Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Бакалаврська кваліфікаційна робота

на тему: «Розробка методу і програмних засобів вебплатформи з постквантовим шифруванням файлів та стійким сховищем до атак»

Виконала: студентка IV курсу
групи 1ПІ-21б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Шиндирук В. Д.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. ПЗ Войтко В. В.
(прізвище та ініціали)

Рецензент: к.т.н., доцент каф. ОТ Городецька О. С.
(прізвище та ініціали)

Допущено до захисту

Зав. кафедри ПЗ Романюк О.Н.
«09» 06 2025 р.

ВНТУ – 2025

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
О. Н. Романюк
«21» березня 2025 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Шиндирук Вікторії Дмитрівні

1. Тема роботи – розробка методу і програмних засобів вебплатформи з постквантовим шифруванням файлів та стійким сховищем до атак.

Керівник роботи: Войтко Вікторія Володимирівна, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від 20 березня 2025 р. №97.

2. Строк подання студентом роботи: 2 червня 2025.

3. Вихідні дані до роботи: середовище розробки – Visual Studio Code, мова розробки – TypeScript; фреймворки – Node.js, Next.js, PostgreSQL; технології безпеки JWT, TOTP, AES, ML-KEM; операційна система – Windows 8/10/11;

4. Зміст текстової частини: вступ, аналіз стану розвитку програм з квантово захищеним сховищем та постановка задач розробки, розробка методу, моделі та алгоритмів роботи програми, розробка програмного забезпечення, тестування програми, висновки, список використаних джерел, додатки, ілюстративна частина.

5. Перелік ілюстративного матеріалу: титульний слайд – автор, науковий керівник бакалаврської кваліфікаційної роботи, тема; актуальність розробки; мета, об'єкт та предмет дослідження; задачі розробки; наукова новизна; порівняльні характеристики аналогів; алгоритм взаємодії клієнта та сервера, схема архітектури платформи; діаграма компонентів платформи; архітектура роботи квантової моделі машинного навчання; принцип роботи квантового генератора; тестування моделі машинного на виявлення аномалій; апробація та публікація результатів роботи; висновки; фінальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1-4	Войтко В. В., к.т.н., доцент кафедри ПЗ	24.03.25	01.06.25

7. Дата видачі завдання 24 березня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз стану розвитку програм з квантово захищеним сховищем	25.03.25 – 01.04.25	Виконано
2	Розробка методів, моделі та алгоритмів роботи програми	02.04.25 – 14.04.25	Виконано
3	Розробка програмного забезпечення	16.04.25 – 03.05.25	Виконано
4	Тестування програми	06.05.25 – 17.05.25	Виконано
5	Оформлення матеріалів до захисту БКР	23.05.25 – 30.05.25	Виконано

Студент _____ **Шиндирук В. Д**
(підпис) (ініціали та прізвище)

Керівник бакалаврської кваліфікаційної роботи _____ **Войтко В. В.**
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота містить 82 сторінки формату А4, на яких є 50 рисунків і 4 таблиці, а список використаних джерел складається з 27 найменувань.

У бакалаврській кваліфікаційній роботі проведено аналіз стану і розвитку програмних застосунків з квантовим захистом даних. Було проведено аналіз аналогів, визначено їх переваги і недоліки та доведено доцільність розробки власного програмного застосунку.

У роботі обґрунтовано вибір мови програмування, середовища розробки та технологій, що використовувалися під час створення середовища. Було розроблено програмний продукт – захищену вебплатформу за допомогою квантово стійких ключів.

Отримані результати бакалаврської кваліфікаційної роботи можуть бути застосовані у компаніяї малого, середнього, велиго бізнесу, а також військових, медичних установ із квантово стійким захищемим сховищем.

Ключові слова: постквантове шифрування, вебплатформа, квантове машинне навчання, розподіл ключів, виявлення аномалій, алгоритм.

ABSTRACT

The bachelor's qualification thesis consists of 82 A4-sized pages, containing 50 figures and 4 tables, with a list of references comprising 27 sources.

The thesis presents an analysis of the state and development of software applications utilizing quantum data protection. A review of existing solutions was conducted, their advantages and disadvantages were identified, and the necessity of developing a proprietary software application was substantiated.

The thesis justifies the choice of programming language, development environment, and technologies used during the creation process. As a result, a software product was developed – a secure web platform using quantum-resistant keys.

The outcomes of this bachelor's qualification thesis may be applied in small, medium, and large businesses, as well as in military and medical institutions requiring quantum-resistant secure storage solutions.

Keywords: post-quantum encryption, web platform, quantum machine learning, key distribution, anomaly detection, algorithm.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Швидка інтеграція квантової теорії у життя стала однією з вагомих причин звернути увагу на безпеку корпоративних даних у документах. Державні та медичні установи, малий, середній та великий бізнес, військові структури можуть бути під загрозою, у зв'язку з посиленням інтересу науковців у створенні квантового комп'ютера, який, у свою чергу, може бути інструментом у отриманні конфіденційних даних користувачів з корпоративних сховищ за допомогою розшифрування захисних ключів передбачуваними способами.

Існуючі успішні методи з шифрування даних базуються на криптографічних алгоритмах RSA та ECC, що при спробі атаки квантовим комп'ютером будуть з легкістю отримані [1].

Захист корпоративних файлів та середовища від несанкціонованого доступу за допомогою постквантових алгоритмів шифрування є актуальним у зв'язку з швидким розвитком кіберзлочинності, цікавості людства у використанні квантових теорій у житті та активною роботою науковців над розширенням можливостей квантового комп'ютера, до прикладу, виконувати не лише розв'язання надвеликих людській свідомості кількості математичних операцій за лічені секунди, а й виконувати дії, ідентичні персональному комп'ютеру, але з більшою швидкістю та точністю.

Захист документів та платформи постквантовими алгоритмами, можливість контролю поведінки користувачів у системі за допомогою квантової моделі машинного навчання чи формування розпізнаного тексту на фото у захищений документ застосовується дуже рідко і є необхідними для великих установ, бізнесів чи державних структур для їх функціонування.

Вебплатформа є перспективною, оскільки містить в собі важливі компоненти для високого захисту середовища, необхідного великій кількості бізнесів, чиї дані можуть бути перехвачені зловмисним проникненням.

Зв’язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Метою роботи є підвищення рівня захищеності критично важливих структур від ураження квантовими комп’ютерами шляхом розробки модулів системи, що підвищить захист передачі даних за рахунок змагальної квантової нейронної мережі.

Відповідно до поставленої мети потрібно вирішити такі завдання:

– розробити шифрування файлів за допомогою ключів, захищених постквантово стійким алгоритмом;

– розробити підсистему з розпізнаванням рукописного тексту та його подальшою конвертацією у вихідний захищений файл;

– розробити захищене сховище для зберігання і обміну ключами між групами користувачів;

– розробити підсистему з квантовою змагальною навчальною нейромережею для виявлення аномалій у поведінці користувачів;

– розробити ступені доступу та пріоритетності у важливості файлів для їх автовидалення з системи через конкретний проміжок часу;

– провести тестування системи у всіх можливих варіантах використання.

Об'єктом дослідження виступає процес безпечної роботи з файлами на платформі, їх зберігання, шифрування, обмін між групами, технології постквантового захисту від атак квантовими комп’ютерами, процес розпізнавання тексту на фото та конвертації його у вихідний файл, процес розробки квантової моделі машинного навчання для виявлення аномалій у поведінці користувачів.

Предметом дослідження є методи та засоби розробки вебплатформи для шифрування, зберігання та обміну файлами у системі, захищеної постквантовими алгоритмами, з використанням навченої змагальної квантово стійкої машинної моделі для виявлення аномалій у поведінці користувачів.

Методи дослідження. У процесі дослідження використовувалися такі методи:

– методи і технології модульної розробки вебплатформи для програмної реалізації модулів системи;

– методи шифрування файлів за допомогою ключів, захищених квантово стійким алгоритмом, для реалізації захисту даних постквантовим шифруванням;

– методи візуалізації для візуалізації даних розроблюваних модулів вебсистеми;

– методи UI/UX для створення та перевірки інтерфейсів вебсистеми;

– методи тестування для перевірки працездатності роботи програми.

Новизна отриманих результатів.

1. Дістав подальшого розвитку метод захисту середовища від несанкціонованого доступу до платформи, який, на відміну від існуючих, використовує розроблену квантову модель машинного навчання для виявлення аномальної поведінки, що виявляє порушення сигналів у системі з достатньою кількістю даних для можливості навчити модель бути стійкішою до таких шумів у сигналах у майбутньому і забезпечує захист ресурсів від атак.

2. Подальшого розвитку отримала квантова модель виявлення аномальної поведінки, яка, на відміну від існуючих, дозволяє встановлювати логічні групи користувачів з різними рівнями доступу, що гарантує сегрегацію даних між групами та спрощує управління системою. Доступ до функцій платформи полегшується за допомогою токенів JWT, тоді як автентифікація користувачів

виконується через MFA, при тому, що кожен користувач має свій персональний TOTP у системі.

