МІНІСТЕРСТОВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
_____ Аліна Савченко
« » _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ЗДОБУВАЧА ОСВІТНЬОГО СПУПЕНЯ **«БАКАЛАВР»**

Тема: «Веб-застосунок для анонімного рейтингового оцінювання студентами викладачів»

Виконавець: Данило ДУБОВИК

Керівник: к.т.н., доцент Олег ЗУДОВ

Нормоконтролер: к.т.н., доцент Вікторія СИДОРЕНКО

КИЇВ 2024

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних наук та технологій*

Кафедра комп'ютерних інформаційних технологій

Спеціальність 122 «Комп'ютерні науки»

Освітньо-професійна програма «Інформаційні технології проєктування»

ЗАТВЕРДЖУЮ:

завідувач кафедри КІТ

_____ Аліна САВЧЕНКО
(підпис)

«_____» _____ 2024 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

Дубовика Данила Віталійовича

(прізвище, ім'я, по батькові здобувача вищої освіти в родовому відмінку)

1. Тема кваліфікаційної роботи: «Веб-застосунок для анонімного рейтингового оцінювання студентами викладачів»

Затверджена наказом ректора від «05» квітня 2024 р. № 517/ст.

2. Термін виконання роботи: з 06 травня 2024 року по 16 червня 2024 року.

3. Вихідні дані до роботи: «Веб-застосунок для анонімного рейтингового оцінювання студентами викладачів»

4. Зміст пояснювальної записки: 1) Огляд аналогів і постановка задачі. 2) Технології та інструменти для створення веб-застосунку для анонімного рейтингового оцінювання студентами викладачів. 3) Розробка веб-застосунку для анонімного рейтингового оцінювання студентами викладачів

5. Перелік обов'язкового ілюстративного матеріалу: слайди презентації PowerPoint

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівника
1.	Аналіз предметної області. Написання 1 розділу, представлення керівнику.	06.05.2024-13.05.2024	
2.	Вибір та опис засобів програмної реалізації. Написання 2 розділу, представлення керівнику.	14.05.2024-21.05.2024	
3.	Розробка застосунку для анонімного оцінювання викладачів. Написання 3 розділу, представлення керівнику.	22.05.2024-27.05.2024	
4.	Загальне редагування та друк пояснювальної записки.	28.05.2024-30.05.2024	
5.	Проходження нормоконтролю, перепліт пояснювальної записки.	31.05.2024-04.06.2024	
6.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації.	05.06.2024-07.06.2024	

7. Дата видачі завдання «06» травня 2024 р.

Керівник кваліфікаційної роботи _____ Олег ЗУДОВ
(підпис керівника)

Завдання прийняв до виконання _____ Данило ДУБОВИК
(підпис здобувача вищої освіти)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту «Веб-застосунок для анонімного рейтингового оцінювання студентами викладачів»: 99 с., 32 рис., 17 літературних джерел.

Об'єкт дослідження – процес рейтингового оцінювання викладачів у вищих навчальних закладах, а також аспекти, пов'язані з проєктуванням інтерфейсів та розробкою веб-застосунків.

Предмет дослідження – веб-застосунок для анонімного рейтингового оцінювання студентами викладачів в Національному авіаційному університеті.

Мета роботи – розробка веб-застосунку, який дозволить студентам анонімно оцінювати роботу викладачів та зберігати дані опитувань в захищеній та конфіденційній формі.

Методи дослідження – технічні та програмні засоби, порівняльний аналіз, розробка бази даних за допомогою MySQL, функціональних компонентів на основі PHP, веб-інтерфейсу системи за допомогою HTML, CSS та Javascript, обробка літературних джерел.

Результат роботи: розроблено функціонал застосунку для проведення рейтингового оцінювання, використавши обраний стек для фронт-енд та бек-енд розробки. Матеріали дипломного проєкту можуть бути використані при розробці аналогічних систем рейтингового оцінювання для закладів вищої освіти.

Ключові слова: РЕЙТИНГ, РЕЙТИНГОВЕ ОЦІНЮВАННЯ, АЛГОРИТМИ ШИФРУВАННЯ, FRONT-END, BACK-END, ВЕБ-РОЗРОБКА, БАЗА ДАНИХ.

ЗМІСТ

ВСТУП

За останнє десятиліття провідні заклади освіти почали все більше приділяти увагу розробці та впровадженню систем управління, одним із завдань яких є моніторинг якості освіти. Забезпечення належного рівня якості освітніх послуг передбачає постійне стеження за параметрами процесів, які протікають в університеті та окремих його підрозділах з метою отримання достовірної інформації для ефективного управління навчальним процесом в цілому.

Ключовим аспектом якості надання освітніх послуг є забезпечення належного рівня професіоналізму персоналу закладу вищої освіти. Проблема оцінки професійної діяльності окремих викладачів та кафедр завжди була однією з актуальних та водночас найважчих проблем за рахунок своєї залежності від міжособистісних відносин професійної спільноти. На практиці достатньо ефективним інструментом для формування загальної оцінки якості професійної діяльності викладачів закладу є впровадження рейтингу. Робота з формування рейтингових оцінок вимагає збору великої кількості анкет, форм, рейтинг-аркушів та іншої документації. При цьому слід розробити збалансовану систему критеріїв оцінки діяльності педагогів, яка містила б невелику кількість параметрів і при цьому забезпечувала отримання адекватних оцінок. Результати рейтингової оцінки професійної діяльності викладачів мають педагогічний, соціальний та економічний аспекти: оцінка дозволяє приймати оптимальні управлінські рішення, пов’язані з атестацією викладачів, професійною мотивацією, визначенням пріоритетних цілей у галузі управління якістю підготовки спеціалістів у ЗВО. Внутрішні університетські рейтингові системи також можуть надавати вихідну інформацію для прийняття адміністративних управлінських рішень. У зв’язку з цим зростає актуальність робіт, пов’язаних зі створенням теоретичної бази для проєктування рейтингових систем оцінки діяльності викладачів, вибору

типу методики обрахунку рейтингу, відбору показників для проведення оцінювання. Важливою складовою даного питання є цифровізація процесів оцінювання шляхом розробки програмних засобів для розрахунку рейтингу з використанням криптографічних засобів захисту інформації, що здатні забезпечити анонімність опитуваних осіб. Для реалізації даного завдання обрано формат веб-застосунку, створеного з використанням таких мов і технологій, як HTML5, CSS, Javascript, PHP, Laravel, MySQL. Веб-застосунок для рейтингового оцінювання викладачів університету дозволить забезпечити доступність системи для всіх користувачів, незалежно від їх територіального розміщення та пристроїв для доступу.

Як правило, застосунки даного типу мають зручний та інтуїтивно зрозумілий інтерфейс для введення та аналізу даних, а також можливість швидкої обробки та агрегації інформації за допомогою обробки на сервері. Крім того, веб-застосунок може забезпечити безпеку та конфіденційність даних, що є критичним аспектом у сфері освіти. Такий підхід дозволить зробити процес оцінювання більш прозорим, об'єктивним та ефективним для всіх учасників освітнього процесу.

Метою даного дипломного проєкту є розробка веб-застосунку для анонімного рейтингового оцінювання студентами викладачів в Національному авіаційному університеті.

Об’єктом дослідження є процес рейтингового оцінювання викладачів у вищих навчальних закладах, а також аспекти, пов’язані з проєктуванням інтерфейсів та розробкою веб-застосунків.

Предметом дослідження є розробка та впровадження веб-застосунку, який дозволить студентам анонімно оцінювати роботу викладачів та зберігати дані опитувань в захищеній та конфіденційній формі.

Для досягнення поставленої мети необхідно виконати наступні задачі:

– проаналізувати актуальні підходи до рейтингового оцінювання та виробити власну методику для подальшого впровадження у веб-застосунку;

– розглянути існуючі варіанти архітектури та функціоналу застосунків для рейтингового оцінювання;

– провести порівняльний аналіз процесів розробки веб-застосунків та технологій, що використовуються для створення користувацького інтерфейсу, кодування фронт-енду та бек-енду проєкту, а також криптографічних методів, що забезпечать анонімність учасників опитування;

– розробити зручний та інтуїтивно зрозумілий інтерфейс веб-застосунку для користувачів;

– реалізувати базовий функціонал застосунку для проведення рейтингового оцінювання, використавши обраний стек для фронт-енд та бек-енд розробки.

Практичне значення даної роботи: розроблена система надасть можливість студентам анонімно оцінювати роботу викладачів, що сприятиме підвищенню якості навчального процесу та збереженню конфіденційності персональних даних. Крім того, система дозволить адміністраторам ефективно керувати рейтинговими оцінками викладачів, швидко реагувати на виявлені проблеми та забезпечувати відповідність роботи викладачів вимогам до якості навчального процесу, встановленим в закладі.

РОЗДІЛ 1
ОГЛЯД АНАЛОГІВ І ПОСТАНОВКА ЗАДАЧІ

1.1. Аналіз актуальних підходів до рейтингового оцінювання

В загальному розумінні «рейтинг» (від англ. «rating» – оцінка, клас, розряд) – це індивідуальний числовий показник оцінки популярності авторитета певної особи, організації або групи, їх діяльності, програм, планів та політики в певний час. Як правило, рейтинг формується на основі підсумків проведеного голосування, соціологічних опитувань або анкетування. Разом з тим, згідно з найбільш поширеними підходами, прийнятими в навчальних закладах України, рейтинг викладача окрім оцінки на основі опитування має також включати певні об'єктивні (кількісні) результати його діяльності, що охоплюють навчальну, навчально-методичну, науково-дослідницьку, виховну та багато інших видів діяльності, кожну з яких, в свою чергу, можна поділити на підвиди (наприклад, науково-дослідницька робота може включати в себе написання монографій і наукових публікацій, виконання фундаментальних та прикладних досліджень, створення об'єктів інтелектуальної власності). Таким чином, рейтинг викладача повинний являти собою комплексну оцінку, сформовану на основі як опитування студентів та колег, так і через визначення якості його роботи через відповідність певному набору параметрів.

Рейтинг, як система оцінки за формальними показниками має незаперечні переваги – чіткість і зрозумілість критеріїв, простоту проведення і відсутність можливості довільно завищувати або занижувати оцінку залежно від певних суб'єктивних факторів стосовно особи, чия діяльність оцінюється. Загалом

рейтингова оцінка професійної діяльності дозволить:

– поєднати у собі практично всі переваги відомих способів оцінки, оскільки рейтинг не суперечить традиційним принципам оцінки (систематичність та об'єктивність);

– наблизитися до подолання основних недоліків оцінки діяльності викладачів – суб'єктивності та фактичної відсутності кількісних вимірювачів ефективності педагогічної діяльності;

– забезпечити інтегративність оцінки, оскільки вищевказана система має ґрунтуватись на накопиченні умовних одиниць за кожен виконаний викладачами вид наукової, педагогічної та професійної діяльності. Залежно від кількості балів за кожен вид діяльності викладач в результаті отримує адекватну сукупну оцінку.

Рейтинг може органічно поєднуватися з іншими видами оцінки діяльності викладачів, прийнятими у ЗВО. Його завданнями є покращення рівня об'єктивності оцінки діяльності викладацького складу, підвищення професійної мотивації, а також диференціація оцінки діяльності для забезпечення підтримки найефективнішої частини викладацького складу.

Для впровадження рейтингової оцінки професійної діяльності у практику роботи ЗВО необхідними є наступні умови:

– готовність керівництва ЗВО до реалізації запровадження рейтингової оцінки професійної діяльності, що виявляється у розробці організаційно-управлінських документів, які дозволять розпочати роботу з проведення оцінювань;

– ретельна розробка оціночних критеріїв;

– автоматизація роботи з обліку та аналізу рейтингових показників на рівні закладу.

Структура рейтингової оцінки має включати наступні компоненти: модель якості діяльності викладача, математичну модель розрахунку рейтингу

викладача, рейтингові кваліфікаційні шкали. Модель якості в свою чергу може включати оцінки, розділені на групи за однією з ознак:

– часом: найбільш поширеним підходом є розподіл рейтингових оцінок за фактичним статусом викладача (минуле), результатами роботи та продуктивністю (поточний момент) та потенційними можливостями (майбутнє);

– специфікою діяльності: індивідуальний рейтинг викладача може включати в себе три комплексні оцінки для наукової, педагогічної та професійної діяльності. Кожна з оцінок в свою чергу може містити розподіл на підпункти, характерний для попереднього підходу (тобто статус, продуктивність та перспективність). Всі параметри повинні мати формальні показники, виражені в балах та певні коефіцієнти. Наприклад, коефіцієнт для параметру «продуктивність» може становити 0,9, для параметру «перспективність» – 0,7, для параметру «статус» – 0,5.

– розподіл видів діяльності викладача на ієрархічні групи. За даного підходу в кожній з трьох груп (наукова, педагогічна та професійна діяльність) виділяються окремі підпункти для оцінювання. Кожен з підпунктів може або мати певну визначену кількість балів (в залежності від його важливості), або ж коефіцієнт, що враховуватиметься при формуванні загальної оцінки.

Приклад схеми комплексного показника рейтингу викладача згідно поділу оцінок за специфікою діяльності показано на рис. 1.1.

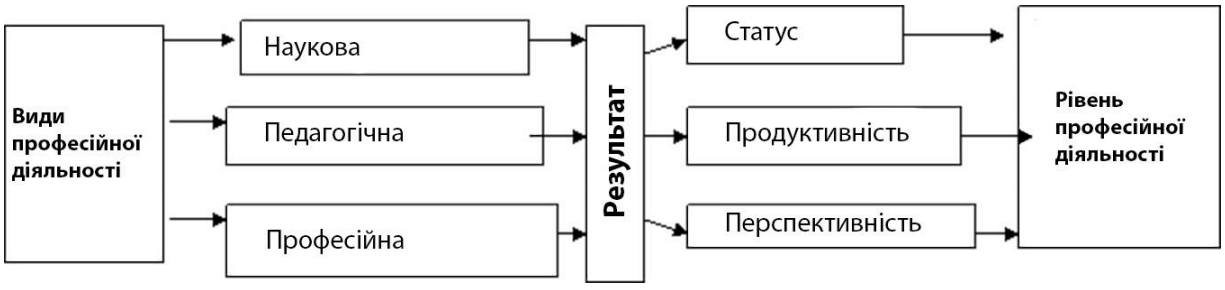

Рис. 1.1. Схема комплексного показника рейтингу

Для обрахунку загального рейтингу викладача пропонується використовувати систему ієрархічної структури, що включає три рівні за

окремими видами діяльності та показниками роботи, які відносяться до даного виду діяльності. При такому підході рейтингом буде виступати структурований набір числових значень, що дозволятиме з різних ракурсів оцінити вклад даного викладача в роботу навчального закладу. Таким чином, ієрархічна структура надаватиме додаткову інформацію для аналізу результатів діяльності викладача, а також проміжну рейтингову інформацію, що може використовуватись для стимулювання конкретного виду діяльності в разі отримання низьких показників. В якості додаткової оціночної інформації можна використовувати динаміку зміни рейтингу, що дозволить оцінити перспективність викладача. Також пропонується проводити самооцінку викладача за тими ж параметрами, що і загальну оцінку.

Узагальнену схему оцінювання за ієрархічною структурою продемонстровано на рис. 1.2.

Рис. 1.2. Ієрархічна структура загального рейтингу викладача

Оцінка, отримана від анонімного опитування студентів, за даного підходу буде використовуватись, як складова для розрахунку загального рейтингу працівника. Пропонується, що загальний рейтинг включатиме бали за наступні види діяльності:

– показник якості навчально-педагогічної роботи за результатами анонімного анкетування студентів;

– показник якості навчально-педагогічної роботи за результатами анонімного анкетування інших педагогічних працівників;

– оцінка наукової діяльності, що формується на основі балів, отриманих за видання підручників, навчально-методичних матеріалів, посібників, публікацію наукових робіт, виступи на конференціях, захист дисертації на здобуття наукового ступеня;

– оцінка педагогічної діяльності, яку формують бали за розробку робочих програм для навчальних дисциплін, оновлення програм практики, пакетів завдань для контрольних робіт або екзаменаційних білетів, видання конспектів лекцій, методичних вказівок до лабораторних та практичних робіт, проведення практики, організацію відкритих занять, тренінгів або майстеркласів, тематичних екскурсій та заходів, керівництво студентськими гуртками;

– оцінка професійної діяльності, до складу якої входять бали за участь в професійних конкурсах, підвищення кваліфікації, виконання обов'язків завідувача відділення або кафедри, організацію конференцій, конкурсів, студентських олімпіад.

Для обчислення загального рейтингу може використовуватись портфоліо, сформоване викладачем на кінець навчального року. Разом з тим рейтинг може знижуватись за рахунок порушень трудової дисципліни, отриманих доган, відсутність без поважних причин на робочому місці, неналежну якість навчально-педагогічної роботи, тощо.

Для обрахунку складової рейтингу викладачів університету, що формується за допомогою анонімного опитування студентів можуть використовуватися різні моделі, які базуються на аналізі отриманих даних. Серед існуючих методів обрахунку можна виділити:

1. Середній бал: підхід, за якого рейтинг викладача обчислюється як середнє арифметичне всіх оцінок, які він отримав від студентів. Даний підхід

простий в реалізації, але не враховує можливих варіацій у важливості різних аспектів роботи викладача.

2. Зважений середній бал: за даного методу різні аспекти роботи викладача оцінюються окремо (наприклад, для окремого оцінювання можна виділити зрозумілість викладання, доступність подачі матеріалу, комунікативність, підготовку до занять, дотримання графіку, тощо), які можуть мати різну вагу. Студенти оцінюватимуть кожен аспект окремо, і потім ці оцінки буде зважено відповідно до їх впливу.

3. Факторний аналіз: застосування даного підходу дозволяє виявити основні фактори, які впливають на загальне задоволення студентів від роботи викладача (наприклад, якість лекцій, доступність для індивідуальних консультацій, якість оцінювання, тощо) та побудувати математичну модель для обрахунку підсумкового рейтингу на основі впливу цих факторів.

4. Використання моделей машинного навчання (наприклад, моделі випадкового лісу або нейронних мереж), які можуть передбачати рейтинг викладача на основі відгуків студентів. Для цього можна застосовувати як числові оцінки, так і текстові коментарі.

Також слід зазначити, що модель оцінювання може враховувати динаміку оцінок викладача з часом, дозволяючи виявляти тенденції та зміни в його роботі.

В якості оптимального способу для формування оцінки на основі анонімного опитування серед студентів було обрано зважений середній бал з відповідним набором критеріїв, до кожного з яких у відповідність поставлено певну вагу для врахування загальної оцінки. Пропонується використовувати набір з 20 параметрів для оцінювання, вага кожного з яких представлена в таблиці 1.1.

Рівень рейтингу кожного викладача встановлюється відповідно до прийнятого інтервалу значень щодо середнього і буває: високим, вищим за середній, нижчим за середній і низьким з опорою на нормальний розподіл.

Наприклад, нехай x – середнє значення рейтингу для викладацького складу відповідного факультету закладу, а n – рейтинг конкретного викладача, тоді положення відносного середнього будуть наступними: високий в інтервалі 1,5x<n, вище за середній – x<n<1,5x, нижче за середній – 0 5x<n<x, низький – n<0,5x.

Таблиця 1.1

Параметри для анонімного оцінювання якості роботи викладача студентами

№	Параметр оцінювання	Ваговий коефіцієнт, %
1	Доступність подачі матеріалу	10
2	Здатність адаптуватись під індивідуальні особливості навчальної групи	7
3	Комунікативність	5
4	Підготовка до занять	6
5	Дотримання графіку занять	3
6	Якість лекцій	7
7	Доступність для індивідуальних консультацій	5
8	Інтерактивність занять	5
9	Інтерес до предмету	3
10	Сприяння активності студентів на заняттях	4
11	Ефективність використання додаткових навчальних матеріалів	5
12	Використання сучасних методів навчання	6
13	Об’єктивність при оцінюванні студентів	4
14	Стимулювання самостійної роботи студентів	4
15	Досвід викладання предмету	4
16	Відповідність навчального процесу програмним вимогам	4
17	Вміння мотивувати студентів до навчання	4
18	Дотримання етичних стандартів	4
19	Здатність роз’яснити складні поняття	5
20	Створення сприятливої атмосфери на заняттях	5

1.2. Характеристика Національного авіаційного університету

Національний авіаційний університет (НАУ) є провідним вищим навчальним закладом, що станом на 2024 рік налічує близько 25 тисяч студентів [1]. Заснований 25 серпня 1933 року, університет визначається своєю високою академічною репутацією та широким спектром наукових

напрямків. НАУ відомий своєю потужною науковою базою та провідними науковими школами в галузях управління, механіки, електроніки, електротехніки, матеріалознавства, інформатики та обчислювальної техніки.

Основною конкурентною перевагою університету як на внутрішньому, так і на міжнародному рівні, є його авіаційна спрямованість. Навчальний заклад готує фахівців для авіаційної галузі, включаючи пілотів, операторів безпілотних літальних апаратів, спеціалістів з наземного обслуговування авіаційної техніки та авіаційних двигунів, диспетчерів, інженерів управління повітряним рухом, дизайнерів аеропортів, споруд та доріг, фахівців аеропортових служб, менеджерів з авіаційних перевезень та логістики, спеціалістів з кіберзахисту та діагностики систем, а також фахівців з інших сфер авіаційної індустрії.

Університет має розвинену науково-педагогічну базу. До складу колективу закладу входить 15 академіків та членів-кореспондентів Національної академії наук України, 184 доктори наук, професори і 677 кандидатів наук, доцентів. До навчального процесу залучаються спеціалісти авіакомпаній і промислових підприємств. Серед викладачів налічується 80 лауреатів Державної премії та Заслужених діячів науки і техніки.

На даний момент Національний авіаційний університет складається з п’яти інститутів, десяти факультетів, кафедри військової підготовки та наступних відокремлених структурних підрозділів:

– Вище професійне училище Національного авіаційного університету (м. Київ);

– Васильківський фаховий коледж Національного авіаційного університету (м. Васильків);

– Криворізький фаховий коледж Національного авіаційного університету (м. Кривий Ріг);

– Слов’янський фаховий коледж Національного авіаційного університету (м. Слов’янськ);

– Фаховий коледж інженерії, управління та землевпорядкування Національного авіаційного університету (м. Київ);

– Київський фаховий коледж комп’ютерних технологій та економіки Національного авіаційного університету (м. Київ);

– Льотна академія Національного авіаційного університету (м. Кропивницький).

