МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

ШАПОВАЛОВ Ярослав Юрійович

Алгоритми проектування компонентів електронних навчальних систем / Algorithms for designing components of electronic educational systems

спеціальність: 123 - Комп'ютерна інженерія
освітньо-професійна програма - Комп'ютерна інженерія
Кваліфікаційна робота

Виконав студент групи КІм-21
Я.Ю. Шаповалов
_____

Науковий керівник:
к.т.н., Г.М. Мельник
_____

Кваліфікаційну роботу допущено
до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____Л. О. Дубчак

Тернопіль – 2023

РЕЗЮМЕ

Кваліфікаційна робота на тему «Алгоритми проектування компонентів електронних навчальних систем» зі спеціальності 123 «Комп'ютерна інженерія» освітнього ступеня «магістр» написана обсягом 70 сторінок і містить 19 ілюстрацій, 6 таблиць, 3 додатки та 50 джерел за переліком посилань.

Метою кваліфікаційної роботи є розроблення алгоритмів проектування компонентів електронних навчальних систем на основі аналізу користувацьких даних для підвищення ефективності індивідуалізації навчального процесу.

Методи дослідження: методи системного аналізу, математичного моделювання.

Наукова новизна одержаних результатів. Розроблено алгоритми проєктування компонентів електронних навчальних систем на основі планування індивідуального навчального шляху, що дозволило враховувати оцінювання для пропозиції інших навчальних компонент.

Практичне значення. Розроблені алгоритми дозволяють інтегрувати індивідуальний навчальний шлях в електронний підручник. Програмно реалізовано алгоритми збору та аналіз користувацьких даних, прогнозування оптимального навчального шляху, адаптації навчальних матеріалів і завдань. Розроблено базу даних для ресурсів електронного підручника. Для тестування алгоритмів створено тестове середовище в системі дистанційного навчання Moodle.

КЛЮЧОВІ СЛОВА: ЕЛЕКТРОННИЙ ПІДРУЧНИК, СИСТЕМА ДИСТАНЦІЙНОГО НАВЧАННЯ, ІНДИВІДУАЛЬНИЙ ШЛЯХ НАВЧАННЯ.

RESUME

Qualification work on «Algorithms for designing components of electronic educational systems» in the specialty 123 "Computer Engineering" educational degree "Master" is written in 70 pages and contains 19 illustrations, 6 tables, 3 appendices and 50 sources by reference.

The purpose of the qualification work is to develop algorithms for designing components of electronic learning systems based on the analysis of user data to improve the efficiency of individualization of the educational process.

Research methods: methods of system analysis, mathematical modeling.

Scientific novelty of the results. Algorithms for designing components of e-learning systems based on planning an individual learning path have been developed, which made it possible to take into account the assessment for the proposal of other learning components.

Practical significance. The developed algorithms allow to integrate an individual learning path into an electronic textbook. The algorithms for collecting and analyzing user data, predicting the optimal learning path, and adapting learning materials and tasks have been programmatically implemented. A database for e-textbook resources was developed. A test environment in the Moodle distance learning system was created to test the algorithms.

KEYWORDS: ELECTRONIC TEXTBOOK, LEARNING MANAGEMENT SYSTEM, INDIVIDUAL LEARNING PATH.

ЗМІСТ

ВСТУП

У сучасному світі освіти, що стрімко розвивається, електронні навчальні системи стають невід'ємною частиною академічного та професійного зростання. Онлайн-освіта надає унікальні можливості для навчання, віддаленого доступу до знань і гнучкості в освітніх процесах. У центрі цієї еволюції знаходяться алгоритми проектування компонентів електронних навчальних систем, які забезпечують інноваційність, адаптивність та персоналізацію навчального досвіду.

Ця робота присвячена розробці та аналізу алгоритмів, які лежать в основі проектування ефективних компонентів для електронних навчальних систем. Вона охоплює систематичний підхід до визначення, моделювання та імплементації алгоритмічних рішень, що відповідають за структурування навчального контенту, організацію взаємодій і оцінювання знань у віртуальному освітньому просторі.

Значимість дослідження полягає у створенні фундаменту для подальшого розвитку інтелектуальних систем, що можуть самостійно адаптуватися до потреб студентів та викладачів, враховуючи індивідуальні особливості користувачів і динаміку освітнього процесу. Акцент робиться на алгоритмах, які сприяють підвищенню ефективності навчання та забезпечують можливість швидкої адаптації навчальних матеріалів до змінних умов і вимог.

Метою кваліфікаційної роботи є розроблення алгоритмів проектування компонентів електронних навчальних систем на основі аналізу користувацьких даних для підвищення ефективності індивідуалізації навчального процесу.

Об'єкт дослідження - процеси індивідуалізації навчання в електронних навчальних системах.

Предмет дослідження - **а**лгоритми та методи проектування компонентів електронних навчальних систем.

Для досягнення поставленої мети роботи можна успішно досягти шляхом виконання таких завдань:

– провести аналітичний огляд алгоритмів проектування компонентів електронних навчальних систем;

– проаналізувати потреби користувачів щодо зручності та функціональності компонентів електронних навчальних систем;

– розробити алгоритми проєктування компонентів системи;

– реалізувати прототип на основі розроблених алгоритмів та провести тестування.

Наукова новизна одержаних результатів. Розроблено алгоритми проєктування компонентів електронних навчальних систем на основі планування індивідуального навчального шляху, що дозволило враховувати оцінювання для пропозиції інших навчальних компонент.

Практичне значення отриманих результатів. Розроблені алгоритми дозволяють інтегрувати індивідуальний навчальний шлях в електронний підручник.

Апробація роботи. Отримані результати опубліковані в межах VIII Науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі», опубліковано тези доповіді [1,2].

Впровадження результатів ДР. Результати роботи рекомендовано до впровадження на підприємстві (додаток Б).

У першому розділі дипломної роботи розглядаються теоретичні аспекти проектування електронних навчальних систем, історія їх розвитку та сучасний стан. Другий розділ присвячений аналізу існуючих алгоритмів та методологій проектування, їх сильних та слабких сторін.

Третій розділ зосереджений на розробці нових алгоритмічних рішень, орієнтованих на оптимізацію навчального процесу та покращення взаємодії між учасниками освітнього процесу. В останньому розділі представлено практичне застосування розроблених алгоритмів.

1 МЕТОДИ ТА АЛГОРИТМИ ПРОЕКТУВАННЯ ЕЛЕКТРОННИХ НАВЧАЛЬНИХ СИСТЕМ

1.1 Дистанційне навчання

Освітній процес є складною і важливою сферою, де відбувається передача знань і спілкування, під час якого активно розвивається пізнавальний процес, відбувається апробація суспільно-історичного досвіду, та формується особистість. Суть навчання полягає в тісному взаємодії між вчителем та учнем, де обидві сторони активно співпрацюють.

У процесі навчання, досягається реалізація освіти і впливу на вихованців. Дії вчителя спонукають учнів до активності, аби досягти конкретної мети, і надають напрямок цій активності. Важливо забезпечити необхідні умови для розвитку ініціативи учнів за допомогою різноманітних педагогічних засобів. Освітній процес включає в себе дидактичну діяльність, стимулювання інтересу учнів до навчання, активну роботу учнів і вчителя з керування навчанням.

Для забезпечення ефективності освітнього процесу важливо розрізняти момент організації діяльності та момент самого навчання в рамках цієї організації. Організація другої складової є прямим завданням вчителя і має критичне значення для успішної передачі знань та інформації. Спосіб взаємодії між учнем і вчителем при освоєнні будь-яких знань і інформації визначає ефективність освітнього процесу.

Сама діяльність учня в освітньому процесі полягає в конкретних діях, спрямованих на досягнення певного результату. Ця діяльність визначається різними мотивами, і найважливішими її якостями є самостійність, готовність подолати труднощі, відданість завданню та оперативність. Остання передбачає правильне розуміння завдань, вибір належної дії та швидкість її виконання.

У зв'язку з постійними змінами в сучасному житті, знання, навички і уміння стають динамічними явищами, які постійно переосмислюються. Тому освітній процес повинен бути адаптований до змін в інформаційному просторі. Таким

чином, мета освітнього процесу включає не лише набуття знань, умінь і навичок, але й розвиток психічних процесів особистості, формування моральних та правових переконань і дій.

Один із ключових аспектів освітнього процесу полягає у його циклічності, де кожен цикл складається з ряду конкретних етапів, визначаючи основні показники для кожного з них: цілі, які можуть бути як загальними, так і специфічними для певної теми, засоби, що використовуються для досягнення цілей, і результати, пов'язані з рівнем освоєння навчального матеріалу. Розрізняють чотири основних цикли в освітньому процесі, кожен з яких відзначається своєю унікальною метою та завданнями.

Перший цикл, що отримав назву "Початковий цикл", спрямований на усвідомлення та розуміння основних концепцій та практичної важливості вивченого матеріалу. На цьому етапі також акцентується на оволодінні шляхами відтворення набутих знань та методами їхнього використання на практиці.

Другий цикл, або "Цикл конкретизації", спрямований на глибоке розширене освоєння вивчених знань і їхнє явне усвідомлення. Учні на цьому етапі більш детально розглядають та аналізують вивчений матеріал.

Третій цикл, що отримав назву "Систематизації", ставить за мету систематизувати та узагальнити поняття, отримані на попередніх етапах, і використовувати їх в реальних життєвих ситуаціях.

Заключний цикл, імовірно найважливіший, називається "Цикл перевірки і оцінки". Його головною метою є перевірка та облік результатів, отриманих на попередніх етапах, з використанням контролю та самоконтролю, що сприяє забезпеченню якості навчання.

Ці цикли в освітньому процесі розкривають важливість системності та послідовності при передачі та освоєнні знань, що допомагає учням краще розуміти і використовувати навчальний матеріал.

Програмоване навчання представляє собою відмінну форму навчання, що сприяє самостійному та індивідуальному освоєнню знань та навичок відповідно до навчальної програми, використовуючи інформаційні ресурси. Теорія

програмованого навчання зародилася в початку 60-х років XX століття в США, надихнувшись досягненнями кібернетики, і вона відіграла ключову роль у розвитку навчальних технологій, створенні теоретичних основ та практичних досліджень у сфері складних систем навчання.

Порівняно з традиційним методом навчання, де учень зазвичай просто читає текст підручника, програмоване навчання відрізняється систематичним керуванням навчальним процесом за допомогою навчальної програми. Навчальна програма може розглядатися як послідовність структурованих рекомендацій або завдань, які передаються учневі через дидактичну машину і виконуються ним.

Однією з головних переваг програмованого навчання є можливість індивідуалізувати темп навчання для кожного учня, стимулюючи самостійну роботу та забезпечуючи постійний контроль над рівнем освоєння навчального матеріалу.

Програмоване навчання ґрунтується на ряді важливих принципів, які сприяють ефективному процесу освоєння матеріалу:

1. Розбиття навчального матеріалу на логічно пов'язані фрагменти, що послідовно об'єднані між собою. Кожен фрагмент є кроком у процесі навчання.

2. Залучення учнів до активної пізнавальної діяльності при вивченні програмованих фрагментів. Цей принцип спрямований на стимулювання інтелектуального зростання учнів і розвиток їхньої самостійності.

3. Урахування індивідуальних особливостей кожного учня. Програмоване навчання підходить до кожного учня окремо, дозволяючи враховувати їхні потреби та темпи навчання.

Завдяки цим принципам, програмоване навчання забезпечує систематичний та постійний обмін інформацією між педагогом і учнем. Ця взаємодія сприяє взаємному самовдосконаленню обох сторін та покращує якість навчання та освоєння знань.

На сучасному етапі існує декілька різновидів програм в програмованому навчанні, кожен з яких має свої особливості та переваги:

1. Лінійна програма: цей тип програми базується на принципі розбиття матеріалу на невеликі кроки і негайного підтвердження відповіді. Спочатку навчальна програма пропонує досить прості завдання, а з часом збільшує їх складність. У цьому варіанті засвоєння інформації передбачається за єдиною схемою.

2. Програма з розгалуженням: цей тип програми ґрунтується на поділі матеріалу на частини, після кожної з яких учню ставиться питання, що вимагає самостійного вибору правильної відповіді із кількох можливих варіантів. Після вибору відповіді відбувається перевірка її правильності. Ця програма сприяє індивідуалізації темпу навчання, враховуючи рівень підготовки кожного учня.

3. Змішана програма: цей різновид програми представляє собою комбінацію лінійного та розгалуженого підходів. Вона дозволяє використовувати різні методи навчання в залежності від потреб та характеристик учнів.

Кожен з цих типів програм в програмованому навчанні має свої переваги і може бути використаний в залежності від конкретних цілей та особливостей навчання.

1.2 Індивідуалізація навчання та потреби користувачів

Стаття [5] важлива для розуміння того, як візуальний аналіз даних може допомогти у вивченні та розумінні паттернів активності студентів у LMS та їх зв'язку з навчальними результатами. Використання ієрархічного кластерного аналізу та візуалізації у вигляді теплових карт (heatmap) відкриває можливості для аналізу взаємодій студентів із системою. Результати цього дослідження можуть допомогти краще зрозуміти, які функції LMS та дані про взаємодію є найбільш корисними для студентів, що є цінним для розробки ефективних алгоритмів проектування компонентів електронних навчальних систем.

Стаття, яка аналізує паттерни активності студентів у LMS за допомогою ієрархічного кластерного аналізу та візуалізації у вигляді теплових карт, може

бути корисною нам для розробки моделі вибору індивідуального шляху в такий спосіб:

1. Ідентифікація паттернів навчання: Розуміння як високопродуктивні та низькопродуктивні/відмовлені студенти взаємодіють з LMS, що дозволяє адаптувати навчальні шляхи під конкретні потреби.

2. Аналіз ефективності контенту: Використання зібраних даних для визначення, які види контенту є найбільш ефективними для різних груп студентів.

3. Поліпшення алгоритмів рекомендацій: Внесення змін до алгоритмів, що визначають індивідуальний шлях навчання, на основі виявлених паттернів активності.

Потреби користувачів щодо зручності та функціональності компонентів електронних навчальних систем.

Сучасні електронні навчальні системи стають дедалі популярнішими, особливо у контексті глобальних змін в освітньому процесі. Зростання цієї популярності породжує питання про потреби користувачів: які вони, чи задовольняються ці потреби сучасними системами та як можна їх вдосконалити.

Потреби користувачів з точки зору зручності:

– Інтуїтивний дизайн: Користувачі очікують, що система буде інтуїтивно зрозумілою, з мінімальними потребами в навчанні.

– Адаптивний дизайн: З урахуванням різних пристроїв, на яких користувачі можуть отримувати доступ до системи (смартфони, планшети, настільні комп'ютери).

– Персоналізація: Можливість налаштовувати інтерфейс та функціональність відповідно до особистих потреб користувача.

Функціональні потреби користувачів:

– Інтерактивні елементи: Інструменти, які дозволяють студентам активно взаємодіяти з матеріалом (наприклад, інтерактивні тести, відео, гейміфікація).

– Системи сповіщень: Автоматичні сповіщення про нові матеріали, завдання, зміни в розкладі та інші важливі події.

– Інструменти колаборації: Можливості для групової роботи, обговорення, спільного редагування документів.

– Зворотний зв'язок: Інструменти для отримання зворотного зв'язку від викладачів або інших студентів.

Проаналізовані проблеми та виклики. Розгляд поточних проблем і викликів, з якими користувачі зіштовхуються при використанні електронних навчальних систем, в тому числі проблеми технічного характеру, обмеженості функціональності, проблеми з інтерфейсом та інше.

Визначення ключових потреб користувачів дозволяє розробникам електронних навчальних систем краще зрозуміти, які аспекти зручності та функціональності важливі для кінцевого користувача, і відповідно адаптувати свої продукти. Це в свою чергу сприяє підвищенню якості освітнього процесу та задоволенності користувачів.

Електронний підручник.

Термін "електронний підручник" став узагальнюючим поняттям, яке замінило різноманітні інші назви у сфері освіти, такі як програмні засоби для педагогів, інструменти для електронного навчання, автоматизовані освітні системи та інші. Оскільки ринок масових електронних підручників все ще розвивається, можна очікувати появи нових термінів, наприклад, "інформаційний освітній ресурс".

В різних джерелах електронний підручник описується як:

– Комплекс інформації у вигляді тексту, графіки, цифрових даних, аудіо та відео матеріалів, доступний на електронних носіях або через мережу.

– Засіб, що систематизує знання в певній науково-практичній сфері, сприяючи активному засвоєнню знань та навичок.

– Офіційно затверджене електронне видання, що доповнює або замінює традиційний підручник.

– Електронний текст з великою кількістю гіперпосилань, які дозволяють швидко переміщатися між розділами згідно з певною ієрархією.

Розробка цифрових освітніх ресурсів здійснюється в спеціалізованих програмних середовищах, включаючи конструктори електронних курсів та системи автоматизованого проектування. Процес проектування включає ідентифікацію проблем, концептуалізацію, формалізацію, реалізацію та тестування.

Кожен етап має свої особливості:

– Ідентифікація: Визначення ролей учасників, цілей та ресурсів.

– Концептуалізація: Визначення змісту та цілей навчання.

– Формалізація: Аналіз методів вирішення дидактичних завдань.

– Реалізація: Переклад методів у сценарії для автоматизованих систем.

– Тестування: Перевірка функціональності та виявлення слабких місць ресурсу.

Процес створення цифрових освітніх ресурсів (ЦОР), зокрема підручників і посібників, охоплює декілька ключових фаз:

1. Підготовча стадія,
2. Розробка змісту,
3. Концептування та дизайн,
4. Процес створення,
5. Верифікація та тестування.

На етапі підготовки важливо розробити нормативні документи, що визначають процедури створення ЦОР. Також необхідно вибрати відповідні інструменти та платформи для ефективної підготовки та розробки освітніх та методичних матеріалів.

Початкова стадія підготовки охоплює кілька ключових кроків:

– Аналіз дидактичних потреб ЦОР: Вивчення конкретних вимог курсу, характеристик потенційних учнів та освітніх цілей для визначення підходящої форми ЦОР.

– Встановлення технічних параметрів: Переконатися в технічній виконуваності обраної технології, при необхідності адаптувати технічні специфікації або зміст курсу.

– Створення структури ЦОР.

– Розробка методології використання ЦОР для викладачів.

– Розробка інструкцій для учнів щодо роботи з ЦОР.

1. Розробка змісту ЦОР включає дві основні частини:

Інформаційна складова:

– Профіль автора курсу з фотографією.

– Методичні настанови для вивчення матеріалу.

– Ясно організовані навчальні матеріали.

– Мультимедійні ілюстрації, включаючи графіку, анімацію, аудіо та відео.

– Практичні завдання для вдосконалення навичок, з прикладами та аналізом помилок.

– Система оцінювання та контролю, включаючи тестові завдання та групові вправи.

– Додаткові ресурси, від глосаріїв до електронних бібліотек.

– Сервісні інструменти, як-то керівництва, словники, глосарії.

Програмна частина: Розробка та налаштування програмного забезпечення, що підтримує змістовну частину.

На стадії розробки змісту важливою є створення деталізованого плану курсу, який іноді супроводжується ілюстративними матеріалами для кращого розуміння інструкцій. Важливо врахувати аспекти інтерактивності, включаючи взаємодію студента з комп'ютером, викладачем та іншими студентами.

Основні компоненти програмної підсистеми цифрових освітніх ресурсів можуть включати:

- Систему реєстрації студентів;
- Модулі навчального матеріалу з різноманітними завданнями для самоконтролю та оцінювання;

- Додаткові ресурси, включаючи глосарії, нормативні документи та електронні бібліотеки;
- Сервісні інструменти, такі як довідники, словники, системи пошуку;
- Комунікаційну систему для взаємодії між викладачами та студентами;
- Систему безпеки.

