Для цього можна використати API відомої мовної моделі, такої як OpenAI.

1.2 Вибір архітектури для побудови інформаційної системи

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				13
Змн.	Арк.	№ докум.	Підпис	Дата		

Вибір архітектури є надзвичайно важливим етапом розробки, оскільки від цього залежить функціональність, продуктивність, масштабованість та зручність використання системи. У виборі архітектури можна виділити два аспекти: 1) Вибір архітектурної моделі; 2) Вибір архітектурного шаблону; Розпочнемо з вибору архітектурної моделі, або архітектурного паттерну. Проаналізувавши статтю [4] складемо порівняльну таблицю архітектурних паттернів 1.1.

Таблиця 1.1 Порівняльна таблиця архітектурних паттернів

Критерій	Клієнт-серверна архітектура	Одношарова архітектура (Монолітна архітектура)	Мікро сервісна архітектура	P2P (Peer-to-Peer) архітектура
Розподіл обов'язків	Розподіл між клієнтом та сервером	Всі функції виконуються на одному рівні	Розподілені між мікро сервісами	Розподіляється між вузлами
Масштабованість	Висока, легко додавати клієнти та сервери	Обмежена	Дуже висока, кожен мікро сервіс може незалежно масштабуватися	Висока, вузли можуть динамічно додаватися та видалятися
Безпека	Висока, за рахунок централізованого контролю доступу	Низька	Висока, кожен мікро сервіс може мати свої заходи безпеки	Середній рівень безпеки, за рахунок важкості забезпечення централізованості
Управління даними	Дані зберігаються та управляються централізовано	Всі дані зберігаються в одному місці	Дані можуть бути розподілені між сервісами	Дані розподілені між вузлами, що спричиняє труднощі
Важкість забезпечення віддаленого доступу	Легко	Віддалений доступ є обмеженим, а тому важко його забезпечити	Легко	Легко
Продуктивність	Висока, за умови оптимальної	Може бути низькою через	Висока, про те залежить від	Залежить від кількості та потужності вузлів

	конфігурації сервера	обмеженість ресурсів	конфігурації сервісів	
Важкість реалізації	Легко	Найпростіша у реалізації	Складна у розробці та підтримці	Складна через необхідність забезпечення децентралізованості

В результаті проведеного порівняльного аналізу, можна зробити наступний висновок: Одношарова або монолітна архітектура добре підходить для реалізації простих та невеликих проектів, а оскільки передбачається що «Помічник викладача» не буде простою системою, можна сказати що даний архітектурний паттерн не є підходящим варіантом; Мікро сервісна та реер-to-реер архітектури є дуже хорошими варіантами через цілу низку переваг, про те складність їх реалізації не дозволяє обрати їх як остаточний варіант.

Зважаючи на наведені вище факти, остаточний вибір архітектури падає на клієнт-серверну архітектуру.

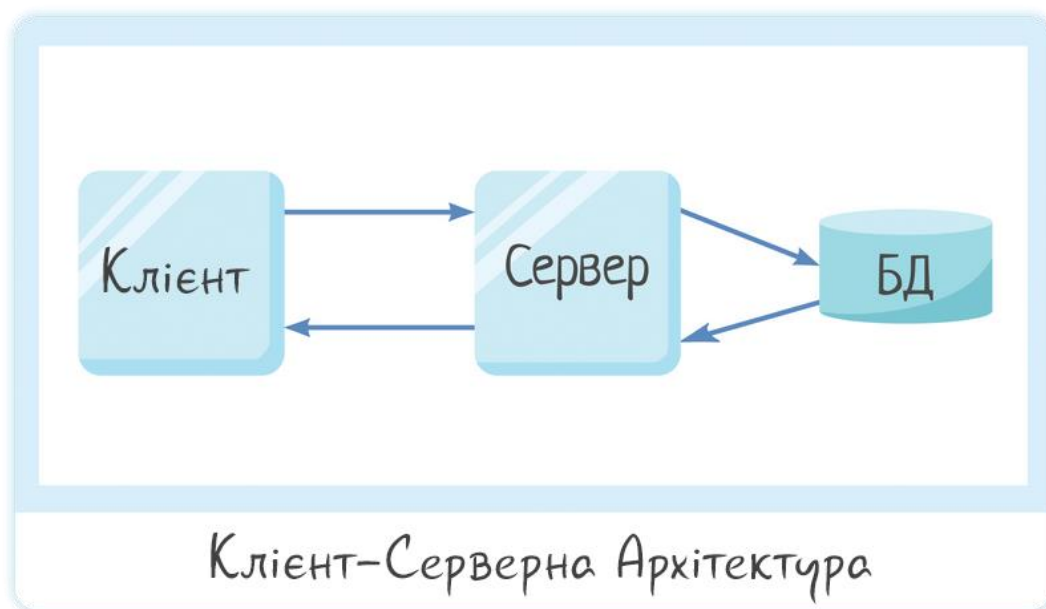


Рисунок 1.1 – Клієнт-Серверна архітектура

Згідно зі статтею [8] та з рисунку 1.1 видно, що дана архітектура передбачає використанні трьох компонентів: Клієнту, Серверу, та Баз даних. Клієнт необхідний для надання користувачам інтерфейсу для роботи з програмою. Прості користувачі не розуміють мови програмування чи SQL, і саме клієнт перетворює програмний код на зручне та зрозуміле візуальне подання.

Сервер є посередником між Клієнтом та Базою даних, в ньому зберігається вся логіка роботи додатку, та саме він відповідає за обробку запитів Клієнтів та відповідей на ці запити від Бази даних. Такий підхід дозволяє зекономити купу ресурсів, адже якщо наприклад велика компанія вводить дану архітектуру, то їй достатньо вкласти кошти у власний сервер, і тоді вона не буде залежати від потужностей Клієнтів. Також дана архітектура унеможлиблює дублювання коду, адже весь код програми знаходиться на Сервері, а клієнтська частина відповідає виключно за візуалізацію та прості перевірки даних. До того ж, прості користувачі не матимуть доступ до ключової інформації, що забезпечує безпеку та конфіденційність даних.

В рамках інформаційної системи «Помічник викладача» вибір даної архітектури є хорошим, з наступних причин:

- 1) Викладачі зможуть централізовано зберігати, оновлювати свої дані, дані освітніх програм, та ін. Адміністрація в свою чергу, зможе централізовано зберігати та управляти інформацією про викладачів та керувати іншими організаційними даними. І одночасно з цим, Гості можуть отримувати доступ до публічної інформації без необхідності зберігання особистих даних.
- 2) Викладачі та адміністрація можуть отримувати доступ до системи з будь-якого місця де є мережа Інтернет
- 3) Серверна частина може забезпечити високий рівень захисту конфіденційних даних, а також може забезпечити різні рівні доступу для викладачів, адміністрації та гостей.
- 4) Використання даної архітектури забезпечить масштабованість додатку, що означає, що зміни чи новий функціонал можуть бути легко імплементовані

Обравши архітектурну модель, тепер потрібно обрати архітектурний шаблон. Проаналізувавши статтю [9] складемо порівняльну таблицю.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		16

Таблиця 1.2. Порівняльна таблиця архітектурних шаблонів

Параметр	MVC (Model-View-Controller)	MVP (Model-View-Presenter)	MVVM (Model-View-ViewModel)
Основний принцип роботи	Розділення моделі, представлення та контролера.	Презентер як посередник між моделлю та представленням.	Використання прив'язки даних між View та ViewModel.
Логіка роботи компонентів	Контролер обробляє вхідні дані, оновлює модель і відображає View.	Презентер обробляє логіку і оновлює представлення.	ViewModel управляє станом і логікою представлення.
Простота реалізації	Середня, потребує суворого дотримання структури.	Відносно проста	Складніша, про те надає потужні можливості прив'язки даних.
Основна сфера використання	Веб-додатки	Десктопні та мобільні додатки	Десктопні додатки

В підсумку ми можемо спостерігати що кожен з архітектурних шаблонів має свої переваги та недоліки, про те для інформаційної системи «Помічник викладача» найкращим вибором буде паттерн MVC, оскільки інформаційна система буде представлена у вигляді веб-додатку.

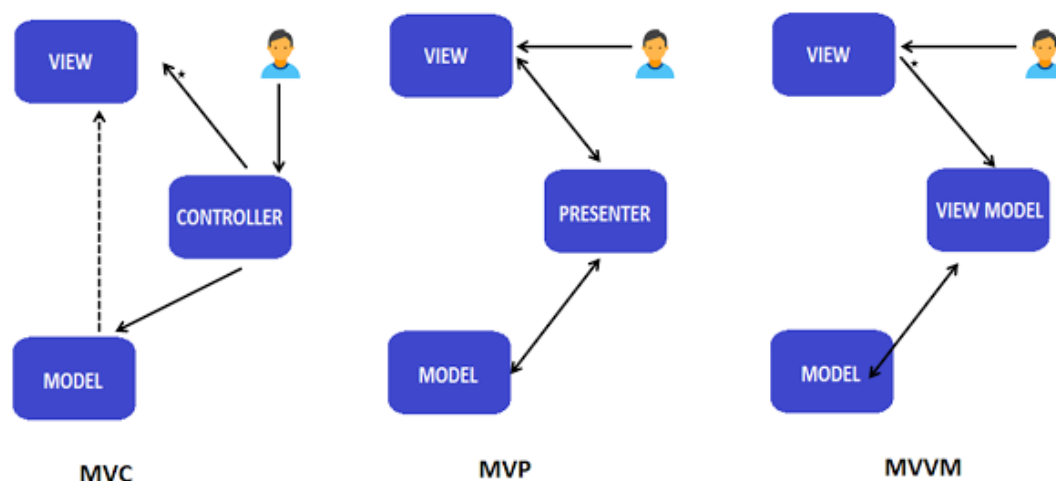


Рисунок 1.2 Візуалізація логіки роботи паттернів

До того ж, взаємодія клієнт-серверної архітектури та паттерну MVC є вдалим поєднанням, оскільки вони доповнюють один одного, забезпечуючи

структурованість, розділення обов'язків і гнучкість у розробці програмного забезпечення.

В підтвердження цьому наведемо приклад типового веб-додатку:

Клієнтська частина (Frontend):

- Використовує HTML, CSS, JavaScript для побудови інтерфейсу.
- Виконує запити до сервера

Серверна частина (Backend):

- Використовує MVC для організації логіки, де Controller обробляє запити, взаємодіє з Model для отримання даних та передає їх у View, який генерує відповідь для клієнта.

1.3 Інструментарій реалізації та вимоги до апаратних засобів для інформаційної системи

1. Інструменти розробки

1.1 Мови програмування та технології:

Клієнтська частина

- Відповідно до статті [1] обов'язковим є використання сучасних фронтенд технологій та фреймворків (наприклад Bootstrap, jQuery і т.д)
- Необхідно забезпечити виконання клієнтської валідації даних перед відправкою на сервер.

Серверна частина

Для вибору мови програмування серверної частини, спочатку потрібно проаналізувати найпопулярніші серверні мови програмування. На основі статті [2] складемо порівняльну таблицю.

Таблиця 1.3 Порівняльна таблиця серверних мов програмування

	PHP	Python	Node.js (JavaScript)
Продуктивність	Висока для веб-додатків	Середня для веб-додатків	Висока для асинхронних додатків

Простота вивчення та використання	Середня	Легка	Середня
Підтримка спільноти	Велика	Велика	Дуже велика
Масштабованість	Середня	Середня	Висока
Асинхронність	Підтримується обмежено	Підтримується окремими бібліотеками (asyncio)	Повна підтримка
Безпека	Висока	Висока	Висока
Популярність	Висока	Дуже висока	Дуже висока
Рівень різноманітності інструментів та бібліотек	Середня	Висока	Висока
Рівень підтримки ООП	Високий	Високий	Середній

В результаті порівняння мов програмування, остаточний вибір падає на мову програмування PHP, це обумовлено наступними причинами: Python хоч і є дуже потужним та універсальним, але його погана продуктивність та масштабованість у великих веб-додатках роблять його не найкращим кандидатом; Node.js орієнтований на асинхронні операції, що не є критичним для «Помічника викладача», до того ж, Node.js має набагато менш розвинену підтримку об'єктно-орієнтованого програмування в порівнянні з PHP.

1.2 Інтегроване середовище розробки:

Зважаючи на обрану мову програмування потрібно обрати відповідне середовище розробки, або IDE. Складемо порівняльну таблицю для деяких найпопулярніших IDE.

Таблиця 1.4 Порівняльна таблиця популярних IDE

Критерій	PHPStorm	Visual Studio Code	NetBeans
Продуктивність	Висока	Залежить від плагінів	Висока
Підтримка мов програмування	PHP, HTML, CSS, JavaScript, SQL	Практично всі	PHP, Java, HTML, CSS, JavaScript
Інструменти налагодження	PHP debugger, Xdebug	Вбудовані, Xdebug	Вбудовані
Плагіни та розширення	Велика кількість	Дуже велика кількість	Помірна кількість

Інтерфейс користувача	Інтуїтивний, багатий на функції	Налаштовуваний	Простий, функціональний
Підтримка баз даних	Вбудована	Забезпечується через плагіни	Вбудована
Ціна	Безкоштовно для студентів, \$199/рік	Безкоштовно	Безкоштовно
Документація та підтримка	Високоякісна документація, активна спільнота	Велика спільнота, багато ресурсів	Посередня документація

В підсумку проведеного аналізу, Visual Studio Code виглядає чудовим варіантом, заважаючи на його можливості кастомізації та велику кількість плагінів та активну спільноту. Про те PHP Storm є набагато більш професійною IDE і має набагато ширший функціонал, який вже налаштований і готовий до використання, а якщо до того ж врахувати той факт що дана IDE є безкоштовною для студентів, то PHP Storm є найкращим кандидатом.

1.3 База даних

Для розробки додатку було обрано СУБД MySQL, оскільки вона відзначається простотою використання, високою продуктивністю та вбудованими засобами забезпечення безпеки даних. До того ж MySQL легко масштабувати, що ідеально підходить для потреб інформаційної системи.

Також варто зазначити, що MySQL набагато краще підходить для розробки інформаційної системи «Помічник викладача» на відміну від своїх конкурентів, наприклад:

- на відміну від PostgreSQL, MySQL має набагато вищу швидкість при читанні даних, що дуже важливо для веб-додатків
- SQLite дуже добре підходить для невеликих додатків та систем, «Помічник викладача» не є таким, а отже MySQL підходить краще
- для «Помічника викладача» є важливим структурованість збережених даних, а отже MongoDB не є підходящим кандидатом, оскільки він є нереляційною базою даних

2. Вимоги до апаратних засобів

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		20

2.1 Веб сервер

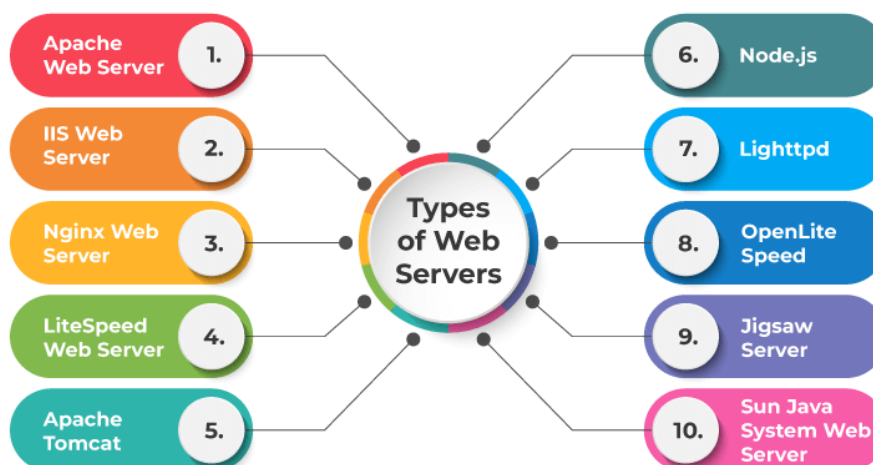


Рисунок 1.3 – Типи веб-серверів

Проаналізувавши статтю [10] було прийнято рішення використовувати Apache, оскільки його можна легко налаштувати під свої потреби завдяки його модульній структурі. Зважаючи на те, що він має відкритий вихідний код, є можливість додавати власні модулі до сервера. До того ж він дуже стабільний у порівнянні з іншими веб-серверами, і більшість адміністративних проблем на ньому можна легко вирішити.

2.2 Апаратні засоби сервера

Для безпосереднього процесу розробки, достатньо машини, яка матиме змогу запустити локальний сервер Apache. Для комфортного процесу розробки рекомендовано наступні характеристики апаратного засобу:

- Процесор: 6-ти ядерний, 12-ти поточний
- Оперативна пам'ять: 16 ГБ
- Диск: 128 ГБ вільного місця

Для розгортання додатку та його подальшого масштабування рекомендується використовувати такі хмарні сервіси як Azure, оскільки це забезпечить масштабованість, економічність (за рахунок схеми оплати pay-as-you-go, а також економії ресурсів на витрати на власне апаратне забезпечення). До того ж використання хмарних технологій забезпечить високу безпеку даних.

