НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО **Декан факультету**	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ** **Завідувач кафедри інформаційних систем і технологій**
_____ **Ігор БОЛБОТ** (підпис)	_____ **Михайло ШВИДЕНКО** (підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА (ПРОЄКТ)

на тему: *«Розробка веб-орієнтованої інформаційної системи для управління студентськими проектами»*

Спеціальність «**Інформаційні системи і технології»**

Освітньо-професійна програма «**Інформаційні системи і технології»**

Гарант освітньої програми

к.е.н., доцент
науковий ступінь та вчене звання _____ (підпис) **Максим Мокрієв**

Керівник бакалаврської кваліфікаційної роботи (проєкту)
д.ф. з к.н. _____ (підпис) **Ярослав Понзель**

Виконав _____ (підпис) **Олександр Євтодій**

Київ – 2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних систем і технологій

к.е.н., доцент _____ Михайло ШВИДЕНКО

«___» _____ 2026 р.

ЗАВДАННЯ ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Євтодію Олександру Павловичу

Спеціальність 126 – «Інформаційні системи та технології»

ОП «Інформаційно управляючі системи та технології»

Тема бакалаврської кваліфікаційної роботи «Розробка веборієнтованої інформаційної системи для управління студентськими проектами» затверджена наказом від «19» грудень 2025 р. № 3205 «С2».

Термін подання завершеної роботи на кафедру «25» травня 2026 р.

Вихідні дані до бакалаврської кваліфікаційної роботи (проєкту): система забезпечує централізований облік заяв, контроль відповідності встановленим вимогам, фіксацію управлінських рішень, координацію взаємодії між підрозділами та формування супровідної документації.

Перелік питань, що підлягають дослідженню:

- Аналіз предметної області контролю завдання та успішності студентів;
- Визначення вимог до інформаційної системи та моделювання її процесів
- Проєктування структури бази даних, архітектури та програмних модулів
- Проєктування архітектури програмного забезпечення з урахуванням вимог до масштабованості, надійності та модульності;
- Тестування, розгортання та оцінювання результатів роботи системи.

Дата видачі завдання «06» січня 2026 р.

Керівник бакалаврської кваліфікаційної роботи _____ **Ярослав ПОНЗЕЛЬ**

Завдання взяв до виконання _____ **Олександр ЄВТОДІЙ**

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

PI – програмний інтерфейс прикладного програмування

CSV – текстовий формат зберігання табличних даних

DB – база даних

GUI – графічний інтерфейс користувача

HTML – мова гіпертекстової розмітки

IdP – постачальник цифрової ідентифікації користувача

KPI – ключовий показник ефективності

ML – машинне навчання

PDF – формат електронного документа

PyQt6 – бібліотека Python для створення графічного інтерфейсу

QSS – мова стилізації інтерфейсу Qt

REST – архітектурний стиль взаємодії програмних компонентів

SSO – єдиний вхід користувача до інформаційних систем

SQL – мова структурованих запитів до баз даних

SQLite – вбудована реляційна система керування базами даних

UML – уніфікована мова моделювання

БД – база даних

ЗВО – заклад вищої освіти

ІС – інформаційна система

ПЗ – програмне забезпечення

СУБД – система управління базами даних

Ki – коефіцієнт знань студента

ВСТУП

В умовах цифрової трансформації освітньої сфери та впровадження компетентнісного підходу особливої актуальності набуває використання інформаційних систем, здатних забезпечити об’єктивний моніторинг навчальних досягнень студентів і підтримку процесу прийняття педагогічних рішень. Зростання обсягів навчальних даних, різноманітність форм контролю знань та індивідуальні освітні траєкторії зумовлюють потребу в автоматизованих засобах аналізу ключових показників успішності студентів. Традиційні підходи до оцінювання результатів навчання часто мають фрагментарний характер, не забезпечують цілісного бачення прогресу студента та не враховують динаміку його навчальної діяльності, що знижує ефективність освітнього процесу.

Використання сучасних веб-технологій, аналітичних модулів оброблення навчальних даних та інтелектуальних методів рекомендаційних систем створює передумови для побудови інформаційних систем нового покоління, орієнтованих на персоналізацію навчання та підвищення академічної успішності. Інтеграція механізмів збору, аналізу та візуалізації показників успішності дозволяє не лише здійснювати контроль навчальних результатів, а й формувати індивідуальні рекомендації щодо покращення засвоєння матеріалу, своєчасного коригування навчальної траєкторії та підвищення мотивації студентів.

Метою кваліфікаційної роботи є проєктування та розробка інформаційної системи для моніторингу ключових показників успішності студентів і формування персоналізованих навчальних рекомендацій на основі аналізу навчальних даних із використанням сучасних інформаційних технологій.

Для досягнення поставленої мети у роботі необхідно розв’язати такі завдання:

1. провести аналіз предметної області моніторингу навчальної успішності та існуючих програмних рішень;
2. сформувати функціональні, нефункціональні та безпекові вимоги до інформаційної системи;

3. розробити архітектуру програмного забезпечення з урахуванням принципів модульності, масштабованості та розширюваності;

4. побудувати UML-моделі основних процесів і компонентів системи;

5. реалізувати програмний прототип інформаційної системи для збору, оброблення та аналізу показників успішності студентів;

6. здійснити тестування та оцінку ефективності роботи системи в умовах освітнього процесу.

Об’єктом дослідження є процес моніторингу та аналізу навчальної успішності студентів у закладах вищої освіти.

Предметом дослідження є методи, моделі та програмні засоби побудови інформаційних систем для аналізу освітніх даних і формування персоналізованих навчальних рекомендацій.

Наукова новизна роботи полягає у розробленні концепції інформаційної системи, яка поєднує моніторинг ключових показників успішності студентів із механізмами аналітичної обробки навчальних даних та формування персоналізованих рекомендацій. Запропоновано підхід до інтеграції модулів збору освітніх даних, аналітики та рекомендаційної підсистеми в єдиній архітектурі, що дозволяє адаптивно реагувати на зміни навчальної активності студентів і підвищувати ефективність управління освітнім процесом.

Методи дослідження ґрунтуються на застосуванні системного аналізу предметної області, методів UML-моделювання для опису функціональних процесів і структури системи, а також методів об’єктно-орієнтованого проєктування при розробленні програмного забезпечення. Для реалізації взаємодії між компонентами системи використано клієнт–серверну архітектуру та REST-підхід, а для оцінювання результатів – методи експериментального тестування та аналізу показників ефективності.

Практична значущість одержаних результатів полягає у можливості використання розробленої інформаційної системи як інструменту підтримки навчального процесу у закладах вищої освіти. Система забезпечує автоматизований моніторинг успішності студентів, формування аналітичних

звітів для викладачів і адміністрації, а також надання персоналізованих рекомендацій студентам. Запропоноване рішення може бути інтегроване з існуючими освітніми інформаційними системами та використане для підвищення якості управління навчальною діяльністю.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

У межах предметної області інформаційна система моніторингу ключових показників успішності студентів розглядається як програмно-орієнтований комплекс, призначений для автоматизованого збору, зберігання, оброблення та аналізу освітніх даних. Система забезпечує формування агрегованих і персоналізованих показників успішності на основі результатів навчальної діяльності студентів, а також підтримує процес прийняття рішень шляхом генерації навчальних рекомендацій.

Функціональне призначення системи полягає у реалізації процесів моніторингу навчальних досягнень, аналізу динаміки успішності, виявлення відхилень від нормативних показників та формування рекомендацій щодо коригування індивідуальної навчальної траєкторії. Система інтегрується з зовнішніми освітніми інформаційними ресурсами закладу освіти, зокрема електронними журналами та ERP-системами, що забезпечує узгодженість і актуальність навчальних даних.

На рис. 1.1 подано DFD-діаграму 0-го рівня інформаційної системи моніторингу успішності студентів, яка відображає взаємодію зовнішніх сутностей (Студент, Викладач, Адміністратор, ERP/Електронний журнал) з центральним процесом «0 Інформаційна система моніторингу успішності студентів». Основними потоками даних є результати оцінювання, показники навчальної активності, аналітичні запити, сформовані рекомендації та дані синхронізації з зовнішніми системами.

Рис. 1.1 – DFD 0-рівня інформаційної системи моніторингу успішності студентів

На рис. 1.2 наведено BPMN-діаграму, що формалізує бізнес-процеси взаємодії користувачів із системою. Студент здійснює автентифікацію, перегляд показників успішності та отримання персоналізованих рекомендацій. Викладач виконує аналіз результатів навчання, формування оцінювань і перегляд аналітичних звітів. Адміністратор забезпечує керування користувачами, параметрами аналітики та інтеграційними механізмами. Аналітичне ядро системи реалізує процеси оброблення навчальних даних, розрахунку ключових показників успішності та генерації рекомендацій.

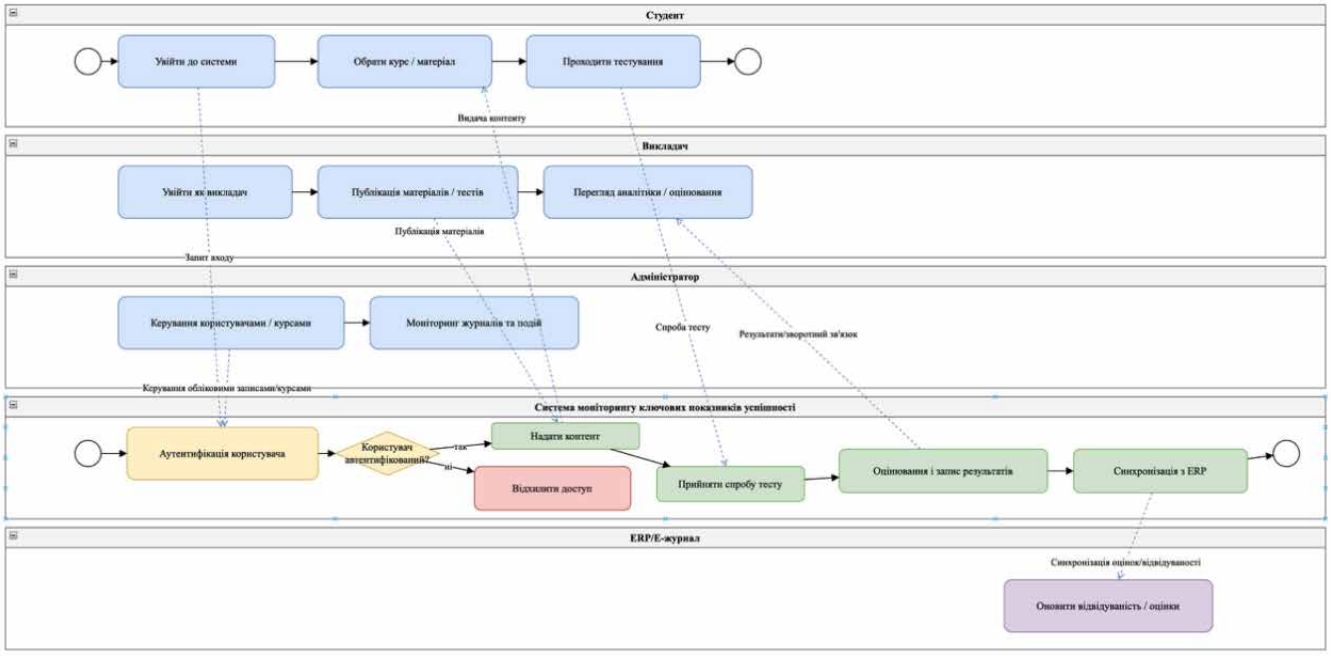


Рис. 1.2 – BPMN-діаграма процесів інформаційної системи моніторингу успішності студентів

Для формалізації предметної області у табл. 1.1 наведено основні сутності системи, їх функціональне призначення та типи оброблюваних даних.

Таблиця 1.1

Основні сутності предметної області інформаційної системи моніторингу успішності студентів

Сутність	Опис	Функціональне призначення	Типи даних
Студент	Кінцевий користувач системи, який навчається у закладі вищої освіти	Перегляд індивідуальних показників успішності, отримання персоналізованих навчальних рекомендацій, аналіз динаміки власного прогресу	Облікові дані, результати оцінювання, показники навчальної активності
Викладач	Користувач, відповідальний за контроль та аналіз результатів навчання	Аналіз успішності студентів і академічних груп, формування оцінок, перегляд аналітичних звітів і показників	Дані навчальних дисциплін, результати контролю, аналітичні показники
Адміністратор	Користувач із розширеними правами доступу до системи	Керування користувачами, налаштування параметрів системи та аналітичних модулів, контроль інтеграції із зовнішніми сервісами	Облікові дані, системні журнали, параметри конфігурації
Аналітичне ядро	Центральна програмна підсистема системи	Збір і оброблення навчальних даних, розрахунок ключових показників успішності, формування персоналізованих рекомендацій	Навчальні показники, статистичні дані, метадані
ERP / електронний журнал	Зовнішня інформаційна система закладу освіти	Синхронізація даних щодо оцінок, відвідуваності та академічної успішності	Записи оцінювання, журнали подій
Бази даних D1–D3	Сховища структурованої інформації системи	D1 – дані студентів і користувачів; D2 – показники успішності; D3 – навчальні рекомендації	Реляційні та NoSQL-структури

Предметна область інформаційної системи моніторингу успішності студентів характеризується централізованим накопиченням навчальних даних, автоматизованим аналізом ключових показників успішності та підтримкою механізмів персоналізації навчання. Застосування аналітичного підходу дозволяє підвищити об’єктивність оцінювання результатів навчання, забезпечити своєчасне виявлення проблемних аспектів освітнього процесу та створити основу для адаптивного управління навчальною діяльністю.

1.2 Теоретико-методологічні засади та стан наукових досліджень

У сучасних дослідженнях дистанційного навчання значна увага приділяється адаптивним моделям подання освітнього контенту, що враховують індивідуальні характеристики користувачів. У працях Barchenko N., Tolbatov V., Lavryk T. [1] представлено математичну модель адаптивної технології в e-learning-системах, де навчальний процес описується як ітераційна взаємодія між користувачем, контентом і коефіцієнтом знань K. На рис. 1.3 показано три етапи формування моделі навчання: початкову лінійну структуру подання матеріалу, замкнену схему з зворотним зв’язком через параметр KK та розширену адаптивну модель, у якій результати оцінювання впливають на побудову наступних завдань.

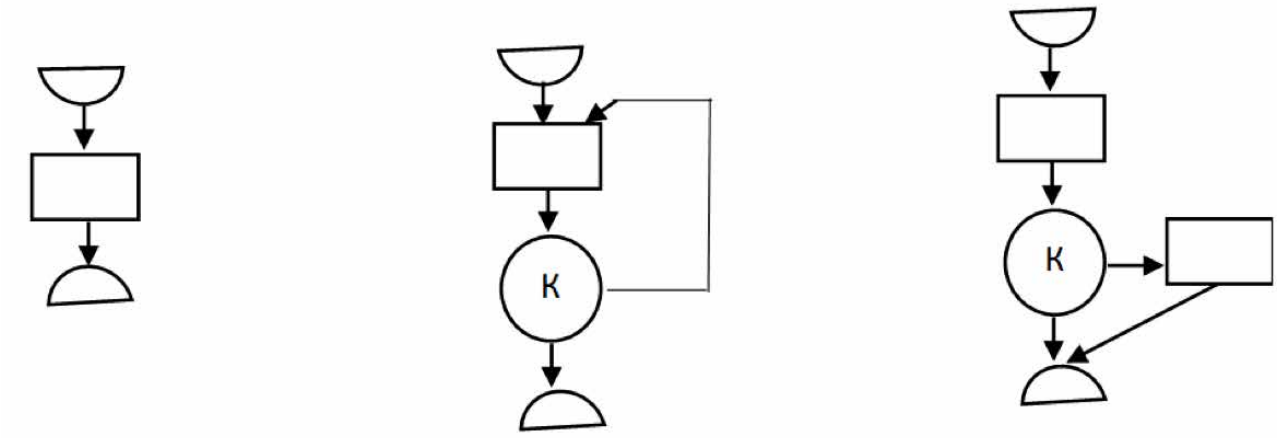


Рис. 1.3 - Етапи формування адаптивної моделі навчання із використанням коефіцієнта знань K (за Barchenko et al., 2022)

Подальший розвиток теоретичних моделей відображено у структурі логічного дерева знань (рис. 1.4), де вузли X,Y,Z,V та їх підмножини (x1,x2),(y1,y2) тощо репрезентують компоненти навчального контенту. Модель забезпечує ієрархічне представлення освітніх об’єктів і формує функцію доцільності D=f(X,Y,Z,V), що використовується для обчислення агрегованого показника відповідності навчального матеріалу індивідуальним особливостям студента [1].

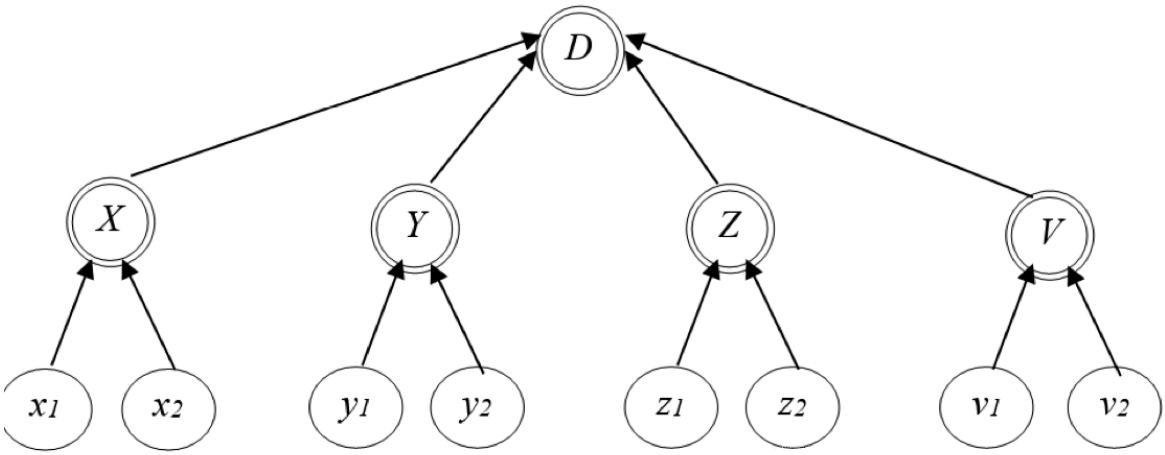


Рис. 1.4 - Ієрархічна структура зв’язків між компонентами навчального контенту (за Barchenko et al., 2022)

На основі ієрархічної моделі було побудовано аналітичний підхід до визначення ступеня експліцитності (відкритості) компонентів е-модулів, де кожен параметр x1,y1,z1,v1 характеризує деталізацію й глибину пояснення навчального матеріалу. На рис. 1.5 показано динаміку зміни ступеня експліцитності для чотирьох послідовних е-модулів, що демонструє перехід від поверхневого ознайомлення до повного розкриття теми та відповідного зростання когнітивного навантаження користувача. Цей підхід дозволяє формалізувати індивідуальний рівень деталізації залежно від результатів тестування.

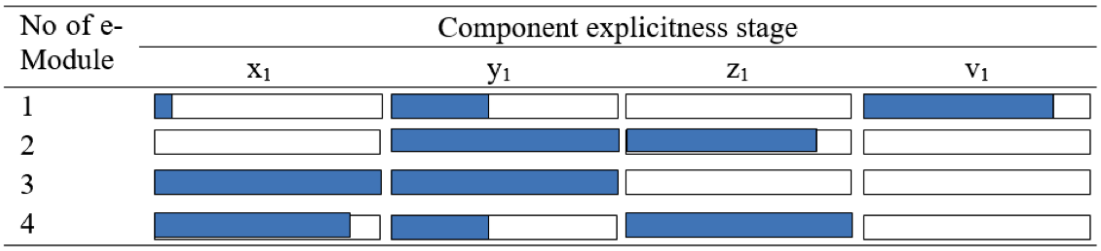


Рис. 1.5 - Графічне представлення ступеня експліцитності компонентів е-модулів (за Barchenko et al., 2022)

Індивідуалізація навчання реалізується через аналіз переваг студента, які оцінюються за рівнем інтересу до кожного компонента контенту (рис. 1.6). Графічна модель показує співвідношення між особистими перевагами та ступенем експліцитності, що дозволяє системі здійснювати адаптивний вибір матеріалів з урахуванням індивідуального профілю користувача. Подібний підхід до персоналізації навчання застосовується в моделях Sinitsyn E. та Larionova V. [2], де визначаються оптимальні маршрути засвоєння знань через аналіз поведінкових метрик.

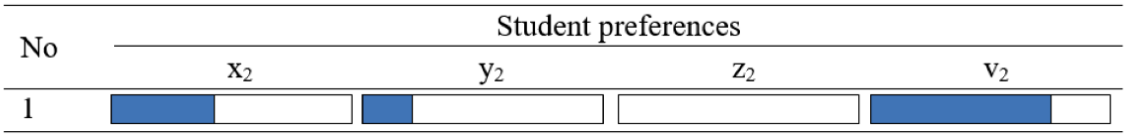


Рис. 1.6 - Модель оцінювання переваг студента відносно навчальних компонентів (за Barchenko et al., 2022)

На завершальному етапі модель Barchenko та співавт. [1] передбачає інтеграцію баз даних та програмних модулів у єдину архітектуру (рис. 1.7). У цій структурі підсистема Evaluation здійснює оцінювання параметрів е-модулів і студентів, модуль Selection вибирає оптимальний навчальний об’єкт на основі визначених переваг, а підсистема Optimization of Human-Machine Interaction виконує аналіз варіантів та формує рекомендації щодо найкращої навчальної траєкторії. Подібна архітектура дозволяє реалізувати адаптацію в реальному часі та оптимізувати інтерактивну взаємодію користувача з освітнім контентом.