Процеси підготовки змісту та програмування взаємодіють, дозволяючи розробникам краще використовувати технологічні можливості для структурування матеріалів.

3. Дизайн. На цьому етапі фіксується загальна структура цифрового ресурсу та створюється детальний план. Процес включає:

- Розробку основної концепції дизайну: визначення стилю, атмосфери курсу, навігації, зворотного зв'язку, вибір елементів управління.
- Деталізацію дизайну: глибоке опрацювання змісту, візуального оформлення, контекстних меню.

4. Виробництво. На цьому етапі відбувається безпосередня розробка: створення модулів, налагодження посилань, взаємодія елементів ресурсу, цифрова обробка графіки та звуку.

5. Тестування. Тестування проводиться на кожному етапі для забезпечення відповідності дидактичним цілям, виявлення програмних помилок.

Фінальне тестування в експериментальних групах включає:

- Перевірку функціональності всіх модулів;
- Виявлення помилок у навчальному матеріалі та програмному забезпеченні;
- Оцінку зручності інтерфейсу;
- Вимірювання середнього часу, який студенти витрачають на курс;
- Збір даних для перевірки валідності тестових завдань.

1.3 Існуючі засоби і алгоритми проектування

Електронні навчальні системи стали невід'ємною частиною сучасної освітньої парадигми. Проектування компонентів цих систем потребує спеціальних методів та алгоритмів, щоб забезпечити ефективність, надійність та зручність використання. Розглянемо основні методи та алгоритми, що використовуються в цій галузі.

1. Модульний підхід. Однією з основних ідей у проектуванні компонентів є використання модульного підходу. Це дозволяє створювати незалежні блоки або модулі, які можна легко інтегрувати у навчальну систему. Модульність забезпечує гнучкість, легкість внесення змін та масштабування системи.

2 Об'єктно-орієнтоване проектування. ООП є основою для більшості сучасних програмних продуктів. В контексті електронних навчальних систем це дозволяє створювати зручні і зрозумілі структури, що відображають реальні об'єкти освітнього процесу: уроки, тести, матеріали тощо.

3 Алгоритми оптимізації ресурсів. З огляду на різноманіття девайсів, які використовуються для доступу до електронних навчальних систем, алгоритми оптимізації ресурсів важливі для забезпечення швидкодії та реактивності системи.

4 Адаптивні алгоритми**.** Сучасні навчальні системи часто мають адаптивні алгоритми, які автоматично налаштовують зміст або навчальний шлях на основі вхідних даних про користувача. Це може включати аналіз рівня знань, стилю навчання або інших параметрів.

5 Принципи гейміфікації**.** Гейміфікація використовує принципи гри для підвищення зацікавленості та мотивації користувачів. В електронних навчальних системах це може бути використано для створення інтерактивних вправ, конкурсів або навчальних квестів.

Існуючі методи та алгоритми проектування компонентів електронних навчальних систем покликані відповідати потребам сучасних користувачів та освітнього середовища. Вони надають можливість створювати ефективні, адаптивні та мотивуючі системи, які сприяють підвищенню якості освітнього процесу.

Інструментальні засоби для проектування компонентів електронних навчальних систем

Сучасні інструментальні засоби надають розробникам широкий спектр можливостей для проектування та створення ефективних електронних навчальних систем. Вони включають різноманітне програмне забезпечення для дизайну, моделювання, тестування та розгортання.

Опишемо підходи.

– Cloud-based (Хмарний підхід)**:** Застосунки і дані розміщені в інтернеті, зазвичай на серверах третіх сторін. Хмарні рішення забезпечують легкий доступ з будь-якого місця, де є інтернет. Також це спрощує процеси оновлення та масштабування.

– On-premises (На місці)**:** Застосунки і дані розміщені на місцевих серверах організації. Це дає більше контролю над даними і інфраструктурою, але може потребувати більше ресурсів для підтримки та обслуговування.

– Hybrid (Гібридний)**:** Комбінація хмарних та локальних підходів. Це може дати можливість використовувати переваги обох підходів в залежності від конкретних потреб.

Методики проектування**:**

– Modular (Модульна)**:** Система побудована з незалежних модулів, які можна додавати, видаляти або модифікувати без завад для інших частин системи.

– Integrated (Інтегрована)**:** Всі компоненти системи тісно пов'язані і працюють як єдиний цілісний комплекс.

Алгоритми проектування**:**

– Personalization (Персоналізація)**:** Система адаптує навчальний контент або досвід на основі індивідуальних потреб та вподобань користувача.

– Standardized (Стандартний)**:** Всі користувачі отримують однаковий навчальний досвід незалежно від їхніх індивідуальних потреб.

– Adaptive (Адаптивний)**:** Система автоматично змінює навчальний контент або досвід на основі взаємодії користувача з матеріалом і його прогресу.

Таблиця 1.1 - Методики та алгоритми проектування

Параметри	Засіб А (наприклад, Moodle)	Засіб В (наприклад, Blackboard)	Засіб С (наприклад, Canvas)
Підхід	Cloud-based	On-premises	Hybrid
Методика	Modular	Integrated	Modular
Алгоритм	Personalization	Standardized	Adaptive
Ліцензія	Open source	Proprietary	Open source
Масштабованість	Висока	Середня	Висока
Інтеграція	API, Plugins	API, Extensions	API, Plugins
Забезпечення зворотного зв'язку	Автоматизоване	Ручне	Комбіноване

1.4 Системи дистанційної освіти

Цифровий освітній ресурс (ЦОР), як і будь-який інший освітній матеріал, має відповідати ряду критичних якісних стандартів. Оцінюючи цифрові освітні ресурси, можна виділити кілька ключових критеріїв, зазначених у наукових дослідженнях:

- Сумісність зі змістом освітніх програм, законодавчими нормами Міністерства освіти і науки та використовуваними програмами;
- Відповідність сучасним освітнім методикам, забезпечення високого рівня інтерактивності та мультимедійності;
- Підтримка рівневої диференціації та індивідуального підходу до навчання, врахування вікових та культурних особливостей учнів;
- Забезпечення різноманітних видів навчальної діяльності, зорієнтованих на практичне застосування знань для вирішення реальних проблем;
- Можливість використання як індивідуальної, так і групової роботи;
- Включення гнучких навчальних планів з модульною структурою;
- Базування на достовірних та перевірених матеріалах;
- Забезпечення більш широкого охоплення тематики, ніж у традиційних підручниках, без надмірного розширення змісту;

- Повне відображення на різноманітних технічних платформах;
- Сумісність з іншими програмами та інструментами;
- Налаштування інтерфейсу та зберігання результатів, де це методично обґрунтовано;
- Наявність вбудованої контекстної допомоги, де це потрібно;
- Інтуїтивно зрозумілий інтерфейс.

В контексті вибору систем дистанційної освіти (СДО) з відкритим вихідним кодом, таких як Atutor, Claroline, Dokeos, LAMS, Moodle, OLAT, Openacs, Sakai, основними критеріями для відбору стали рівень підтримки та наявність російськомовного супроводу. SCORM стандарт служить міжнародною основою для обміну електронними курсами, тоді як специфікація сумісності питань і систем тестування IMS визначає структури даних для забезпечення сумісності інтернет-орієнтованих питань та систем тестування.

Atutor [13] як приклад, представляє собою вільно розповсюджувану веб-орієнтовану систему керування навчальним контентом, яка відрізняється своєю доступністю та адаптивністю. Ця система дозволяє адміністраторам ефективно оновлювати або встановлювати програму, а викладачі можуть легко організовувати навчальний матеріал для проведення занять в онлайн-режимі. Студенти отримують доступ до гнучкого і адаптивного середовища навчання.

"Claroline" [14] (Класна кімната в мережі) представляє собою платформу для створення веб-сайтів дистанційного навчання, розроблену з урахуванням вимог і побажань викладачів. Це безкоштовний і відкритий продукт, що працює на PHP/MySQL/Apache. Система була випробувана в середовищі Mandrake Linux та Windows і підтримує до 20000 учнів. Claroline дозволяє створювати та керувати уроками, містить генератор вікторин, форуми, календар, можливість обмеження доступу до документів, каталог посилань та систему моніторингу успіхів студентів, а також модуль авторизації.

"Dokeos" [16] є розвитком платформи Claroline, представляючи собою клон з відкритим вихідним кодом, створений з метою адаптації та модифікації оригінального додатку.

"LAMS" [18] (Система керування послідовністю навчальних дій) базується на специфікації IMS Learning Design, розробленій у 2003 році, яка включає принципи "Мови освітнього моделювання" (EML), розроблені Відкритим університетом Нідерландів (OUNL). LAMS надає викладачам інструменти для візуального проектування навчальних процесів та встановлення послідовності навчальних діяльностей.

"Moodle" [11] є системою для організації онлайн-уроків і освітніх веб-сайтів, підходящою для традиційних та гібридних стилів навчання. Система підтримує мультимовний зміст і працює на будь-якому комп'ютері, де можлива установка PHP і запуск бази даних MySQL або PostgreSQL. Веб-сайт Moodle надає безкоштовну підтримку користувачам, завдяки великій спільноті користувачів.

"Openacs" [10] (Система відкритої архітектури спільноти) є платформою для розробки масштабованих та надійних освітніх ресурсів, яка використовується багатьма компаніями та університетами для реалізації проектів в галузі електронного навчання.

"Sakai" [19] представляє собою інноваційну онлайн-платформу для організації навчального середовища. Ця система повністю ґрунтується на відкритому вихідному коді і активно розвивається громадськістю відданих розробників. Серед основних особливостей Sakai - вбудована підтримка стандартів та специфікацій IMS Common Cartridge і SCORM, що робить її ідеальним інструментом для організації навчання та сприяє високій унікальності освітнього процесу.

Підсумкова таблиця наведена в додатку А.

Забезпечення конфіденційності та захисту персональних даних користувачів у системах управління навчанням (LMS) вимагає комплексного підходу, що включає ряд ключових заходів. На першому етапі, важливо використовувати шифрування даних, як під час їх передачі, так і при зберіганні на серверах. Це забезпечує захист інформації від несанкціонованого доступу або витоку.

Управління доступом відіграє ключову роль, забезпечуючи, що лише авторизовані користувачі мають доступ до конкретних даних. Строгі процедури автентифікації та авторизації повинні бути на місці, а також обмеження доступу до даних лише для необхідних користувачів і процесів.

Забезпечення прозорої політики конфіденційності та отримання згоди користувачів на обробку їхніх даних є фундаментальними аспектами. Користувачі повинні бути ясно інформовані про те, як збираються, використовуються та зберігаються їхні дані, а також мати можливість дати свою згоду на таку обробку.

Відповідність законодавчим вимогам, таким як GDPR, є обов'язковою. Це включає дотримання встановлених правил щодо захисту даних та їх обробки. Регулярні аудити безпеки та оновлення безпеки програмного забезпечення допомагають виявляти та усувати потенційні вразливості [5-11].

Навчання персоналу основам захисту даних і найкращим практикам є ще однією важливою складовою. Підготовка співробітників допомагає підвищити загальний рівень безпеки. Також важливо мати план реагування на інциденти, щоб швидко та ефективно відповідати на будь-які проблеми з безпекою даних.

Нарешті, важливими є видалення та анонімізація даних. Системи повинні дозволяти видаляти дані за запитом користувача та використовувати анонімізацію чи псевдонімізацію для зниження ризиків пов'язаних з витоком даних. Врахування цих аспектів забезпечує, що LMS не лише захищає персональні дані користувачів, але й забезпечує їх довіру та впевненість у системі.

Функціональні вимоги, зазначені для систем управління навчанням (LMS), відіграють ключову роль у створенні ефективної, адаптивної та персоналізованої навчальної середовища. Розглянемо детально кожну з цих вимог.

Збір та аналіз користувацьких даних (результати тестувань, переваги в навчанні, історія взаємодії з системою):

– Підвищення Ефективності Навчання: Збір даних про результати тестувань та взаємодію користувачів з системою дозволяє зрозуміти, як студенти навчаються, та виявити, які методи найбільш ефективні.

– Персоналізація Досвіду Навчання: Розуміння переваг в навчанні кожного студента допомагає адаптувати навчальні матеріали та методики для кращого задоволення їхніх індивідуальних потреб.

– Покращення Навчальних Ресурсів: Аналіз даних дозволяє виявляти недоліки в навчальних матеріалах та методах, що сприяє постійному їх вдосконаленню.

Розробка моделі для прогнозування оптимального навчального шляху для кожного користувача:

– Індивідуалізація Навчального Процесу: Використання прогнозувальних моделей дозволяє розробляти індивідуалізовані навчальні шляхи, що враховують унікальні потреби та здібності кожного студента.

– Підвищення Мотивації та Залученості Студентів: Навчальні шляхи, розроблені з урахуванням інтересів та потреб студентів, збільшують їх мотивацію та залученість у навчальний процес.

– Ефективність Навчання: Оптимальні навчальні шляхи спрямовані на підвищення ефективності навчання, забезпечуючи краще засвоєння матеріалу та швидший прогрес.

Адаптація навчальних матеріалів та завдань з урахуванням індивідуальних потреб та прогресу студента:

– Різноманітність Навчальних Методів: Адаптація навчальних матеріалів та завдань дозволяє використовувати різні формати та методи навчання, що відповідають різним стилям навчання.

– Зосередження на Слабких Місцях: Адаптивні системи можуть виявляти області, в яких студенти мають труднощі, та пропонувати відповідні матеріали для покращення знань у цих областях.

– Гнучкість та Індивідуальний Підхід: Адаптація дозволяє надавати студентам можливість вивчати матеріали у власному темпі та згідно з власними інтересами, підвищуючи тим самим їхню автономію та залученість.

В цілому, ці функціональні вимоги до LMS сприяють створенню більш ефективного, гнучкого та персоналізованого навчального середовища, яке

підвищує якість освіти та задоволення користувачів системи.

1.5 Постановка задач дослідження

Технічне завдання для розробки алгоритму індивідуалізації навчального процесу на основі аналізу користувацьких даних [11-33]:

Функціональні вимоги:

– збір та аналіз користувацьких даних (результати тестувань, переваги в навчанні, історія взаємодії з системою);

– розробка моделі для прогнозування оптимального навчального шляху для кожного користувача;

– адаптація навчальних матеріалів та завдань з урахуванням індивідуальних потреб та прогресу студента.

Нефункціональні вимоги:

– висока точність та надійність прогнозувань;

– швидкість обробки даних;

– масштабованість та інтеграційність з існуючими електронними системами навчання.

Обмеження та умови:

– Забезпечення конфіденційності та захисту персональних даних користувачів.

– Сумісність з різними платформами та пристроями.

Стадії розробки:

– аналіз та проектування системи;

– розробка алгоритмічної моделі;

– тестування і валідація моделі;

– документація та підтримка.

Об'єкт дослідження - процеси індивідуалізації навчання в електронних навчальних системах.

Предмет дослідження - **а**лгоритми та методи проектування компонентів електронних навчальних систем.

Для досягнення поставленої мети роботи можна успішно досягти шляхом виконання таких завдань:

– провести аналітичний огляд алгоритмів проектування компонентів електронних навчальних систем;

– проаналізувати потреби користувачів щодо зручності та функціональності компонентів електронних навчальних систем;

– розробити алгоритми проєктування компонентів системи;

– реалізувати прототип на основі розроблених алгоритмів та провести тестування.

1.6 Висновки до розділу

Проведено аналітичний огляд алгоритмів проектування компонентів електронних навчальних систем. Систематизовано потреби користувачів щодо зручності та функціональності компонентів електронних навчальних систем.

2 РОЗРОБКА АЛГОРИТМІВ ПРОЄКТУВАННЯ

2.1 Розроблення структури навчального посібника

Підготовка електронного посібника передбачає створення його структури з урахуванням найкращих практик навчання. Ось як ми можемо оформити його розділи:

1. Вступний блок**:** Тут ми надамо коротку оглядову інформацію про курс, його цільову аудиторію та важливість змісту для успішного вивчення.

2. Теоретичні модулі**:** Кожен розділ міститиме теоретичний матеріал, організований у вигляді модулів. Кожен модуль завершується контрольними запитаннями для самоперевірки та проблемними запитаннями для індивідуальної та групової роботи.

3. Практичні та лабораторні завдання**:** Ця частина включає завдання, які необхідні для закріплення та практичного застосування отриманих знань.

4. Попередній допуск**:** Студенти можуть пройти перевірку своїх знань теоретичного матеріалу перед виконанням практичних і лабораторних завдань.

5. Глосарій**:** Довідковий матеріал із ключових термінів та понять, пов'язаних із предметом курсу.

6. Творчі завдання**:** Цей розділ міститиме завдання, спрямовані на самостійне застосування отриманих знань, навичок та умінь у реальних ситуаціях. Завдання можуть виконуватися як індивідуально, так і в групах.

Структура електронного посібника також включатиме наступні складові:

1. Реєстрація і ідентифікація студента**:** Кожен студент отримує свій особистий "логін" і "пароль", що дозволяє ідентифікувати студента та відстежувати його активність і оцінки.

2. Інформаційно-змістовий блок**:** Цей блок включає програмний зміст курсу, мультимедійні матеріали, електронні конспекти і електронний альбом схем.

Робоча програма курсу побудована на основі стандарту HTML і має зручне меню для навігації між розділами та підрозділами.

Це формулювання не лише робить текст більш унікальним, але й покращує зрозумілість і структуру посібника.

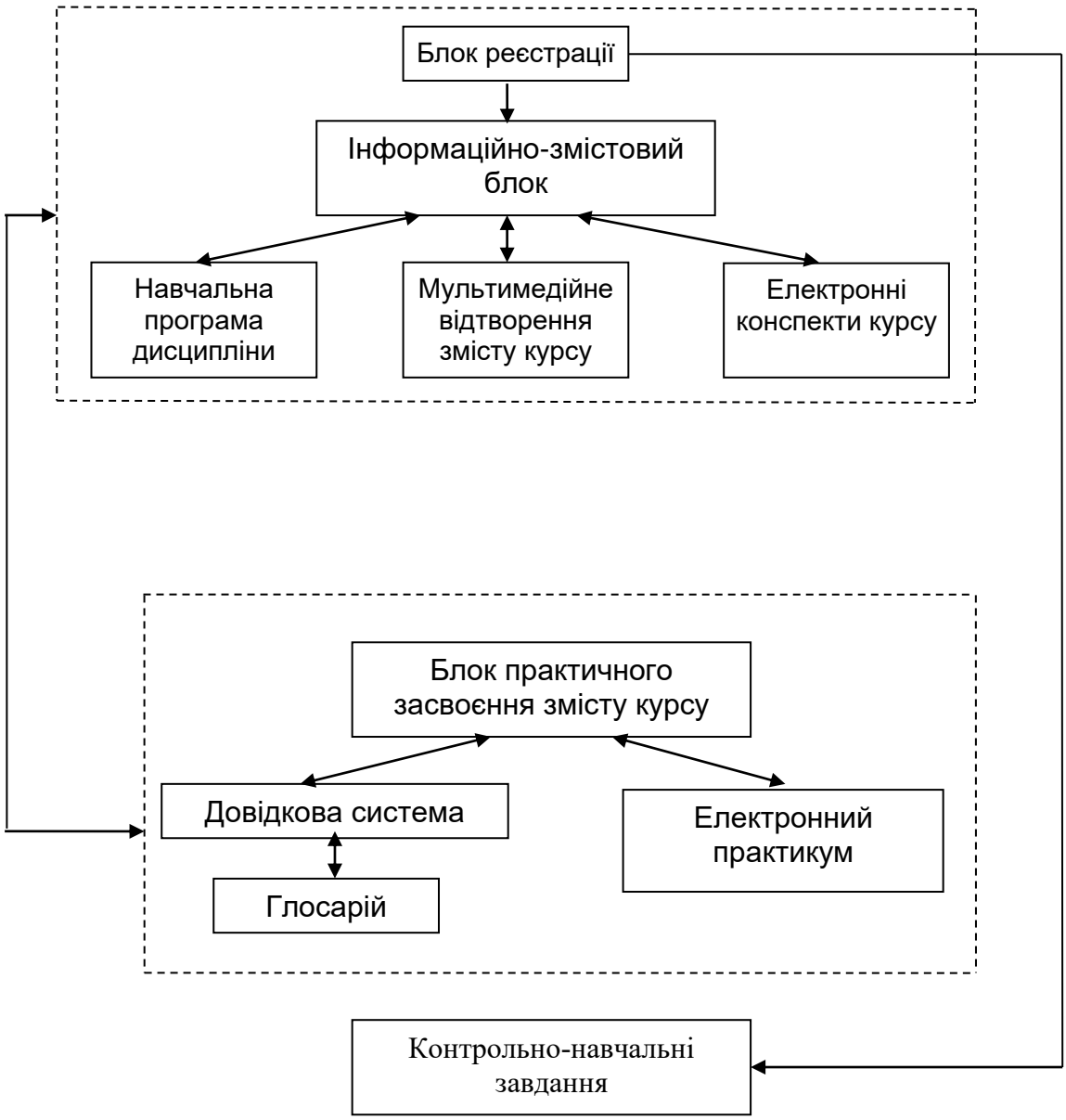

Рисунок 2.1 - Інтерактивний електронного навчального посібника, який інтегрує мультимедійні елементи для підвищення навчального досвіду.