В НАУ функціонують наступні інститути та факультети:

– Інститут ІКАО;

– Інститут новітніх технологій та лідерства;

– Навчально-науковий інститут міжнародного співробітництва та освіти;

– Навчально-науковий інститут неперервної освіти;

– Навчально-науковий інститут розвитку освіти;

– Аерокосмічний факультет;

– Кафедра військової підготовки НАУ (факультет);

– Факультет аеронавігації, електроніки та телекомунікацій;

– Факультет екологічної безпеки, інженерії та технологій;

– Факультет економіки та бізнес-адміністрування;

– Факультет кібербезпеки та програмної інженерії;

– Факультет комп`ютерних наук та технологій;

– Факультет лінгвістики та соціальних комунікацій;

– Факультет міжнародних відносин;

– Факультет наземних споруд і аеродромів;

– Факультет транспорту, менеджменту і логістики;

– Юридичний факультет.

Крім того, в університеті функціонують навчальний аеродром, полігон авіаційної наземної техніки, тренажерні комплекси та навчально-науковий аеродинамічний комплекс на базі дозвукової аеродинамічної труби ТАД-2. Студентське містечко університету включає 11 гуртожитків, їдальню на 1000

місць, інтернет-кафе, студентське «Бістро», медичний центр, профілакторій, Центр культури і мистецтв із залом на 1500 місць та навчально-спортивний оздоровчий центр. Книжковий фонд Науково-технічної бібліотеки НАУ складає понад 2,6 млн. примірників.

У 2008 році університет отримав сертифікат відповідності системи менеджменту якості освітніх послуг та наукових досліджень стандарту якості ISO9001:2000.

У жовтні 2023 року Національний авіаційний університет був розділений на два заклади вищої освіти – державний університет «Київський авіаційний інститут» і Українську державну льотну академію.

1.3. Огляд архітектури та функціоналу застосунків для рейтингового оцінювання

Рейтингова система, за якої студенти оцінюють якість викладання за певним пройденим курсом або дисципліною, широко використовується для збору статистики про якість навчального процесу. В деяких випадках системи для оцінювання реалізуються як окремі модулі до LMS навчального закладу, проте, частіше за все, вони являють собою окремі веб-програми, що дозволяють адміністраторам додавати коледжі, факультети, курси, класи, предмети та користувачів, а також створювати критерії оцінювання і анкети, після чого власне і проводити оцінювання викладачів студентами. Деякі з подібних систем також дозволяють студентам залишати коментарі. Після того, як всі студенти приймуть участь в оцінюванні, система автоматично сформує звіт, який може бути використаний завідувачами факультетів та кафедр, або адміністрацією закладу для вдосконалення якості освітніх процесів.

Для розробки системи даного класу необхідне поєднання наступних технологій та мов програмування:

– мови програмування для бек-енду, що дозволить налаштувати відображення динамічного контенту та збереження інформації в базі даних;

– набору мов та засобів для програмування фронт-енду (мови розмітки HTML, каскадних таблиць стилів CSS з можливим використанням одного з препроцесорів, наприклад SCSS, мови Javascript з можливим використанням фреймворків);

– реляційної (SQL) або NoSQL (в залежності від структури даних) системи управління базами даних;

– веб-серверу;

– середовища розробки.

Передбачається, що користувачі системи для оцінювання викладачів будуть належать до однієї з шести наступних груп: адміністратори, студенти, викладачі, завідувачі кафедр, завідувачі факультетів, декани. Адміністратори несуть відповідальність за управління даними користувачів, коледжів, відділів, курсів, класів, предметів, оцінювання та звітів. Користувачам-студентам доступний лише функціонал для авторизації в системі та участі в опитуванні (опціонально розглядається можливість додавати коментарі). Викладачі мають можливість переглядати власні індивідуальні звіти з загальним рейтингом та середніми значеннями за окремими параметрами. Завідувачі кафедр та факультетів можуть переглядати індивідуальні звіти викладачів, що належать до їх кафедри або факультету відповідно. Декани мають доступ до всіх індивідуальних звітів викладачів, а також зведених звітів по кафедрах та факультетах закладу.

Зі сторони адміністратора важливим є забезпечення наступного функціоналу:

1. Керування оцінюванням – адміністратор може створювати структуру анкет та керувати критеріями оцінювання, а також встановлювати обмеження для оцінювань (це означає, що студенти зможуть оцінювати лише тих викладачів, які проводили заняття з ними). Адміністратор також може встановити період для оцінювання (наприклад, оцінка за результатами проведеного опитування може бути прив’язаною до певного часового

діапазону і слугуватиме характеристикою викладача лише за певний навчальний рік або семестр.

2. Створення звітів з результатами оцінювання – оскільки викладачі, завідувачі кафедр та факультетів, а також декани не можуть переглядати окремі опитування, система має формувати зведений рейтинг (звіт) та відображати його в інтерфейсі відповідних груп користувачів. Система може або ж автоматично оновлювати звіти щоразу, коли новий студент прийме участь в опитуванні (при цьому опитування будуть відкритими під час всього навчального року), або ж формувати звіт, коли закінчиться опитування, на проведення якого буде відведено певний проміжок часу (наприклад, два останній тижні семестру).

3. Управління відділами, курсами, класами та предметами – функція, що полягає в побудові ієрархічної структури закладу. Може виконуватись або вручну адміністратором, або через синхронізацію з іншими інформаційними системами університету. Передбачається, що адміністратор має можливість редагувати дані всередині системи, створювати профілі користувачів, встановлювати їх зв’язок з певними кафедрами, факультетами та навчальними групами, а також керувати рівнями доступу до інформації.

Для користувачів, що належать до груп викладачі, декани, керівники кафедр та керівники факультетів основним функціоналом в системі є авторизація та перегляд звітів.

Для користувачів-студентів єдиним доступним функціоналом в системі є авторизація та участь в опитуваннях. В залежності від підходу, студенти матимуть або доступ до переліку відкритих на даний момент опитувань (якщо опитування проводитимуться в рамках певного фіксованого періоду часу), або ж до переліку власних викладачів, яких вони зможуть оцінювати в будь-який момент (при цьому також важливо забезпечити студентам можливість відредагувати власну оцінку в майбутньому в разі необхідності). Передбачається, що для оцінювання кожного з 20 параметрів буде

використовуватись десятибальна шкала. Оцінки та коментарі студентів є конфіденційними, а тому жоден інший учасник не має права перегляду проведених опитувань.

На рис. 1.3 представлено діаграму «вхід-процес-вихід», яка описує процеси перетворення всіх вхідних даних, включаючи дані для авторизації, дані користувача, структуру відділів, курсів та предметів, а також деталі оцінювання, анкет та встановлених обмежень, необхідні для обробки інформації з подальшим формуванням звітності.

Рис. 1.3. Діаграма «вхід-процес-вихід»

На рис. 1.4 показано діаграму послідовності для облікового запису студента. Студент представляє актора системи, що взаємодіє з користувацьким інтерфейсом системи, сервером та базою даних. Послідовність починається зі створення студентом облікового запису (у випадку, якщо він ще не має облікового запису), після чого відбуваються процеси авторизації в системі, вибору викладача, введення оцінок в анкету та передачі анкети на обробку (результати опитування будуть оброблені та записані в базі даних).

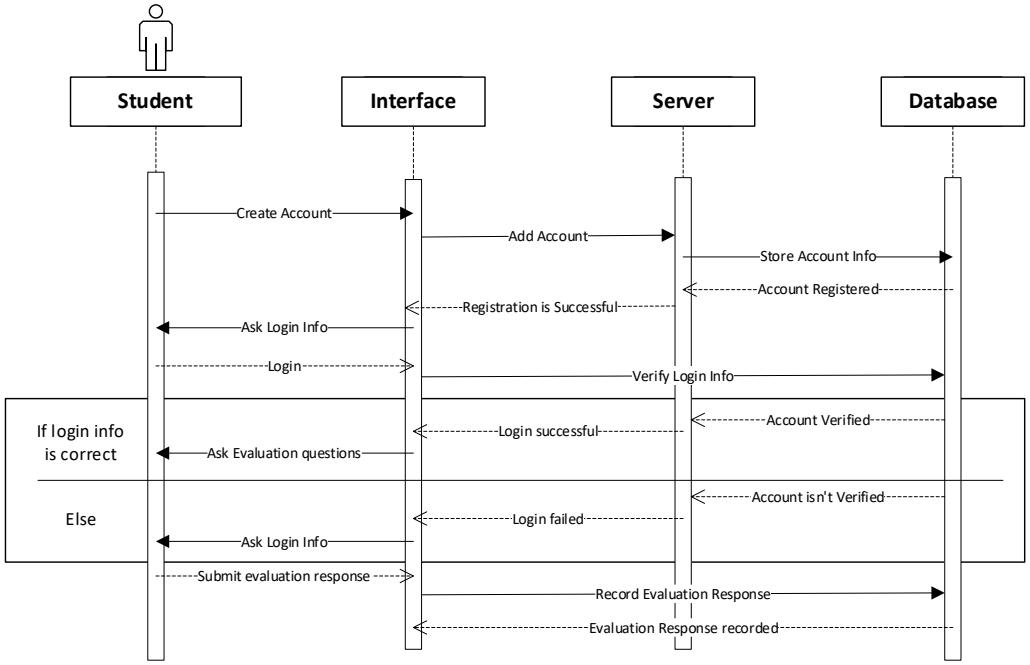

Рис. 1.4. Діаграма послідовності для категорії «студент»

На рис. 1.5. показана діаграма діяльності для категорії «студент». Вона описує поетапні процеси, класифіковані за доступними модулями системи («профіль» та «оцінювання»).

На рис. 1.6. продемонстровано діаграму послідовності для адміністратора системи. Як і у випадку зі студентом, адміністратор взаємодіє з користувацьким інтерфейсом, сервером та базою даних. Послідовність починається з авторизації для доступу до адміністративної панелі. Отримавши доступ, адміністратор може створювати або редагувати структуру відділів, курсів та предметів, створювати опитування, додавати критерії оцінювань, запитання та умови (або обмеження) для їх проведення. Адміністратор також може переглядати зведені рейтинги викладачів, сформовані системою.

Діаграма діяльності адміністратора системи показана на рис. 1.7. Для більш точного представлення робочих процесів, дії класифіковано на основі модулів, включених до облікового запису адміністратора – «профілі», «структура ЗВО», «користувачі», «оцінювання», «звіти».

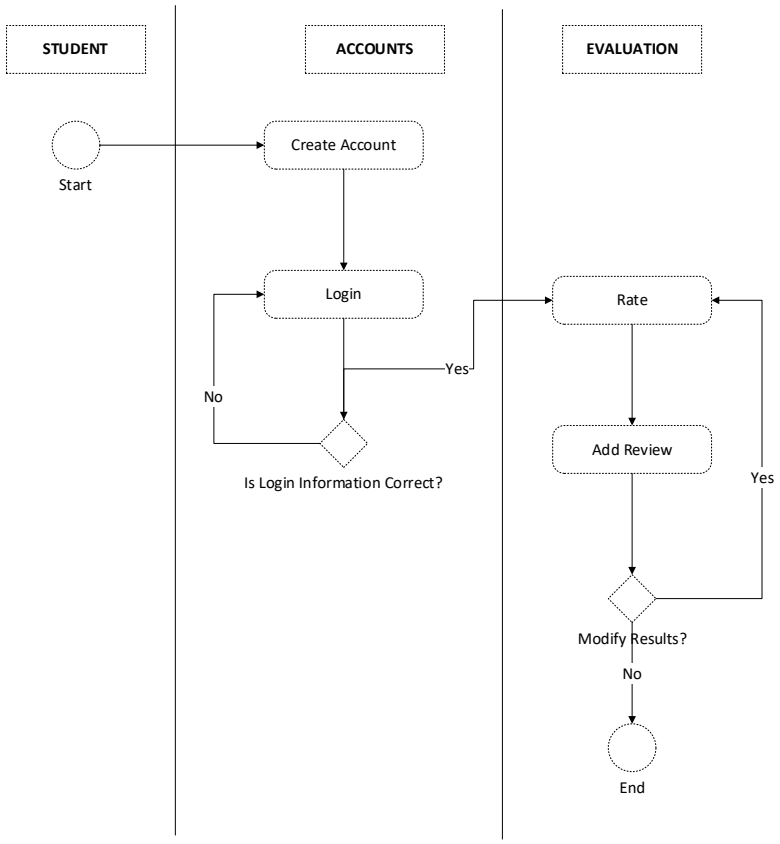

Рис.1.5. Діаграма діяльності для категорії «студент»

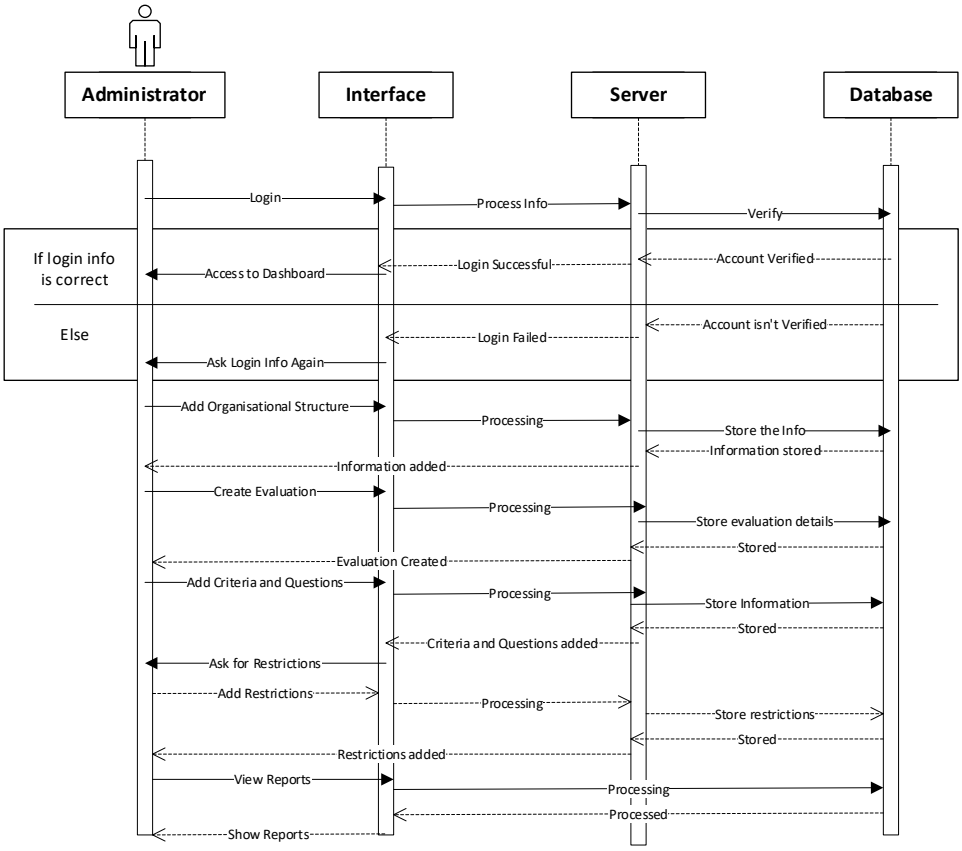

Рис. 1.6. Діаграма послідовності для категорії «адміністратор»

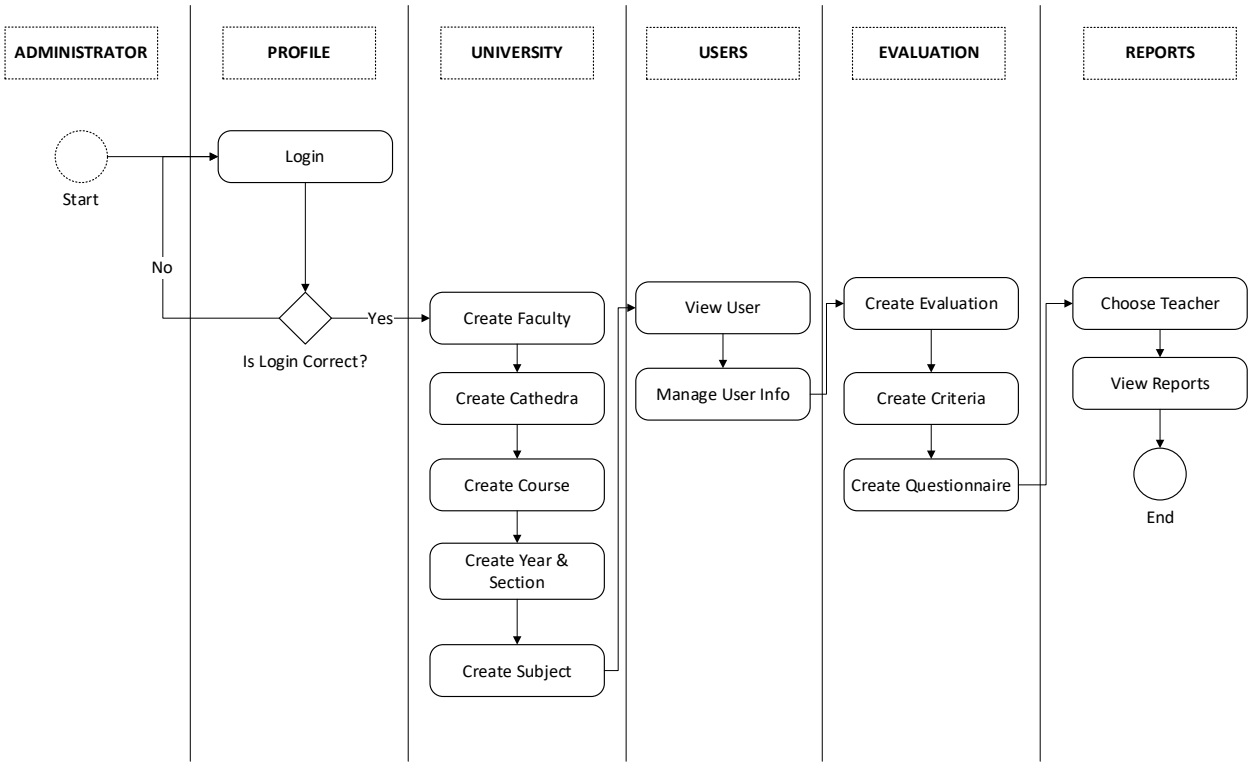

Рис.1.7. Діаграма діяльності для категорії «адміністратор»

На рис. 1.8 показана діаграма варіантів використання запропонованої системи, яка містить шість різних типів користувачів: студент, викладач, завідувач кафедри, завідувач факультету, декан та адміністратор системи. Діаграма варіантів використання відображає всі дії, які можуть здійснювати користувачі кожного з перерахованих типів. Так, наприклад, студенти можуть увійти, зареєструватися, оцінити вчителів і додати коментарі. Викладачі можуть увійти та переглянути індивідуальні звіти. Завідувачі кафедр та факультетів можуть увійти та переглянути індивідуальні звіти всіх викладачів кафедри або факультету. Декани можуть увійти та переглянути індивідуальні звіти всіх викладачів у закладі. Системний адміністратор може входити в систему, керувати рейтингами, користувачами, рейтингами, а також створювати звіти для подання деканам, керівникам кафедр і факультетів.

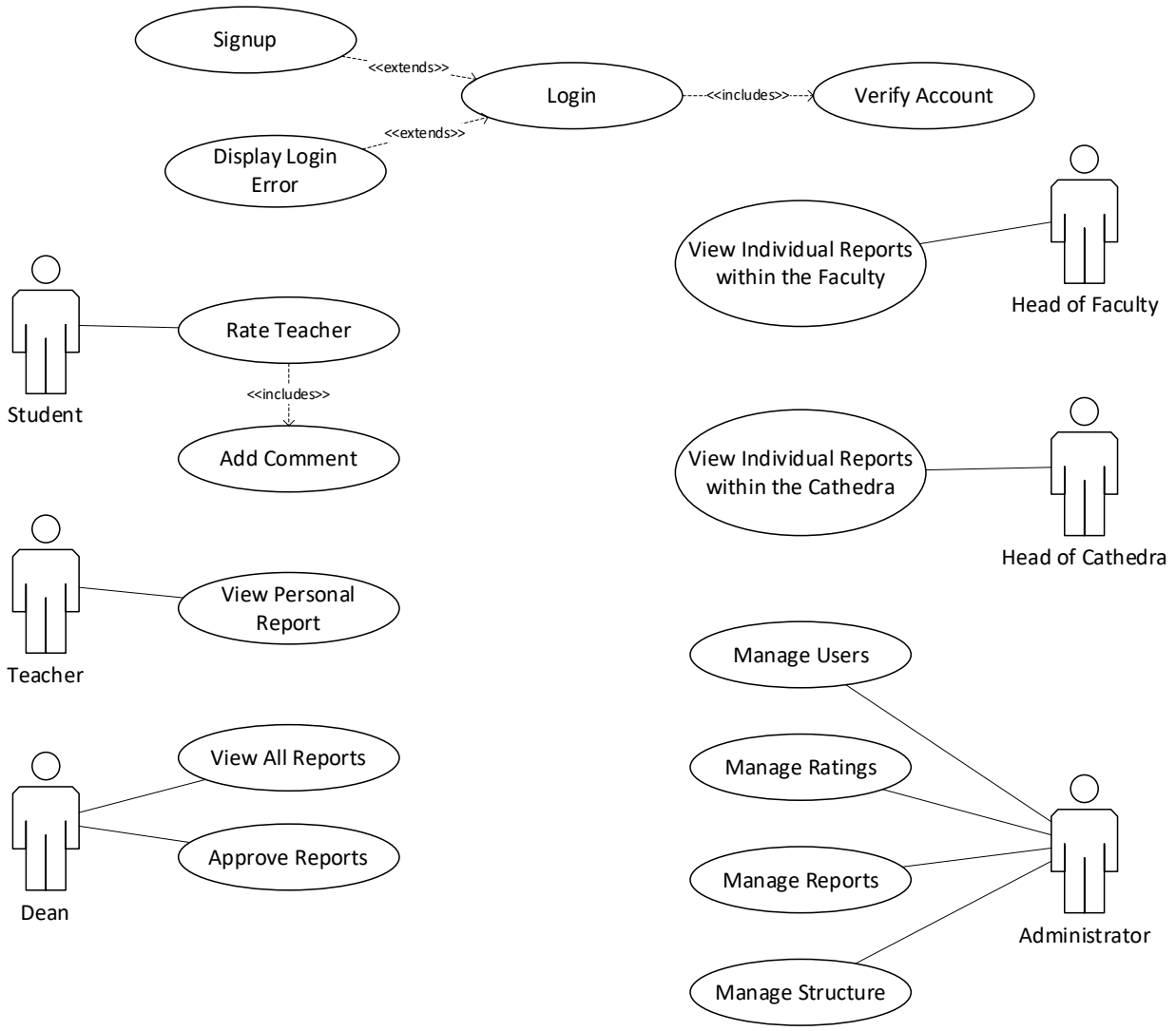

Рис. 1.8. Діаграма варіантів використання системи

1.4. Порівняльний аналіз застосунків

При порівнянні існуючих застосунків слід враховувати, що в Україні на даний момент не існує загальнодоступних веб-порталів, які б давали можливість студентам оцінювати конкретних викладачів власного закладу за певним набором критеріїв. Є лише внутрішньоуніверситетські розділи на веб-сайтах або в рамках наявної LMS, що дозволяють проводити опитування та формувати підсумкову оцінку. Деякі заклади також проводять опитування, використовуючи Google Forms, що також мають недоліки через обмеженість доступного функціоналу. Разом з тим, існує велика кількість закордонних веб-проєктів, що дозволяють оцінювати викладацький склад (як публічно, у

форматі соціальної мережі з відгуками студентів, так і анонімно). Поширеність даного типу веб-проєктів можна пояснити практикою індивідуального вибору курсів і викладачів в закордонних закладах освіти. Розглянемо дані веб-застосунки більш детально.

