МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК, ФІЗИКИ ТА МАТЕМАТИКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

СУЧАСНИЙ УНІВЕРСИТЕТ У СЕРЕДОВИЩІ WEB 3.0: ЦИФРОВІЗАЦІЯ, ДЕЦЕНТРАЛІЗАЦІЯ ТА ІНТЕГРАЦІЯ AI-ТЕХНОЛОГІЙ

Кваліфікаційна робота (проєкт)

на здобуття ступеня вищої освіти «бакалавр»

Виконав: здобувач 4 курсу 12-431 групи
Спеціальності Комп'ютерні науки
Освітньо-професійної програми
Комп'ютерні науки
Кривошеєнко Євген Вадимович

Керівник: доктор філософії, доцент
Полторацький М. Ю.

Рецензент: Тарасіч Ю. Г., науковий співробітник, Інституту кібернетики ім. В.М. Глушкова НАН України

Івано-Франківськ – 2026

ЗМІСТ

ВСТУП 3

РОЗДІЛ 1. ТЕОРЕТИЧНІ ЗАСАДИ ЦИФРОВІЗАЦІЇ, ДЕЦЕНТРАЛІЗАЦІЇ ТА ІНТЕГРАЦІЇ AI-ТЕХНОЛОГІЙ В УНІВЕРСИТЕТСЬКОМУ СЕРЕДОВИЩІ WEB 3.0 6

1.1. Еволюція цифрових освітніх середовищ: від Web 1.0 до Web 3.0 6

1.2. Можливості Web 3.0 в освіті 7

1.3. Інтеграція Blockchain та IPFS як основа децентралізованої інфраструктури університету 9

1.4. Використання AI-технологій в освітніх процесах університету 11

1.5. Агентний підхід та оркестрація AI-агентів 12

РОЗДІЛ 2. Розробка прототипу децентралізованої системи тестування 14

2.1. Постановка задачі та вимоги до MVP-прототипу 14

2.2. Архітектура прототипу та взаємодія компонентів 15

2.3. Збереження об’єктів у IPFS та формування CID 17

2.4. Реєстрація CID у реєстрі та перевірка цілісності 18

2.5. AI-модуль оцінювання та логіка обробки результатів 19

2.6. Тестування прототипу та аналіз демонстраційних сценаріїв 20

2.7. Обмеження прототипу та напрями подальшого розвитку 21

ВИСНОВКИ 23

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 25

ДОДАТКИ 27

ВСТУП

Сучасна цифровізація вищої освіти вже давно вийшла за межі окремих електронних сервісів. Університет дедалі частіше функціонує як цілісне цифрове середовище, у якому поєднуються дистанційне навчання, електронний документообіг, сервіси сертифікації, навчальна аналітика та інструменти автоматизованого оцінювання. Відповідно зростають не лише можливості роботи з даними, а й вимоги до їх надійності, простежуваності та коректного зберігання.

Попри це, велика частина університетських цифрових рішень і далі працює в межах централізованої моделі, де зберігання результатів, фіксація змін і контроль доступу зосереджені в одній платформі. Для освітнього процесу це створює помітне обмеження: зовнішній користувач зазвичай не має простого способу перевірити, чи не були результати тестування, сертифікати або службові записи змінені після первинного внесення.

Саме тому підходи Web 3.0 привертають дедалі більшу увагу як основа для іншої логіки організації цифрового університетського середовища. Поєднання контент-адресного зберігання, реєстрації технічних ідентифікаторів і криптографічної перевірки цілісності дає змогу зробити освітні дані прозорішими та придатними до перевірки. Одночасно інтеграція AI-технологій розширює можливості автоматизованого оцінювання, навчальної аналітики та підтримки рішень у межах університетської цифрової інфраструктури.

Актуальність теми «Сучасний університет у середовищі Web 3.0: цифровізація, децентралізація та інтеграція AI-технологій» зумовлена потребою поєднати зручність цифрових сервісів із вищим рівнем довіри до результатів оцінювання. Для університетських систем важливими стають не лише швидкість і функціональність, а й можливість підтвердити цілісність даних, простежити історію їх фіксації та зменшити залежність від закритої централізованої логіки.

Мета роботи полягає в обґрунтуванні підходу до побудови децентралізованої системи тестування для університетського середовища на основі Web 3.0 та AI-технологій, а також у проєктуванні MVP-прототипу, у якому тестові матеріали й результати зберігаються як контент-адресні об’єкти, а їх ідентифікатори фіксуються в окремому реєстрі.

Для досягнення поставленої мети визначено такі завдання дослідження:

проаналізувати еволюцію цифрових освітніх середовищ у контексті переходу від Web 1.0 до Web 3.0;

визначити можливості та обмеження застосування принципів Web 3.0 в університетських інформаційних системах;

розкрити роль Blockchain та IPFS у забезпеченні цілісності та перевірюваності освітніх даних;

охарактеризувати напрями використання AI-технологій у системах тестування й навчальної аналітики;

описати агентний підхід та логіку оркестрації AI-компонентів у цифровому освітньому середовищі;

спроєктувати архітектуру MVP-прототипу децентралізованої системи тестування;

подати демонстраційний сценарій роботи прототипу та визначити напрями його подальшого розвитку.

Об’єкт дослідження — інформаційні процеси цифрового освітнього середовища університету. Предмет дослідження — архітектурні, програмні та організаційні рішення щодо інтеграції технологій Web 3.0 та AI у систему тестування.