Цей посібник включає:

Мультимедійні модулі які об'єднують аудіо та відео контент, наприклад, записи лекцій, коментарі експертів, відеоролики та інтерактивні допоміжні матеріали, що забезпечують різноманітність способів навчання.

Електронні слайди. Кожна тема курсу представлена у вигляді набору слайдів, які містять конденсовані відомості та ключові концепції, що сприяє легшому засвоєнню матеріалу.

Динамічні комп'ютерні слайди і електронний альбом схем дозволяють студентам створювати власні електронні альбоми схем, підсилюючи практичне засвоєння матеріалу.

Довідкова система та електронний практикум. Ці компоненти включають глосарії, гіпертекстові структури, навчальні теми, практичні завдання та рекомендації, що сприяють самостійному вивченню.

Контрольно-Навчальні завдання дозволяють студентам перевірити свої знання і оцінити рівень засвоєння матеріалу.

Посібник розроблений згідно з модульним принципом, де кожен модуль є відносно самостійною частиною інформації, що може бути використаний для тестування та самооцінки знань (рисунок 2.2).

Модуль у цифровому посібнику є навчальним шаром, на якому можна впровадити ідею багаторівневого навчання. Під час створення посібника викладач встановлює критерії для різних рівнів навчання. Перший рівень, пов'язаний з конкретною темою, є найлегшим і вибирається самим студентом. Подальші рівні модуля в електронному посібнику залежать від успішності засвоєння навчального матеріалу та керуються системою навчання [29-32]. Якщо модуль має кілька рівнів, то викладач створює окремі рівні-шари змісту модуля і відповідні тести для самоперевірки та підсумкового оцінювання.

Після успішного завершення підсумкового тестування модуля система може запропонувати студенту більш високий рівень для вивчення наступного модуля у цифровому посібнику. У випадку неуспішного тестування студента може залишити на тому ж рівні або відправити на повторне вивчення модулю. Схема адаптації навчального процесу наведена на рисунку 2.3.

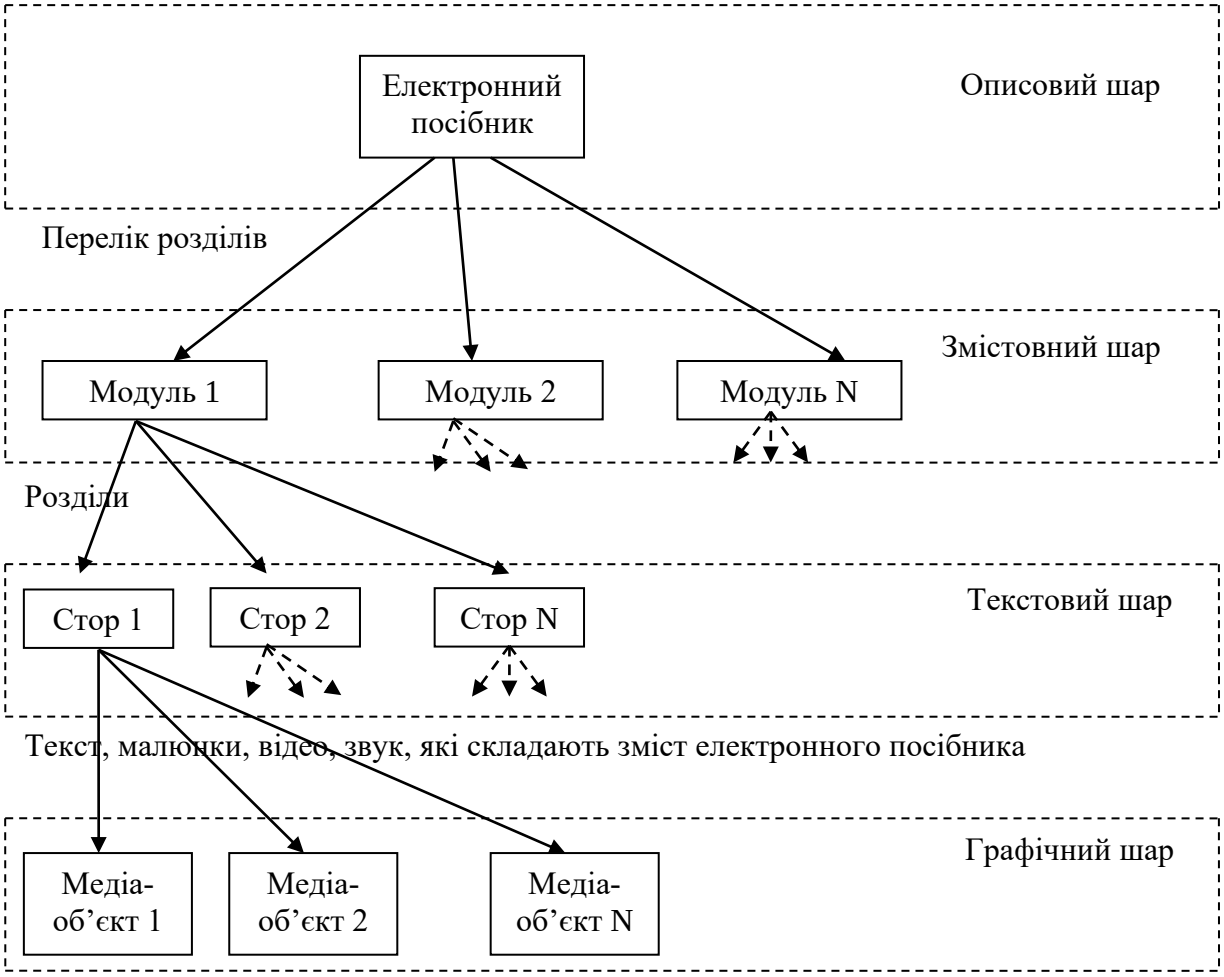

Рисунок 2.2 – Структурна схема електронного посібника

Отже, в електронних посібниках присутня унікальна можливість реалізації модульного підходу до структури навчального матеріалу, що дозволяє комплексно представити інформацію. Електронні посібники виділяються своєю візуальною насиченістю, оскільки, окрім текстового контенту, вони містять різноманітний мультимедійний матеріал. Перевагою електронних посібників є можливість працювати з віддаленими ресурсами та легкість переходу між різними їх частинами. Студентам надається можливість вибору рівня складності, що сприяє підвищенню ефективності навчання, адаптуючи навчальний процес до їхніх потреб і можливостей.

Рисунок 2.3 – Блок-схема алгоритму адаптації електронного посібника за рівнем навчання

При розробці цифрових освітніх матеріалів, важливо врахувати відмінності традиційного навчання від індивідуального. Основним завданням є уникнення студентської пасивності та стимулювання активного засвоєння знань. Ефективність досягається за допомогою двох ключових дидактичних принципів:

1. Лектор повинен подавати матеріал науково обґрунтовано, живо та захоплююче;

2. Усна подача знань має включати педагогічні методи, що підвищують розумову активність та утримують увагу студентів.

Одним із таких методів є створення проблемних ситуацій, які вимагають чіткого визначення теми та ключових питань для дослідження. Дистанційне навчання ефективно вирішує питання активності студентів, включаючи елементи як:

- Демонстрації;
- Практичні завдання.

Періодичне оцінювання є невід'ємною частиною освітнього процесу. Методи оцінки включають:

- Поточні опитування;
- Модульне тестування та опитування;
- Презентації результатів;
- Оцінювання проектів;
- Студентські виступи на наукових заходах;
- Лабораторні завдання;
- Контрольні роботи та екзамени.

Структура електронного посібника передбачає:

1. Вступ із оглядом курсу;
2. Короткий теоретичний матеріал, поділений на модулі з контрольними та проблемними запитаннями;
3. Лабораторні роботи;
4. Модульні тестові завдання та екзамени;
5. Довідкові матеріали.

Електронний посібник базується на принципі модульності, де кожен модуль - це окрема інформаційна одиниця з можливістю самотестування. Модульність дозволяє студентам самостійно обирати рівень складності навчання, що сприяє кращому засвоєнню матеріалу.

2.2 Розроблення бази даних для ресурсів електронного підручника

Створимо модель електронного підручника який має такі властивості:

1) містить Мультимедійні ілюстрації, включаючи графіку, анімацію, аудіо та відео;

2) містить практичні завдання;

3) містить тестові завдання та групові вправи;

4) адаптований для створення індивідуального навчального шляху.

Створення ER-діаграми (Entity-Relationship Diagram) для електронного підручника . Детально розглянемо кожну сутність та її атрибути у цій ER-діаграмі для електронного підручника:

1. Textbook (Підручник)
   - id: Унікальний ідентифікатор підручника.
   - title: Назва підручника.
   - subject: Предмет, якому відповідає підручник.
   - level: Рівень навчання, на який розрахований підручник (наприклад, початкова школа, середня школа тощо).
2. Multimedia (Мультимедіа)
   - id: Унікальний ідентифікатор мультимедійного ресурсу.
   - type: Тип мультимедійного контенту (графіка, анімація, аудіо, відео).
   - url: URL-адреса, де розміщено мультимедійний ресурс.
3. PracticalTask (Практичне Завдання)
   - id: Унікальний ідентифікатор практичного завдання.
   - description: Опис завдання, яке потрібно виконати.
4. Test (Тест)
   - id: Унікальний ідентифікатор тестового завдання.
   - question: Тестове питання.
   - correctAnswer: Правильна відповідь на тестове питання.
5. GroupExercise (Групова Вправа)

– id: Унікальний ідентифікатор групової вправи.

– description: Опис групової вправи.

6. LearningPath (Навчальний Шлях)

– id: Унікальний ідентифікатор навчального шляху.

– name: Назва навчального шляху.

Зв'язки між сутностями:

– Підручник і мультимедіа: Кожен підручник може містити багато мультимедійних елементів, але кожен мультимедійний елемент належить лише одному підручнику.

– Підручник і практичне завдання: Кожен підручник може включати в себе багато практичних завдань. Кожне практичне завдання асоціюється з одним підручником.

– Підручник і тест: Кожен підручник може містити різні тестові завдання. Кожне тестове завдання пов'язане з одним підручником.

– Підручник і групова вправа: Кожен підручник може включати багато групових вправ. Кожна групова вправа асоціюється з конкретним підручником.

– Підручник і навчальний шлях: Підручник може бути адаптований для різних навчальних шляхів. Це означає, що вміст підручника може бути налаштований або вибірково використаний для створення унікального навчального шляху.

У цій діаграмі:

– Textbook - основна сутність, яка представляє електронний підручник.

– Multimedia, PracticalTask, Test, і GroupExercise - сутності, які представляють різні типи контенту в підручнику.

– LearningPath - сутність, яка відображає індивідуальні навчальні шляхи, адаптовані до підручника.

Кожен тип контенту (мультимедійні ілюстрації, практичні та тестові завдання, групові вправи) має відношення "один до багатьох" з підручником, що означає, що один підручник може містити багато елементів кожного типу.

Сутність LearningPath асоційована з підручником, показуючи, що підручник може бути адаптований для різних навчальних шляхів.

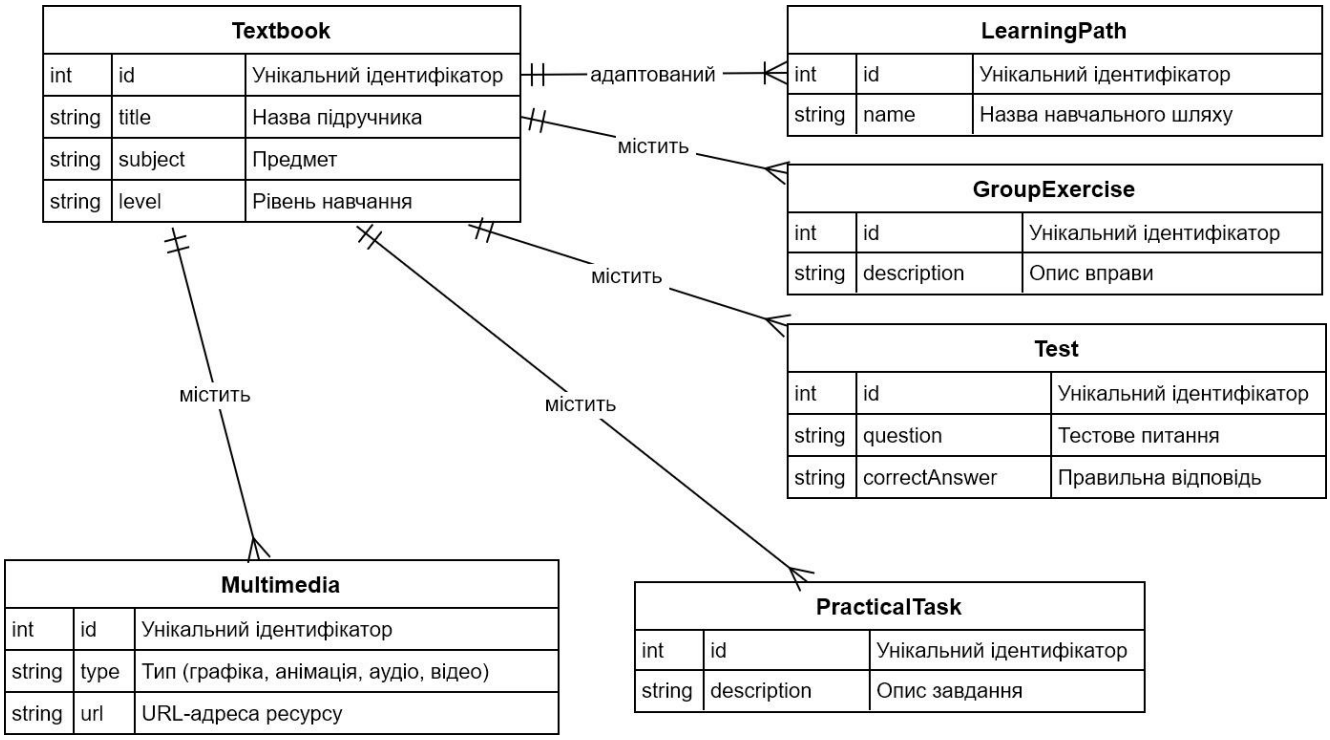

Рисунок 2.4 – Діаграма бази даних для електронного підручника

2.2 Розроблення об’єктної моделі програмного забезпечення

Отже функціональні вимоги:

- 1) Збір та аналіз користувацьких даних (результати тестувань, переваги в навчанні, історія взаємодії з системою).
- 2) Розробка моделі для прогнозування оптимального навчального шляху для кожного користувача.
- 3) Адаптація навчальних матеріалів та завдань з урахуванням індивідуальних потреб та прогресу студента.

Для розробки об'єктної моделі програми, що включає зазначені функції, можна розглянути наступні основні класи та їхні взаємозв'язки:

1. Клас User

– Атрибути: ID користувача, ім'я, історія взаємодій, результати тестувань, переваги в навчанні.

– Методи: Збір даних користувача, отримання історії взаємодій, оцінка результатів тестувань.

2. Клас LearningMaterial:

– Атрибути: ID матеріалу, тип (лекція, тест, відео), вміст.

– Методи: Адаптація вмісту до потреб користувача.

3. Клас LearningPath:

– Атрибути: Послідовність навчальних матеріалів, прогрес користувача.

– Методи: Генерація оптимального навчального шляху, оновлення прогресу користувача.

4. Клас DataAnalyzer:

– Методи: Аналіз користувацьких даних, виявлення тенденцій та переваг у навчанні.

5. Клас PredictionModel:

– Методи: Прогнозування оптимального навчального шляху на основі даних користувача та аналізу.

Ці класи взаємодіють для створення індивідуалізованого навчального досвіду, де DataAnalyzer збирає та аналізує дані користувача, PredictionModel використовує ці дані для прогнозування навчального шляху, а LearningPath та LearningMaterial адаптують навчальні матеріали з урахуванням індивідуальних потреб та прогресу студента.

Методи і змінні для кожного з класів:

1. Клас User. Змінні:

– userID: унікальний ідентифікатор користувача.

– name: ім'я користувача.

– interactionHistory: історія взаємодій з системою.

– testResults: результати тестувань.

– learningPreferences: переваги в навчанні.

- Методи:

– collectData(): збір даних про користувача.

– getInteractionHistory(): отримання історії взаємодій.

– evaluateTestResults(): оцінка результатів тестувань.

2. Клас LearningMaterial. Змінні:

– materialID: унікальний ідентифікатор матеріалу.

– type: тип матеріалу (лекція, тест, відео).

– content: вміст матеріалу.

- Методи:

– adaptContent(User user): адаптація вмісту до потреб конкретного користувача.

3. Клас LearningPath. Змінні:

– sequenceOfMaterials: послідовність навчальних матеріалів.

– userProgress: прогрес користувача в навчанні.

- Методи:

– generateOptimalPath(User user): генерація оптимального навчального шляху для користувача.

– updateProgress(Material material): оновлення прогресу користувача після проходження навчального матеріалу.

4. Клас DataAnalyzer. Методи:

– analyzeData(User user): аналіз даних користувача.

– identifyTrends(): виявлення тенденцій та переваг у навчанні.

5. Клас PredictionModel. Методи:

– predictPath(User user, DataAnalyzer analyzer): прогнозування оптимального навчального шляху на основі даних користувача та аналізу.

Функція evaluateTestResults оцінює результати тестувань користувача. Вона аналізує, як користувач виконував різні тести або завдання, і може використовувати ці дані для визначення сильних та слабких сторін користувача у навчанні. Це допомагає зрозуміти рівень знань та навичок користувача.

Функція predictOptimalLearningPath функція використовується для прогнозування найбільш підходящого шляху навчання для користувача. Вона аналізує загальну інформацію про користувача, включаючи його історію взаємодій, переваги в навчанні та результати тестів, щоб визначити, які курси або матеріали будуть найкориснішими для його подальшого розвитку.

Діаграма класів на рисунку.