Практичне значення проєкту полягає у можливості використання створеної високозахищеної вебсистеми, що дозволяє забезпечити конфіденційність, цілісність та доступність інформації в умовах постквантових загроз. Розроблена вебплатформа «Quantum Information Science» базується на використанні сучасних засобів криптографічного захисту, зокрема постквантового шифрування ML-KEM, симетричного шифрування AES, механізмів двофакторної автентифікації та токенів доступу, що дає змогу ефективно захищати чутливу інформацію незалежно від її природи чи джерела.

Особистий внесок здобувача. Усі результати, подані в бакалаврській кваліфікаційній роботі, були отримані автором особисто. У науковій праці [6] опублікованій у співавторстві, автору належать такі результати: розробка квантової моделі машинного навчання, UML діаграма застосування взаємодії клієнта і сервера для передачі ключа, захищеного постквантовим алгоритмом шифрування.

Апробація та публікація результатів проєкту. Результати бакалаврської кваліфікаційної роботи доповідалися та обговорювалися на конференціях: Всеукраїнська науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи (МН-2024)» (Вінниця, 2024), Всеукраїнська науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи (МН-2025)» (Вінниця, 2025).

Публікації. Результати роботи опубліковані в двох наукових працях – тезах-доповідях на Всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2024)» (Вінниця, 2024), та Всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2025)» (Вінниця, 2025).

Аналіз. У пояснювальній записці до бакалаврської кваліфікаційної роботи було розглянуто чотири розділи та було використано 27 літературних джерел.

1 АНАЛІЗ СТАНУ РОЗВИТКУ ПРОГРАМ З ПОСТКВАНТОВИМ ЗАХИСТОМ ТА ПОСТАНОВКА ЗАДАЧ РОЗРОБКИ

1.1 Аналіз розвитку квантових технологій

Станом на сьогоднішній день квантова криптографія поступово набуває значущість у питанні кібербезпеки. Кількість атак на бізнеси, державні установи, секретні служби значно збільшилась. Витоки конфіденційних даних несуть велику шкоду існуванню структур та загрозу при використанні здобутої інформації не за призначенням.

Існуючі технології криптографії, вже не забезпечують безпеку в умовах квантових атак. У результаті, з кожним роком зростає ймовірність того, що сьогоднішні методи захисту стануть недостатніми для забезпечення довгострокової безпеки даних [1].

Питання квантових технологій є новітнім і не широко впровадженим. Станом на 2025 рік [2], в умовах постійних кібератак на критично важливі структури, з ціллю пошкодження хмарних сховищ для витягнення конфіденційної інформації, є велика необхідність підвищити захист до квантово стійкого для підвищення непроникності. Зокрема, це необхідно і для посилення політик конфіденційності, покращення довіри між клієнтом на бізнесом. Щодо державних, медичних та військових структур впровадження квантового рівню захисту є невід’ємним для їх захищеності від перехвату ключів, сигналів, несанкціонованих проникнень.

Для підвищення захищеності програми на міжнародному рівні дієвим рішенням є впровадження квантової машинної моделі на основі змагальної мережі, яка самостійно навчатиметься протидіяти будь-яким видам існуючих атак, зокрема, з боку квантового комп’ютера. Це забезпечить сховище від зловмисного проникнення, а порушення цілісності сигналів передачі одразу фіксуватимуться і

передаватимуться на графік, де можна буде відслідкувати з достатньою кількістю даних пристрій, з якого відбулось втручання.

Відсутність досконалого модуля для розпізнавання рукописного вводу на фото та конвертацією його у захищений квантовим методом ключем також наштовхнула на розробку інструменту, що полегшить процес оцифрування документів швидко, захищено, з можливістю збереження безпосередньо у сховищі вебплатформи.

Така програма є перспективною, оскільки містить у собі одну захищену систему з важливими і корисними модулями для шифрування, конвертування і зберігання файлів, захищених від квантових атак. Сам тому було вирішено розробити власну вебсистему захищеного простору для використання всіх типів бізнесів.

1.2 Аналіз аналогів розроблюваного програмного застосунку

На сьогоднішній момент на ринку квантових технологій є кілька великих і аналогів розроблюваної системи, що вже були реалізовані провідними світовими технологічними компаніями. Серед них слід виділити Amazon Web Services, Google і IBM, всі з яких створюють власні інфраструктури та платформи для квантових обчислень.

Amazon AWS через свою платформу Bracket як і надає доступ до квантового комп’ютера, так і розвиває елементи квантового захисту на серверах (рисунок 1.1).

Braket є складовою частиною хмарної екосистеми компанії, що пропонує безперебійну інтеграцію з традиційними обчислювальними ресурсами Amazon. Це дає можливість користувачам розробляти, перевіряти та розширювати гібридні квантово-класичні алгоритми, що є важливим напрямом досліджень і розробок у сфері квантового машинного навчання та квантової оптимізації (рисунок 1.1) [3].

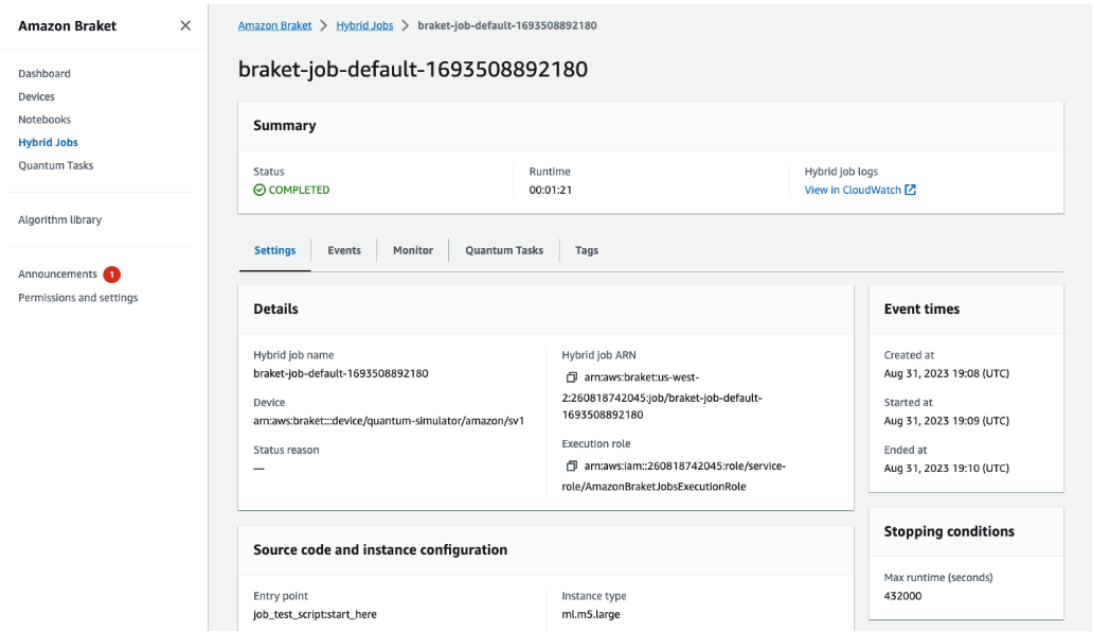


Рисунок 1.1 – Інтерфейс платформи Bracket компанії Amazon AWS

Google Cloud спільно зі своєю платформою Quantum AI впроваджує криптографічні схеми для захисту даних на світовому масштабі для впливу на квантові обчислення у майбутньому (рисунок 1.2). Також компанія бере активну участь у наукових дослідженнях у цьому напрямку [4].

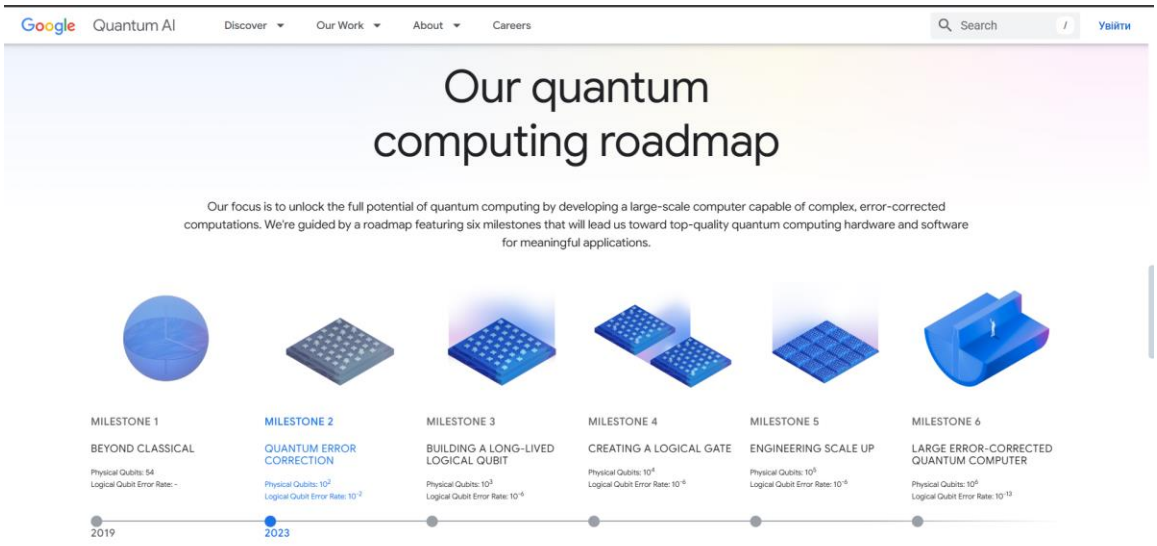


Рисунок 1.2 – Головна сторінка платформи Quantum AI

IBM також має свій проект IBM Quantum Platform який у свою чергу виділяється серед інших тим, що дає доступ користувачам до симуляторів квантових комп'ютерів та займається стандартизацією квантово стійких алгоритмів.