У роботі використано аналіз наукових і навчально-методичних джерел, системний аналіз, моделювання архітектури інформаційної системи, порівняння централізованих і децентралізованих підходів, програмне проєктування та сценарне тестування прототипу. Практичне значення отриманих результатів

полягає в тому, що запропоновану архітектуру можна використати під час розроблення університетських сервісів, де важливими є прозорість оцінювання, фіксація змін і подальша інтеграція інтелектуальних модулів аналізу даних.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ЗАСАДИ ЦИФРОВІЗАЦІЇ, ДЕЦЕНТРАЛІЗАЦІЇ ТА ІНТЕГРАЦІЇ AI-ТЕХНОЛОГІЙ В УНІВЕРСИТЕТСЬКОМУ СЕРЕДОВИЩІ WEB 3.0

1.1. Еволюція цифрових освітніх середовищ: від Web 1.0 до Web 3.0

Розвиток цифрових освітніх середовищ варто розглядати як поступову зміну способів взаємодії користувачів з інформацією та моделей контролю над даними. Кожний етап еволюції веб-технологій впливав не лише на інтерфейс систем, а й на логіку зберігання відомостей, побудову сервісів і рівень довіри до цифрових результатів. [1]

Для етапу Web 1.0 були характерні переважно статичні веб-сторінки. Університетські сайти того часу виконували інформаційну функцію: розклад занять, контакти кафедр, короткі методичні матеріали, інколи електронні бібліотеки. Взаємодія студента з такими ресурсами була односторонньою, а самі системи не передбачали ані гнучкої персоналізації, ані автоматизованого оцінювання.

Перехід до Web 2.0 приніс інтерактивність, особисті кабінети, хмарні сервіси, дистанційні курси й спільну роботу з контентом. Саме на цьому етапі університети почали масово використовувати LMS, електронні журнали, системи тестування та централізовані освітні платформи. З точки зору зручності це був значний крок уперед, оскільки викладач отримав інструменти швидкої перевірки, а студент — постійний онлайн-доступ до навчальних матеріалів. [2]

Водночас Web 2.0 зберіг ключову особливість клієнт-серверної моделі: дані залишаються під контролем провайдера або адміністратора системи. Якщо оцінка змінена в базі даних, то зовнішній користувач зазвичай не має простого механізму довести сам факт редагування. Саме тут виникає проблема, яка в умовах освітнього процесу безпосередньо пов’язана з довірою до результатів навчання.

Web 3.0 не заперечує попередні підходи, але помітно змінює акценти. У центрі уваги опиняються розподілені сервіси, цифрові ідентичності, криптографічна верифікація та контроль над доступом до даних без повної залежності від одного вузла або одного адміністратора. Для університетського середовища це означає можливість організувати зберігання та перевірку даних так, щоб підсумковий результат сприймався не лише як запис у системі, а як технічно підтверджений факт. [3]

У сфері освіти такий перехід має не лише технологічний, а й організаційний зміст. З'являється можливість по-іншому працювати з сертифікатами, результатами тестів, електронними портфоліо студентів, журналами подій і міжуніверситетським обміном даними. При цьому мова не йде про повну відмову від централізованих платформ. Практичнішим є гібридний підхід, коли звичний backend поєднується з розподіленим сховищем і механізмами криптографічного підтвердження.

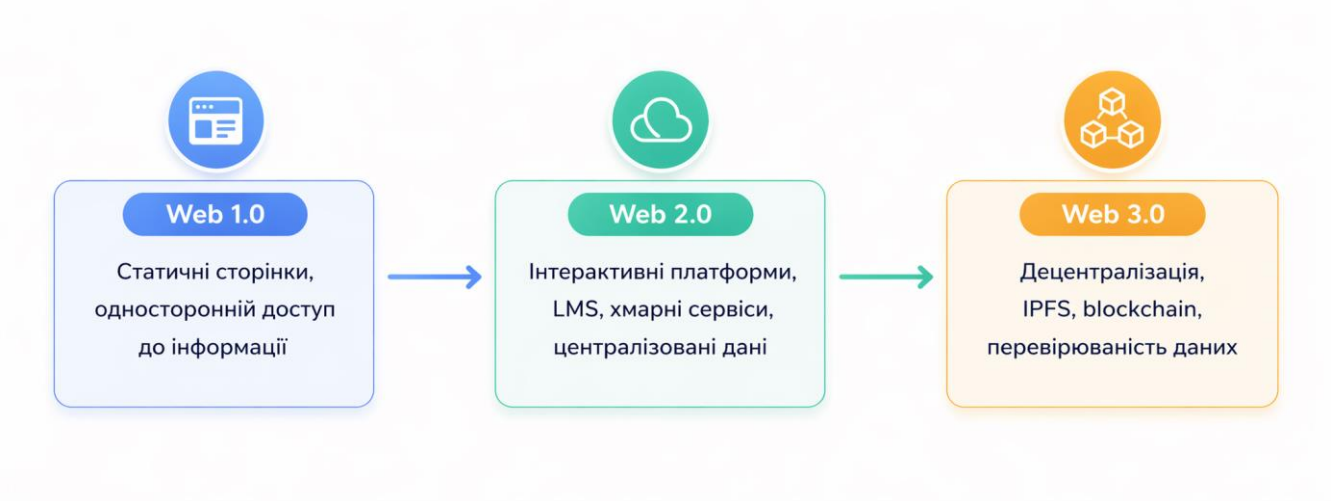


Рисунок 1.1 - Еволюція веб-середовищ від Web 1.0 до Web 3.0 в освітньому контексті

1.2. Можливості Web 3.0 в освіті

Університетська інфраструктура включає велику кількість даних різної природи: навчальні матеріали, результати тестування, відомості про проходження курсів, сертифікати, записи про активність користувачів. Не всі ці дані однаково

чутливі, але для значної частини з них важливими є незмінність, відтворюваність і можливість зовнішньої перевірки. Саме тут принципи Web 3.0 можуть принести реальну користь.

Першою практично значущою можливістю є фіксація результатів навчання в такому вигляді, щоб їхню цілісність можна було підтвердити незалежно від конкретної LMS. Якщо тест або результат подано як контент-адресний об’єкт, а його ідентифікатор зареєстрований у незмінному журналі, тоді будь-яка зміна вмісту автоматично створить нову версію. Це дає університету не декларативну, а технічну основу для аудиту. [4]

Друга важлива можливість пов’язана з цифровою ідентичністю. Підходи DID і верифікованих облікових даних перспективні для ситуацій, коли потрібно підтвердити окремий факт: наприклад, статус студента, успішне завершення курсу або право доступу до певного ресурсу. Перевага такого підходу полягає в тому, що користувач може розкривати лише потрібний атрибут, а не увесь набір персональних даних. [5]

Третій напрям стосується автоматизації правил доступу. У межах освітньої системи це можуть бути сценарії допуску до тесту після виконання попередніх модулів, автоматичне відкриття сертифіката після досягнення необхідного результату, фіксація ключових подій проходження курсу або реєстрація змін версій навчальних матеріалів. У промисловій системі такі правила може реалізовувати смарт-контракт, однак для прототипу їх допустимо моделювати на рівні backend-логіки та реєстру подій.