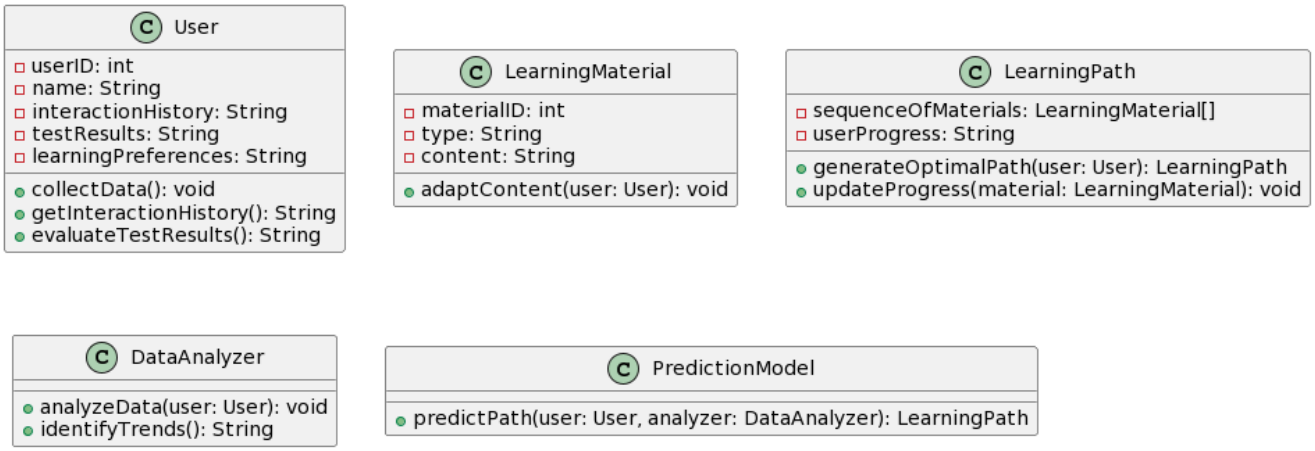

Рисунок 2.5 – Діаграма класів ПЗ

Для системи навчання, яка акцентується на адаптації навчального матеріалу та шляху до індивідуальних потреб та переваг користувачів, основні типи користувачів можуть включати:

1. Студенти - Основні користувачі системи, які використовують навчальні матеріали для свого освітнього прогресу.
2. Викладачі/Тренери - Професіонали, які підготовлюють та надають навчальний контент та вправи.
3. Адміністратори Курсу - Відповідають за управління курсами, включаючи структуру, доступність та якість навчальних матеріалів.
4. Аналітики Даних - Спеціалісти, які аналізують дані користувачів для поліпшення та адаптації навчальних шляхів.

Можливі сценарії використання (Use Cases):

1. Персоналізоване навчання для студентів

– Ціль: Адаптувати навчальний процес до індивідуальних потреб та стилів навчання студентів.

– Основні Кроки:

- Студент входить в систему та виконує тест на визначення стилю навчання.
- Система аналізує результати та визначає переваги студента.
- Генерація персоналізованого навчального шляху на основі аналізу.

2. Підготовка адаптивних навчальних матеріалів викладачами

– Ціль: Створення гнучких навчальних матеріалів, які можуть бути адаптовані під різні потреби.

– Основні Кроки:

- Викладач підготовлює матеріали з можливістю адаптації (різні формати, рівні складності).
- Матеріали завантажуються в систему для використання студентами.

3. Моніторинг та оцінка прогресу студентів адміністраторами курсу

– Ціль: Відстеження успішності та прогресу студентів у навчальних курсах.

– Основні Кроки:

- Адміністратор переглядає звіти про успішність студентів.
- Вносяться необхідні корективи в курси для покращення навчального процесу.

4. Аналіз взаємодій студентів з матеріалами аналітиками даних

– Ціль: Виявлення тенденцій та покращення матеріалів на основі аналізу даних.

– Основні Кроки:

а)Аналіз даних про взаємодії студентів з матеріалами.

б)Виявлення патернів та внесення рекомендацій для оптимізації контенту.

5. Розробка моделей прогнозування для персоналізованих навчальних шляхів

Ціль: Створення алгоритмів для прогнозування найефективніших навчальних шляхів.

Основні Кроки:

а) Використання історичних даних для розробки моделей машинного навчання.

б) Використання моделей для прогнозування оптимальних навчальних шляхів для нових студентів.

Ці сценарії використання демонструють, як різні користувачі системи можуть взаємодіяти з нею, забезпечуючи адаптивний та ефективний навчальний досвід. Use case діаграма для описаних п’яти випадків використання:

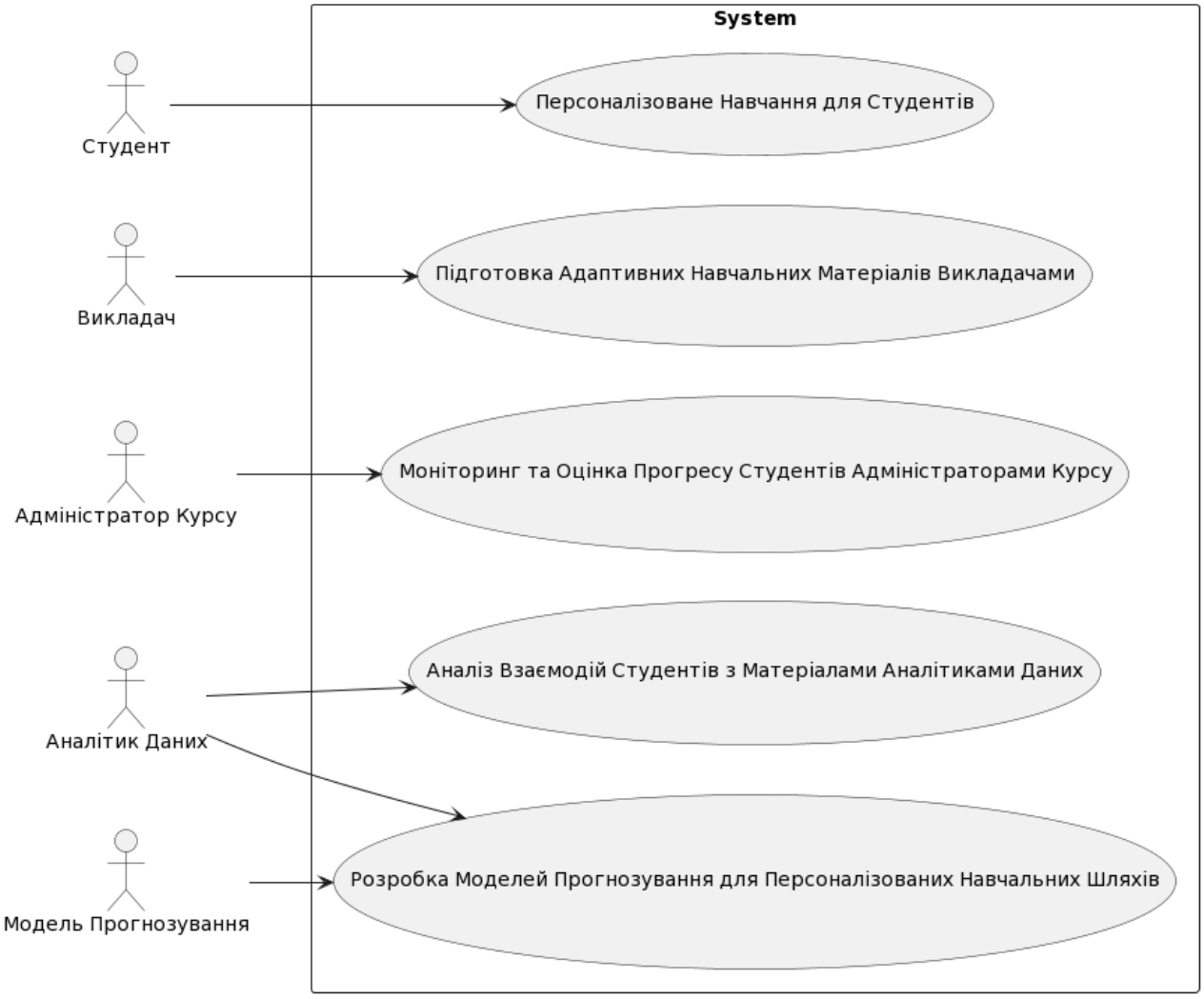

Рисунок 2.6 - Use case діаграма програмної системи

Ця діаграма включає такі елементи:

Актори: Студент, Викладач, Адміністратор Курсу, Аналітик Даних, Модель Прогнозування.

Випадки використання:

– Персоналізоване навчання для студентів.

– підготовка адаптивних навчальних матеріалів викладачами.

– моніторинг та оцінка прогресу студентів адміністраторами курсу.

– аналіз взаємодій студентів з матеріалами аналітиками даних.

– розробка моделей прогнозування для персоналізованих навчальних шляхів.

2.3 Алгоритм проектування компонентів електронних навчальних систем

Моделі оцінки результатів тестувань для визначення навчального шляху:

1. Проста середня оцінка (Simple Average Score)**:** Розрахунок середнього балу з усіх результатів тестувань.

2. Вагова середня оцінка (Weighted Average Score)**:** Врахування важливості кожного тесту або теми, присвоюючи вищу вагу більш значущим тестам.

3. Аналіз відхилень (Deviation Analysis)**:** Визначення областей, де користувач перевершує або відстає, порівнюючи з середніми показниками або очікуваними результатами.

4. Оцінка прогресу (Progress Assessment)**:** Відстеження змін у результативності користувача протягом часу, щоб оцінити його навчальний прогрес.

5. Кластерний аналіз (Cluster Analysis)**:** Групування результатів тестувань для виявлення шаблонів або груп користувачів зі схожими результатами.

6. Статистичний аналіз (Statistical Analysis)**:** Застосування статистичних методів, таких як стандартне відхилення, для оцінки розподілу та варіативності результатів.

7. Адаптивна оцінка (Adaptive Assessment)**:** Адаптування оцінки залежно від індивідуальних цілей та потреб користувача, враховуючи його попередні досягнення та особливості навчання.

Різні моделі оцінки результатів тестувань можна буде використати в методі evaluateTestResults в класі User (див 2.2, 3.1).

Інтеграція компонентів.

Для розробки моделі планування навчального шляху, сумісної з LMS Moodle, яка враховує неправильні відповіді на тести і пропонує навчальні компоненти для вивчення відповідних тем, можна використати таку структуру:

1. Навчальний Компонент (Learning Component)
   – Тип: Тест або практичне завдання.
   – Функція: Оцінити розуміння студентом певної теми.
2. Аналізатор відповідей (Answer Analyzer)
   – Функція: Аналізує відповіді студентів на тест чи завдання.
   – Дія: Якщо відповідь неправильна, визначає відповідну тему для подальшого вивчення.
3. Менеджер навчального контенту (Learning Content Manager)
   – Функція: Вибирає та пропонує навчальний контент, який пояснює виявлену проблемну тему.
   – Тип Контенту: Лекції, відео, текстові матеріали, інтерактивні завдання.
4. Функція повторного тестування (Retesting Function)
   – Функція: Після вивчення пропонує повторний тест або практичне завдання для перевірки розуміння теми.

Ця модель може бути інтегрована в Moodle за допомогою плагінів або спеціалізованих розширень, що дозволяють автоматизувати процес навчання, реагуючи на результати тестів студентів та адаптуючи навчальний шлях відповідно до їхніх потреб.

Алгоритм інтеграції:

1. Отримання Відповідей з LMS Moodle:
   – Система отримує відповіді студентів на тести або практичні завдання з LMS Moodle.
2. Аналіз Відповідей:

– Answer Analyzer проаналізовує відповіді студентів.

– Для кожної неправильної відповіді визначається відповідна тема для додаткового вивчення.

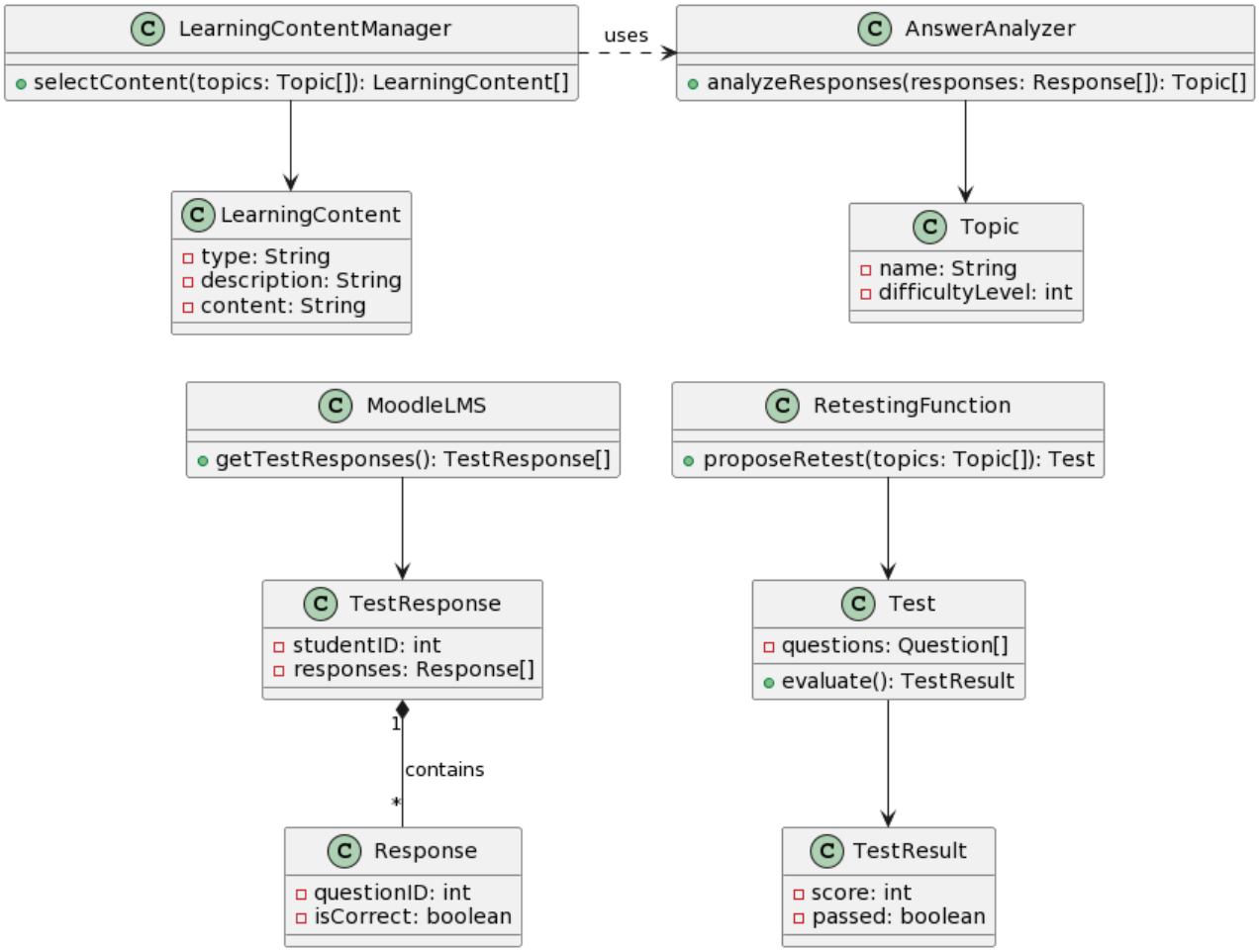

Рисунок 2.7 - Діаграма класів для алгоритму планування навчального шляху

3. Вибір Навчального Контенту:

– Learning Content Manager отримує інформацію про проблемні теми.

– Вибирає та пропонує відповідний навчальний контент (лекції, відео, текстові матеріали, інтерактивні завдання).

4. Вивчення Матеріалів Студентами:

– Студенти вивчають запропонований контент.

5. Повторне Тестування:

– Retesting Function пропонує повторний тест або практичне завдання.

– Перевіряється розуміння теми після вивчення матеріалів.

6. Оцінка Прогресу та Зворотній Зв'язок:

– Після повторного тестування система оцінює прогрес студента та надає зворотній зв'язок.

Ця діаграма класів представляє структуру системи для планування навчального шляху. Вона включає класи для взаємодії з LMS Moodle, аналізу відповідей, вибору навчального контенту, та функціоналу для повторного тестування.

Взаємодія між компонентами моделі планування навчального шляху.

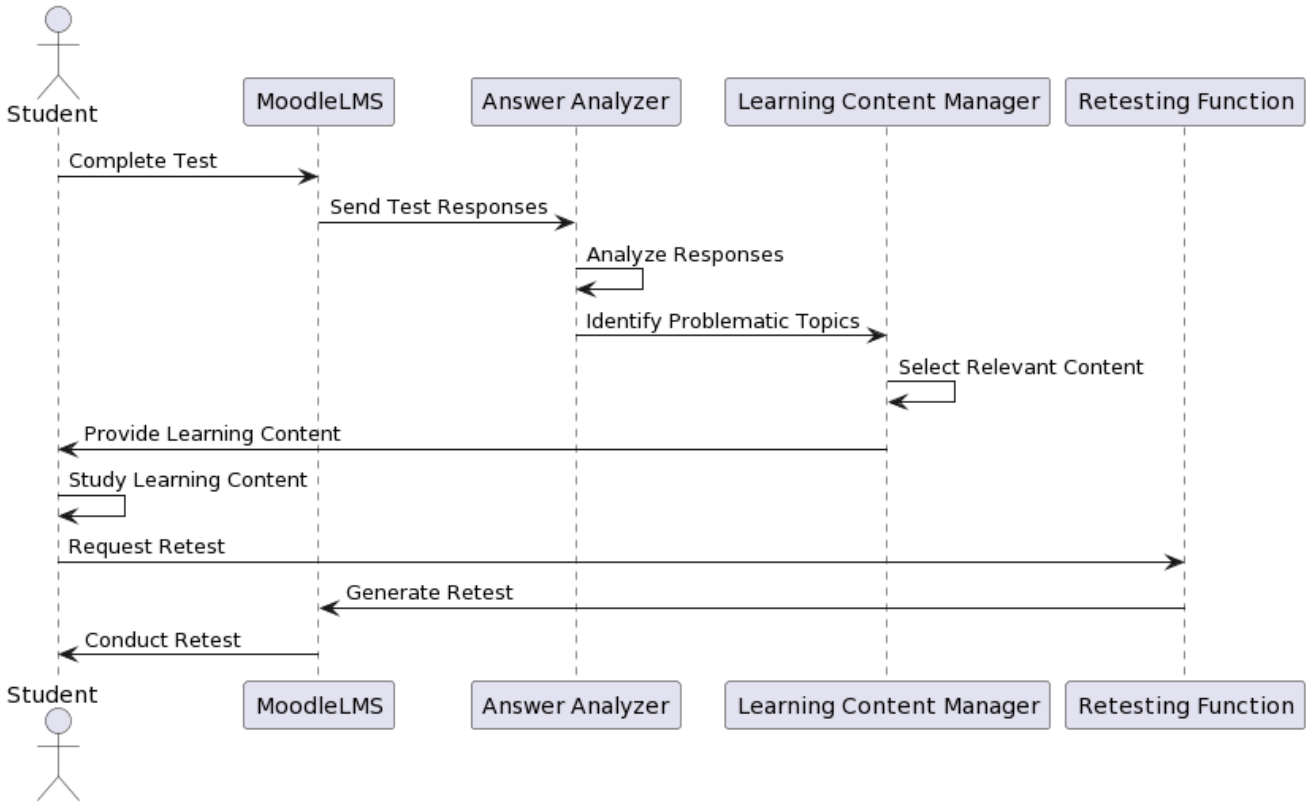

Рисунок 2.8 - Діаграма послідовності алгоритму

Ця діаграма послідовності відображає наступні кроки:

1. Студент завершує тест у MoodleLMS.
2. MoodleLMS відправляє відповіді на тест до Answer Analyzer.
3. Answer Analyzer аналізує відповіді та визначає проблемні теми.