2.3 Апаратні засоби клієнтів

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		21

Оскільки інформаційна система буде представлена у вигляді веб-додатку, то вимоги до апаратних засобів клієнтів є мінімальними, й достатньо, щоб вони мали змогу запустити браузер.

Мінімальні системні вимоги:

- Процесор: 2 ядра, 2 потоки
- Оперативна пам'ять: 512 МБ

2.4 Безпекові заходи

- Необхідно передбачити захисні міри від таких методів зловмисників як SQL-ін'єкції та ін.
- Використання протоколу HTTPS
- Валідація усіх даних, які відправляються клієнтом на сервер
- Системи рівнів доступу

Висновки до першого розділу

В рамках першого розділу кваліфікаційної роботи було складено технічне завдання, в якому було описано такі пункти як:

- Назва системи
- Призначення системи
- Загальні вимоги до системи
- Вимоги до інтерфейсу
- Функціональні вимоги
- Нефункціональні вимоги
- Додаткові вимоги

Було проведено детальний аналіз існуючих архітектурних шаблонів та моделей, за підсумками якого було вирішено використовувати симбіоз клієнт-серверної архітектури та архітектурного шаблону MVC. Таке поєднання забезпечує чітку організацію коду, легкість у розробці, тестуванні та масштабуванні додатків.

Також було вибрано інструментарій реалізації та сформовано вимоги до апаратних засобів інформаційної системи. В межах аналізу інструментів розробки було обрано мови програмування та технології, які будуть використані

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		22

для розробки, а саме Bootstrap, jQuery і т.д для Frontend, та PHP для серверної частини. До того ж було проаналізовано популярні IDE, в результаті чого було прийнято рішення використовувати PhpStorm. Також в межах даного пункту було обрано СУБД, якою виявилась MySQL та веб-сервер, щобуде використано для розробки додатку, для роботи ІС вирішено використовувати Apache.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		23

РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Визначення варіантів використання системи

Проаналізувавши джерело [12] ми можемо зробити висновок, що правильний розподіл ролей у будь-якій інформаційній системі є запорукою її успішної роботи та безпеки. Визначення чітких ролей та повноважень для кожного користувача дозволяє забезпечити ефективне управління системою, підвищити її функціональність та допомагає захистити дані від несанкціонованого доступу. Кожна роль у системі має свій набір прав та обмежень, що дозволяє користувачам виконувати лише ті дії, які необхідні для виконання їхніх завдань. Це допомагає уникнути помилок та запобігає можливим зловмисним діям. Крім того, розподіл ролей спрощує процес управління системою, оскільки дозволяє адміністраторам легко контролювати доступ користувачів до різних функцій та даних. Це також полегшує процес навчання нових користувачів, оскільки вони можуть швидко зрозуміти, які дії вони можуть виконувати, а які ні.

У інформаційній системі "Помічник викладача" виділено чотири основні ролі: супер адміністратор, адміністратор, викладач та гість. Цей розподіл ролей обумовлений специфікою завдань, які повинна виконувати система, та необхідністю забезпечити належний рівень безпеки та функціональності.

Розглянемо детальніше кожен з ролей:

- Супер адміністратор - це користувач, який має найвищий рівень доступу до системи. Він відповідає за управління системою в цілому, включаючи налаштування, конфігурацію, резервне копіювання та відновлення даних. Супер адміністратор має повний доступ до всіх функцій та даних системи.
- Адміністратор - це користувач, який має доступ до адміністративної панелі системи та відповідає за управління викладачами, реєстрацію нових користувачів у системі та виконання CRUD операцій з даними в системі.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		24

- Викладач - це користувач, який використовує систему для виконання дій, пов'язаних з навчальним процесом, наприклад заповнення освітньої програми. Викладач має доступ до функцій, необхідних виключно для виконання своїх завдань.
- Гість - це користувач, який не має облікового запису в системі, але може отримати доступ до певних функцій та даних. Гість має змогу переглядати інформацію про факультети, кафедри, спеціальності, дисципліни, списки викладачів та іншу загальнодоступну інформацію.

Також в системі передбачений функціонал який має бути доступний всім користувачам, наприклад перегляд інформації про факультети, спеціальності і т.д.

Провівши дослідження на тему use-case діаграм [11], складемо use-case діаграму для кожної ролі в системі.



Рисунок 2.1 – Use Case діаграма користувача з роллю «Викладач»

Use-case діаграма зображена на рисунку 2.1 відображає функціональність системи для користувача, який є викладачем. Опишемо детальніше дану діаграму:

Головна сторінка – кожен викладач має доступ до Головної сторінки.

Перегляд списку факультетів - кожен викладач має змогу переглядати список всіх факультетів, де він може:

- Переглянути структуру факультету, тобто список кафедр факультету
- Переглянути спеціальності, представлені на факультеті
- Переглянути загальну інформацію про факультет

При перегляді структури факультету, кожен викладач має доступ до наступних дій:

- Перегляд інформації про кафедру
- Перегляд списку викладачів кафедри
- Перегляд списку освітніх програм, звідки в нього є змога перейти до певної освітньої програми, та почати редагувати компоненти цієї освітньої програми

Вхід до облікового запису – кожен викладач, маючи свій логін та пароль, може увійти до свого облікового запису. Якщо користувач успішно пройшов автентифікацію, то йому стають доступні наступні дії:

- Перегляд профілю, де відображена основна інформація про викладача
- Редагування профілю, де наприклад можна змінити пароль для входу в систему

Use Case діаграма користувача з роллю «Адміністратор» знаходиться у додатку А.

Якщо підсумувати, то «Адміністратор» має набагато більше дозволів для редагування інформації, і якщо коротко, то «Адміністратор» може редагувати все те, що не може редагувати «Викладач». А також «Адміністратор» може додавати новий користувачів до системи.

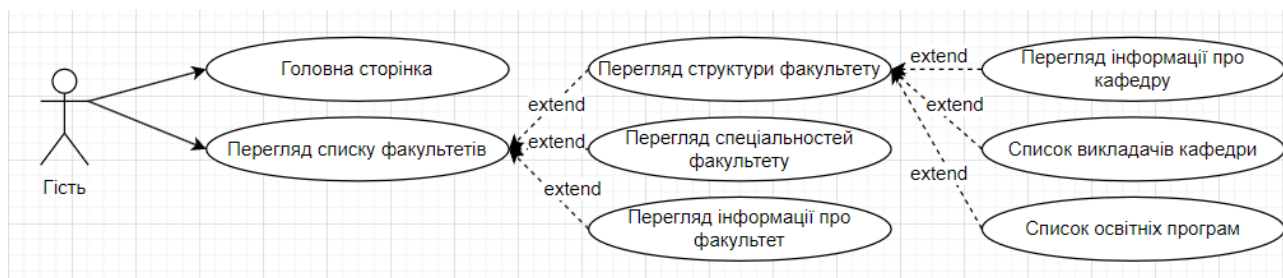


Рисунок 2.2 – Use Case діаграма користувача з роллю «Гість»

Як було вже зазначено раніше Гість - це користувач, який не має облікового запису в системі. Єдиний дозвіл який він має – перегляд публічної інформації.

2.2 Проектування архітектури та узагальненої структури системи

В технічному завданні до інформаційної системи проаналізувавши деякі джерела, а саме [4][8][9], було детально розглянуто існуючі типи архітектури. В підсумку було прийнято рішення, для розробки системи використовувати поєднання клієнт-серверної архітектури та шаблону MVC.

Відповідно до того, як влаштована архітектура MVC необхідно спочатку продумати з яких модулів буде складатись система.



Рисунок 2.3 – модулі системи «Помічник викладача»

Як видно з рисунку 2.3 в системі передбачено розробку дев'яти змістовних модулів, розглянемо кожен модуль детальніше:

- **Профіль:**
Модуль відповідає за відображення особистих даних користувачів. Тут можна редагувати свої дані такі як наприклад, пароль.
- **Освітні програми:**
Модуль для перегляду та управління освітніми програмами, де викладачі можуть керувати її компонентами (які будуть розглянуті пізніше), а адміністратори можуть додавати їх.
- **Викладачі:**
Умовний модуль, який відповідає за керування викладачами, інформацією про них, їх дозволами можливостями і т.д.
- **Адміністратори:**
Ще один умовний модуль, який аналогічний модулю Викладачі, про те присвячений адміністраторам системи.

- Факультети:

Модуль який містить інформацію про факультети університету. Адміністратори можуть додавати нові факультети, редагувати дані про існуючі, переглядати перелік факультетів і пов'язані з ними кафедри та спеціальності.

- Кафедри:

Схожий до модуля факультетів, але сфокусований на кафедрах. Додавання та редагування кафедр, перегляд інформації про кафедри та викладачів, які до них відносяться.

- Спеціальності:

Модуль спрямований на управління спеціальностями. Додавання нових спеціальностей, редагування даних про існуючі, перегляд інформації про спеціальності, пов'язані з кафедрами і факультетами.

- Адміністративна панель:

Центральний модуль для адміністраторів, де вони можуть здійснювати певні операції, такі як наприклад реєстрація нового користувача.

Сформувавши змістовні модулі, можемо перейти до наступного етапу проектування. Будь яка архітектура починається з ядра, тому спроектуємо ядро системи та його компоненти.

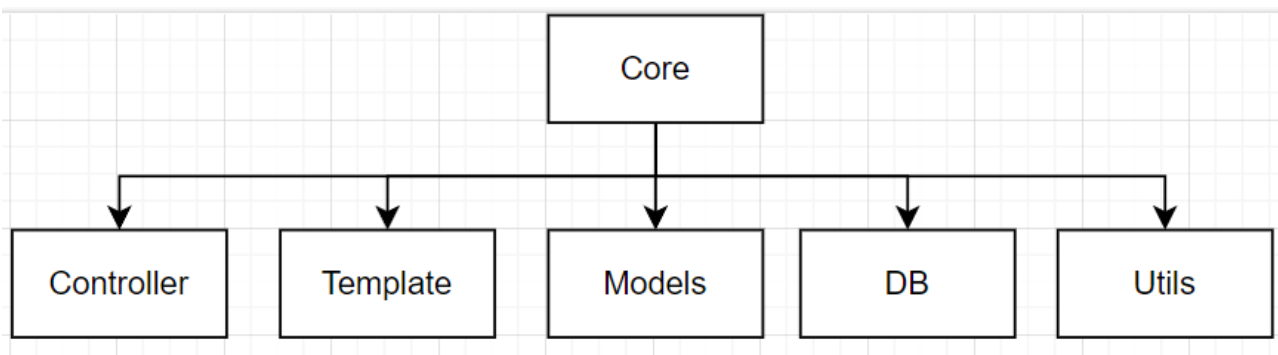


Рисунок 2.4 – Компоненти ядра системи

Детально розглянемо кожен компонент ядра системи:

- Controller – клас від якого буде наслідуватись кожен створений контролер системи, містить методи й поля які будуть необхідні для кожного контролера

- Models – аналогічно до класу Controller, про те для моделей системи
- Template – клас який відповідає за обробку View
- DB – клас який повністю відповідає за роботу з базою даних
- Utils – клас який містить допоміжні методи, який має на меті винести певні методи, щоб уникнути дублювання коду
- Core – основний клас, в якому відбувається ініціалізація системи та подальше її підтримання

Надалі відповідно до кожного спроектованого змістовного модулю, потрібно створити контролер та модель. Тобто потрібно створити наступні пари контролер-модель: Profile, EducationalProgram, Teachers, Administrators, Faculties, Departments, Specializations, AdminPanel.

Також варто додати спеціальний контролер/модуль Site, який буде відповідати за Головну сторінку, а також модулі Teachers та Administrators можна об'єднати в один модуль Users.

Таким чином ми повністю спроектували структуру додатку і можемо переходити до наступного етапу розробки.

2.3 Проектування структури та розробка бази даних системи

Проектування бази даних інформаційної системи є надзвичайно важливим кроком, неправильне проектування бази даних може призвести до критичних проблем, в той же час як правильне проектування принесе наступні переваги:

- Добре спроектована база даних дозволяє ефективно зберігати та отримувати дані, що призводить до швидшої роботи веб-додатку
- Правильне проектування допомагає уникнути неточних чи «пошкоджених» даних, забезпечуючи їх цілісність та відповідність
- Добре спроектована база даних може бути легко масштабована
- Правильне проектування допомагає забезпечити повноцінну безпеку даних

Враховуючи важливість цього процесу та проаналізувавши джерело [13] можемо приступити до безпосереднього проектування БД, попередньо

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		29

врахувавши структуру освітньої програми, яка є чітко визначеною, та приклад якої було взято з джерела [16]

Загальна структура Баз даних системи знаходиться у Додатку Б.

Розглянемо зв'язки, якими пов'язані таблиці у базі даних: administrators:

Не мають зв'язків з іншими таблицями

departments:

- Пов'язана з таблицею faculties через поле faculty_id, тип зв'язку «один до багатьох», оскільки один факультет може бути пов'язаний з багатьма кафедрами, в той же час як одна кафедра може бути пов'язана лише з одним факультетом

teachers:

- Має поле department_id, яке пов'язане з таблицею departments, тип зв'язку «один до багатьох», оскільки одна кафедра може бути пов'язана з багатьма викладачами, в той же час як один викладач може бути пов'язаний лише з однією кафедрою

specializations:

- Пов'язана з таблицею faculties через поле faculty_id, тип зв'язку «один до багатьох», оскільки один факультет може бути пов'язаний з багатьма спеціальностями, в той же час як одна спеціальність може бути пов'язана лише з одним факультетом.

educational_programs:

- Має поле specialization_id, яке пов'язане з таблицею specializations, тип зв'язку «один до багатьох», оскільки одна спеціальність може бути пов'язана з багатьма освітніми програмами, в той же час як одна освітня програма може бути пов'язана лише з однією спеціальністю.

Таблиці educational_program_general_info, educational_program_objective, educational_program_characteristic, educational_program_academic_mobility, educational_program_suitability, educational_program_components, educational_program_competences, educational_program_teaching_assessment,

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		30

educational_program_resource_support, educational_program_learning_outcomes, educational_program_attestation, кожна з них має зв'язок з таблицею educational_programs, тому що вони виступають компонентами освітньої програми. Такі компоненти як educational_program_components, educational_program_competences, educational_program_learning_outcomes мають зв'язок один до багатьох, тому що одна освітня програма може мати декілька компонентів, компетенцій та наслідків навчання відповідно. Всі інші компоненти мають з таблицею educational_programs зв'язок один до одного.