На сьогоднішній день компанія також активно підтримує проекти, спрямовані на моделювання криптографічних протоколів, які повинні забезпечити захист інформаційних систем в умовах виникнення квантових атак. До таких ініціатив відносять розробку та тестування механізмів шифрування, що стійкі до квантових обчислень, оцінку їхньої продуктивності за умов обмежених обчислень, а також вивчення можливостей інтеграції цих рішень у існуючі інформаційні інфраструктури (рисунок 1.3) [5].

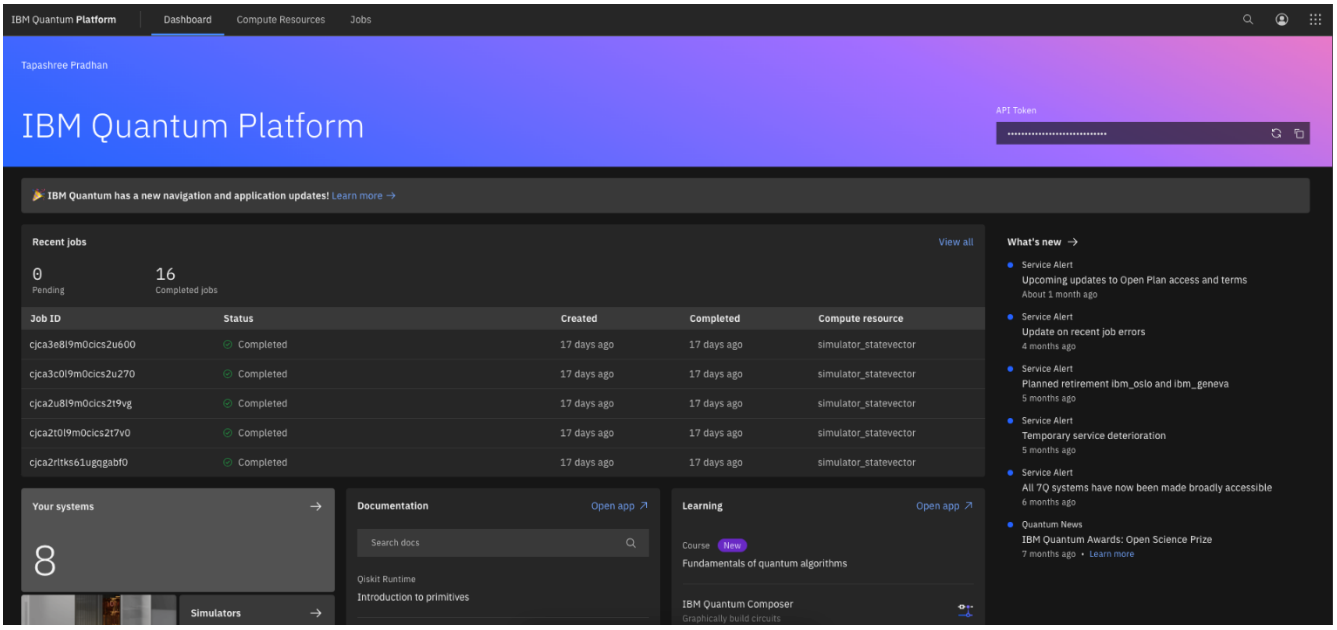


Рисунок 1.3 – Інтерфейс платформи IBM Quantum platform

Проаналізувавши популярні аналоги, визначимо їх можливості та недоліки, сильні й слабкі сторони, які будуть враховані при створенні вебплатформи

«Quantum Information Science». Результати порівняльного аналізу аналогів зведено в таблицю 1.1.

Таблиця 1.1 – Порівняльні характеристики аналогів

Критерії	AWS Bracket	Google Quantum AI	IBM Quantum Platform	Quantum Information Science
Інтеграція в мережеві протоколи	+	+	+	+
Використання постквантових алгоритмів	-	+	-	+
Квантовий розподіл ключів	-	-	+	+
Моделі машинного навчання для кіберзахисту	-	-	-	+
Самонавчальна система для криптозахисту	-	-	-	+
Загальна оцінка	20%	40%	40%	100%

Таблиця порівняльних характеристик (табл 1.1) показала, що розробка власної системи захисту від несанкціонованого доступу з використанням квантових комп’ютерів є доцільною. В результаті буде отримано продукт, який покриває недоліки існуючих програмних рішень, дасть можливість бізнесам та установам захищати усі свої дані з урахуванням активного розвитку квантових технологій.

1.3 Аналіз методів розв’язання задачі

Розробка вебплатформи з квантово стійким сховищем вимагає уважного підходу до вибору технічних рішень, щоб ефективно вирішувати потреби

користувачів у системі. Існує кілька стратегій, кожна з яких має свої переваги та обмеження, впливаючи на загальну продуктивність і адаптацію платформи.

Одна зі стратегій розв'язання задачі побудови криптостійкої вебплатформи полягає у використанні модульної архітектури з постквантовим шифруванням та багаторівневим управлінням доступом. Такий підхід передбачає інтеграцію незалежних модулів: окремий компонент для шифрування трафіку з використанням алгоритму ML-KEM, підсистема управління рівнями доступу до файлів, а також сховище для обміну криптографічними ключами між користувачами. Такий підхід забезпечує високу гнучкість, можливість адаптації до різних систем і повторного використання модулів, але може вимагати додаткових зусиль на етапі інтеграції компонентів та налаштування взаємодії між ними, особливо в складних організаційних середовищах.

Інша стратегія передбачає створення централізованої вебплатформи з інтегрованими засобами захисту, комп'ютерного зору та інтелектуального аналізу поведінки користувачів. У межах цього підходу пропонується єдине середовище, яке об'єднує в собі постквантову криптографію, систему виявлення аномалій на основі самонавчальної нейронної мережі та механізми автоматизованого розпізнавання вмісту файлів через Tesseract. Такий підхід забезпечує максимальну цілісність, контроль над безпекою всієї платформи та спрощує підтримку, але може вимагати значних інвестицій у розробку, тестування та масштабування, зокрема в умовах зростання кількості користувачів і обсягу даних.

Ще одна стратегія полягає у створенні захищеного сервісу для обміну файлами між групами користувачів з гнучким розподілом прав і пріоритетів, що може бути вбудований у більші системи як додатковий функціональний блок. Такий сервіс використовує технології JWT, AES, ML-KEM, TOTP та дозволяє легко забезпечити захищену взаємодію між установами та агентствами. Такий метод забезпечує зниження навантаження на основну систему, спрощення інтеграції та

можливість використання як готового продукту, але може вимагати обмеження функціональності у порівнянні з повноцінною платформою та створення окремих механізмів сумісності для кожного типу користувача.

Після аналізу переваг та недоліків кожного з підходів було вирішено обрати стратегію розробки вебплатформи з квантово стійким сховищем, яка є найбільш доцільною стратегією з-поміж описаних вище методів. Обрана архітектура дозволяє ефективно поєднати постквантову криптографію, багаторівневе управління доступом, поглиблений аналіз дій користувачів і розпізнавання вмісту файлів, у єдине інтегроване середовище. Це забезпечує цілісність системи, масштабованість, зручність підтримки та адаптацію до зростаючих вимог бізнесу й державних установ, у зв’язку з впровадженням низки законопроектів.

1.4 Постановка задач дослідження

Після детального аналізу існуючих платформ з розвитком квантової криптографії було визначено задачі, які необхідно виконати в бакалаврській кваліфікаційній роботі:

– удосконалити існуючу змагальну машинну модель для виявлення аномальної поведінки у вебсистемі шляхом збільшення кількості даних про спробу отримати несанкціонованим шляхом доступ до файлів системи для уникнення таких випадків у майбутньому;

– розробити модуль для шифрування ключів до документів постквантовим алгоритмом;

– розробити удосконалену машинну модель для розпізнавання тексту на фото та подальшою конвертацією у вихідний захищений файл;

– розробити захищене сховище для зберігання файлів з розподілом документів по пріоритетах, можливістю керування групами та її учасниками з постквантовим захистом;

– розробити модуль для управління процесами у користувацькому середовищі;

– провести тестування вебплатформи.

Виконання перелічених завдань у повному обсязі може продемонструвати можливості розроблених модулів програмної системи, а також покаже її перспективність використання у системах захисту програмних застосунків.

Технічне завдання на розробку наведено в додатку А.

1.5 Висновки

У першому розіді розділі було розглянуто актуальність проблеми захисту файлів та середовища в умовах стрімкого квантового розвитку.