1. Rate My Professors – портал, що в контексті вищої освіти виконує кілька функцій [2]:

– Підвищення якості викладання: професори можуть отримати вигоду від конструктивної критики, яка надається через портал. Рейтинг та коментарі дають чітке розуміння щодо ставлення студентів до їх стилю викладання та сфер, які, можливо, потрібно вдосконалити.

– Вибір курсу – студенти часто використовують даний портал для вибору навчальних курсів. Логічно, що студенти будуть надавати пріоритет саме тим курсам, які проводять викладачі з найвищим рейтингом та утримуватись від занять у викладачів, які регулярно отримують низькі оцінки.

Rate My Professors містить понад 19 млн. оцінок для понад 1,7 мільйона викладачів від студентів з усього світу. Оцінки отримують викладачі 7500 закладів у США, Канаді, Англії, Шотландії та Уельсі. На сторінці кожного закладу доступні рейтинги найкращих викладачів, а також середнє значення для закладу та позиція в порівнянні з іншими закладами міста або країни.

Пошук на сайті доступний як за прізвищем викладача, так і за навчальним закладом. Існує можливість перегляду інших оцінок та відправки власного коментаря з оцінкою. Портал дозволяє приховувати особу автора відгуку, зберігаючи повну анонімність. Відгуки на сайті можна категоризувати за інформацією про бажання повторно записуватись на курси конкретного викладача або рівень їх складності. Доступна також категорія, що ідентифікує методологію навчання, а також архів з відгуками колишніх студентів.

2. RateMyProfessor.io – портал, що містить оцінки мільйонів студентів з міжнародних навчальних закладів. Користувачі можуть висловлювати свої думки, залишаючи коментарі до будь-якого матеріалу на платформі та

оцінюючи його за 5-бальною шкалою. Окрім оцінювання студенти можуть обмінюватися навчальними матеріалами та науковими роботами. Сприяння двосторонньому спілкуванню між студентами та викладачами на порталі дозволяє звести до мінімуму потенційні труднощі у стосунках, забезпечуючи узгодженість їх цілей в навчальному процесі.

3. Rate My Teachers – портал з аналогічним функціоналом, головним чином зосереджений на отриманні коментарів щодо вчителів початкової та середньої школи. Після реєстрації учні отримують вичерпний список вчителів, виконавши пошук за конкретною школою або прізвищем. Обравши вчителя, користувачі можуть прочитати відгуки або додати свої оцінки. Портал працює для шкіл у таких країнах, як США, Великобританія, Канада, Ірландія, Австралія та Нова Зеландія. У випадку, якщо школа відсутня в списку, існує можливість її додати вручну. Оцінки виставляються за 5-бальною шкалою. Рейтингова система даного порталу формується на основі оцінок за такими параметрами, як корисність занять, зрозумілість, володіння предметом, легкість подачі інформації, ефективність, чесність, емпатія, повага, рішучість, тощо. Враховуючи те, що цільовою аудиторією порталу є учні шкіл, питання в анкеті є виглядають наступним чином:

– Чи доступний вчитель для додаткової допомоги?

– Чи швидко вчитель відповідає на електронні листи?

– Чи розміщує вчитель довідкові матеріали в Інтернеті?

– Чи інформує вчитель учнів про те, як підраховуються їх загальні оцінки?

– Скільки домашніх завдань задає вчитель?

– Чи багато потрібно писати на уроках цього вчителя?

– Чи дозволяє вчитель використовувати технологічні прилади?

– Як багато матеріалу треба запам’ятовувати, щоб добре впоратися з курсом даного вчителя?

4. Rateyourprof.com – веб-сайт, що збирає дані про викладачів коледжів і університетів США. Користувачі мають можливість оцінювати викладачів і репетиторів на основі різноманітних показників ефективності. При пошуку за закладом портал відображає рейтинг кращих викладачів. Послуга оцінювання безкоштовна, а всі дані користувачів залишаються анонімними. Унікальною функцією є розширений пошук, що дозволяє користувачам фільтрувати результати за кількома параметрами.

5. Uloop – портал, що надає можливість студентам виставляти оцінки та публікувати коментарі стосовно професорів, викладачів і вчителів [3]. Огляди, написані колишніми студентами, які навчалися у цих професорів, дозволяють зрозуміти численні аспекти їх навчальних стратегій, манер та особистих якостей. Портал діє в межах США, при чому до Uloop включені майже всі американські навчальні заклади. Окрім оцінювання викладачів студенти можуть отримати доступ до різноманітних послуг, як, наприклад, оголошення про роботу в університетському містечку, можливість обмінюватися записами курсу, інформацією про варіанти навчання, тощо. Студенти оцінюють викладачів відповідно до певних стандартів, включаючи зрозумілість подачі інформації і загальну готовність допомогти.

6. Niche – портал трохи іншого спрямування ніж попередні веб-проєкти. Niche не дозволяє оцінювати окремих викладачів, а натомість оцінює заклади на основі різноманітних показників, включаючи загальну оцінку викладачів в якості складової. На сайті Niche представлено майже 100 000 навчальних закладів. Інформація про якість викладання формується за такими показниками:

– Відсоток студентів, які погоджуються з тим, що викладачі проводять цікаві заняття;

– Відсоток студентів, які погоджуються, що викладачі піклуються про них;

– Відсоток студентів, які погоджуються, що викладачі адекватно керують групою і контролюють навчальний процес.

7. Student Reviews – сайт, що містить базу даних із відгуків стосовно викладачів з понад 11000 університетів світу. Доступний розширений пошук за такими показниками, як ступінь і спеціальність, рівень задоволеності від занять, тощо. Окрім оглядів викладачів і рейтингів закладів в цілому, платформа містить такі інструменти, як:

– Підбір закладу;
– Матеріали, пов’язані з кар’єрою та освітою;
– Рейтинг кращих навчальних закладів країни;
– Кар’єра та спеціальності;
– Стажування.

8. Rate Your Lecturer – платформа з рейтингом викладачів для навчальних закладів Великобританії. Оцінити викладача можна за наступними параметрами: семінари, підручники, лекції, зворотній зв’язок, підтримка внутрішньої мережі, робочий час, доступність. Для оцінювання використовується 10-бальна шкала. Пошук по університету дозволяє отримати рейтинг з 5 найкращих викладачів.

Підсумовуючи, можна сказати, що правила оцінювання та коментування можуть відрізнятись в залежності від конкретного порталу. Деякі портали можуть обмежувати студентів одним рецензуванням на викладача на семестр або навчальний рік, щоб запобігти надмірним або упередженим відгукам. Також в певних випадках використовуються механізми перевірки того, що рецензенти є справжніми студентами викладача, який оцінюється.

1.5. Висновки до розділу 1

1. Рейтингова оцінка діяльності – кількісний, інтегральний показник результатів якості роботи педагогічних працівників коледжу, який формується за основними напрямами роботи викладачів: методична, організаційна,

навчально-виховна, інноваційна, а також з урахуванням дотримання виконавчої та трудової дисципліни викладачів за звітний період. В рамках формування комплексного рейтингу викладачів закладу пропонується проводити анонімне анкетування студентів. Для цього має бути створений веб-застосунок, що міститиме анкети з 20 питань, оцінка за кожне з яких враховуватиметься при формуванні підсумкового рейтингу шляхом обрахування зваженого середнього значення у відповідності до вагових коефіцієнтів для кожного параметра оцінки.

2. Виявлено, що в Україні для реалізації подібного функціоналу навчальні заклади користуються переважно опитуваннями через Google Forms, модулями для анкетування в складі LMS, або ж закритими власними розробками, що діють в межах локальної мережі університетів. Проаналізовано найбільш популярні на даний момент зарубіжні рішення, виявлено їх сильні та слабкі сторони та на основі проведеного аналізу сформовано базову структуру для власної розробки. Так, наприклад, визначено, що опитування можуть як проводитись впродовж всього навчального року (при цьому загальна оцінка викладача оновлюється щоразу після участі нового студента), так і протягом певного фіксованого часу, після якого викладач отримує підсумковий бал, що не змінюється до періоду проведення нового опитування.

3. Побудовано архітектурну модель для веб-застосунку, що описує його складові модулі, а також діаграми діяльності та активності для таких типів користувачів, як адміністратори, студенти, викладачі, керівники кафедр, керівники відділів та декани. Визначено перелік прав та доступ до функціоналу системи, що має бути наданий кожному типу користувачів. Також створено ER-діаграму для бази даних застосунку для оцінювання рейтингу викладачів, що описує зв’язки між визначеними сутностями даної предметної області та містить атрибути для кожної сутності.

РОЗДІЛ 2
ТЕХНОЛОГІЇ ДЛЯ СТВОРЕННЯ ВЕБ-ЗАСТОСУНКУ

2.1. Технології для розробки користувацького інтерфейсу застосунку

Першим етапом розробки веб-застосунку для анонімного опитування студентів є проєктування його інтерфейсу, для чого важливим кроком є вибір програмного забезпечення – графічного редактора та системи прототипування.

В загальному розумінні програмне забезпечення для розробки інтерфейсу користувача – це цифровий інструмент, який використовується для створення, редагування та оптимізації візуальних аспектів інтерфейсу користувача в цифрових продуктах. Основні функції такого редактора включають можливості використання бібліотек елементів дизайну, створення прототипів, моделювання адаптивності та забезпечення експорту дизайну при передачі на розробку. Вибір правильного програмного забезпечення для розробки інтерфейсу користувача – складний процес, що залежить від розуміння як основних функцій, які пропонують ці інструменти, так і того, як вони відповідають конкретним сценаріям використання та потребам дизайнера. Функціональність основного програмного забезпечення для розробки інтерфейсу користувача має покривати як стандартні випадки використання, так і специфічні вимоги для конкретного типу проєкта.

Найважливіші стандартні випадки використання графічних редакторів включають:

– Наявність засобів для обробки векторної графіки. Використання векторних елементів важливе для створення чітких і зрозумілих інтерфейсів, які

добре виглядатимуть на будь-якому пристрої, забезпечуючи можливість зміни розміру елементів без втрати якості.

– Інтерактивне створення прототипів для тестування користувачами. Існує чітка тенденція переходу до більш складних можливостей створення прототипів, включаючи високоточні симуляції та інтерактивні прототипи, які точно імітують кінцевий продукт. Ця еволюція задовольняє потреби в більш точному тестуванні та обробці відгуків користувачів, забезпечуючи функціональність і зручність дизайну перед початком розробки.

– Механізми для співпраці в режимі реального часу для командних проєктів. Спільне редагування в режимі реального часу, обмін на основі хмарних технологій та інтегровані інструменти зворотного зв’язку стають стандартним функціоналом, підкреслюючи важливість безперебійної командної роботи в процесі проєктування.

– Бібліотеки плагінів та компонентів для ефективного проєктування. Доступ до широкого спектру ресурсів, таких як піктограми, шрифти та шаблони, значно прискорює процес проєктування. Це дозволяє дизайнерам легко включати різні елементи у свої проєкти, підвищуючи візуальну привабливість інтерфейсу. Бібліотека попередньо розроблених компонентів інтерфейсу користувача, таких як кнопки або повзунки, може значно скоротити час розробки. Ці компоненти допомагають підтримувати послідовність і можуть бути налаштовані відповідно до конкретних потреб проєкту.

– Засоби побудови адаптивних дизайнів. Інструменти, що підтримують адаптивний дизайн, допомагають гарантувати, що інтерфейси добре виглядатимуть і функціонуватимуть на різних розмірах екранів та пристроях. Це надзвичайно важливо для створення доступних і зручних продуктів у сучасному світі. Функціонал для підтримки адаптивного дизайну

включає можливість створювати сітку, керувати розміром вікна з адаптивним підлаштуванням розмірів елементів, тощо.

– Параметри експорту та інтеграції. Гнучкі варіанти експорту та безперебійна інтеграція з іншими інструментами та платформами гарантують, що проєкти можна буде легко передати розробникам або використати в презентаціях.

Додаткові функції можуть включати:

– Засоби контролю версій. Контроль версій полегшує повернення до попередніх версій або дослідження інших напрямків дизайну при аналізі попередніх варіантів оформлення. Дана функція є найбільш важливою для ітеративних підходів до проєктування.

– Розширені інструменти анімації для динамічних інтерфейсів. Даний клас програмних засобів дозволяє продемонструвати інтерактивність певних елементів оформлення, як реакцію на певні тригери з боку користувача (наприклад, натискання кнопки, рух курсора, скроллінг сторінки).

– Інтеграцію з інструментами дослідження відгуків користувачів. Пряма інтеграція відгуків тестової групи у процес проєктування дає змогу дизайнерам ітерувати дизайн на основі взаємодії з реальними користувачами. Ця функція має вирішальне значення для створення інтерфейсів, які справді резонують із аудиторією.

– Пропозиції дизайну на основі штучного інтелекту для підвищення ефективності та креативності. Дана функція є досить новою і недостатньо сформованою, проте вже зараз дозволяє як мінімум швидко отримати набір стартових ідей для подальшого вдосконалення.

Вибір правильного програмного забезпечення для розробки інтерфейсу користувача вимагає ретельного розгляду цих та інших функцій. Загалом на сьогоднішній момент найбільш поширеними є наступні засоби розробки інтерфейсів [4]:

1. Figma – онлайн-редактор інтерфейсів та векторної графіки, який використовується для дизайну веб-сайтів, мобільних застосунків, настільних програм, а також інших проєктів. Figma – це інноваційний інструмент для дизайну, який працює у веб-браузері і настільних платформах, зберігаючи робочі файли в хмарному сховищі. Однією з головних переваг Figma є можливість спільної роботи над проєктами у реальному часі, що дозволяє команді дизайнерів спільно працювати над проєктом, вносячи зміни безпосередньо в онлайн-редакторі.

Figma має широкий набір інструментів для створення векторних малюнків, інтерфейсів, прототипів і дизайн-систем. Існують вбудовані модулі для роботи над інтерактивними прототипами, адаптивними макетами, бібліотеками компонентів та інспектуванням коду.

Figma має засоби інтеграції з такими інструментами, як Axure, Bubble, Flinto, Framer Web, Maze, Microsoft Teams, Mixpanel, Pendo, Principle, ProtoPie, Sprig та Zeplin. Програма має безкоштовний тарифний план з обмеженнями на використання функцій прототипування та режиму розробника, низьку кількість публічних проєктів та невелику кількість учасників команди. Вартість платної версії починається від 15 доларів за місяць.

2. Axure – програма для прототипування і дизайну інтерфейсів, що спеціалізується на UX/UI дизайні. Основною перевагою Axure є широкий функціонал та гнучкість, що дозволяє дизайнерам створювати складні інтерактивні прототипи з різноманітними елементами та варіантами інтерактивності. Так, за допомогою Axure можна реалізувати умовну логіку, наприклад, зміну інтерфейсу в залежності від певної послідовності if-then-else. Віджети Axure Rapid Prototyping (RP) в свою чергу допомагають із динамічним вмістом для додавання нового виміру до прототипування. Незважаючи на те, що Axure – це в першу чергу інструмент UX, його дизайн і можливості створювати повноцінні прототипи, близькі до реальних сайтів без кодування,

можуть використовуватись менеджерами із продуктів і дизайнерами інтерфейсу користувача.

Ключові особливості Axure включають швидке прототипування, співпрацю в реальному часі, створення функціональних прототипів, діаграми user-flow, адаптивне відображення макетів, динамічний вміст з врахуванням умовної логіки та тригерів для взаємодії з користувачем. Підтримується інтеграція з такими інструментами, як Adobe XD, Figma, Jira, Microsoft Teams, Sketch, Slack та інші.

3. Sketch – векторний графічний редактор, який спеціалізується на дизайні інтерфейсів для веб-сайтів та мобільних застосунків. Sketch має простий і зрозумілий інтерфейс, що дозволяє дизайнерам легко і швидко створювати професійні макети. Серед його функціональних можливостей – редагування векторного графічного дизайну, спільні бібліотеки, що забезпечують можливість повторного використання активів у різних дизайнах, колірні змінні, можливість використовувати додаткові плагіни, підтримка підходу pixel perfect, створення user-flow, наявність засобів A/B-тестування прототипів. Недоліком Sketch є відсутність кросплатформенності: він може бути встановленим лише на MacOS. Інтеграція включає такі інструменти, як Abstract, Avocode, Flinto, Flow, InVision, Lingo, Marvel, Maze, Origami, Plant, ProtoPie, Prott, Zeplin та інші.

4. Adobe XD – інтегроване середовище для створення макетів дизайну і прототипування інтерфейсів користувача для веб-сайтів, мобільних застосунків та інших цифрових продуктів. Adobe XD надає широкі можливості для роботи з векторною графікою, анімаціями, переходами та інтерактивними елементами. Однією з головних переваг Adobe XD є його інтеграція з іншими продуктами Adobe, такими як Photoshop та Illustrator, що дозволяє легко і швидко переносити проєкти між різними програмами.

Слід зазначити, що на даний момент XD більше не підтримується компанією Adobe, проте все ще використовується в дизайнерському

середовищі. Більшість функціоналу даного продукту є схожим на Figma, проте Adobe XD не підтримує командну розробку. Разом з тим, XD має певні функції для пришвидшення створення інтерфейсу, наприклад, можливість швидкого копіювання певних блоків з подальшою заміною тексту або зображень в кожному з них шляхом перетягування текстового файлу з переліком варіантів для кожного блоку, або папки з зображеннями у вікно редактора.

Разом з редакторами, що містять в собі повний набір інструментарію як для проєктування інтерфейсу, так і для його прототипування, слід розглянути інструменти, що використовуються лише для прототипування і можуть затосовуватись в комплексі з редакторами, які не мають вбудованих інструментів даного типу (наприклад, такими як Sketch, Photoshop або Illustrator) [5].

1. InVision – веб-сервіс та програмне забезпечення для створення прототипів інтерфейсів користувача. Основними програмами, з якими разом використовується InVision, є Photoshop і Sketch. Дана програма надає зручний інтерфейс для створення високоякісних інтерактивних прототипів, а також забезпечує можливість спільної роботи над проєктами у реальному часі. InVision має функціонал для створення інтерактивних макетів, роботи з анімаціями та переходами, а також віртуальною дошкою.

InVision відрізняється від інших програм для прототипування тим, що має вбудовані засоби для тестування та збору відгуків, що дозволяє дизайнерам отримувати цінні відомості від користувачів та вносити необхідні зміни у дизайн проєкту. Також забезпечується інтеграція з XD, Asana, Azure DevOps, Google Sheets, Jira, Microsoft Word, Sketch, Slack, Spotify, Webex, Zoom.

2. UXPin – програма для прототипування інтерфейсів та дизайну веб-застосунків, яка дозволяє створювати високоякісні інтерактивні прототипи безпосередньо у веб-браузері. Основною перевагою UXPin є його широкий набір інструментів для роботи з прототипами, включаючи бібліотеку

компонентів, можливість створення анімацій та переходів між сторінками, а також вбудовані засоби тестування та збору відгуків.

UXPin дозволяє дизайнерам будувати макети на основі існуючих прототипів. Замість візуалізації та створення дизайну з нуля, використовуються існуючі бібліотеки компонентів для розробки нових продуктів. Основні функції UXPin включають прототипування на основі компонентів, проєктування методом перетягування, інтерактивне створення прототипів, віддалену співпрацю, тегування учасників, простий процес передачі дизайну розробникам. Підтримується інтеграція з такими інструментами, як Asana, Google Sheets, Jira, Microsoft Dynamics 365, Mosaic, NPM, Salesforce, Storybook, Zendesk, тощо.

3. Zeplin – веб-сервіс для співпраці між дизайнерами та розробниками при розробці дизайну веб-застосунків. Zeplin надає інтерфейс для завантаження дизайну, спільної роботи над проєктом та генерації коду для розробки. Серед його функціональних можливостей – завантаження дизайну з редакторів векторної графіки, можливість коментування та обговорення проєкту, автоматична генерація коду для розробки.

Zeplin дозволяє організовувати екрани в логічну ієрархію, щоб спростити процес передачі дизайну команді розробників. За допомогою Zeplin маршрути користувачів при використанні застосунку можна представляти у вигляді карти. Інші функції включають керування версіями дизайну, формування user-flow, зв’язування Storybook і GitHub, автоматичне формування посібника зі стилю (styleguide). Інтеграція включає понад 10 інструментів, серед яких Adobe Photoshop, Adobe XD, Figma, Microsoft Teams, Sketch, Storybook, Trello, Visual Studio Code, Zapier та інші.

2.2. Технології для розробки фронт-енду застосунку

Інтерфейси відіграють ключову роль у створенні сучасних, адаптивних і зручних веб-застосунків. Інтерфейсна розробка на рівні фронт-енду – це

розробка GUI (графічного інтерфейсу користувача) веб-сайту за допомогою певного набору мов програмування, що дозволяє користувачам переглядати сайт і взаємодіяти з ним. Фронт-енд розробник використовує в своїй роботі такі мови та технології веб-програмування, як HTML, CSS і Javascript, а також відповідні препроцесори та фреймворки [6].

HTML (HyperText Markup Language) – стандартна мова розмітки для документів в мережі Інтернет. Браузери отримують HTML-документи з веб-сервера та транслюють їх вміст у мультимедійні веб-сторінки. HTML дозволяє описувати структуру сторінок семантично, використовуючи для цього спеціальні теги – елементи HTML-розмітки. Кожен тег може мати певний набір атрибутів, а також класи та ідентифікатори, що дозволяють уточнювати їх стильове оформлення за допомогою CSS, або ж налаштовувати взаємодію з вмістом даного тега при виконанні певних скриптів. Теги використовуються для визначення різних елементів, таких як заголовки, абзаци, списки, таблиці, форми, тощо.

HTML без особливих зусиль інтегрується з кодом, написаним іншими мовами і може легко включатися в існуючі проєкти. Програмісти, які добре володіють різними інтерфейсними та серверними мовами, успішно використовують його у своїй роботі. Однією з важливих особливостей HTML є спрощене тестування та налагодження. Навіть у випадку, якщо код містить помилки, вміст буде відображатись (при цьому браузер спробує знайти найбільш оптимальний спосіб представлення контенту).