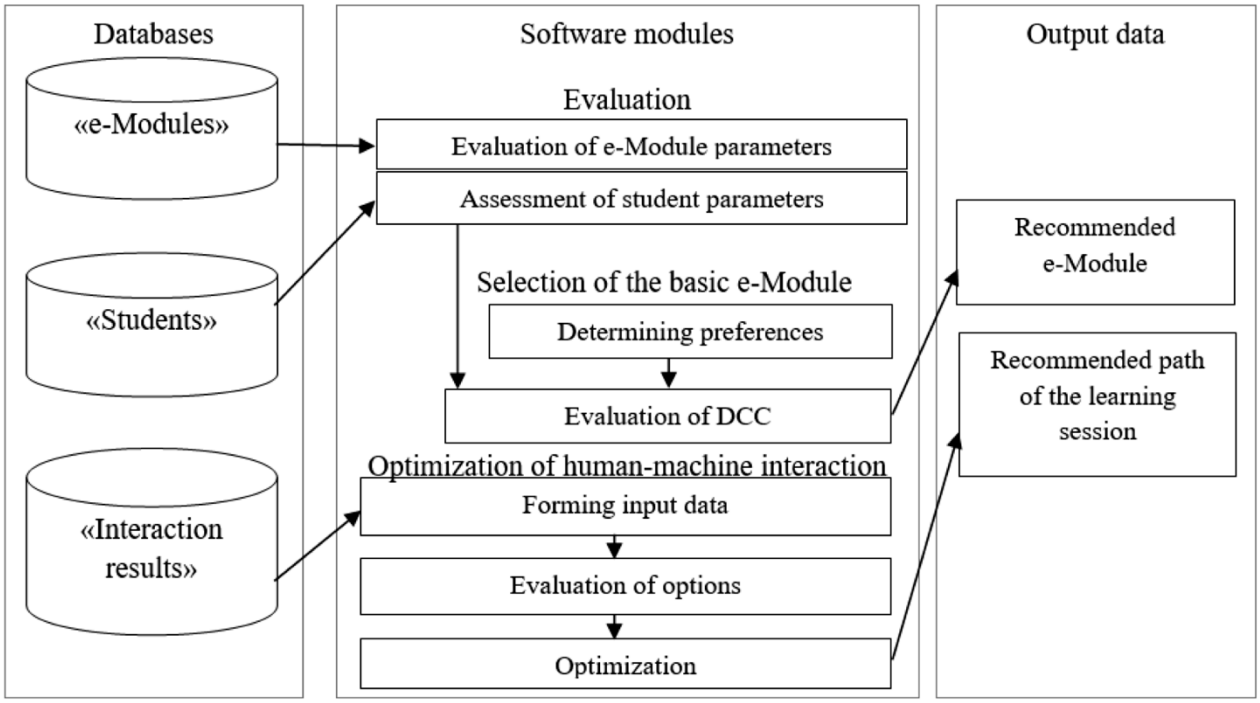


Рис. 1.7 - Архітектурна модель адаптивної e-learning системи (за Barchenko et al., 2022)

На основі аналізу наведених моделей можна зробити висновок, що сучасні підходи до адаптивного навчання поєднують математичні моделі когнітивного стану, ієрархічні структури контенту та аналітику користувацьких переваг. У рамках нашого дослідження розвинено ці методологічні засади шляхом створення **інтелектуального алгоритмічного ядра Java Swing-системи**, що поєднує два нові елементи:

1.3 Аналіз існуючих рішень

Розвиток сучасних систем електронного навчання супроводжується активним формуванням інтегрованих середовищ управління контентом, користувачами та аналітикою навчального процесу. Серед найпоширеніших рішень слід відзначити Moodle, Google Classroom, Canvas LMS, Blackboard Learn та Blend-ed Platform, кожна з яких реалізує власну архітектурну ідеологію та підхід до автоматизації освітнього циклу.

На рис. 1.8 подано інтерфейс системи Moodle, яка є найпопулярнішою платформою з відкритим кодом для організації дистанційного навчання. Moodle підтримує модульну структуру плагінів, що забезпечує гнучке налаштування функцій - від завдань і форумів до аналітики успішності. Вона орієнтована на клієнт-серверну архітектуру та вимагає веб-середовища з підтримкою PHP, MySQL або PostgreSQL. Основна перевага - масштабованість і розгалужена спільнота, проте інтерфейс Moodle потребує додаткової адаптації для офлайн-роботи або десктопного використання.

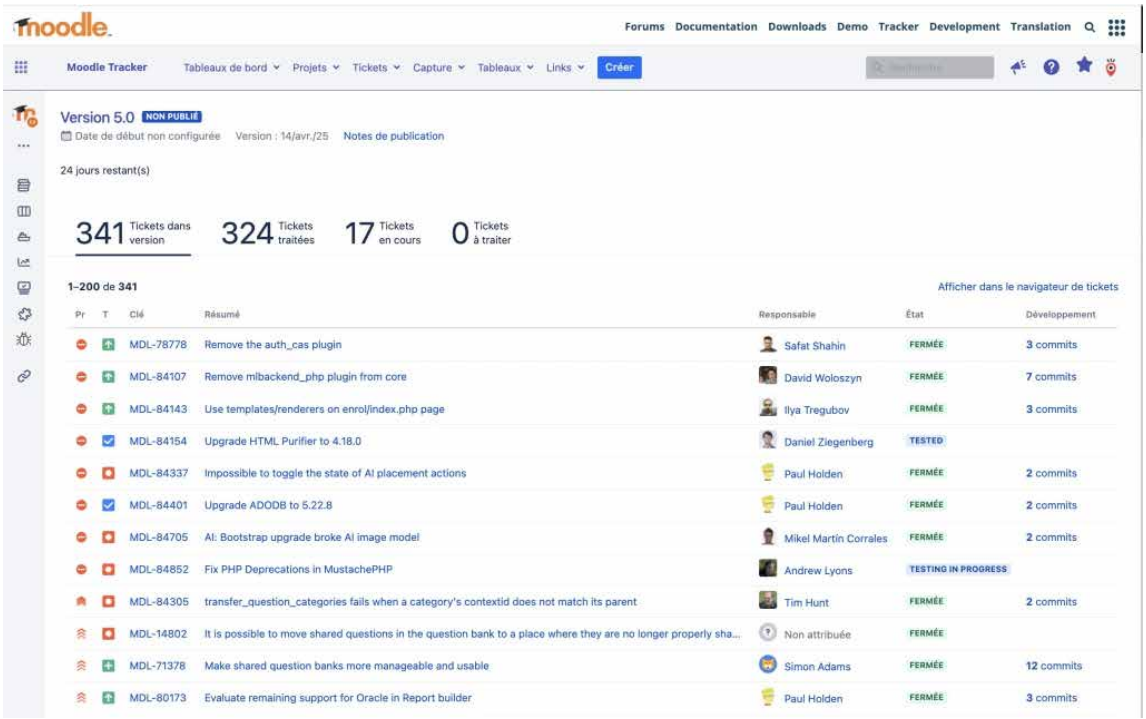


Рис. 1.8 - Інтерфейс і панель управління задачами системи Moodle (версія 5.0, розробка в JIRA Tracker)

Наступне рішення - Google Classroom, що реалізує хмарну модель інтеграції між користувачами, контентом і засобами Google Workspace (Docs, Drive, Meet). На рис. 1.9 зображено типову панель курсів Google Classroom, де акцент зроблено на візуальній простоті та мобільності. Система орієнтована на швидку публікацію завдань, але має обмеження у функціях розширеної аналітики, що робить її менш придатною для наукових і корпоративних освітніх програм [2].

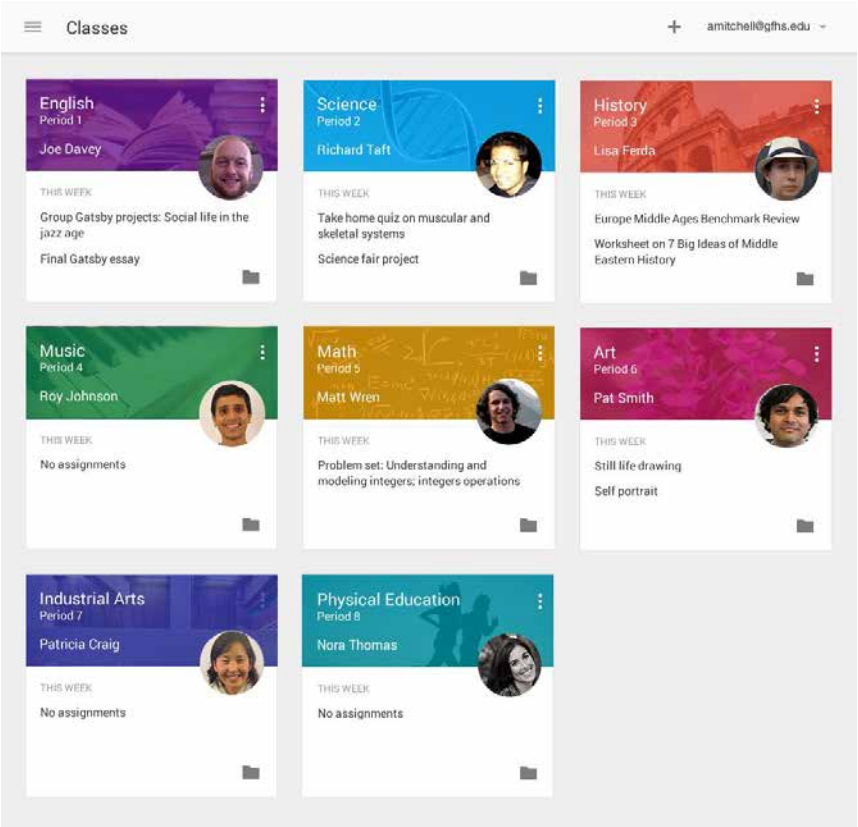


Рис. 1.9 - Інтерфейс курсів у Google Classroom (приклад панелі викладача)

На рис. 1.10 представлено систему Canvas LMS, розроблену компанією Instructure. Вона характеризується сучасним REST-орієнтованим API, підтримкою SCORM-стандартів і розвиненими засобами створення тестів та адаптивних маршрутів. Canvas має потужний механізм інтеграції з аналітичними панелями успішності, проте її використання потребує постійного з’єднання з сервером і суттєвих ресурсів бази даних, що унеможливлює автономну роботу без мережевого доступу.

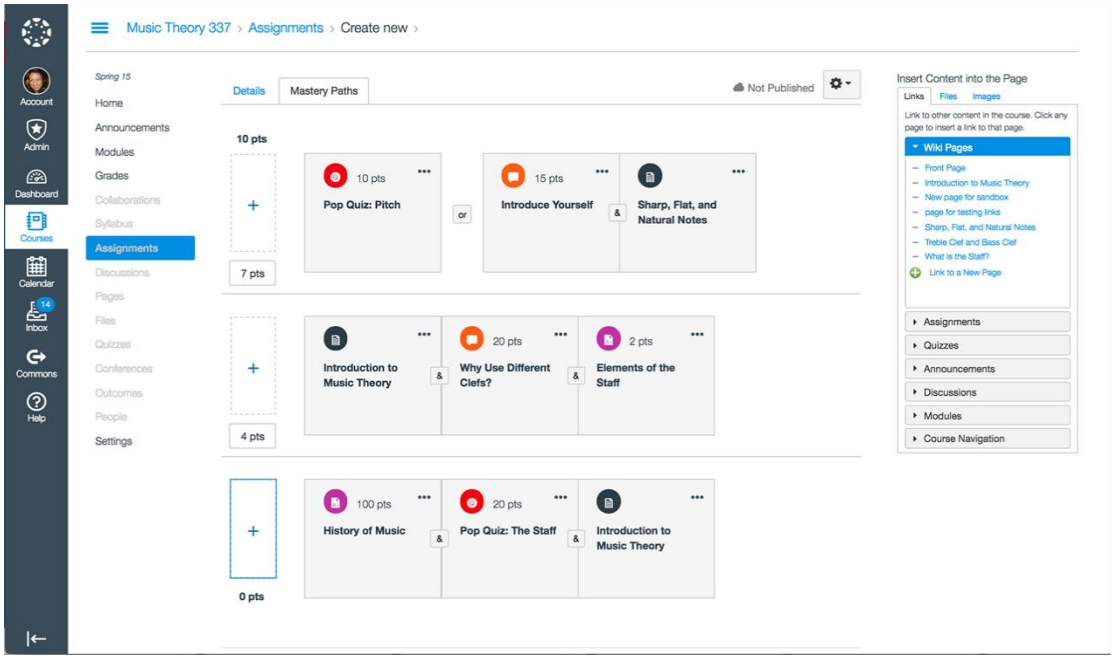


Рис. 1.10 - Інтерфейс модулів і завдань у Canvas LMS

Система Blackboard Learn, подана на рис. 1.11, орієнтована на академічне середовище з акцентом на адміністративному контролі, аналітиці й комунікації між студентами та викладачами. Blackboard підтримує централізовану модель зберігання курсів і звітів, що полегшує управління великими освітніми установами. Основним недоліком залишається складність користувацького інтерфейсу та висока вартість корпоративної ліцензії [3].

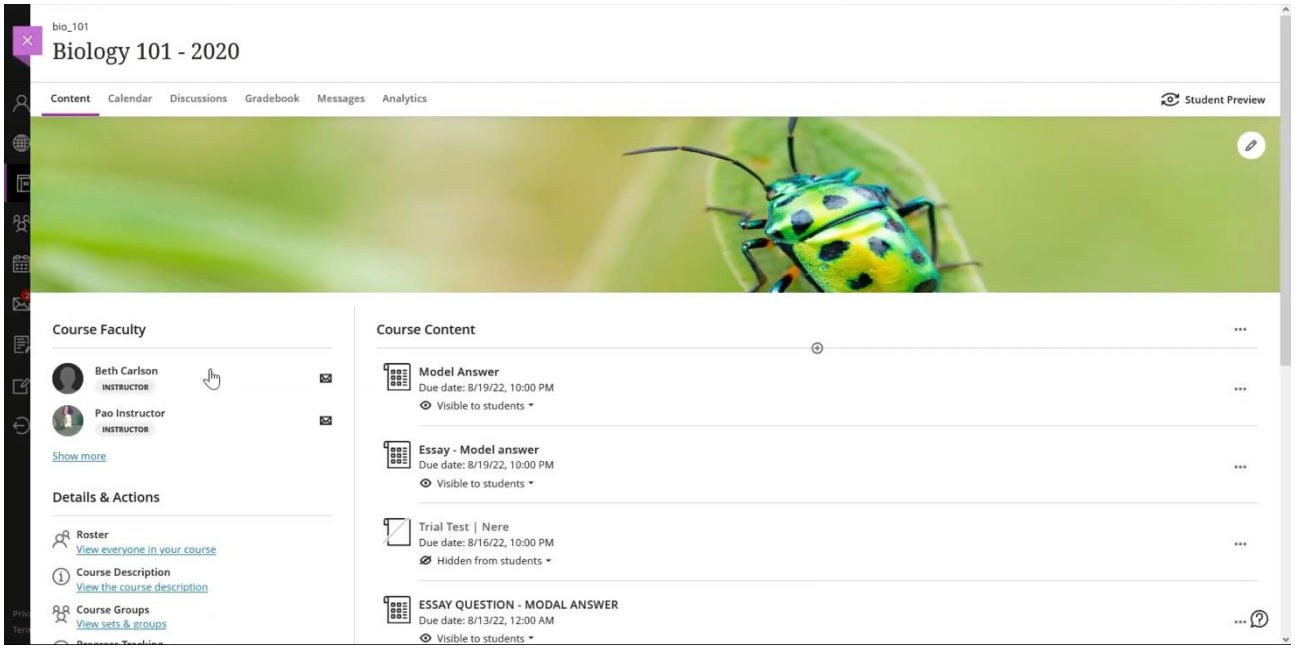


Рис. 1.11 - Навчальний курс у Blackboard Learn з аналітичними віджетами викладача

Платформа Blend-ed Platform (рис. 1.12) є новітнім SaaS-рішенням, яке поєднує елементи гейміфікації, моніторинг активності, рекомендаційні механізми та персоналізовані аналітичні панелі. Її архітектура базується на мікросервісній моделі з підтримкою REST API, що забезпечує масштабування для корпоративних клієнтів. Система має розширену візуалізацію прогресу та інтеграцію з LMS-платформами, однак не допускає повного офлайн-режиму й вимагає стабільного підключення до хмарного середовища [4].

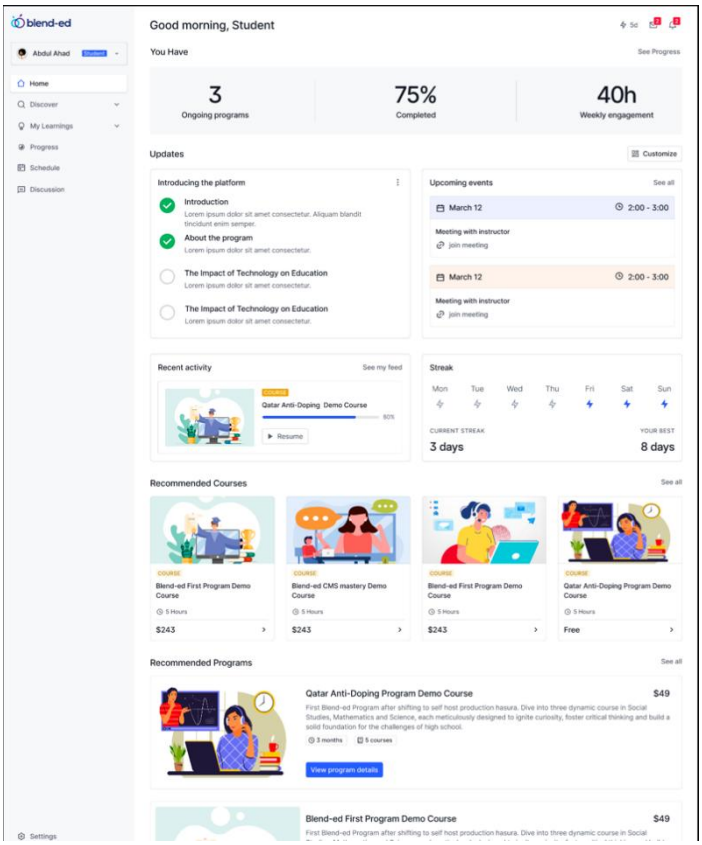


Рис. 1.12 - Персоналізована аналітична панель користувача в Blend-ed Platform

Порівняльний аналіз основних рішень (табл. 1.4) показує, що більшість систем реалізує веб-архітектуру з орієнтацією на браузерну взаємодію. Натомість розроблювана нами система дистанційного електронного навчання створюється у середовищі Java Swing як автономне десктопне програмне забезпечення. Вона поєднує адаптивну модель навчання, аналітичний модуль оцінювання когнітивного коефіцієнта знань KiKi і механізм локальної бази даних SQLite для збереження прогресу без постійного з’єднання з сервером.

Таблиця 1.2

Порівняння існуючих систем дистанційного навчання та запропонованої Java Swing-системи

Система	Тип архітектури	Адаптивність	Офлайн-режим	Інтеграція з аналітикою	Призначення
Moodle	Веб/клієнт-сервер	Обмежена (через плагіни)	Ні	Так	Освітні установи
Google Classroom	Хмарна	Часткова	Ні	Обмежена	Школи, дистанційні курси
Canvas LMS	Веб/REST API	Так	Ні	Розширена	Університети, МООС
Blackboard Learn	Централізована	Так	Ні	Так	ВНЗ, корпоративне навчання
Blend-ed Platform	SaaS/мікросервісна	Так	Ні	Так	Корпоративні середовища
Java Swing System (розробка)	Десктоп/автономна	Так (через модуль оцінки K_i)	Так	Так (локальна аналітика)	Автономне навчання без сервера

Результати аналізу свідчать, що переважна більшість LMS орієнтована на хмарну або клієнт-серверну модель, яка потребує постійного з’єднання з мережею та використання браузера. У межах цього дослідження сформульовано альтернативний підхід - створення адаптивного десктопного ПЗ для дистанційного навчання, що реалізує інтелектуальні алгоритми оцінювання, рекомендацій та візуалізації прогресу у локальному середовищі Java Swing. Наукова новизна полягає у поєднанні методів адаптивного контент-менеджменту, локальної аналітики знань і модульної архітектури, що забезпечує повну автономність користувача без залежності від зовнішніх веб-сервісів.

1.4 Моделювання програмної системи

Моделювання предметної області інформаційної системи моніторингу ключових показників успішності студентів передбачає формалізацію основних суб’єктів, процесів та інформаційних взаємозв’язків, що забезпечують повний

цикл збору, оброблення й аналізу навчальних даних. Метою моделювання є побудова формального опису логіки функціонування системи, починаючи від автентифікації користувачів і завершуючи формуванням аналітичних показників та персоналізованих навчальних рекомендацій.

Для опису предметної області застосовано комплекс UML-діаграм, зокрема діаграму прецедентів, діаграму послідовності та діаграму активності, які дозволяють відобразити функціональні можливості системи, динаміку взаємодії між компонентами та поведінкові сценарії користувачів. Використання UML-моделювання забезпечує уніфіковане представлення архітектурних рішень і підвищує узгодженість між етапами проєктування та реалізації програмного забезпечення.