Проаналізовано перспективи вирішення актуального питання шляхом розгляду аналогів, таких як «Amazon Bracket», «Quantum AI», «IBM Quantum» визначенням їх сильних та слабких сторін, а саме, можливість використання постквантових алгоритмів для розподілу ключів, використання квантової моделі машинного навчання для виявлення аномальної поведінки користувачів у системі, використання комп’ютерного зору для розробки захищених документів з тексту на фото, наявність самонавчальної системи для криптозахисту, можливість інтеграції в мережеві протоколи.

В результаті порівняння було обґрунтовано доцільність розробки програмного продукту. З огляду на недоліки аналогів для подальшої розробки було визначено основні завдання, які необхідно розв’язати. Такі задачі враховують недоліки розглянутих аналогів, включають розробку модулів для зручного користування платформою, модуль проектування зрозумілого інтерфейсу, розробку UI\UX дизайну, а також проведення тестування отриманої програми.

2 РОЗРОБКА МЕТОДІВ, МОДЕЛЕЙ ТА АЛГОРИТМІВ РОБОТИ СИСТЕМИ

2.1 Аналіз вхідних даних системи

Початковим етапом взаємодії з системою є процес автентифікації, під час якого користувач вводить свої реєстраційні дані, такі як ім’я користувача, електронну пошту або пароль. Ці дані є вхідними параметрами, які система порівнює з наявною базою даних облікових записів. У разі збігу ідентифікаторів та паролів, модуль авторизації визначає роль користувача в системі та обсяг доступу до її функціоналу.

Як результат, вихідними даними цього модуля є створення захищеного сеансу користувача, надання йому персонального доступу до визначених ресурсів та відображення інтерфейсу, відповідного до його ролі. Наприклад, учасник, адміністратор або власник групи. Невдалі спроби входу фіксуються у системному журналі для подальшого аналізу.

Після автентифікації користувач отримує змогу створювати нові групи або приєднуватися до вже існуючих. На вхід модуль отримує команду користувача, що вказує на створення групи, запрошення учасників або зміну ролей. Вхідними даними є ідентифікатори запрошених користувачів, встановлені рівні доступу, назви груп або повідомлення-запрошення.

У результаті обробки цих даних, модуль формує логічну структуру груп, встановлює правила взаємодії між учасниками, надає доступ до відповідних ресурсів та фіксує інформацію про ролі. Вихідними даними є оновлені групові таблиці доступу, список членів із правами та інформація про їхні дії, яка далі надходить до модуля моніторингу.

Функціональність модуля роботи з файлами полягає у забезпеченні конфіденційної обробки даних користувача. На вхід надходить файл, завантажений

через інтерфейс системи, а також обрана користувачем операція – зашифрувати, розшифрувати або переглянути. Система автоматично застосовує постквантовий криптографічний алгоритм до вхідного файлу, шифрує його в середовищі, що не допускає витоку ключів, і зберігає у захищеному вигляді.

Вихідними даними є збережений зашифрований файл, який доступний лише тим учасникам групи, які мають відповідні права. Під час дешифрування вхідними даними є запит користувача та його особистий ключ, а вихідними – доступний для перегляду оригінальний файл.

Забезпечення зручної навігації серед численних файлів і документів покладено на модуль управління файловою структурою. До вхідних параметрів належать дії користувача, такі як створення нової папки, переміщення файлу, перейменування, видалення або зміна структури дерева каталогів.

Вихідними даними виступає оновлена файлово-папкова ієрархія, яка миттєво відображається для всіх учасників групи з відповідними правами доступу. Цей модуль також взаємодіє з іншими частинами системи, надаючи дані для логування дій і подальшого аналізу активності.

Усі події, що відбуваються в системі, включаючи вхід, вихід, завантаження файлів, зміну прав доступу, шифрування або перегляд, надходять до спеціального модуля реєстрації дій. Вхідними даними для нього є метаінформація про кожну дію: ідентифікатор користувача, час, тип операції, ID ресурсу та результат виконання.

Вихідною інформацією є детальний лог, що зберігається у базі даних журналів подій. Ці дані надалі використовуються не лише для перегляду історії активності, а й для поведінкового аналізу. Також журнал доступний для адміністраторів у спеціальному інтерфейсі спостереження.

Однією з найскладніших частин системи є аналітичний модуль, який містить машинне навчання[6] із квантовими обчислювальними моделями. На вхід цей модуль отримує накопичені журнали активності користувачів, а також історичні

патерни їхньої поведінки, що раніше були класифіковані як звичні. У процесі обробки здійснюється порівняння поточних дій із типовими шаблонами. Якщо модель виявляє істотні відхилення – наприклад, незвичну годину доступу, надмірну кількість шифрувань за короткий період, або зміну IP-адреси – це інтерпретується як потенційна загроза.

Вихідними даними цього модуля є сигнали, що надходить до адміністратора та автоматична реакція системи у вигляді обмеження доступу або запиту на повторну верифікацію особи.

2.2 Розробка моделі реляційної бази даних системи

Вебплатформа «Quantum Information Science » містить в собі зручний UI\UX дизайн, задані структуризовані дані, моделі, алгоритми, інтерфейси, що є небхідними для реалізації квантових процесів, обробки інформації, візуалізації графіків, зведення результатів у зручний для аналізу спосіб. За допомогою цього реалізовується.

Система орієнтована на створення високозахищеного середовища, орієнтованого на потреби бізнесу та критичних структур. Основне завдання платформи полягає у забезпеченні стійкості до квантових атак документів, файлів, сховища та груп.

Подана на рисунку 2.1 блок-схема є систематичним представленням реляційної бази даних, розробленої для створення безпечної системи зберігання та управління файлами користувачів, включаючи положення щодо групового доступу, багаторівневої аутентифікації та регулювання сеансів. Рамка моделі складається з сутностей, кожен з яких виконує конкретну функцію в системі.

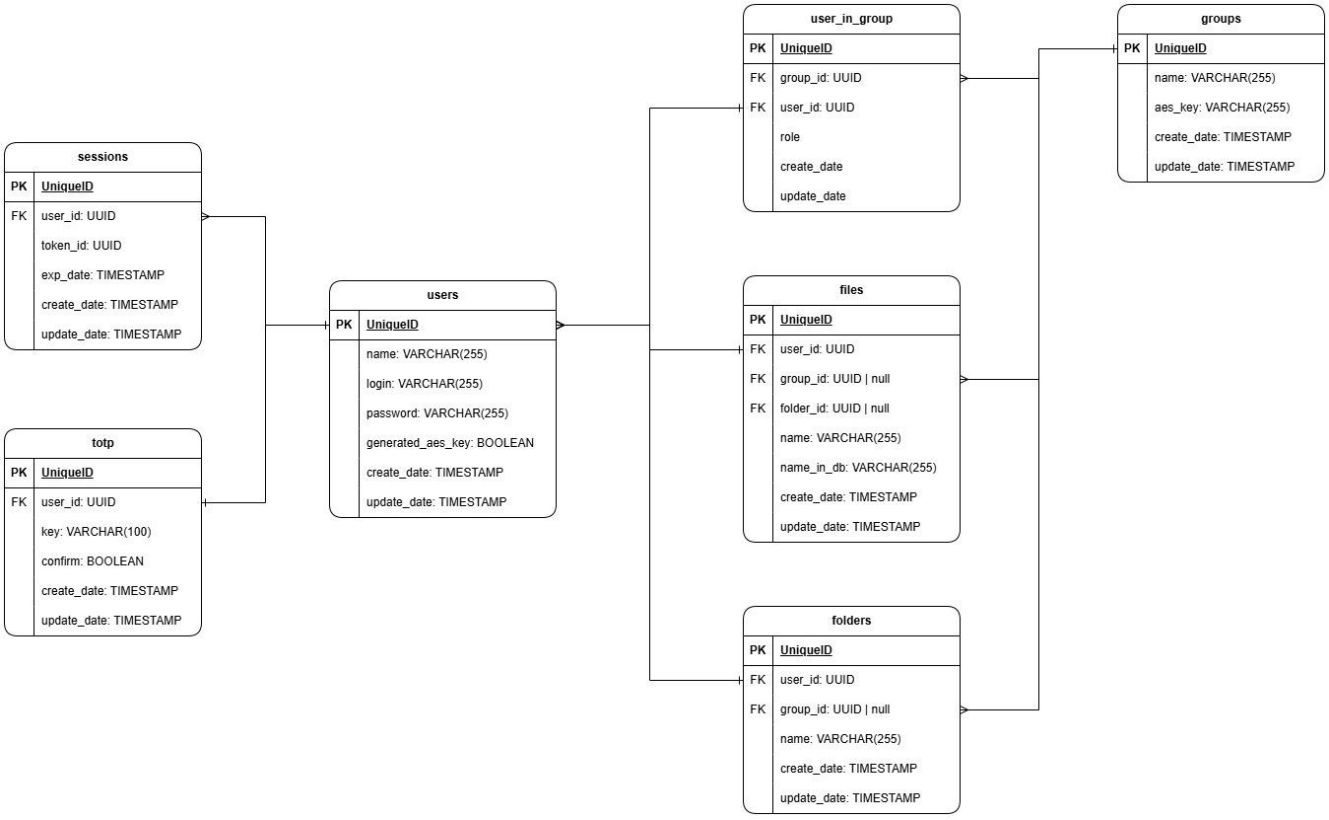


Рисунок 2.1 – Діаграма «сутність – зв'язок» проєктування бази даних

Основним елементом системи є таблиця користувачів, яка охоплює основні дані, що стосуються кожного окремого користувача. Кожен запис відрізняється унікальним ідентифікатором типу UUID. Таблиця містить такі атрибути: ім'я користувача, логін, захищений пароль та логічне поле – generated_aes_key, що позначає, чи створено окремий криптографічний ключ для шифрування даних. Крім того, атрибути включені для документування дати створення та останнього оновлення запису – create_date, update_date.