CSS (Cascading Style Sheets) – технологія для опису оформлення веб-сторінок за допомогою каскадних таблиць стилів. CSS може використовуватись для опису представлення документа, написаного на різних мовах розмітки, наприклад HTML, dHTML або XML. Застосування каскадних таблиць дозволяє розділити презентацію та вміст, включаючи макет (тобто позиціонування елементів), кольори та типографіку. Такі популярні браузери,

як Chrome, Firefox і Edge, використовують CSS в якості власної мови інтерфейсу.

CSS використовує різні види правил, які визначають, які стилі застосовуються до конкретних елементів HTML. Правила складаються з селекторів (елементів, до яких застосовуються стилі) і опису стилів (пар, що складаються з властивості і її значення). CSS можна включати безпосередньо в HTML-документ за допомогою тегу <style>, використовувати зовнішні файли CSS або вкладати стилі безпосередньо в HTML-елементи за допомогою атрибуту style. Каскадна структура CSS дозволяє прописувати декілька правил стилів, кожне з яких може перекривати інше у випадку, якщо є більш пріоритетним (для прикладу, можна задати один стиль оформлення для всіх параграфів тексту та інший для параграфів, що мають певний клас). Подібна гнучка структура дозволяє реалізовувати оформлення макетів різного рівня складності.

CSS містить широкий спектр функцій для стилізації тексту, шрифтів, кольорів, анімації елементів, тощо. Дана технологія надає повний контроль над візуальним виглядом веб-сайту. На сьогоднішній день CSS забезпечує послідовне відтворення та сумісність у різних веб-браузерах, забезпечуючи правильне розміщення та ефективне виконання коду, а також сприяє узгодженості між платформами – як мова програмування зовнішнього вигляду, CSS забезпечує правильний синтаксис, що веде до покращення точності вмісту та читабельності, а також підтримує одноманітність відображення контенту на різних платформах.

Проте, враховуючи певну недосконаліть, притаманну синтаксису мови CSS, сторонніми розробниками було створено CSS-препроцесори – інструменти, що розширюють стандартні можливості CSS, додаючи новий функціонал, такий як змінні (можливість використовувати змінні для збереження певних часто повторюваних значень, таких як кольори, шрифти та розміри, що дозволяє легко змінювати ці значення по всьому проєкту),

вкладені правила (дана можливість полегшує структурування CSS-коду і робить його більш зрозумілим), міксіни (засіб для визнчення наборів властивостей, які можна використовувати повторно, що допомагає уникнути дублювання коду) та умови (деякі CSS-препроцесори підтримують умовні конструкції, такі як if/else).

Код, написаний за допомогою препроцесорів, компілюється в CSS і передається на обробку браузеру. Порівняння найбільш поширених на даний момент препроцесорів представлене в таблиці 2.1. Слід зазначити, що існує два дуже схожих варіанти одного з них – Sass і SCSS. Їх відрізняє лише підхід до використання дужок та крапок з комами.

Таблиця 2.1

Порівняння препроцесорів CSS

Особливості	**Sass/SCSS**	**Less**	**Stylus**
Синтаксис	Схожий на CSS, але з додатковими можливостями. Sass має два синтаксиси: SCSS (Sassy CSS) і SASS, який використовує меншу кількість дужок і крапок з комами.	Синтаксис Less схожий на звичайний CSS, але з можливістю використання змінних, вкладених правил, міксінів, функцій та інших вдосконалень, що полегшують розробку та підтримку коду.	Синтаксис Stylus є більш простим порівняно з Sass і Less. Він використовує менше дужок, що робить код більш читабельним та компактним.
Змінні	Sass підтримує змінні, що дозволяє визначати значення, які можна використовувати багаторазово	У Less є підтримка змінних, які дозволяють визначати значення, що можна використовувати у всьому коді.	Stylus підтримує змінні, які визначаються за допомогою оператора присвоєння =.
Вкладеність	Sass дозволяє вкладати один блок коду в інший, що полегшує структурування CSS	В Less підтримується вкладений синтаксис, що дозволяє вкладати правила CSS одне в одне.	В Stylus також підтримується вкладеність
Міксіни	Sass має потужну систему міксінів, що дозволяє визначати набори властивостей CSS, які можна повторно використовувати у вашому коді.	Less має підтримку міксінів, але їх функціональність не настільки розширена, як у Sass.	У Stylus також є підтримка міксінів, але вони визначаються як функції.

Продовження таблиці 2.1.

Умови та цикли	Sass дозволяє використовувати умовні конструкції, такі як if/else, що полегшує створення динамічних та адаптивних стилів.	Less не має підтримки умовних конструкцій, підтримуються лише оператори для циклів.	Stylus підтримує умовні конструкції, але в менш розширеному вигляді, ніж Sass.
Коментарі	Sass, подібно до CSS, дозволяє використовувати коментарі CSS (/* */) та коментарі в стилі Sass (//)	В Less можна використовувати як CSS коментарі CSS (/* */) , так і коментарі в стилі Less (//)	Stylus підтримує два типи коментарів: CSS коментарі (/* */) та коментарі в стилі Stylus (//)
Наслідування	Sass підтримує механізм наслідування, що дозволяє визначати загальні властивості для багатьох селекторів одночасно.	Наслідування підтримується в Less повною мірою.	У Stylus наслідування підтримується частково.
Компіляція	Компілюється в CSS	Компілюється в CSS	Компілюється в CSS
Підтримка	Широкий розповсюджений	Широкий розповсюджений	Поступово зростає

JavaScript – мова програмування, що разом із HTML і CSS є однією з основних технологій мережі Інтернет. Станом на 2024 рік 98% веб-сайтів використовують JavaScript на стороні клієнта для забезпечення інтерактивності. Більшість веб-браузерів мають спеціальні вбудовані механізми для виконання JavaScript-коду на пристроях користувачів. Основні особливості JavaScript – виконання програм на стороні клієнта (дана модель архітектури дозволяє зменшити навантаження на сервер), підтримка зовнішніх бібліотек, легка інтеграція з іншими мовами веб-програмування та акцент на безпеці (програми на JavaScript потребують шифрування та ретельних перевірок безпеки завдяки виконанню на стороні клієнта. Дотримання надійних заходів безпеки має важливе значення для забезпечення захисту даних і цілісності системи).

Основна ідея JavaScript полягає в тому, щоб зробити веб-сторінки більш інтерактивними, динамічними та здатними реагувати на дії користувачів.

JavaScript використовується для реалізації таких функцій, як валідація форм, анімація, взаємодія з сервером без перезавантаження сторінки (AJAX), створення ігор, розробка веб-застосунків, тощо. JavaScript може бути використаний безпосередньо в HTML-документі, або збережений в окремому файлі і підключений до HTML-сторінки за допомогою тегу <script>.

Для мови JavaScript також широко використовується бібліотека jQuery. Незважаючи на те, що jQuery не є повноцінним фреймворком, вона залишається широко використовуваною завдяки своїй легкій природі та сумісності з різними браузерами. jQuery спрощує маніпулювання DOM, обробку подій і взаємодію з AJAX, що робить її оптимальним вибором для розробників, які прагнуть покращити застарілі кодові бази або швидко додати до веб-сторінок інтерактивність. На сьогоднішній день jQuery використовують 77% із 10 мільйонів найпопулярніших веб-сайтів. JQuery підтримують такі браузери, як Firefox, Chrome, Safari, та Edge [7].

Також окремо слід виділити TypeScript – розроблену Microsoft мову програмування, що являє собою строгу синтаксичну надмножину для JavaScript (це означає, що будь-який код JavaScript буде вважатись дійсним кодом TypeScript). Однією з головних переваг використання мови TypeScript є те, що вона може допомогти розробникам виявити помилки під час компіляції, а не під час виконання. TypeScript містить такі функції, як класи, інтерфейси та модулі, які можуть полегшити написання об’єктно-орієнтованого коду та організацію кодової бази. TypeScript підтримує JSX – синтаксис, який використовується для вбудовування елементів HTML у код JavaScript та часто використовується в таких фреймворках, як React. TypeScript є популярним вибором для створення великомасштабних програм, і часто використовується разом з іншими фреймворками та бібліотеками, такими як Angular, React і Vue.js. Його також можна використовувати з різноманітними інструментами збірки та збирачами модулів, такими як Webpack і Rollup [8].

JavaScript-фреймворки – це набори готових компонентів, бібліотек і інструментів, які допомагають розробникам швидше створювати складні веб-застосунки і сайти за допомогою JavaScript. Фреймворки надають набір готових компонентів, таких як кнопки, форми, таблиці, модальні вікна тощо, які можна легко і швидко використовувати для побудови інтерфейсу користувача. Більшість фреймворків передбачають побудову застосунку за певною структурною моделлю, що полегшує його організацію та підтримку. Також активно використовуються вбудовані засоби для маршрутизації, що дозволяють розробникам визначати, які компоненти повинні відображатися на сторінці в залежності від URL.

Багато фреймворків мають вбудовані інструменти для здійснення HTTP-запитів до сервера, що дозволяє отримувати або відправляти дані з API сервера без перезавантаження сторінки. Також більшість фреймворків зазвичай надають засоби для автоматизованого тестування коду, що допомагає забезпечити його якість та стабільність. До найпопулярніших JavaScript-фреймворків на сьогоднішній день відносяться Angular, React, Vue.js, Ember.js, Svelte та інші. Кожен з цих фреймворків має свої особливості і переваги, і вибір залежить від конкретних потреб проєкту та особистого вподобання розробників. Огляд Javascript-фреймворків представлено в додатку А.

Вибір правильного набору технологій та інтерфейсного фреймворку має вирішальне значення для веб-розробників, оскільки безпосередньо впливає на ефективність та успіх їх проєктів. Кожен фреймворк, перерахований вище, має унікальні сильні сторони, задовольняючи різним вимогами проєкту та перевагам розробників.

2.3. Технології, що застосовуються при розробці бек-енду проєкту

Бек-енд (backend development) служить проміжним рівнем між інтерфейсом на стороні клієнта та рівнем даних. Розробка бек-енду проводиться на стороні сервера та переважно полягає у вирішенні питань

обробки, зберігання і отримання даних, організації логіки серверної архітектури, інтеграції баз даних і розробки API та веб-сервісів для забезпечення належного функціонування клієнтської сторони веб-сайту або застосунку. Серверні технології, з якими працюють бек-енд розробники, включають серверні мови програмування, фреймворки та бази даних. Вибір серверної мови має безпосередній вплив на швидкість виконання проєкту та якість веб- або мобільного застосунку.

Бек-енд розробка гарантує синхронізацію внутрішньої частини веб-сайту з фронт-ендом під час обробки запитів, точної та ефективної модифікації даних і надсилання відповідей до клієнтського застосунку. Основні відмінності фронт-енду від бек-енду полягають в наступному: фронт-енд забезпечує правильне відображення клієнтської сторони сайту (візуального дизайну та макету), зосереджується на створенні інтерфейсу користувача та його UX, відповідає за дотримання питань доступності і зручності використання, а також забезпечення сумісності при використанні різних браузерів та пристроїв. В свою чергу бек-енд розробка має справу з базою даних на стороні сервера веб-сайту або програми, зосереджена переважно на створенні логіки та функціональності серверної частини, відповідає за обробку, зберігання та керування даними і включає роботу з API для підключення різних частин веб-сайту або програми, а також забезпечує процеси захисту даних, тестування та налагодження для дотримання належного рівня функціонування та безпеки.

Серед засобів та технологій, що використовуються при написанні бек-енду веб-сайтів найбільш поширеними є наступні мови та фреймворки, представлені додатку Б.

Як було зазначено в таблиці, для програмування на стороні бек-енду можуть застосуватись як спеціалізовані мови для веб-розробки, так і мови загального призначення, що мають адаптовані для вебу фреймворки. Разом з тим, фреймворки можуть використовуватись і в спеціалізованих для веб-

розробки мовах з метою оптимізації певних процесів та автоматизації виконання рутинних задач. Використання серверних фреймворків надає розробникам наступні переваги [9]:

– можливість застосовувати готові компоненти та інструменти для прискорення робочих процесів, забезпечуючи ефективну обробку та скорочуючи час і витрати на розробку;

– можливість масштабування, що має вирішальне значення для обробки великих обсягів трафіку та запитів користувачів;

– наявність вбудованих функцій безпеки для захисту від SQL-ін’єкцій, міжсайтових сценаріїв (XSS) та інших типів вразливостей;

– послідовна кодова база, що полегшує розробку завдяки зменшенню помилок і підвищенню продуктивності;

– вбудована інтеграція з іншими інструментами та службами, що забезпечує просте підключення до баз даних, платіжних шлюзів, служб електронної пошти та інших сторонніх служб;

– спільнота розробників, яка надає повну підтримку, гарантує виправлення помилок і публікує регулярні оновлення, щоб зберегти актуальність фреймворку з часом.

Разом з мовою серверного програмування та фреймворком для серверного програмування має використовуватись певна мова запитів до бази даних і сумісна з нею система управління базами даних. Однією з найбільш поширених на даний момент мов є SQL. Набір функцій і синтаксис SQL забезпечують оптимальну розробку серверної частини та допомагають працювати з великими наборами даних. При роботі над бек-ендом можуть бути використані такі системи управління базами даних, як PostgreSQL, SQLite, MySQL, тощо. Суть використання реляційних СУБД полягає в тому, що [10]:

– реляційна структура ідеально підходить для забезпечення масштабованості та обробки великих обсягів даних і трафіку;

– дані залишаються послідовними, точними та безпомилковими – структура реляційних баз унеможливлює проблеми з цілісністю та дублюванням;

– узгодженість синтаксису, легке з’єднання з іншими реляційними базами даних;

– вбудовані функції безпеки, такі як шифрування даних, автентифікація користувача та контроль доступу;

– висока продуктивність завдяки здатності швидко обробляти складні запити.

Кожна серверна мова має свої плюси та мінуси. Зрештою, оптимально підібрана мова має відповідати специфікаціям проєкту, часовому графіку, розподілу витрат і внутрішнім навичкам розробників.

2.4. Огляд середовища розробки

IDE (Integrated Development Environment) – це інтегроване середовище розробки, яке надає необхідні інструменти для написання, відлагодження і тестування програмного забезпечення в одній зручній оболонці. Основні компоненти IDE загального призначення, як правило, включають в себе [11]:

– Текстовий редактор, що дозволяє розробникам писати код на різних мовах програмування з підсвіткою та перевіркою синтаксису;

– Відлагоджувач (Debugger) – засіб, який допомагає виявляти та виправляти помилки в програмному коді шляхом поетапного виконання програми і аналізу її стану;

– Компілятор або інтерпретатор, що використовується для перетворення вихідного коду виконуваної програми в зрозумілий для комп’ютера формат;

– Систему керування версіями (Version Control System), яка дозволяє вести історію змін в коді та спільно працювати над проєктом з іншими розробниками;

– Вбудовані засоби підтримки веб-розробки: автодоповнення коду, підказки щодо синтаксису (інтегровані інструменти для аналізу коду, перевірки орфографії), засоби для підтримки роботи з базами даних, тощо.

Разом з тим специфіка веб-розробки ставить певні додаткові вимоги до IDE, що покликані забезпечити продуктивність і зручність у роботі:

– Підтримку мов веб-програмування: IDE має підтримує всі основні мови програмування для веб-розробки, такі як HTML, CSS, JavaScript, PHP, Python, а також популярні фреймворки і бібліотеки, такі як React, Angular, Vue.js, тощо.

– Автодоповнення коду не лише для основного набору мов, а також і для широковживаних препроцесорів та засобів скорочення коду: можливість автоматичного доповнення, формування підказок щодо синтаксису, методів, властивостей, функцій та інших елементів коду.

– Вбудована підтримка версійного контролю: інтеграція з системами контролю версій, такими як Git, що може спростити роботу з кодом у команді та дозволятиме відстежувати зміни.

– Можливість відлагодження коду безпосередньо з IDE, використовуючи вбудовані інструменти.

– Візуальні редактори і дизайнери: інтеграція з візуальними редакторами для роботи з HTML та CSS дозволить швидше створювати і відлагоджувати веб-сторінки.

– Підтримка плагінів і розширень: можливість розширення функціональності IDE за допомогою плагінів і розширень, що дозволить налаштувати середовище з урахуванням потреб розробника.

– Інтеграція з сервером: можливість запускати та оновлювати (перезавантажувати) тестовий сервер безпосередньо з IDE для швидкої перевірки змін.

– Підтримка мобільної розробки: у випадку, якщо потрібно розробляти не тільки веб-сторінки, але й мобільні застосунки, важливо, щоб IDE підтримувала мобільні платформи, такі як Android та iOS.

– Кросплатформеність: підтримка різних операційних систем, таких як Windows, macOS та Linux.

В процесі веб-розробки можуть бути поєднані кілька IDE для різних етапів роботи, це залежить від вимог проєкту та можливостей IDE. До найбільш поширених на даний момент середовищ розробки відносять наступні [12]:

1. Visual Studio Code (VS Code) – розроблена Microsoft програма, що містить інструменти та розширення для 36 мов програмування, включаючи C#, C++, PHP і Python, а також має підтримку JavaScript, TypeScript і NodeJS. VS Code може бути інстальована в таких операційних системах, як Windows, Mac і Linux. Однією з основних переваг даного IDE є IntelliSense – механізм надання інтелектуальних рекомендацій щодо коду за допомогою інструментів інтелектуального завершення команд (програма аналізує введення користувача і пропонує автодоповнення рядків, базуючись на синтаксисі обраної мови та ключових словах, що в ній використовуються). Інша характерна особливість, важлива для командної роботи – створення середовищ розробки в хмарі під час роботи над проєктом. Для забезпечення додаткового функціоналу можуть використовуватись плагіни. Перелік деяких функцій, доступних у Visual Studio Code:

– інтеграція з GitHub;
– автоматизовані засоби тестування;
– підсвічування синтаксису;
– імпортовані модулі та готові шаблони;
– створення програмних компонентів і керування ними.

2. IntelliJ IDEA – IDE, що переважно використовується для програмування мовою Java, проте також підтримує засоби веб-розробки, як,

наприклад, HTML, JavaScript та SQL. IntelliJ IDEA має можливість автоматично додавати зручні інструменти, що відповідають контексту робочого процесу. Наявні засоби інтелектуальної допомоги в кодуванні, аналізі потоку даних й індексації коду для надання відповідних пропозицій і доповнень. IntelliJ IDEA має наступні особливості:

– виявлення повторюваних фрагментів коду;
– підтримка Google App Engine, Grails, GWT;
– наявність інструментів розгортання та налагодження;
– автоматичне доповнення коду.

3. Atom – редактор, розроблений командою GitHub, що підтримує такі операційні системи, як Windows, Mac і Linux. Є повністю безкоштовним продуктом з відкритим кодом. Його особливістю є те, що редактор повністю створений за допомогою веб-технологій, що дозволяє додавати інструменти, теми та функції за допомогою вбудованого менеджера пакетів. Atom базується на Electron (фреймворку, який дозволяє створювати міжплатформні настільні програми за допомогою Node.js). Він підтримує HTML, CSS і JavaScript, а також C, C++, Java, Python і PHP. IDE має наступні особливості:

– підтримка Intellisense для розумного завершення рядків коду;
– інструменти для пошуку та заміни фрагментів коду;
– інтегрований менеджер пакетів;
– розділення інтерфейсу на окремі робочі простори.

4. NetBeans – IDE, що може використовуватись при роботі з JavaScript, Java, PHP, CSS, HTML5 та багатьми іншими мовами веб-розробки. Має підтримку таких операційних систем, як Windows, macOS, Linux і Solaris. Забезпечує не лише перевірку синтаксису коду, а і його семантики. NetBeans має наступні особливості:

– підтримка додаткових плагінів;
– вбудований рефакторинг коду;
– використання модулю FindBug для написання коду без помилок;

– ефективний процес управління проєктами.

5. Eclipse – IDE, що використовується в робочих проєктах Google, Netflix, Facebook, GE і Walmart. Підтримує такі операційні системи, як Windows, macOS і Linux. Eclipse – розширювана платформа, що має можливість інсталювати набір додаткових інструментів, включаючи конструктори GUI та інструменти для моделювання, побудови діаграм, тестування та звітності. Також важливою перевагою даної платформи є можливість створення власних плагінів за допомогою середовища розробки плагінів (PDE). Eclipse є безкоштовним програмним продуктом із відкритим вихідним кодом. Його особливості:

– швидкий компілятор;
– широкий вибір плагінів;
– рефакторинг коду;
– автоматична перевірка синтаксису.

6. Sublime Text – безкоштовний програмний продукт для веб-розробників, що підтримує Windows, Mac і Linux та широкий спектр мов програмування, серед яких C, HTML, Python, JavaScript і CSS. Має великий набір плагінів та модульну структуру (практично кожен функціональний елемент можна вилучити або інсталювати за потреби). Завдяки потужному API та широкій підтримці плагінів Sublime Text – одне з найбільш популярних середовищ розробки серед веб-програмістів. Основні особливості Sublime Text:

– повна кастомізація;
– репозиторій плагінів;
– підтримка використання фрагментів та макросів;
– покращене керування панелями робочого простору;
– вбудовані пропозиції при написанні коду.

7. Brackets – легкий але потужний засіб для веб-розробників із набором візуальних інструментів та підтримкою таких операційних систем, як macOS,

Windows і більшості дистрибутивів Linux. Під час роботи з препроцесорами (для спрощення проєктування) полегшується редагування файлів LESS і SCSS. Мови програмування, які можна використовувати в Brackets – це HTML, CSS, JavaScript, Perl, Python, Java та Ruby. Основні особливості редактора:

- попередній перегляд у реальному часі;
- вбудований відладчик JavaScript;
- кросплатформенність і розширюваність;
- спільна платформа для розробників;
- засоби доповнення коду.

8. Webstorm – IDE для веб-розробників, сумісне з Windows, Mac і Linux. Підтримує такі мови і фреймворки, як HTML, JavaScript, Node.js, Angular, TypeScript, CSS, React тощо. Має функціонал для завершення коду та рефакторингу при роботі з популярними фреймворками, виявляє помилки за допомогою функції аналізу якості коду. Можлива інтеграція WebStorm з такими лінтерами, як Stylelint та ESLint. Крім того, Webstorm містить вбудований HTTP-клієнт у редакторі для створення, редагування та виконання HTTP-запитів. WebStorm має наступні особливості:

- широка підтримка плагінів;
- зручні засоби навігації проєктними файлами;
- вбудований відладчик.

9. Notepad++ – популярний редактор для веб-розробників, сумісний лише з Windows. Має близько 140 плагінів для розширення функціоналу, 10 з яких стандартні. Підтримує програмування на 78 мовах. Notepad++ має наступні фукнції:

- настроюваний GUI;
- підтримка мультиперегляду та багатомовності;
- підсвічування синтаксису для мов веб-програмування (JavaScript, HTML, CSS).