4. Answer Analyzer передає інформацію про проблемні теми до Learning Content Manager.
5. Learning Content Manager вибирає відповідний навчальний контент.
6. Learning Content Manager надає студенту навчальний контент.
7. Студент вивчає наданий контент.
8. Студент запитує повторний тест через Retesting Function.
9. Retesting Function генерує повторний тест у MoodleLMS.
10. MoodleLMS проводить повторний тест для студента.

2.4 Висновки до розділу

Розроблено структуру інтерактивного навчального посібника, який інтегрує мультимедійні елементи для підвищення навчального досвіду. Посібник розроблений згідно з модульним принципом, де кожен модуль самостійною частиною, що може бути використаний для тестування та самооцінки знань. Розроблено алгоритм адаптації електронного посібника за рівнем навчання

3 ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ

3.1 Розроблення архітектури системи дистанційного навчання

Moodle, скорочення для "Modular Object-Oriented Dynamic Learning Environment," представляє собою комплексний системний інструмент для управління освітнім контентом (LCMS). Ця платформа надає можливість розробки цифрових освітніх курсів і реалізації як очного, так і дистанційного навчання.

Розробником цієї концепції є австралійський освітній теоретик Мартін Доугіамас. Його амбіцією було створити унікальну платформу, яка б відрізнялася від існуючих на ринку альтернатив, з особливим акцентом на педагогічні аспекти. Основою Moodle лягла теорія конструктивізму з психології навчання, яка передбачає активну роль студента у формуванні власної системи знань через використання різноманітних ресурсів. Водночас, вчитель або тьютор виступає у ролі мотиватора та підтримки, організовуючи задачі для самостійної роботи, оцінюючи результати та коригуючи знання.

Завдання, розроблені в Moodle, сприяють розвитку нових знань у студентів, особливо в умовах колаборативного навчання, де студенти працюють у групах, діляться досвідом та ідеями, будучи відкритими до думок інших.

Особливістю платформи Moodle є її постійний розвиток з 1999 року, включаючи регулярні оновлення, доповнення новими функціями та інструментами. Платформа написана на мові PHP і використовує безкоштовні бази даних, такі як MySQL або PostgreSQL. Moodle сумісний з різними операційними системами, включаючи MS Windows, Unix та Linux.

Платформа Moodle оснащена різноманітними модулями, які підтримують ефективну взаємодію між студентами та між студентами та викладачами. Включені модулі охоплюють:

№	Модуль
1	опитування
2	анкети
3	чати
4	форуми
5	уроки
6	журнали
7	тести
8	пакети SCORM
9	словники
10	семінари
11	wiki
12	завдання

Рисунок 3.1 - Модулі

Однією з ключових особливостей Moodle є його офіційний веб-сайт, який служить центральним ресурсом для інформації про платформу. Він також функціонує як форум для обміну ідей і співпраці серед спільноти користувачів Moodle, включаючи системних адміністраторів, викладачів, дослідників, дизайнерів та розробників. Підтримка інтерфейсу Moodle на понад 80 мовах, у тому числі українською, розширює його глобальну доступність. Станом на кінець 2010 року, Moodle активно використовувався в понад 60 тисячах освітніх організацій у більш ніж 200 країнах світу.

Для запуску системи управління навчальним контентом Moodle необхідний веб-сервер, що підтримує PHP, та сервер бази даних, такий як MySQL або PostgreSQL. Зазвичай використовується комбінація Apache та MySQL. Рекомендовано використовувати Unix-подібні операційні системи, наприклад, GNU/Linux або FreeBSD, хоча можливе використання MacOSX від Apple або Windows від Microsoft. Останній варіант є менш ефективним для ролі веб-сервера Moodle та може мати деякі обмеження у функціональності. Moodle створено згідно зі стандартною інтернет-архітектурою, використовуючи веб-браузери як клієнтське програмне забезпечення.

Інтеракція користувачів з системою через інтернет відображена на діаграмі 3.2.

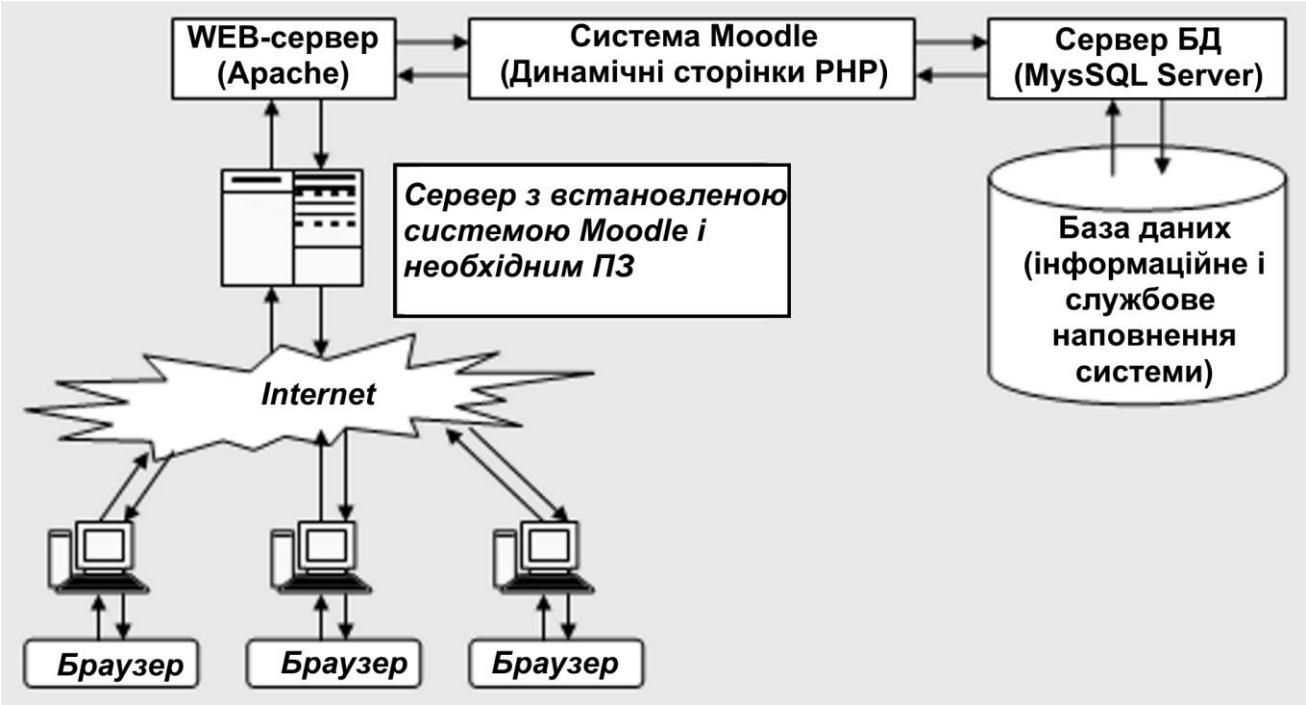

Рисунок 3.2 – Схема системи

Ключовою перевагою такої архітектури є її здатність до широкого розповсюдження інформації в мережі, забезпечуючи доступність системи кожному, хто має підключення до Інтернету та операційну систему, сумісну з веб-контентом. Це дозволяє користуватися системою не тільки в домашніх умовах, але й в навчальних закладах, під час лабораторних чи семінарських занять. Ще одна перевага - зосереджене адміністрування лише серверної частини системи. Встановлення одного комплекту програмного забезпечення на сервер дозволяє використовувати його для різних курсів та на домашніх комп'ютерах викладачів і студентів, зменшуючи потребу у постійному оновленні та адмініструванні.

Сама система вимагає невеликих ресурсів від серверної та клієнтської частини. Початково Moodle було розроблено в середовищі Linux з використанням Apache, MySQL і PHP (LAMP), але вона також була протестована і сумісна з Windows (WAMP), Solaris та Mac OS X. Moodle підтримує СУБД PostgreSQL, Oracle та Microsoft SQL Server. Схема розгортання системи дистанційного навчання на рисунку 3.3.

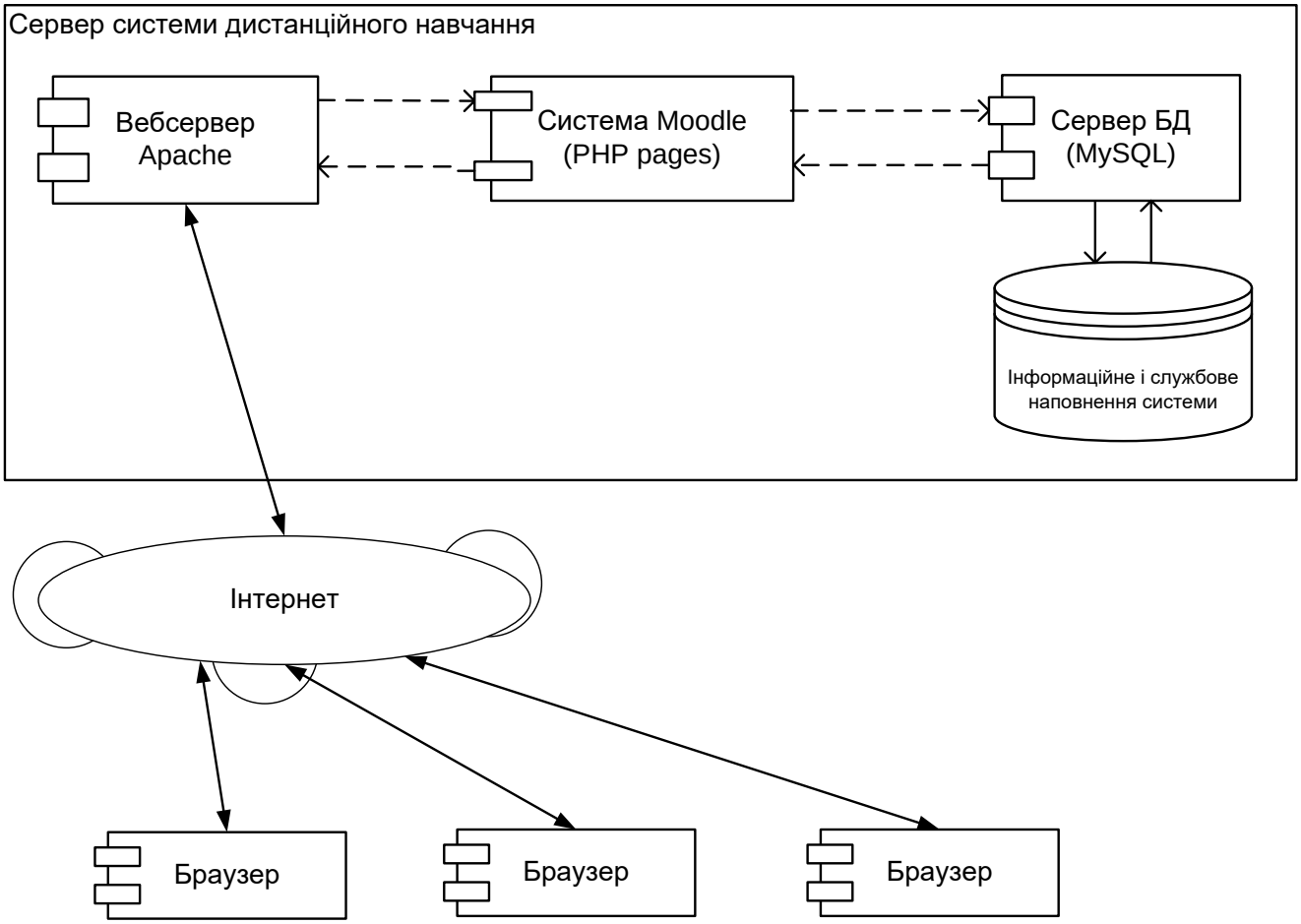

Рисунок 3.3 – Схема розгортання

Щодо технічних вимог для роботи Moodle, система потребує наступного обладнання: мінімум 160 МБ вільного місця на диску, хоча потреба збільшується для зберігання навчального контенту. Мінімальний обсяг пам'яті - 256 МБ, рекомендований - 1 ГБ. Приблизний обсяг пам'яті розраховується за правилом: 50 одночасно активних користувачів на кожен 1 ГБ пам'яті, враховуючи можливі відхилення залежно від комбінації обладнання та програмного забезпечення.

Технічні вимоги до програмного забезпечення

2.1 Веб-сервер: Moodle ефективно працює з Apache, але також сумісний з іншими веб-серверами, що підтримують PHP, включаючи IIS на платформі Windows. Рекомендується використання найновіших і стабільних версій веб-серверів.

2.2 PHP: Moodle підтримує PHP версії 7.3 і вище. Залежно від версії Moodle, вимоги можуть варіюватися.

3. Сервер СУБД: Moodle повністю підтримує MySQL і PostgreSQL. З Moodle 1.7 додано підтримку Microsoft SQL Server та Oracle. Для Moodle 4 необхідний MySQL версії 5.7 і вище, PostgreSQL 9.6 і вище, або Microsoft SQL Server 2017. Важливо, що для MySQL повинен бути вимкнений "strict mode". Рекомендовані версії PostgreSQL - 9.6 та вище.

Додатково до основних вимог Moodle, необхідно визначити потенційне навантаження на систему:

– Активні користувачі: максимальна кількість одночасних користувачів на СДО.

– Конкурентні користувачі БД: максимальне число користувачів, які одночасно працюють з базою даних.

Ці показники важливі для оцінки здатності системи витримувати очікуване навантаження. Вплив на продуктивність залежить від обладнання, програмного забезпечення і швидкості мережевого з'єднання. Загальне правило: максимальна кількість конкурентних користувачів БД становить 50 користувачів на кожен 1 ГБ оперативної пам'яті сервера, а максимальна кількість активних користувачів - у п'ять разів більше.

Давайте розглянемо структуру та вміст електронного навчального матеріалу, присвяченого темі "Комп’ютерні мережі". На рисунку представлено деталізацію електронних ресурсів та відповідні формати, які використовуються в середовищі Moodle, відносно кожного елементу цього посібника.

3.2 Створення навчального ресурсу

Для тестування розроблених алгоритмів створимо в системі Moodle дисципліну на тему «Комп’ютерні мережі» .

Для ініціювання процесу додавання контенту та редагування курсу в системі Moodle, користувачам слід вибрати опцію «Редагувати» у розділі «Керування» або скористатися відповідною зеленою кнопкою у верхній частині

інтерфейсу. Це дія відкриває сторінку, на якій можна внести зміни та додати навчальні матеріали.

Складові частини ЕНК		Електронний формат	Формат MOODLE
Загальна інформація	Візитка курсу, оголошення	html	ресурс «Веб-сторінка»
	довідник для студента	html	ресурс «Веб-сторінка»
		pdf	ресурс «Посилання на файл»
	робоча програма, графік навчання та консультацій	html	ресурс «Веб-сторінка»
		pdf	ресурс «Посилання на файл»
Тематичні розділи	методичні рекомендації	html	ресурс «Книга»
	лекційний матеріал	html	ресурс «Веб-сторінка»
		pdf	ресурс «Посилання на файл»
	наочні матеріали - зображення - відеоролик - анімація - аудіоролик - презентація	 jpeg, png, gif flv,wmv,mp4, avi flv mp3, wav ppt, pptx	 ресурс «Посилання на файл» ресурс «Посилання на файл» ресурс «Посилання на файл» ресурс «Посилання на файл» ресурс «Посилання на файл»
	глосарій	-	елемент «Глосарій»
	тестові завдання	-	елемент «Тест»

Рисунок 3.4 - Формати файлів елементів підручника

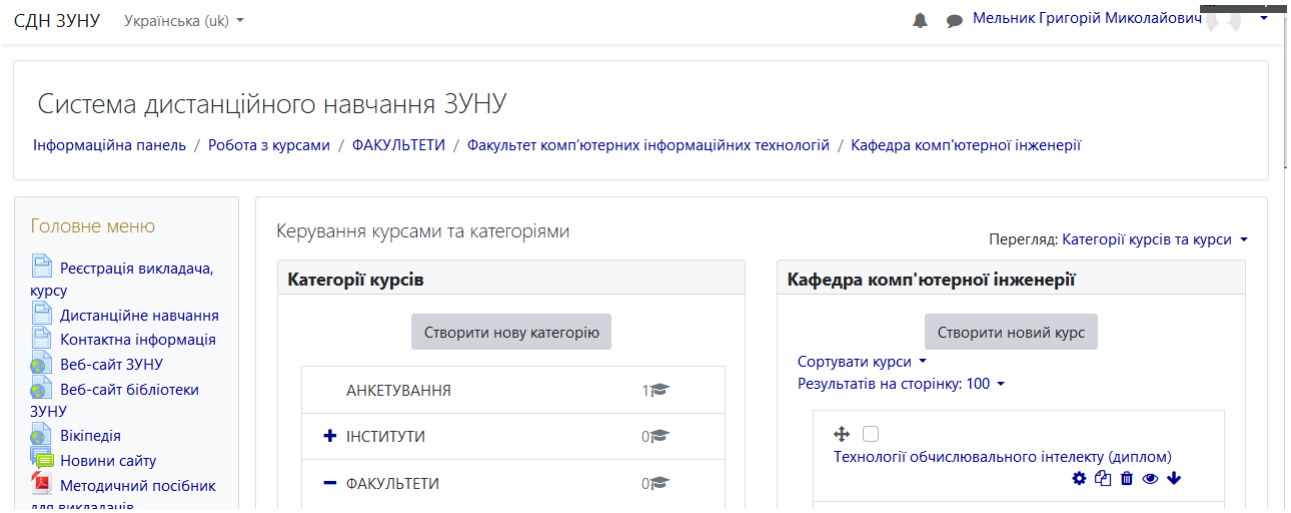

Рисунок 3.5 - представлено меню для керування дисциплінами в рамках окремої кафедри університету

У розділі «Керування» (рисунок 3.5) вибираємо «Редагувати параметри» для переходу на сторінку налаштувань дисципліни. Тут можна вказати повну та скорочену назви курсу, встановити дату початку та інші важливі параметри.

На сторінці редагування в рамках конкретної теми обираємо опцію «Додати діяльність або ресурс» (ілюстрація 3.6). За потреби можна змінювати кількість тем, додаючи або зменшуючи їх.

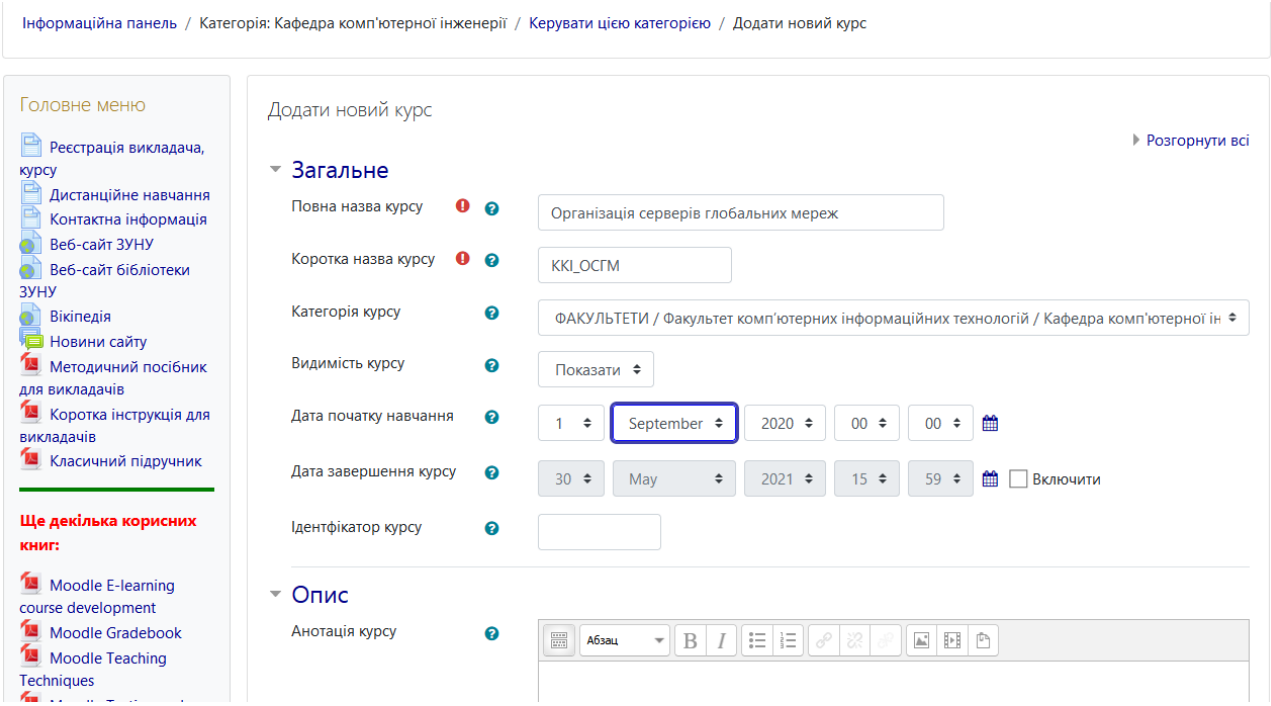

Рисунок 3.5 - Процес створення нового курсу.