Для полегшення механізму сеансу для авторизованих користувачів використовується таблиця сеансів [7]. Вона складається з полів token_id – первинний ключ), user_id – зовнішній ключ, що посилаються на таблицю користувачів, exp_date – час закінчення сеансу, а також технічних полів –

create_date та update_date. Ця таблиця гарантує збереження інформації як щодо активних, так і щодо завершених сесій, що дуже важливою у захисті доступу.

Додатковий рівень безпеки надається через таблицю totp, яка оновлює функціональність TOTP – одноразовий пароль на основі часу. Кожен запис містить user_id – служить як первинним, так і зовнішнім ключем, key – секретний ключ, який використовується для генерації OTP, логічне поле підтвердження, яке реєструє перевірку конфігурації TOTP, і відповідні технічні мітки часу.

Також система вміщує механізм агрегації користувачів у групи через таблицю груп. Кожна група має унікальний ідентифікатор, позначене ім'я, криптографічний AES aes_key, що полегшує обмін зашифрованими даними, крім полів для запису створення та оновлення запису.

Для ілюстрації асоціації «багато до багатьох» між користувачами та групами створено таблицю user_in_group. Він містить складений первинний ключ, утворений полями group_id та user_id, які є зовнішніми ключами, що посилаються на відповідні таблиці. Крім того, є поле ролі для розмежування функції користувача в групі, наприклад, учасника або адміністратора, супроводжуване полями для документування дат створення та оновлення запису.

Користувачам надається можливість створювати особисті папки, які каталогізуються в таблиці папок. Унікальний ідентифікатор id, сторонні ключі, що відносяться до user_id – власник папки і необов'язково group_id, у випадку якщо папка є спільною, а також призначене ім'я складають основний вміст таблиці. Таке розташування дозволяє організувати файли в деревовидній структурі каталогів.

Таблиця файлів зберігає інформацію про завантажені файли. Він містить id – основний ключ, user_id – власник, folder_id – позначає папку, до якої належить файл, ім'я файлу та два технічні поля name_iv та name_hmac, що використовуються для шифрування – вектор ініціалізації у базі даних та перевірки цілісності даних –

HMAC. Таблиця також містить поля для реєстрації створення та оновлення часових міток запису.

Уся архітектура бази даних побудована на принципах безпеки, масштабованості та підтримки багатокористувацької взаємодії. Вона дозволяє реалізувати функціональність персональних сховищ, спільного доступу до ресурсів, контролю автентифікації та авторизації, а також зберігання зашифрованих даних.

2.3 Розробка архітектури взаємодії клієнта з серверною частиною

Питання постквантової захищеності набуває актуальності при розробці вебплатформ для користувачів усіх типів корпоративних бізнесів, що містять засекречені дані, дослідження, річні стратегічні плани, обрахунки та витрати на закупівлю чи продаж, що не повинні ніяким чином потрапити у руки зловмисника. Мова йде про захист стратегічно важливих даних, які включають внутрішні дослідження, заявки на патенти, стратегічні плани розвитку, детальні фінансові звіти, а також інформацію про закупівлі, угоди, ринкові операції та буюджет компанії. Втрати чи перехоплення подібної інформації можуть суттєво вплинути не лише на конкретну організацію, а й на загальну економічну стабільність країни

Найбільш актуальним на сьогоднішній день є квантове розприділення ключів між користувачами системи. Це допомагає колегам обмінюватись ключами без будь-яких можливих способів втручань в процес. Якщо відбудеться спроба перехоплення квантового сигналу, відповідні зміни будуть у двох користувачів, що дасть їм зрозуміти що відбулась спроба несанкціонованого доступу файлу.

Нижче, на рисунку 2.2, наведено діаграму використання [8], що ілюструє взаємодію клієнта і сервера у передачі ключа з метою шифрування файлу. Ця архітектура гарантує безпечний обмін криптографічними ключами між клієнтом і

сервером за допомогою постквантового алгоритму. Як ілюструє UML-діаграма застосування, користувач розпочинає запит на шифрування документа.

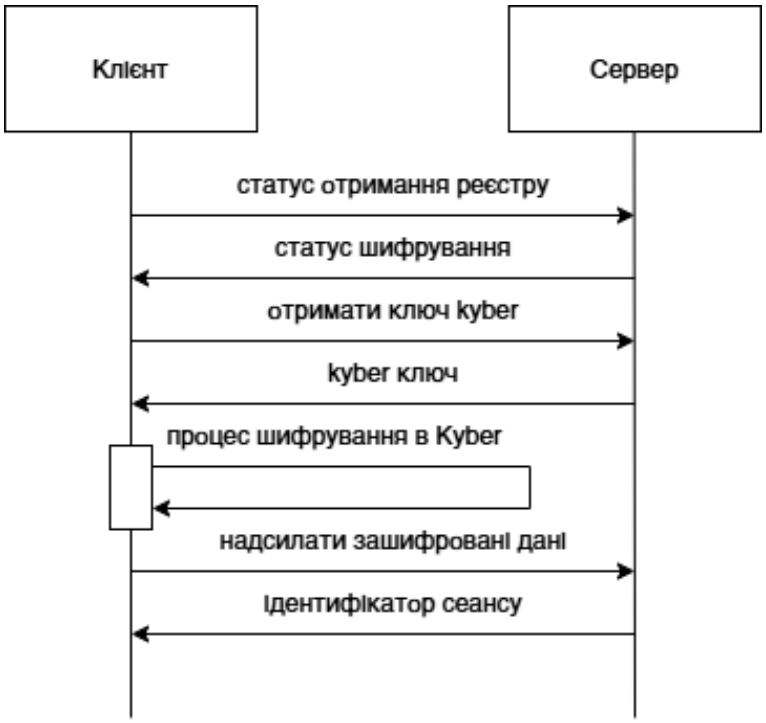


Рисунок 2.2 – UML діаграма застосування взаємодії клієнта і сервера для передачі ключа, захищеного постквантовим алгоритмом шифрування

Сервер створює унікальний ключ, стійкий до квантових атак, і пересилає його клієнту через безпечний канал. Після цього клієнт проводить локальне шифрування файлу і відправляє його назад на сервер для зберігання. Якщо відбувається спроба втручання або перехоплення сигналу на будь-якому етапі, система реєструє зміну квантового стану, що попереджає обидві сторони про потенційну загрозу. Цей механізм втілює основні принципи квантового розподілу ключів, до прикладу, перевірку цілісності каналу через аналіз змін поляризації фотонів або станів кубітів у змодельованому середовищі.

2.4 Розробка загальної моделі роботи програми

Вебплатформа складається з важливих функціональних модулів, які безперешкодно взаємодіють в єдиній безпечній системі (рисунок 2.3).

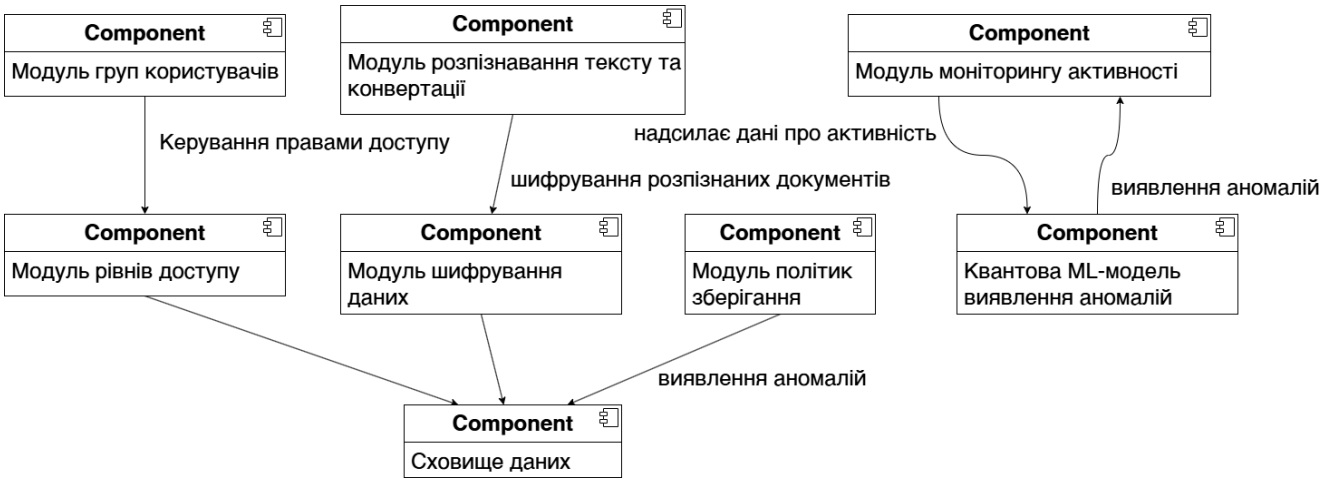


Рисунок 2.3 – Модель роботи системи у виглядіUML діаграми компонентів

Основним компонентом системи є модуль шифрування даних, який використовує постквантові алгоритми, такі як ML-KEM [9] для асиметричного шифрування та AES як еталон для симетричного шифрування. Ці алгоритми гарантують стійкість до атак, які потенційно можуть бути виконані квантовими комп'ютерами. Дані користувача залишаються зашифрованими до тих пір, поки вони не зберігаються, при цьому ключі передаються та зберігаються в захищеному середовищі.