Тепер розглянемо поля кожної таблиці окремо:

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐	1 id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐	2 first_name	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐	3 last_name	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐	4 patronymic	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐	5 position	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐	6 scientific_degree	varchar(255)	utf8mb4_unicode_ci		Ні	Немає	Науковий ступінь	
☐	7 academic_title	varchar(255)	utf8mb4_unicode_ci		Ні	Немає	Вчене звання	
☐	8 department_id 🔑	int(11)			Ні	Немає		
☐	9 level_of_access	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐	10 login	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐	11 password	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐	12 email	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐	13 photo	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		

Рисунок 2.5 – Структура таблиці «teachers»

Таблиця «teachers», відповідає за зберігання інформації про викладачів навчального закладу, та містить наступні поля:

- id (int): Унікальний ідентифікатор вчителя
- first_name (varchar(50)): Ім'я вчителя
- last_name (varchar(50)): Прізвище вчителя
- patronymic (varchar(50)): По батькові вчителя
- position (varchar(255)): Посада вчителя

- scientific_degree (varchar(255)): Науковий ступінь вчителя
- academic_title (varchar(255)): Вчене звання
- department_id (int): Ідентифікатор кафедри, до якої належить вчитель
- login (varchar(255)): Логін для входу в систему
- password (varchar(255)): Пароль для входу в систему
- email (varchar(255)): Електронна адреса вчителя
- photo (varchar(255)): Назва файлу, що відповідає за фото викладача

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	name	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	info	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	faculty_id 🔑	int(11)			Ні	Немає		

Рисунок 2.6 – Структура таблиці «departments»

Таблиця «departments» відповідає за збереження інформації про кафедри факультетів, та містить наступні поля:

- id (int): Унікальний ідентифікатор кафедри
- name (varchar(255)): Назва кафедри
- info (text): Інформація про кафедри
- faculty_id (int): Ідентифікатор факультету, до якого належить кафедра

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	name	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	info	text	utf8mb4_unicode_ci		Ні	Немає		

Рисунок 2.7 – Структура таблиці «faculties»

Таблиця «faculties» відповідає за збереження інформації про факультети навчального закладу:

- id (int): Унікальний ідентифікатор факультету
- name (varchar(255)): Назва факультету
- info (text): Інформація про факультет

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	name	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	code	int(11)			Ні	Немає		
☐ 4	info	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	faculty_id 🔑	int(11)			Ні	Немає		

Рисунок 2.8 – Структура таблиці «specializations»

Таблиця «specializations» відповідає за збереження інформації про спеціальності факультетів:

- id (int): Унікальний ідентифікатор спеціалізації
- name (varchar(255)): Назва спеціалізації
- code (int): Код спеціалізації
- info (text): Інформація про спеціалізацію
- faculty_id (int): Ідентифікатор факультету, до якого належить спеціалізація

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	name	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	foreword	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	work_group_members	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	specialization_id 🔑	int(11)			Ні	Немає		

Рисунок 2.9 – Структура таблиці «educational_programs»

Таблиця «educational_programs» відповідає за збереження інформації про освітні програми факультетів:

- id (int): Унікальний ідентифікатор освітньої програми
- name (varchar(255)): Назва освітньої програми
- foreword (text): Передмова до освітньої програми
- work_group_members (text): Інформація про робочу групу, що складала освітню програму

- specialization_id (int): Ідентифікатор спеціалізації, до якої належить освітня програма

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	first_name	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	last_name	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	patronymic	varchar(50)	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	position	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 6	level_of_access	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 7	login	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 8	password	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		
☐ 9	email	varchar(255)	utf8mb4_unicode_ci		Ні	Немає		

Рисунок 2.10 – Структура таблиці «administrators»

Таблиця «administrators» відповідає за збереження інформації про адміністраторів навчального закладу:

- id (int): Унікальний ідентифікатор адміністратора
- first_name (varchar(50)): Ім'я адміністратора
- last_name (varchar(50)): Прізвище адміністратора
- patronymic (varchar(50)): По батькові адміністратора
- position (varchar(255)): Посада адміністратора
- level_of_access (varchar(255)): Рівень доступу адміністратора
- login (varchar(255)): Логін для входу в систему
- password (varchar(255)): Пароль для входу в систему
- email (varchar(255)): Електронна адреса адміністратора

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	national_credit_mobility	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	international_credit_mobility	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	foreign_students_education	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.11 – Структура таблиці «educational_program_academic_mobility»

Таблиця «educational_program_academic_mobility» відповідає за збереження інформації про академічну мобільність освітніх програм:

- id (int): Унікальний ідентифікатор запису
- national_credit_mobility (text): Інформація про національну кредитну мобільність
- international_credit_mobility (text): Інформація про міжнародну кредитну мобільність
- foreign_students_education (text): Інформація про освіту іноземних студентів
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	attestation_description	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.12 – Структура таблиці «educational_program_attestation»

Таблиця «educational_program_attestation» відповідає за збереження інформації про форму атестації освітніх програм:

- id (int): Унікальний ідентифікатор запису
- attestation_description (text): Опис форми атестації
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	subject_area	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	orientation	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	main_focus	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	features	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 6	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.13 – Структура таблиці «educational_program_characteristic»

Таблиця «educational_program_attestation» відповідає за збереження інформації про характеристику освітніх програм:

- id (int): Унікальний ідентифікатор запису
- subject_area (text): Опис предметної області освітньої програми
- orientation (text): Опис орієнтації освітньої програми
- main_focus (text): Опис основного фокусу освітньої програми
- features (text): Опис особливостей освітньої програми
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	competence_description	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	competence_type	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	educational_program_id	int(11)			Ні	Немає		

Рисунок 2.14 – Структура таблиці «educational_program_competences»

Таблиця «educational_program_competences» відповідає за збереження інформації про компетенції освітніх програм:

- id (int): Унікальний ідентифікатор запису
- competence_description (text): Опис компетенції
- competence_type (text): Тип компетенції
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐	1 id	int(11)			Ні	Немає		AUTO_INCREMENT
☐	2 component_type	text	utf8mb4_unicode_ci		Ні	Немає		
☐	3 component_block	text	utf8mb4_unicode_ci		Ні	Немає		
☐	4 component_name	text	utf8mb4_unicode_ci		Ні	Немає		
☐	5 credit_count	int(11)			Ні	Немає		
☐	6 final_control_form	text	utf8mb4_unicode_ci		Ні	Немає		
☐	7 involved_semesters	text	utf8mb4_unicode_ci		Ні	Немає		
☐	8 general_competences	text	utf8mb4_unicode_ci		Ні	Немає		
☐	9 special_competences	text	utf8mb4_unicode_ci		Ні	Немає		
☐	10 learning_outcomes	text	utf8mb4_unicode_ci		Ні	Немає		
☐	11 educational_program_id	int(11)			Ні	Немає		

Рисунок 2.15 – Структура таблиці «educational_program_components»

Таблиця «educational_program_components» відповідає за збереження інформації про компоненти освітніх програм:

- id (int): Унікальний ідентифікатор запису
- component_type (text): Тип компонента
- component_block (text): Блок компонента (у випадку якщо дисципліна вибіркова)
- component_name (text): Назва компонента
- credit_count (int): Кількість кредитів
- final_control_form (text): Форма підсумкового контролю
- involved_semesters (text): Семестри, в яких включено компонент
- general_competences (text): Загальні компетенції
- special_competences (text): Спеціальні компетенції
- learning_outcomes (text): Результати навчання
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

☐	1	id 🔗		int(11)		Hi	Немає	AUTO_INCREMENT
☐	2	full_name_of_institute_and_structure	text	utf8mb4_unicode_ci		Hi	Немає	
☐	3	degree_and_title	text	utf8mb4_unicode_ci		Hi	Немає	
☐	4	official_name	text	utf8mb4_unicode_ci		Hi	Немає	
☐	5	diploma_type	text	utf8mb4_unicode_ci		Hi	Немає	
☐	6	credits_count		int(11)		Hi	Немає	
☐	7	term_of_study	text	utf8mb4_unicode_ci		Hi	Немає	
☐	8	accreditation	text	utf8mb4_unicode_ci		Hi	Немає	
☐	9	cycle_and_level	text	utf8mb4_unicode_ci		Hi	Немає	
☐	10	prerequisites	text	utf8mb4_unicode_ci		Hi	Немає	
☐	11	languages	text	utf8mb4_unicode_ci		Hi	Немає	
☐	12	duration	text	utf8mb4_unicode_ci		Hi	Немає	
☐	13	internet_address	text	utf8mb4_unicode_ci		Hi	Немає	
☐	14	educational_program_id 🔗		int(11)		Hi	Немає	

Рисунок 2.15 – Структура таблиці «educational_program_general_info»

Таблиця «educational_program_general_info» відповідає за збереження загальної інформації про освітні програми:

- **id (int):** Унікальний ідентифікатор запису
- **full_name_of_institute_and_structure (text):** Повна назва інституту та структури підрозділу
- **degree_and_title (text):** Ступінь та назва кваліфікації
- **official_name (text):** Офіційна назва
- **diploma_type (text):** Тип диплому
- **credits_count (int):** Кількість кредитів
- **term_of_study (text):** Термін навчання
- **accreditation (text):** Акредитація (її наявність)
- **cycle_and_level (text):** Цикл і рівень
- **prerequisites (text):** Передумови
- **languages (text):** Мови навчання
- **duration (text):** Термін дії
- **internet_address (text):** Інтернет-адреса

- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	outcome_description	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.16 – Структура таблиці «educational_program_learning_outcomes»

Таблиця «educational_program_learning_outcomes» відповідає за збереження інформації про результати навчання освітніх програм:

- id (int): Унікальний ідентифікатор запису
- outcome_description (text): Опис результату
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	objective	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.17 – Структура таблиці «educational_program_objective»

Таблиця «educational_program_learning_objective» відповідає за збереження інформації про мету освітньої програми:

- id (int): Унікальний ідентифікатор запису
- objective (text): Опис мети
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	stuff_support	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	logistical_support	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	methodological_support	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 5	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.18 – Структура таблиці «educational_program_resource_support»

Таблиця «educational_program_resource_support» відповідає за збереження інформації про ресурсну підтримку освітньої програми:

- id (int): Унікальний ідентифікатор запису
- stuff_support (text): Опис кадрової підтримки
- logistical_support (text): Опис матеріально-технічної підтримки
- methodological_support (text): Опис навчально-методичної підтримки
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	employment_suitability	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	further_education	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.19 – Структура таблиці «educational_program_suitability»

Таблиця «educational_program_suitability» відповідає за збереження інформації про придатність освітньої програми:

- id (int): Унікальний ідентифікатор запису
- employment_suitability (text): Опис придатності для працевлаштування
- further_edification (text): Опис можливостей для подальшого навчання
- educational_program_id (int): Унікальний ідентифікатор освітньої програми

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT
☐ 2	teaching_and_learning	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 3	assessment	text	utf8mb4_unicode_ci		Ні	Немає		
☐ 4	educational_program_id 🔑	int(11)			Ні	Немає		

Рисунок 2.20 – Структура таблиці «educational_program_teaching_assessment»

Таблиця «educational_program_teaching_assessment» відповідає за збереження інформації про навчання та оцінювання освітньої програми:

- id (int): Унікальний ідентифікатор запису
- teaching_and_learning (text): Опис процесу навчання та викладання
- assessment (text): Опис процесу оцінювання

- `educational_program_id (int)`: Унікальний ідентифікатор освітньої програми

2.4 Розробка основних функціональних алгоритмів системи

Після проектування структури, архітектури та бази даних системи, потрібно переходити до наступного етапу розробки, а саме розробка алгоритмів, за якими буде функціонувати система. Про те перед цим варто розглянути які паттерни проектування будуть застосовані при розробці системи, адже паттерни проектування відіграють величезну роль в тому, як себе поводить система.

Перш за все буде використано паттерн Singleton. За джерелом [14], Singleton, або Одинак - це породжувальний патерн проектування, який гарантує, що клас має лише один екземпляр, та надає глобальний доступ до нього. Особливістю патерну є те, що він приховує від клієнтів всі способи створення нового об'єкта, окрім спеціального методу який або створює об'єкт, або віддає існуючий об'єкт, якщо він вже був створений. У системі «Помічник викладача» цей патерн використовуватиметься для забезпечення єдиного екземпляра ядра системи. Структура патерну зображена на рисунку 2.17.

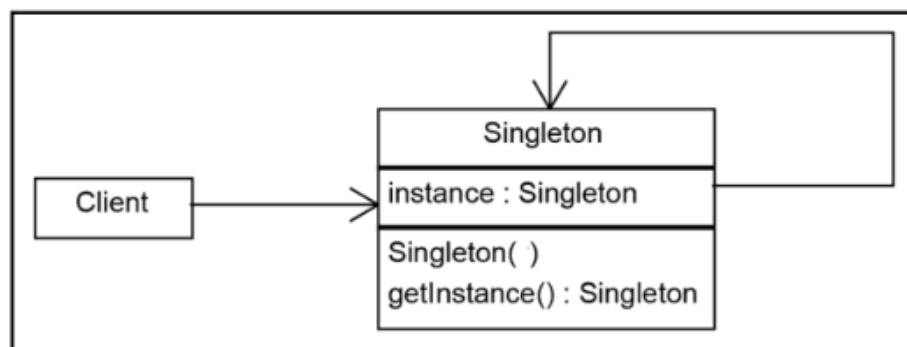


Рисунок 2.21 Структура патерну Singleton

Ще одним патерном проектування який буде використаний при розробці системи, є патерн Builder, або Будівельник. Відповідно до джерела [15] будівельник — це породжувальний патерн проектування, що дає змогу створювати складні об'єкти крок за кроком. Особливістю патерну є те, що він дозволяє збирати об'єкти покроково, викликаючи тільки ті кроки, які потрібні. Такий підхід може бути використаний майже повсюди, наприклад при створенні

освітньої програми, яка, як ми вже з’ясували, складається з дуже багатьох компонентів. Структура патерну зображена на рисунку 2.18.

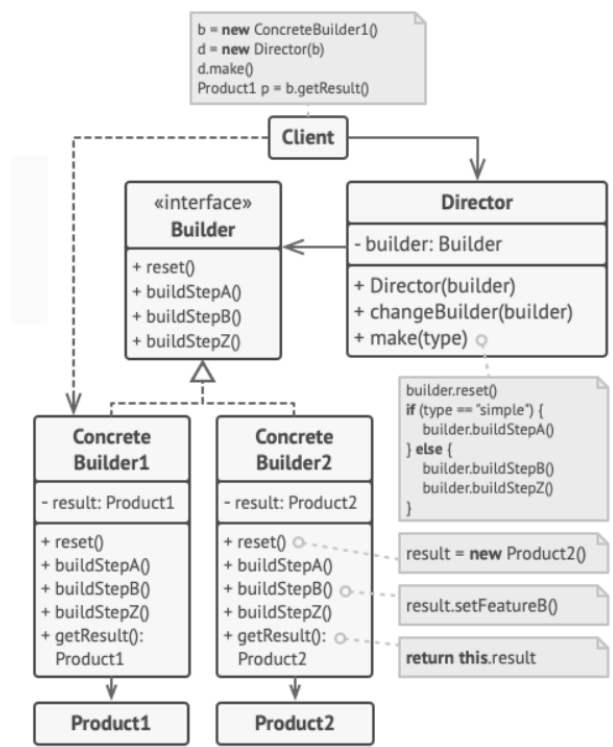


Рисунок 2.22 Структура патерну Builder

Окрім патернів проектування система діє за численної кількості маленьких алгоритмів, кожен з яких виконує ту чи іншу задачу.

Висновки до другого розділу

В рамках другого розділу кваліфікаційної роботи було визначено варіанти використання системи, спроектовано архітектуру та узагальнену структуру, спроектовано та розроблену базу даних, а також розроблено деякі функціональні алгоритми.

В наслідок визначення варіантів використання системи в системі було чітко виділено чотири ролі: Суперадмін, Адміністратор, Викладач, Гість. Кожен з них має свої привілеї та відповідні права доступу.

В наслідок проектування архітектури та узагальненої структури, було визначено основні змістовні модулі системи, кількість яких дорівнює семи. Також в межах цього підпункту було спроектовано ядро системи, та визначено які контролери та моделі знадобляться для розробки системи.

В наслідок виконання третього підпункту було повністю розроблено базу даних системи, яка в підсумку налічує 17 таблиць, які містять усі необхідні поля для повноцінної роботи системи.

В межах четвертого підрозділу, було визначено деякі патерни проектування, які будуть використані під час розробки системи.

Після виконання зазначених підпунктів, можна приступати до безпосередньої розробки веб-додатку.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		43

РОЗДІЛ 3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ, ОГЛЯД ІНТЕРФЕЙСУ ТА ЇЇ ТЕСТУВАННЯ

3.1 Розробка інформаційної системи

Розробка будь-якої інформаційної системи розпочинається з планування її файлової структури.

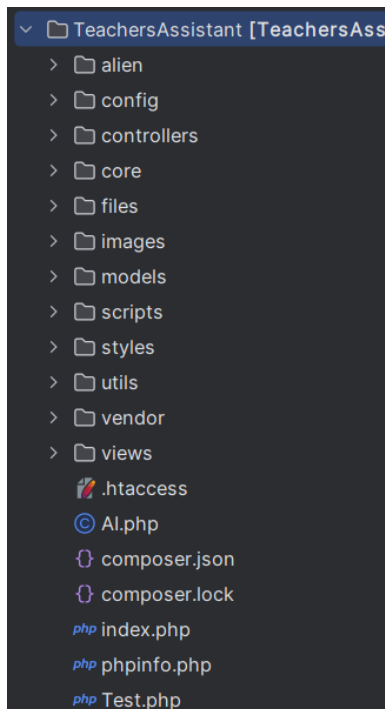


Рисунок 3.1 Файлова структура розробленої системи

Коротко опишемо за що відповідає кожен файл/папка у системі:

Папки

1. alien: Містить сторонні бібліотеки або пакети, які не входять до основного проекту. Таким наприклад є sk-editor, який використовується для «просунутих» полів введення
2. config: Тут знаходяться файли конфігурації проекту, які містять налаштування бази даних, параметри середовища тощо.
3. controllers: Містить файли контролерів, які обробляють HTTP-запити та керують взаємодією між моделями та видами, як і планувалось раніше, їх кількість 9.
4. core: Містить основні класи та функції, які використовуються по всьому проекту.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		44

5. files: Використовується для зберігання завантажених файлів або інших даних, що не є частиною коду. Наприклад фото викладачів
6. images: Містить статичні зображення, які використовуються в проекті.
7. models: Містить моделі, які представляють дані та логіку роботи з базою даних, в проекті реалізовано 7 моделей.
8. scripts: Містить скрипти на JavaScript, які використовуються на стороні клієнта або сервера.
9. styles: Містить файли CSS для стилізації веб-сторінок.
10. utils: Тут знаходяться утилітарні функції та класи, які використовуються по всьому проекту.
11. vendor: Містить сторонні бібліотеки, які встановлені через Composer (менеджер пакетів для PHP).
12. views: Містить шаблони представлення, які відповідають за відображення даних на веб-сторінках. На кожен контролер своя кількість представлень, для AdminPanel їх 3, для departments їх 4, для educationalProgram їх 28, для faculties їх 4, для layout (що містить файли які присутні на всіх сторінках, наприклад хедер та футер) їх 2, для profile їх 4, для site (головна сторінка), їх 1, та ще по одному для specializations та users

Файли

1. .htaccess: Файл конфігурації веб-сервера Apache. Використовується для налаштування перезапису URL, захисту директорій тощо.
2. AI.php: Файл, що містить логіку для роботи з штучним інтелектом, дозволяє робити запити до мовної моделі через підключене API.
3. composer.json: Файл конфігурації для Composer, який визначає залежності проекту.
4. composer.lock: Файл, що створюється Composer і містить зафіксовані версії всіх залежностей проекту.
5. index.php: Головний вхідний файл для веб-застосунку, який ініціалізує додаток та обробляє вхідні HTTP-запити.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		45

6. `phpinfo.php`: Файл для виведення інформації про конфігурацію PHP на сервері (викликає функцію `phpinfo()`).
7. `Test.php`: Містить деякі тести для перевірки функціональності коду проекту.