Під час додавання матеріалів, у вікні, що з'явиться (ілюстрація 3.7), вибираємо тип ресурсу «Файл» (або «Тека», за необхідності) і натискаємо на «Додати». У вказаних полях вводимо назву та опис для навчального матеріалу та завантажуємо файл у форматі PDF з комп'ютера (ілюстрації 3.8-3.9).

Після натискання на «Зберегти й повернутися до курсу», відбувається перехід саме на головну сторінку дисципліни, де видно доданий ресурс. Вибір «Зберегти і показати» прямо відкриває завантажений файл.

Для додавання наступних матеріалів повторюємо вищевказані дії, включаючи програму дисципліни, методичні вказівки, лекційні матеріали, бібліографічні списки та посилання на електронні ресурси, глосарій тощо.

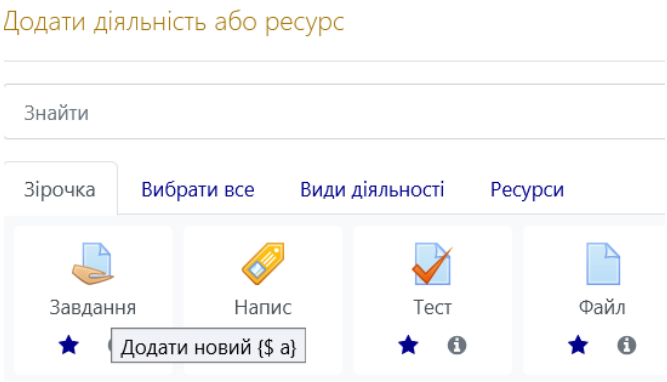

Рисунок 3.7 – Створення ресурсу певного типу

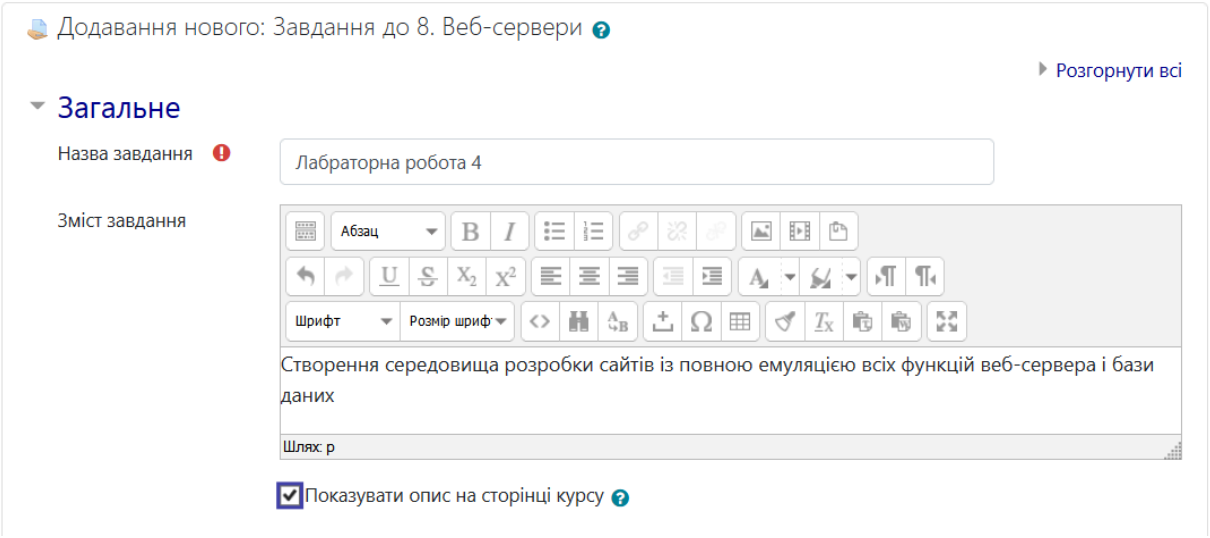

Рисунок 3.8 – Ресурс «завдання» для лабораторного заняття

Лекції та лабораторні роботи заносяться аналогічно, із завантаженням відповідних документів у PDF-форматі, а також додаткових інструкцій і критеріїв оцінювання (рисунок 3.9). Структура посібника в Moodle включає всі лабораторні роботи на весь період навчання та графіки їх оцінювання.

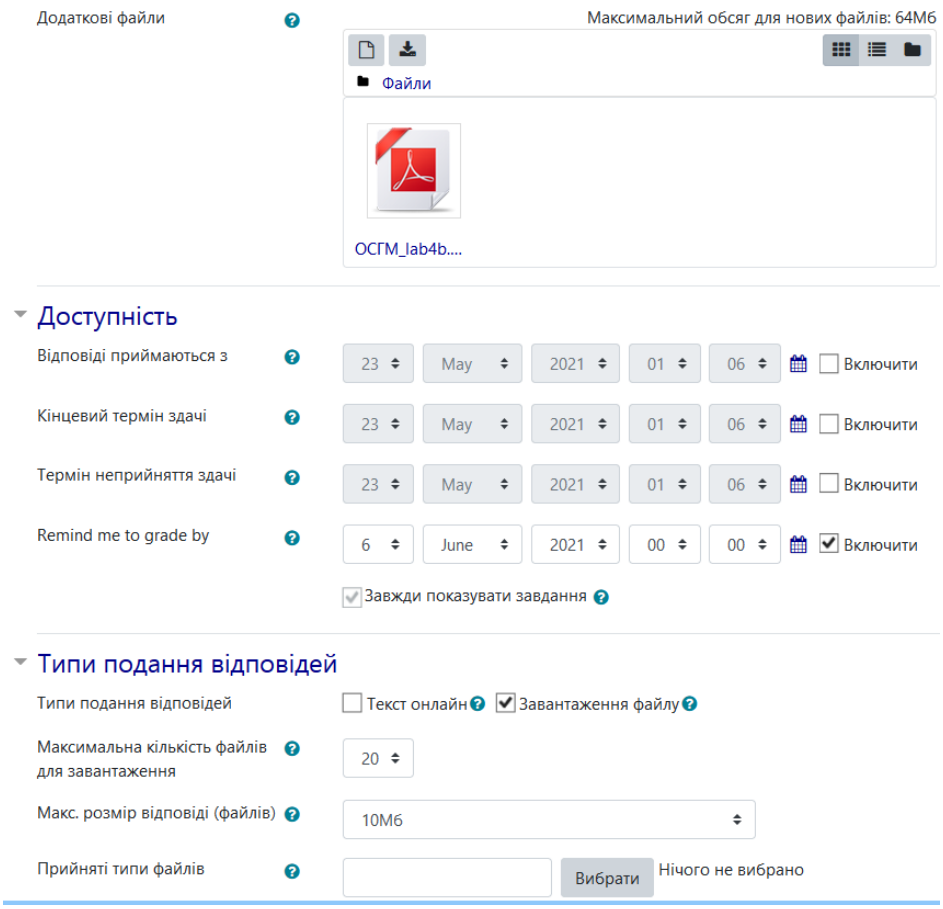

Рисунок 3.9 – Параметри проведення і оцінювання лабораторного заняття

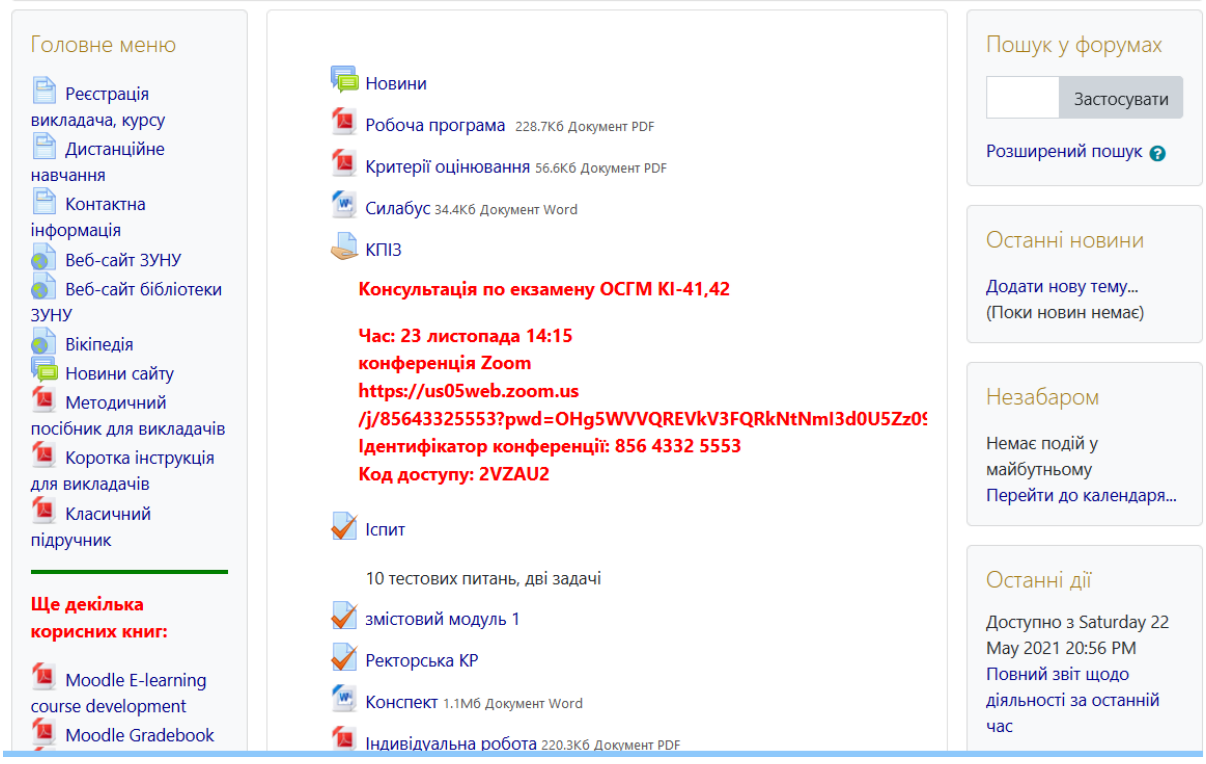

Рисунок 3.10 – Заголовок сторінки електронного підручника

На рисунку представлено знімок екрану з інтерфейсу Moodle або подібної системи управління навчальним контентом. Він показує список розділів курсу або модулів, кожен з яких містить навчальні матеріали. Існує п'ять розділів, кожен з яких має вкладені файлові ресурси, більшість з яких у форматі PDF. Ось деталі кожного розділу:

1. "Сервери Інтернет" - цей розділ містить три PDF-файли, один з яких має заголовок "Стек TCP/IP", другий - "1 Сервери інтернет", і третій - "Лабораторна робота 1 Робота з мережею в командному рядку", а також додатковий файл, ймовірно з завданням для лабораторної роботи №1.
2. "Структура Інтернет" - у цьому розділі є один PDF-файл з назвою "2 Структура Інтернет".
3. "Протоколи прикладного рівня" - тут розташовані два PDF-файли: один з них - "3. Протоколи прикладного рівня(Допомогти з Файл", інший - "Лабораторна робота 2. Встановлення і налаштування FTP серверів".
4. "Сервери мережевих файлових систем та проксі-сервери" - містить один PDF-файл з назвою "4. Сервери мережевих файлових систем".
5. "MySQL як компонент динамічного контенту для сервісів" - включає PDF-файл з назвою "5. MySQL як компонент динамічного контенту для сервісів"

Сформовано веб-сторінку електронного навчального ресурсу у рамках Moodle. Розроблені тези лекцій, презентації, а також підготовлені методичні вказівки для проведення лабораторних занять.

Процес створення секцій підручника, що фокусуються на оцінюванні засвоєння матеріалу студентами.

Виготовлення тестів для цифрового посібника передбачає спочатку створення окремих категорій в "Банку питань", куди подальше додаються відповідні питання. Потім в контексті навчального заняття організовується сам тест. Щоб це здійснити, переходимо до "Банку питань" через розділ "Керування". Там, у меню, обираємо "Категорії".

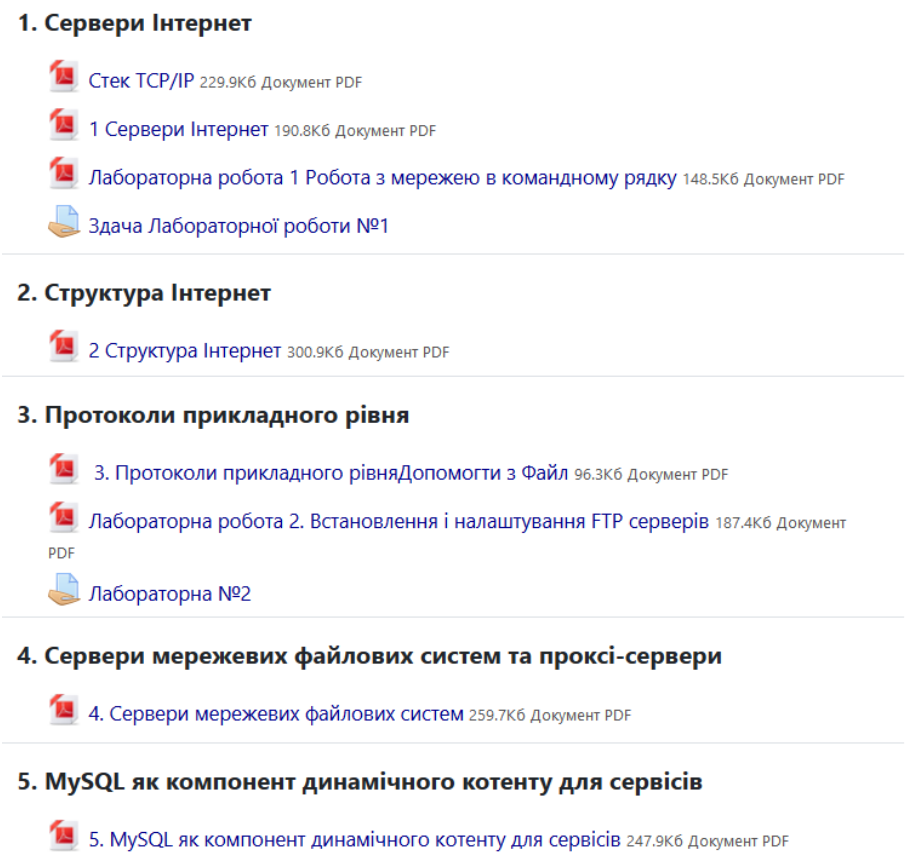

Рисунок 3.11 – Головна сторінка електронного навчального ресурсу

У вкладці "Назва" (рисунок 3.12) задаємо назву новій категорії, такій як "Тест до теми 1", і після цього застосовуємо опцію "Додати категорію". За цією методикою ми створюємо потрібну кількість категорій.

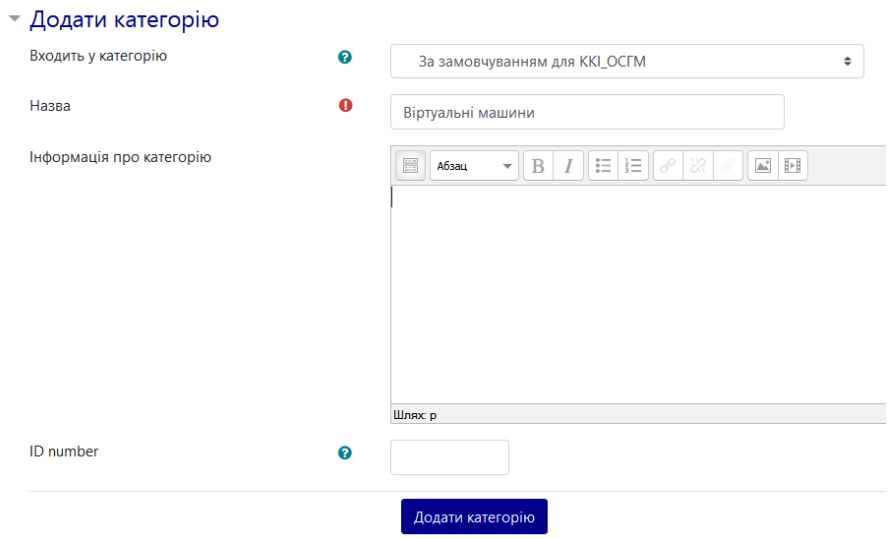

Рисунок 3.12 – Створення категорії для тестів

Для більш еластичного створення тестів за модулями чи фінального екзамену, розробляємо окремі категорії для тестових питань, які відносяться до лабораторних занять та екзаменаційних завдань (рисунок 3.13).

Рисунок 3.13 – Додані категорії тестових питань

Потім приступаємо до додавання питань у вибрані категорії "Банку питань". Для кожної категорії готуємо окремий текстовий файл у форматі TXT (використовуючи приміром Блокнот), де попередньо зберігаємо всі питання. При збереженні файлу вказуємо кодування "UTF-8". У цьому файлі питання не нумеруються, відповіді маркуємо великими літерами латинського алфавіту. Правильну відповідь визначаємо відповідним чином.

В "Керуванні" обираємо опцію "Імпорт", де в розділі "Формат файлу" на сторінці "Імпорт" вказуємо формат "Aiken" рисунок 3.15). Далі в "Основному" вибираємо категорію для імпорту питань.