Система зберігання даних структурована як централізоване захищене сховище, яке вміщує резервне копіювання, реєстрацію дій користувача та політику автоматичного знищення файлів. Фреймворк зберігання дозволяє класифікувати інформацію на групи на основі прав доступу та ролей користувачів.

Групи користувачів контролюються за допомогою окремого модуля, що дозволяє встановлювати логічні групи з різними рівнями доступу. Це гарантує

сегрегацію даних між групами та спрощує управління. Доступ до функцій платформи полегшується за допомогою токенів JWT, тоді як автентифікація користувачів досягається за допомогою інтеграції з GoogleAuth.

Незалежний модуль розпізнавання тексту дає вам змогу оцифрувати як друковані, так і рукописні документи, перетворюючи їх у зручні для користувача формати PDF, DOCX, TXT та дозволяючи подальше шифрування з метою безпеки. Ця методологія пропонує комплексний робочий процес обробки фізичних документів.

2.5. Розробка архітектури квантової моделі виявлення аномальної поведінки

Станом на сьогодні існує генеративна змагальна мережа [6], яка була взята як основа для розробки квантової моделі виявлення спроби пошкодження сигналів передачі ключа між користувачами системи. Основна логіка її роботи полягає у взаємодії між генератором та дискримінатором, головна задача якого є розрізняти, чи подані сигнали звичайні, чи аномальні. При навчанні дискримінатор стає кращим у розпізнаванні атак, а генератор у спробі обманути його.

Щоб виявити аномальну поведінку, потрібні тести, результати яких вводяться в розроблену і навчену квантову машинну модель. Потім ступінь загрози виявленої поведінки оцінюється мережею виявлення та оцінки поведінки для того, щоб оцінити безпеку поведінки користувача. Розроблена програма матиме за основу квантово-генеративну змагальну мережу.

На основі неї було створено архітектуру квантової моделі машинного навчання [10], яка матиме здатність визначати рівень аномальних сигналів на вебплатформі за допомогою навченого дешифратора у квантовій суперпозиції, що проілюстровано на рисунку 2.4.

Рисунок 2.4 – Архітектура роботи квантової моделі машинного навчання

Розроблена архітектура моделі базується на заданих квантових вентилях. Їх головна ціль бути квантовою операцією, яка створить квантову заплутаність між кубітами. Кожен шар складається з вентилів на кшталт Pauli-Y, які виконують обертання за допомогою операції $RY(\theta_{ij})$, де θ_{ij} — це кут обертання для і-го кубіта в j-му шарі, а Z – це контролюючі вентилі. На етапі навчання на виході проводиться перебудова для мінімізації похибки реконструкції.

При виявленні аномалії зразки аномальної поведінки, як правило, менші, ніж нормальні зразки. Зокрема, що стосується внутрішньої аномальної поведінки, вона зустрічається набагато рідше і навіть покривається великою кількістю нормальних даних.

Модель виявлення та оцінки поведінки аномалій використовує двокласову згорткову нейронну мережу, як показано на рисунку 2.5. Ця мережа складається з двох шарів згортки, двох шарів максимального об’єднання та кінцевого вихідного шару. Вихідний шар складається лише з одного нейрона.

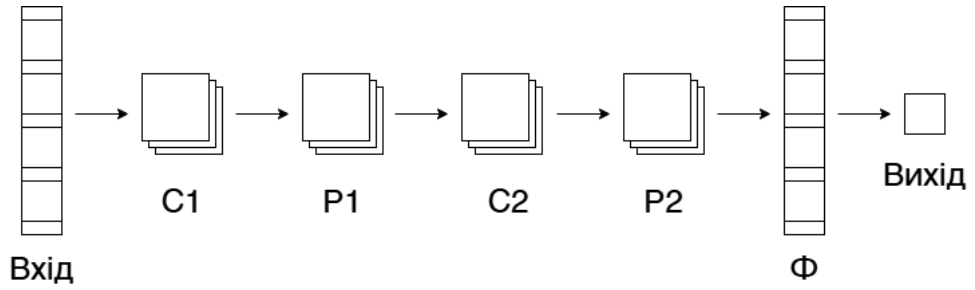


Рисунок 2.5 – Модель виявлення та оцінки поведінки для внутрішнього користувача.

У розробці методу машинної моделі дані про активність користувача спочатку збираються з вихідного джерела і згодом піддаються попередній обробці, після чого вони перетворюються на числові вектори. Потім ці вектори кодуються в квантові стани через обертання навколо осі Y, тим самим утворюючи вхід для квантової схеми.

Основна архітектура квантової моделі базується на варіаційному квантовому класифікаторі, який мітить клапани та елементи заплутування, які полегшують моделювання складних кореляцій. Після квантової обробки кубіти піддаються вимірюванню, а отримані дані конвертуються в класичний вектор, який потім передається дискримінатору. Він у свою чергу проводить порівняльний аналіз отриманих векторів проти моделі нормативної поведінки і при виявленні істотного відхилення видає сигнал аномалії.

В результаті виявлення незвичайного сигналу програма за поданою схемою аналізуватиме якого рівня це втручання – високого чи низького. Відслідковувати сигнали можна буде цілодобово, що гарантує за допомогою графіків чи заходить рівень за допустимий рівень.

2.6. Розробка методу роботи розпізнавання тексту з зображення

У ході розробки модуля з комп'ютерним баченням було розроблено метод розпізнавання тексту з зображення в модальному вікні.

1. Процес розпочинається з актора «User», який запускає дію «Відкрити модальне вікно». У відповідь на це активується «TesseractModal» і запускає процес рендерингу. Компонент відображає основні елементи інтерфейсу модального вікна, зокрема та, які створюють загальну структуру і візуальний стиль. Одночасно TesseractModal також генерує HTML-елемент, що буде призначений для завантаження зображень. Цей елемент в діаграмі позначено як «HTML Input Element», що вказує доступ до нього через useRef у компоненті React.

2. Після вдалого рендерингу, користувач взаємодіє з інтерфейсом, здійснюючи дію «Натиснути Upload files і обрати файл». Ця операція виконується безпосередньо через «HTML Input Element (ref)», який викликає подію onChange після вибору файлу. Ця подія повертається до «TesseractModal». У реакцію на подію «onChange TesseractModal» активує свій метод «handleFileChange». У межах цього методу компонент «Зберігає обраний файл setImageFile» у своєму внутрішньому стані готує його до подальшого опрацювання. Суттєвим етапом є «Очищення значення input (ref.current.value = " ")», що дає змогу користувачеві повторно обрати той самий файл у разі потреби, без потреби у скиданні всього компонента. По завершенні цих дій компонент TesseractModal та «HTML Input Element» переходять до пасивного стану, чекаючи нових вказівок.

3. Наступним важливим кроком є дія користувача «Submit», що запускає метод handlerSubmit() в «TesseractModal». Компонент негайно перевіряє «Наявність imageFile» у своєму стані. Якщо файл зображення не знайдено, логіка діаграми передбачає альтернативний шлях «imageFile відсутній», де компонент «нічого не виконує або відображає повідомлення», можливо, інформуючи користувача про потребу вибору файлу.

4. Якщо «imageFile» присутній, активується основний потік розпізнавання тексту, що є важливою частиною алгоритму. «TesseractModal» запускає робочий процес Tesseract.js, викликавши createWorker("eng") для «Tesseract.js Worker». Після запуску, «Tesseract.js Worker» надсилає підтвердження «worker (ініціалізовано)» назад до компонента. Потім TesseractModal надсилає обране зображення для розпізнавання, викликаючи recognize(imageFile) на «Tesseract.js Worker». «Tesseract.js Worker» активно здійснює оптичне розпізнавання символів – складний етап аналізу зображення і виділення текстової інформації. По завершенні OCR, «Tesseract.js Worker» передає результат розпізнавання назад до TesseractModal. Негайно після отримання результату TesseractModal викликає terminate() у «Tesseract.js Worker», звільняючи ресурси.