На рис. 1.13 наведено діаграму прецедентів інформаційної системи моніторингу успішності студентів, яка визначає основних акторів системи: студента, викладача, адміністратора, зовнішній сервіс автентифікації (IdP/SSO) та зовнішні освітні інформаційні ресурси (ERP / електронний журнал). Діаграма описує ключові сценарії використання системи, зокрема «Вхід до системи», «Перегляд показників успішності», «Отримання навчальних рекомендацій», «Аналіз результатів навчання», «Формування аналітичних звітів» та «Керування користувачами».

Зв’язки типу «include» та «extend» використовуються для повторного застосування процесів автентифікації, перевірки прав доступу та оновлення навчальних показників. Такий підхід забезпечує декомпозицію функціональності та відображає поділ системи на підсистеми збору даних, аналітики, рекомендацій і адміністрування, що відповідає принципам модульної архітектури інформаційних систем.

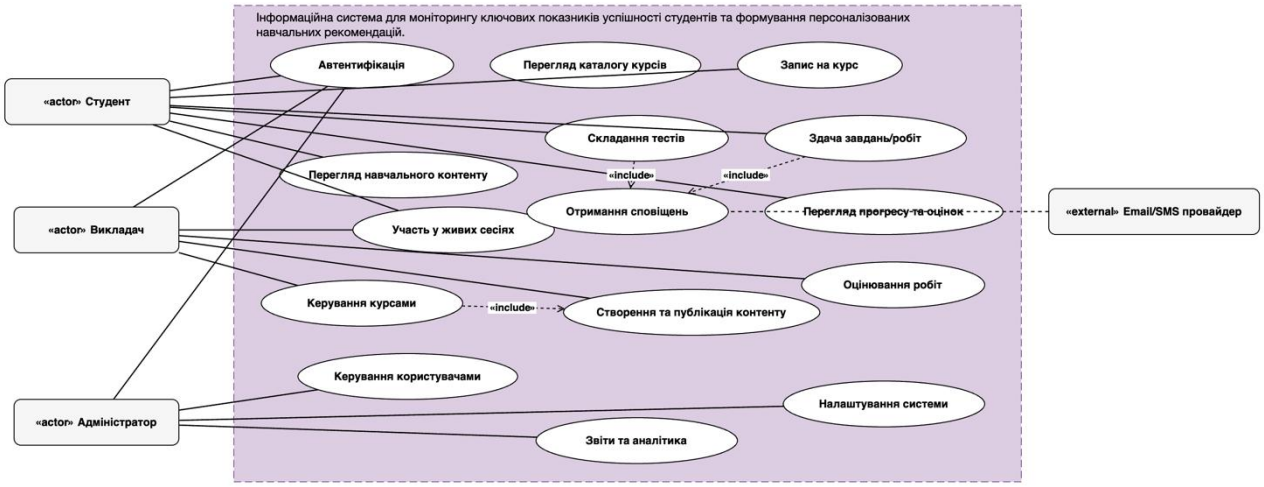


Рис. 1.13 - Діаграма прецедентів інформаційної системи моніторингу успішності студентів

Деталізацію взаємодії між програмними компонентами представлено на діаграмі послідовності (рис. 1.14), яка відображає типовий сценарій роботи користувача із системою. Сценарій включає автентифікацію користувача, запит на отримання показників успішності, оброблення навчальних даних та формування персоналізованих рекомендацій. Обмін повідомленнями між клієнтським застосунком, сервісом автентифікації, модулем оброблення навчальних даних, аналітичним ядром та базою даних реалізується через REST-взаємодію з використанням захищених токенів доступу.

На етапі аналізу навчальних результатів дані передаються до аналітичного модуля, який здійснює розрахунок ключових показників успішності та генерує рекомендації відповідно до заданих правил або моделей аналізу. Така послідовність взаємодії забезпечує узгодженість інформаційних потоків і чітку логіку функціонування компонентів у клієнт–серверному середовищі програмного забезпечення.

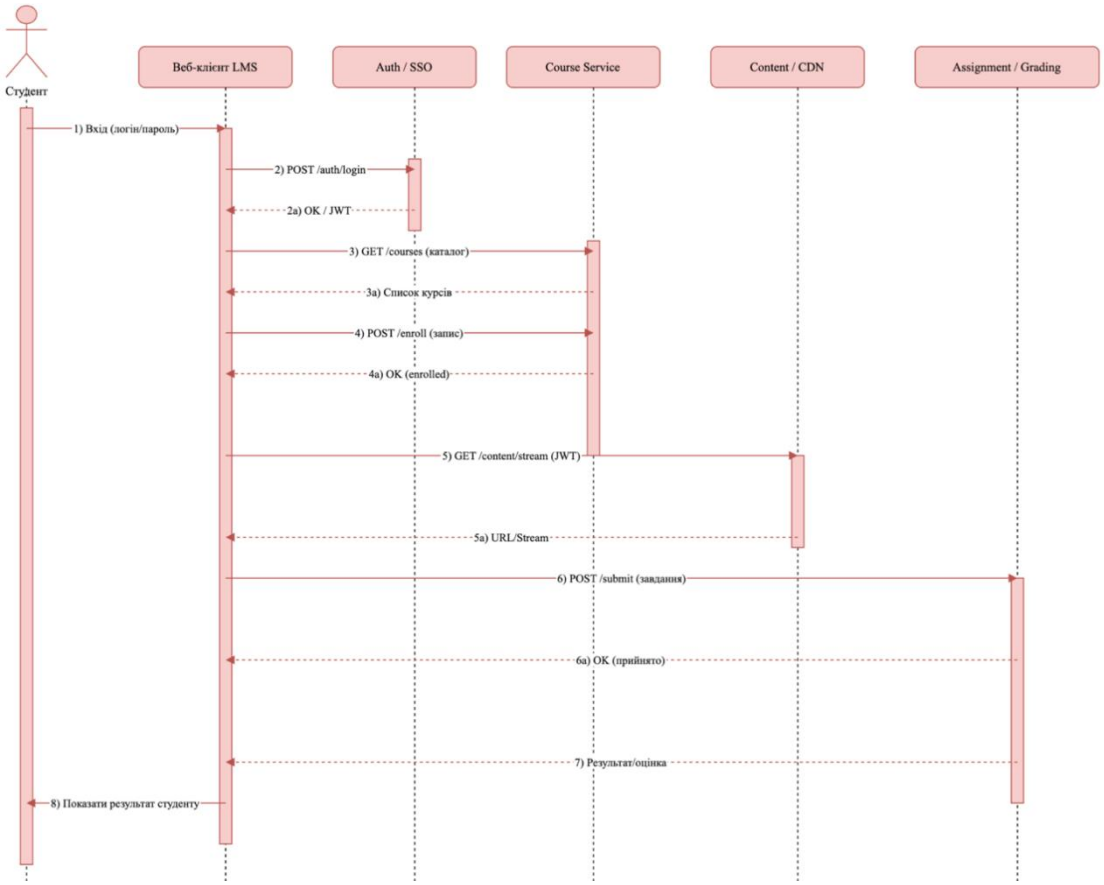


Рис. 1.14 - Діаграма послідовності процесу взаємодії користувача з інформаційною системою моніторингу успішності

Узагальнення поведінкових сценаріїв системи представлено на діаграмі активності (рис. 1.15), яка моделює основні етапи життєвого циклу взаємодії користувача з інформаційною системою. Діаграма охоплює процеси введення облікових даних, автентифікації, отримання навчальних показників, аналізу успішності, формування рекомендацій та завершення сеансу роботи.

Для формалізації відповідальності між учасниками процесу використано декілька смуг активності (swimlane): «Студент», «Інформаційна система» та «Аналітичний модуль». Логічні шлюзи («користувач автентифікований?», «дані достатні для аналізу?») визначають альтернативні гілки виконання процесів. Така модель дозволяє формалізувати поведінку системи та спростити реалізацію алгоритмів аналізу успішності у програмному середовищі.

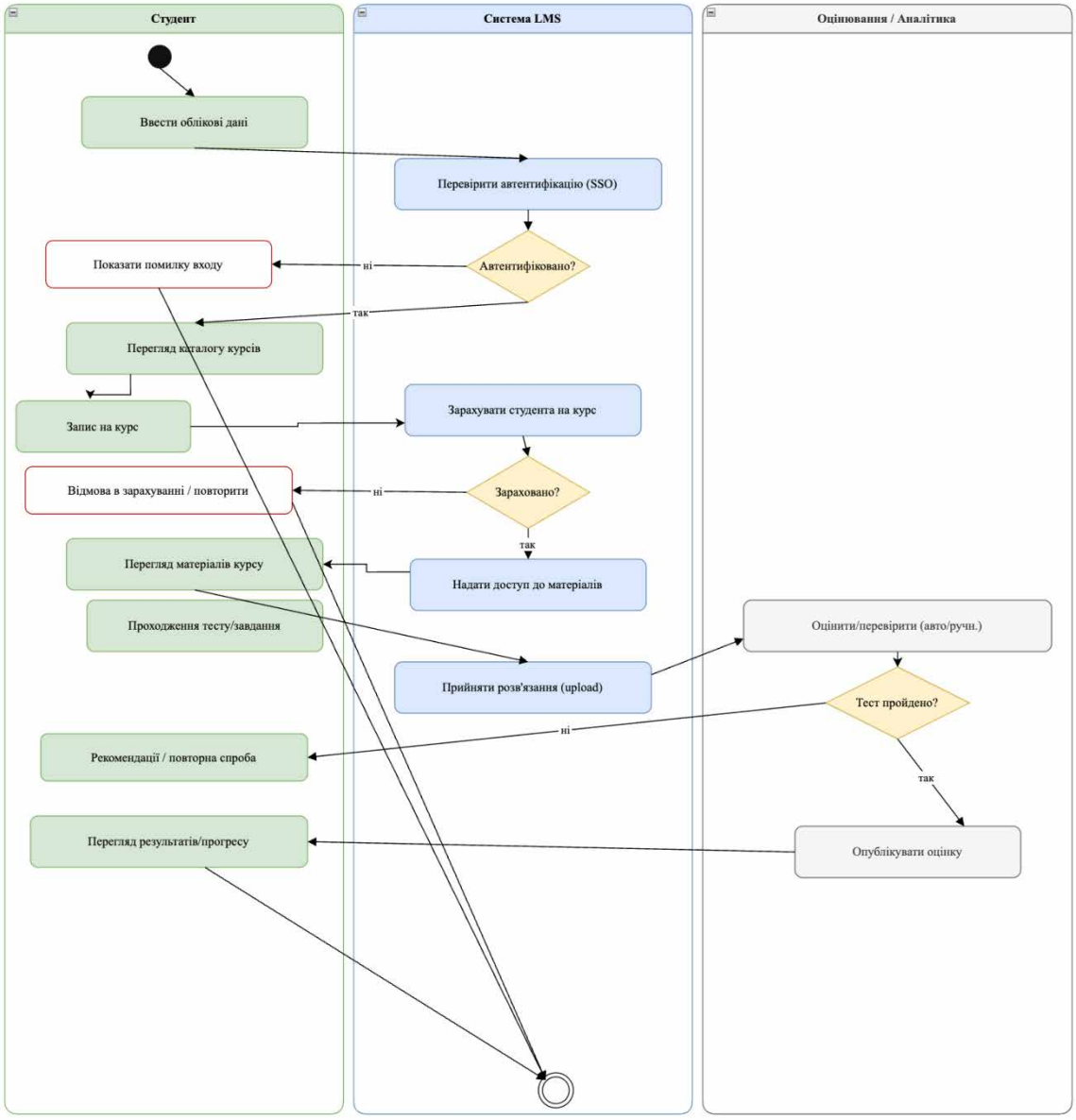


Рис. 1.15 - Діаграма активності взаємодії користувача з інформаційною системою моніторингу успішності

Узагальнення результатів моделювання предметної області дозволяє виділити такі структурно-функціональні особливості системи:

– підтримка рольової взаємодії між студентом, викладачем і адміністратором;

– централізований збір і оброблення навчальних даних із різних джерел;

– поділ функціональності на підсистеми автентифікації, моніторингу успішності, аналітики та формування рекомендацій;

– наявність механізмів зворотного зв’язку у вигляді персоналізованих навчальних рекомендацій.

Побудована модель предметної області формує логічну основу для проєктування програмного ядра інформаційної системи моніторингу успішності студентів. Запропонований підхід забезпечує модульність, масштабованість і адаптивність системи, що створює передумови для подальшої реалізації інтелектуальних методів аналізу навчальних даних та підтримки прийняття рішень у навчальному процесі.

1.5 Аналіз вимог інформаційної системи

Аналіз вимог і системи моніторингу ключових показників успішності є одним із базових етапів проєктування, що визначає цільове призначення системи, її функціональні можливості, експлуатаційні характеристики та вимоги до захисту даних. Формування вимог здійснювалося з урахуванням результатів моделювання предметної області та необхідності забезпечення автоматизованого збору, оброблення й аналізу навчальних даних з подальшим формуванням персоналізованих навчальних рекомендацій. Проєктована система розглядається як автономний програмний продукт із десктопною архітектурою, реалізований у середовищі Java Swing, що зумовлює специфічні вимоги до швидкодії інтерфейсу, локального зберігання даних і надійності аналітичних обчислень без постійного мережевого з’єднання.

Функціональні вимоги визначають перелік дій і операцій, які система повинна виконувати для забезпечення моніторингу успішності та підтримки користувачів різних ролей. Основними користувачами системи є студент, викладач і адміністратор. Система повинна забезпечувати автентифікацію користувачів, збір і зберігання результатів навчальної діяльності, розрахунок ключових показників успішності, формування аналітичних звітів і генерацію персоналізованих рекомендацій. Узагальнений перелік функціональних вимог, сформований на основі UML-моделей предметної області, наведено в таблиці 1.5.

Таблиця 1.3

Функціональні вимоги інформаційної системи моніторингу успішності студентів

№	Вимога	Опис	Роль користувача
1	Автентифікація користувача	Вхід до системи з використанням локального або зовнішнього облікового запису	Усі користувачі
2	Збір навчальних даних	Отримання результатів оцінювання та показників активності	Система
3	Перегляд показників успішності	Відображення індивідуальних KPI у табличному та графічному вигляді	Студент
4	Формування рекомендацій	Генерація персоналізованих рекомендацій на основі аналітики	Студент
5	Аналіз групових результатів	Агрегація та аналіз показників академічних груп	Викладач
6	Формування звітів	Генерація аналітичних звітів щодо успішності	Викладач, адміністратор
7	Керування користувачами	Створення, редагування та блокування облікових записів	Адміністратор
8	Налаштування показників	Визначення правил і вагових коефіцієнтів оцінювання	Адміністратор
9	Експорт результатів	Збереження аналітичних даних у форматах PDF або CSV	Викладач, адміністратор

Нефункціональні вимоги визначають якісні характеристики системи, що впливають на стабільність її роботи, зручність використання та можливість масштабування. З огляду на автономний характер програмного забезпечення основний акцент зроблено на продуктивності графічного інтерфейсу, надійності локального зберігання даних і кросплатформній сумісності. Система повинна забезпечувати мінімальний час відгуку інтерфейсу, коректну роботу з локальною базою даних і можливість подальшого розширення аналітичної функціональності. Узагальнення нефункціональних вимог наведено в таблиці 1.6.

Таблиця 1.4

Нефункціональні вимоги інформаційної системи моніторингу успішності

№	Категорія	Вимога	Показник
1	Продуктивність	Час відгуку інтерфейсу	$\leq 0{,}2$ с
2	Надійність	Збереження даних при аварійному завершенні	$\geq 99{,}9$ %

Продовження таблиці 1.4

3	Обсяг даних	Підтримка локальної БД	$\geq 10^4$ записів
4	Сумісність	Робота в Windows, Linux, macOS	Повна
5	Інтерфейс	Стабільна робота Java Swing GUI	Висока
6	Розширюваність	Підключення нових аналітичних модулів	Передбачено

Окрему групу становлять вимоги до безпеки, оскільки система оперує персональними даними студентів і результатами навчальної діяльності. Система повинна забезпечувати контроль доступу, захист даних від несанкціонованих змін і збереження цілісності аналітичної інформації. Основні вимоги до безпеки подано в таблиці 1.7.

Таблиця 1.5

Вимоги до безпеки інформаційної системи моніторингу успішності

№	Вимога	Опис
1	Контроль доступу	Автентифікація та перевірка прав користувачів
2	Рольова модель	Розмежування доступу за ролями
3	Шифрування	Захист локальних даних бази
4	Журнали подій	Фіксація входів і змін даних
5	Цілісність	Контроль коректності збереженої інформації
6	Захист звітів	Обмеження доступу до аналітичних файлів

Проведений аналіз вимог дозволяє сформувати рамкову специфікацію інформаційної системи, орієнтованої на аналітичну підтримку навчального процесу. Запропонована система зосереджена на моніторингу ключових показників успішності та формуванні персоналізованих рекомендацій, а не на управлінні навчальним контентом. Десктопна архітектура забезпечує автономність роботи, високу швидкодію та незалежність від мережевої інфраструктури, що створює передумови для ефективного використання системи в умовах реального освітнього процесу.

1.6 Постановка завдання

На основі проведеного аналізу предметної області, сформульованих вимог і результатів моделювання процесів інформаційної системи моніторингу успішності студентів було визначено постановку завдання, яка формалізує мету,

структуру та принципи функціонування програмного забезпечення. Завдання полягає у проєктуванні та розробці інформаційної системи, призначеної для автоматизованого збору, оброблення й аналізу навчальних даних з метою моніторингу ключових показників успішності студентів і формування персоналізованих навчальних рекомендацій із використанням методів машинного навчання.

Проєктована система розглядається як автономний програмний комплекс із десктопною архітектурою, реалізований у середовищі Java Swing, що забезпечує незалежність від мережевої інфраструктури, високу швидкодію та локальне зберігання даних. Система повинна підтримувати повний цикл аналітичної обробки навчальної інформації – від автентифікації користувачів і збору первинних освітніх даних до формування аналітичних звітів і рекомендацій для коригування навчальної траєкторії.

Для досягнення поставленої мети необхідно спроєктувати архітектуру системи, що складається з взаємопов’язаних функціональних підсистем. Підсистема користувацької взаємодії забезпечує роботу студентів, викладачів і адміністраторів через локальний графічний інтерфейс, надаючи доступ до показників успішності, звітів і рекомендацій. Підсистема управління даними відповідає за зберігання, оновлення та цілісність навчальної інформації в локальній базі даних, а також за підготовку даних до аналітичної обробки. Аналітично-рекомендаційна підсистема реалізує алгоритми аналізу навчальних даних, обчислення ключових показників успішності та формування персоналізованих рекомендацій із використанням методів машинного навчання.

Вхідними даними системи є облікові дані користувачів (ідентифікатор, роль, параметри доступу), результати контролю знань, оцінки та кількість спроб виконання завдань, показники навчальної активності, зокрема час роботи з матеріалами та частота взаємодії з системою, а також параметри конфігурації аналітичних моделей і правил оцінювання. Дані можуть надходити як безпосередньо від користувача, так і шляхом імпорту з зовнішніх освітніх систем.

Вихідними даними системи є агреговані та індивідуальні показники успішності студентів, аналітичні звіти у графічному та табличному вигляді, рекомендації щодо повторного опрацювання навчального матеріалу або зміни навчальної траєкторії, а також прогнозні оцінки ризику зниження успішності. Результати оброблення зберігаються у базі даних і можуть експортуватися у формати PDF або CSV для подальшого використання викладачами й адміністрацією.

Аналітична складова системи базується на застосуванні методів машинного навчання для виявлення закономірностей у навчальних даних і прогнозування результатів навчання. Для цього передбачається використання алгоритмів класифікації та регресії з метою оцінювання рівня засвоєння матеріалу, а також моделей рекомендаційних систем для формування індивідуальних навчальних порад. Узагальненим інтегральним показником успішності є коефіцієнт навчального прогресу, який визначається як функція результатів контролю знань, рівня активності та динаміки навчання студента і використовується як вхідний параметр для моделей машинного навчання.

Основною метою постановки завдання є створення інформаційної системи, яка забезпечує адаптивну аналітичну підтримку освітнього процесу, автоматизує моніторинг ключових показників успішності та надає інтелектуальні рекомендації на основі аналізу даних. Реалізація такого підходу дозволяє підвищити об’єктивність оцінювання навчальних результатів, зменшити навантаження на викладачів і забезпечити персоналізований підхід до навчання в умовах автономного функціонування програмного забезпечення.

Результатом виконання поставленого завдання є інформаційна система моніторингу успішності студентів, здатна автоматично аналізувати навчальні дані, застосовувати методи машинного навчання для прогнозування результатів і формувати персоналізовані рекомендації, що створює передумови для підвищення ефективності управління освітнім процесом і якості навчання.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Діаграма класів і кооперації системи для моніторингу ключових показників успішності студентів

Логічна структура програмного забезпечення системи моніторингу ключових показників успішності студентів формується через формалізацію взаємозв'язків між ключовими компонентами доменної моделі, що реалізують повний цикл навчального процесу: автентифікація, керування курсами, проведення тестування, обчислення когнітивного коефіцієнта знань Ki та формування аналітичних звітів. На рис. 2.1 наведено діаграму класів, побудовану відповідно до принципів об'єктно-орієнтованого моделювання, інкапсуляції та структурної нормалізації доменних об'єктів. Вона не лише відображає статичну структуру системи, але й демонструє, як класові сутності узгоджено взаємодіють із логічною моделлю даних, забезпечуючи несуперечливість між інформаційною та поведінковою складовими архітектури.