Завершивши імпорт файлу з тестовими питаннями, переходимо до "Банку питань", де вже видно нові записи. Аналогічні дії виконуємо для інших категорій (рисунок 3.16).

```
Команда для запуску запуску діалогового режиму роботи з БД mySQL Windows?
A) mysql --user=root
B) mysqld -standalone
C) mysql-max-nt --remove
D) mysql-max-nt -install
ANSWER: A
СУБД це?
A) безліч інформаційних об'єктів (та їх властивостей) а також зв'язків між ними
B) набір взаємозалежних даних, що відображають інформацію про певну предметну область
C) комп'ютерна програма, що дає змогу описувати дані у вигляді об'єктів і зв'язків, маніпулювати ними і має зручний інтерфейс
D) об’єкт предметної області, що є множиною елементів
ANSWER: C
Сутність це?
A) об’єкт предметної області, що є множиною елементів
B) безліч інформаційних об'єктів (та їх властивостей) а також зв'язків між ними
C) набір взаємозалежних даних, що відображають інформацію про певну предметну область
D) комп'ютерна програма, що дає змогу описувати дані у вигляді об'єктів і зв'язків, маніпулювати ними і має зручний інтерфейс
ANSWER: A
команда для Видалення MySQL сервісу Windows?
A) mysql-max-nt --remove
B) mysql-max-nt -install
C) mysqld -standalone
D) mysql --user=root
ANSWER: A
команда для встановлення MySQL в якості системного процесу Windows?
```

Normal text file | length : 14 348 lines : 371 | Ln : 45 Col : 27 Pos : 3 065 | Unix (LF) | UTF-8 | INS

Рисунок 3.14 –Тестові запитання для і імпорту

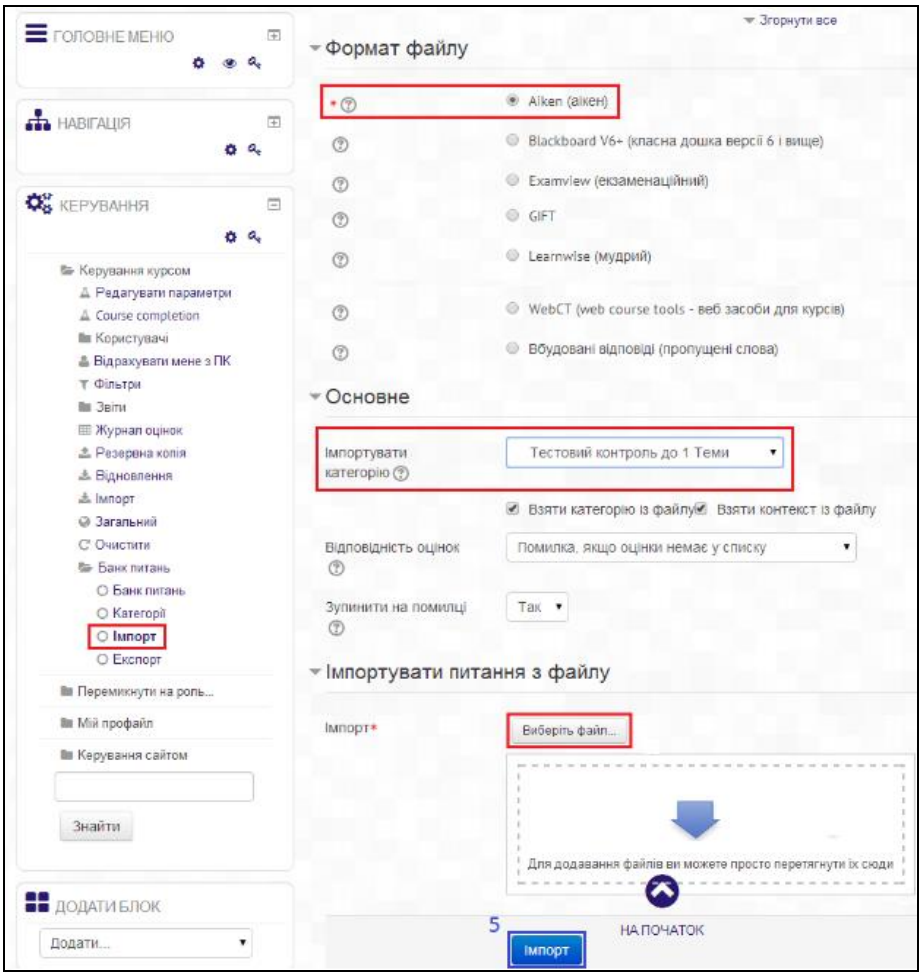

Рисунок 3.15 – Сторінка «Імпорт»

Після завантаження всіх питань в Moodle створюємо тест і додаємо до нього питання з різних категорій. Це здійснюємо, повернувшись до сторінки редагування курсу, де вибираємо "Додати діяльність або ресурс" та з наявних варіантів обираємо "Тест", щоб завершити процес його створення.

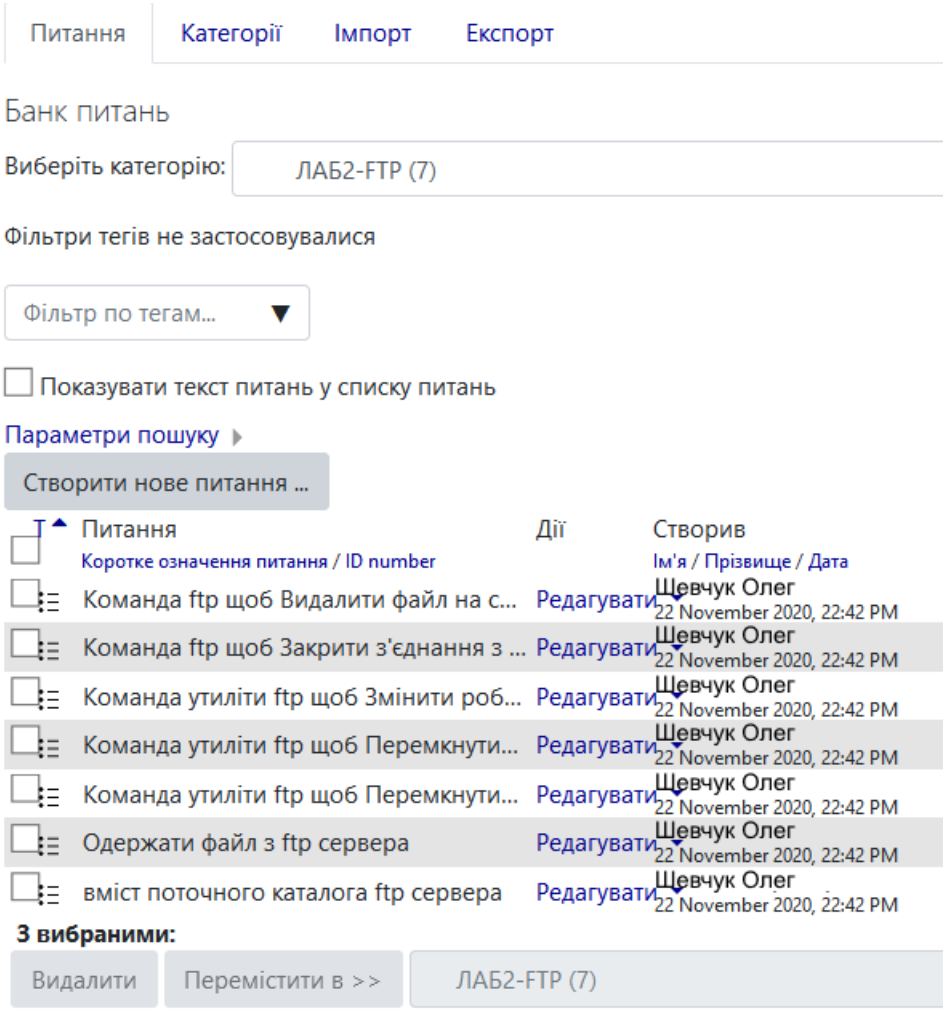

Рисунок 3.16 – Область «Банк питань»

На екрані для формування тестів (рисунок 3.12), у введеному розділі "Назва", реєструємо відповідну назву для тесту, таку як "Перевірка знань модуля". Завершивши цей крок, ми зберігаємо внесені дані та переходимо назад до основного інтерфейсу курсу.

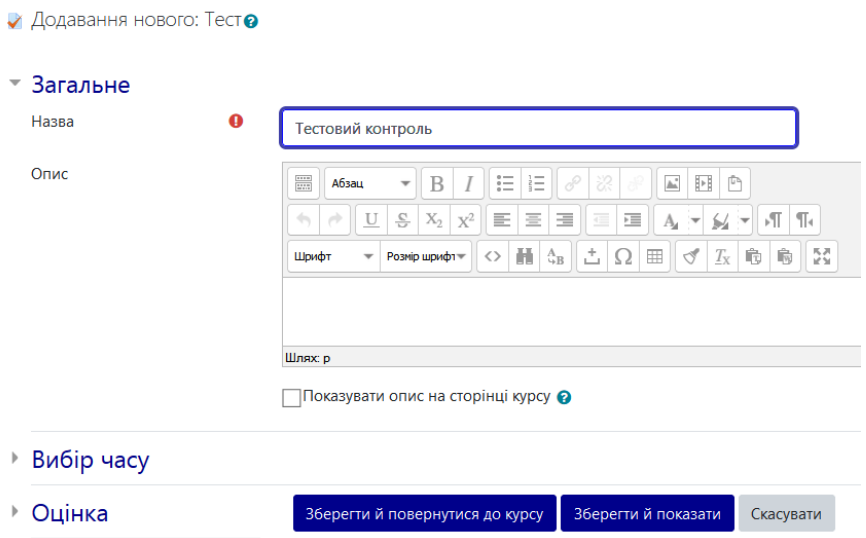

Рисунок 3.17 – Створення тестів

Перед додаванням питань до тестів редагуємо тест (рисунок 3.18).

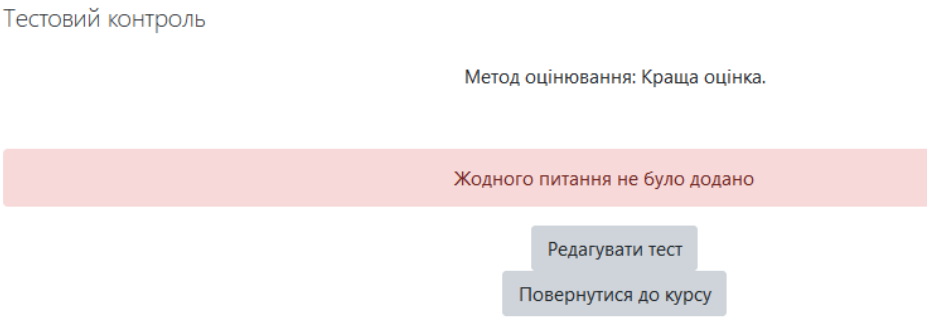

Рисунок 3.18 – Тестовий контроль знань

У інтерфейсі налаштування тестових завдань (позначено на рисунках 3.14-3.15), ми визначаємо число питань для включення у тест з використанням функції вибору випадкових запитань і застосовуємо опцію «Внести до тесту».

Також слід призначити оцінки: максимальна сумарна оцінка за тест становить 100 балів, що при десяти запитаннях виходить по 10 балів за кожне.

Для налаштування тривалості тесту потрібно перейти на сторінку вказаного тесту і вибрати «Редагувати параметри» у розділі «Керування». Тут встановлюємо бажаний часовий ліміт тестування та зберігаємо зміни.

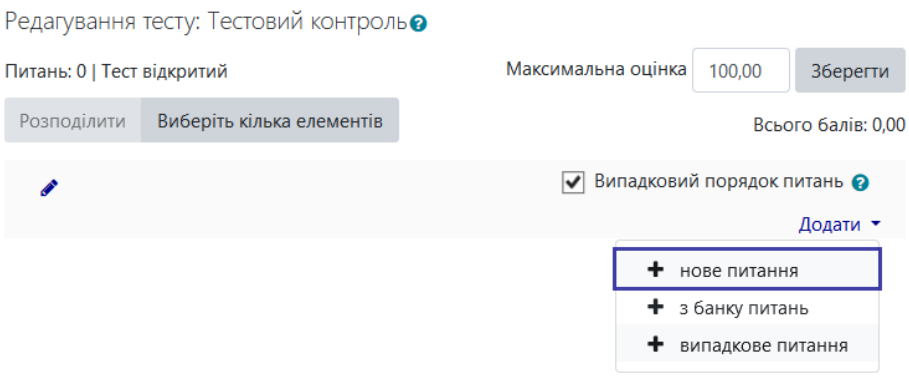

Рисунок 3.14 –Встановлення оцінки за тест

Після внесення заданої кількості запитань, що вибираються випадковим чином, користувача перенаправлять на екран компонування тесту (вказано на рисунку 3.16). На цій сторінці виконується розподіл запитань по різних сторінках, які студенти побачать послідовно під час проходження тесту.

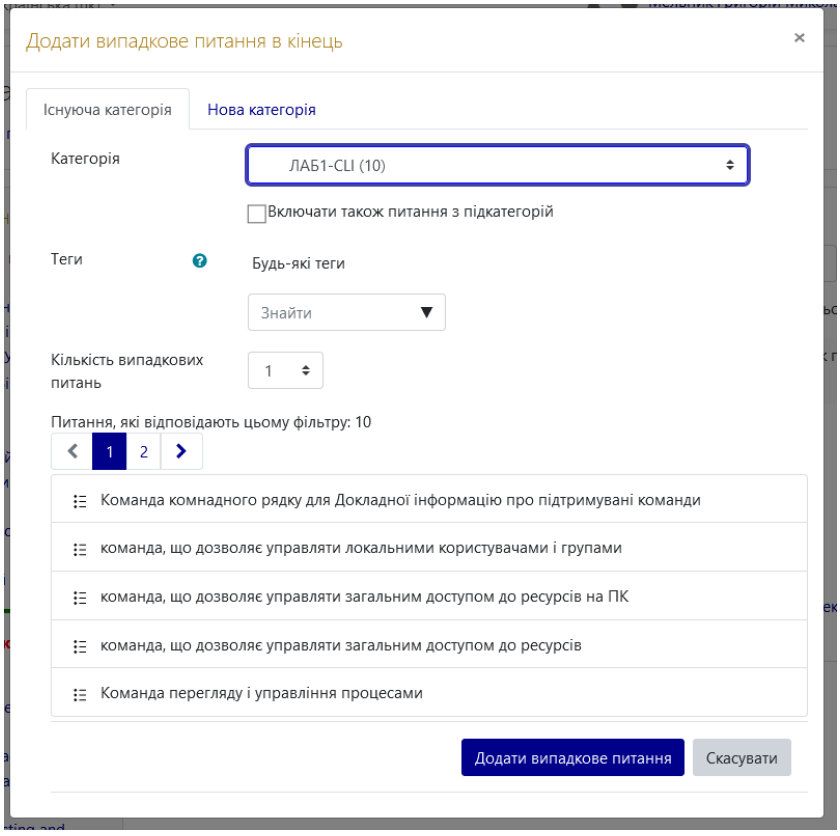

Рисунок 3.15 – Додавання питань із категорій

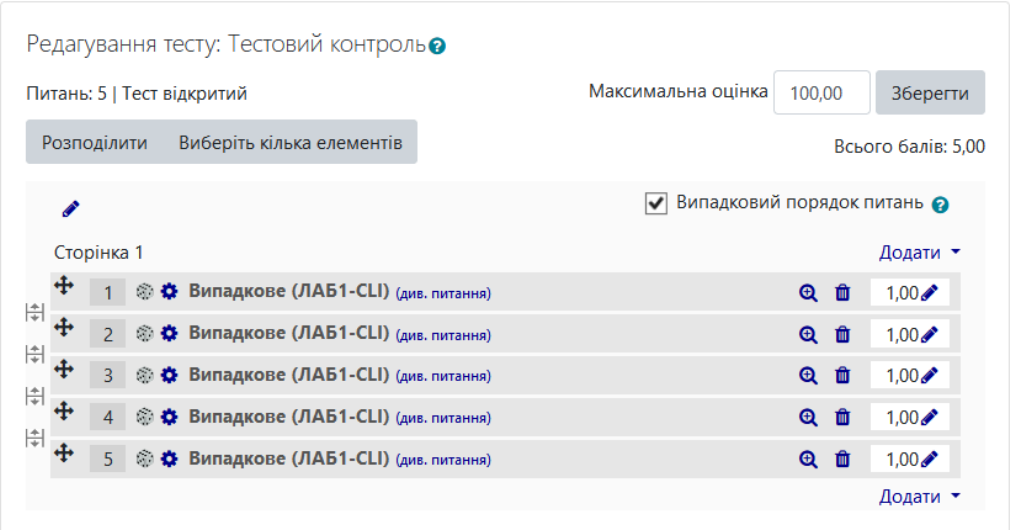

Рисунок 3.16 – Макет сформованого тесту

Успішно запроваджено електронний навчальний ресурс через інструменти Moodle, з усіма важливими елементами та тестами, які потрібні для курсу.

3.3 Алгоритми індивідуалізації навчання

Збір та аналіз користувацьких даних

На основі об'єктної моделі, процедура "Збір та аналіз користувацьких даних" може виглядати наступним чином:

```
class User:
    def __init__(self, userID, name):
        self.userID = userID
        self.name = name
        self.interactionHistory = []
        self.testResults = []
        self.learningPreferences = []

    def collectData(self):
        # дані збираються через інтерфейс користувача або з БД
        self.interactionHistory =
getInteractionHistoryFromDatabase(self.userID)
        self.testResults = getTestResultsFromDatabase(self.userID)
        self.learningPreferences =
getLearningPreferencesFromDatabase(self.userID)

    def getInteractionHistory(self):
```

```
        return self.interactionHistory

    def evaluateTestResults(self):
        # оцінка результатів
        averageScore = sum(self.testResults) / len(self.testResults)
        return averageScore
```

Функція для збору та аналізу даних

```
def collectAndAnalyzeUserData(user):
    user.collectData()
    interactionHistory = user.getInteractionHistory()
    testResultsAnalysis = user.evaluateTestResults()
    # Тут може бути додатковий аналіз даних
    return interactionHistory, testResultsAnalysis

# Приклад використання
user = User("12345", "Іван")
interactionHistory, testResultsAnalysis =
collectAndAnalyzeUserData(user)
```

Ця процедура демонструє основні кроки збору та первинного аналізу користувацьких даних. Вона може бути розширена та адаптована відповідно до специфічних вимог та структури даних.

прогнозування оптимального навчального шляху

На основі об'єктної моделі, функція predictOptimalLearningPath для прогнозування оптимального навчального шляху користувача може бути організована так:

```
class PredictionModel:
    def predictPath(self, user, dataAnalyzer):
        # Аналіз даних користувача
        userData = dataAnalyzer.analyzeData(user)

        # Визначення оптимального навчального шляху
        optimalPath = self.calculateOptimalPath(userData)

        return optimalPath

    def calculateOptimalPath(self, userData):
        # Логіка для визначення оптимального шляху
        # Повернення послідовності навчальних матеріалів
        return learningPath

# Приклад використання
```

```
user = User("12345", "Іван")
dataAnalyzer = DataAnalyzer()
predictionModel = PredictionModel()
optimalLearningPath = predictionModel.predictPath(user,
dataAnalyzer)
```

Функція PredictionModel використовує DataAnalyzer для аналізу даних користувача, а потім використовує ці дані для визначення оптимального навчального шляху. Логіка визначення цього шляху залежить від специфіки системи та потреб користувачів.

Адаптація навчальних матеріалів і завдань

На основі розробленої об'єктної моделі, функція для адаптації навчальних матеріалів і завдань може виглядати так:

```
class LearningMaterial:
    # ... (інші атрибути та методи класу)

    def adaptToUser(self, user, learningPath):
        # Оцінка поточного рівня та потреб користувача
        userLevel = learningPath.assessUserLevel(user)

        # Адаптація вмісту до рівня та переваг користувача
        adaptedContent = self.customizeContent(userLevel,
user.learningPreferences)

        return adaptedContent

    def customizeContent(self, userLevel, learningPreferences):
        # Логіка адаптації матеріалів
        # Може включати зміну складності, типу контенту (відео,
текст, інтерактивні завдання)
        adaptedContent = "Adapted Content based on Level: {},
Preferences: {}".format(userLevel, learningPreferences)
        return adaptedContent
```

Тестування методів класу

```
# Приклад використання
user = User("12345", "Іван")
learningMaterial = LearningMaterial()
learningPath = LearningPath()
adaptedMaterial = learningMaterial.adaptToUser(user, learningPath)
```

У цій процедурі клас LearningMaterial має метод adaptToUser, який використовує інформацію про користувача та його навчальний шлях для адаптації навчальних матеріалів з урахуванням його індивідуальних потреб та прогресу.