2.5. Огляд криптографічних методів для анонімізації

Основними запоруками безпеки та конфіденційності веб-сайтів є шифрування та криптографія. Загалом, криптографічні засоби інформаційної безпеки – це спосіб зробити вихідні дані (повідомлення чи файли) недоступними для читання сторонніми особами, забезпечуючи доступ до них лише користувачам, що мають на це право. Для цього використовуються складні алгоритми для шифрування і дешифрування даних за допомогою ключа, наданого відправником повідомлення.

Правила або інструкції для процесу шифрування визначаються алгоритмами. Ефективність шифрування залежить від довжини ключа, функціональності та певних особливостей використовуваного алгоритму. Ключ – це випадковий рядок бітів, який використовується для шифрування та дешифрування. Кожен ключ є унікальним і чим довшим він є, тим важче його зламати. Існує два типи систем криптографічних ключів: симетричні та асиметричні. Типова довжина становить 128 і 256 біт для закритих ключів і 2048 для відкритих.

При використанні симетричних алгоритмів обидві сторони (відправник та отримувач даних) мають однаковий ключ, що використовується як для шифрування, так і для розшифрування. Передбачається, що такий ключ має залишатися секретним для забезпечення конфіденційності. На практиці безпечне розподілення ключів для забезпечення належного контролю робить симетричне шифрування непрактичним для широкого комерційного використання.

Асиметрична система ключів (також відома як система відкритих/приватних ключів) використовує два ключі. Один з них – приватний, що лишається в таємниці, тоді як інший, відкритий ключ, стає широко доступним для всіх учасників обміну інформацією. Приватний і відкритий ключі математично пов’язані між собою, тому відповідний

закритий ключ може розшифрувати лише інформацію, зашифровану за допомогою відкритого ключа.

Технологія шифрування з використанням асиметричних та симетричних ключів існує в багатьох формах, причому найбільш суттєвими відмінностями одного різновиду від іншого є саме розмір та криптографічна міцність ключа.

Узагальнену схему криптографічної системи, що забезпечує шифрування переданої інформації, показано на рис. 2.1.

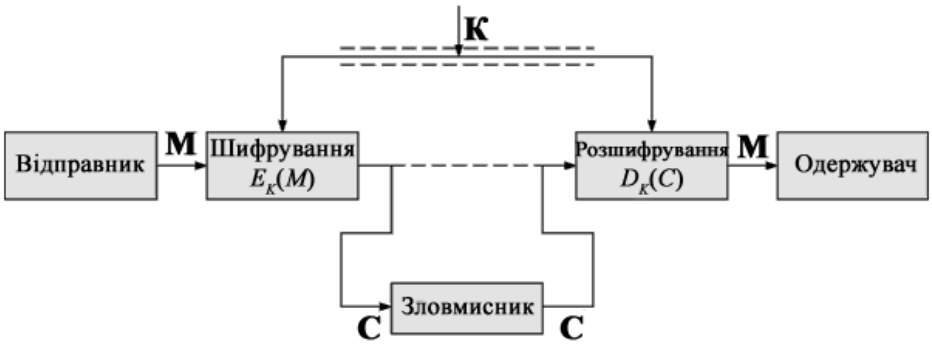

Рис. 2.1. Узагальнена схема криптосистеми

Якщо і відправник, і одержувач інформації використовують той самий ключ, система називається симетричною (інші її назви – система з одним ключем, система із секретним ключем або схема традиційного шифрування). Якщо ж відправник і одержувач використають різні ключі (один відкритий, а інший секретний), система називається асиметричною(системою із двома ключами, або схемою шифрування з відкритим ключем).

У симетричних криптосистемах ключі шифрування і розшифрування співпадають (k1= k2 = k), обидва ключі при цьому завжди є закритими. Принцип обміну при цьому виглядає наступним чином: створюється ключ, файл разом з ключем пропускається через програму шифрування, а отриманий результат пересилається адресатові [13]. Сам ключ при цьому також пересилається окремо, використовуючи інший (захищений або дуже надійний) канал зв’язку (рис. 2.2).

Рис. 2.2. Приклад симетричного шифрування

Симетричному шифруванню притаманний один недолік – складність роботи з секретними ключами при наявності множини віддалених абонентів. Для роботи з кожним абонентом необхідно створювати і зберігати окремий секретний ключ, а також передавати його віддаленим абонентам.

У несиметричних криптосистемах використовуються два ключі: відкритий для шифрування і закритий для розшифрування, причому не існує алгоритмів прийнятної обчислювальної складності для визначення секретного ключа за відомим відкритим ключем. В основі застосування систем з відкритим ключем лежить використання необоротних або односторонніх функцій. У криптографії використовуються односторонні функції, що мають так званий «потаємний хід». Їх називають також односторонніми функціями з секретом. Сутність методів шифрування з несиметричним ключем полягає у виконанні наступних процедур (рисунок 2.3): абонент А генерує пару взаємопов’язаних ключів, секретний ключ він залишає у себе, а відкритий публікує, поширюючи між своїми потенційними кореспондентами. Всім абонентам при цьому також відомий алгоритм шифрування. У випадку, якщо абоненту В необхідно надіслати зашифроване повідомлення абоненту А, він зашифровує повідомлення за допомогою відкритого ключа абонента А і передає йому шифртекст. Абонент А за допомогою секретного ключа розшифровує отримане повідомлення. Таким чином, при шифруванні з використанням відкритого ключа немає необхідності в передачі секретного

ключа між взаємодіючими суб'єктами, що істотно спрощує організацію криптозахисту переданої інформації.

Проте асиметричним шифрам притаманні і недоліки. Для всіх відомих методів шифрування з відкритим ключем математично строго не доведено відсутність інших методів криптоаналізу крім рішення NP-повної задачі (завдання повного перебору ключів). Якщо з'являться методи ефективного вирішення таких завдань, криптосистеми подібного типу будуть дискредитовані. Іншим істотним недоліком алгоритмів асиметричних шифрів є їх значна обчислювальна складність в порівнянні з симетричними.

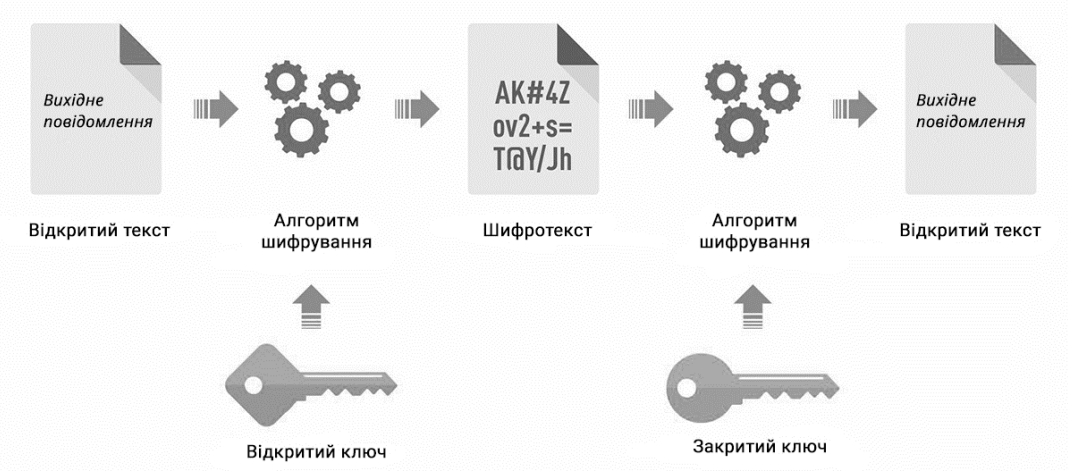

Рис. 2.3. Приклад асиметричного шифрування

До найбільш поширених алгоритмів шифрування, які можна використовувати в веб-системах, можна віднести наступні [14]:

1. Triple DES – алгоритм, розроблений в якості заміни для оригінального алгоритму стандарту шифрування даних (DES). Triple DES використовує три окремі ключі по 56 біт кожен. Загальна довжина ключа становить 168 біт, але експерти стверджують, що значення міцності ключа в 112 біт є більш точним. Свого часу Triple DES вважався галузевим стандартом і одним з найбільш поширених симетричних алгоритмів, проте наразі Triple DES поступово виходить з вжитку, поступаючись алгоритму Advanced Encryption Standard (AES).

2. Удосконалений стандарт шифрування (AES) – це стандартизований алгоритм, прийнятий урядом США, Національним Інститутом Стандартів і Технологій США (NIST) та чисельними світовими організаціями. AES також використовується в програмному забезпеченні Arcserve Unified Data Protection (UDP). Незважаючи на високу ефективність у 128-бітній формі, AES також використовує ключі в 192 та 256 бітів для більш надійного шифрування. AES вважається несприйнятливим до всіх атак, за винятком перебору грубої сили, тобто методу перебору всіх можливих комбінацій у 128, 192 або 256-бітному шифрі.

3. RSA (Rivest-Shamir-Adleman) – алгоритм шифрування з відкритим ключем. Стандарт для шифрування даних, що надсилаються через Інтернет, а також один із методів, які використовуються в програмах PGP і GPG. На відміну від Triple DES, RSA є асиметричним, оскільки використовує пару з відкритого та закритого ключів.

4. Blowfish – алгоритм, створений в якості заміни для DES. Являє собою симетричний шифр, що розбиває повідомлення на блоки по 64 біти кожен та шифрує їх окремо. Blowfish відомий своєю надзвичайною швидкістю та загальною ефективністю. Завдяки відкритому доступу Blowfish поширений серед різних категорій програмного забезпечення, починаючи від платформ електронної комерції для забезпечення платежів і закінчуючи інструментами керування паролями. Це один із найбільш гнучких доступних серед методів шифрування.

5. Twofish – симетричний алгоритм, створений Брюсом Шнайером, автором Blowfish. Ключі, які використовуються в Twofish, можуть мати довжину до 256 біт. Twofish є одним із найшвидших у своєму роді та ідеально підходить для використання в апаратних і програмних середовищах. Як і Blowfish, Twofish є у вільному доступі.

6. ECDHE (Elliptic Curve Diffie-Hellman Ephemeral) – асиметричний алгоритм обміну ключами, який використовується для встановлення

захищеного з’єднання TLS. Він забезпечує конфіденційність і цілісність даних під час їх передачі через мережу.

Зважаючи на те, що кібератаки постійно еволюціонують, фахівці із інформаційної безпеки змушені постійно розробляти нові схеми та методи захисту від них. На даний момент NIST анонсовано чотири нові стандартизовані алгоритми шифрування, три з яких, як очікується, будуть готові у 2024 році, а інший – через декілька років. Дані чотири алгоритми були відібрані в 2016 році на основі 69 варіантів, поданих експертами з криптографії з десятків країн. Після кількох раундів відкритого та прозорого оцінювання було обрано наступні [15]:

- CRYSTALS-Kyber (FIPS 203) – алгоритм, розроблений для загальних цілей шифрування, таких як створення веб-сайтів;
- CRYSTALS-Dilithium (FIPS 204) – призначений для захисту цифрових підписів, що використовуються під час віддаленого підписання документів;
- SPHINCS+ (FIPS 205) – розроблений для цифрових підписів;
- FALCON – також розроблений для цифрових підписів.

Окрім використання одного з алгоритмів шифрування для збереження даних на сервері, для захисту інформації, представленої на веб-сайті, можуть використовуватись наступні принципи:

– Встановлення зашифрованих каналів зв’язку, що підтримуються такими криптографічними протоколами, як SSL/TLS та перешкоджають перехопленню і маніпулюванню даними.

– Підтримка цілісності даних за допомогою цифрових підписів. Використовуючи даний підхід, веб-сайти можуть підтвердити автентичність представлених даних, зміцнюючи довіру до цілісності онлайн-взаємодій. Для веб-програми може використовуватись підхід JWT (JSON Web Tokens).

– Хешування даних. Веб-сайти використовують алгоритми криптографічного хешування для безпечного зберігання паролів користувачів,

зменшуючи ризик несанкціонованого доступу та витоку даних. Це гарантує, що паролі залишаються недоступними навіть у разі зламу, і захищає облікові записи користувачів та їх конфіденційні дані. Окрім захисту паролів для забезпечення анонімності даних у опитуванні студентів може бути використано хешування ідентифікаторів: перетворення ідентифікаторів студентів і викладачів в хеш-значення перед збереженням їх у базі даних дозволить зберігати ідентифікатори анонімними.

Хеш-функція – це складнозворотне перетворення даних з використанням однобічної функції, реалізованої, як правило, засобами симетричного шифрування зі зв'язуванням блоків. Загальним результатом хеш-функції при цьому слугує результат шифрування останнього блоку (що залежить від усіх попередніх). Структуру типової захищеної функції хешування показано на рис. 2.4. Дана структура, що також називається ітерованою функцією хешування, властива для більшості алгоритмів хешування, актуальних на даний момент: MD5, SHA-1, SHA-2, Whirlpool і RIPEMD.

На зображенні використані наступні умовні позначення: IV – початкове значення; CV – змінна зчеплення; Y_i – i-й блок даних; f – алгоритм стиснення; L – число блоків даних; n – довжина хеш-коду; b – довжина блоку даних.

У веб-застосунку для анонімного оцінювання викладачів можуть використовуватись наступні алгоритми хешування – SHA та HMAC. SHA (Secure Hash Algorithm) – криптографічний хеш-алгоритм, що використовується для створення унікальних хеш-значень для даних. Як правило, на практиці ним користуються для перевірки цілісності даних та цифрового підпису. HMAC (Hash-based Message Authentication Code) – алгоритм аутентифікації повідомлень, який використовується для перевірки цілісності даних під час їх передачі через мережу. В певних випадках описані алгоритми можуть використовуватись разом для забезпечення конфіденційності, цілісності та аутентичності даних. Наприклад, на веб-сайті

AES може використовуватись для шифрування даних, RSA або ECDHE – для обміну ключами, а SHA та HMAC – для перевірки цілісності даних.

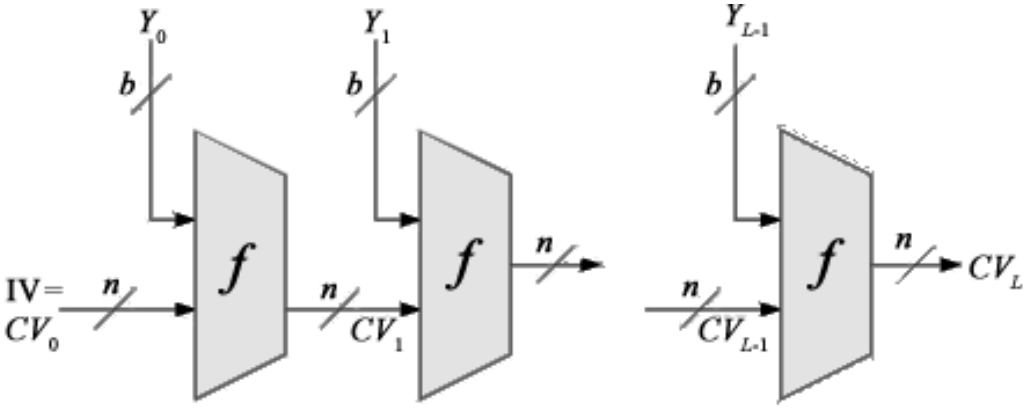

Рис. 2.4. Загальна структура алгоритму хешування

Окремо також слід розглянути протоколи нульового знання (Zero Knowledge Proofs – ZKP), що дозволяють проводити перевірку даних без розкриття конфіденційної інформації [16]. Прикладом є протокол з нульовим розголошенням. Протоколи нульового знання в загальному розумінні дозволяють перевіряючій стороні (verifier) довіряти іншій доказуючій стороні (prover) в тому, що певне твердження є правдою, без розкриття будь-якої іншої інформації, крім самого факту підтвердження цього твердження. Таким чином, окремий студент може підтвердити, що прийняв участь в опитуванні без розкриття конкретних відповідей чи будь-якої іншої інформації. Одним із відомих протоколів нульового знання є ZK-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge), що з одного боку забезпечує анонімність учасників опитування, а з іншого – перешкоджає подвійному голосуванню [17]. Для цього кожен учасник опитування отримує унікальний токен, який відповідає його ідентифікатору. Після участі в опитуванні для учасника створюється криптографічний доказ (ZK-SNARK), який підтверджує, що він володіє вірною інформацією, але не розкриває цю інформацію. Оскільки кожен учасник отримує лише один токен і він може використовувати його лише один раз.

2.6. Висновки до розділу 2

1. Визначено, що процес розробки веб-застосунку полягає в послідовному виконанні наступних процесів – аналіз вимог та формування технічного завдання на розробку, проєктування інтерфейсу, розробка та тестування програмного продукту, його впровадження. Проведено порівняльний аналіз для засобів проєктування інтерфейсу, серед яких розглянуто як універсальні редактори, що дозволяють не лише створювати макет інтерфейсу, але й формувати інтерактивний прототип, а також програми, що слугують лише для прототипування. Визначено, що найбільш оптимальним вибором для проєктування інтерфейсу на сьогоднішній день є редактор Figma.

2. Проаналізовано технології та мови програмування, які використовуються в процесі розробки веб-застосунків. Враховуючи розподіл даного процесу на програмування для фронт-енду та програмування для бек-енду, обрано наступний стек: HTML5, CSS3 та Javascript для розробки фронт-енду та PHP 8 і MySQL для програмування бек-енду. Разом з тим проведено огляд середовищ розробки та прийнято рішення використовувати Sublime Text 4, що має модульну структуру та дозволяє інсталювати потрібні для роботи віджети і плагіни, враховуючи специфіку проєкту.

3. Досліджено прийняті на даний момент підходи до криптографічного захисту інформації в системі. Проведено порівняльний аналіз найбільш поширених для вебу алгоритмів шифрування даних та прийнято рішення використовувати для інформації, що зберігатиметься в базі даних такий алгоритм, як RSA. Разом з тим розглянуто варіант застосування є ZK-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) – протоколу, що дозволить видавати студентами віртуальні токени для голосування, які будуть підтверджувати їх участь в опитуванні, не розкриваючи конфіденційну інформацію стосовно оцінок для викладачів та не допускаючи можливості повторного голосування з боку одного й того ж студента.

РОЗДІЛ 3
РОЗРОБКА ВЕБ-ЗАСТОСУНКУ

3.1. Створення бази даних застосунку

Важливим етапом розробки застосунку є проєктування структури даних – опис різних типів даних, що зберігаються в системі, а також зв'язків між сутностями даної предметної області. На рис. 3.1 показана ER-діаграма, що відображає зв'язки між об'єктами всередині системи. Створено наступні таблиці – quiz_questions (питання в опитуваннях), questions (питання), quiz (опитування), users (користувачі), quiz_question_grades (оцінки), quiz_completes (завершені опитування). Зв'язки між таблицями дозволяють реалізувати каскадну зміну (наприклад, при видаленні користувача з роллю «викладач» автоматично буде видалено опитування, створені для нього та інформацію про пройдені студентами опитування). Передбачається, що інформація про студента, який виставив конкретну оцінку при збереженні в таблиці quiz_question_grades буде хешуватись. Також для забезпечення анонімності учасників, інформація про студентів, що пройшли опитування в таблиці завершених опитувань теж буде хешованою.

На рисунку 3.2 представлено службові таблиці, необхідні для правильної роботи бек-енду проєкту. Нижче надано їх опис:

– cache – використовується для кешування даних. Застосунок зберігає кешовані дані у цій таблиці для прискорення доступу до них;

– cache_locks – використовується для блокування доступу до кешованих даних. Іноді може виникнути ситуація, коли кілька процесів намагаються одночасно оновити кешовані дані, і в такому випадку потрібно використовувати

блокування для запобігання конфліктам;

– migrations – використовується для ведення журналу міграцій бази даних. Кожен раз, коли у застосунку виконуються міграції, інформація про них зберігається в цій таблиці;

– password_reset_tokens – використовується для зберігання токенів скидання пароля. Коли користувачі надсилають запит на скидання паролю, застосунок створює токен, який вони можуть використовувати для цього процесу;

– sessions – використовується для зберігання сеансових даних користувачів. Зстосунок може використовувати базу даних для зберігання сеансових даних замість стандартного сеансового механізму PHP, що дозволяє легше керувати сесіями і зберігати їх на довший термін;

– failed_jobs – використовується для ведення журналу неуспішних задач черги. Якщо задача черги не була коректно завершена під час спроби виконання, застосунок зберігає інформацію про неї для подальшого аналізу та відновлення.

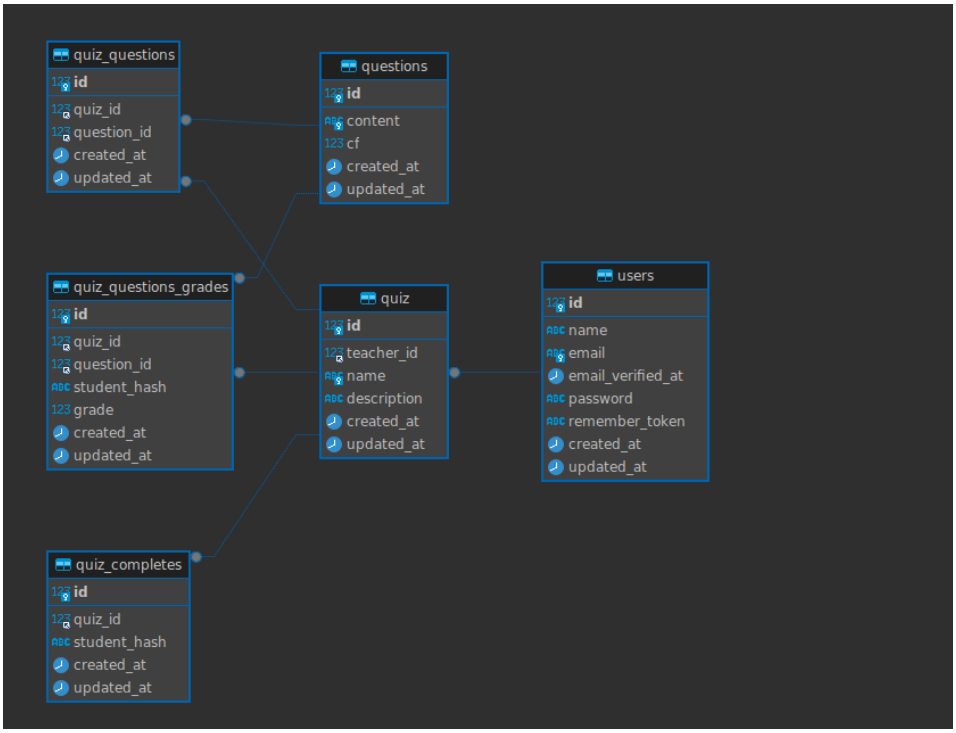

Рис. 3.1. ER-діаграма системи рейтингового оцінювання викладачів

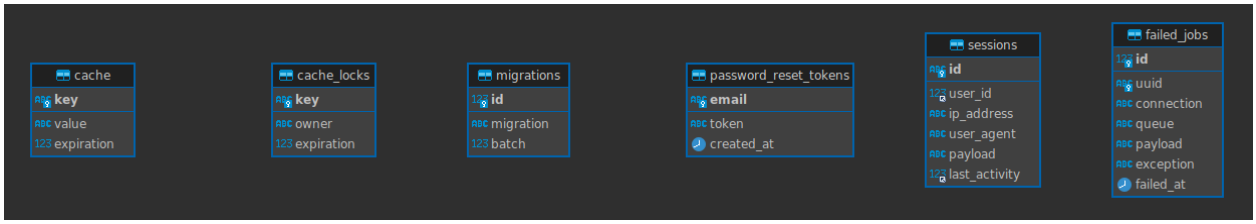

Рис. 3.2. ER-діаграма службових таблиць бази даних

Наступні таблиці в базі даних відносяться до системи управління дозволами (ACL) та черги завдань (Queue) (рис. 3.3):

– permissions: містить перелік доступних дозволів у застосунку. Дозволи використовуються для визначення, які дії або ресурси можуть бути доступні користувачам або ролям;

– roles: містить перелік ролей. Ролі зазвичай групують дозволи і призначаються користувачам, щоб визначити їх права доступу;

– role_has_permissions: відображає зв’язок між ролями та дозволами. Кожен запис у цій таблиці вказує, які дозволи призначені конкретній ролі;

– model_has_roles: відображає зв’язок між об’єктами застосунку (наприклад, користувачами або іншими моделями) та їх ролями. Використовується для призначення ролей конкретним об’єктам;

– model_has_permissions: відображає зв’язок між об’єктами застосунку та їх дозволами. Використовується для призначення дозволів конкретним об’єктам;

– jobs: містить інформацію про всі завдання, які були додані до черги. Використовується для виконання асинхронних задач у фоновому режимі;

– job_batches: містить інформацію про пакети завдань, які були додані до черги. Використовується для керування та відстеження групи задач, які пов’язані між собою.