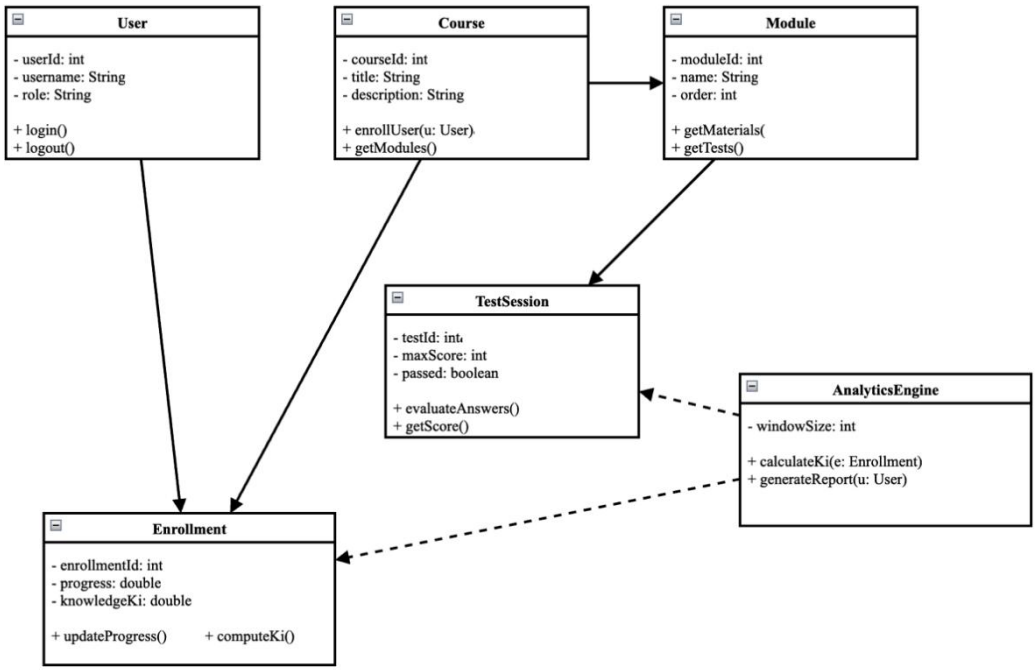


Рис. 2.1 – UML-діаграма класів підсистеми

Клас User реалізує уніфіковану модель користувача (студента або викладача), зберігаючи атрибути ролі та автентифікаційні методи. Клас Course інкапсулює параметри навчальних курсів і є центральною сутністю,

що агрегує модулі (Module) та дозволяє виконувати операції запису користувачів. Елемент Enrollment забезпечує нормалізований зв’язок «багато-до-багатьох» між користувачами та курсами, усуваючи дублювання атрибутів і дозволяючи зберігати параметри прогресу, включаючи когнітивний показник Ki. Компонент TestSessionінкапсулює логіку оброблення тестових спроб, а AnalyticsEngine виконує алгоритмічний аналіз даних, забезпечуючи адаптивність навчання на рівні окремого студента.

Для відображення поведінкової взаємодії класів застосовано кооперації, які визначають послідовність обміну повідомленнями між об’єктами в конкретних сценаріях. На рис. 2.2 подано кооперацію для процесу автентифікації та відображення каталогу курсів, де LoginForm комунікує з AuthService, отримує підтвердження автентичності та ініціює завантаження навчальних матеріалів через CourseCatalog.

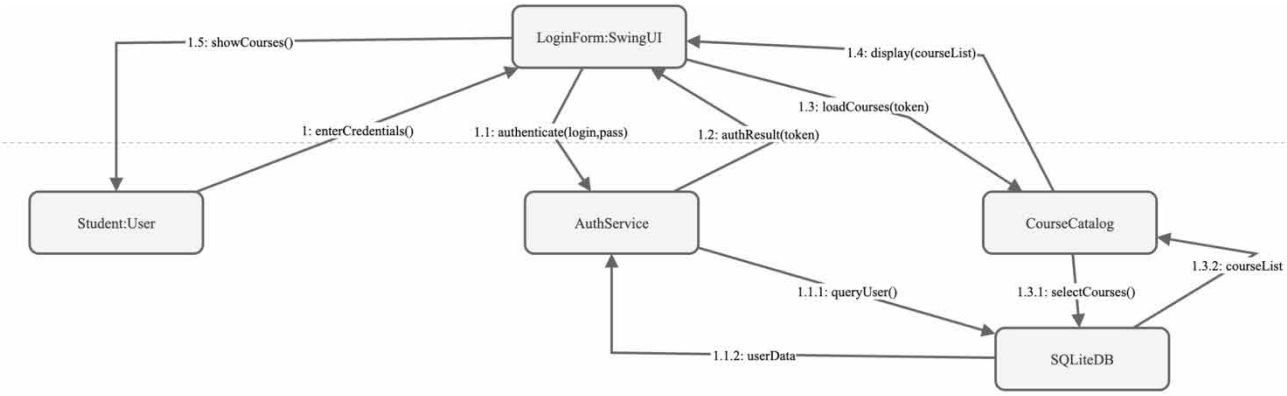


Рис. 2.2 – Кооперація «Автентифікація та завантаження курсу»

На рис. 2.3 відтворено кооперацію проведення тестування, яка демонструє взаємодію між CourseUI, TestEngine, AnalyticsEngine та SQLiteDB. Така модель дозволяє чітко відокремити бізнес-логіку тестування, механізми оцінювання відповідей та модуль аналітики, що обчислює значення Ki на основі результатів студента. Важливо, що структура кооперації відповідає вимогам до модульності та забезпечує можливість незалежного удосконалення аналітичного блоку без зміни інших компонентів системи.

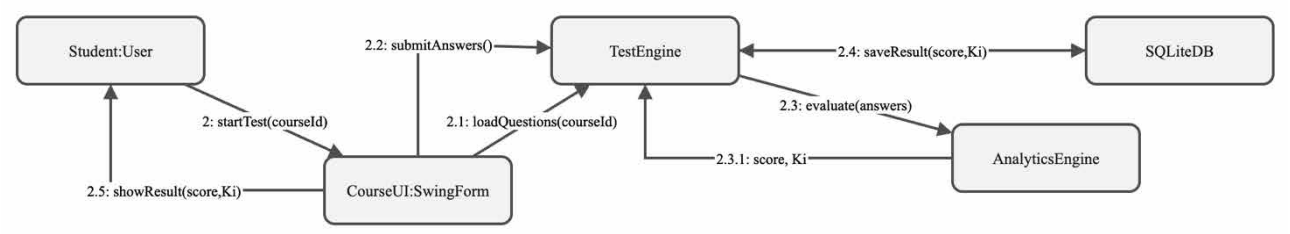


Рис. 2.3 – Кооперація «Проведення тестування та розрахунок коефіцієнта знань Ki»

На рис. 2.4 наведено кооперацію роботи викладача з аналітичним модулем, де Dashboard, AnalyticsEngine та ReportGenerator реалізують циклічну взаємодію для формування звітів про успішність студентів. Така поведінкова модель повністю відповідає вимогам до експертної системи, орієнтованої на автономну роботу у середовищі Java Swing, та забезпечує формування PDF/CSV-звітів на основі локально збережених даних.

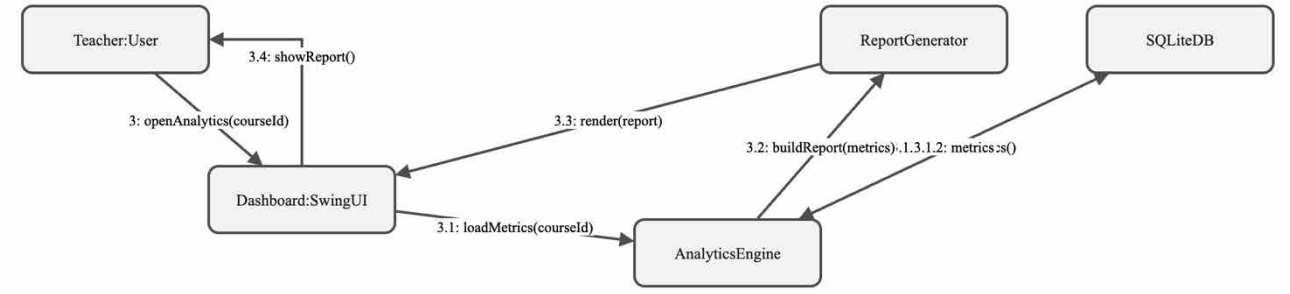


Рис. 2.4 – Кооперація «Аналітична панель викладача та формування звітів»

У таблиці 2.1 подано узагальнену характеристику ключових класів, що входять до програмної архітектури. Представлені атрибути та функції відображають нормалізовану структуру інформаційних одиниць, узгоджену з логічною моделлю даних, та забезпечують мінімальну зв’язаність класів при максимальній функціональній декомпозиції.

Таблиця 2.1

Основні класи та їх функціональне призначення

Клас	Основні атрибути	Призначення
User	userId, username, role	Зберігання облікових даних, автентифікація
Course	courseId, title, description	Логічний контейнер навчальних модулів
Module	moduleId, name, order	Структурна одиниця курсу, містить матеріали і тести
Enrollment	enrollmentId, progress, knowledgeKi	Відстеження навчального прогресу студента

Продовження таблиці 2.1

TestSession	testId, maxScore, passed	Реалізація тестування й оброблення відповідей
AnalyticsEngine	windowSize	Обчислення когнітивного коефіцієнта Ki та генерація звітів

У результаті узагальнення можна стверджувати, що діаграма класів і кооперацій формує основу поведінкової логіки системи й забезпечує структурну узгодженість усіх компонентів. Вона демонструє модульність, низьку зв’язаність і високу внутрішню когерентність підсистем, що критично важливо для адаптивного навчального середовища, орієнтованого на локальну роботу без серверної інфраструктури. Структуризація системи через класи та кооперації забезпечує масштабованість, можливість подальшого розширення і формує надійну архітектурну основу для наступних етапів розроблення.

2.2 Архітектурне подання системи у вигляді діаграми компонентів

Архітектурне подання експертної системи базується на модульному принципі побудови, який забезпечує незалежність функціональних блоків, спрощує модернізацію та підтримує можливість автономного виконання без серверної інфраструктури. На рис. 2.5 наведено діаграму компонентів, яка відображає структурну взаємодію між клієнтським інтерфейсом, підсистемою автентифікації, менеджером курсів і модулів, аналітичним ядром з обчислення коефіцієнта знань Ki, а також підсистемою збереження даних на основі локальної бази SQLite. Обрана архітектурна декомпозиція повністю відповідає принципам інкапсуляції й низької зв’язаності, що дозволяє розподілити відповідальності між компонентами та мінімізує міжмодульні залежності.

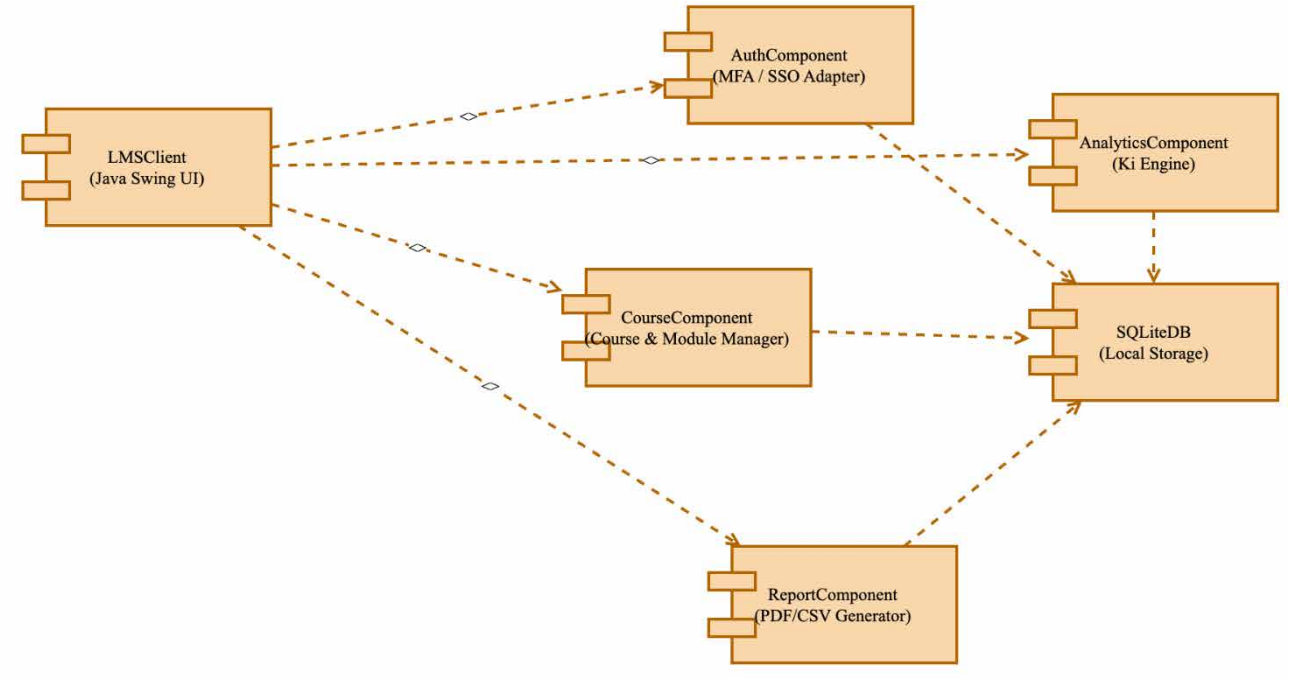


Рис. 2.5 – Діаграма компонентів системи

Як показано на рис. 2.5, основним елементом є компонент LMSClient, реалізований засобами Java Swing, який формує графічний інтерфейс користувача та ініціює виклики до інших підсистем. Компонент AuthComponentзабезпечує підтримку локальної автентифікації, адаптера MFA/SSO та логіку керування доступом згідно з ролями користувачів. Модуль CourseComponent відповідає за завантаження, структурування та керування навчальними курсами й модулями, а також реалізує стандартизований API для інших частин системи. Аналітичний компонент AnalyticsComponent виконує обчислення коефіцієнта знань Ki, аналіз навчальних траєкторій та формування агрегованих метрик успішності, що є ядром інтелектуальної складової системи. Компонент ReportComponentреалізуЄ генерацію звітів у форматах PDF/CSV, використовуючи агреговані аналітичні дані. Локальна база SQLiteDB виступає універсальним сховищем структурованої інформації – від облікових записів і курсів до результатів тестування та значень індивідуальних коефіцієнтів знань.

У таблиці 2.2 подано узагальнену характеристику компонентів системи, їх функціональні зони відповідальності та ключові дані, з якими вони працюють. Такий поділ забезпечує незалежне оновлення аналітичних і адміністративних функцій, відповідність принципам модульної архітектури та можливість

масштабування системи шляхом додавання нових підсистем (наприклад, чат-сервісу, адаптивної рекомендаторної моделі, інтеграції з ERP або зовнішніми освітніми платформами).

Таблиця 2.2

Основні компоненти системи та їх функціональні ролі

Компонент	Функціональне призначення	Основні дані / API
LMSClient	Взаємодія з користувачем, навігація, відображення контенту	UI-події, запити до менеджерів
AuthComponent	Автентифікація, SSO/MFA, контроль доступу	Облікові записи, токени доступу
CourseComponent	Управління курсами, модулями, тестами	Метадані курсів і модулів
AnalyticsComponent	Обчислення *Ki*, аналіз навчального прогресу	Дані спроб тестування, історія навчання
ReportComponent	Генерація PDF/CSV звітів	Аналітичні метрики, шаблони звітів
SQLiteDB	Локальне зберігання результатів, курсів, користувачів	SQL-таблиці, CRUD-операції

Узагальнюючи, діаграма компонентів дозволяє оцінити архітектурну зрілість системи та виокремити ключові механізми забезпечення її адаптивності: чітку модульність, стандартизовані інтерфейси між компонентами, автономне локальне зберігання даних і незалежний аналітичний модуль. Такий підхід сприяє гнучкому масштабуванню, спрощує тестування та забезпечує основу для безпечного розширення функціональності системи в рамках загальної архітектурної моделі дистанційного електронного навчання.

2.3 Пакетна структуризація програмної архітектури системи

Пакетна декомпозиція експертної системи забезпечує формалізоване розділення програмної логіки на ізольовані підсистеми, що спрощує підтримку коду, модульне тестування та поступове масштабування функціональності. На рис. 2.6 подано діаграму пакетів, у якій структуру системи поділено на логічні домени відповідно до функціональних зон - ядро програми, інтерфейс користувача, модулі курсового менеджменту, підсистему аналітики Ki, модуль автентифікації та пакет збереження даних. Такий поділ є необхідним для

побудови автономної Java Swing-системи, яка об’єднує офлайн-взаємодію, аналітичні алгоритми та локальну персистентність у єдиний структурований простір.

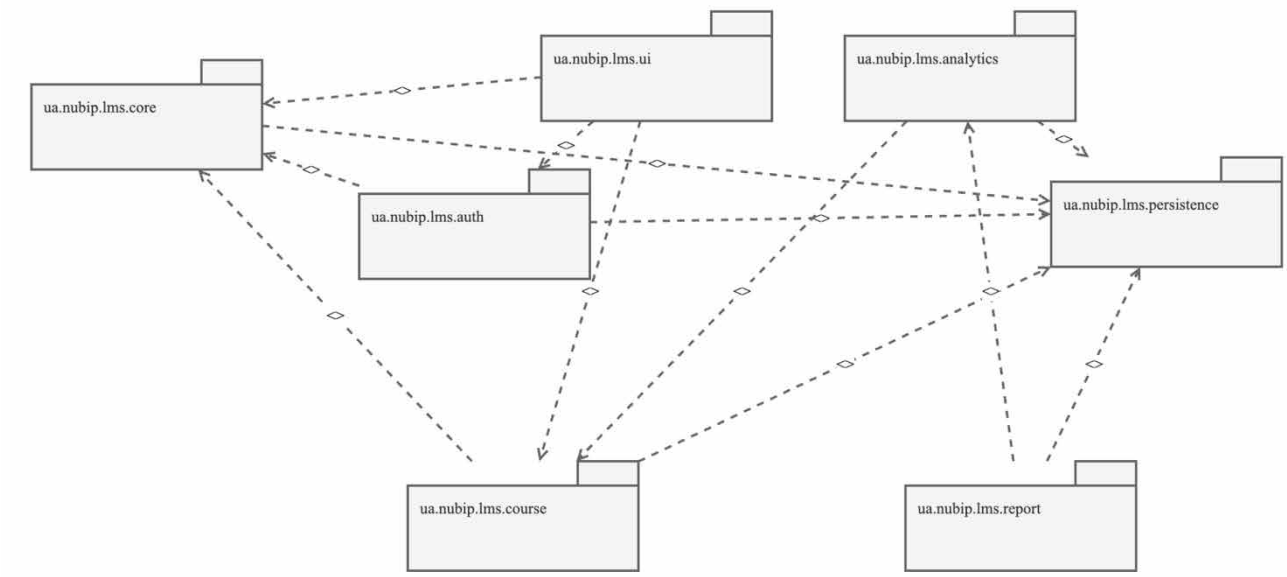


Рис. 2.6 – Пакетна структура експертної системи дистанційного електронного навчання

Структура пакетів узгоджується з вимогами модульності, визначеними на етапах аналізу вимог та моделювання. Зокрема, пакет ua.nubip.lms.core виконує роль системного ядра, що координує загальні механізми роботи, тоді як ua.nubip.lms.ui ізольовано відповідає за графічний інтерфейс. Пакет ua.nubip.lms.course містить засоби управління курсами та навчальними модулями, а ua.nubip.lms.analytics реалізує аналітичні алгоритми та обчислення когнітивного коефіцієнта знань Ki, інтегруючи дані про прогрес навчання. Блок ua.nubip.lms.auth забезпечує незалежність механізмів автентифікації, що важливо для безпеки та підтримки SSO/MFA-розширень, а ua.nubip.lms.persistence відповідає за роботу з локальною базою SQLite. Пакет ua.nubip.lms.report забезпечує формування та експорт звітів успішності. Така структура пакетів дозволяє мінімізувати перехресні залежності, відокремити бізнес-логіку від інтерфейсу та забезпечити передбачувану еволюцію системи за рахунок можливості розширення окремих модулів без рефакторингу всієї програми.

У таблиці 2.3 подано узагальнення функціонального призначення пакетів, що входять до архітектури системи. Представлена структуризація чітко відображає розподіл відповідальностей між підсистемами, підвищує внутрішню когерентність і забезпечує відповідність принципам побудови складних автономних навчальних платформ.

Таблиця 2.3

Пакети системи та їх функціональні області

Пакет	Функціональна роль
ua.nubip.lms.core	Базова логіка системи, координація роботи модулів
ua.nubip.lms.ui	Графічний інтерфейс користувача, навігація, взаємодія з компонентами
ua.nubip.lms.auth	Автентифікація, MFA/SSO, механізми контролю доступу
ua.nubip.lms.course	Керування курсами, навчальними модулями та тестовими елементами
ua.nubip.lms.analytics	Алгоритми аналізу, обчислення коефіцієнта знань *Ki*, агрегування метрик
ua.nubip.lms.persistence	Локальне зберігання даних, робота з SQLite, транзакційний доступ
ua.nubip.lms.report	Генерація PDF/CSV-звітів, формування аналітичних представлень

Узагальнення результатів свідчить, що пакетна структура не лише впорядковує програмний код, але й формує архітектурну основу для стійкої роботи системи в автономному режимі, забезпечуючи незалежність модулів, структурованість взаємозв’язків і можливість інтегрування нових функціональних блоків зі збереженням цілісності програмного середовища.