Варто враховувати і те, що Web 3.0 не є універсальним рішенням для всіх освітніх задач. Частина університетських процесів краще працює в класичних централізованих системах, особливо там, де важливі швидкість, просте адміністрування та робота з конфіденційними даними. Тому доцільно говорити не про повну заміну існуючих платформ, а про вибіркове застосування децентралізованих механізмів там, де вони дійсно підсилюють довіру і прозорість.

Отже, практична цінність Web 3.0 для освіти полягає не стільки в самій новизні технологій, скільки в можливості розв’язувати конкретні прикладні завдання: фіксувати результати, підтверджувати їх справжність, зменшувати ризик непомітного редагування та формувати більш надійне середовище взаємодії між студентом, викладачем і навчальною платформою.

1.3. Інтеграція Blockchain та IPFS як основа децентралізованої інфраструктури університету

Для освітніх систем важливо розділити дві задачі: де зберігаються дані і як довести, що вони не змінювалися після фіксації. У централізованій платформі ці дві функції зазвичай поєднані в одній базі даних, через що вся довіра фактично концентрується в адміністраторі сервісу. У децентралізованій моделі ці ролі можна розвести між різними компонентами. [6]

IPFS доцільно використовувати як середовище зберігання навчальних матеріалів і результатів тестування. Його ключова властивість полягає в контент-адресації: об’єкт отримує ідентифікатор, який залежить від самого вмісту. Якщо змінити питання в тесті, порядок відповідей або підсумковий бал, система отримає вже інший CID. Саме тому CID можна розглядати як цифровий відбиток певної версії даних. [7]

Однак сам факт збереження в IPFS ще не гарантує, що користувач знає, яка саме версія є офіційно зафіксованою. Для цього потрібен окремий реєстр, у який записується CID разом із мінімальними технічними метаданими. У повній реалізації цю роль може виконувати блокчейн-контракт. Його функція полягає не в зберіганні великих файлів, а в реєстрації коротких перевірюваних записів про події.

Поєднання IPFS та блокчейн-підходу утворює логічний ланцюг. Спочатку тест або результат серіалізується та зберігається у контент-адресному сховищі. Далі отриманий CID передається до реєстру подій. Під час перевірки достатньо

повторно отримати об’єкт за CID, переконатися, що вміст відповідає зареєстрованому ідентифікатору, і таким чином підтвердити цілісність запису. [8]



Рисунок 1.2 - Взаємодія IPFS та blockchain у моделі децентралізованого зберігання

Для університету така архітектура має кілька переваг. По-перше, зникає потреба зберігати великі файли безпосередньо в незмінному реєстрі, що було б технічно надмірним. По-друге, спрощується аудит: реєстр містить стислий слід подій, а самі об’ємні дані залишаються у файловому сховищі. По-третє, система стає зручнішою для масштабування, оскільки логіка перевірки не залежить від формату конкретного навчального матеріалу. [9]

Не менш важливим є питання доступності даних. Для реального використання IPFS-об’єкти необхідно закріплювати на контрольованих вузлах або за допомогою pinning-сервісів. Якщо цього не зробити, виникне ризик, що об’єкт формально має CID, але фізично не може бути швидко отриманий. Отже, побудова децентралізованої інфраструктури для університету передбачає не лише криптографічну логіку, а й продуману організацію зберігання. [10]

1.4. Використання AI-технологій в освітніх процесах університету

AI-технології в університетському середовищі доцільно розглядати не як заміну викладача, а як інструмент підтримки рішень і автоматизації рутинних процесів. У сфері тестування це насамперед швидке оцінювання, аналіз типових помилок, виявлення повторюваних патернів у відповідях та формування рекомендацій для студента і викладача. [11]

Для закритих тестів найнадійнішим є алгоритмічний підрахунок балів за наперед визначеними правилами. Такий механізм легко перевірити, відтворити та пояснити. У практичному університетському сервісі саме він має залишатися базовим, оскільки забезпечує передбачуваний результат і не викликає сумнівів у критеріях оцінювання.

Складнішим напрямом є обробка відкритих відповідей. Тут можуть використовуватися NLP-моделі, які аналізують зміст тексту, виявляють тематичну відповідність, оцінюють повноту відповіді або шукають збіг із еталонними формулюваннями. Однак для навчальної системи важливо, щоб подібні інструменти працювали як допоміжні, а не як повністю автономні. Вони потребують валідації, прозорих критеріїв та можливості перегляду результату викладачем. [12]

Ще один вагомий напрям — навчальна аналітика. На основі накопичених результатів система може показувати, які завдання виявилися найскладнішими, де студенти помиляються системно, як змінюється успішність групи з часом. Такі дані цінні не лише для оцінювання, а й для коригування методики навчання, перегляду структури курсу та підготовки додаткових матеріалів. [13]

Окремої уваги заслуговує виявлення аномалій. Наприклад, система може помітити нетипово короткий час виконання тесту, різкий стрибок результату після тривалого періоду слабких показників або однакові патерни відповідей у кількох користувачів. Важливо підкреслити, що подібні сигнали не є автоматичним

доказом порушення. Їх слід сприймати як підставу для додаткової перевірки з урахуванням контексту.

Поєднання AI з Web 3.0 є особливо важливим саме тому, що результати автоматизованої обробки також можуть бути зафіксовані як окремі контент-адресні об’єкти. У такому разі зберігається не лише підсумковий бал, а й логіка формування результату, набір відповідей, часові параметри та інші технічні дані, які за потреби можна перевірити повторно. [14]

1.5. Агентний підхід та оркестрація AI-агентів

У сучасних цифрових системах дедалі частіше використовується агентний підхід, за якого окремі програмні модулі виконують вузькі функції та взаємодіють між собою через визначені правила. Для університетського середовища це зручно, оскільки різні завдання — оцінювання, аналітика, перевірка цілісності, повідомлення користувачів — можуть бути реалізовані незалежними компонентами без надмірного ускладнення однієї центральної служби. [15]

У межах системи тестування можна умовно виділити кілька агентів. Перший агент відповідає за перевірку структури тесту і формування об’єкта для збереження. Другий — за оцінювання відповідей студента. Третій — за аналітику й побудову коротких рекомендацій. Четвертий — за взаємодію з модулем реєстрації CID. Така декомпозиція спрощує розвиток системи: зміна одного модуля не вимагає повної перебудови всієї архітектури.