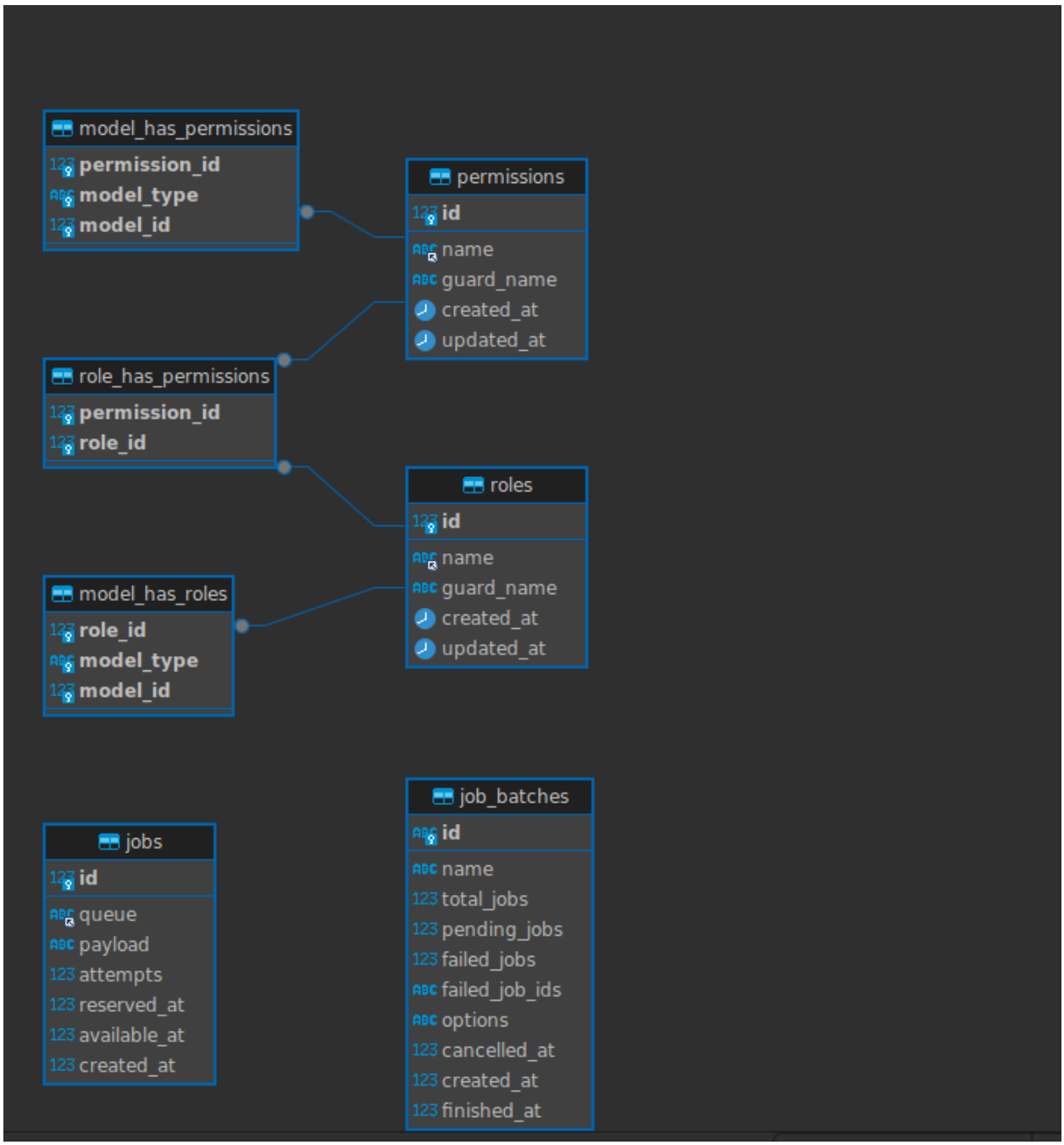

Рис. 3.3. ER-діаграма таблиць для управління дозволами та обробки асинхронних завдань

3.2. Розробка функціональності веб-застосунку

Структуру проєкту підготовано з врахуванням розгортання за допомогою Composer – інструменту управління залежностями для PHP, який дозволяє встановлювати, оновлювати та керувати бібліотеками і пакетами у проєкті. Composer передбачає опис потрібних залежностей у файлі composer.json, після

чого вони можуть бути автоматично встановлені разом з усіма їх залежностями, що значно спрощує управління сторонніми бібліотеками.

Основні налаштування для з'єднання з базою даних містяться в файлі project/.env.

Структурно файли проєкту розділено на міграції, сіди, маршрутизатори, посередники, контролери, сервіси, моделі, відображення та скрипти.

Міграції (migrations) – це спосіб автоматизувати процес зміни схеми бази даних. Вони описують, як створювати, змінювати або видаляти таблиці, поля та інші об'єкти бази даних. Кожен міграційний файл зазвичай містить пару методів: up, який виконує необхідні зміни в базі даних для встановлення нової схеми, і down, який відміняє ці зміни. Міграційні файли знаходяться за шляхом project/database/migrations.

За наповнення бази початковими даними відповідають сіди (seeders). Вони знаходяться за project/database/seeders/Setup. В проєкті було використано наступні сіди:

- сід наповнення ролей та доступів PermissionTableSeeder.php
- сід наповнення користувачів та видачі ролей UserTableSeeder.php
- сід наповнення запитань для опитування QuestionTableSeeder.php
- сід наповнення опитувань QuizTableSeeder.php

Стандартними місцями для визначення маршрутів веб-застосунку та API відповідно є файли routes/web.php та routes/api.php. Це можуть бути маршрути для відображення сторінок, обробки форм або будь-яких інших дій, які відбуваються в браузері. URL-маршрути знаходяться за шляхом: project/routes/web.php та відповідають за прийом вхідних запитів клієнта (рис. 3.4).

Рис. 3.4. Приклад коду файлу web.php

Посередники (middleware) розміщуються в папці project/app/Http/Middleware. Посередники слугують для обробки HTTP-запитів, перш ніж вони досягнуть фактичних маршрутів застосунку. Вони можуть виконувати деякі дії перед обробкою запиту (наприклад, перевірку автентифікації користувача) або після обробки запиту (наприклад, логування). Кожен клас посередника повинен реалізовувати інтерфейс Middleware, який містить один метод handle, що приймає $request (запит) та функцію-замикання $next, яка представляє наступний обробник запиту. У методі handle можна виконати потрібні дії до або після обробки запиту, а потім передати управління далі за допомогою $next($request). В даному випадку файли Middleware відповідають за обробку вхідного запиту та являються посередниками між маршрутизатором та контролером. Наприклад, за допомогою GuestLoginMiddleware.php відбувається переадресація користувача на сторінку входу у випадку коли він неавторизований в системі (тобто є гостем).

Після авторизації відбувається переадресація на головну сторінку його особистого кабінету відповідно до ролі користувача.

У директорії project/app/Http/Controllers знаходяться контролери, що відповідають за обробку HTTP-запитів та виконання бізнес-логіки застосунку. Кожен контролер – це клас, який містить методи, що відповідають за обробку різних типів запитів (GET, POST, PUT, DELETE) для певного ресурсу або функціональності. Вони можуть повертати відповідь в браузер або в API клієнта, виконувати дії з базою даних, викликати сервіси та інші об’єкти. В даному проєкті використано наступні контролери:

1. Сесійний контролер – доступний за шляхом: project/app/Http/Controllers/Auth/AuthenticatedSessionController.php. Дозволяє користувачеві авторизуватись та вийти. Має наступні основні методи:

– create – повертає відображення форми логіна;

– store – обробляє вхідний запит автентифікації. Піля успішної автентифікації повертає на попередню сторінку;

– destroy – обробляє вхідний запит користувача на вихід із застосунку.

2. Контролери адміністратора (доступні через адресу project/app/Http/Controllers/Admin/HomeController.php). Мають наступні методи:

– метод boot завантажує залежності контролера;

– метод index оброблює вхідний запит та формує відображення головної сторінки;

Крім того для адміністратора доступні контролери, що відповідають за список, створення, оновлення, видалення користувачів (UserController.php), список, створення, оновлення, видалення запитань (QuestionController.php), список, створення, оновлення, видалення опитувань (QuizController.php).

3. Контролери студента, доступні за адресою project/app/Http/Controllers/Student/HomeController.php. Основні контролери

слугують для відображення списку опитувань які студент ще не проходив. Також доступний елемент опитування QuizController.php, що містить метод show (обробляє вхідний запит та формує відображення одного елементу опитування) та метод store (обробляє вхідний запит на збереження даних результату опитування).

4. Контролери викладача – доступні з файлу project/app/Http/Controllers/Teacher/HomeController.php. Слугують для відображення рейтингу викладача.

Запити (requests) відповідають за валідацію даних вхідних запитів клієнта. Вони знаходяться за адресою project/app/Http/Requests/. В даному проєкті доступні наступні запити:

- створення нового користувача UserCreateRequest.php;
- оновлення користувача UserUpdateRequest.php;
- створення нового запитання QuestionCreateRequest.php;
- оновлення запитання QuestionUpdateRequest.php;
- створення опитування QuizCreateRequest.php;
- оновлення опитування QuizUpdateRequest.php.

Сервіси (services) знаходяться за адресою project/app/Services та містять бізнес-логіку, яка відповідає за формування даних для відображення, збереження, видалення і оновлення даних. В проєкті присутні наступні сервіси:

1. Сервіси Адміністратора:

- сервіс користувачів project/app/Services/Admin/UserService.php;
- сервіс запитань project/app/Services/Admin/QuestionService.php;
- сервіс опитувань project/app/Services/Admin/QuizService.php;
- сервіс статистики project/app/Services/Admin/StatisticService.php.

2. Сервіси Студента: список опитувань, елемент опитування, збереження результату голосування з використанням хеш-функцій та шифрування для забезпечення анонімності (project/app/Services/Student/QuizService.php).

3. Сервіс Викладача: статистика викладача (project/app/Services/Teacher/StatisticService.php).

Моделі (project/app/Models) дозволяють взаємодіють з базою даних, будувати запити на вибірку, створювати, оновлювати, видаляти записи. В проєкті доступні наступні моделі:

– модель користувача User.php;
– модель запитання Question.php;
– модель опитування Quiz.php;
– модель для окремого запитання опитування QuizQuestion.php;
– модель оцінки опитування QuizQuestionGrade.php;
– модель завершення опитування студентами QuizComplete.php;

Відображення знаходяться за адресою project/resources/views та відповідають за рендер html-файлів.

Переклади знаходяться за шляхом: project/resources/lang.

Клієнтські скрипти та стилі знаходяться за шляхом project/resources/assets. Основний клієнтський скрипт – project/resources/assets/js/app.js.

Приклади коду файлів представлено в додатках В-И.

3.3. Огляд інтерфейсу та функціональних можливостей застосунку

Вікно для входу в застосунок представлено на рис. 3.5.

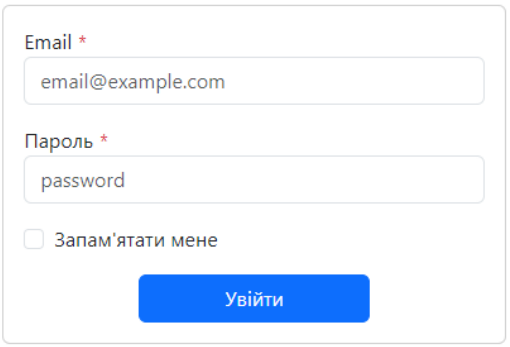

Рис. 3.5. Форма для авторизації

Після авторизації будуть доступні різні варіанти особистих кабінетів в залежності від того, до якої групи належить конкретний користувач. На рис. 3.6. представлено початковий екран адміністратора системи, що містить перелік викладачів, їх середній рейтинг, сформований на основі проведених опитувань та кількість опитаних студентів.

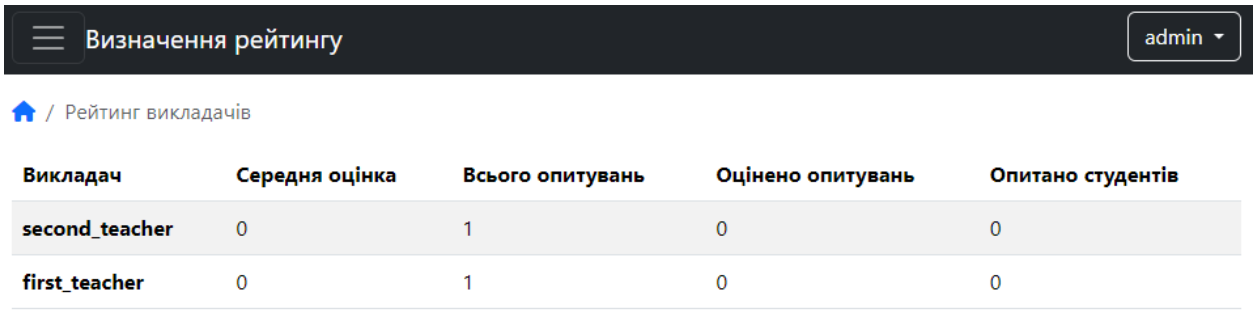

Рис. 3.6. Початковий екраз із статистикою для адміністратора

Меню адміністратора представлено на рисунку 3.7. Кожен з пунктів містить функціонал для відображення списку та редагування.

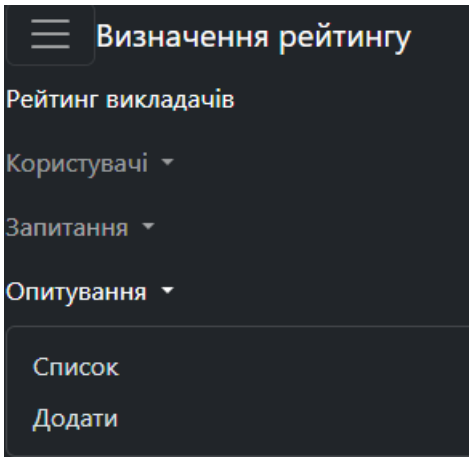

Рис. 3.7. Меню адміністратора

На рис. 3.8. відображено інтерфейс для редагування списку користувачів. В даній системі для проведення тестування створені два викладачі та два студенти.

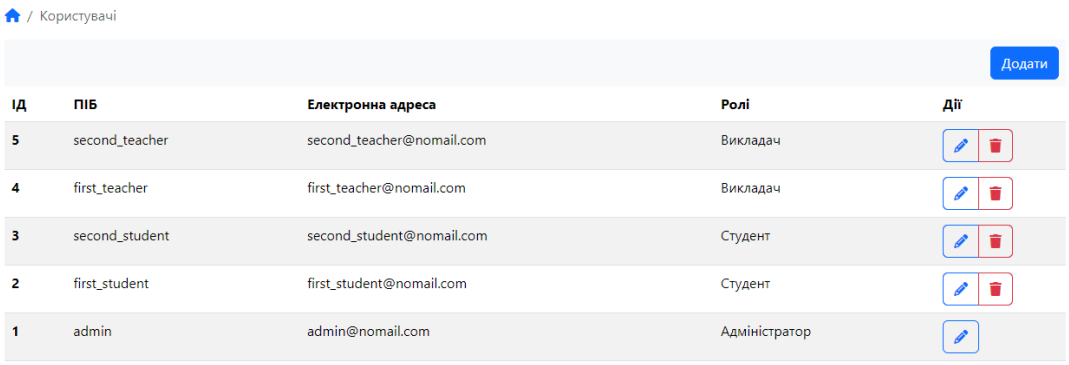

Рис. 3.8. Таблиця зі списком користувачів системи

Створювати нових користувачів може лише адміністратор (рисунок 3.9). Будь-який користувач, крім самого адміністратора, може бути видаленим із системи. Редагування передбачає зміну ПІБ, імейлу та пароля для входу.

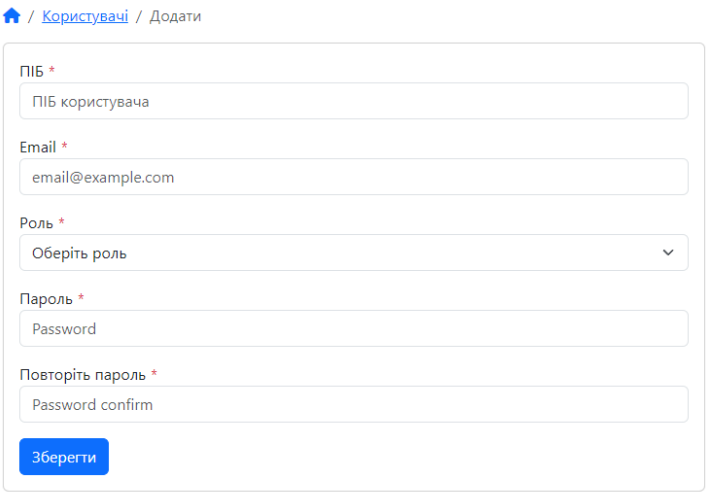

Рис. 3.9. Форма для додавання нового користувача

Адміністратор має можливість редагувати наявні запитання та додавати нові. Таблиця зі списком запитань представлена на рис. 3.10.

На рис. 3.11. зображено інтерфейс для редагування запитання. Можна як змінити текст запитання, так і виставити йому новий ваговий коефіцієнт для обрахунку зваженого середнього рейтингу.

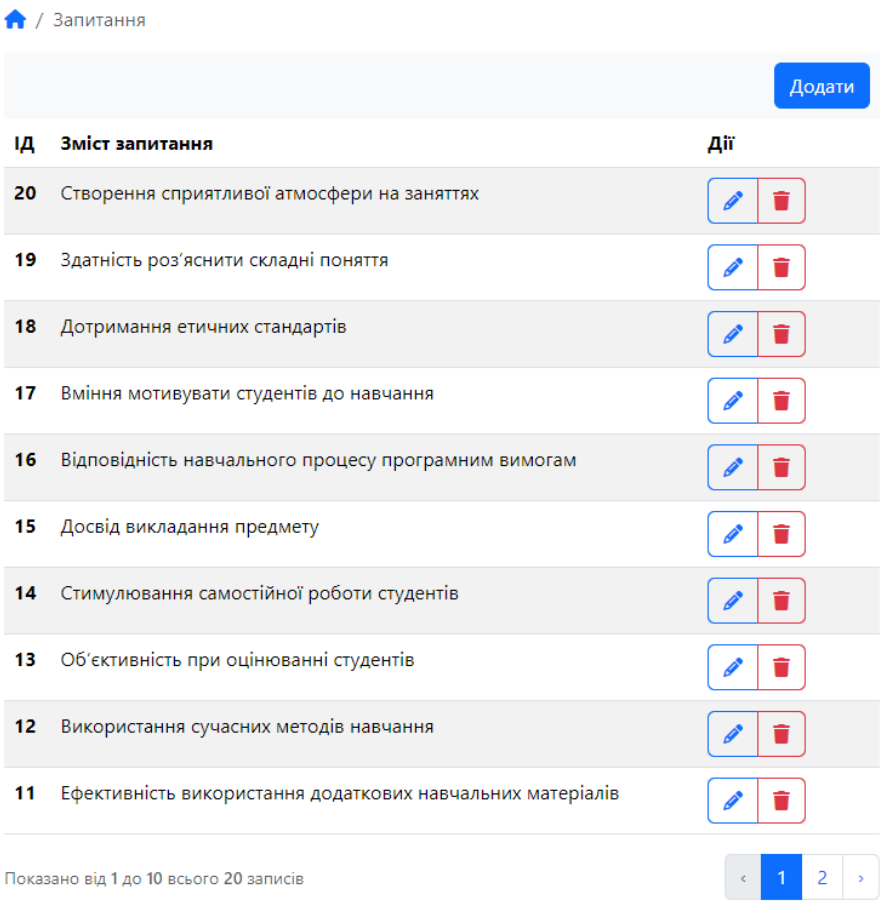

Рис. 3.10. Таблиця з переліком запитань

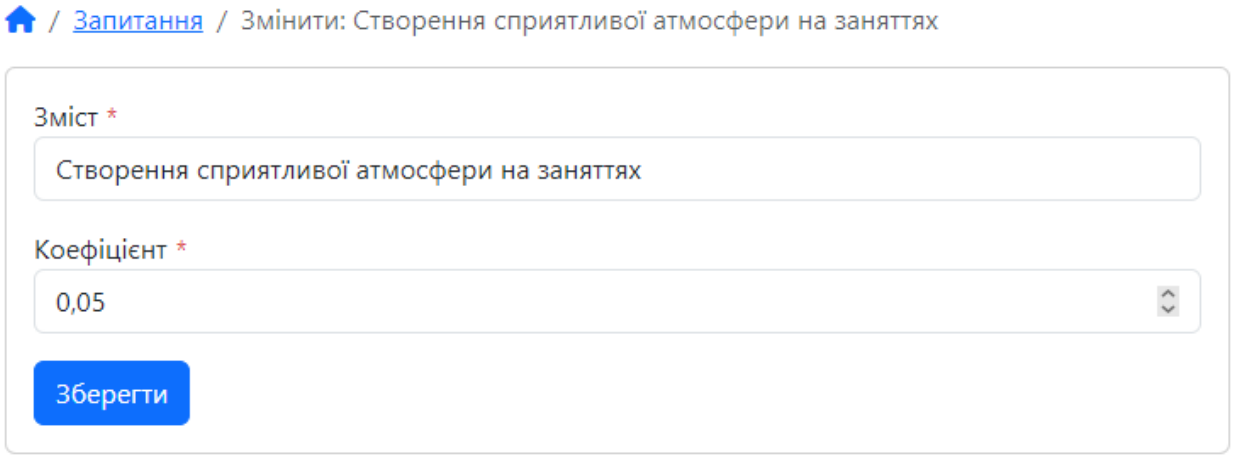

Рис. 3.11. Форма для редагування запитань

Рис. 3.12. демонструє таблицю зі списком доступних опитувань. Опитування призначаються конкретному викладачеві. Їх можна редагувати або видаляти.