2.4 Висновки до другого розділу

У другому розділі було сформовано повний комплекс архітектурно-структурних моделей, які визначають логічну, інформаційну та функціональну організацію експертної системи дистанційного електронного навчання. На основі аналізу доменних вимог побудовано нормалізовану логічну модель даних у вигляді ER-діаграми, що забезпечує узгодженість між інформаційними об’єктами, мінімізує дублювання даних і створює формальний базис для

подальшої реалізації локального сховища на SQLite. Розроблена діаграма класів відображає структурну декомпозицію програмних об’єктів, демонструє чіткий розподіл відповідальностей між підсистемами та забезпечує цілісність поведінкової логіки через побудовані кооперації ключових сценаріїв — автентифікації, роботи з курсами, тестування та аналітичного оцінювання знань за показником *Ki*. Діаграма компонентів системи підтвердила модульний характер архітектури, її відповідність принципам інкапсуляції, низької зв’язаності та можливості автономного функціонування в офлайн-середовищі Java Swing. Пакетна структуризація інтегрувала отримані моделі в єдину логічну архітектуру, забезпечивши масштабованість, передбачуваність еволюції програмного коду та готовність системи до розширення (аналітичних модулів, засобів звітності та механізмів автентифікації). Сукупність побудованих моделей формує комплексне бачення архітектури системи, яке дозволяє перейти до етапу алгоритмізації та програмної реалізації з повним розумінням структури, функціональних залежностей і механізмів оброблення даних.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір технологій та інструментальних засобів реалізації системи

Для реалізації інформаційної системи моніторингу ключових показників успішності студентів було обрано технологічний стек, орієнтований на створення автономного настільного застосунку з локальним зберіганням даних, зручним графічним інтерфейсом та можливістю аналітичної обробки результатів навчання. Основними критеріями вибору технологій були кросплатформність, простота розгортання, стабільність роботи без серверної інфраструктури, підтримка локальної бази даних, можливість побудови аналітичних розрахунків та формування звітів.

Базовою мовою програмування обрано Python, оскільки вона забезпечує швидку розробку прикладного програмного забезпечення, має велику кількість бібліотек для роботи з даними та добре підходить для реалізації навчально-аналітичних систем. Для побудови графічного інтерфейсу використано PyQt6, що дозволяє створити зручне настільне середовище з формами авторизації, панелями перегляду успішності, таблицями результатів, аналітичними віджетами та засобами формування рекомендацій.

Зберігання даних реалізовано за допомогою SQLite. Такий вибір є доцільним для локальної інформаційної системи, оскільки база даних не потребує окремого серверного розгортання, зберігається у вигляді одного файлу та забезпечує достатню продуктивність для оброблення облікових записів користувачів, курсів, результатів тестування, показників успішності та рекомендацій. Для аналітичної обробки даних доцільно використовувати pandas та NumPy, які забезпечують обчислення середніх балів, коефіцієнта навчального прогресу, індексів активності та рівня ризику зниження успішності. Для побудови простих прогнозних або класифікаційних моделей може застосовуватися scikit-learn.

Формування звітів і візуалізація результатів можуть виконуватися за допомогою pandas, matplotlib та засобів експорту у CSV/PDF. Це дає змогу

викладачу або адміністратору отримувати узагальнені результати щодо академічної групи, окремого студента або навчального курсу. Для запуску застосунку передбачено файли start_windows.bat, start_macos.command та start_macos_linux.sh, що спрощує використання системи на різних операційних системах. Файл requirements.txt використовується для встановлення залежностей, а файл style.qss забезпечує єдине оформлення інтерфейсу.

Таблиця 3.1

Порівняльна характеристика технологій та інструментальних засобів реалізації системи

№	Компонент системи	Обрана технологія	Основне призначення	Переваги використання
1	Мова програмування	Python 3.x	Реалізація логіки застосунку, обробка даних, взаємодія з БД	Простота розробки, велика кількість бібліотек, кросплатформність
2	Інтерфейс користувача	PyQt6	Побудова графічного інтерфейсу системи	Зручні форми, таблиці, панелі керування, підтримка стилів QSS
3	База даних	SQLite	Локальне зберігання користувачів, курсів, результатів і рекомендацій	Не потребує сервера, швидке розгортання, достатня продуктивність
4	Аналітична обробка	pandas / NumPy	Розрахунок середнього бала, активності, Кі та ризику	Зручна робота з таблицями, швидкі обчислення, підтримка агрегації даних
5	Моделі оцінювання	scikit-learn	Класифікація рівня ризику та підтримка рекомендацій	Можливість подальшого розширення аналітичного модуля
6	Звітність	pandas / matplotlib / CSV / PDF	Формування таблиць, графіків і звітів	Експорт результатів, наочне подання показників успішності
7	Оформлення інтерфейсу	QSS	Налаштування стилю графічного інтерфейсу	Єдиний дизайн, зручність сприйняття, відокремлення стилів від логіки
8	Запуск системи	BAT / SH / COMMAND	Запуск застосунку в різних ОС	Спрощення розгортання для Windows, Linux і macOS

Обраний стек технологій відповідає вимогам до інформаційної системи моніторингу успішності студентів, оскільки забезпечує автономну роботу, локальне зберігання даних, просте розгортання та достатні можливості для аналітичної обробки навчальних результатів. Python у поєднанні з PyQt6 і SQLite

дозволяє реалізувати повноцінний настільний застосунок без потреби у серверній інфраструктурі, а використання pandas, NumPy та scikit-learn створює основу для розрахунку ключових показників успішності, визначення ризику зниження результатів і формування персоналізованих навчальних рекомендацій.

3.2 Представлення бази даних розроблювальної системи

Інформаційна база розроблювальної системи призначена для локального зберігання даних, необхідних для моніторингу ключових показників успішності студентів, аналізу результатів навчання та формування персоналізованих рекомендацій. Оскільки система реалізується як настільний застосунок на Python, для організації зберігання даних використано SQLite. Такий підхід забезпечує автономну роботу програмного забезпечення без необхідності розгортання окремого серверу бази даних.

Файл бази даних має назву `edu_navigator_kpi.db` і створюється автоматично під час першого запуску застосунку. У програмному коді робота з базою даних реалізована через окремий клас `DBLayer`, який відповідає за встановлення з'єднання, створення таблиць, виконання SQL-запитів, отримання записів і збереження змін. Така структура дозволяє відокремити логіку зберігання даних від інтерфейсної та аналітичної частин системи.

На рисунку 3.1 наведено фрагмент програмної реалізації шару доступу до бази даних, у якому визначено підключення до SQLite-файлу та службові методи виконання SQL-запитів.

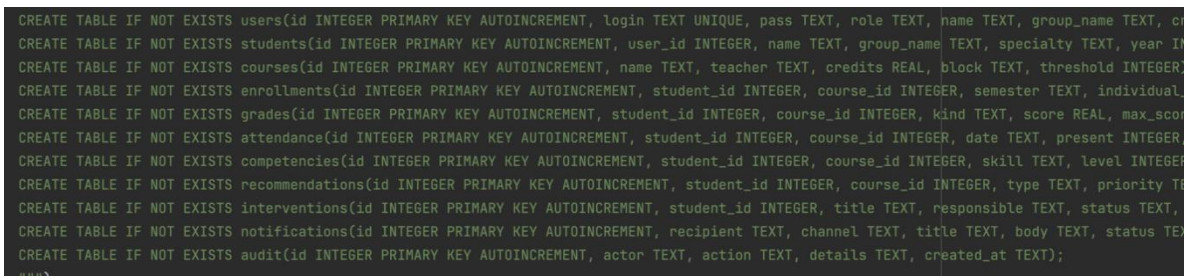


Рис. 3.1 – Фрагмент реалізації шару доступу до бази даних у програмному коді системи

Структура бази даних охоплює основні інформаційні об’єкти навчального процесу: користувачів, студентів, навчальні курси, записи про зарахування студентів на курси, результати оцінювання, відвідуваність, компетентності, рекомендації, педагогічні втручання, сповіщення та журнал аудиту. Такий склад таблиць дає змогу зберігати як первинні навчальні дані, так і результати їх подальшої аналітичної обробки.

На рисунку 3.2 показано фрагмент SQL-структури бази даних, де визначаються таблиці системи. Усі таблиці створюються командою `CREATE TABLE IF NOT EXISTS`, що дозволяє безпечно ініціалізувати базу даних під час запуску застосунку без втрати вже наявної інформації.

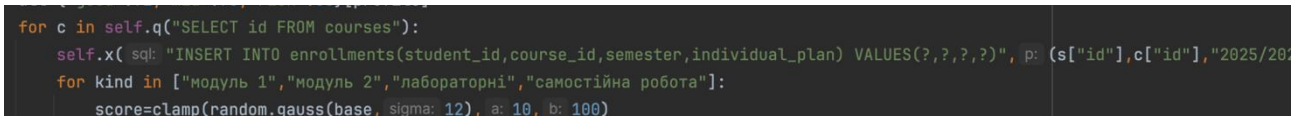


Рис. 3.2 – Фрагмент SQL-структури локальної бази даних системи

Основною таблицею є `users`, у якій зберігаються облікові записи користувачів системи. Вона містить логін, пароль, роль користувача, ім’я та назву академічної групи. Роль користувача визначає доступні функції інтерфейсу: студент переглядає власні показники та рекомендації, викладач аналізує результати групи, а адміністратор керує даними системи.

Таблиця `students` деталізує інформацію про студентів і пов’язується з таблицею `users` через поле `user_id`. Вона містить ПІБ студента, назву групи, спеціальність і рік навчання. Окрема таблиця студентів потрібна для того, щоб розділити облікові дані користувача та навчальні характеристики студента.

Таблиця `courses` зберігає інформацію про навчальні дисципліни: назву курсу, викладача, кількість кредитів, блок дисципліни та порогове значення успішності. Пороговий показник використовується під час аналітичної обробки для визначення студентів, які мають ризик зниження навчальних результатів.

Зв’язок між студентами та курсами реалізовано через таблицю `enrollments`. Вона зберігає ідентифікатор студента, ідентифікатор курсу, семестр та індивідуальний план навчання. Така структура дозволяє

реалізувати зв'язок «багато-до-багатьох», оскільки один студент може вивчати кілька дисциплін, а один курс може бути призначений багатьом студентам.

Результати оцінювання зберігаються в таблиці `grades`. У ній фіксуються студент, курс, тип контрольного заходу, отриманий бал, максимальний бал і дата внесення результату. Саме ці дані є основою для розрахунку середнього відсотка успішності студента.

Дані про відвідуваність і активність студента містяться в таблиці `attendance`. Вона зберігає дату заняття, ознаку присутності, тривалість запізнення та показник активності. На основі цих записів система визначає середню відвідуваність, рівень залученості студента та вплив активності на загальний коефіцієнт успішності.

Таблиця `competencies` використовується для зберігання рівня сформованості окремих навчальних компетентностей. Для кожного студента й курсу фіксується назва навички та рівень її засвоєння. Ці дані доповнюють оцінки та відвідуваність, оскільки дозволяють враховувати не лише формальні бали, а й якість засвоєння окремих змістових компонентів.

На рисунку 3.3 наведено фрагмент SQL-запитів, які використовуються для розрахунку ключових показників успішності. Зокрема, система обчислює середній відсоток результатів оцінювання, показник присутності, активність, середнє запізнення та рівень компетентностей студента.

```
g=self.one( sql: "SELECT AVG(score/max_score*100) v FROM grades WHERE student_id=?", p: (sid,))["v"] o
a=self.one( sql: "SELECT AVG(present*100) v FROM attendance WHERE student_id=?", p: (sid,))["v"] or 0
act=self.one( sql: "SELECT AVG(activity) v FROM attendance WHERE student_id=?", p: (sid,))["v"] or 0
late=self.one( sql: "SELECT AVG(late_min) v FROM attendance WHERE student_id=?", p: (sid,))["v"] or 0
comp=self.one( sql: "SELECT AVG(level) v FROM competencies WHERE student_id=?", p: (sid,))["v"] or 0
```

Рис. 3.3 – Фрагмент SQL-запитів для обчислення показників успішності студента

Аналітична частина системи використовує агреговані дані з таблиць `grades`, `attendance` та `competencies`. На їх основі формується інтегральний показник успішності, який враховує навчальні результати, відвідуваність, активність і рівень компетентностей. Такий підхід дозволяє

оцінювати не лише підсумковий бал, а й динаміку навчальної поведінки студента.

Таблиці `recommendations` та `interventions` призначені для збереження результатів рекомендаційної та педагогічної роботи. У таблиці `recommendations` фіксуються тип рекомендації, пріоритет, текст рекомендації та дата її створення. У таблиці `interventions` зберігаються заплановані або виконані заходи впливу, відповідальні особи та статус виконання. Це дозволяє системі не лише виявляти проблемні ситуації, а й підтримувати процес їх подальшого супроводу.

Таблиця `notifications` використовується для збереження системних повідомлень, які надсилаються студентам, викладачам або адміністраторам. Журнал `audit` фіксує основні дії користувачів у системі, що підвищує контрольованість роботи застосунку та дозволяє відстежувати зміни в базі даних.

Фізична модель бази даних подана на рис. 3.4. Вона відображає основні таблиці системи, первинні та зовнішні ключі, а також зв’язки між сутностями. Центральними таблицями моделі є `users`, `students`, `courses`, `enrollments`, `grades`, `attendance`, `competencies`, `recommendations`, `interventions`, `notifications` та `audit`.

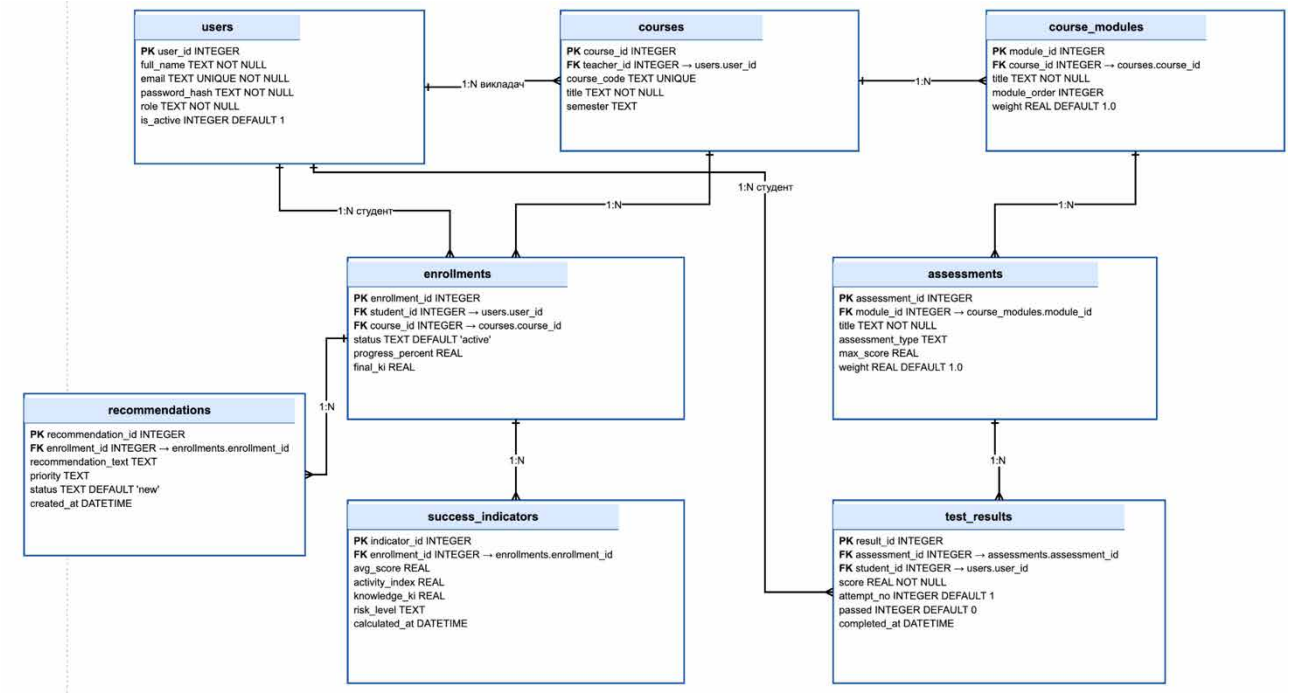


Рис. 3.4 – Фізична модель даних інформаційної системи моніторингу успішності студентів

Побудована база даних забезпечує логічну цілісність інформації та підтримує всі ключові функції системи: автентифікацію користувачів, зберігання даних студентів, облік навчальних курсів, фіксацію результатів оцінювання, аналіз відвідуваності, розрахунок компетентностей, формування рекомендацій і ведення журналу подій. Використання SQLite є обґрунтованим для розроблювальної системи, оскільки забезпечує простоту розгортання, достатню швидкодію та повну автономність роботи застосунку.

3.3 Представлення архітектури системи

Основою архітектури є клієнтський застосунок, реалізований засобами Python та PyQt6. У межах клієнтського середовища функціонують окремі програмні модулі, які відповідають за автентифікацію користувачів, моніторинг успішності, аналітичну обробку даних, формування рекомендацій і генерацію звітності. Взаємодія між компонентами відбувається через внутрішні механізми Python Runtime та локальну базу даних SQLite.

На рисунку 3.5 наведено діаграму архітектури системи, яка відображає основні програмні модулі, середовище виконання, локальне сховище даних та зовнішні сервіси інтеграції.

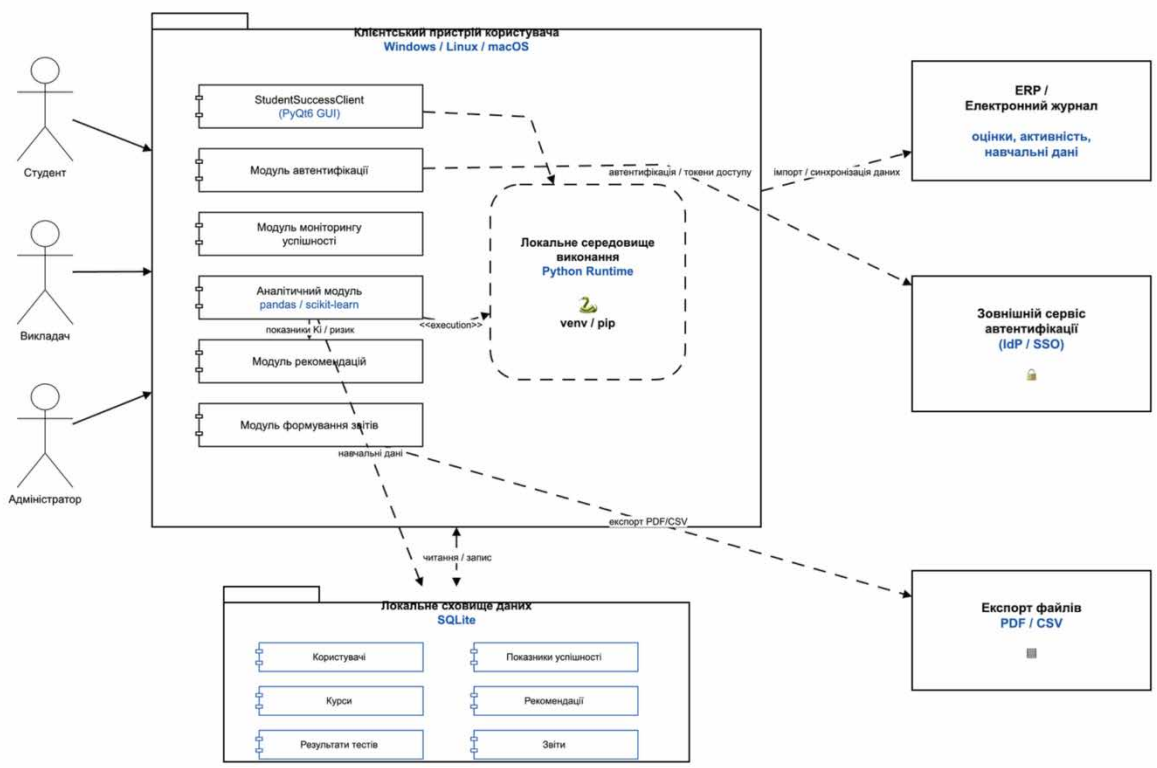


Рис. 3.5 – Архітектура інформаційної системи моніторингу успішності студентів

Центральним компонентом системи є модуль `StudentSuccessClient`, реалізований із використанням бібліотеки PyQt6. Він забезпечує побудову графічного інтерфейсу користувача, оброблення подій, роботу з формами авторизації, таблицями результатів, панелями аналітики та засобами відображення рекомендацій. Саме через цей модуль студент, викладач або адміністратор взаємодіє з інформаційною системою.

Модуль автентифікації відповідає за перевірку облікових даних користувача, визначення ролі та контроль доступу до функціональних можливостей системи. Після успішного входу застосунок формує відповідний інтерфейс залежно від типу користувача. Для забезпечення сумісності з корпоративними освітніми платформами передбачена можливість інтеграції із зовнішнім сервісом автентифікації IdP/SSO.

Модуль моніторингу успішності реалізує функції оброблення навчальних результатів студентів. Він отримує інформацію про оцінки, проходження тестів, відвідуваність та активність користувачів із локальної бази даних. Отримані дані передаються до аналітичного модуля для подальших розрахунків.

Аналітичний модуль побудований із використанням бібліотек pandas та scikit-learn. Його основним призначенням є розрахунок середнього бала, показника активності, коефіцієнта знань Ki та рівня ризику зниження успішності. На основі агрегованих показників модуль формує структуровані результати для подальшого використання у рекомендаційній підсистемі.

Модуль рекомендацій виконує генерацію персоналізованих навчальних рекомендацій залежно від рівня успішності студента, показників активності та ризику академічної неуспішності. Система може формувати рекомендації підтримувального, коригувального або поглибленого типу. Результати роботи модуля зберігаються у локальному сховищі та відображаються в інтерфейсі користувача.

Модуль формування звітів відповідає за створення підсумкових таблиць, графіків і аналітичних документів. Передбачено експорт результатів у формати PDF та CSV. Це дозволяє викладачам і адміністраторам формувати підсумкову документацію щодо результатів навчання академічних груп або окремих студентів.