Ключовим елементом у такій моделі є оркестрація, тобто узгодження послідовності дій агентів. Наприклад, модулю аналітики доцільно формувати рекомендації лише після завершення оцінювання та збереження первинного результату. Аналогічно, реєстрація CID має відбуватися тоді, коли об’єкт уже сформований у коректному форматі й готовий до збереження. Отже, оркестрація забезпечує впорядкований рух даних через систему. [16]



Рисунок 1.3 - Агентний підхід та оркестрація модулів у прототипі

Перевагою такого підходу є модульність. Якщо університет надалі захоче додати інший AI-модуль, наприклад систему перевірки відкритих відповідей або окремий сервіс виявлення аномалій, це можна буде зробити без зміни базової логіки зберігання результатів. Достатньо підключити новий агент до вже існуючого сценарію обробки.

Водночас агентний підхід вимагає дисципліни в проєктуванні. Модулі мають працювати з узгодженими форматами даних, мати чітко визначені вхідні та вихідні параметри і не дублювати відповідальність один одного. Інакше система втрачатиме прозорість, а саме вона є однією з ключових цілей досліджуваної архітектури.

Для бакалаврського прототипу достатньо не повної мультиагентної реалізації, а переконливої демонстрації самої логіки: оцінювання виконується

окремим модулем, аналітика — окремим шаром, а реєстрація технічних артефактів — окремим реєстром. Такий підхід добре узгоджується з концепцією Web 3.0 і водночас створює основу для подальшого розвитку системи.

РОЗДІЛ 2. Розробка прототипу децентралізованої системи тестування

2.1. Постановка задачі та вимоги до MVP-прототипу

Практична частина роботи спрямована на створення MVP-прототипу, який показує основну ідею дослідження без зайвого ускладнення промисловими деталями. Йдеться не про побудову повної університетської платформи, а про демонстрацію наскрізного сценарію: створення тесту, проходження тесту студентом, автоматичне оцінювання, збереження результату в контент-адресному середовищі та фіксацію ідентифікатора в реєстрі. [17]

Функціональні вимоги до такого прототипу можна окреслити досить чітко. Система має надавати викладачеві можливість створити тест із набором запитань і правильних відповідей, а студенту — пройти тест та отримати підсумковий бал. Після завершення спроби результат доцільно перетворювати на структурований об’єкт і передавати до модуля збереження. Паралельно реєструється технічний ідентифікатор версії, який у подальшому використовується для перевірки цілісності.

Не менш важливими є нефункціональні вимоги. Прототип має бути відтворюваним: однакові дані мають давати однаковий результат серіалізації та той самий CID. Система також має коректно обробляти помилки, пов’язані з недоступністю сховища, некоректною структурою запиту або спробою звернення до неіснуючого об’єкта. Крім того, у прототипі варто передбачити базове розділення ролей між викладачем і студентом.