Наступним по важливості пунктом у виконанні, на мою думку є реалізація класу, який відповідає за взаємодію з БД. Взаємодія з БД реалізована через PDO [17] (PHP Data Objects) - це розширення для PHP, яке забезпечує інтерфейс для доступу до баз даних за допомогою уніфікованого об'єктно-орієнтованого підходу. Було прийнято рішення використовувати саме його, оскільки це один із рекомендованих способів роботи з базами даних у PHP який відомий завдяки своїй гнучкості, безпеці та простоті використання. Ініціалізація `pdo` відбувається наступним чином:

```
public function __construct($server, $login, $password, $database){
    $this->pdo = new
    \PDO("mysql:host={$server};dbname={$database};charset=UTF8", $login, $password);
}
```

Окрім цього, даний клас забезпечує простий інтерфейс для виконання основних CRUD (Create, Read, Update, Delete) операцій з базою даних у PHP.

Також для відслідковування помилок, для кожної операції було використано `try...catch` блок, який у випадку помилки чітко виводив код помилки та повідомлення помилки. Наприклад один з таких блоків виглядає наступним чином:

```
try {
    $sth = $this->pdo->prepare($sql);

    $sth->execute($whereValues);

    $errorInfo = $sth->errorInfo();
    if ($errorInfo[0] != '00000') {
        throw new Exception("SQLSTATE: " . $errorInfo[0] . " - Error Code: " .
            $errorInfo[1] . " - Message: " . $errorInfo[2]);
    }

    return $sth->fetchAll();
} catch (Exception $e) {
    echo "Error: " . $e->getMessage();
    return false;
}
```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		46

```

public function actionView(){...}
no usages
public function actionGeneralInfoForm(){...}
no usages
public function actionGeneralInfoFormEdit(){...}
no usages
public function actionGeneralInfoFormDelete(){...}

```

Рисунок 3.2 Приклад роботи контролеру

На рисунку 3.2 зображено приклад роботи контролеру. В наслідок взаємодії користувача з системою, він переходить на ті чи інші сторінки, url цих сторінок контролюється саме контролерами, і якщо користувач захоче зманіпулювати url-адресою та введе адресу не існуючу, тобто поза межами дії контролерів, то отримає помилку.

Кожен action по своєму унікальний, та в процесі викликає різні методи зі своєї відповідної моделі, та сусідніх моделей. Наприклад в контролері educationalProgram окрім методів однойменної моделі, також застосовується модель specialization. Якщо знову звернутись до БД, то ми побачимо що ці два компоненти пов'язані між собою, а тому не дивно що вони інколи можуть використовувати методи один одного.

Не дивлячись на свою унікальність, кожен action наприкінці повинен віддати на вихід відповідне view. Робиться це наступним чином:

```

return $this->render('generalInfoForm', ['model' => $model], [
    'MainTitle' => $mainTitle,
    'PageTitle' => $title
]);

```

Варто зазначити, що метод render (який прописаний у класі Template, який разом з іншими класами формує ядро системи) може викликатися необов'язково наприкінці action. В процесі обробки даних, в залежності від дій користувача, в межах одного action може виникнути потреба віддавати клієнту різні view.

Взагалі, в методі render ми можемо спостерігати наявність трьох параметрів: 1) Назва view, яку потрібно передати на сервер; 2) Локальні параметри, які будуть використані в межах view. Зазвичай у системі, під час роботи контролера всі необхідні параметри збирались в єдиний model, що

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		47

утворювало асоціативний масив, яким дуже зручно користуватись, та підставляти потрібні дані у view; 3) Третій параметр – глобальні параметри, які зазвичай використовуються для глобальних змінних, як от наприклад Заголовок веб-сторінки, чи заголовок view, який підставляється таким чином:

```
<div class="row">
  <div class="card d-flex justify-content-center align-items-center">
    <?= $PageTitle ?>
  </div>
</div>
```

Де \$PageTitle і є заголовок view.

І ось саме за таким принципом і працює вся система. Користувач взаємодіє з системою, в результаті чого тригерить різні action в контролерах, які в свою чергу взаємодіють з моделями, які в свою чергу містять методи для роботи з БД чи методи для обробки даних, і в результаті цієї взаємодії, клієнту повертається відповідне view.

Отож, можемо перейти до безпосереднього огляду інтерфейсу інформаційної системи.

3.2 Огляд інтерфейсу інформаційної системи

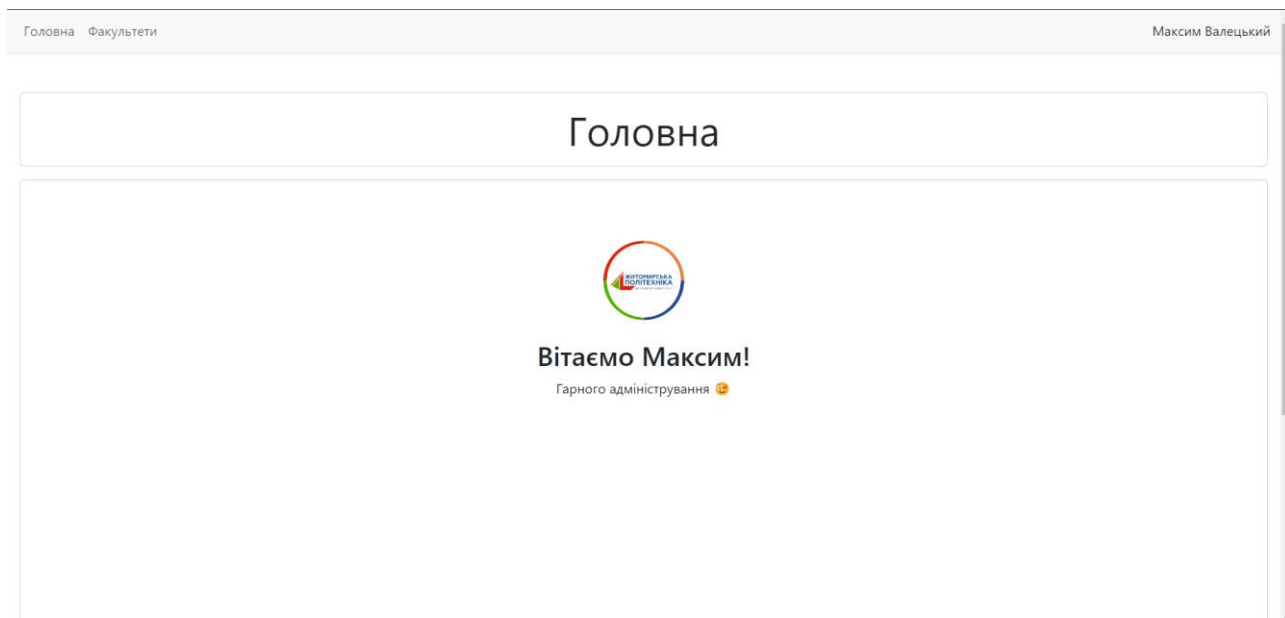


Рисунок 3.3 Головна сторінка

Оскільки на разі у системі не реалізовано дуже багато функціоналу, то на Головну сторінку мало чого можна винести. У системі не реалізований наприклад модуль Новин, який би дозволив винести на Головну останні новини.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		48

Про те, система є доволі масштабованою, і тому в майбутньому з введенням нових функцій, головна сторінка теж буде наповнюватись.

Зараз кожного користувача система зустрічає по різному. Вибір повідомлення напряму залежить від ролі користувача:

```
<div class="welcome-message">
  <?php if (isset($user['level_of_access'])) : ?>
    <div id="admin-message">
      <h2>Вітаємо <span id="admin-
name"><?=$user['first_name']?></span>!</h2>
      <p>Гарного адміністрування 😊</p>
    </div>
    <?php elseif (isset($user['scientific_degree'])) : ?>
      <div id="teacher-message">
        <h2>Вітаємо!</h2>
        <p>Чудового вам дня та гарного викладання <span id="teacher-
name"><?=$user['first_name']?></span> 😊</p>
      </div>
      <?php else : ?>
        <div id="unlogged-message">
          <h2>Вітаємо!</h2>
          <p>Тут ви зможете дізнатись про те, як влаштований університет
"Житомирська політехніка",
          детально дізнатись про структуру університету, викладачів на
кафедрах та детальну інформацію
          про освітні програми, які надає можливість вивчати університет.
Тож мерщій досліджувати
          😊</p>
        </div>
        <?php endif; ?>
      </div>
```

В даному фрагменті коду продемонстровано як система перевіряє роль користувача. В системі передбачено, що у викладачів та адміністраторів є свої унікальні поля, і саме тому система може користуватись цим.

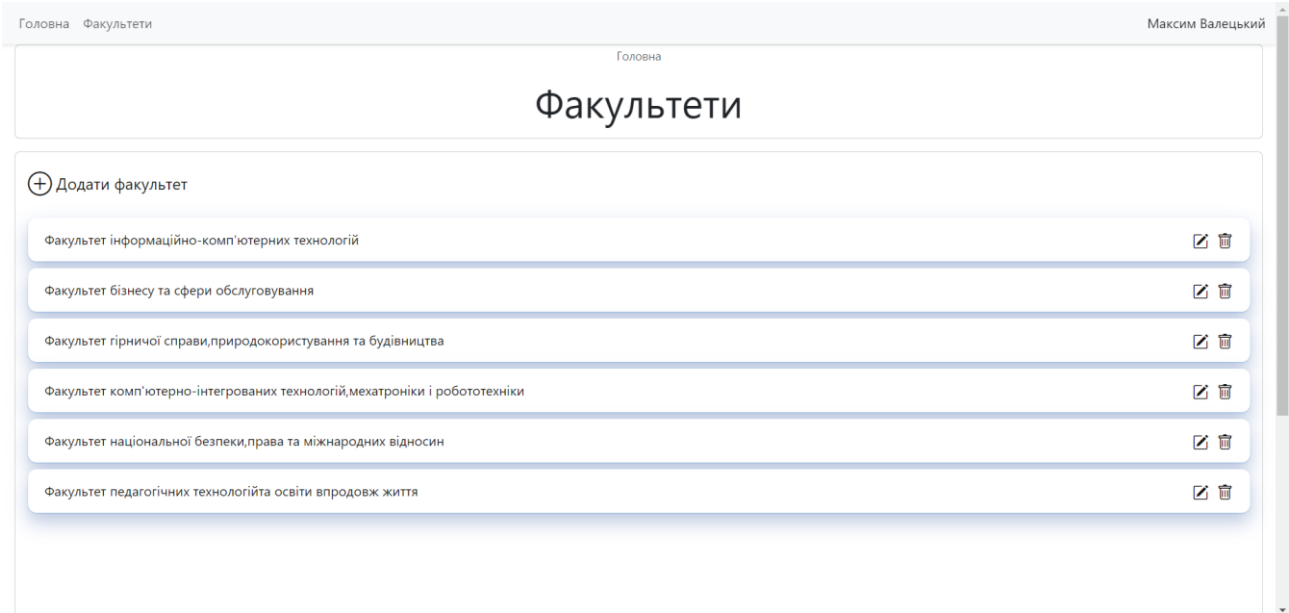


Рисунок 3.4 Сторінка факультетів

Перейшовши на сторінку факультетів, будь-який користувач має змогу переглянути список факультетів у системі. Кожен факультет стилізований за допомогою CSS. Варто зазначити, що подібних елементів «списків» зустрічається дуже велика кількість, наприклад список факультетів, список кафедр, список кометентностей і т.д. Кожен з таких «списків» формується за допомогою `for` чи `foreach` (в залежності від потреби), в якому ітерується певний ключ `$model`, як вже було зазначено раніше.

Якщо користувач є адміністратором, то окрім перегляду, йому доступні й інші CRUD операції з факультетами, зовнішній вигляд кожної з відповідних форм, зображені на рисунках 3.5, 3.6, 3.7.

Рисунок 3.5 Форма додавання факультету

Рисунок 3.6 Форма редагування факультету

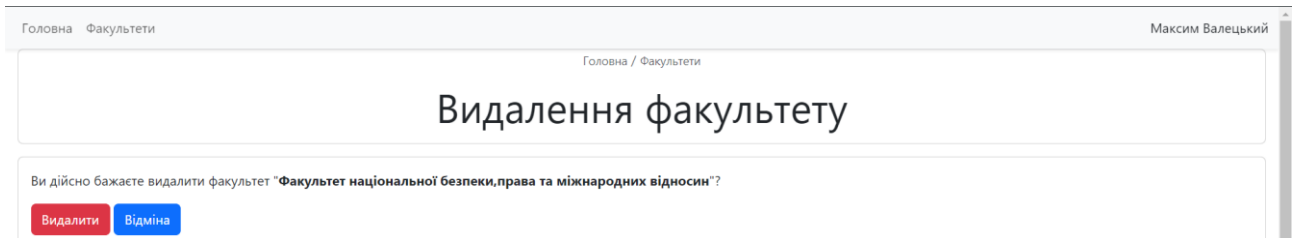


Рисунок 3.7 Форма видалення факультету

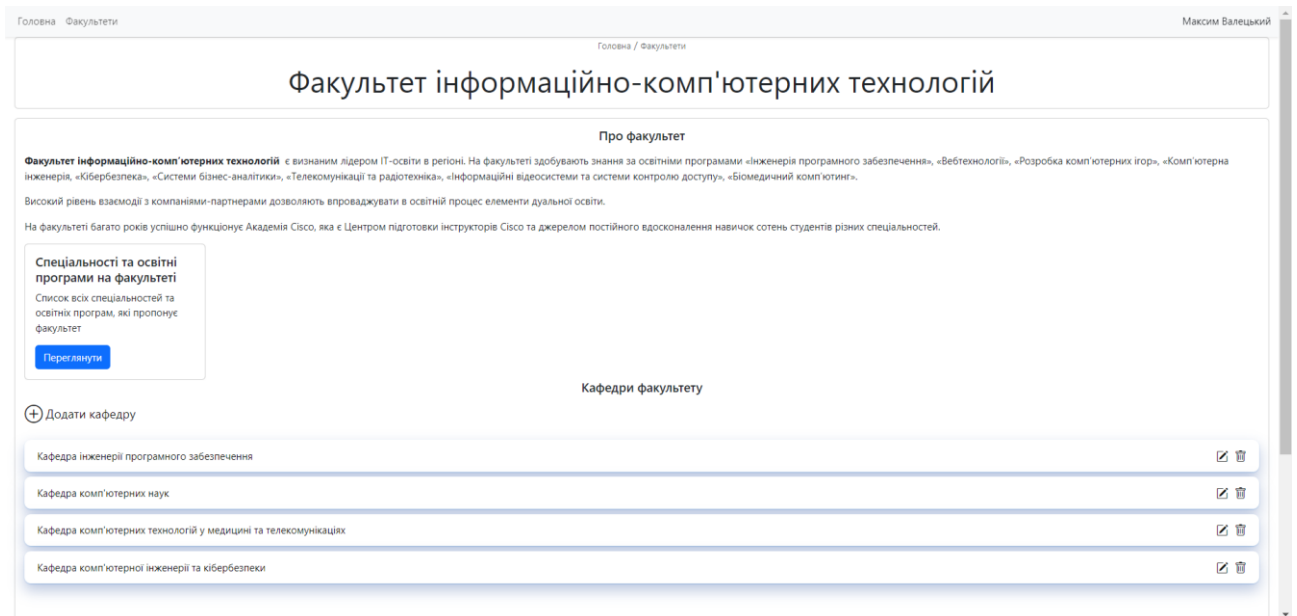


Рисунок 3.8 Сторінка факультету (масштаб зменшено)

Обравши один з факультетів, будь-який користувач може перейти на сторінку факультету й детальніше ознайомитись з інформацією про той чи інший факультет. Окрім основної інформації, трохи нижче, користувач має змогу перейти до сторінки спеціальностей та освітніх програм, які пропонує факультет. Також, секція, де знаходиться перехід до вказаної сторінки, передбачає розширення опцій та додавання інших факультетських пунктів. Наприклад з майбутнім введенням модулю Дисципліни, туди можна внести можливість переглядати Загальнофакультетські дисципліни.