Локальне середовище виконання базується на Python Runtime та механізмах `venv/pip`. Такий підхід забезпечує ізоляцію залежностей проєкту, стабільність роботи бібліотек і спрощення процесу розгортання системи на різних операційних системах. Для запуску застосунку використовуються окремі сценарії запуску для Windows, Linux та macOS.

Зберігання інформації реалізовано через локальне сховище SQLite. База даних містить таблиці користувачів, курсів, результатів тестування, показників успішності, рекомендацій та звітів. Використання SQLite забезпечує

автономність системи та дозволяє уникнути потреби у використанні окремого серверу баз даних.

У таблиці 3.2 наведено характеристику основних компонентів архітектури системи.

Таблиця 3.2

Характеристика компонентів архітектури системи

№	Компонент	Основне призначення	Використані технології
1	StudentSuccessClient	Графічний інтерфейс користувача	PyQt6
2	Модуль автентифікації	Перевірка облікових даних та контроль доступу	Python, SQLite
3	Модуль моніторингу успішності	Отримання та оброблення навчальних показників	Python
4	Аналітичний модуль	Розрахунок KPI, Ki та рівня ризику	pandas, NumPy, scikit-learn
5	Модуль рекомендацій	Формування персоналізованих рекомендацій	Python
6	Модуль звітності	Формування PDF/CSV-звітів і графіків	pandas, matplotlib
7	Локальне сховище даних	Зберігання навчальної інформації	SQLite
8	Сервіс автентифікації	Інтеграція із зовнішнім входом користувачів	IdP / SSO

Запропонована архітектура забезпечує чітке розділення функціональних компонентів системи та дозволяє реалізувати повний цикл роботи з навчальними даними - від автентифікації користувача та збору результатів до аналітичної обробки, формування рекомендацій і генерації звітності. Використання Python, PyQt6 та SQLite дозволяє створити автономний кросплатформний програмний продукт із достатнім рівнем продуктивності та можливістю подальшого функціонального масштабування.

3.4 Алгоритмізація програмної системи

Алгоритмізація інформаційної системи моніторингу успішності студентів спрямована на забезпечення послідовної обробки навчальних даних, автоматизації аналітичних розрахунків та формування персоналізованих рекомендацій для користувачів системи. Основна логіка функціонування реалізована засобами Python та поділена на окремі функціональні алгоритми, що

відповідають за автентифікацію користувача, аналіз навчальних показників і генерацію рекомендацій.

Ключовою особливістю алгоритмічної структури системи є модульний підхід, за якого кожен функціональний компонент виконує незалежну обробку власного набору даних. Така організація забезпечує гнучкість програмної архітектури, спрощує супровід програмного забезпечення та дозволяє масштабувати систему без суттєвого перепроєктування внутрішньої логіки.

Першим етапом роботи системи є автентифікація користувача. Після запуску застосунку система відображає форму входу, перевіряє коректність введених даних, виконує пошук користувача в локальній базі даних та здійснює перевірку хешу пароля. У разі успішної авторизації система визначає роль користувача й відкриває відповідний робочий інтерфейс.

На рисунку 3.6 наведено блок-схему алгоритму автентифікації користувача та відкриття робочого інтерфейсу системи.

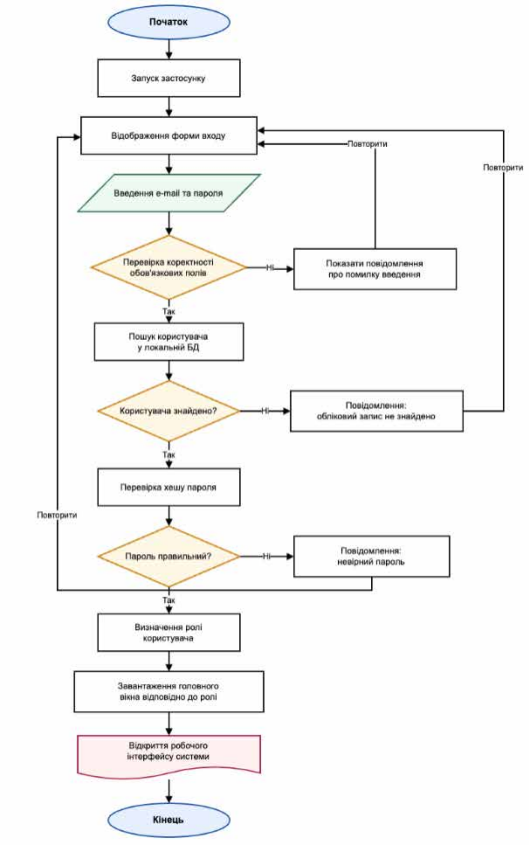


Рис. 3.6 – Алгоритм автентифікації користувача та відкриття робочого інтерфейсу системи

Після успішного входу користувач отримує доступ до модулів системи відповідно до власної ролі. Студент може переглядати власні результати, показники успішності та рекомендації, викладач – аналізувати результати академічної групи, а адміністратор – керувати користувачами, курсами та параметрами системи.

Основний процес аналітичної обробки реалізується в модулі моніторингу успішності. Система отримує навчальні дані з локальної бази SQLite, виконує перевірку їх цілісності та здійснює обчислення ключових показників ефективності навчання. До таких показників належать середній бал, рівень активності, коефіцієнт знань Ki та рівень ризику зниження успішності.

Для розрахунку середнього бала система використовує результати оцінювання з таблиці `grades`, а показник активності формується на основі записів відвідуваності та участі студента у навчальному процесі. Після цього виконується інтегральний розрахунок коефіцієнта знань Ki, який використовується як один із базових показників аналітичної оцінки.

На рисунку 3.7 наведено блок-схему алгоритму збору та аналізу показників успішності студента. Після завершення розрахунків система формує аналітичні таблиці, KPI та графічні візуалізації результатів навчання. Отримані значення зберігаються у таблиці `success_indicators`, що дозволяє накопичувати статистику успішності та використовувати її для подальшого аналізу.

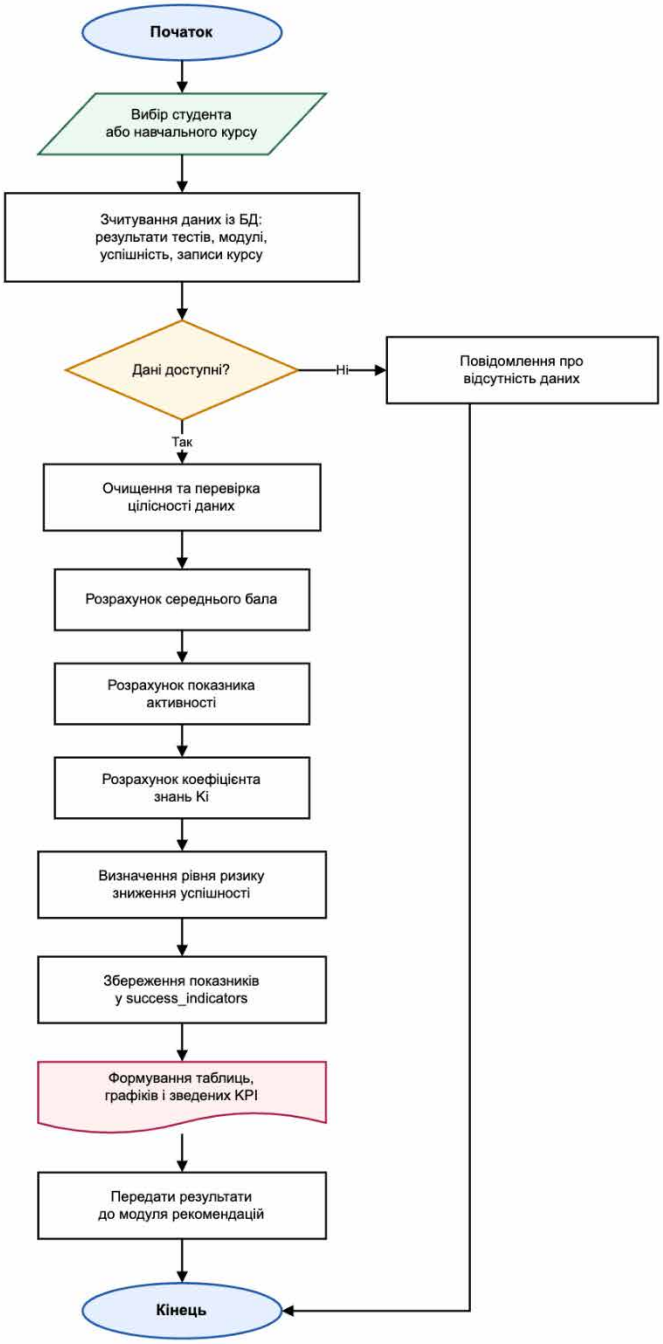


Рис. 3.7 – Алгоритм збору та аналізу показників успішності студента

На основі результатів аналітичного модуля працює модуль формування персоналізованих рекомендацій. Його завданням є автоматичне визначення проблемних зон студента та генерація відповідних рекомендацій залежно від рівня ризику академічної неуспішності.

На початковому етапі модуль отримує інтегральні аналітичні показники студента та параметри моделі оцінювання. Далі система перевіряє, чи перебувають показники в межах допустимих значень. Якщо результати

відповідають нормі, формується підтримувальна рекомендація. Якщо ж система виявляє зниження успішності або активності, виконується аналіз проблемних зон: результатів тестування, навчальних модулів та поведінкових характеристик.

Залежно від визначеного рівня ризику система може формувати рекомендації трьох типів:

– підтримувальні;
– рекомендації поглибленого опрацювання;
– термінові коригувальні рекомендації.

Після генерації рекомендацій система автоматично визначає їх пріоритет, присвоює статус та зберігає результати у таблицю `recommendations`. Далі сформовані рекомендації відображаються в інтерфейсі студента.

На рисунку 3.8 наведено блок-схему алгоритму формування персоналізованих навчальних рекомендацій.

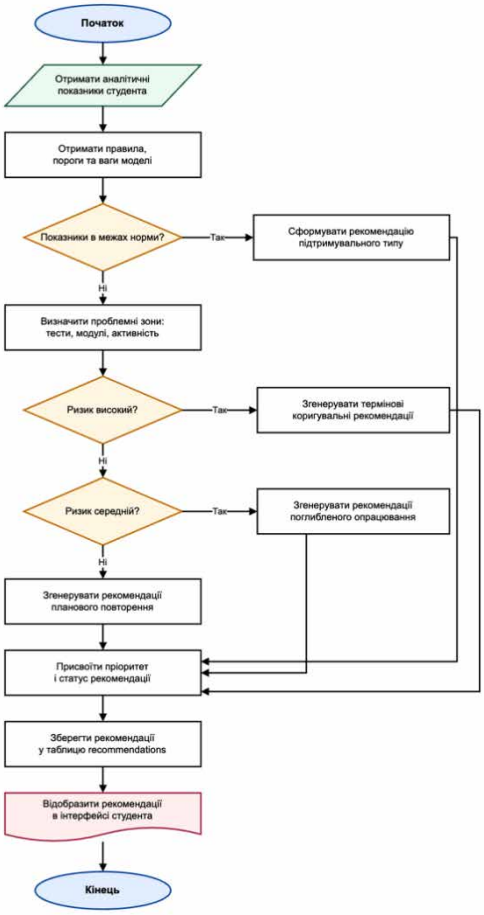


Рис. 3.8 – Алгоритм формування персоналізованих навчальних рекомендацій

У таблиці 3.3 наведено характеристику основних алгоритмів програмної системи.

Таблиця 3.3

Основні алгоритми програмної системи

№	Алгоритм	Основне призначення	Результат виконання
1	Автентифікація користувача	Перевірка облікових даних та визначення ролі	Відкриття відповідного інтерфейсу системи
2	Аналіз успішності	Розрахунок KPI, середнього бала, активності та Ki	Формування аналітичних показників
3	Формування рекомендацій	Визначення ризику та генерація рекомендацій	Персоналізовані навчальні рекомендації
4	Формування звітів	Генерація таблиць, графіків та експорт даних	PDF/CSV-звіти
5	Обробка даних БД	Зчитування та збереження інформації	Оновлення локальної бази SQLite

Розроблена алгоритмічна структура забезпечує послідовне функціонування всіх компонентів програмної системи та дозволяє реалізувати повний цикл роботи з навчальними даними – від автентифікації користувача та збору результатів до аналітичної обробки, оцінювання рівня ризику та формування персоналізованих рекомендацій. Використання модульного підходу та локальної архітектури на основі Python і SQLite забезпечує достатню продуктивність системи, простоту супроводу та можливість подальшого функціонального розширення.

3.5 Висновок до третього розділу

У третьому розділі виконано проєктування та програмну реалізацію інформаційної системи моніторингу успішності студентів із формуванням персоналізованих навчальних рекомендацій. Обґрунтовано вибір технологічного стеку, який включає Python 3.x, PyQt6, SQLite, pandas, scikit-learn та інші програмні засоби, необхідні для створення настільної аналітичної системи.

У межах розділу розроблено структуру бази даних, що забезпечує зберігання інформації про користувачів, студентів, дисципліни, оцінки,

показники успішності, рекомендації, втручання та журнал аудиту. Побудована модель даних дозволяє реалізувати цілісну систему оброблення навчальної інформації та підтримує взаємозв’язки між основними сутностями програмного забезпечення.

Представлено архітектуру системи, яка базується на модульному підході та включає клієнтський застосунок, модуль автентифікації, аналітичний модуль, рекомендаційну підсистему, модуль звітності та локальне сховище SQLite. Запропонована структура забезпечує автономність роботи системи, простоту розгортання та можливість подальшого функціонального розширення.

Окрему увагу приділено алгоритмізації програмної системи. Розроблено алгоритми авторизації користувачів, аналізу показників успішності та генерації персоналізованих рекомендацій. Побудовані блок-схеми відображають логіку оброблення навчальних даних, визначення рівня академічного ризику та формування рекомендацій відповідно до результатів аналітичної оцінки.

У результаті виконання третього розділу сформовано повноцінну програмну основу інформаційної системи моніторингу успішності студентів, яка забезпечує збирання, оброблення, аналіз і візуалізацію навчальних даних, а також підтримує формування аналітичних висновків і рекомендацій для користувачів системи. Отримані результати є базою для подальшого тестування, оцінювання ефективності та практичного використання розробленого програмного забезпечення.

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ АНАЛІТИЧНОЇ СИСТЕМИ

4.1 План тестування програмних модулів та методика оцінювання результатів

План тестування інформаційної системи моніторингу успішності студентів сформовано з урахуванням модульної архітектури програмного забезпечення, що включає підсистему автентифікації, модуль моніторингу успішності, аналітичний модуль, рекомендаційну підсистему, локальне сховище даних SQLite та модуль формування звітності. Основною метою тестування є перевірка коректності роботи алгоритмів аналізу навчальних показників, стабільності взаємодії між програмними компонентами, правильності збереження даних та надійності функціонування користувацького інтерфейсу.

Тестування системи орієнтоване на оцінювання функціональних, аналітичних, продуктивних і стійкісних характеристик програмного забезпечення. Особлива увага приділяється правильності розрахунку коефіцієнта знань Ki, аналізу показників активності студентів, формуванню персоналізованих рекомендацій та стабільності роботи локальної бази даних.

План тестування структуровано за окремими програмними модулями, як наведено у табл. 4.1. Для кожного компонента визначено тестові дії, контрольні критерії та очікувані результати функціонування.

Таблиця 4.1 – План тестування програмних модулів інформаційної системи моніторингу успішності студентів

№	Модуль / компонент	Тестова дія	Контрольний критерій	Очікуваний результат
1	AuthModule	Авторизація користувача та перевірка ролей	Відсутність несанкціонованого доступу	Коректна автентифікація та відкриття відповідного інтерфейсу
2	StudentSuccessClient	Відображення GUI та взаємодія з елементами інтерфейсу	Час реакції ≤ 100 мс	Стабільна робота графічного інтерфейсу

Продовження таблиці 4.1

3	MonitoringModule	Зчитування результатів навчання з БД	Повнота та цілісність даних	Коректне завантаження показників студентів
4	AnalyticsEngine	Розрахунок середнього бала, Ki та рівня ризику	Відхилення розрахунків ≤ 1%	Правильне формування KPI успішності
5	RecommendationModule	Генерація персоналізованих рекомендацій	Відповідність типу рекомендації рівню ризику	Коректне формування рекомендацій
6	SQLite Repository	CRUD-операції над таблицями системи	Час виконання ≤ 10 мс	Стабільне збереження та читання даних
7	AttendanceProcessor	Оброблення показників активності та відвідуваності	Коректність обчислення activity index	Правильне визначення рівня активності
8	ReportEngine	Формування PDF/CSV-звітів	Час генерації ≤ 1 с	Звіти без помилок і втрати даних
9	NotificationService	Формування системних повідомлень	Час відображення ≤ 50 мс	Своєчасне відображення повідомлень
10	AuditModule	Реєстрація дій користувачів	Повнота журналювання подій	Коректне ведення журналу аудиту

Проведення тестування передбачає використання як модульних перевірок (`unit testing`), так і інтеграційних сценаріїв, що моделюють повний цикл роботи програмної системи.

Особливу увагу приділено тестуванню сценаріїв із помилковими або неповними даними. До деградаційних сценаріїв належать:

- відсутність результатів тестування;
- пошкоджені записи бази даних;
- порожні поля оцінювання;
- дублювання записів студентів;
- неправильні облікові дані користувача;
- перевищення допустимого часу відповіді інтерфейсу;
- порушення цілісності таблиць SQLite.

Методика оцінювання результатів базується на порівнянні фактичних показників роботи системи з нормативними значеннями, визначеними у функціональних і нефункціональних вимогах. Оцінювання здійснюється за допомогою журналів подій, SQL-вибірок, аналітичних таблиць та внутрішніх статистичних показників системи.

Основними метриками оцінювання є:

– середній час відповіді інтерфейсу;
– точність розрахунку коефіцієнта знань Ki;
– правильність визначення рівня ризику;
– коректність формування рекомендацій;
– час виконання SQL-запитів;
– швидкість формування звітів;
– стабільність роботи локальної бази даних;
– повнота журналювання подій користувачів.

Позитивним результатом тестування вважається виконання всіх контрольних критеріїв у межах допустимих значень. Це підтверджує коректність роботи алгоритмів аналітичної системи, стабільність взаємодії між програмними модулями та готовність програмного забезпечення до практичного використання в умовах освітнього процесу.

Сформований план тестування дозволяє комплексно оцінити якість реалізованих програмних модулів, виявити потенційні проблемні ділянки системи та підтвердити відповідність розробленого програмного забезпечення функціональним, аналітичним і продуктивним вимогам, визначеним на етапі проєктування інформаційної системи моніторингу успішності студентів.

4.2 Тестування системи

Тестування інформаційної системи моніторингу успішності студентів виконувалося з метою перевірки коректності роботи основних програмних модулів, стабільності графічного інтерфейсу, правильності оброблення навчальних даних та надійності збереження результатів у локальній базі SQLite.

Перевірка охоплювала авторизацію користувачів, роботу головної аналітичної панелі, модулі студентів, освітніх даних, ML-рекомендацій, аналітики, кураторських втручань, звітності та аудиту.

На рисунку 4.1 наведено вікно авторизації системи EduNavigator KPI. Під час тестування цього модуля перевірялося введення логіна і пароля, оброблення помилкових даних, визначення ролі користувача та відкриття відповідного інтерфейсу. У результаті встановлено, що система коректно виконує автентифікацію, не допускає доступу без правильних облікових даних і стабільно переходить до головного вікна після успішного входу.

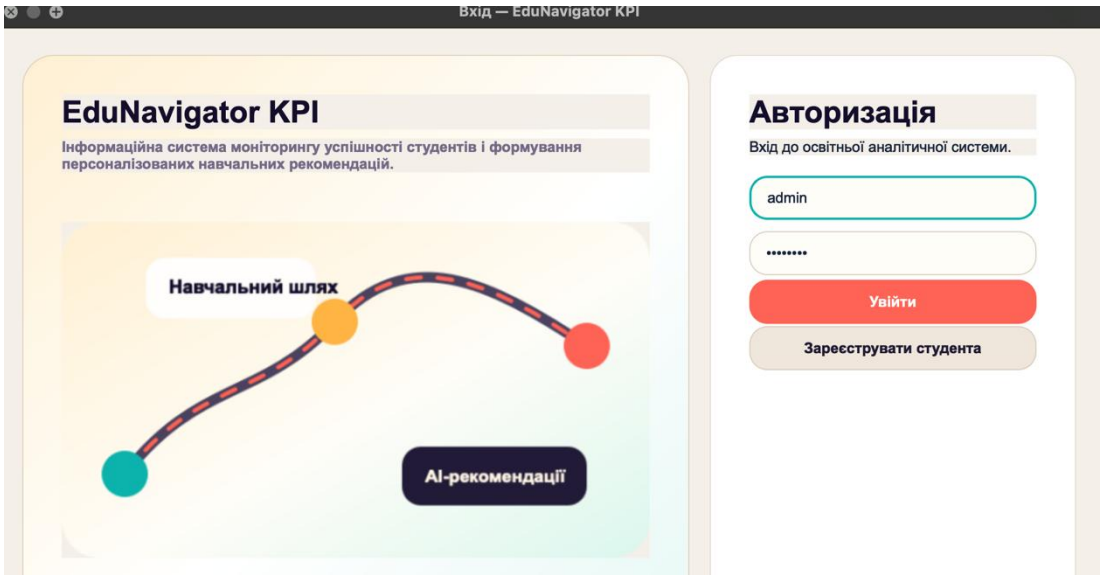


Рис. 4.1 – Вікно авторизації інформаційної системи EduNavigator KPI

Після входу до системи було протестовано головну панель навігатора успішності, подану на рис. 4.2. Цей інтерфейс відображає зведені показники кількості студентів, середнього KPI, інтегрального індексу успішності, відвідуваності, кількості ML-дій та кураторських втручань. Також у вікні подано інтегральний ризик і короткий аналітичний висновок. Перевірка показала, що система правильно завантажує дані з бази, оновлює показники після перерахунку та формує узагальнений висновок щодо поточного стану навчальної успішності.