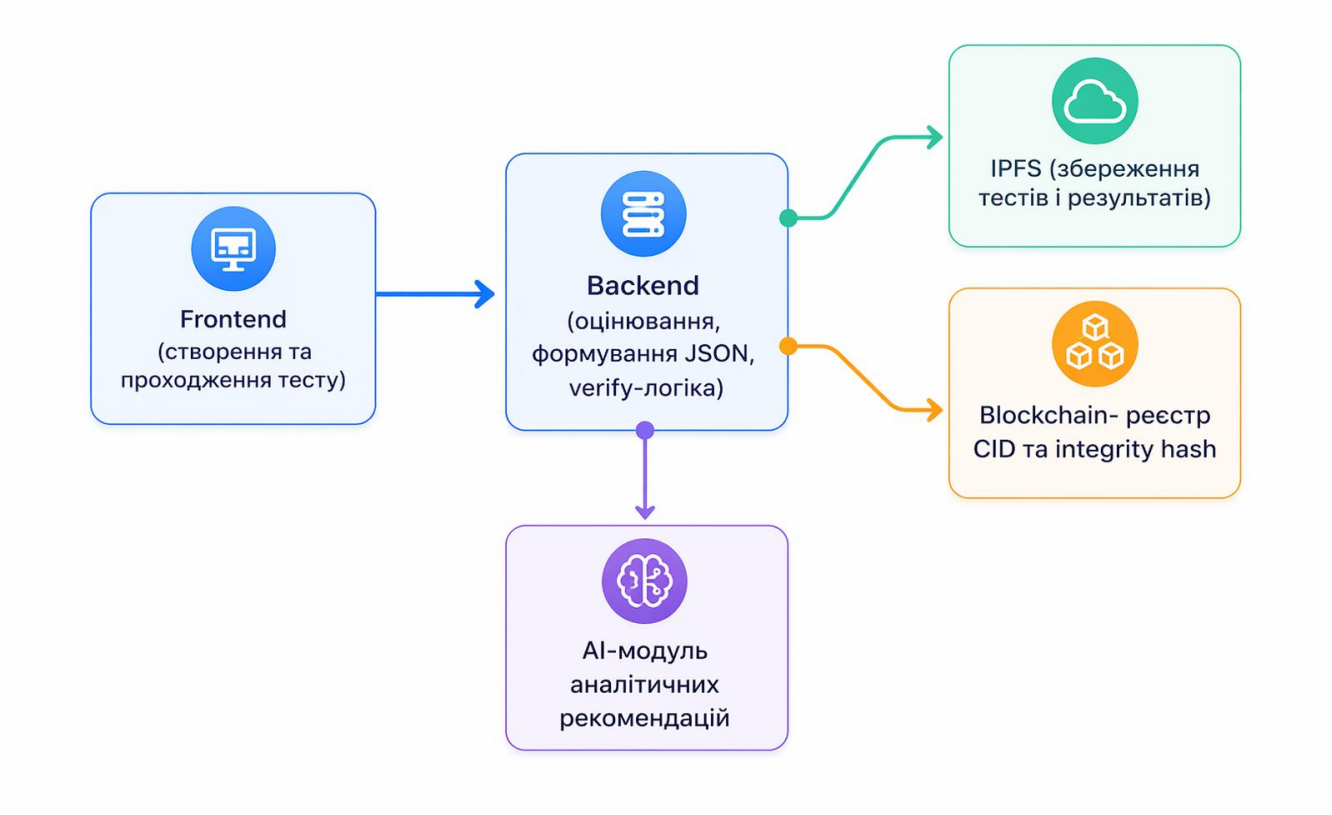


Рисунок 2.1 - Архітектура MVP-прототипу децентралізованої системи тестування

Окремою вимогою є мінімізація відкритих персональних даних. Навіть у демонстраційному сценарії недоцільно зберігати в IPFS повні персональні профілі користувачів. Для технічної перевірки достатньо використати службові ідентифікатори, псевдоніми або скорочені метадані, які не розкривають зайвої інформації про конкретного студента.

Таким чином, MVP-прототип має одночасно виконувати кілька функцій: демонструвати логіку децентралізованого зберігання, залишатися зрозумілим для перевірки та створювати основу для подальшого переходу до повноцінної системи з блокчейн-контрактом, розширеною авторизацією і глибшою AI-аналітикою.

2.2. Архітектура прототипу та взаємодія компонентів

Архітектура прототипу побудована за принципом розподілу відповідальності між кількома компонентами. Інтерфейс користувача відповідає за взаємодію з тестом: відображення питань, збір відповідей і передання даних на

сервер. Серверний рівень виконує перевірку запитів, оцінювання результату, підготовку об’єкта до збереження та взаємодію з реєстром CID. [18]

На backend-рівні доцільно виділити кілька логічних модулів. Перший модуль працює з тестами: створення, оновлення, отримання активної версії. Другий модуль відповідає за спроби проходження: приймає набір відповідей студента, нормалізує їх і передає в модуль оцінювання. Третій модуль виконує підрахунок балів та формує детальну структуру результату. Четвертий модуль забезпечує контент-адресне збереження і повертає CID.

Окремо функціонує реєстр подій, який у межах MVP моделює блокчейн-компонент. Його завдання полягає в тому, щоб прийняти CID і зафіксувати мінімальний набір метаданих: тип об’єкта, час реєстрації, технічний ідентифікатор ініціатора та, за потреби, посилання на попередню версію. Сам реєстр не зберігає вміст тесту або результату, а лише короткий слід події.

Логіка роботи системи виглядає так. Викладач створює тест, backend формує структурований опис і передає його до IPFS. Після отримання CID сервер записує цей ідентифікатор до реєстру. Коли студент проходить тест, система зберігає вже не тест, а результат спроби: відповіді, підсумковий бал, час подання та службові метадані. Для цього результату також отримується CID, який знову реєструється як окрема подія.

Подібна архітектура дозволяє чітко відокремити функції інтерфейсу, логіки обробки, зберігання та перевірки. Для бакалаврського дослідження це важливо ще й тому, що така структура легко пояснюється під час захисту: кожен компонент має зрозумілу роль, а взаємозв’язки між ними відповідають загальній концепції Web 3.0.

2.3. Збереження об’єктів у IPFS та формування CID

У прототипі тест і результат доцільно подавати у вигляді структурованих JSON-об’єктів. Це спрощує і серіалізацію, і подальшу перевірку, і логічне розділення даних за типами. Для тесту типовими полями є назва, перелік запитань, типи запитань, варіанти відповідей, правильні відповіді та службова інформація про автора або версію. Для результату — ідентифікатор тесту, відповіді користувача, підсумковий бал, максимально можливий бал, час проходження й технічні мітки. [19]



Рисунок 2.2 - Сценарій перевірки цілісності результату тестування

Ключовою вимогою є стабільність формату. Якщо одна й та сама структура щоразу формується по-різному, наприклад зі змінним порядком полів, це ускладнює відтворюваність результату. Тому перед передаванням до IPFS об’єкт повинен бути нормалізований: поля розташовані в узгодженій послідовності, зайві змінні вилучені, а формат числових і часових значень приведений до єдиного стандарту.

Після серіалізації об’єкт передається в IPFS, де йому присвоюється CID. Саме цей ідентифікатор використовується в усій подальшій логіці системи. Якщо змінити хоча б один елемент тесту або результату, система сформує інший CID.

На практиці це означає, що будь-яка нова редакція автоматично стає окремою версією, а не непомітним редагуванням уже наявного запису.

Під час проєктування важливо також врахувати питання доступності. Для навчального середовища недостатньо лише отримати CID; потрібно бути впевненим, що об’єкт реально зберігається і може бути отриманий пізніше. Саме тому в реальних системах застосовують pinning. У межах прототипу досить закласти цю вимогу в архітектуру і чітко визначити точку, у якій сервіс збереження повинен або підтвердити успішну фіксацію об’єкта, або повернути помилку.

2.4. Реєстрація CID у реєстрі та перевірка цілісності

Після формування контент-адресного об’єкта система має зафіксувати сам факт його появи. Для цього в прототипі використовується реєстр CID, який моделює поведінку незмінного журналу. Запис у такому реєстрі містить щонайменше сам CID, тип об’єкта, часову мітку та службовий ідентифікатор суб’єкта, що ініціював дію. Редагування наявного запису не передбачається: нова версія додається як окрема подія.

Саме наявність окремого реєстру дозволяє відокремити «дані» від «доказу їх фіксації». Якщо зберігати лише об’єкт в IPFS, то теоретично можна мати кілька версій і не знати, яка з них є офіційною для певного етапу роботи. Реєстр вирішує цю проблему, оскільки закріплює той CID, який система визнає чинним на момент створення тесту або завершення спроби.