Окрім вище згаданого функціоналу, у будь якого користувача є можливість переглянути структуру факультету, тобто які кафедри утворюють даний факультет. Тут все аналогічно факультетам, які були описані раніше. Форми для виконання CRUD операцій виглядають ідентично тим що й у Факультетів, про те там по іншому названі поля.

Ось така вкладеність, мається на увазі можливість відображення кафедр лише даного факультету реалізовано за допомогою зовнішніх ключів у БД.

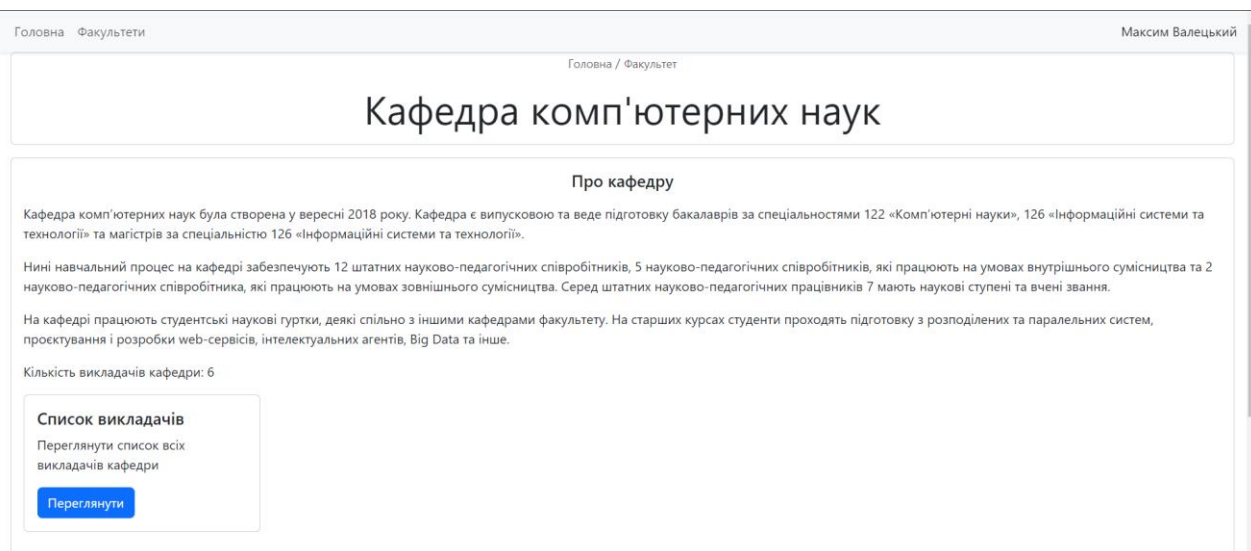


Рисунок 3.9 Сторінка кафедри

При переході на сторінку кафедри, будь-який користувач має змогу переглянути детальну інформацію про кафедру, а також кількість викладачів кафедри. На разі, з додаткового функціоналу є лише можливість переглядати список викладачів кафедри, але в майбутньому планується значне розширення даної секції, шляхом додавання інформації про завідуючого/завідуючої кафедри, інформації про деканат, та багато ін.

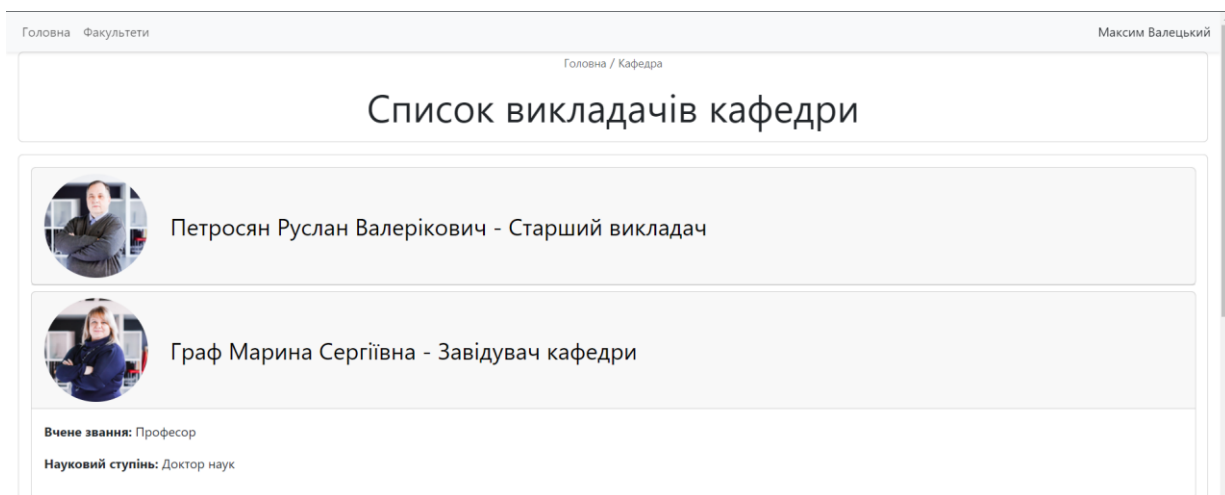


Рисунок 3.10 Список всіх викладачів кафедри

При переході на сторінку зі всіма викладачами, користувачі можуть переглянути основну інформацію про кожного викладача, а саме: їх посаду, вчене звання, науковий ступінь. В даному пункті також планується розширення,

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		52

шляхом додавання шкали навантаженості викладача, яка буде заповнюватись в залежності від кількості призначених викладачу дисциплін.

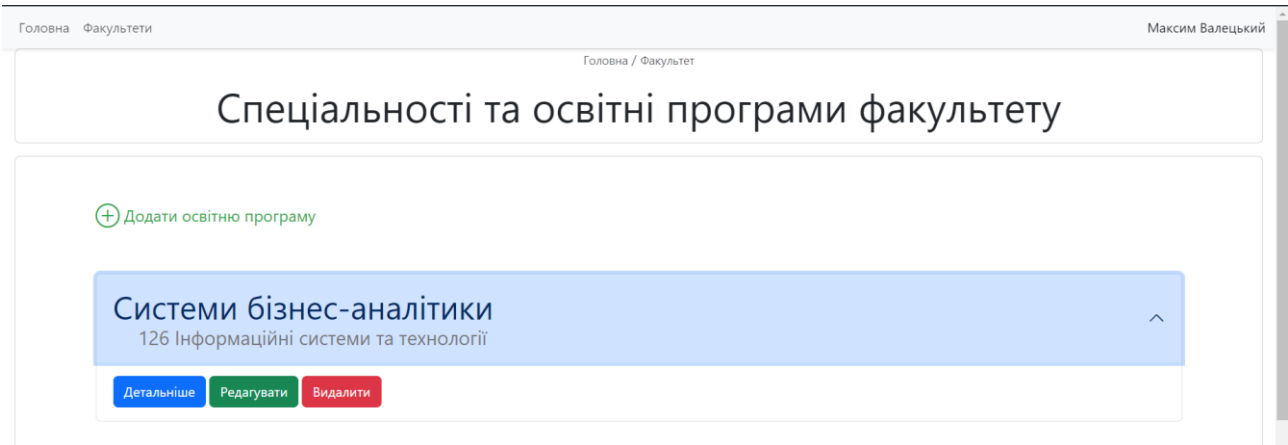


Рисунок 3.11 Сторінка спеціальностей та освітніх програм

Якщо повернутись на сторінку факультетів, то ми матимемо змогу перейти до сторінки спеціальностей та освітніх програм. Аналогічно до всіх «списків» у системі, тут ми також маємо змогу подивитись всі освітні програми, які пропонує факультет. Адміністратор має змогу виконати CRUD операції, пов’язані з освітніми програмами, форма додавання/редагування зображена на рисунку 3.12.

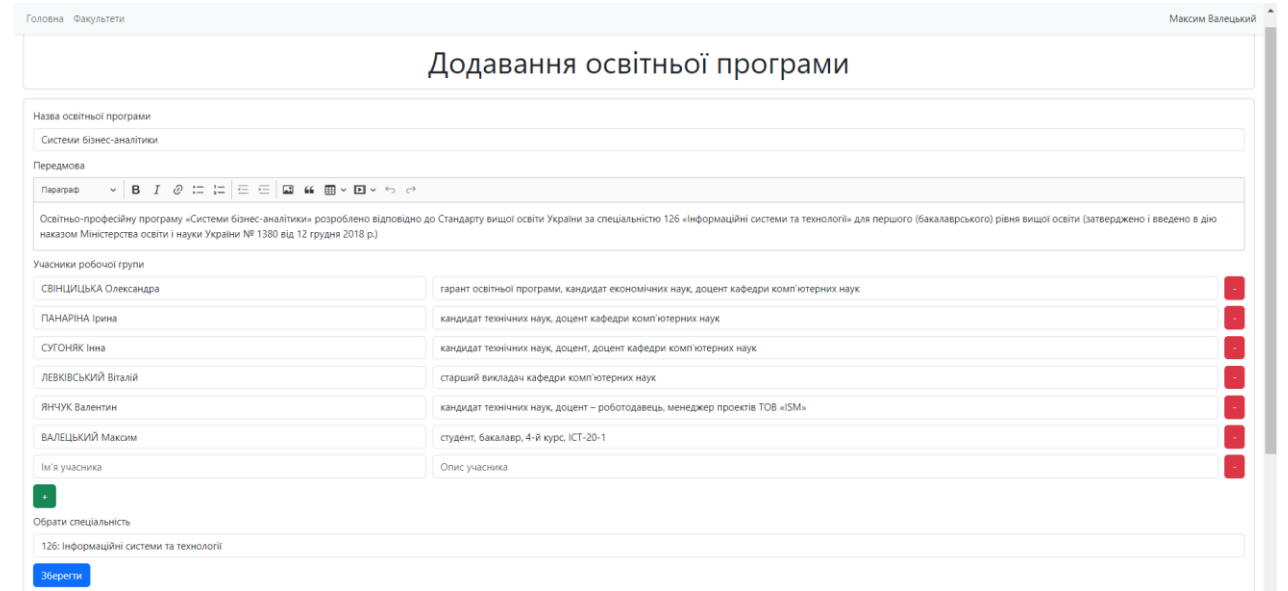


Рисунок 3.12 Форма додавання/редагування освітньої програми

Тут ми можемо заповнити початкову інформацію про освітню програму, а саме передмову, та склад робочої групи, яка складала освітню програму. Форми для динамічного додавання та видалення реалізовані за допомогою бібліотеки jQuery. Дана форма слугує таким собі ініціалізатором для початку повноцінного

додавання освітньої програми. Оскільки як ми вже з'ясували раніше, кожна освітня програма складається з 12 менших компонентів, а саме: Основна інформація, Загальна інформація, Мета освітньої програми, Характеристика освітньої програми, Придатність випускників до працевлаштування та подальшого навчання, Викладання та оцінювання, Програмні компетентності (які включають в себе Інтегральну компетентність, Загальні компетентності, Спеціальні компетентності), Результати навчання, Ресурсне забезпечення реалізації програми, Академічна мобільність, Перелік компонент освітньо-професійної програми (який включає в себе обов'язкові компоненти та вибіркові), а також Форма атестації здобувачів вищої освіти. Також виходячи з цих даних, автоматично формується структурно-логічна схема освітньої програми, а також матриці відповідності програмних компетентностей та результатів навчання обов'язковим компонентам освітньо-професійної програми.

Враховуючи об'єм та кількість компонентів освітньої програми, було вирішено зробити декомпозицію освітньої програми, і для кожного меншого компоненту реалізувати свій CRUD. Тому розглянемо кожен з елементів по порядку:

Головна / Факультети

Головна / Спеціальності факультету

Максим Валецький

Освітня програма "Системи бізнес-аналітики"

Основна інформація

Назва програми
Системи бізнес-аналітики

Передмова
Освітньо-професійну програму «Системи бізнес-аналітики» розроблено відповідно до Стандарту вищої освіти України за спеціальністю 126 «Інформаційні системи та технології» для першого (бакалаврського) рівня вищої освіти (затверджено і введено в дію наказом Міністерства освіти і науки України № 1380 від 12 грудня 2018 р.)

Учасники робочої групи

Ім'я учасника	Опис учасника
СВІНЦІЦЬКА Олександра	гарант освітньої програми, кандидат економічних наук, доцент кафедри комп'ютерних наук
ПАНАРИНА Ірина	кандидат технічних наук, доцент кафедри комп'ютерних наук
СУТОНЯК Ірина	кандидат технічних наук, доцент, доцент кафедри комп'ютерних наук
ЛЕВКІВСЬКИЙ Віталій	старший викладач кафедри комп'ютерних наук
ЯНЧУК Валентин	кандидат технічних наук, доцент – роботодавець, менеджер проєктів ТОВ «ISM»
ВАЛЕЦЬКИЙ Максим	студент, бакалавр, 4-й курс, ІСТ-20-1

Спеціальність освітньої програми
126. Інформаційні системи та технології

Рисунок 3.13 Основна інформація освітньої програми

Основна інформація це ті дані, що ми вводили при ініціалізації освітньої програми. Їх можна відредагувати на сторінці, де міститься перелік усіх освітніх програмю.

Далі, на рисунках 3.14 – 3.16 буде зображено відображення компоненти Загальна інформація

Загальна інформація ✎ 🔖	
Повна назва закладу вищої освіти та структура підрозділу	Державний університет «Житомирська політехніка», факультет інформаційно-комп'ютерних технологій
Ступінь вищої освіти та назва кваліфікації мовою оригіналу	Перший (бакалаврський) рівень вищої освіти Кваліфікація – «Бакалавр з інформаційних систем та технологій»
Офіційна назва освітньої програми	Системи бізнес-аналітики
Тип диплому	Диплом бакалавра, одиничний
Кількість кредитів ЄКТС	240
Термін навчання	3 роки 10 місяців
Навність акредитації	Відсутня
Цикл / рівень	НРК України – 6 рівень, FQ-EHEA – перший цикл, EQF-LLL – 6 рівень
Передумови	Повна загальна середня освіта або наявність освітньо-кваліфікаційного рівня «Молодий спеціаліст», освітнього ступеня «Молодий бакалавр», освітньо-професійного ступеня «Фаховий молодший бакалавр»
Мова(и) викладання	Українська Англійська
Термін дії освітньої програми	Постійно
Інтернет-адреса постійного розміщення опису освітньої програми	https://vstu.ztu.edu.ua/bakalavr/

Рисунок 3.14 Заповнена загальна інформація

Рисунок 3.15 Форма загальної інформації

Рисунок 3.16 Відображення не заповненої загальної інформації

Аналогічним чином до рисунку 3.16 реалізовано відображення кожного не заповненого об'єкту. Тобто якщо інформації про певний пункт ще не було додано, то система так буде відображати повідомлення про її відсутність.

Так само як на рисунку 3.14 та 3.15 реалізовані й інші «інформаційні поля», мається на увазі поля, які несуть в собі текстову інформацію. Відрізняються вони виключно полями, які відображаються, й звідси впливає що й при редагуванні оновлюються інші поля. До таких «інформаційних полів» відносяться такі компоненти як: Загальна інформація, Мета, Характеристика, Придатність випускників, Викладання та оцінювання, Ресурсне забезпечення, Академічна мобільність, Форма атестації. Вони реалізовані за одним принципом, а тому кожен з них детально розглядати немає сенсу.

Тому перейдемо до розгляду компненти Програмні компетентності.