5. Отримавши результат ret, «TesseractModal» витягує розпізнаний текст, застосовуючи recognizedText = ret.data.text. Цей текст потім конвертується у двійковий формат (Blob) через дію «Створити Blob із recognizedText» для «Browser (File API)». Браузер створює тимчасову URL-адресу об'єкта (URL.createObjectURL), яка передається до TesseractModal. Компонент генерує тимчасовий HTML-елемент і програмно симулює його натискання (a.click()) для того, щоб «Завантажити 'recognized_text.txt'» на пристрій користувача. Ця дія показується стрілкою від «Користувача» до «Браузера». По завершенню завантаження, TesseractModal виконує URL.revokeObjectURL(url) для «Browser (File API)», звільняючи системні ресурси, асоційовані з тимчасовою URL-адресою. На цьому етапі завершальний етап шифрування відбувається.

Діаграма послідовності демонструє динамічний процес взаємодії та етапи, що реалізуються в React-компоненті TesseractModal для розпізнавання тексту з обраного зображення користувача. Діаграма яскраво ілюструє взаємодію між користувачем, основним React-компонентом, елементом HTML для введення

файлів, бібліотекою Tesseract.js та можливостями браузера, що дає змогу отримати детальне уявлення про алгоритм функціонування (рисунок 2.6).

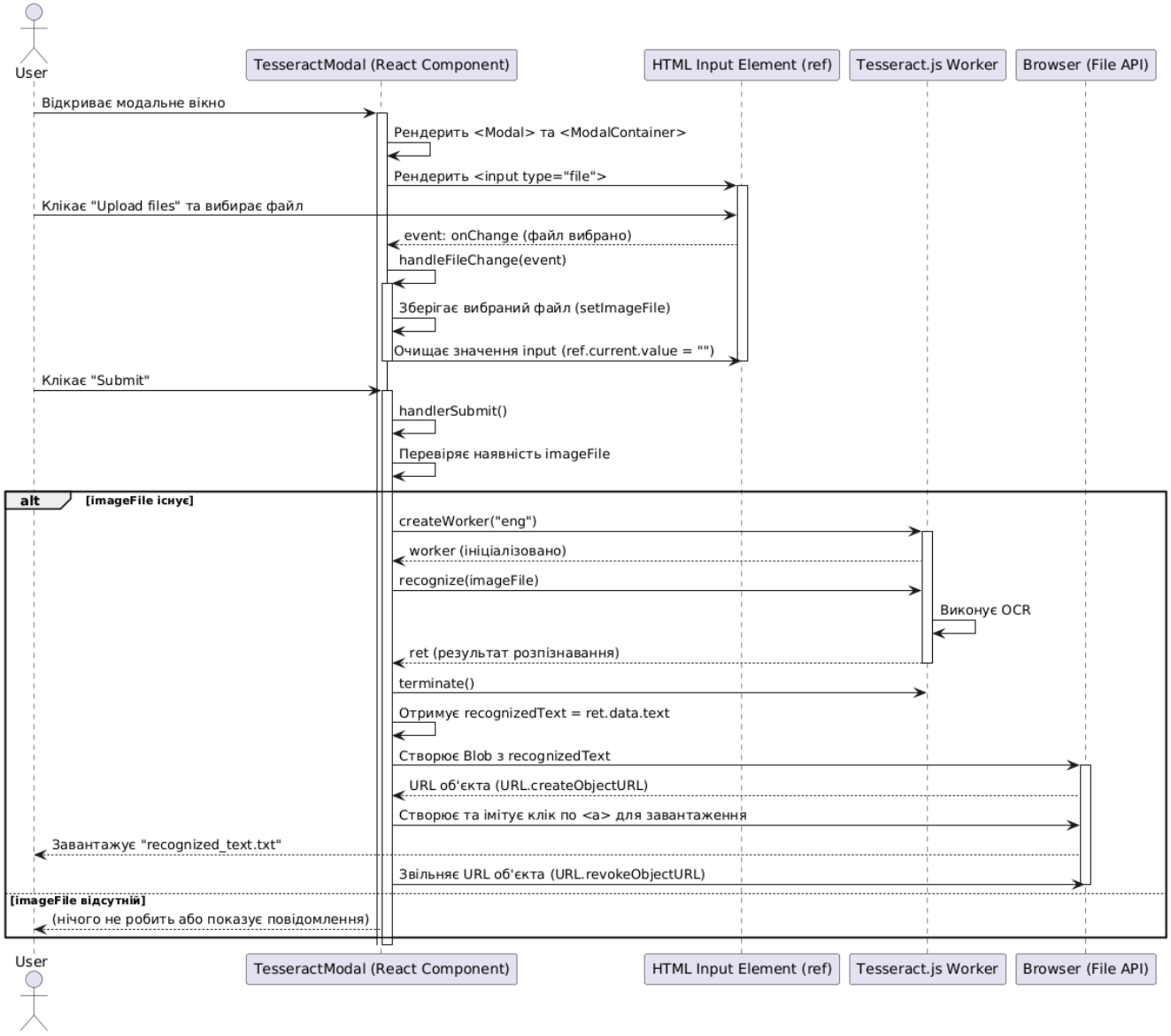


Рисунок 2.6 – Діаграма послідовності

Отже, ця діаграма послідовності детально ілюструє архітектуру та логіку взаємодії елементів, від ініціалізації модального вікна і вибору файлу, до складного процесу OCR та фінального завантаження розпізнаного тексту.

2.6. Розробка AES алгоритму шифрування

Описаний алгоритм криптографічних операцій AES (рисунок 2.7) починається етапу ініціалізації, що налаштовує систему для обробки запитів. Цей алгоритм може одночасно генерувати ключі, шифрувати інформацію та її дешифрувати. Алгоритм отримує 32 випадкових байти для генерації ключа і кодує їх у форматі Base64, який потім повертається як результат. Процес шифрування починається зі створення унікального 16-байтового вектора ініціалізації та декодування ключа з формату Base64, потім створюється і ініціюється об'єкт шифрувальника.

Безпосереднє шифрування інформації здійснюється шляхом її трансформування у буфер, відправлення до Cipher та завершення процесу з розрахунком аутентифікаційного тегу. Зашифровані дані та IV формуються у Hex та Base64 та повертаються у вигляді структурованого об'єкта. Дешифрування потребує зворотнього перетворення IV, ключа та зашифрованих даних з їхніх закодованих форматів, після чого створюється і запускається об'єкт декодувальника Decipher. Інформація передається Decipher, яка завершує процес, включаючи контроль цілісності отрманої інформації.

Успішне завершення операції дешифрування призводить до повернення оригінальних розшифрованих даних, тоді як будь-який збій у процесі викликає відповідну помилку. Кожна з трьох криптографічних операцій завершується збігом потоків виконання, що свідчить про успішне або неуспішне завершення відповідного завдання.