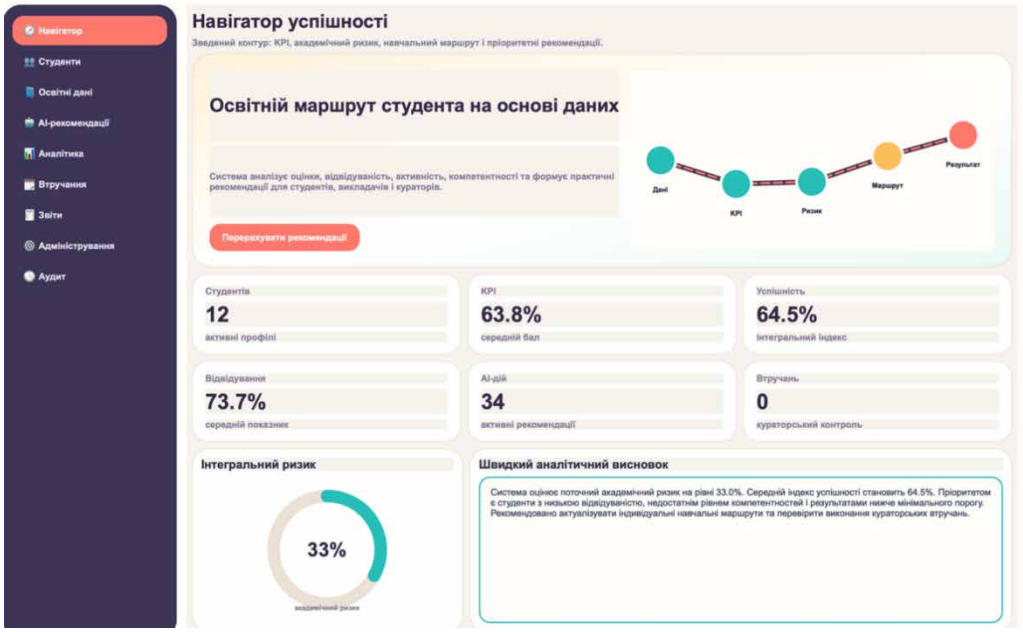
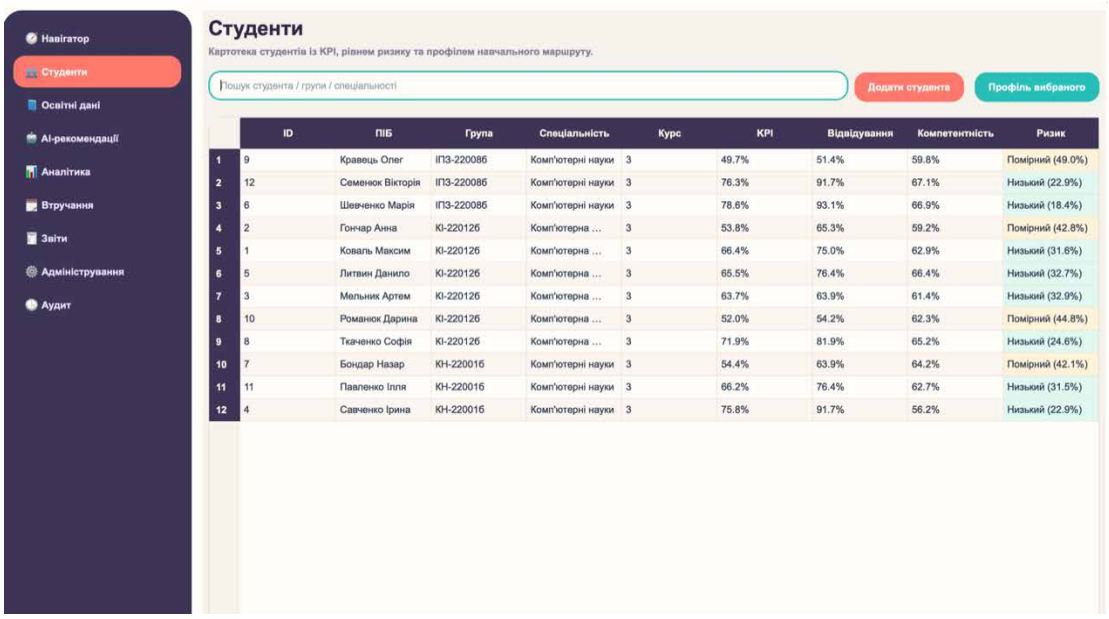


Рис. 4.2 – Головна панель моніторингу успішності студентів

На рисунку 4.3 наведено модуль роботи зі студентами. У межах тестування перевірялося відображення картотеки студентів, пошук за ПІБ, групою та спеціальністю, а також коректність виведення KPI, відвідуваності, компетентностей і рівня ризику. Результати перевірки підтвердили, що таблиця студентів формується без втрати записів, а ризикові показники відповідають даним, збереженим у базі.

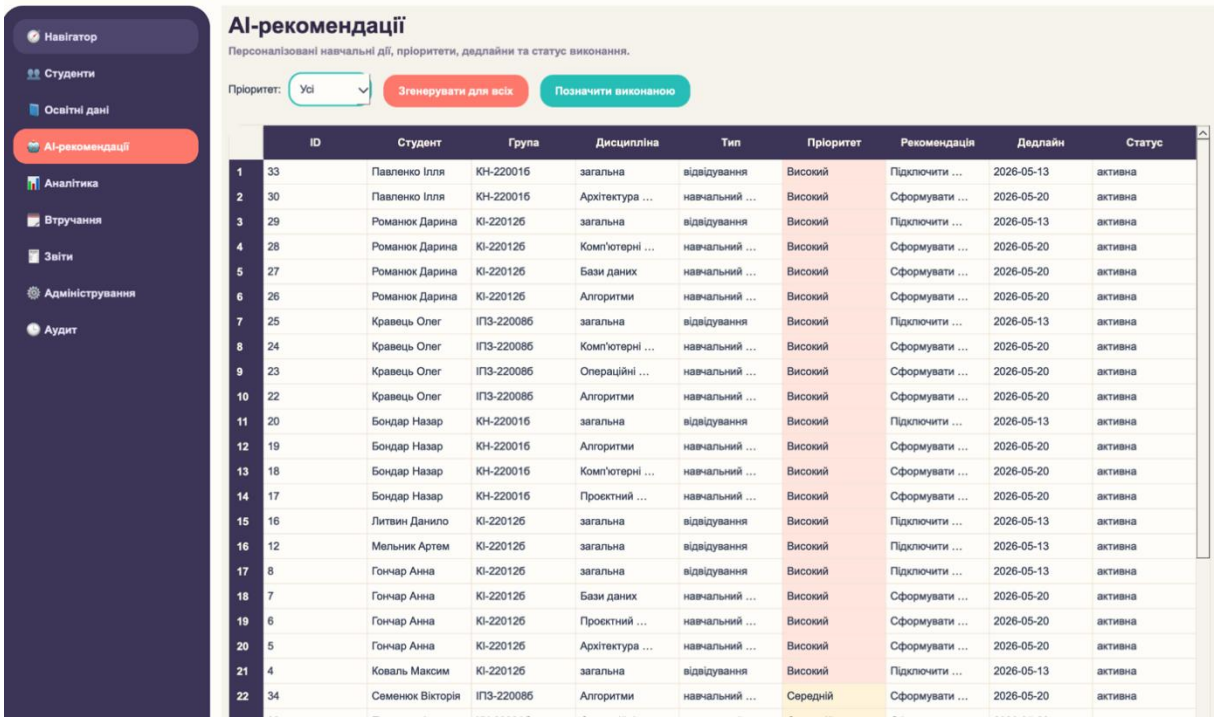


	ID	ПІБ	Група	Спеціальність	Курс	KPI	Відвідування	Компетентність	Ризик
1	9	Кравець Олег	ІПЗ-220086	Комп'ютерні науки	3	49.7%	51.4%	59.8%	Помірний (49.0%)
2	12	Семенок Вікторія	ІПЗ-220086	Комп'ютерні науки	3	76.3%	91.7%	67.1%	Низький (22.9%)
3	6	Шевченко Марія	ІПЗ-220086	Комп'ютерні науки	3	78.6%	93.1%	66.9%	Низький (18.4%)
4	2	Гончар Анна	КІ-220126	Комп'ютерна ...	3	53.8%	65.3%	59.2%	Помірний (42.8%)
5	1	Коваль Максим	КІ-220126	Комп'ютерна ...	3	66.4%	75.0%	62.9%	Низький (31.6%)
6	5	Литвин Данило	КІ-220126	Комп'ютерна ...	3	65.5%	76.4%	66.4%	Низький (32.7%)
7	3	Мельник Артем	КІ-220126	Комп'ютерна ...	3	63.7%	63.9%	61.4%	Низький (32.9%)
8	10	Романюк Дарина	КІ-220126	Комп'ютерна ...	3	52.0%	54.2%	62.3%	Помірний (44.8%)
9	8	Ткаченко Софія	КІ-220126	Комп'ютерна ...	3	71.9%	81.9%	65.2%	Низький (24.6%)
10	7	Бондар Назар	КН-220016	Комп'ютерні науки	3	54.4%	63.9%	64.2%	Помірний (42.1%)
11	11	Павленко Ілля	КН-220016	Комп'ютерні науки	3	66.2%	76.4%	62.7%	Низький (31.5%)
12	4	Савченко Ірина	КН-220016	Комп'ютерні науки	3	75.8%	91.7%	56.2%	Низький (22.9%)

Рис. 4.3 – Модуль перегляду студентів та їхніх показників успішності

Тестування підсистеми ML-рекомендацій виконувалося для перевірки правильності формування персоналізованих навчальних дій. Інтерфейс цього

модуля наведено на рис. 4.4. Система аналізує показники студента, визначає тип рекомендації, пріоритет, дисципліну, дедлайн і поточний статус виконання. У процесі перевірки встановлено, що рекомендації генеруються відповідно до рівня академічного ризику та коректно зберігаються в базі даних.



	ID	Студент	Група	Дисципліна	Тип	Пріоритет	Рекомендація	Дедлайн	Статус
1	33	Павленко Ілля	КН-220016	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
2	30	Павленко Ілля	КН-220016	Архітектура ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
3	29	Романюк Дарина	КІ-220126	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
4	28	Романюк Дарина	КІ-220126	Комп'ютерні ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
5	27	Романюк Дарина	КІ-220126	Бази даних	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
6	26	Романюк Дарина	КІ-220126	Алгоритми	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
7	25	Кравець Олег	ІПЗ-220086	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
8	24	Кравець Олег	ІПЗ-220086	Комп'ютерні ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
9	23	Кравець Олег	ІПЗ-220086	Операційні ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
10	22	Кравець Олег	ІПЗ-220086	Алгоритми	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
11	20	Бондар Назар	КН-220016	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
12	19	Бондар Назар	КН-220016	Алгоритми	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
13	18	Бондар Назар	КН-220016	Комп'ютерні ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
14	17	Бондар Назар	КН-220016	Проектний ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
15	16	Литвин Данило	КІ-220126	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
16	12	Мельник Артем	КІ-220126	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
17	8	Гончар Анна	КІ-220126	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
18	7	Гончар Анна	КІ-220126	Бази даних	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
19	6	Гончар Анна	КІ-220126	Проектний ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
20	5	Гончар Анна	КІ-220126	Архітектура ...	навчальний ...	Високий	Сформувати ...	2026-05-20	активна
21	4	Коваль Максим	КІ-220126	загальна	відвідування	Високий	Підключити ...	2026-05-13	активна
22	34	Семенюк Вікторія	ІПЗ-220086	Алгоритми	навчальний ...	Середній	Сформувати ...	2026-05-20	активна

Рис. 4.4 – Модуль формування персоналізованих ML-рекомендацій

На рисунку 4.5 подано модуль освітніх даних, який використовується для роботи з дисциплінами, оцінками, відвідуваністю та компетентностями. Під час тестування перевірялося додавання навчальних дисциплін, внесення оцінок, оновлення записів відвідуваності та збереження рівнів сформованості компетентностей. Перевірка підтвердила стабільність CRUD-операцій і правильність відображення навчальних даних у відповідних вкладках.

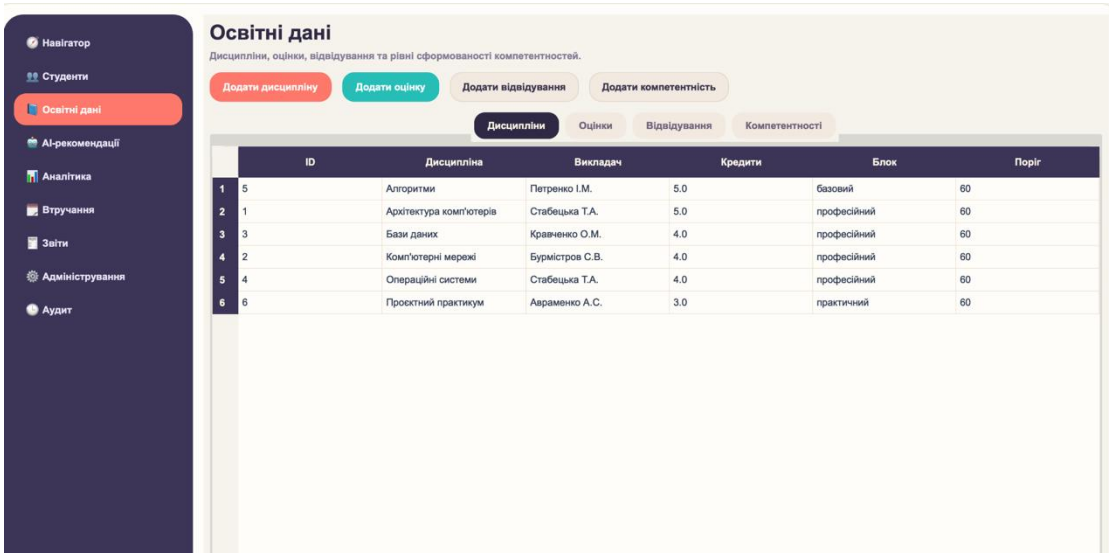


Рис. 4.5 – Модуль управління освітніми даними

Аналітичний модуль системи наведено на рис. 4.6. Його тестування проводилося для перевірки побудови графіків ризику за дисциплінами, ранжування студентів із підвищеним академічним ризиком та формування текстового аналітичного висновку. Отримані результати засвідчили, що система коректно обробляє статистичні вибірки, правильно визначає проблемні дисципліни та наочно відображає групи студентів із найвищими ризиковими показниками.

Рис. 4.6 – Аналітичний модуль оцінювання академічних ризиків

На рисунку 4.7 представлено модуль кураторських втручань, який забезпечує створення індивідуальних заходів підтримки студентів. Під час тестування перевірялося створення запису втручання, зазначення відповідального, вибір дати, збереження інформації та подальше відображення запису у таблиці. Система коректно зберігає такі дані та дає змогу фіксувати виконання запланованих дій.

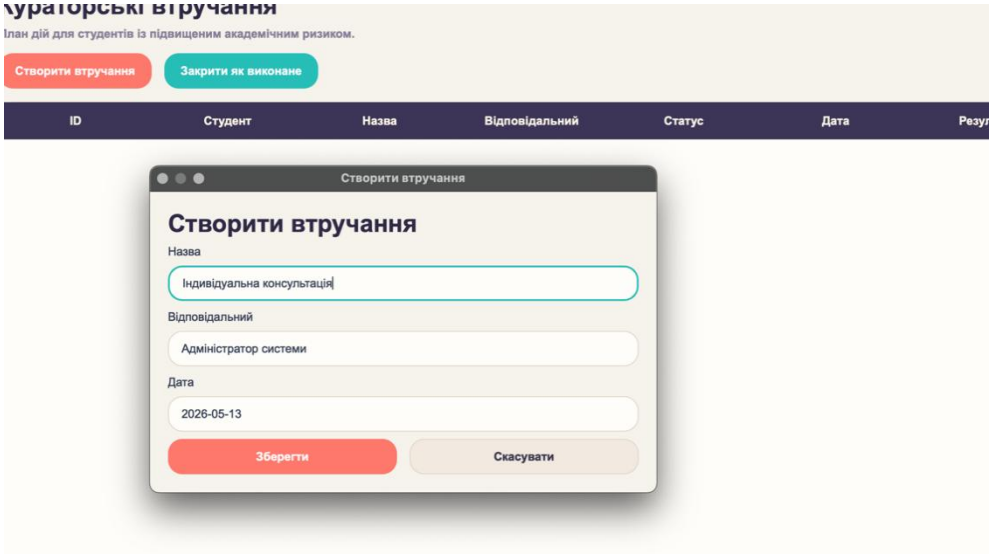


Рис. 4.7 – Модуль створення кураторського втручання

Модуль звітності подано на рис. 4.8. Його тестування передбачало формування зведеного академічного звіту та експорт результатів у HTML і CSV. Перевірка показала, що система правильно формує структуру звітного документа, не втрачає аналітичні дані та забезпечує збереження результатів для подальшого використання викладачем або адміністратором.

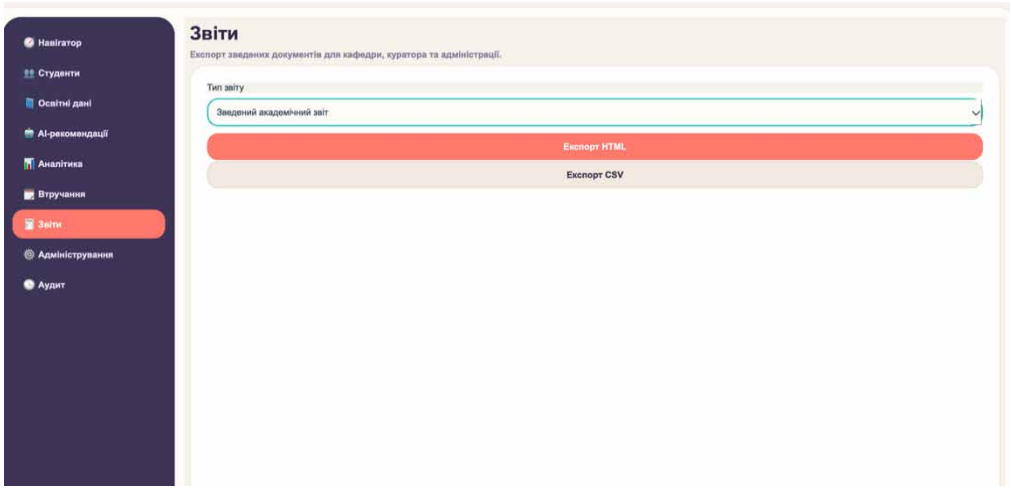


Рис. 4.8 – Модуль формування та експорту звітів

На рисунку 4.9 наведено журнал аудиту системи. У межах тестування перевірялося збереження подій авторизації, генерації рекомендацій, створення демонстраційної бази та інших системних операцій. Результати підтвердили, що аудит фіксує користувача, дію, деталі операції та часову мітку, що підвищує контрольованість роботи системи.

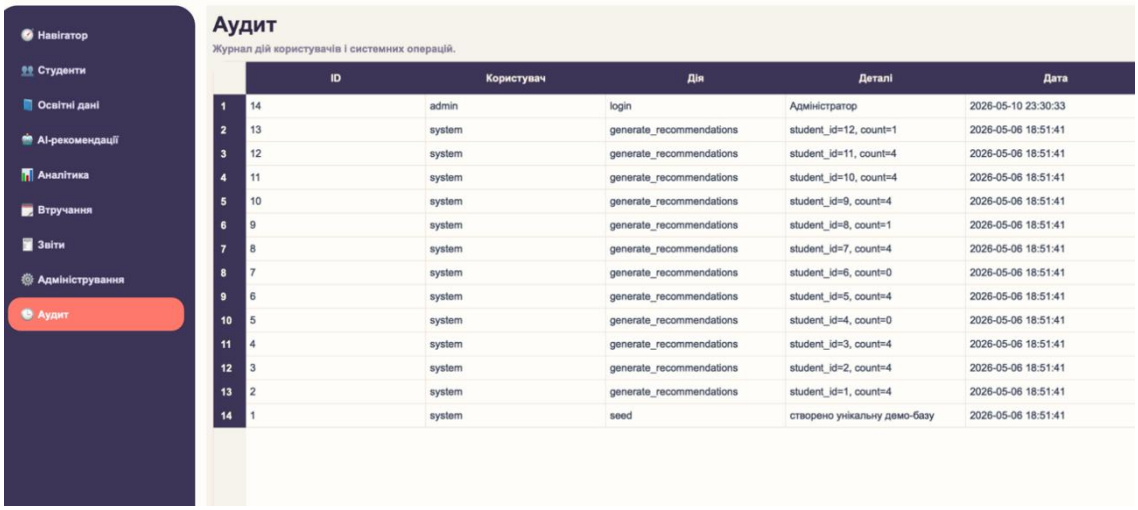


Рис. 4.9 – Журнал аудиту дій користувачів і системних операцій

Узагальнені результати перевірки основних модулів наведено в таблиці 4.2.