Програмні компетентності

© Доради

Інтегральна компетентність  

Здатність розв'язувати складні спеціалізовані задачі та практичні проблеми в області інформаційних систем та технологій, або в процесі навчання, що характеризуються комплексністю та невизначеністю умов, які потребують застосування теорій та методів інформаційних технологій		
Загальні компетентності (ЗК)		
ЗК 1.	Здатність до абстрактного мислення, аналізу та синтезу	 
ЗК 2.	Здатність застосовувати знання у практичних ситуаціях	 
ЗК 3.	Здатність до розуміння предметної області та професійної діяльності	 
ЗК 4.	Здатність спілкуватися іноземною мовою	 
ЗК 5.	Здатність вчитися і оволодівати сучасними знаннями	 
ЗК 6.	Здатність до пошуку, оброблення та узагальнення інформації з різних джерел.	 
ЗК 7.	Здатність розробляти та управляти проектами.	 
ЗК 8.	Здатність оцінювати та забезпечувати якість виконуваних робіт.	 
ЗК 9.	Здатність реалізувати свої права і обов'язки як члена суспільства, усвідомлювати цінності громадянського (вільного демократичного) суспільства та необхідність його сталого розвитку, верховенства права, прав і свобод людини і громадянина в Україні	 
ЗК 10.	Здатність зберігати та примножувати моральні, культурні, наукові цінності і досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про	 

Рисунок 3.17 Інтегральна та Загальні компетентності

Спеціальні (фахові, предметні) компетентності (СК)		
СК 1.	Здатність аналізувати об'єкт проектування або функціонування та його предметну область.	 
СК 2.	Здатність застосовувати стандарти в області інформаційних систем та технологій при розробці функціональних профілів, побудові та інтеграції систем, продуктів, сервісів і елементів інфраструктури організації.	 
СК 3.	Здатність до проектування, розробки, налагодження та вдосконалення системного, комунікаційного та програмно-апаратного забезпечення інформаційних систем та технологій, Інтернету речей (IoT), комп'ютерно-інтегрованих систем та системної мережної структури, управління ними.	 
СК 4.	Здатність проектувати, розробляти та використовувати засоби реалізації інформаційних систем, технологій та інфокомунікацій (методичні, інформаційні, алгоритмічні, технічні, програмні та інші).	 
СК 5.	Здатність оцінювати та враховувати економічні, соціальні, технологічні та екологічні фактори на всіх етапах життєвого циклу інфокомунікаційних систем	 
СК 6.	Здатність використовувати сучасні інформаційні системи та технології (виробничі, підтримки прийняття рішень, інтелектуального аналізу даних та інші), методики й техніки кібербезпеки під час виконання функціональних завдань та обов'язків	 
СК 7.	Здатність застосовувати інформаційні технології у ході створення, впровадження та експлуатації системи менеджменту якості та оцінювати витрати на її розроблення та забезпечення.	 
СК 8.	Здатність управляти якістю продуктів і сервісів інформаційних систем та технологій протягом їх життєвого циклу	 
СК 9.	Здатність розробляти бізнес-рішення та оцінювати нові технологічні пропозиції.	 
СК 10.	Здатність вибору, проектування, розгортання, інтегрування, управління, адміністрування та супроводжування інформаційних систем, технологій та інфокомунікацій, сервісів та інфраструктури організації.	 
СК 11.	Здатність до аналізу, синтезу і оптимізації інформаційних систем та технологій з використанням математичних моделей і методів.	 
СК 12.	Здатність управляти та користуватися сучасними інформаційно-комунікаційними системами та технологіями (у тому числі такими, що базуються на використанні Інтернет).	 
СК 13.	Здатність проводити обчислювальні експерименти, порівнювати результати експериментальних даних і отриманих рішень	 

Рисунок 3.18 Спеціальні компетентності

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		56

Рисунок 3.19 Форма додавання компетентностей

Рисунок 3.20 Форма редагування компетентності

На рисунках 3.17-3.20 зображені основні складові Програмних компетентностей. На сторінці освітньої програми вони відображаються у вигляді таблиць, і кожен з компетентностей можна окремо відредагувати чи видалити.

На рисунку 3.19 ми можемо спостерігати, що форма додавання є єдиною для всіх компетентностей, про те ми маємо обрати який саме тип компетентності ми хочемо додати. Як видно на тому ж рисунку вибір інтегральної компетентності недоступний, адже вона має існувати в єдиному екземплярі, і в нас вона вже наявна, що видно з рисунку 3.17. А при редагуванні компетенції, автоматично підтягується тип компетентності, який редагується, що ми можемо спостерігати на рисунку 3.20.

Результати навчання

Додати

ПР 1.	Знати лінійну та векторну алгебру, диференціальне та інтегральне числення, теорію функцій багатьох змінних, теорію рядів, диференціальні рівняння для функцій однієї та багатьох змінних, операційне числення, теорію ймовірностей та математичну статистику в обсязі, необхідному для розробки та використання інформаційних систем, технологій та інфокомунікацій, сервісів та інфраструктури організації.	 
ПР 2.	Застосовувати знання фундаментальних і природничих наук, системного аналізу та технологій моделювання, стандартних алгоритмів та дискретного аналізу при розв'язанні задач проектування і використання інформаційних систем та технологій.	 
ПР 3.	Використовувати базові знання інформатики й сучасних інформаційних систем та технологій, навички програмування, технології безпечної роботи в комп'ютерних мережах, методи створення баз даних та інтернет-ресурсів, технології розроблення алгоритмів і комп'ютерних програм мовами високого рівня із застосуванням об'єктно-орієнтованого програмування для розв'язання задач проектування і використання інформаційних систем та технологій.	 
ПР 4.	Проводити системний аналіз об'єктів проектування та обґрунтовувати вибір структури, алгоритмів та способів передачі інформації в інформаційних системах та технологіях.	 
ПР 5.	Аргументувати вибір програмних та технічних засобів для створення інформаційних систем та технологій на основі аналізу їх властивостей, призначення і технічних характеристик з урахуванням вимог до системи і експлуатаційних умов; мати навички налагодження та тестування програмних і технічних засобів інформаційних систем та технологій	 

Рисунок 3.21 Перелік результатів навчання

Аналогічним чином до окремої компетентності реалізовано й результати навчання. Відрізняються вони кодом, ну й звісно місцем, де вони зберігаються.

Перелік компонент освітньо-професійної програми

Додати

Обов'язкові компоненти ОП

Код к/д	Назва компоненти освітньої програми (навчальні дисципліни, курсові проекти/роботи, практики кваліфікаційна робота)	Кількість кредитів	Форма підсумкового контролю	
ОК1	Новозна кова	10	Залік, Екзамен	 
ОК2	Розвиток комунікативних навичок та групова динаміка	3	Залік	 
ОК3	Лінгвістична та аналітична компетенція	3	Залік	 
ОК4	Українська мова, професійна та академічна лінгвістика	3	Залік	 
ОК5	Чисельні методи	3	Екзамен	 
ОК6	Математичний аналіз	7	Залік, Екзамен	 
ОК7	Теорія ймовірностей і математична статистика	7	Залік	 
ОК8	Комп'ютерна дискретна математика	4	Екзамен	 
ОК9	Біологія, безпека життєдіяльності та охорона праці	3	Залік	 
ОК10	Українська історико-культурна та політико-соціальна студія	3	Залік	 
ОК11	Архітектура комп'ютера	3	Екзамен	 
ОК12	Захисні системи безпеки	3	Залік	 
ОК13	Прогнозування	12	Залік, Екзамен, Кваліфікаційна атестація	 
ОК14	Алгоритми та структури даних	4	Екзамен	 
ОК15	Веб-технології	4	Залік	 
ОК16	Інтернет-програмування	6	Екзамен, КР	 
ОК17	Основи економіки та бізнес-аналізу	6	Екзамен	 
ОК18	Безпека даних	9	Залік, Екзамен, КР	 
ОК19	Інформаційна безпека та захист ПЗ	5	Екзамен	 

Вибіркові компоненти ОП

Вибірковий блок 1

У даній навчальній програмі відсутній даний пункт

Вибірковий блок 2

Код к/д	Назва компоненти освітньої програми (навчальні дисципліни, курсові проекти/роботи, практики кваліфікаційна робота)	Кількість кредитів	Форма підсумкового контролю	
ВК2.1	Дисципліна професійної підготовки №1	5	Залік	 

Рисунок 3.22 Відображення компонентів освітньо-професійної програми

Головна

Факультети

Максим Валецький

Головна / Спеціальності факультету / Системи бізнес-аналітики

Додавання компоненту освітньої програми

Виберіть тип компонента:

☐ Обов'язковий
 ☒ Вибірковий

Оберіть блок:

☐ Перший
 ☒ Другий

Назва компонента:

Дисципліна професійної підготовки №1

Кількість кредитів:

5

Форма підсумкового контролю:

Вибір

Форма контролю

☒ Залік
 ☐ Екзамен
 ☐ КР
 ☐ Диф. залік

Рисунок 3.23 Додавання вибіркової компоненти (частина 1)

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		58

Головна Факультети Максим Валецький

☐ КР
☐ Диф. залік
☐ Кваліфікаційна атестація

В яких семестрах буде задіяна компонента:

☐ 1 курс 1 семестр
☐ 1 курс 2 семестр
☒ 2 курс 1 семестр
☐ 2 курс 2 семестр
☒ 3 курс 1 семестр
☐ 3 курс 2 семестр
☐ 4 курс 1 семестр
☐ 4 курс 2 семестр

2 курс 1 семестр
Залік

3 курс 1 семестр
Залік

Надіслати

Головна Факультети

Рисунок 3.23 Додавання вибіркової компоненти (частина 2)

Головна Факультети Максим Валецький

Головна / Спеціальності факультету / Системи бізнес-аналітики

Додавання компоненту освітньої програми

Вибірть тип компоненту:
☒ Обов'язковий ☐ Вибірковий

Назва компоненту:
 Веб-технології

Кількість кредитів:
 4

Форма підсумкового контролю:

Вибір	Форма контролю
☒	Залік
☐	Екзамен
☐	КР
☐	Диф. залік
☐	Кваліфікаційна атестація

В яких семестрах буде задіяна компонента:

☐ 1 курс 1 семестр
☐ 1 курс 2 семестр
☐ 2 курс 1 семестр
☒ 2 курс 2 семестр
☐ 3 курс 1 семестр
☐ 3 курс 2 семестр
☐ 4 курс 1 семестр
☐ 4 курс 2 семестр

Рисунок 3.24 Додавання основної компоненти (частина 1)

2 курс 2 семестр
Залік

Вибірть які загальні компетенції забезпечує компонент:

Вибір	Порядковий номер	Опис
☐	ЗК-1	Здатність до абстрактного мислення, аналізу та синтезу
☐	ЗК-2	Здатність застосовувати знання у практичних ситуаціях
☐	ЗК-3	Здатність до розуміння предметної області та професійної діяльності
☐	ЗК-4	Здатність спілкуватися іноземною мовою
☐	ЗК-5	Здатність вчитися і оволодівати сучасними знаннями
☐	ЗК-6	Здатність до пошуку, оброблення та узагальнення інформації з різних джерел.
☐	ЗК-7	Здатність розробляти та управляти проектами.
☐	ЗК-8	Здатність оцінювати та забезпечувати якість виконуваних робіт.
☐	ЗК-9	Здатність реалізувати свої права і обов'язки як члена суспільства, усвідомлювати цінності громадянського (вільного демократичного) суспільства та необхідність його сталого розвитку, верховенства права, прав і свобод людини і громадянина в Україні

Здатність зберігати та примінювати моральні, культурні, наукові цінності і досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про природу і

Рисунок 3.25 Додавання основної компоненти (частина 2)

Виберіть які спеціальні компетенції забезпечує компонент:		
Вибір	Порядковий номер	Опис
☐	СК-1	Здатність аналізувати об'єкт проектування або функціонування та його предметну область.
☐	СК-2	Здатність застосовувати стандарти в області інформаційних систем та технологій при розробці функціональних профілів, побудові та інтеграції систем, продуктів, сервісів і елементів інфраструктури організації.
☐	СК-3	Здатність до проектування, розробки, налагодження та вдосконалення системного, комунікаційного та програмно-апаратного забезпечення інформаційних систем та технологій, Інтернету речей (IoT), комп'ютерно-інтегрованих систем та системної мережної структури, управління ними.
☐	СК-4	Здатність проектувати, розробляти та використовувати засоби реалізації інформаційних систем, технологій та інфокомунікацій (методичні, інформаційні, алгоритмічні, технічні, програмні та інші).
☐	СК-5	Здатність оцінювати та враховувати економічні, соціальні, технологічні та екологічні фактори на всіх етапах життєвого циклу інфокомунікаційних систем
☐	СК-6	Здатність використовувати сучасні інформаційні системи та технології (виробничі, підтримки прийняття рішень, інтелектуального аналізу даних та інші), методики й техніки кібербезпеки під час виконання функціональних завдань та обов'язків
☐	СК-7	Здатність застосовувати інформаційні технології у ході створення, впровадження та експлуатації системи менеджменту якості та оцінювати витрати на її розроблення та забезпечення.
☐	СК-8	Здатність управляти якістю продуктів і сервісів інформаційних систем та технологій протягом їх життєвого циклу
☐	СК-9	Здатність розробляти бізнес-рішення та оцінювати нові технологічні пропозиції.
☐	СК-10	Здатність вибору, проектування, розгортання, інтегрування, управління, адміністрування та супроводжування інформаційних систем, технологій та інфокомунікацій, сервісів та інфраструктури організації.

Рисунок 3.26 Додавання основної компоненти (частина 3)

Виберіть які навчальні результати забезпечує компонент:		
Вибір	Порядковий номер	Опис
☐	ПР-1	Знати лінійну та векторну алгебру, диференціальне та інтегральне числення, теорію функцій багатьох змінних, теорію рядів, диференціальні рівняння для функцій однієї та багатьох змінних, операційне числення, теорію ймовірностей та математичну статистику в обсязі, необхідному для розробки та використання інформаційних систем, технологій та інфокомунікацій, сервісів та інфраструктури організації.
☐	ПР-2	Застосовувати знання фундаментальних і природничих наук, системного аналізу та технологій моделювання, стандартних алгоритмів та дискретного аналізу при розв'язанні задач проектування і використання інформаційних систем та технологій.
☐	ПР-3	Використовувати базові знання інформатики й сучасних інформаційних систем та технологій, навички програмування, технології безпечної роботи в комп'ютерних мережах, методи створення баз даних та інтернет-ресурсів, технології розроблення алгоритмів і комп'ютерних програм мовами високого рівня із застосуванням об'єктно-орієнтованого програмування для розв'язання задач проектування і використання інформаційних систем та технологій.
☐	ПР-4	Проводити системний аналіз об'єктів проектування та обґрунтовувати вибір структури, алгоритмів та способів передачі інформації в інформаційних системах та технологіях.
☐	ПР-5	Аргументувати вибір програмних та технічних засобів для створення інформаційних систем та технологій на основі аналізу їх властивостей, призначення і технічних характеристик з урахуванням вимог до системи і експлуатаційних умов; мати навички налагодження та тестування програмних і технічних засобів інформаційних систем та технологій
☐	ПР-6	Демонструвати знання сучасного рівня технологій інформаційних систем, практичні навички програмування та використання прикладних і спеціалізованих комп'ютерних систем та середовищ з метою їх запровадження у професійній діяльності.
☐	ПР-7	Обґрунтовувати вибір технічної структури та розробляти відповідне програмне забезпечення, що входить до складу інформаційних систем та технологій.
☐	ПР-8	Застосовувати правила оформлення проектних матеріалів інформаційних систем та технологій, знати склад та послідовність виконання проектних робіт з урахуванням вимог відповідних нормативно-правових документів для запровадження у професійній діяльності.
☐	ПР-9	Здійснювати системний аналіз архітектури підприємства та його IT-інфраструктури, проводити розроблення та вдосконалення її елементної бази і структури

Рисунок 3.27 Додавання основної компоненти (частина 4)

Процес додавання компоненти відображено на рисунках 3.22-3.27. При додаванні нам надається вибір типу компоненти: основний чи вибірковий. В залежності від цього вибору засобами jQuery відображаються інші поля. При виборі основної компоненти також потрібно вибрати які компетентності та результати навчання забезпечує дана компонента, це необхідно для автоматичного формування матриці відповідності, а також у майбутньому це може знадобитись у формуванні робочої програми компоненти.

Окрім цього, в незалежності від типу компоненти, нам необхідно обрати в яких семестрах буде викладатись дана компонента. Це необхідно для автоматичного формування структуро-логічної схеми освітньої програми. Кількість кредитів рівномірно розподіляється між семестрами, які було обрано.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		60

Структурно-логічна схема освітньо-професійної програми

Зазначена кількість кредитів не збігається з фактичною, перерахуйте компоненти програми

Код к/д	Назва компоненти освітньої програми (навчальні дисципліни, курсові проекти/ роботи, практичні кваліфікаційна робота)	Кількість кредитів	Загальний обсяг год.	Форма підсумкового контролю
1 курс 1 семестр				
ОК1	Іноземна мова	2,25	67,5	Залік
ОК2	Розвиток комунікативних навичок та групової динаміки	3	90	Залік
ОК3	Лінійна алгебра та аналітична геометрія	3	90	Залік
ОК6	Математичний аналіз	3,5	105	Залік
ОК11	Архітектура комп'ютера	3	90	Есе/напис
ОК12	Лінійні алгоритми та структури даних	3	90	Залік
ОК13	Програмування	6	180	Залік
Разом:		23,75	712,5	
1 курс 2 семестр				
ОК1	Іноземна мова	2,25	67,5	Залік
ОК4	Українська мова, професійне та владнічне мовлення	3	90	Залік
ОК5	Числові методи	3	90	Есе/напис
ОК6	Математичний аналіз	3,5	105	Есе/напис
ОК7	Теорія ймовірностей і математична статистика	7	210	Залік
ОК13	Програмування	6	180	Есе/напис
ОК14	Алгоритми та структури даних	4	120	Есе/напис
ОК15	Веб-технології	4	120	Залік
Разом:		32,75	992,5	
2 курс 1 семестр				
ОК1	Іноземна мова	2,25	67,5	Залік
ОК6	Комп'ютерна дисципліна математика	4	120	Есе/напис
ОК16	Інтернет-програмування	6	180	Есе/напис
ОК17	Основи економіки та банків-аналізу	6	180	Есе/напис
ОК18	Базі даних	4,5	135	Залік
ВК2.1	Дисципліна професійної підготовки ІВТ	2,5	75	Залік
Разом:		26,25	797,5	

Рисунок 3.28 Відображення структурно-логічної схеми

Таблиця на рисунку 3.28 формується шляхом видобування з кожної компоненти попередньо обраного в них семестру, та сортування саме за цим ключем. Також в межах даної таблиці ведеться підрахунок кількості кредитів та загальної кількості годин. Наприкінці таблиці (рисунок 3.29), ведеться загальний підрахунок кредитів, і якщо ця загальна кількість тобто фактична кількість, не сходиться з тою що має бути (яка була зазначена при ініціалізації освітньої програми), то нам буде про це повідомлено, що ми й можемо спостерігати на рисунку 3.28 (зверху).