Функція assessUserLevel в класі LearningPath може виглядати наступним чином:

```
class LearningPath:
    # ... (інші атрибути та методи класу)

    def assessUserLevel(self, user):
        # Аналіз історії взаємодій та результатів тестувань користувача
        interactionHistoryAnalysis = self.analyzeInteractionHistory(user.interactionHistory)
        testResultsAnalysis = user.evaluateTestResults()

        # Визначення рівня користувача на основі аналізу
        userLevel = self.determineLevel(interactionHistoryAnalysis, testResultsAnalysis)
        return userLevel

    def analyzeInteractionHistory(self, interactionHistory):
        # Логіка аналізу історії взаємодій
        # Наприклад, визначення переваг у навчанні або частих помилок
        # Повертає результати аналізу
        return analysisResult

    def determineLevel(self, interactionHistoryAnalysis, testResultsAnalysis):
        # Логіка визначення рівня на основі аналізу
        # Може включати вирішення про рівень складності матеріалів
        userLevel = "Beginner" # Приклад, реальна логіка буде складнішою
        return userLevel

# Приклад використання
user = User("12345", "Іван")
learningPath = LearningPath()
userLevel = learningPath.assessUserLevel(user)
```

Ця функція аналізує історію взаємодій користувача та його результати тестувань, щоб визначити його рівень знань та вмінь у контексті навчальної системи. Результат цього аналізу може бути використаний для подальшої адаптації навчальних матеріалів.

Ось приклади функцій analyzeInteractionHistory та determineLevel для класу LearningPath:

```
class LearningPath:
    # ... (інші атрибути та методи класу)

    def analyzeInteractionHistory(self, interactionHistory):
        # Припустимо, interactionHistory - це список об'єктів з деталями про взаємодії користувача
        # Аналіз для визначення, наприклад, найчастіше вивчених тем, ступеня залучення тощо
        frequentTopics = self.findFrequentTopics(interactionHistory)
        engagementLevel = self.calculateEngagementLevel(interactionHistory)
        return {'frequentTopics': frequentTopics, 'engagementLevel': engagementLevel}
```

Функція знаходження тем до яких звертались найчастіше

```
    def findFrequentTopics(self, interactionHistory):
        # Логіка виявлення найчастіших тем
        topicsCounter = Counter([interaction.topic for interaction in interactionHistory])
        mostFrequentTopics = topicsCounter.most_common(3)
        return mostFrequentTopics

    def calculateEngagementLevel(self, interactionHistory):
        # Логіка визначення рівня залучення
        totalInteractions = len(interactionHistory)
        if totalInteractions > 50:
            return "High"
        elif totalInteractions > 20:
            return "Medium"
        else:
            return "Low"
```

Функція Визначення рівня на основі аналізу історії взаємодій та результатів тестів

```
    def determineLevel(self, interactionHistoryAnalysis, testResultsAnalysis):
        if testResultsAnalysis > 80 and interactionHistoryAnalysis['engagementLevel'] == "High":
            return "Advanced"
        elif testResultsAnalysis > 50:
            return "Intermediate"
        else:
            return "Beginner"
```

У цьому прикладі, analyzeInteractionHistory аналізує історію взаємодій користувача для виявлення найпопулярніших тем та загального рівня залучення. determineLevel використовує результати цього аналізу разом із результатами тестів для визначення загального рівня користувача у навчальній системі.

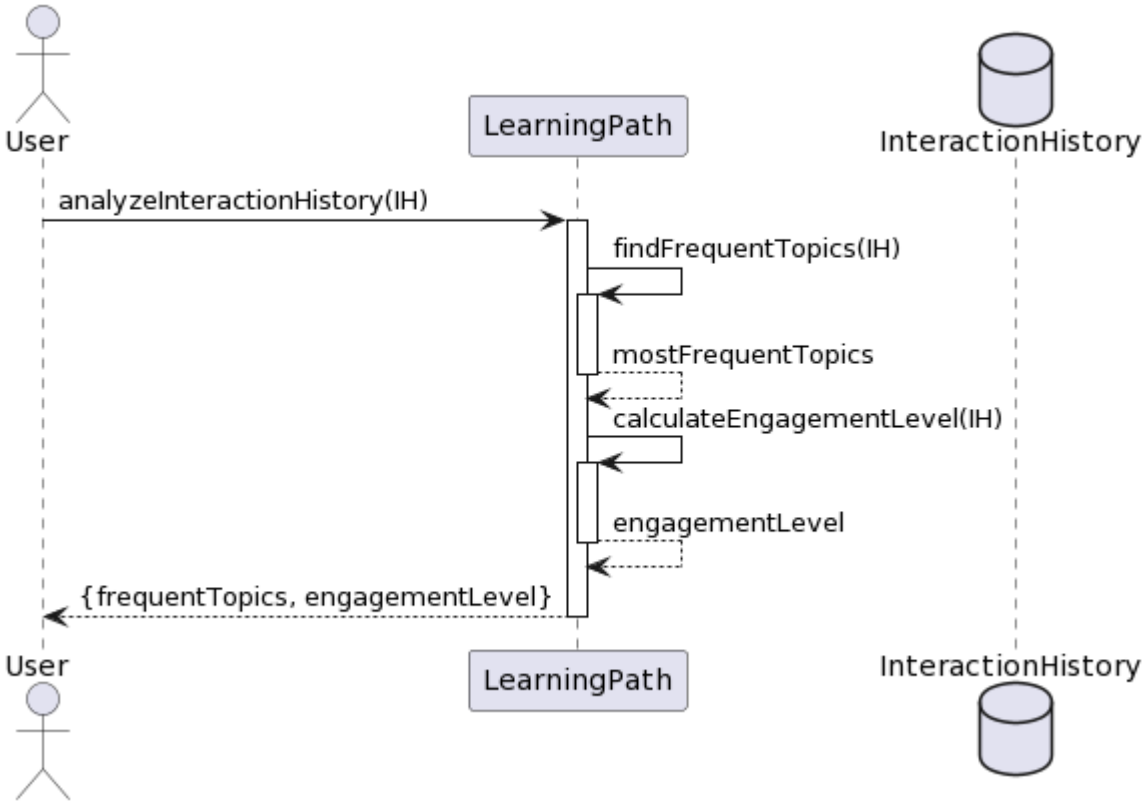

Рисунок - Діаграма послідовності функції analyzeInteractionHistory

3.4 Методи тестування розроблених алгоритмів

Згенеруємо тестову базу даних яка місить історію взаємодії користувача з предметом.

1) Період навчання 1 версеня 2023 р. по 1 грудня 2023 р.

2) Предмет "Комп'ютерні мережі" читається 15 тижнів, кожного тижня проводиться 1 лекційне заняття та 1 лабораторне заняття.

3) Теми лекцій предмета : "Тема 1. Вступ до мережних технологій. Модель OSI Тема 2. Середовища передавання сигналів Тема 3. Базові мережні технології. Тема 4. Безпровідні мережі Тема 5. Локальні мережі Ethernet. Тема 6. Пристрої та

обладнання локальних мереж Тема 7. Протоколи та засоби керування обладнанням Тема 8. Комутатори Тема 9. Обладнання для волоконнооптичних каналів Тема 10. Стек протоколів TCP/IP. Протокол IP Тема 11. Протоколи TCP/UDP Тема 12. Маршрутизація у комп'ютерних мережах Тема 13. Протоколи EIGRP, OSPF, BGP Тема 14. Сучасні маршрутизатори та їх основні характеристики" 4) Теми лабораторних занять: 1 Тема: Вивчення мережних засобів операційних систем 2 Тема: Створення мережі між двома комп’ютерами 3 Тема: Вивчення устаткування локальних мереж 4 Тема: Розподіл адресного простору IP засобами маскування 5 Тема: Побудова віртуальних локальних мереж.

Для створення тестової бази даних історії взаємодії користувача з предметом "Комп'ютерні мережі", можна використати наступний приклад у форматі Python:

```
from datetime import datetime, timedelta

# Створення тестових даних
class Interaction:
    def __init__(self, date, topic, type):
        self.date = date
        self.topic = topic
        self.type = type  # 'lecture' або 'lab'

def generate_interaction_history(start_date, weeks, lecture_topics, lab_topics):
    interactions = []
    for week in range(weeks):
        # Додавання лекційного заняття
        lecture_date = start_date + timedelta(weeks=week)
        lecture_topic = f"Тема {week + 1}. {lecture_topics[week]}"
        interactions.append(Interaction(lecture_date, lecture_topic, 'lecture'))

        # Додавання лабораторного заняття
        lab_date = lecture_date + timedelta(days=3)
        lab_topic = f"Тема {week + 1}: {lab_topics[week]}"
        interactions.append(Interaction(lab_date, lab_topic, 'lab'))

    return interactions
```

Генерація тестової БД та тестування функції

```
# Період навчання та теми
start_date = datetime(2023, 9, 1)
weeks = 15
lecture_topics = ["Вступ до мережних технологій. Модель OSI",
"Середовища передавання сигналів", ...]  # Доповнити теми
lab_topics = ["Вивчення мережних засобів операційних систем",
"Створення мережі між двома комп’ютерами", ...]  # Доповнити теми

# Генерація історії взаємодій
interaction_history = generate_interaction_history(start_date,
weeks, lecture_topics, lab_topics)
```

У цьому прикладі, generate_interaction_history генерує історію взаємодій для кожного тижня навчання, створюючи записи для лекційних та лабораторних занять із вказаними темами. Перелік тем для лекцій та лабораторних робіт потрібно доповнити відповідно до навчального плану.

3.5 Висновки до розділу

Програмно реалізовано алгоритми збору та аналіз користувацьких даних, прогнозування оптимального навчального шляху, адаптації навчальних матеріалів і завдань. Розроблено базу даних для ресурсів електронного підручника. Для тестування алгоритмів створено тестове середовище в системі дистанційного навчання Moodle.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи отримано наступні результати:

1. Проведено аналітичний огляд алгоритмів проектування компонентів електронних навчальних систем.

2.Систематизовано потреби користувачів щодо зручності та функціональності компонентів електронних навчальних систем.

3. Розроблено структуру інтерактивного навчального посібника, який інтегрує мультимедійні елементи для підвищення навчального досвіду. Посібник розроблений згідно з модульним принципом, де кожен модуль самостійною частиною, що може бути використаний для тестування та самооцінки знань. Розроблено алгоритм адаптації електронного посібника за рівнем навчання.

4. Розроблено алгоритми проєктування компонентів електронних навчальних систем на основі планування індивідуального навчального шляху, що дозволило враховувати оцінювання для пропозиції інших навчальних компонент.

5. Програмно реалізовано алгоритми збору та аналіз користувацьких даних, прогнозування оптимального навчального шляху, адаптації навчальних матеріалів і завдань. Розроблено базу даних для ресурсів електронного підручника. Для тестування алгоритмів створено тестове середовище в системі дистанційного навчання Moodle.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шаповалов Я. Ю. Моделі індивідуального навчального шляху в системах дистанційного навчання. Матеріали науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі» (ІКСМ-2023)., м. Тернопіль, 5 груд. 2023 р. С. 74.

2. Шаповалов Я. Ю. Алгоритми проєктування компонентів систем дистанційного навчання. Матеріали науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі» (ІКСМ-2023)., м. Тернопіль, 5 груд. 2023 р. С. 75.

3. Березький О.М., Дубчак Л.О., Мельник Г.М. Методичні рекомендації до виконання кваліфікаційної роботи освітнього ступеня «Магістр». Спеціальність: комп'ютерна інженерія. Магістерська програма – «Комп'ютерна інженерія» / Під ред. О.М. Березького. Тернопіль: ЗУНУ, 2020. 32 с.

4. Гураль І.В., Дубчак Л.О. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп'ютерна інженерія» / Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.

5. Hierarchical Cluster Analysis Heatmaps and Pattern Analysis: An Approach for Visualizing Learning Management System Interaction Data / J. E. Lee et al. Proceedings of the 9th International Conference on Educational Data Mining. 2016. P. 603-604. URL: https://www.educationaldatamining.org/EDM2016/proceedings/paper_34.pdf.

6. Zhao N., Cao Y., Lau R. Modeling fonts in context: Font prediction on web designs. *Computer Graphics Forum*. 2018. P. 385--395.

7. Synthtiger: Synthetic text image generator towards better text recognition models / M. Yim et al. *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*. 2021. P. 109--124.

8. Deepfont: Identify your font from an image / Z. Wang et al. *Proceedings of the International Conference on Multimedia*. 2015. P. 451--459.

9. Tibshirani R., Walther G., Hastie T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*. 2001. Vol. 63, no. 2. P. 411--423.

10. Takeshita K., Shioyama J., Uchida S. Label or message: A large-scale experimental survey of texts and objects co-occurrence. *Proceedings of the International Conference on Pattern Recognition (ICPR)*. 2021. P. 6227--6234.

11. Gupta A., Vedaldi A., Zisserman A. Synthetic Data for Text Localisation in Natural Images. CoRR. 2016. Abs/1604.06646. URL: http://arxiv.org/abs/1604.06646.

12. TextBoxes: A Fast Text Detector with a Single Deep Neural Network / M. Liao et al. CoRR. 2016. Abs/1611.06779. URL: http://arxiv.org/abs/1611.06779.

13. EAST: An Efficient and Accurate Scene Text Detector / X. Zhou et al. URL: 10.48550/ARXIV.1704.03155.

14. Text/Non-Text Separation from Handwritten Document Images Using LBP Based Features: An Empirical Study / S. Ghosh et al. Journal of Imaging. 2018. Vol. 4, no. 4. URL: 10.3390/jimaging4040057.

15. Dropout: a simple way to prevent neural networks from overfitting / N. Srivastava et al. *The Journal of Machine Learning Research*. 2014. Vol. 15, no. 1. P. 1929--1958.

16. Serif or sans: Visual font analytics on book covers and online advertisements / Y. Shinahara et al. *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*. 2019. P. 1041--1046.

17. Exploratory font selection using crowdsourced attributes / P. O'Donovan et al. *ACM Transactions on Graphics*. 2014. Vol. 33, no. 4. P. 1--9.

18. Distributed representations of words and phrases and their compositionality / T. Mikolov et al. *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 2013.

19. Efficient estimation of word representations in vector space / T. Mikolov et al. *International Conference on Learning Representations*. 2013.

20. Matsumura S., Choi S., Aizawa K. Font search across various languages based on multimodal learning. *Proceedings of the Conference on Multimedia Information Processing and Retrieval (MIPR)*. 2020. P. 173--176.

21. Kulahcioglu T., Melo G. D. Fonts like this but happier: A new way to discover fonts. *Proceedings of the International Conference on Multimedia*. 2020. P. 2973--2981.

22. Kulahcioglu T., Melo G. D. Paralinguistic recommendations for affective word clouds. *Proceedings of the International Conference on Intelligent User Interfaces*. 2019. P. 132--143.

23. Kulahcioglu T., Melo G. D. Predicting semantic signatures of fonts. *Proceedings of the International Conference on Semantic Computing (ICSC)*. 2018. P. 115--122.

24. Openimages: A public dataset for large-scale multi-label and multi-class image classification / I. Krasin et al. 2017.

25. Kingma D., Ba J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 2014.

26. Visual font pairing / S. Jiang et al. *IEEE Transactions on Multimedia*. 2019. Vol. 22, no. 8. P. 2086--2097.

27. Ikoma M., Iwana B., Uchida S. Effect of text color on word embeddings. *Proceedings of the International Workshop on Document Analysis Systems (DAS)*. 2020. P. 341--355.

28. Deep residual learning for image recognition / K. He et al. *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016. P. 770--778.

29. BERT: pre-training of deep bidirectional transformers for language understanding / J. Devlin et al. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2019. P. 4171--4186.

30. Choi S., Matsumura S., Aizawa K. Assist users' interactions in font search with unexpected but useful concepts generated by multimodal learning. *Proceedings of the International Conference on Multimedia Retrieval*. 2019. P. 235--243.

31. Large-scale visual font recognition / G. Chen et al. *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014. P. 3598--3605.

32. Character region awareness for text detection / Y. Baek et al. *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019. P. 9365--9374.

33. Analyzing Font Style Usage and Contextual Factors in Real Images / N. Yasukochi et al. *Document Analysis and Recognition - ICDAR 2023 - 17th International Conference, San Jose, CA, USA, August 21-26, 2023, Proceedings, Part III*. 2023. P. 331--347. URL: 10.1007/978-3-031-41682-8_21.

34. Sundermeyer M., Schlüter R., Ney H. LSTM neural networks for language modeling. *Proc. Interspeech 2012*. 2012. P. 194--197. URL: 10.21437/Interspeech.2012-65.

35. Papers with Code - SVT Dataset. The latest in Machine Learning | Papers With Code. URL: https://paperswithcode.com/dataset/svt.

36. IAM Handwriting Database. Research Group on Computer Vision and Artificial Intelligence – Computer Vision and Artificial Intelligence. URL: https://fki.tic.heia-fr.ch/databases/iam-handwriting-database.

37. Europarl Parallel Corpus. Statistical and Neural Machine Translation. URL: https://www.statmt.org/europarl/.

38. The Analysis of Performance for Priority Distinction Double-queue and Double-server Communication Network / J. L. Xiong et al. *International Journal of Communication*. 2014. Vol. 3. URL: https://api.semanticscholar.org/CorpusID:62382654.

39. Du J.-q. The strategy research and method realization for the computer network flow control. *International Conference on Graphic and Image Processing*. 2013. URL: https://api.semanticscholar.org/CorpusID:62720079.

40. Gao F., Liu Q., Zhan H. Research of SIP DoS Defense Mechanism Based on Queue Theory. *International Conference on Computer Science, Environment, Ecoinformatics, and Education*. 2011. URL: https://api.semanticscholar.org/CorpusID:14887144.

41. Minkevius S., Sakalauskas L. On the law of iterated logarithm for extreme queue length in an open queueing network. *International Journal of Computer Mathematics: Computer Systems Theory*. 2021. Vol. 6. P. 220 - 235. URL: https://api.semanticscholar.org/CorpusID:237548367.

42. Stuckey N. Stochastic Estimation and Control of Queues within a Computer Network. 2012. URL: https://api.semanticscholar.org/CorpusID:59897689.

43. RouteNet-Erlang: A Graph Neural Network for Network Performance Evaluation / M. F. Galmes et al. *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*. 2022. P. 2018-2027. URL: https://api.semanticscholar.org/CorpusID:247159041.

44. The Analysis of Performance for Priority Distinction Double-queue and Double-server Communication Network / J. L. Xiong et al. *International Journal of Communication*. 2014. Vol. 3. URL: https://api.semanticscholar.org/CorpusID:62382654.

45. Andreji M. Analysis of the influence of queue type on packet delay in computer networks. 2015. URL: https://api.semanticscholar.org/CorpusID:195971842.

46. Queue Control Model in a Clustered Computer Network using M/M/m Approach / A. Ejem et al. *International Journal of Computer Trends and Technology*. 2016. Vol. 35. P. 12-20. URL: https://api.semanticscholar.org/CorpusID:38081651.

47. Розрахунок ефективності використання обчислювальних ресурсів самовідновлювальної комп'ютерної системи / N. Kuchuk та ін. *Системи управління, навігації та зв'язку. Збірник наукових праць*. 2021. Т. 3, № 65. С. 92–95. URL: 10.26906/sunz.2021.3.092.

48. Poslaiko N. I. Дослідження стаціонарного режиму в системі масового обслуговування типу gi /m /1 зі слабкою післядією. *Problems of applied mathematics and mathematic modeling*. 2022. URL: 10.15421/322119.

49. Порівняльна ефективність класифікаторів зображень під час розпізнавання зон інтересу при лапароскопічних втручаннях / М. Р. Баязітов та ін. Medical Informatics and Engineering. 2020. № 2. С. 62–69. URL: 10.11603/mie.1996-1960.2020.2.11175.

50. Франчук Н. П. Стан та перспективи технологій машинного перекладу тексту. Theory and methods of e-learning. 2014. Т. 3. С. 319–325. URL: 10.55056/e-learn.v3i1.356.