Таблиця 4.2

Результати тестування основних модулів системи

№	Об’єкт тестування	Перевірена функція	Результат
1	Авторизація	Вхід користувача та перевірка ролі	Виконано коректно
2	Головна панель	Відображення KPI та ризику	Виконано коректно
3	Модуль студентів	Пошук і перегляд показників студентів	Виконано коректно
4	Освітні дані	Робота з дисциплінами, оцінками й відвідуваністю	Виконано коректно
5	AI-рекомендації	Генерація персоналізованих рекомендацій	Виконано коректно
6	Аналітика	Побудова графіків і визначення ризиків	Виконано коректно
7	Кураторські втручання	Створення та збереження плану дій	Виконано коректно
8	Звіти	Експорт HTML і CSV	Виконано коректно
9	Аудит	Журналювання системних подій	Виконано коректно

Проведене тестування підтвердило, що інформаційна система стабільно виконує основні функції, передбачені технічним завданням. Програмні модулі коректно взаємодіють із локальною базою даних, інтерфейс забезпечує доступ до ключових функцій, а аналітичний блок правильно формує показники ризику, рекомендації та звітні результати. Отже, система може бути використана для практичного моніторингу успішності студентів і підтримки прийняття рішень у навчальному процесі.

4.3 Розгортання системи, склад інсталяційного пакету

Розгортання інформаційної системи моніторингу успішності студентів передбачає встановлення програмного середовища на робочу станцію користувача та підготовку локальної бази даних для зберігання навчальних показників, рекомендацій, звітів і службових записів. Оскільки система реалізована як настільний застосунок на Python, її експлуатація не потребує окремого серверу баз даних або складної мережевої інфраструктури. Основні програмні компоненти виконуються безпосередньо на клієнтському пристрої користувача в межах операційного середовища Windows, Linux або macOS.

На рисунку 4.10 подано діаграму розгортання системи, яка відображає склад програмного середовища, внутрішні артефакти застосунку, локальну базу даних SQLite, каталог експорту звітів та зовнішні вузли інтеграції.

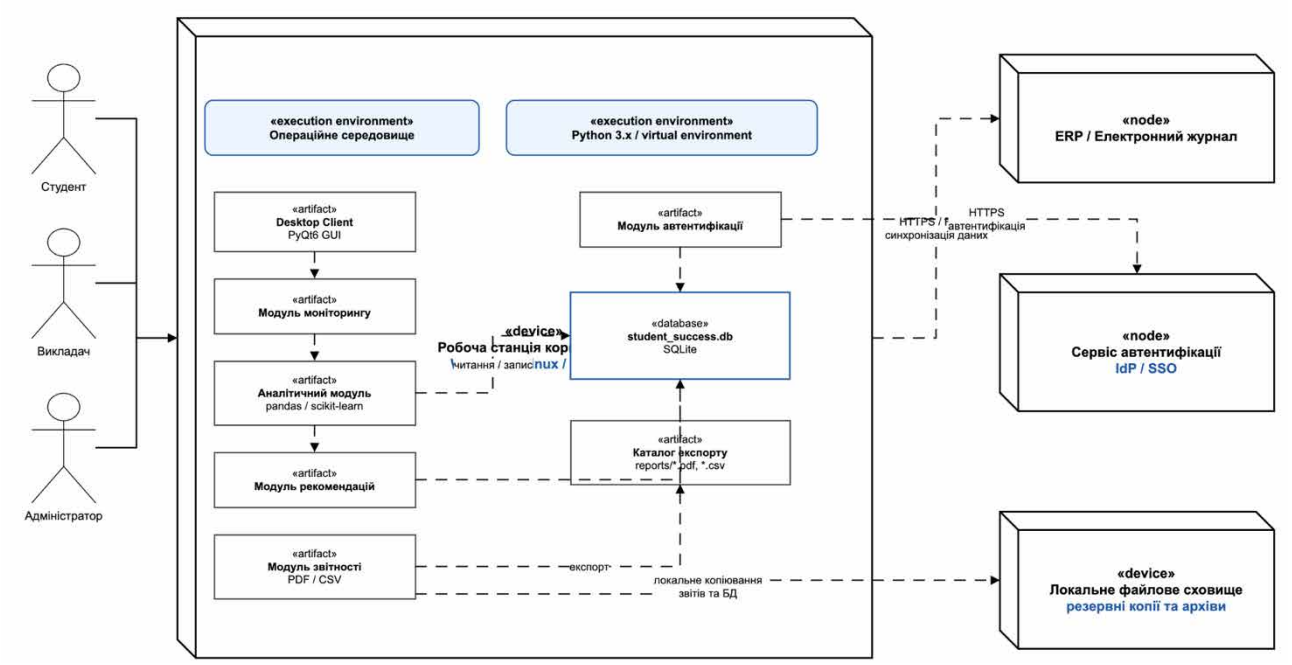


Рис. 4.10 – Діаграма розгортання інформаційної системи моніторингу успішності студентів

Відповідно до наведеної діаграми, центральним вузлом розгортання є робоча станція користувача. На ній розміщується операційне середовище, віртуальне середовище Python 3.x, графічний клієнт PyQt6, модуль автентифікації, модуль моніторингу, аналітичний модуль, модуль рекомендацій і модуль звітності. Усі програмні артефакти працюють у межах одного локального застосунку, що спрощує встановлення, супровід та оновлення системи.

Графічний клієнт Desktop Client забезпечує взаємодію користувача із системою. Через нього студент переглядає власні показники успішності та персоналізовані рекомендації, викладач аналізує результати групи, а адміністратор виконує керування даними, аудитом і звітністю. Інтерфейс побудовано засобами PyQt6, що дозволяє забезпечити однакову логіку роботи застосунку на різних операційних системах.

Модуль автентифікації відповідає за перевірку облікових даних, визначення ролі користувача та надання доступу до відповідних функцій системи. У базовому режимі автентифікація виконується через локальну базу

SQLite. У розширеному варіанті передбачена можливість взаємодії із зовнішнім сервісом автентифікації IdP/SSO через захищене HTTPS-з'єднання.

Модуль моніторингу забезпечує зчитування навчальних даних із бази, формування таблиць студентів, дисциплін, оцінок, відвідуваності та компетентностей. Аналітичний модуль виконує обчислення інтегральних показників, середнього KPI, рівня академічного ризику та інших параметрів, що використовуються для оцінювання успішності студентів. Модуль рекомендацій на основі отриманих аналітичних результатів формує персоналізовані навчальні дії для студентів із різним рівнем ризику.

Локальна база даних student_success.db створюється та оновлюється автоматично під час роботи застосунку. Вона містить записи користувачів, студентів, дисциплін, оцінок, відвідуваності, компетентностей, рекомендацій, кураторських втручань, сповіщень і журналу аудиту. Використання SQLite забезпечує автономну роботу системи та усуває потребу у встановленні окремої СУБД.

Модуль звітності формує підсумкові документи у форматах PDF, CSV або HTML. Звіти зберігаються у локальному каталозі експорту, що може використовуватися для передавання результатів викладачам, кураторам або адміністрації. Для підвищення надійності передбачено локальне копіювання бази даних і сформованих звітів до файлового сховища резервних копій.

Інсталяційний пакет системи має містити всі файли, необхідні для запуску застосунку, ініціалізації бази даних, встановлення залежностей та експорту результатів. Його склад наведено в таблиці 4.3.

Таблиця 4.3

Склад інсталяційного пакету інформаційної системи

№	Елемент пакету	Призначення
1	main.py	Головний файл запуску програмного застосунку
2	edu_navigator_kpi.db	Локальна база даних SQLite
3	requirements.txt	Перелік бібліотек Python, необхідних для роботи системи
4	style.qss	Файл оформлення графічного інтерфейсу PyQt6
5	assets	Каталог графічних ресурсів і службових зображень

Продовження таблиці 4.3

6	student_map.svg	Графічний ресурс для відображення навчального маршруту
7	start_windows.bat	Сценарій запуску застосунку в середовищі Windows
8	start_macos.command	Сценарій запуску застосунку в середовищі macOS
9	start_macos_linux.sh	Сценарій запуску застосунку в середовищі macOS/Linux
10	README.md	Коротка інструкція з розгортання та використання системи
11	reports	Каталог для збереження сформованих звітів
12	backups	Каталог для резервних копій бази даних і звітних файлів

Процес розгортання системи складається з копіювання інсталяційного пакету на робочу станцію, створення або активації віртуального середовища Python, встановлення залежностей із файлу requirements.txt та запуску головного файлу main.py або відповідного стартового сценарію для операційної системи. Після першого запуску система перевіряє наявність локальної бази даних, створює необхідні таблиці, виконує початкове наповнення демонстраційними даними та відкриває вікно авторизації користувача.

Для Windows запуск може виконуватися через start_windows.bat, для macOS через start_macos.command, а для Linux або macOS через start_macos_linux.sh. Такий підхід спрощує використання системи користувачами без глибоких технічних знань і забезпечує однакову логіку розгортання на різних платформах.

Зовнішня взаємодія системи є допоміжною та не є обов’язковою для базового режиму експлуатації. За потреби система може синхронізувати навчальні дані з ERP або електронним журналом, а також використовувати зовнішній сервіс автентифікації. У разі відсутності мережевого підключення основні функції системи залишаються доступними завдяки локальному зберіганню даних.

Отже, запропонована схема розгортання забезпечує просту інсталяцію, автономну роботу, мінімальні вимоги до інфраструктури та можливість подальшої інтеграції із зовнішніми освітніми сервісами. Склад інсталяційного пакету є достатнім для практичного запуску системи, тестування її

функціональних модулів, збереження навчальних даних і формування аналітичної звітності.

4.4 Висновок до четвертого розділу

У четвертому розділі виконано комплексне тестування інформаційної системи моніторингу успішності студентів та проведено оцінювання ефективності її програмних модулів. Сформовано план тестування, який охоплює підсистему авторизації, модулі аналітики, рекомендацій, звітності, аудиту, управління освітніми даними та механізми кураторських втручань.

У процесі тестування перевірено коректність функціонування графічного інтерфейсу, стабільність взаємодії між програмними компонентами, правильність оброблення навчальних даних і надійність локального збереження інформації в базі SQLite. Проведена перевірка підтвердила правильність реалізації механізмів автентифікації, формування KPI-показників, визначення рівня академічного ризику та генерації персоналізованих рекомендацій.

Виконано тестування аналітичного модуля, що забезпечує побудову графічних візуалізацій, аналіз проблемних дисциплін та визначення студентів групи ризику. Перевірено роботу підсистеми звітності, яка формує експортні документи у форматах HTML і CSV, а також модуль аудиту, що забезпечує журналювання системних подій і дій користувачів.

У межах розділу також розглянуто особливості розгортання програмної системи, структуру інсталяційного пакету та організацію локального середовища виконання. Встановлено, що система не потребує складної серверної інфраструктури та може функціонувати автономно на робочій станції користувача під керуванням Windows, Linux або macOS.

Отримані результати тестування підтвердили відповідність розробленого програмного забезпечення функціональним і нефункціональним вимогам, визначеним на попередніх етапах проєктування. Інформаційна система забезпечує стабільну роботу основних модулів, коректне формування аналітичних показників і можливість практичного використання для

моніторингу успішності студентів та підтримки прийняття рішень в освітньому середовищі.

ВИСНОВКИ

У кваліфікаційній роботі вирішено актуальне завдання розроблення інформаційної системи моніторингу успішності студентів із формуванням персоналізованих навчальних рекомендацій. Актуальність теми зумовлена необхідністю підвищення ефективності аналізу навчальних результатів, автоматизації контролю академічної успішності та підтримки прийняття рішень у сучасному освітньому середовищі.

У процесі виконання роботи проведено аналіз предметної області моніторингу успішності студентів та досліджено сучасні підходи до побудови освітніх аналітичних систем. Виконано огляд існуючих програмних рішень, визначено їх переваги та недоліки, а також обґрунтовано доцільність створення власної інформаційної системи, орієнтованої на локальне використання, автоматизовану аналітику та персоналізовані рекомендації.

У межах роботи сформовано функціональні та нефункціональні вимоги до системи, визначено структуру вхідних і вихідних даних, а також побудовано модель предметної області. Розроблено логічну та фізичну моделі бази даних, які забезпечують зберігання інформації про студентів, дисципліни, оцінки, компетентності, показники успішності, рекомендації, звіти та журнал аудиту.

У результаті проєктування створено архітектуру програмної системи на основі модульного підходу. Запропонована архітектура включає графічний клієнт, модуль автентифікації, аналітичний модуль, підсистему AI-рекомендацій, модуль звітності та локальне сховище SQLite. Використання Python, PyQt6, pandas, scikit-learn та SQLite дозволило реалізувати кросплатформний настільний застосунок із можливістю локального розгортання без необхідності використання окремого серверного середовища.

У роботі реалізовано алгоритми автентифікації користувачів, аналізу навчальних показників, визначення рівня академічного ризику та формування персоналізованих рекомендацій. Побудовані алгоритмічні моделі забезпечують автоматизоване оброблення освітніх даних, формування KPI-показників і підтримку прийняття рішень щодо навчального процесу.

Розроблена програмна система забезпечує:

– моніторинг успішності студентів;
– аналіз відвідуваності та активності;
– розрахунок інтегральних показників успішності;
– визначення рівня академічного ризику;
– формування персоналізованих рекомендацій;
– створення кураторських втручань;
– побудову аналітичних графіків;
– формування та експорт звітів;
– аудит дій користувачів і системних операцій.

У четвертому розділі виконано тестування розробленого програмного забезпечення. Перевірено коректність функціонування основних модулів, стабільність взаємодії між компонентами системи, правильність оброблення навчальних даних і надійність роботи локальної бази даних. Результати тестування підтвердили відповідність системи поставленим функціональним і нефункціональним вимогам.

Практичне значення роботи полягає у можливості використання розробленої системи в закладах освіти для автоматизації моніторингу успішності студентів, підвищення ефективності аналітичної обробки навчальних даних та підтримки роботи викладачів, кураторів і адміністрації. Використання системи дозволяє своєчасно виявляти студентів групи ризику, формувати індивідуальні рекомендації та підвищувати якість організації освітнього процесу.

Перспективами подальшого розвитку роботи є інтеграція системи з електронними журналами та ERP-платформами закладів освіти, реалізація веб-версії програмного забезпечення, використання розширених моделей машинного навчання для прогнозування академічної успішності та впровадження адаптивних механізмів персоналізації навчальних маршрутів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Python Software Foundation. Python 3 Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/ – Назва з екрана. – Дата звернення: 04.05.2026.

2. Python Software Foundation. sqlite3 - DB-API 2.0 interface for SQLite databases : офіційна документація. [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/library/sqlite3.html – Назва з екрана. – Дата звернення: 04.05.2026.

3. Riverbank Computing. PyQt6 Reference Guide : офіційна документація. [Електронний ресурс]. – Режим доступу: https://www.riverbankcomputing.com/static/Docs/PyQt6/ – Назва з екрана. – Дата звернення: 04.05.2026.

4. Qt Group. Qt for Python Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://doc.qt.io/qtforpython-6/ – Назва з екрана. – Дата звернення: 04.05.2026.

5. SQLite Consortium. SQLite Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://sqlite.org/docs.html – Назва з екрана. – Дата звернення: 04.05.2026.

6. SQLite Consortium. Datatypes In SQLite : офіційна документація. [Електронний ресурс]. – Режим доступу: https://www.sqlite.org/datatype3.html – Назва з екрана. – Дата звернення: 04.05.2026.

7. McKinney W. pandas: Python Data Analysis Library : офіційна документація. [Електронний ресурс]. – Режим доступу: https://pandas.pydata.org/docs/ – Назва з екрана. – Дата звернення: 04.05.2026.

8. Harris C. R., Millman K. J., van der Walt S. J. et al. Array programming with NumPy. Nature. 2020. Vol. 585. P. 357-362. [Електронний ресурс]. – Режим доступу: https://www.nature.com/articles/s41586-020-2649-2 – Назва з екрана. – Дата звернення: 04.05.2026.

9. NumPy Developers. NumPy Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://numpy.org/doc/ – Назва з екрана. – Дата звернення: 04.05.2026.

10. Hunter J. D. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering. 2007. Vol. 9, No. 3. P. 90-95. [Електронний ресурс]. – Режим доступу: https://matplotlib.org/stable/users/project/history.html – Назва з екрана. – Дата звернення: 04.05.2026.

11. Matplotlib Development Team. Matplotlib Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://matplotlib.org/stable/index.html – Назва з екрана. – Дата звернення: 04.05.2026.

12. Pedregosa F., Varoquaux G., Gramfort A. et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011. Vol. 12. P. 2825-2830. [Електронний ресурс]. – Режим доступу: https://jmlr.org/papers/v12/pedregosa11a.html – Назва з екрана. – Дата звернення: 04.05.2026.

13. Scikit-learn Developers. Scikit-learn: Machine Learning in Python : офіційна документація. [Електронний ресурс]. – Режим доступу: https://scikit-learn.org/stable/ – Назва з екрана. – Дата звернення: 04.05.2026.

14. PyInstaller Development Team. PyInstaller Manual : офіційна документація. [Електронний ресурс]. – Режим доступу: https://pyinstaller.org/en/stable/ – Назва з екрана. – Дата звернення: 04.05.2026.

15. pytest Development Team. pytest Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://docs.pytest.org/en/stable/ – Назва з екрана. – Дата звернення: 04.05.2026.

16. GitHub Docs. GitHub Actions Documentation : офіційна документація. [Електронний ресурс]. – Режим доступу: https://docs.github.com/en/actions – Назва з екрана. – Дата звернення: 04.05.2026.

17. OWASP Foundation. OWASP Top 10: Web Application Security Risks. [Електронний ресурс]. – Режим доступу: https://owasp.org/www-project-top-ten/ – Назва з екрана. – Дата звернення: 04.05.2026.

18. OWASP Foundation. Authentication Cheat Sheet. [Електронний ресурс]. – Режим доступу: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html – Назва з екрана. – Дата звернення: 04.05.2026.

19. OWASP Foundation. Password Storage Cheat Sheet. [Електронний ресурс]. – Режим доступу: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html – Назва з екрана. – Дата звернення: 04.05.2026.

20. Moodle Docs. Analytics : офіційна документація Moodle. [Електронний ресурс]. – Режим доступу: https://docs.moodle.org/en/Analytics – Назва з екрана. – Дата звернення: 04.05.2026.

21. Google. Classroom Help : офіційний довідковий центр. [Електронний ресурс]. – Режим доступу: https://support.google.com/edu/classroom/ – Назва з екрана. – Дата звернення: 04.05.2026.

22. Instructure. Canvas LMS API Documentation. [Електронний ресурс]. – Режим доступу: https://developerdocs.instructure.com/services/canvas – Назва з екрана. – Дата звернення: 04.05.2026.

23. Instructure. Canvas Analytics API Documentation. [Електронний ресурс]. – Режим доступу: https://developerdocs.instructure.com/services/canvas/resources/analytics – Назва з екрана. – Дата звернення: 04.05.2026.

24. Anthology. Analytics for Learn : Blackboard Help. [Електронний ресурс]. – Режим доступу: https://help.anthology.com/blackboard/instructor/en/analytics/analytics-for-learn.html – Назва з екрана. – Дата звернення: 04.05.2026.

25. Lang C., Siemens G., Wise A. F., Gašević D., Merceron A. Handbook of Learning Analytics. 2nd ed. Vancouver : Society for Learning Analytics Research,

2022. [Електронний ресурс]. – Режим доступу: https://www.solaresearch.org/publications/hla-22/ – Назва з екрана. – Дата звернення: 04.05.2026.

26. Siemens G., Long P. Penetrating the Fog: Analytics in Learning and Education. EDUCAUSE Review. 2011. Vol. 46, No. 5. P. 30-40. [Електронний ресурс]. – Режим доступу: https://er.educause.edu/articles/2011/9/penetrating-the-fog-analytics-in-learning-and-education – Назва з екрана. – Дата звернення: 04.05.2026.

27. Romero C., Ventura S. Educational data mining and learning analytics: An updated survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2020. Vol. 10, No. 3. [Електронний ресурс]. – Режим доступу: https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1355 – Назва з екрана. – Дата звернення: 04.05.2026.

28. Sclater N., Peasgood A., Mullan J. Learning Analytics in Higher Education: A review of UK and international practice. London : Jisc, 2016. [Електронний ресурс]. – Режим доступу: https://sclater.com/blog/learning-analytics-in-higher-education-a-review-of-uk-and-international-practice/ – Назва з екрана. – Дата звернення: 04.05.2026.

29. Chatti M. A., Dyckhoff A. L., Schroeder U., Thüs H. A reference model for learning analytics. International Journal of Technology Enhanced Learning. 2012. Vol. 4, No. 5/6. P. 318-331. [Електронний ресурс]. – Режим доступу: https://www.researchgate.net/publication/259904630_A_reference_model_for_learning_analytics – Назва з екрана. – Дата звернення: 04.05.2026.

30. Ferguson R. Learning analytics: drivers, developments and challenges. International Journal of Technology Enhanced Learning. 2012. Vol. 4, No. 5/6. P. 304-317. [Електронний ресурс]. – Режим доступу: https://oro.open.ac.uk/36374/ – Назва з екрана. – Дата звернення: 04.05.2026.

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Євтодій Олександр Павлович, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Розробка веб-орієнтованої інформаційної системи для управління студентськими проектами» є результатом моєї особистої праці.

2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.

3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):

- [✓] інструменти генеративного ШІ не використовувалися.
- [✓] інструменти ШІ (вказати назву: ChatGPT, Gemini, Claude, DeepL, _____інші (вказати назву)) були використані для:
  - [ ] перекладу іншомовних джерел;
  - [✓] стилістичного редагування та перевірки граматики;
  - [✓] допомоги у написанні/відлагодженні програмного коду;
  - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«__» _____2026р. _____ Євтодій Олександр
(підпис) **(Ім'я та ПРІЗВИЩЕ)**