4 курс 2 семестр				
ОК1	Іноземна мова	2,25	67,5	Екзамен
Разом:		2,25	67,5	
Загальний обсяг:		29	870	

Рисунок 3.29 Загальний підрахунок кредитів

Матриця відповідності програмних компетентностей обов'язковим компонентам освітньо-професійної програми

	ЗК-1	ЗК-2	ЗК-3	ЗК-4	ЗК-5	ЗК-6	ЗК-7	ЗК-8	ЗК-9	ЗК-10	СК-1	СК-2	СК-3	СК-4	СК-5	СК-6	СК-7	СК-8	СК-9	СК-10	СК-11	СК-12	СК-13	СК-14	СК-15
ОК 1			+	+					+	+															
ОК 2			+	+					+	+															
ОК 3	+	+	+			+					+			+		+				+		+			

Рисунок 3.30 Матриця відповідності програмних компетентностей обов'язковим компонентам освітньо-професійної програми

Матриця забезпечення програмних результатів навчання відповідними обов'язковими компонентами освітньо-професійної програми

	ПР-1	ПР-2	ПР-3	ПР-4	ПР-5	ПР-6	ПР-7	ПР-8	ПР-9	ПР-10	ПР-11	ПР-12	ПР-13
ОК 1													+
ОК 2													+
ОК 3	+												

Рисунок 3.31 Матриця забезпечення програмних результатів навчання відповідними обов'язковими компонентами освітньо-професійної програми

На рисунку 3.30 та 3.31 зображено сформовані матриці відповідностей, які формуються автоматично, відразу ж після оновлення списку компонент, компетентностей та результатів навчання.

В даній секції також передбачені можливі розширення, наприклад у основних компонентах планується додати комірку, за допомогою якої можна буде призначити відповідального за неї викладача, і цей викладач в майбутньому матиме змогу формувати робочу програму. А от наприклад для вибіркового дисциплін можна додати комірку з посиланням на вибірку можливих дисциплін.

Головна / Спеціальності факультету / Системи бізнес-аналітики

Редагування загальної інформації

Рисунок 3.32 Ланцюжок посилань

Виходячи за рамки одного модуля, варто звернути увагу що для зручності у навігації, окрім наявного футеру та хедеру, на кожній сторінці автоматично формується ланцюжок посилань, щоб користувач міг швидко повернутись на попередні по ланцюжки сторінки.

3.3 Тестування інформаційної системи

Заключним етапом в будь-якій ітерації розробки, є тестування розробленого функціоналу. Тестування дозволяє переконатися, що система відповідає всім специфікаціям та вимогам, які були визначені на етапі планування. Це включає як функціональні, так і нефункціональні вимоги. Також якісно протестовані системи легше підтримувати та оновлювати в майбутньому.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Існує чимало видів тестування, про те на даному етапі розвитку системи, доречно скласти чеклист та принаймі один тест-кейс.

Розпочнемо з чеклисту.

Таблиця 3.1 Тестувальний чеклист

	Валецький М.С.	Нотатки
Сайт “TeacherAssistant”	Brave 122.1.63.174	
Факультети		
Перегляд списку факультетів	Passed	Все відображається коректно
Редагування факультета	Passed	
Додавання факультету	Passed	
Видалення факультету	Passed	
Перехід на сторінку факультету	Passed	Перехід відбувається без затримок, при наведенні на кнопку спрацьовує hover ефект
Перехід на сторінку спеціальностей та освітніх програм	Passed	
Кафедри		
Перегляд списку кафедр	Passed	
Редагування кафедри	Passed	Також здійснено перевірку на несанкціонований доступ
Додавання кафедри	Passed	Також здійснено перевірку на несанкціонований доступ
Видалення кафедри	Passed	Також здійснено перевірку на несанкціонований доступ
Перехід на сторінку кафедри	Passed	
Відображення кількості викладачів кафедри	Passed	
Відображення списку викладачів кафедри	Passed	
Освітні програми		
Ініціалізація освітньої програми	Passed	
Редагування початкової інформації	Passed	
Видалення освітньої програми	Passed	Також перевірено чи видалено всі асоційовані записи – перевірка пройдена успішно

Додавання загальної інформації	Passed	
Редагування загальної інформації	Passed	
Видалення загальної інформації	Passed	
Додавання характеристики програми	Passed	
Редагування характеристики програми	Passed	
Видалення характеристики програми	Passed	
Додавання спеціальної компетентності	Passed	
Редагування спеціальної компетентності	Passed	
Видалення спеціальної компетентності	Passed	

Test Case ID	001	Test Case Title	Test the Registration of new user		
Tester's Name	Max	Date Tested	14-Jun-2024	Test Case (Pass/Fail/Not Executed)	
S #	Prerequisites:		S #	Test Data	
1	Access to browser		1	Email = “test-mail@gmail.com”	
2			2	Логін = “test-login”	
3			3	Пароль = “Test-password-1”	
4			4	Пароль(ще раз) = “Test-password-1”	
			5	ПІБ = {Тестовий Тест Тестенко}	
			6	Посада = “Старший викладач”	
			7	Науковий ступінь = Відсутній	
			8	Вчене звання = доцент	
			9	Кафедра = Комп’ютерних наук	
Test Description	Verify if new teacher can be registered				
Step #	Step Details		Actual Results		

		М.С. ВАЛЕЦЬКИЙ		
		М.С. ГРАФ		
Змн.	Арк.	№ докум.	Підпис	Дата

		Expected Results		Pass / Fail / Not executed / Suspended
1	Navigate to teachersassistant/adminPanel/teacherRegistration	New user registration form should open	As Expected	Pass
2	Enter test data in appropriate fields	Data can be entered and entered data is displayed in fields	As Expected	Pass
3	Click Register	Message about successful registration should appear	As Expected	Pass

Висновки до третього розділу

В межах даного розділу було розглянуто окремі рішення в реалізації інформаційної системи, наприклад було розглянуто файлову структуру та загальний принцип роботи системи. За допомогою скріншотів було продемонстровано основний функціонал системи, а також виділено перспективи на майбутнє розширення функціоналу.

Також в межах третього підпункту було проведено функціональне тестування методом чеклисту, а також було складено один тест-кейс. Система успішно пройшла всі тести, що означає, що основних вимог було дотримано та система готова до майбутнього масштабування та експлуатації.

ВИСНОВОК

В результаті виконання кваліфікаційної роботи, було розроблено інформаційну систему «Помічник викладача». Розроблена ІС допомагає викладачам заощадити час шляхом автоматизації та спрощення процесу формування освітніх програм.

В межах роботи було проведено детальний аналіз навчальних процесів, та визначено які саме процеси можуть бути оцифровані та автоматизовані. В межах цієї роботи було вирішено автоматизувати найбільший з цих процесів, а саме формування освітніх програм. Враховуючи наявність такого запиту, було сформовано технічне завдання до виконання, результатом виконання якого, стала готова інформаційна система.

В розробленій ІС реалізовано 6 модулів, а саме: Панель адміністратора, Факультети, Кафедри, Освітні програми, Профіль, Спеціальності. Всі модулі були ретельно протестовані, й в підсумку можна з впевненістю сказати, що система відповідає сформованим раніше вимогам.

Також система готова до розширення та масштабування, адже процес складання освітньої програми – не єдиний що може бути автоматизований. Саме тому розроблена система має надзвичайно високу практичну цінність, адже дозволяє зекономити велику кількість часу викладачам, які в наш час дуже заклопотані.

В якості тестування розробленого продукту, систему було розгорнуто на базі місцевого закладу освіти у місті Здолбунів, а саме у "Здолбунівське вище професійне училище залізничного транспорту". В результаті тестування користувачі залишились задоволені якістю розробленої системи, що означає що поставлена мета була виконана успішно.

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		66

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- [1]Електронний ресурс: How to Benefit Your Business With The Best Frameworks for Web Development in 2024 URL: <https://triare.net/insights/framework-mobile-app/>
- [2]Електронний ресурс: На чому писати сервер: основні мови для бекенда? URL: <https://hyperhost.ua/info/uk/na-comu-pisati-server-osnovni-movi-dlya-bekenda>
- [3]Електронний ресурс: Стандарти правильного розміщення та визначення дій елементів сайту URL: <https://training.qatestlab.com/blog/technical-articles/standards-for-correct-placement-and-definition-of-actions-for-site-elements/>
- [4]Електронний ресурс: Software Architecture Patterns: What Are the Types and Which Is the Best One for Your Project URL: <https://www.turing.com/blog/software-architecture-patterns-types/>
- [5]Електронний ресурс: CSS Media Queries URL: https://www.w3schools.com/css/css3_mediaqueries.asp
- [6]Електронний ресурс: What is Bootstrap Responsive and How to Use It URL: <https://www.simplilearn.com/tutorials/bootstrap-tutorial/bootstrap-responsive>
- [7]Граф М.С., Валецький М.С., Аналіз сучасних тенденцій використання штучного інтелекту при розробці інформаційних систем. Комп'ютерні технології: інновації, проблеми, рішення: тези VI Всеукраїнської науково-технічної конференції, 29-30 листопада 2023 року. Житомир : «Житомирська політехніка», 2023. С.259-261
- [8]Електронний ресурс: Розуміння Клієнт-Серверної Архітектури на прикладах URL: <https://dou.ua/forums/topic/44636/>
- [9]Електронний ресурс: Comparing Software Architecture Patterns MVC Vs. MVVM Vs. MVP: <https://www.masaischool.com/blog/comparing-software-architecture-patterns/>

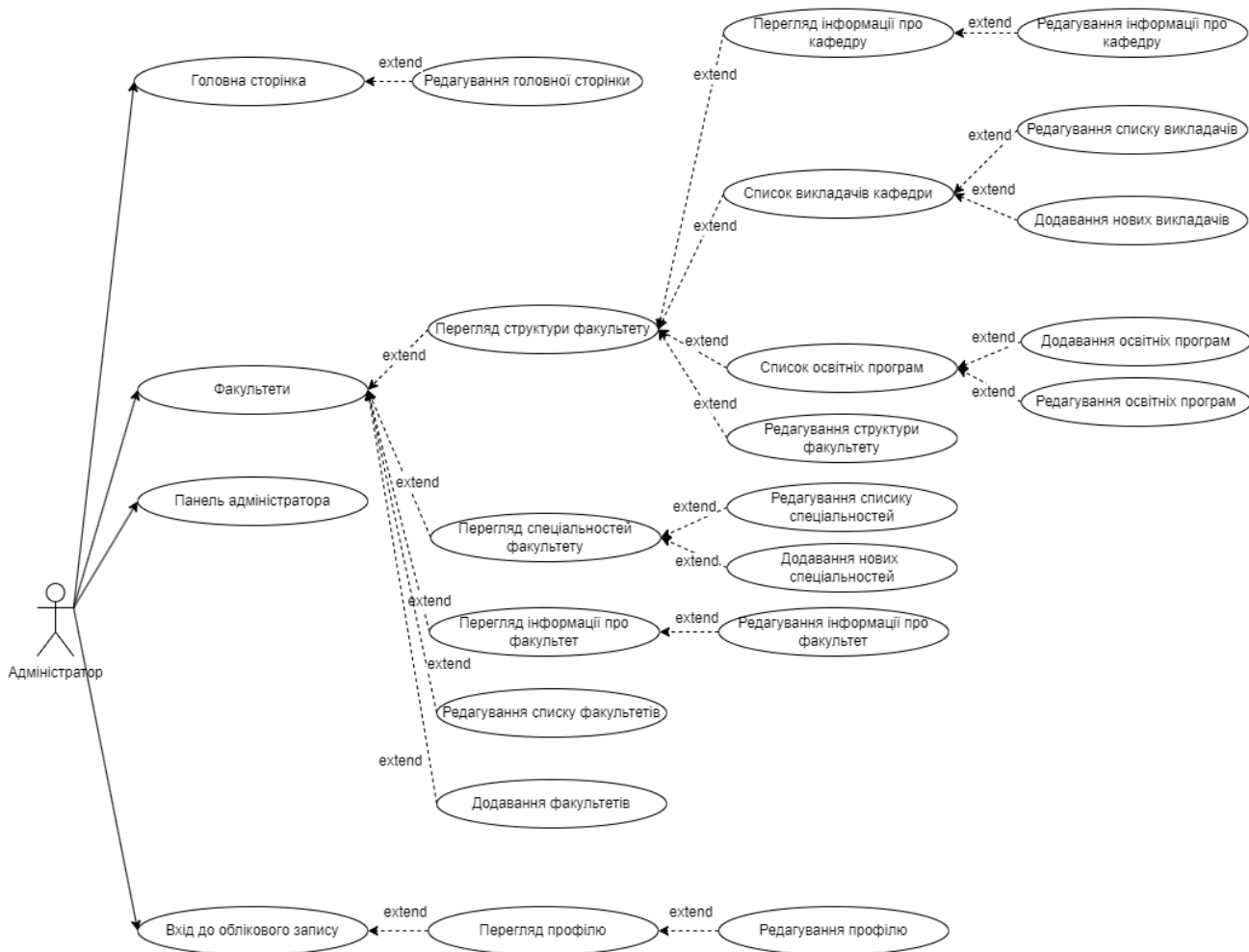
		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		67

- [10]Електронний ресурс: Web Server and Types of Web Servers: Detailed Guide
URL: <https://medium.com/@imjacksonlane/web-server-and-types-of-web-servers-detailed-guide-a7c42061c047>
- [11]Електронний ресурс: Use Case Diagrams | Unified Modeling Language (UML) URL: <https://www.geeksforgeeks.org/use-case-diagram/>
- [12]Електронний ресурс: User Role and Permission Management: Designing Efficient Models URL: <https://frontegg.com/guides/user-role-and-permission>
- [13]Електронний ресурс: Database design basics URL: <https://support.microsoft.com/en-gb/office/database-design-basics-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5>
- [14]Електронний ресурс: Патерн Одинак URL: <https://refactoring.guru/uk/design-patterns/singleton>
- [15]Електронний ресурс: Патерн Будівельник URL: <https://refactoring.guru/uk/design-patterns/builder>
- [16]Електронний ресурс: Освітня програма ICT URL: https://learn.ztu.edu.ua/pluginfile.php/317636/mod_resource/content/1/%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82_%D0%9A-04.01-126_O%D0%9F%D0%9F_2023_%D0%B1%D0%B0%D0%BA.pdf
- [17]Електронний ресурс: PDO - Manual URL: <https://www.php.net/manual/en/book.pdo.php>