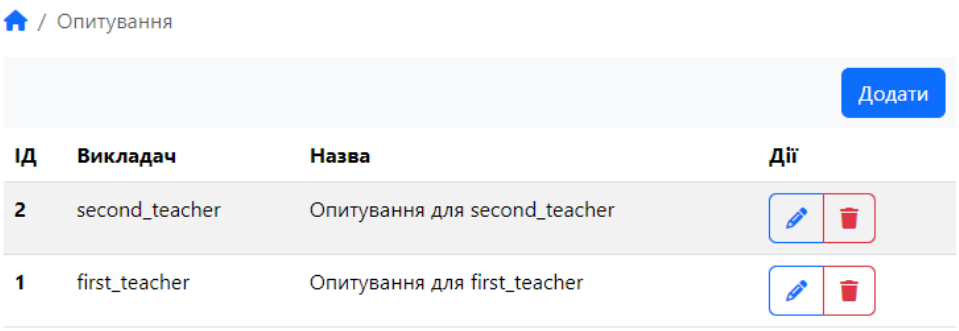

Рис. 3.12. Таблиця з переліком опитувань

При редагуванні опитування (рис. 3.13) можна визначити перелік питань, які будуть включені до опитування, призначити його відповідному викладачеві, задати опис.

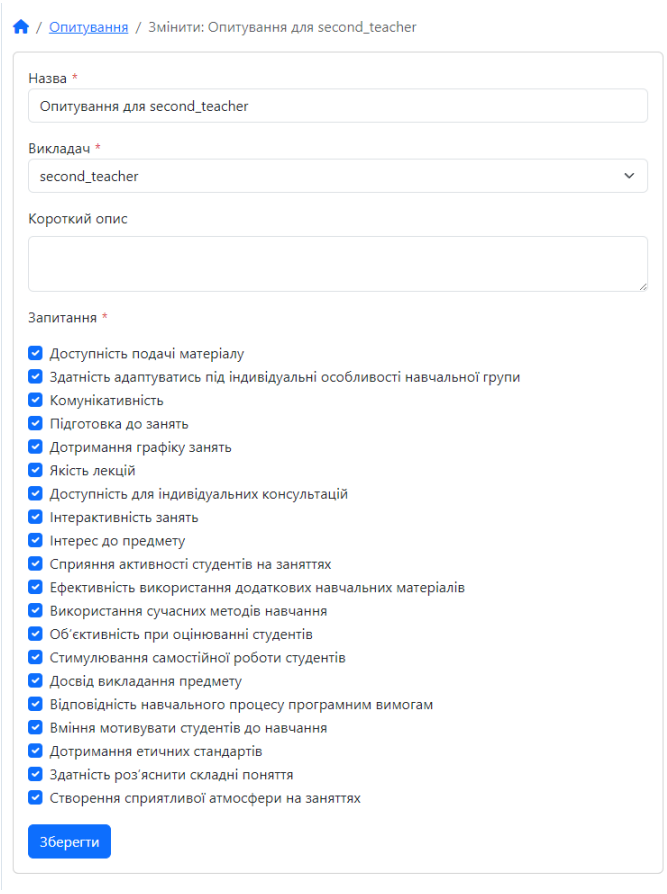

Рис. 3.13. Форма для редагування опитувань

Початковий інтерфейс студента після авторизації служить для відображення переліку доступних опитувань (рис. 3.14).

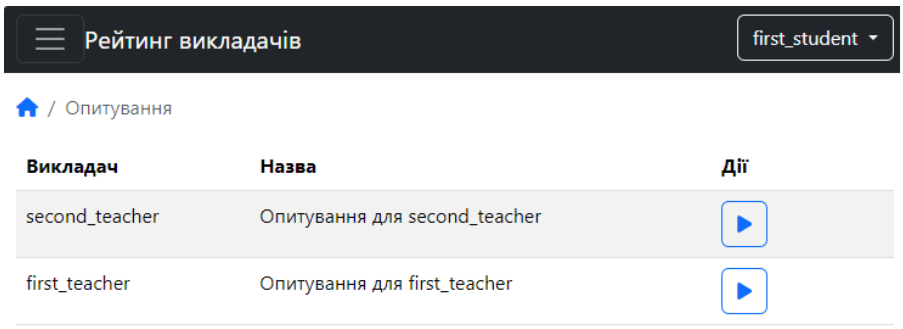

Рис. 3.14. Інтерфейс студента з переліком опитувань

При переході до опитування пропонується оцінити від 1 до 10 діяльність викладача за 20 параметрами (рис. 3.15).

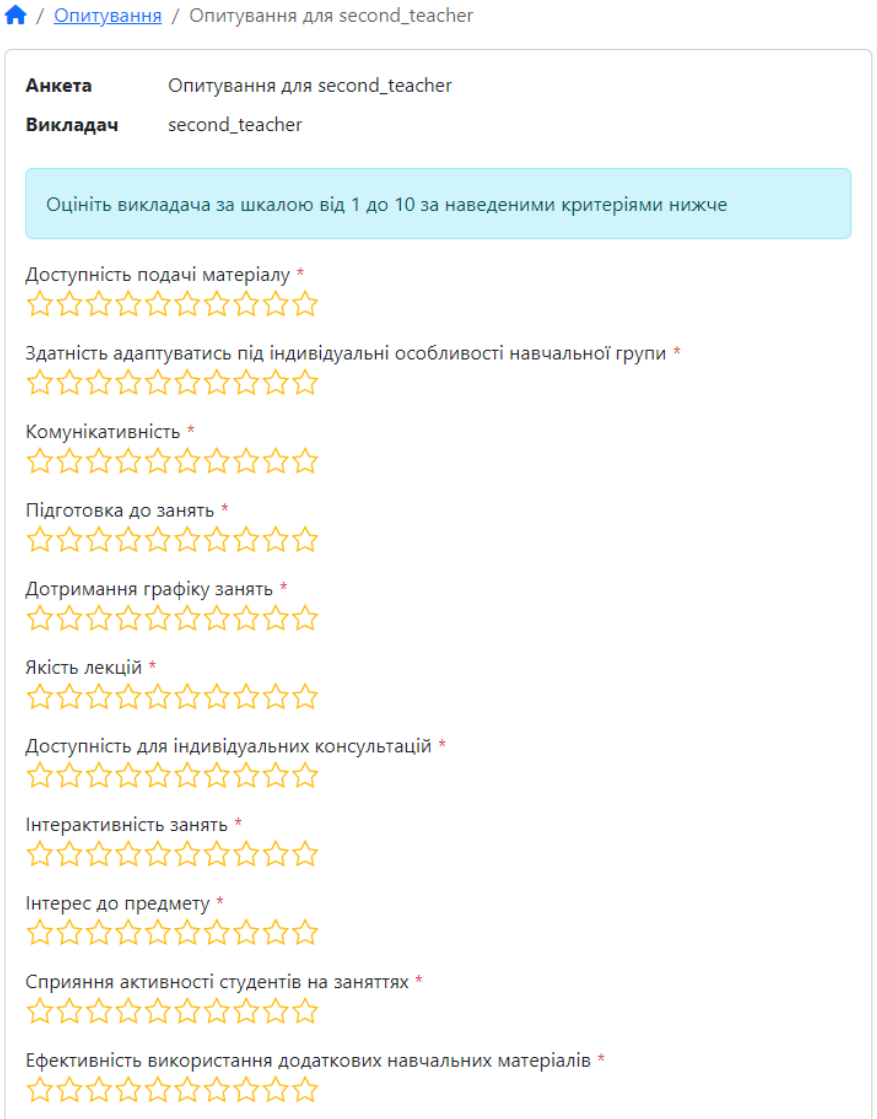

Рис. 3.15. Інтерфейс опитування

У тому випадку, якщо студент прийняв участь в опитуванні, воно зникає із списку. Приклад оновленого списку після прийняття участі в одному з двох тестових опитувань представлено на рис. 3.16.

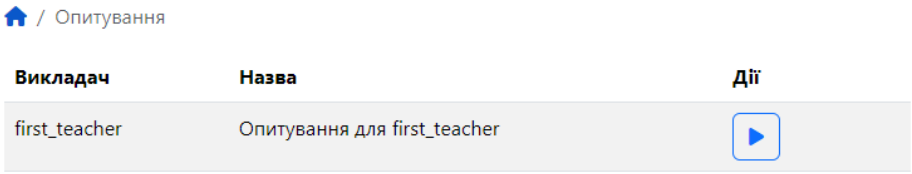

Рис. 3.16. Приклад інтерфейсу студента після проходження одного з двох доступних опитувань

Приклад інтерфейсу в разі, якщо всі доступні опитування пройдені, представлено на рис. 3.17.

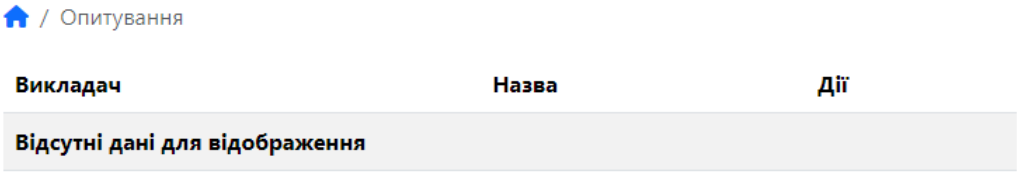

Рис. 3.17. Приклад інтерфейсу студента без доступних опитувань

Також слід зазначити, що всі 20 параметрів є обов'язковими для заповнення. В разі, якщо студент пропустить хоч один з них, він отримає помилку при спробі відправити результат (рис. 3.18).

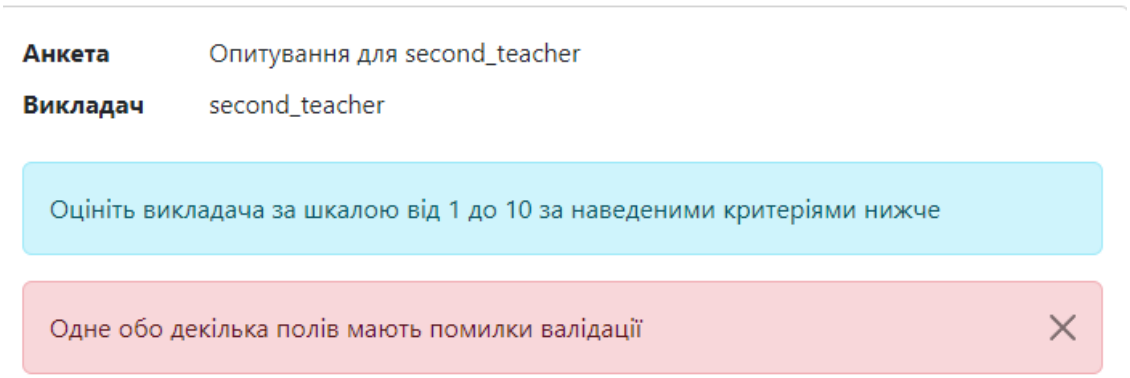

Рис. 3.18. Приклад повідомлення про помилку

Після авторизації викладача буде доступна інформація про загальний рейтинг (рис. 3.19). Разом з тим вирішено приховувати інформацію про кількість студентів, що прийняли участь в опитуванні.

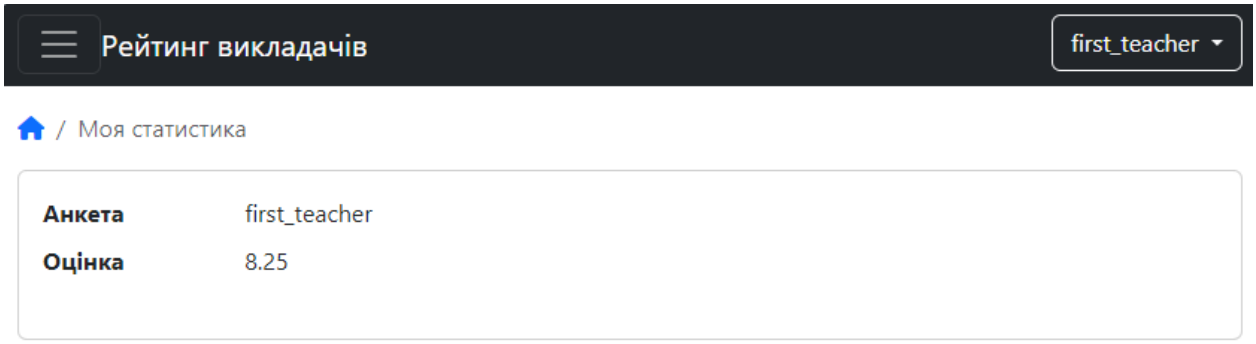

Рис. 3.19. Інтерфейс викладача

Інтерфейс адміністратора після проведення декількох тестових опитувань студентів представлено на рис. 3.20. Відображається кількість оцінених опитувань, кількість студентів, що прийняли участь, та середня оцінка, сформована на основі отриманих зважених середніх значень за підсумками кожного опитування.

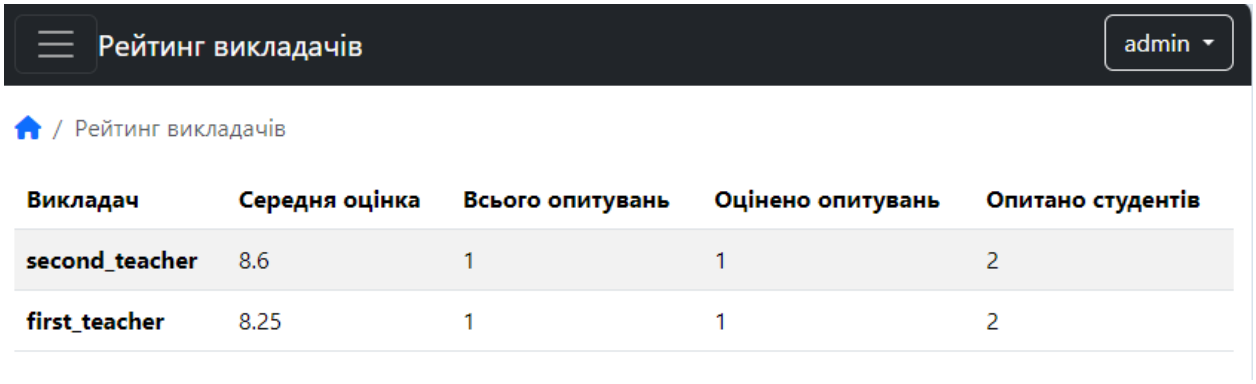

Рис. 3.20. Оновлена статистика опитувань

3.4. Висновки до розділу 3

За допомогою засобів веб-програмування PHP, mySQL, HTML, CSS, Javascript та фреймворку Laravel створено систему рейтингового оцінювання викладачів. Структура бази даних включає таблиці quiz_questions, questions, quiz, users, quiz_question_grades, quiz_completes, службові таблиці для бек-

енду проєкту, що використовуються при роботі з кешем і користувацькими сесіями, а також групу таблиць, що відносяться до до системи управління дозволами та черги завдань.

Структура проєкту застосунку поділена на наступні групи: маршрутизатори (routes), що визначають URL-шляхи та відповідні контролери або замикання для обробки запитів; контролери (controllers), що відповідають за логіку бізнес-процесів і містять методи, які обробляють запити та взаємодіють з моделями та представленнями; моделі (models), що представляють структуру та логіку роботи з даними застосунку; представлення (views), що містять HTML, CSS та JavaScript для відображення інтерфейсу користувача; міграції, що використовуються для визначення структури бази даних застосунку та сіди (seeders), що використовуються для наповнення бази даних початковими даними для розробки та тестування.

В створеному проєкті присутнє розділення на ролі користувачів. Кожна з ролей має власний особистий кабінет з врахуванням прав доступу та функціональних можливостей.

Розроблена система рейтингового оцінювання може забезпечити університетське середовище зручним та надійним інструментом для оцінки викладацької діяльності, сприяючи покращенню якості навчання та взаємовідносин між викладачами та студентами.

Враховуючи динаміку університетського середовища та потреби користувачів, можливими напрямками подальшого розвитку системи можуть бути вдосконалення інтерфейсу користувача, додавання нових функціональних можливостей або інтеграція додаткових модулів для поліпшення функціональності.

ВИСНОВКИ

Системи рейтингового оцінювання викладачів для закладів вищої освіти стають невід'ємною складовою сучасного університетського середовища, відіграючи важливу роль у підвищенні якості навчального процесу та покращенні взаємодії між викладачами та студентами. Подібні системи надають можливість оцінювати ефективність роботи викладачів з точки зору якості викладання, спілкування зі студентами, наукової діяльності та іншими аспектами їхньої професійної діяльності.

Однією з ключових переваг систем рейтингового оцінювання є їх внесок у стимулювання якісного навчального процесу. Шляхом збору та аналізу відгуків студентів щодо роботи викладачів, ці системи дозволяють ідентифікувати сильні та слабкі сторони педагогічної діяльності, а також виявляти потреби студентів у покращенні якості освіти. Такий зворотний зв'язок сприяє впровадженню корекційних заходів та розвитку викладацького потенціалу закладу. Крім того, системи рейтингового оцінювання викладачів сприяють підвищенню відкритості та прозорості управління університетом. Вони створюють можливість об'єктивної оцінки професійних досягнень викладачів, що сприяє формуванню довіри до внутрішнього середовища закладу та підвищує рівень відкритості та взаємодії між усіма учасниками освітнього процесу.

Дану роботу виконано на базі Національного авіаційного університету. Для використання в навчальному процесі запропоновано систему рейтингового оцінювання, згідно якої студенти зможуть проводити оцінку викладачів, надаючи відповіді на комплекс із 20 запитань, кожне з яких має певний ваговий коефіцієнт при обрахунку. Визначення оцінки за підсумком опитування конкретного студента проводиться за методикою зваженого середнього значення. Загальний рейтинг викладача формується як середня оцінка, отримана на основі всіх опитувань.

Розробку веб-застосунку виконано з використанням технологій веб-програмування HTML, CSS, Javascript, PHP (з фреймворком Laravel) та MySQL.

HTML (HyperText Markup Language) – це мова розмітки, яка використовується для створення структури і вмісту веб-сторінок.

CSS (Cascading Style Sheets) – мова стилізації, яка використовується для відображення вмісту HTML сторінок. Вона дозволяє змінювати вигляд елементів сторінки, такі як кольори, шрифти, розміри та розташування, тощо.

JavaScript – мова програмування, яка використовується для додавання інтерактивності до веб-сторінок. Вона дозволяє створювати різноманітні ефекти, обробляти події користувача та взаємодіяти зі сторінкою без необхідності перезавантаження.

PHP (Hypertext Preprocessor) – мова програмування, яка використовується для розробки веб-застосункыв та обробки даних на стороні сервера. PHP дозволяє створювати динамічний вміст сторінок, взаємодіяти з базою даних та забезпечувати таку функціональність, як аутентифікація користувачів та обробка форм. Laravel – веб-фреймворк для мови PHP, що базується на моделі MVC (Model-View-Controller) і надає широкий спектр функціональності для швидкої розробки без втрати якості.

MySQL – система управління базами даних, яка використовується для зберігання та управління даними веб-застосунків. Вона забезпечує можливість створення, зчитування, оновлення та видалення даних з бази даних за допомогою мови структурованих запитів SQL.

Для забезпечення анонімності студентів при участі в опитуваннях використано алгоритми шифрування та хешування даних. В таблицях бази даних інформація про студентів, що прийняли участь в опитуванні, зберігається у вигляді хешу. Таким чином, доступні лише рейтингові оцінки, але встановити, чи прийняв конкретний студент участь в опитуванні, та які саме оцінки за 20 параметрами і якому викладачеві він поставив – неможливо.

Розроблена система рейтингової оцінки викладачів Національного авіаційного університету становить вагому та актуальну ініціативу у контексті сучасного закладу вищої освіти. Інтеграція цієї системи в бізнес-процеси університету відкриває шлях до систематизації та об’єктивного оцінювання роботи викладацького складу, що є критичним у забезпеченні якості навчання та вдосконаленні педагогічного процесу.

Створена система дозволить ефективно відстежувати та аналізувати якість викладацької роботи на основі об’єктивних та структурованих даних, зібраних в ході рейтингового оцінювання. Це сприятиме впровадженню цілеспрямованих заходів для підвищення професійної компетентності та ефективності навчання.

Крім того, впровадження системи рейтингової оцінки викладачів створює фундамент для розвитку культури відкритості, взаємовідповідальності та постійного вдосконалення університетського середовища. Це сприятиме підвищенню мотивації викладацького складу, підтримці взаємовідносин між факультетами та студентською спільнотою, а також створенню сприятливих умов для забезпечення якісної освіти та академічного зростання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Національний авіаційний університет [Електронний ресурс]. Режим доступу: https://nau.edu.ua/ua/menu/universitet/pro-universitet.html (дата звернення 15.05.2024) – Про університет

2. 5 Best Portals To Rate Your Professors and Find Teacher Reviews [Electronic resource]. Access mode: https://geekflare.com/best-portals-to-rate-your-professors/ (last access 15.05.2024) – 5 Best Portals To Rate Your Professors and Find Teacher Reviews .

3. How to Use Professor Rating Sites [Electronic resource]. Access mode: https://www.fastweb.com/student-life/articles/what-to-know-about-professor-rating-websites (last access 15.05.2024) – How to Use Professor Rating Sites.

4. Figma Learn [Electronic resource]. Access mode: https://help.figma.com/hc/en-us (last access 15.05.2024) – Figma Learn.

5. 17 best prototyping tools for designers in 2024 [Electronic resource]. Access mode: https://webflow.com/blog/prototyping-tools (last access 15.05.2024) – What to look for in a prototyping tool.

6. Web Front end Developers and Technologies: a Complete Review [Electronic resource]. Access mode: https://nomadicsoft.io/web-front-end-developers-and-technologies-a-complete-review/ (last access 15.05.2024) – Best front-end technology.