Перевірка цілісності в такій архітектурі виконується послідовно. Спочатку система або зовнішній перевіряльник читає CID із реєстру. Далі за цим значенням отримується об’єкт зі сховища. Після цього ідентифікатор повторно звіряється з вмістом. Якщо отриманий об’єкт відповідає зареєстрованому CID, можна стверджувати, що його зміст не змінювався після фіксації.

Для демонстраційного прототипу такої моделі цілком достатньо. Вона відтворює головну ідею майбутнього смарт-контракту: реєстр не повинен зберігати великі масиви даних, проте має фіксувати незмінний факт події. Саме тому перехід від локального журналу до блокчейн-контракту не змінює загальної логіки системи, а лише переносить запис у інше середовище.

З практичної точки зору доцільно реєструвати не лише CID, а й тип події: створення тесту, оновлення тесту, завершення спроби, перевірка результату. Такий підхід покращує аудит, дозволяє відновити послідовність дій і робить систему зрозумілішою як для внутрішнього використання, так і для зовнішнього аналізу.

2.5. AI-модуль оцінювання та логіка обробки результатів

У межах MVP недоцільно одразу впроваджувати складні генеративні моделі оцінювання. Практичніше побудувати прозорий AI-модуль навколо двох рівнів логіки: детермінованого оцінювання закритих завдань і додаткової аналітичної обробки після підрахунку балів. Такий підхід допомагає зберегти керованість системи та уникнути зайвих сумнівів щодо критеріїв виставлення оцінки. [20]

На першому рівні модуль зіставляє відповіді студента з еталонним набором і обчислює підсумковий бал. Результат має бути не лише числовим, а й структурованим: по кожному питанню варто фіксувати, чи була відповідь правильною, скільки балів нараховано, чи відбулося пропущення питання, скільки часу тривало проходження всієї спроби. Це підсилює прозорість і створює основу для подальшого аналізу.

На другому рівні система може формувати короткі аналітичні висновки. Йдеться, наприклад, про виділення тем, у межах яких студент помилявся частіше, визначення частки правильних відповідей за окремими категоріями або формування рекомендацій щодо повторення матеріалу. Такі висновки не мають

підміняти академічне рішення викладача, однак істотно підвищують практичну цінність сервісу.

Якщо надалі система розширюватиметься до перевірки відкритих відповідей, доцільно передбачити окремий контур валідації. Попередня AI-оцінка в такому разі може бути допоміжним орієнтиром, але остаточне підтвердження доцільно залишати за викладачем або за набором чітко визначених рубрик. Саме така модель видається найбільш прийнятною для університетського середовища, де особливого значення набуває справедливість оцінювання.

Результат роботи AI-модуля також може бути зафіксований як частина об’єкта результату. Це означає, що в системі зберігатиметься не тільки фінальний бал, а й деталі його формування, які надалі можна перевірити. Завдяки цьому поєднуються переваги автоматизації та відтворюваності.

2.6. Тестування прототипу та аналіз демонстраційних сценаріїв

Для перевірки працездатності запропонованої моделі доцільно використати кілька послідовних сценаріїв. Перший сценарій стосується створення тесту. Викладач формує набір запитань, після чого система серіалізує тест, зберігає його в IPFS і реєструє отриманий CID. На цьому етапі перевіряється, чи створюється запис у реєстрі та чи можна за зафіксованим ідентифікатором повторно отримати сам тестовий об’єкт. [21]

Другий сценарій пов’язаний зі зміною даних. Якщо в уже створеному тесті змінити хоча б один елемент, наприклад текст питання або правильну відповідь, після повторного збереження система повинна сформувати інший CID. Саме цей результат наочно демонструє контент-адресну природу сховища та підтверджує, що нова редакція не підміняє собою попередню непомітно.

Третій сценарій пов’язаний із завершенням спроби студента. Після надсилання відповідей система виконує оцінювання, формує об’єкт результату, передає його до IPFS і реєструє окремий CID уже для результату. Надалі цей запис

можна перевірити незалежно від інтерфейсу, що підтверджує коректність обраної моделі для збереження підсумків тестування.

Окремо слід врахувати сценарії помилок. Якщо сховище недоступне, система не повинна повідомляти про успішну фіксацію результату. Якщо користувач намагається перевірити неіснуючий CID, має повертатися зрозуміле службове повідомлення. Якщо ж структура об’єкта не відповідає очікуваному формату, backend повинен відхилити запит ще до моменту передавання даних у сховище.

Під час оцінки результатів демонстрації можна зробити кілька узагальнень. Насамперед контент-адресне зберігання справді дає змогу відстежувати версії без складних внутрішніх механізмів аудиту. Крім того, реєстр CID формує достатній мінімум перевірюваності для бакалаврського прототипу. Водночас інтеграція AI-модуля не суперечить цій архітектурі, а органічно її доповнює, оскільки результати обробки також можуть бути формально зафіксовані.

2.7. Обмеження прототипу та напрями подальшого розвитку

Розроблений прототип свідомо має демонстраційний характер. У ньому не реалізовано повний цикл автентифікації, керування ролями, журналювання адміністративних дій та захисту ключів доступу. Такі механізми є необхідними для промислового університетського сервісу, але для бакалаврської роботи важливішим було показати сам принцип перевірюваного зберігання результатів.

Другим обмеженням є використання реєстру CID як моделі незмінного журналу замість повноцінного смарт-контракту. З одного боку, цього достатньо для демонстрації логіки. З іншого — перехід до тестової блокчейн-мережі був би наступним кроком розвитку системи, оскільки дав би незалежне середовище зберігання подій і можливість перевірки транзакцій поза межами локального сервера.

Третє обмеження стосується конфіденційності. У прототипі закладено принцип мінімізації відкритих персональних даних, однак повноцінна система потребує шифрування чутливої інформації, розмежування доступу до розшифрованих результатів і продуманої політики зберігання ключів. Без цього децентралізоване зберігання не можна вважати завершеним з точки зору практичного застосування.

Що стосується AI-компонента, у поточній версії він орієнтований переважно на оцінювання закритих відповідей і побудову базової аналітики. Перспективними є три напрями розвитку: перевірка відкритих відповідей за допомогою NLP-моделей, розширення навчальної аналітики на рівень курсу або групи та впровадження обережних механізмів виявлення аномалій без автоматичного винесення санкцій.