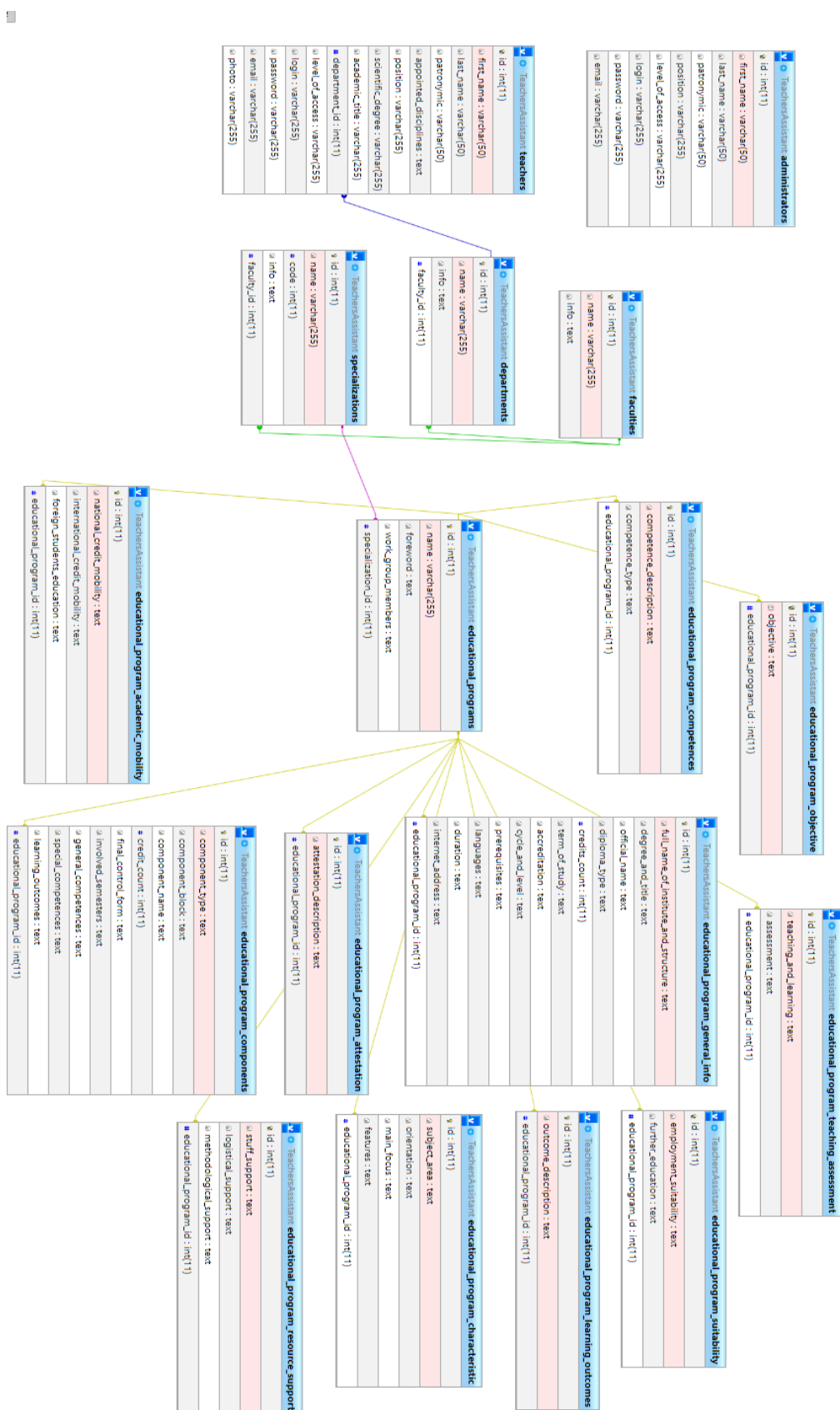
ДОДАТКИ

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		69

Use case діаграма користувача з роллю «Адміністратор»



Загальна структура Бази Даних системи



Контролер модуля «Адміністративної панелі» - клас AdminPanel.php

```

<?php

namespace controllers;

use core\Controller;

class AdminPanel extends Controller
{
    protected $usersModel;
    protected $departmentsModel;
    function __construct()
    {
        $this->usersModel = new \models\Users();
        $this->departmentsModel = new \models\Departments();
    }
    function actionIndex()
    {
        $mainTitle = 'Панель адміністратора';
        $title = '<h1 class="m-0 text-center page-title">Панель
адміністратора</h1>';
        return $this->render('index', null, [
            'PageTitle' => $title,
            'MainTitle' => $mainTitle
        ]);
    }

    function actionTeacherRegistration()
    {
        $mainTitle = 'Реєстрація викладача';
        $title = '<div><a class="text-muted small text-decoration-none"
href="/">Головна</a>
        <span class="text-muted"></span> <a class="text-muted small text-
decoration-none" href="/adminPanel">Панель адміністратора</a>
        </div>
        <h1 class="m-0 text-center page-title">Реєстрація викладача</h1>';
        $titleForbidden = 'Доступ заборонено';
        $departments = $this->departmentsModel->getAllDepartments();
        if($this->usersModel->getLoggedUser()['level_of_access'] == 'admin') {
            if ($this->isPost()) {
                $result = $this->usersModel->addTeacher($_POST);
                if (!is_array($result)) {
                    $allowedTypes = ['image/png', 'image/jpeg'];
                    if(is_file($_FILES['photo']['tmp_name']) &&
in_array($_FILES['photo']['type'], $allowedTypes))
                    {
                        switch ($_FILES['photo']['type']){
                            case 'image/png':
                                $extension = 'png';
                                break;
                            default:
                                $extension = 'jpg';
                        }
                        $name = $result.'_' . uniqid() . '.' . $extension;
                        move_uploaded_file($_FILES['photo']['tmp_name'],
'files/teachers/' . $name);
                        $this->usersModel->handleTeacherPhoto($result, $name);
                    }
                }
                return $this->renderMessage('success', 'Викладач успішно
zareestrovаний', null, [

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		72

```

        'MainTitle' => $mainTitle,
        'PageTitle' => $title
    });
} else {
    $message = implode("<br>", $result);
    return $this->render('teacherRegistration', null, [
        'PageTitle' => $title,
        'MainTitle' => $mainTitle,
        'MessageText' => $message,
        'MessageClass' => 'danger'
    ]);
}
} else {
    $params = [
        'PageTitle' => $title,
        'MainTitle' => $mainTitle
    ];
    return $this->render('teacherRegistration', ['departments' =>
$departments], $params);
}
}
else {
    return $this->render('forbidden', null, [
        'MainTitle' => $titleForbidden,
        'PageTitle' => $titleForbidden
    ]);
}
}
function actionAdminRegistration()
{
    $mainTitle = 'Реєстрація адміністратора';
    $title = '<div><a class="text-muted small text-decoration-none"
href="/">Головна</a>
        <span class="text-muted"></span> <a class="text-muted small text-
decoration-none" href="/adminPanel">Панель адміністратора</a>
    </div>
    <h1 class="m-0 text-center page-title">Реєстрація
адміністратора</h1>';
    $titleForbidden = 'Доступ заборонено';
    if($this->usersModel->getLoggedInUser()['level_of_access'] == 'admin') {
        if ($this->isPost()) {
            $result = $this->usersModel->addAdmin($_POST);
            if (!is_array($result)) {
                return $this->renderMessage('success', 'Адміністратор
успішно зареєстрований', null, [
                    'MainTitle' => $mainTitle,
                    'PageTitle' => $title
                ]);
            }
        } else {
            $message = implode("<br>", $result);
            return $this->render('adminRegistration', null, [
                'PageTitle' => $title,
                'MainTitle' => $mainTitle,
                'MessageText' => $message,
                'MessageClass' => 'danger'
            ]);
        }
    } else {
        $params = [
            'PageTitle' => $title,
            'MainTitle' => $mainTitle
        ];
        return $this->render('adminRegistration', null, $params);
    }
}

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		73

```

    }
    else {
        return $this->render('forbidden', null, [
            'MainTitle' => $titleForbidden,
            'PageTitle' => $titleForbidden
        ]);
    }
}
}

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		74

Моделя модуля для роботи з користувачами – Users.php

```

<?php

namespace models;

use core\Model;
use core\Utils;
use utils\Validator;

class Users extends Model
{
    public function handleTeacherPhoto($id, $file)
    {
        $folder = 'files/teachers/';
        $file_path = pathinfo($folder.$file);
        $file_big = $file_path['filename'].'_b';
        $file_middle = $file_path['filename'].'_m';
        $file_small = $file_path['filename'].'_s';
        $teacher = $this->getTeacherById($id);
        $this->deletePhotoFromFiles($folder, $teacher, $file, '_b.jpg');
        $this->deletePhotoFromFiles($folder, $teacher, $file, '_m.jpg');
        $this->deletePhotoFromFiles($folder, $teacher, $file, '_s.jpg');
        $teacher['photo'] = $file_path['filename'];
        $this->magickImage($folder, $file, $file_big, 1095, 1643);
        $this->magickImage($folder, $file, $file_middle, 256, 384);
        $this->magickImage($folder, $file, $file_small, 131, 131);
        unlink($folder.$file);
        $this->editTeacher($teacher, $id);
    }
    public function addTeacher($teacherData)
    {
        $validationResult = $this->generalTeacherValidation($teacherData);
        if (empty($validationResult)) {
            $fieldsToPush = ['first_name', 'last_name', 'patronymic',
'position',
            'scientific_degree', 'academic_title', 'login', 'password',
'department_id', 'email', 'photo'];
            $filteredTeacherData = Utils::arrayFilter($teacherData,
$fieldsToPush);
            $filteredTeacherData['password'] =
md5($filteredTeacherData['password']);
            $filteredTeacherData['photo'] = '..photo';
            return \core\Core::getInstance()->getDB()->insert('teachers',
$filteredTeacherData);
        }
        return $validationResult;
    }
    public function addAdmin($adminData)
    {
        $validationResult = $this->generalAdminValidation($adminData);
        if (empty($validationResult)) {
            $fieldsToPush = ['first_name', 'last_name', 'patronymic',
'position', 'level_of_access',
            'login', 'password', 'email'];
            $filteredAdminData = Utils::arrayFilter($adminData, $fieldsToPush);
            $filteredAdminData['password'] =
md5($filteredAdminData['password']);
            $filteredAdminData['level_of_access'] = 'admin';
            return \core\Core::getInstance()->getDB()->insert('administrators',
$filteredAdminData);
        }
    }
}

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		75

```

    }
    return $validationResult;
}
public function updateAdmin($adminData, $user)
{
    $insertResult = [];
    $filteredData = Utils::arrayFilter($adminData, ['first_name',
'last_name', 'patronymic', 'position', 'login', 'email']);
    if (Validator::hasEmptyValues($filteredData)) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Усі поля повинні бути заповнені";
        return $insertResult;
    }
    if (!Validator::checkIfLoginIsUsed($filteredData['login'])) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Користувач з таким логіном вже існує";
        return $insertResult;
    }
    if (!Validator::checkIfEmailIsUsed($filteredData['email'])) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Користувач з такою поштою вже існує";
        return $insertResult;
    }
    $insertResult['result'] = true;
    unset($insertResult['error']);
    \core\Core::getInstance()->getDB()->update('administrators',
$filteredData, ['id' => $user['id']]);
    return $insertResult;
}
public function updateTeacher($teacherData, $teacher)
{
    $insertResult = [];
    $fieldsToPush = ['first_name', 'last_name', 'patronymic', 'position',
'scientific_degree', 'academic_title', 'login', 'department_id',
'email', 'photo'];
    $filteredTeacherData = Utils::arrayFilter($teacherData, $fieldsToPush);
    if (Validator::hasEmptyValues($filteredTeacherData)) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Усі поля повинні бути заповнені";
        return $insertResult;
    }
    if (!Validator::checkIfLoginIsUsed($filteredTeacherData['login'])) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Користувач з таким логіном вже існує";
        return $insertResult;
    }
    if (!Validator::checkIfEmailIsUsed($filteredTeacherData['email'])) {
        $insertResult['result'] = false;
        $insertResult['error'] = "Користувач з такою поштою вже існує";
        return $insertResult;
    }
    $insertResult['result'] = true;
    unset($insertResult['error']);
    $insertResult['id'] =
        \core\Core::getInstance()->getDB()->update('teachers',
$filteredTeacherData, ['id' => $teacher['id']]);
    return $insertResult;
}
public function editTeacher($teacher, $id)
{
    $userModel = new \models\Users();
    $fields = ['first_name', 'last_name', 'patronymic', 'position',
'scientific_degree', 'academic_title', 'login', 'password',
'department_id', 'email', 'photo'];

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		76

```

        $filteredRow = Utils::ArrayFilter($teacher, $fields);
        \core\Core::getInstance()->getDB()->update('teachers', $filteredRow,
['id' => $id]);
        return true;
    }
    public function authUser($login, $password)
    {
        $password = md5($password);
        $users = \core\Core::getInstance()->getDB()->select('teachers', '*',
        [
            'login' => $login,
            'password' => $password
        ]);
        if (count($users) > 0)
            return $users[0];
        else {
            $users = \core\Core::getInstance()->getDB()-
>select('administrators', '*',
        [
            'login' => $login,
            'password' => $password
        ]);
            if (count($users) > 0)
                return $users[0];
            else
                return false;
        }
    }

    public function changePassword($data, $user)
    {
        $insertResult = [];
        $fields = ['old_password', 'password', 'new_password_repeat'];
        $filteredData = Utils::arrayFilter($data, $fields);
        if(!Validator::doPasswordsMatch($user['password'],
md5($filteredData['old_password']))) {
            $insertResult['result'] = false;
            $insertResult['error'] = "Неправильно введенний старий пароль";
            return $insertResult;
        }
        if(!Validator::doPasswordsMatch($filteredData['password'],
$filteredData['new_password_repeat'])) {
            $insertResult['result'] = false;
            $insertResult['error'] = "Нові паролі не співпадають";
            return $insertResult;
        }
        if (!Validator::isPasswordValid($filteredData['password'])) {
            $insertResult['result'] = false;
            $insertResult['error'] = "Новий пароль не відповідає мінімальним
вимогам";
            return $insertResult;
        }
        $insertResult['result'] = true;
        unset($insertResult['error']);
        $filteredData['password'] = md5($filteredData['password']);
        $dataToPush = Utils::arrayFilter($filteredData, ['password']);
        if(isset($user['scientific_degree'])) {
            \core\Core::getInstance()->getDB()->update('teachers',
md5($filteredData['password']), ['id' => $user['id']]);
            return $insertResult;
        }
        else {
            \core\Core::getInstance()->getDB()->update('administrators',
$dataToPush, ['id' => $user['id']]);

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		77

```

        return $insertResult;
    }
}
public function generalTeacherValidation($data): array
{
    $arrayOfErrors = [];
    $filteredData = Utils::arrayFilter($data, ['first_name', 'last_name',
'patronymic', 'position',
        'scientific_degree', 'academic_title', 'login', 'password',
        'passwordRepeat', 'department_id', 'email']);
    if (Validator::hasEmptyValues($filteredData))
        $arrayOfErrors [] = "Усі поля повинні бути заповнені";
    if (!Validator::checkIfLoginIsUsed($filteredData['login']))
        $arrayOfErrors [] = "Користувач з таким логіном вже існує";
    if (!Validator::checkIfEmailIsUsed($filteredData['email']))
        $arrayOfErrors [] = "Користувач з такою поштою вже існує";
    if (!Validator::isPasswordValid($filteredData['password']))
        $arrayOfErrors [] = "Пароль не відповідає мінімальним вимогам";
    if (!Validator::doPasswordsMatch($filteredData['password'],
$filteredData['passwordRepeat']))
        $arrayOfErrors [] = "Паролі не співпадають";
    return $arrayOfErrors;
}
public function generalAdminValidation($data): array
{
    $arrayOfErrors = [];
    $filteredData = Utils::arrayFilter($data, ['first_name', 'last_name',
'patronymic', 'position',
        'login', 'password', 'passwordRepeat', 'email']);
    if (Validator::hasEmptyValues($filteredData))
        $arrayOfErrors [] = "Усі поля повинні бути заповнені";
    if (!Validator::checkIfLoginIsUsed($filteredData['login']))
        $arrayOfErrors [] = "Користувач з таким логіном вже існує";
    if (!Validator::checkIfEmailIsUsed($filteredData['email']))
        $arrayOfErrors [] = "Користувач з такою поштою вже існує";
    if (!Validator::isPasswordValid($filteredData['password']))
        $arrayOfErrors [] = "Пароль не відповідає мінімальним вимогам";
    if (!Validator::doPasswordsMatch($filteredData['password'],
$filteredData['passwordRepeat']))
        $arrayOfErrors [] = "Паролі не співпадають";
    return $arrayOfErrors;
}

public function isUserLogged(): bool
{
    return isset($_SESSION['user']);
}

public function getLoggedUser()
{
    if ($this->isUserLogged())
        return $_SESSION['user'];
    else
        return null;
}
public function getTeacherByLogin($login) : array
{
    return \core\Core::getInstance()->getDB()->select('teachers', '*',
['login' => $login]);
}
public function getAdminByLogin($login) : array
{
    return \core\Core::getInstance()->getDB()->select('administrators', '*',
['login' => $login]);
}

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		78

```

    }
    public function getTeacherByEmail($email) : array
    {
        return \core\Core::getInstance()->getDB()->select('teachers', '*',
['email' => $email]);
    }
    public function getAdminByEmail($email) : array
    {
        return \core\Core::getInstance()->getDB()->select('administrators', '*',
['email' => $email]);
    }
    public function getCountOfTeachersByDepartment($department_id) : int
    {
        return count(\core\Core::getInstance()->getDB()->select('teachers', '*',
['department_id' => $department_id]));
    }
    public function getTeachersByDepartment($department_id) : array
    {
        return \core\Core::getInstance()->getDB()->select('teachers', '*',
['department_id' => $department_id]);
    }
    public function getTeacherById($id)
    {
        return \core\Core::getInstance()->getDB()->select('teachers', '*', ['id'
=> $id])[0];
    }
}

```

		М.С. ВАЛЕЦЬКИЙ			ІСТ.КР.Б – 126 – 24 – ПЗ	Арк.
		М.С. ГРАФ				
Змн.	Арк.	№ докум.	Підпис	Дата		79