7. Front-End 2024 Trends | New Front-End Technologies [Electronic resource]. Access mode: https://www.axon.dev/blog/front-end-2023-trends-new-front-end-technologies (last access 15.05.2024) – Frameworks.

8. Top Backend Technologies in 2024: choose the right one for your solution [Electronic resource]. Access mode: https://doit.software/blog/backend-technologies (last access 15.05.2024) – What are the backend technologies?.

9. Двірничук К.В., Вацек Д.О. Веб-програмування та веб-дизайн: навч. посіб. Чернівці: Чернівец. нац. ун-т ім. Ю. Федьковича, 2022. 472 с.

10. Top Backend Technologies You Must Know [Electronic resource]. Access mode: https://herovired.com/learning-hub/blogs/back-end-technology/ (last access 15.05.2024) – Features of Back-End Technologies.

11. The Best IDE For Web Development [Electronic resource]. Access mode: https://algorithmman.com/best-ide-web-development/ (last access 15.05.2024) – Advantages of Using These Web Development IDEs.

12. Top 15 Web Development IDEs for 2024: Boost Your Coding Workflow [Electronic resource]. Access mode: https://www.geeksforgeeks.org/best-ide-for-web-developers/ (last access 15.05.2024) – 10 Best IDE For Web Developers.

13. Cryptographic Algorithm [Electronic resource]. Access mode: https://www.sciencedirect.com/topics/computer-science/cryptographic-algorithm (last access 15.05.2024) – Cryptographic Algorithm.

14. 5 Common Encryption Algorithms and the Unbreakables of the Future [Electronic resource]. Access mode: https://www.arcserve.com/blog/5-common-encryption-algorithms-and-unbreakables-future (last access 15.05.2024) – Common Encryption Algorithms.

15. Basics of Cryptographic Algorithms [Electronic resource]. Access mode: https://www.geeksforgeeks.org/basics-of-cryptographic-algorithms/ (last access 15.05.2024) – Types of Cryptographic Algorithms.

16. What Is a Zero-Knowledge Proof? [Electronic resource]. Access mode: https://chain.link/education/zero-knowledge-proof-zkp (last access 15.05.2024) – What Is a Zero-Knowledge Proof?.

17. What Are ZK-Snarks? [Electronic resource]. Access mode: https://z.cash/learn/what-are-zk-snarks/ (last access 15.05.2024) – What Are ZK-Snarks.

ДОДАТКИ

Додаток А

Таблиця А.1.

Порівняльний аналіз фреймворків для веб-розробки

Особливості	Angular	React	Vue.js
Характеристика	Потужна та надійна платформа для створення програм корпоративного рівня. За допомогою Angular розробники можуть створювати адаптивні веб-застосунки з чітким розподілом задач, забезпечуючи якісні масштабованість та тестування. Присутні двостороннє прив'язування даних і функції ін'єкції залежностей.	Розроблений Facebook фреймворк, що здобув популярність завдяки своїй компонентній архітектурі та віртуальному рендерингу DOM. ReactJS дозволяє ефективно створювати масштабні, складні застосунки, сприяючи повторному використанню та зручності обслуговування.	Прогресивний інтерфейсний фреймворк,який дозволяє створювати інтерактивні інтерфейси користувача та single-page застосунки (застосунки, що працюють в рамках одного робочого вікна без перезавантаження сторінки), що робить його оптимальним вибором для малих та середніх проєктів.
Вид віртуального DOM	Використовує справжній DOM та обробляє зміни за допомогою Angular Change Detection Mechanism. Це може призвести до менш ефективної роботи у масштабних проєктах.	Використовує віртуальний DOM для ефективного відображення змін, що робить React швидким та ефективним при відображенні компонентів.	Використовує віртуальний DOM, який дозволяє оптимізувати відображення змін та зменшує кількість маніпуляцій з реальним DOM.
Компоненти	Angular має вбудовану систему компонентів, яка дозволяє організовувати застосунок у вигляді ієрархічної структури.	React базується на компонентах, що робить структуру застосунку більш організованою та керованою.	Vue.js має вбудовану систему компонентів, схожу на Angular та React.
Маршрутизація	Angular має вбудований модуль маршрутизації, який дозволяє визначати маршрути застосунку та відображати відповідні компоненти.	React не має вбудованої системи маршрутизації, але для цього можна використовувати додаткові бібліотеки, такі як React Router.	Vue.js має вбудований модуль маршрутизації.
Управління станом	У Angular можна використовувати RxJS для реактивного програмування та NgRx для керування станом.	React дозволяє використовувати стейт-менеджери, такі як Redux або MobX, для ефективного управління станом застосунку.	В Vue.js для керування станом використовується бібліотека Vuex, яка забезпечує централізоване управління станом застосунку.
Серверний рендеринг	Angular підтримує серверний рендеринг, що дозволяє прискорити завантаження сторінки та покращити індексацію пошуковими системами.	React підтримує серверний рендеринг	Vue.js підтримує серверний рендеринг
Реактивність	Angular є реактивним фреймворком, оскільки використовує RxJS та спостерігачі для слідкування за змінами стану застосунку.	React не є повністю реактивним, оскільки використовує віртуальний DOM та зміни стану, а не спостерігачі для слідкування за змінами.	Vue.js є реактивним фреймворком, оскільки використовує віртуальний DOM та спостерігачі.

Продовження таблиці А.1.

Спільнота	Спільнота Angular велика та активна, адже фреймворк існує довгий час та має велику корпоративну підтримку.	Спільнота React велика, оскільки фреймворк став де-факто стандартом у веб-розробці.	Спільнота Vue.js швидко зростає, але поки що вона менша порівняно з Angular та React.
Документація	Angular має офіційну документацію, а також багато матеріалів для вивчення на сторонніх ресурсах.	React також має офіційну документацію та матеріали на сторонніх ресурсах.	Vue.js має офіційну документацію
Швидкодія	Angular зазвичай повільніший порівняно з React та Vue.js, особливо на великих проєктах.	React дуже швидкий та ефективний завдяки використанню віртуального DOM та оптимізаціям.	Vue.js також швидкий та ефективний.
Розмір	Angular має великий розмір, особливо для малих проєктів.	React має середній розмір, що робить його досить легким для використання.	Vue.js має малий розмір.

Додаток Б

Таблиця Б.1

Технології розробки бек-енду веб-застосунку

Мова програмування	Характеристика	Фреймворки	Відомі користувачі
PHP	Серверна крос-платформенна мова сценаріїв із відкритим вихідним кодом, що слугує основою приблизно для 80% веб-сайтів в мережі Інтернет. Переважно використовується для створення динамічних веб-сайтів, але також знаходить застосування при керуванні файлами cookie, програмуванні серверних файлів та при створенні настільних програм. PHP має вбудовану підтримку таких баз даних, як MySQL, PostgreSQL і Oracle.	Laravel, CodeIgniter, Symfony	Facebook, Wikipedia, Tumblr, Slack, DocuSign, WordPress, Yahoo.
Java	Універсальний засіб розробки, що може використовуватись як для веб-сайтів та застосунків для Android, так і для розробки настільних програм та ігор. Основною характеристикою Java є незалежність від платформи та загальне призначення. Одним з найбільш поширених засобів для веб-розробки бек-енду мовою Java є фреймворк Spring Boot. Завдяки екосистемі Spring, Spring Boot пропонує безліч функцій, таких як автоматичне налаштування та підготовку до продакшену. Java може використовуватись для розробки надійних і масштабованих серверних модулів корпоративного рівня. Java має власні платформи для розробки, такі як Eclipse IDE і Maven.	Spring Boot, Struts, Hibernate.	Twitter, LinkedIn, eBay і NASA.
Python	Універсальна мова з відкритим вихідним кодом, що часто використовується для розробки серверних програм. Зважаючи на своє загальне призначення, окрім розробки серверної частини веб-сайтів, Python може застосовуватись для 3D-моделювання, обробки зображень, забезпечення кібербезпеки, наукових обчислень, аналізу даних і машинного навчання.	Django, Flask.	Mozilla, Spotify, Pinterest, Google, Facebook, Youtube.

Продовження таблиці Б.1

	При розробці веб-застосунків активно використовуються фреймворки Django, який має вбудований інтерфейс адміністратора і потужний ORM, а також Flask – легкий і розширюваний мікрофреймвокр, що ідеально підходить для проєктів малого та середнього розміру, Flask дає розробникам свободу вибору необхідних інструментів, не перевантажуючи їх попередньо налаштованими параметрами.		
JavaScript	Традиційно вважається мовою для фронт-енду та користувацького інтерфейсу, але з використанням таких фреймворків, як Node.js, може виступати в ролі повноцінного бек-енд рішення. Подвійна функціональність мови забезпечує швидкі цикли розробки, а неблокуючий і асинхронний характер, обробка помилок Throw-catch, розширюваність, безпека, ефективність, продуктивність і швидкість реагування в режимі реального часу допомагають йому виділитися під час створення мікросервісів (REST API) і безсерверних програм. Інший популярний фреймворк – Express.js – дозволяє розробникам легко створювати надійні API та веб-застосунки за допомогою обширної екосистеми проміжного програмного забезпечення. JavaScript пропонує багатопотоковість для виконання одночасних запитів, динамічну типізацію даних, широку підтримку, перевірку на стороні клієнта, стабільність, безпеку, повний цикл розробки, засоби асинхронного програмування та легку масштабованість під великим навантаженням	Next.js, Express.js, Node.js, Meteor.js	LinkedIn, GoDaddy, eBay, PayPal, Netflix, Uber і Google.
Kotlin	Мова, розроблена для сумісності з JVM, відома своєю стислістю та виразністю. Завдяки динамічному зростанню на таких платформах, як GitHub, легко інтегрується з бібліотеками Java.	Spring і Micronaut	Trello, Shazam, Uber, Postmates, Evernote, Kickstarter, Coursera.

Продовження таблиці Б.1

	Основні характеристики: статична типізація, можливість використання співпрограм для паралелізму, об'єктно-орієнтоване і функціональне програмування та інтеграція з такими фреймворками, як Spring і Micronaut, підтримка лямбда-виразів, функцій вищого порядку, кросплатформенність.		
Ruby	Мова для програмування настільних застосунків з динамічною типізацією, що має спеціалізований фреймворк, адаптований для використання в якості бек-енд платформи – Ruby on Rails. Ruby відрізняється високою масштабованістю, швидкістю та чіткістю коду. Основні її переваги: підтримка інтеграції баз даних, асинхронне введення-виведення та здатність запускати кілька потоків (паралелізм), акцент на тестуванні за допомогою таких інструментів, як RSpec, Capybara та Factory Bot, зріла екосистема бібліотек	Ruby on Rails, Sinatra.	Airbnb, GitHub, Shopify.
Go (Golang)	Мова із відкритим вихідним кодом, створена Google. Golang відома своїм чітким синтаксисом, ефективністю керування одночасними запитами, швидкістю, простотою та масштабованістю. Основні характеристики мови – безпечне використання пам'яті, краще «збирання сміття», висока обчислювальна потужність, керування об'єктами та широка підтримка спільноти. Паралелізм досягається запуском багатопотокових функцій через GoRoutines із малим рівнем залучення оперативної пам'яті. Одним з найбільш популярних фреймворків для використання у веб-розробці є Gin – веб-платформа Golang, що характеризується низьким обсягом пам'яті та швидким маршрутизатором, які дозволяють сворювати високопродуктивні API та мікросервіси для вебу.	Gin, Web.go, Revel, GORM, Gorilla	Google, Dropbox, Docker.
C#	Початково створена як об'єктно-орієнтована та синхронна мова програмування для настільних програм в ОС Windows, C# розширила свою присутність на macOS і Linux завдяки використанню .NET Framework та .NET Core.	ASP.NET Core, Blazor	MSN, Bing, StackOverflow, GoDaddy.

Продовження таблиці Б.1

	ASP.NET Core має покращену продуктивність, модульну архітектуру та засоби інтеграції з сучасними інтерфейсними фреймворками. Постійний розвиток ASP.NET Core гарантує, що він залишається актуальним і надійним для розробників на платформі .NET. C# використовує Entity Framework для роботи з базами даних. Мова характеризується підтримкою кількох серверних систем, надійною типізацією та засобами для покращення розподілу і очистки пам’яті.		
Perl	Мова програмування серверної частини з підтримкою розширених засобів обробки тексту, зіставлення шаблонів та швидкої розробки. Мова здатна обробляти регулярні вирази, що допомагає виконувати пошук, заміну та інші маніпуляції з текстовою інформацією. Підтримує кілька бібліотек і модулів для створення масштабованих, безпечних, придатних для обслуговування та надійних серверних систем, які гарантують високу продуктивність. Perl має потужну підтримку спільноти. Функціонал мови можна розширити через використання бібліотеки пакетів Comprehensive Perl Archive Network.	Catalyst, Mojolicious, Dancer2	Redhat, Mozilla, Webkit, Apache

Програмна реалізація UserTableSeeder.php

```
<?php

namespace Database\Seeders\Setup;

use App\Models\User;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\Hash;

class UserTableSeeder extends Seeder
{
    protected const DEFAULT_PASSWORD = 'W8yJG4Ms6sEby)X$';

    public function run(): void
    {
        foreach ($this->getUsers() as $options) {
            $user = User::create($options['user']);
            $user->assignRole($options['role']);
        }
    }

    protected function getUsers(): array
    {
        $users = [
            [
                'user' => [
                    'name' => 'admin',
                    'email' => 'admin@nomail.com',
                    'email_verified_at' => now(),
                    'password' => Hash::make(self::DEFAULT_PASSWORD),
                ],
                'role' => PermissionTableSeeder::ROLE_ADMIN,
            ],
            [
                'user' => [
                    'name' => 'first_student',
                    'email' => 'first_student@nomail.com',
                    'email_verified_at' => now(),
                    'password' => Hash::make(self::DEFAULT_PASSWORD),
                ],
                'role' => PermissionTableSeeder::ROLE_STUDENT,
            ],
            [
                'user' => [
                    'name' => 'second_student',
                    'email' => 'second_student@nomail.com',
                    'email_verified_at' => now(),
                    'password' => Hash::make(self::DEFAULT_PASSWORD),
                ],
                'role' => PermissionTableSeeder::ROLE_STUDENT,
            ],
        ];
```

```
 return array_merge($users, self::getTeachers());
    }

    public static function getTeachers(): array
    {
        return [
            [
                'user' => [
                    'name' => 'first_teacher',
                    'email' => 'first_teacher@nomail.com',
                    'email_verified_at' => now(),
                    'password' => Hash::make(self::DEFAULT_PASSWORD),
                ],
                'role' => PermissionTableSeeder::ROLE_TEACHER,
            ],
            [
                'user' => [
                    'name' => 'second_teacher',
                    'email' => 'second_teacher@nomail.com',
                    'email_verified_at' => now(),
                    'password' => Hash::make(self::DEFAULT_PASSWORD),
                ],
                'role' => PermissionTableSeeder::ROLE_TEACHER,
            ],
        ];
    }
}
```

Програмна реалізація PermissionTableSeeder.php

```
<?php

namespace Database\Seeders\Setup;


use Illuminate\Database\Seeder;
use Spatie\Permission\Models\Role;
use Spatie\Permission\Models\Permission;
use Spatie\Permission\PermissionRegistrar;

class PermissionTableSeeder extends Seeder
{
    public const ROLE_ADMIN = 'admin';
    public const ROLE_TEACHER = 'teacher';
    public const ROLE_STUDENT = 'student';

    public function run(): void
    {
        // Reset cached roles and permissions
        app()[PermissionRegistrar::class]->forgetCachedPermissions();

        // Create permissions
        Permission::create(['name' => 'admin.access']);
        Permission::create(['name' => 'teacher.access']);
        Permission::create(['name' => 'student.access']);

        // Create roles
        $adminRole = Role::create(['name' => self::ROLE_ADMIN]);
        $teacherRole = Role::create(['name' => self::ROLE_TEACHER]);
        $studentRole = Role::create(['name' => self::ROLE_STUDENT]);

        // Assign roles permissions
        $adminRole->givePermissionTo('admin.access');
        $teacherRole->givePermissionTo('teacher.access');
        $studentRole->givePermissionTo('student.access');
    }
}
```

Програмна реалізація PermissionTableSeeder.php

```
<?php
namespace Database\Seeders\Setup;
use App\Models\Quiz\Question;
use Illuminate\Database\Seeder;

class QuestionTableSeeder extends Seeder
{
    public function run(): void
    {
        foreach (self::getQuestions() as $question) {
            Question::create([
                'cf' => $question['cf'],
                'content' => $question['content'],
            ]);
        }
    }

    public static function getQuestions(): array
    {
        return [
            [
                'cf' => 0.1,
                'content' => 'Доступність подачі матеріалу',
            ],
            [
                'cf' => 0.07,
                'content' => 'Здатність адаптуватись під індивідуальні
особливості навчальної групи',
            ],
            [
                'cf' => 0.05,
                'content' => 'Комунікативність',
            ],
            [
                'cf' => 0.06,
                'content' => 'Підготовка до занять',
            ],
            [
                'cf' => 0.03,
                'content' => 'Дотримання графіку занять',
            ],
            [
                'cf' => 0.07,
                'content' => 'Якість лекцій',
            ],
            [
                'cf' => 0.05,
                'content' => 'Доступність для індивідуальних
консультацій',
            ],            [
                'cf' => 0.05,
```

```
'content' => 'Інтерактивність занять',
            ],
            [
                'cf' => 0.03,
                'content' => 'Інтерес до предмету',
            ],
            [
                'cf' => 0.04,
                'content' => 'Сприяння активності студентів на
заняттях',
            ],
            [
                'cf' => 0.05,
                'content' => 'Ефективність використання додаткових
навчальних матеріалів',
            ],
            [
                'cf' => 0.06,
                'content' => 'Використання сучасних методів навчання',
            ],
            [
                'cf' => 0.04,
                'content' => 'Об'єктивність при оцінюванні студентів',
            ],
            [
                'cf' => 0.04,
                'content' => 'Стимулювання самостійної роботи
студентів',
            ],
            [
                'cf' => 0.04,
                'content' => 'Відповідність навчального процесу
програмним вимогам',
            ],
            [
                'cf' => 0.04,
                'content' => 'Вміння мотивувати студентів до
навчання',
            ],
            [
                'cf' => 0.04,
                'content' => 'Дотримання етичних стандартів',
            ],
            [
                'cf' => 0.05,
                'content' => 'Здатність роз'яснити складні поняття',
            ],
            [
                'cf' => 0.05,
    'content' => 'Створення сприятливої атмосфери на заняттях',
            ],
        ];
    }
}
```

Програмна реалізація web.php

```
<?php


use App\Http\Controllers;

use Illuminate\Support\Facades\Route;

use App\Http\Middleware\GuestLoginMiddleware;


/** Auth routes */

Route::middleware([GuestLoginMiddleware::class])->group(static
function () {

    Route::get('/',
[Controllers\Auth\AuthenticatedSessionController::class, 'create'])

         ->name('login');

    Route::post('/',
[Controllers\Auth\AuthenticatedSessionController::class, 'store']);

});


Route::middleware('auth:web')->group(static function () {

    /** Logout route */

    Route::post('logout',
[Controllers\Auth\AuthenticatedSessionController::class, 'destroy'])

        ->name('logout');


    /** Admin routes */

    Route::group([

        'as' => 'admin.',

        'prefix' => 'admin',

        'middleware' => ['permission:admin.access'],

    ], static function () {

        Route::get('/', [Controllers\Admin\HomeController::class,
'index'])

             ->name('home');

        Route::resource('user',
Controllers\Admin\UserController::class);
```

```
        Route::resource('question',
Controllers\Admin\QuestionController::class);

        Route::resource('quiz',
Controllers\Admin\QuizController::class);

    });


    /** Student routes */

    Route::group([

        'as' => 'student.',

        'prefix' => 'student',

        'middleware' => ['permission:student.access'],

    ], static function () {

        Route::get('/', [Controllers\Student\HomeController::class,
'index'])

            ->name('home');

        Route::get('/quiz/{quiz}',
[Controllers\Student\QuizController::class, 'show'])

            ->name('quiz.show');

        Route::post('/quiz/{quiz}',
[Controllers\Student\QuizController::class, 'store'])

            ->name('quiz.store');

    });


    /** Teacher routes */

    Route::group([

        'as' => 'teacher.',

        'prefix' => 'teacher',

        'middleware' => ['permission:teacher.access'],

    ], static function () {

        Route::get('/', [Controllers\Teacher\HomeController::class,
'index'])

            ->name('home');

    });

});
```

Програмна реалізація quizservice.php

```
<?php

namespace App\Services\Student;

use App\Models\Quiz\QuizComplete;
use App\Models\User;
use App\Models\Quiz\Quiz;
use Illuminate\Http\Request;
use App\Services\BaseService;
use App\Models\Quiz\QuizQuestion;
use App\Models\Quiz\QuizQuestionGrade;
use Illuminate\Database\Eloquent\Builder;
use Illuminate\Pagination\LengthAwarePaginator;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
use Throwable;

class QuizService extends BaseService
{
    protected User $user;

    protected function boot(): void
    {
        $this->user = Auth::user();
    }

    public function getModel(int $id): ?Quiz
    {
        $studentHash = $this->getUserHash();

        $qb = Quiz::whereId($id)->whereDoesntHave('grades',
function(Builder $query) use ($studentHash) {
```

```
     /** Records that the current student did not rate */
            $query->where('quiz_questions_grades.student_hash',
$studentHash);
        });

        return $qb->first();
    }

    public function getDataProvider(Request $request):
LengthAwarePaginator
    {
        $studentHash = $this->getUserHash();

        $qb = Quiz::with(['teacher', 'questions'])-
>whereDoesntHave('grades', function(Builder $query) use ($studentHash)
{
            /** Records that the current student did not rate */
            $query->where('quiz_questions_grades.student_hash',
$studentHash);
        });

        $qb->orderBy('id', 'desc');

        return $qb->paginate(self::PER_PAGE);
    }

    public function getUserHash(): string
    {
        return hash('sha256', $this->user->id);
    }

    public function create(array $attributes = []): bool
    {
        $grades = $attributes['grades'];
        $quizId = $attributes['quiz_id'];
```

```
        $studentHash = $this->getUserHash();

        DB::beginTransaction();
        try {
            foreach ($grades as $questionId => $grade) {
                $quizQuestion = QuizQuestion::where(['quiz_id' =>
$quizId, 'question_id' => $questionId])->first();

                QuizQuestionGrade::create([
                    'quiz_id' => $quizQuestion->quiz_id,
                    'question_id' => $quizQuestion->question_id,
                    'student_hash' => $studentHash,
                    'grade' => $grade['value'],
                ]);
            }

            QuizComplete::create([
                'quiz_id' => $quizId,
                'student_hash' => $studentHash,
            ]);

            DB::commit();

            return true;
        } catch (Throwable $e) {
            DB::rollBack();

            throw $e;
        }
    }
}
```