Отже, запропонований прототип не претендує на повністю завершену платформу, але він демонструє життєздатну архітектурну ідею. Саме це робить його придатним для бакалаврського дослідження: система є достатньо простою для пояснення і водночас достатньо змістовною, щоб показати практичну цінність поєднання Web 3.0 та AI в університетському середовищі.

ВИСНОВКИ

У ході дослідження стало очевидно, що перехід від Web 1.0 і Web 2.0 до Web 3.0 для університетського середовища пов’язаний не лише з появою нових цифрових інструментів, а й зі зміною самої логіки роботи з освітніми даними. Для сучасних університетських сервісів важливими стають не тільки функціональність і швидкість доступу, а й можливість перевірити цілісність результатів, простежити історію їх фіксації та зменшити залежність від однієї централізованої платформи. Саме тому принципи Web 3.0 найбільш виправдано застосовувати там, де потрібні прозорість, перевірюваність і контроль версій.

Розгляд поєднання IPFS із реєстрацією CID показав, що така зв’язка може слугувати практичною основою для контент-адресного зберігання тестів і результатів, а також для фіксації конкретної версії даних з подальшою перевіркою її цілісності. Аналіз ролі AI-технологій у системах університетського тестування підтвердив доцільність їх використання передусім для автоматизованого оцінювання, навчальної аналітики, виявлення типових помилок і первинного виявлення нетипових сценаріїв без підміни академічного рішення викладача. Не менш важливим виявився і агентний підхід, оскільки він дає змогу розподілити функції оцінювання, аналізу та перевірки між окремими модулями й зробити систему більш керованою.

Практичним підсумком роботи стало проєктування архітектури MVP-прототипу децентралізованої системи тестування, у якій узгоджено взаємодію frontend, backend, IPFS, реєстру CID та AI-модуля. Демонстраційні сценарії дали змогу показати, як така модель може працювати в університетському середовищі, а також окреслити подальші кроки її розвитку: перенесення реєстру CID у смарт-контракт, посилення механізмів захисту даних і розширення AI-аналітики. Отже, поставленої мети досягнуто, а запропонований підхід можна розглядати як основу для майбутнього розроблення університетських сервісів перевірюваного

тестування, у яких поєднуються прозорість зберігання результатів, можливість їх незалежної перевірки та інструменти інтелектуального аналізу освітніх даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Berners-Lee T. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web. San Francisco : HarperSanFrancisco, 1999. 246 p.

2. O'Reilly T. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. URL: https://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html (дата звернення: 03.02.2026).

3. European Commission. Digital Education Action Plan 2021–2027. URL: https://education.ec.europa.eu/focus-topics/digital-education/action-plan (дата звернення: 05.02.2026).

4. W3C. Decentralized Identifiers (DID) v1.0. URL: https://www.w3.org/TR/did-core/ (дата звернення: 07.02.2026).

5. W3C. Verifiable Credentials Data Model v1.1. URL: https://www.w3.org/TR/vc-data-model-1.1/ (дата звернення: 09.02.2026).

6. Benet J. IPFS – Content Addressed, Versioned, P2P File System. 2014. URL: https://arxiv.org/abs/1407.3561 (дата звернення: 11.02.2026).

7. IPFS Docs. Content Identifiers (CIDs). URL: https://docs.ipfs.tech/concepts/content-addressing/ (дата звернення: 13.02.2026).

8. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. URL: https://bitcoin.org/bitcoin.pdf (дата звернення: 15.02.2026).

9. Tapscott D., Tapscott A. Blockchain Revolution: How the Technology Behind Bitcoin and Other Cryptocurrencies Is Changing the World. New York : Portfolio, 2016. 368 p.

10. IPFS Docs. Address IPFS on the Web. URL: https://docs.ipfs.tech/how-to/address-ipfs-on-web/ (дата звернення: 18.02.2026).

11. Pinata Docs. Uploading JSON to IPFS. URL: https://docs.pinata.cloud/ (дата звернення: 20.02.2026).

12. Russell S., Norvig P. Artificial Intelligence: A Modern Approach. 4th ed. Harlow : Pearson, 2021. 1232 p.

13. UNESCO. Guidance for Generative AI in Education and Research. Paris : UNESCO, 2023. URL: https://unesdoc.unesco.org/ark:/48223/pf0000386693 (дата звернення: 24.02.2026).

14. Siemens G. Learning Analytics: The Emergence of a Discipline. American Behavioral Scientist. 2013. Vol. 57, No. 10. P. 1380–1400.

15. Wooldridge M. An Introduction to MultiAgent Systems. 2nd ed. Chichester : Wiley, 2013. 484 p.

16. Franklin S., Graesser A. Is It an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. In: Intelligent Agents III, Agent Theories, Architectures, and Languages. Berlin ; Heidelberg : Springer, 1997. P. 21–35.

17. Bray T. The JavaScript Object Notation (JSON) Data Interchange Format. RFC 8259. 2017. URL: https://www.rfc-editor.org/rfc/rfc8259 (дата звернення: 05.03.2026).

18. Node.js Documentation. Crypto. URL: https://nodejs.org/api/crypto.html (дата звернення: 10.03.2026).

19. React Documentation. Learn React. URL: https://react.dev/learn (дата звернення: 12.03.2026).

20. Express Documentation. Routing Guide. URL: https://expressjs.com/en/guide/routing.html (дата звернення: 16.03.2026).

21. Vite Documentation. Getting Started. URL: https://vite.dev/guide/ (дата звернення: 19.03.2026).

ДОДАТКИ

Додаток А. Інтерфейс створення тесту у прототипі

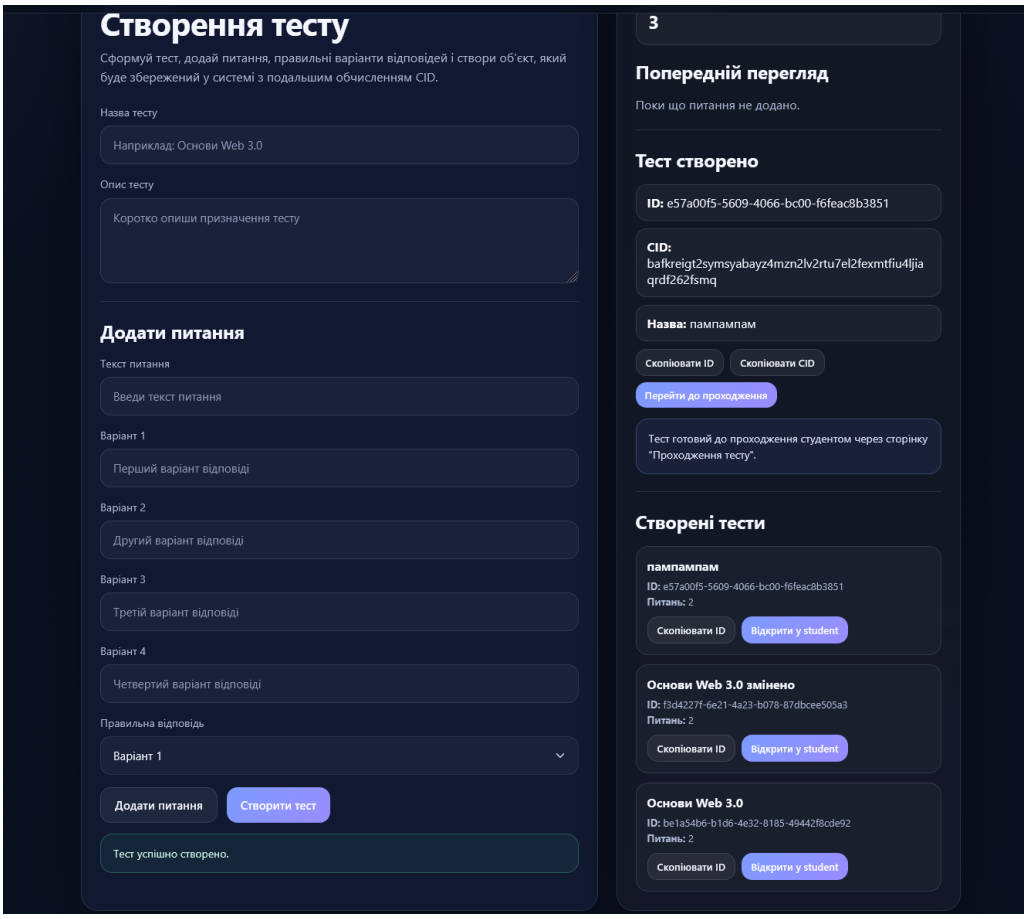


Рисунок А.1 - Інтерфейс створення тесту та відображення CID, IPFS URL і blockchain-параметрів

Додаток Б. Інтерфейс проходження тесту та перевірки достовірності

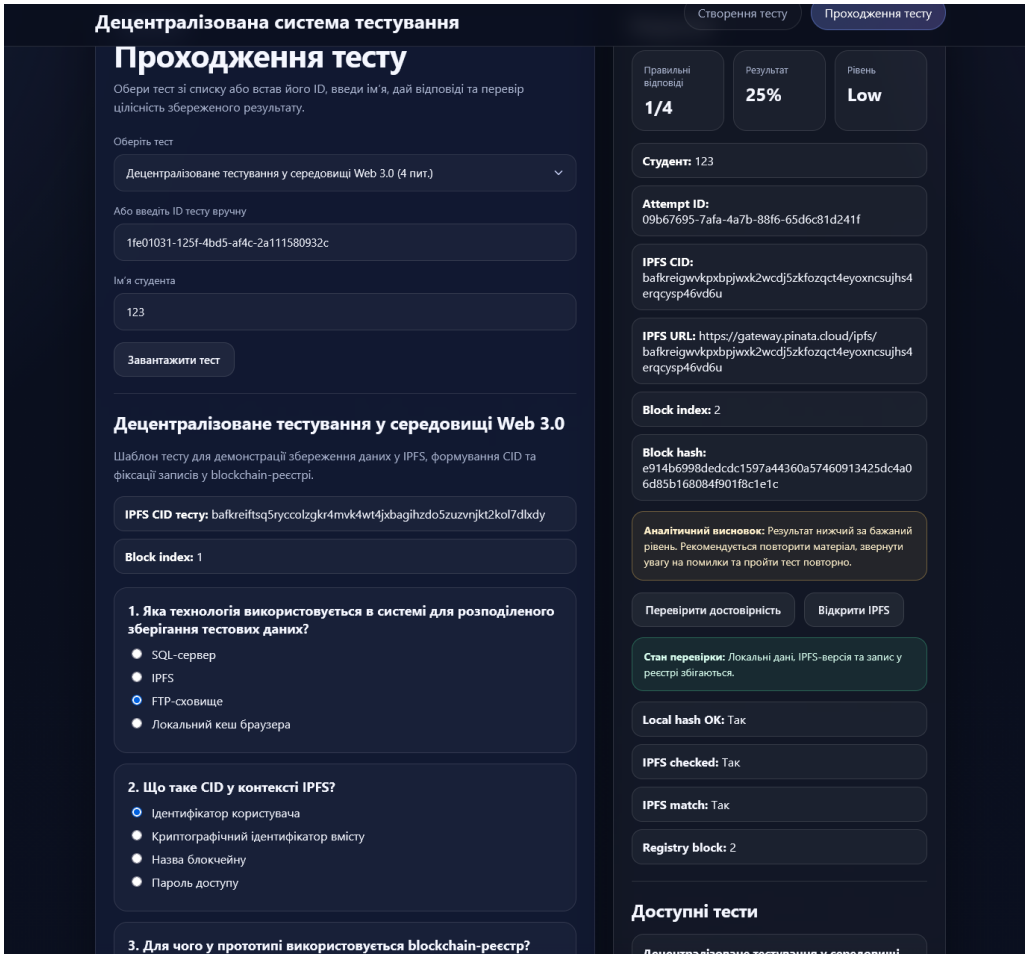


Рисунок Б.1 - Інтерфейс проходження тесту, результати оцінювання та перевірка цілісності збережених даних