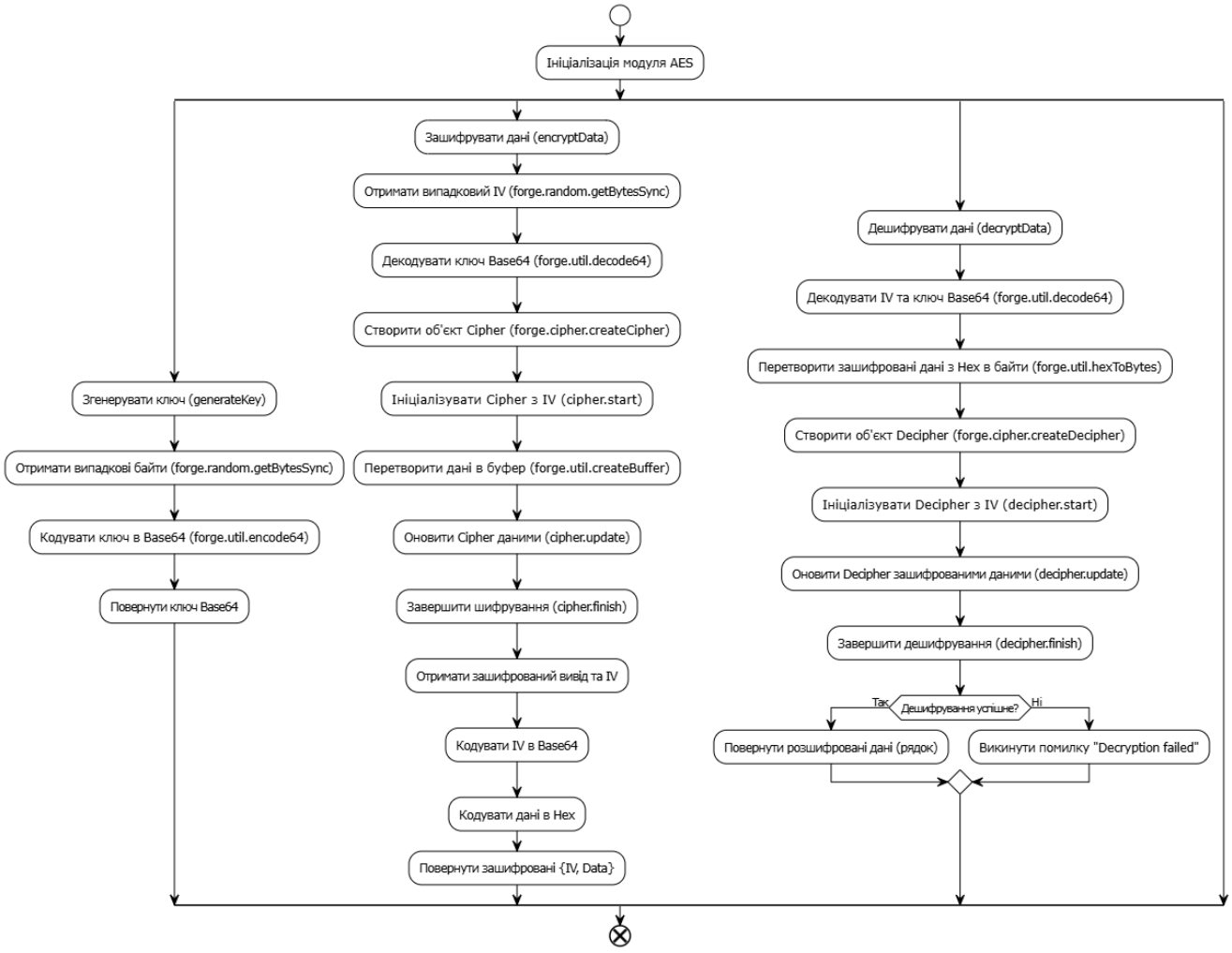


Рисунок 2.7 – Алгоритм шифрування AES

У результаті, представлений алгоритм забезпечує повний цикл криптографічної обробки даних за стандартом AES, гарантуючи конфіденційність та цілісність інформації. Його структура дозволяє платформу уникнути можливих витоків інформації при спробі проникнення.

2.7 Розробка методу захисту середовища від несанкціонованого доступу

Система захисту середовища від несанкціонованого доступу базується на комплексній валідації та перевірці вхідних даних. Метод забезпечує, що всі дані, які надходять до системи, відповідають очікуваним форматам і типам, мінімізуючи

ризики, пов'язані з некоректними або шкідливими вхідними даними. В основі методу лежать три компоненти: модуль Validate, функція генерації випадкових шістнадцяткових значень genRanHex та стандартизований обробник HTTP-response. Метод працює наступним чином:

1. Перш ніж будь-які дані будуть оброблені чи збережені системою, вони проходять сувору перевірку за допомогою статичних методів класу Validate.

2. Для рядків використовується «Validate.string()», який перевіряє, чи є вхідне значення дійсним непустим рядком.

3. Для булевих значень застосовується «Validate.boolean()», що гарантує тип true або false.

4. Числові дані перевіряються за допомогою «Validate.number()» для забезпечення, що значення є дійсним, кінцевим числом.

5. Дати валідуються методом «Validate.date()», який перевіряє рядкове представлення на відповідність формату дати та перетворює його на об'єкт Date.

6. JSON об'єкти проходять перевірку за допомогою «Validate.json()», що підтверджує їхню коректну серіалізацію.

7. Унікальні ідентифікатори валідуються через «Validate.uuid()», який за допомогою регулярного виразу підтверджує відповідність стандарту.

8. Якщо під час будь-якої з цих перевірок дані виявляються некоректними, клас Validate генерує виняток «Error("Validate error")», який дозволяє системі централізовано обробляти помилки валідації.

9. У випадках, коли системі потрібно згенерувати унікальні або тимчасові шістнадцяткові рядки, наприклад, для сесійних токенів, використовується функція «genRanHex».

10. Після завершення всіх етапів перевірки та обробки даних, функція response відповідає за відправку стандартизованих HTTP-відповідей клієнту.

11. Якщо всі валідації пройшли успішно і операція виконана коректно, функція response відправляє клієнту відповідь зі статусом 200 OK та необхідними даними.

12. У випадку, якщо під час валідації або виконання операції виникла помилка, функція response автоматично встановлює відповідний HTTP-статус відповіді, наприклад, 400 Bad Request або 500 Internal Server Error, і відправляє JSON-відповідь, що містить повідомлення про помилку без даних.

В основі методу захисту середовища від несанкціонованого доступу лежить алгоритм валідації вхідних даних, що реалізовано з використанням засобів та статичних методів, що надаються мовою програмування «TypeScript». Блок-схему алгоритму валідації продемонстровано на рисунку 2.8.

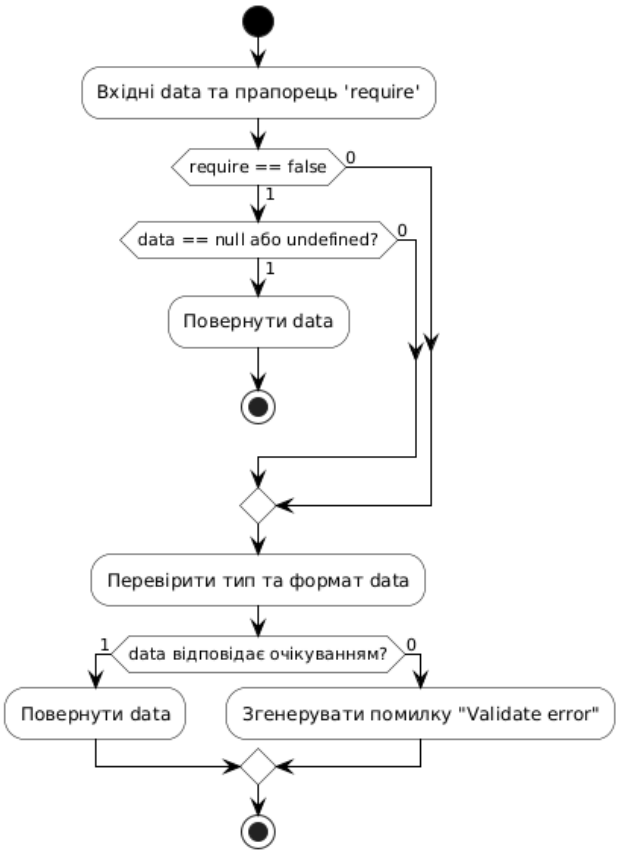


Рисунок 2.8 – Блок-схема алгоритму роботи методу валідації даних

Таким чином, метод захисту інтегрує валідацію даних на ранніх етапах, гарантуючи обробку лише коректних та безпечних даних, а також забезпечує стандартизовану обробку відповідей.

2.8 Висновки

У другому розділі було розроблено метод та моделі системи захисту вебплатформи Quantum Information Science.

Було описано аналіз вхідних та вихідних даних системи. Особливу увагу було приділено розробці загальної моделі роботи програмного застосунку, розробці методу захисту середовища

Велику увагу було приділено розробці методу моделі квантового машинного навчання, яку було розгорнуто як незалежну службу, що доступна через API. Ця модель заснована на генеративній конкурентній мережі і розроблена для виявлення аномальної поведінки користувачів у реальному часі.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення

При розробці вебплатформи зі стійким сховищем да атак було застосовано сучасний стек технологій, орієнтований на забезпечення високої надійності, масштабованості, зручності у використанні та стійкості до сучасних і майбутніх кіберзагроз.

Механізм інкапсуляції ключів навчання модулів з помилками, або ML-KEM – це криптографічний алгоритм, що базується на складності задач у решітках. Він є частиною постквантових алгоритмів, рекомендованих NIST [11] для стандартного використання в майбутньому, коли квантові комп’ютери зможуть порушувати класичні криптографічні системи.

Мкханізм у проекті використовується для безпечного обміну симетричними ключами між клієнтом і сервером. Завдяки своїй стійкості до атак з боку квантових комп’ютерів, ML-KEM гарантує збереження конфіденційності навіть у майбутньому квантовому середовищі.

AES [12] – симетричний блоковий шифр, який був стандартизований Національним інститутом стандартів і технологій США NIST. Він шифрує дані блоками по 128 біт, використовуючи ключі довжиною 128, 192 або 256 біт.

У розробці AES застосовується для шифрування збережених файлів, повідомлень, журналів дій користувачів та іншої чутливої інформації. Комбінація AES з ML-KEM створює надійний механізм шифрування, де AES забезпечує швидкість, а ML-KEM стійкість до квантових атак.

Маркер автентифікації JSON Web Token це відкритий стандарт RFC 7519, який визначає компактний і автономний спосіб безпечного передачі інформації між

сторонами як об’єкта JSON. Він може бути підписаний за допомогою секретного ключа або пари відкритий-приватний ключ.

У реалізації застосунку JWT [13] використовується для підтвердження особи користувача після авторизації. Платформа формує токен, що містить зашифровану інформацію про користувача, яку потім перевіряє сервер для надання доступу до захищених функцій.

Time-based One-Time Password[14]– алгоритм, що генерує одноразові паролі на основі часу. Він є частиною стандарту багатофакторної автентифікації, де код дійсний лише протягом короткого періоду. У розробці він застосовується як додатковий рівень захисту під час входу користувача. Користувач, окрім введення пароля, вводить код, що генерується мобільним додатком, і дійсний лише протягом 30 секунд.

Node.js [15] – середовище виконання JavaScript, побудоване на рушії V8 від Google. Воно дозволяє запускати JavaScript-код на сервері, підтримуючи подієво-орієнтовану, неблокуючу модель вводу/виводу. У кодингу використовується для обробки всіх бекенд-запитів, таких як автентифікація, логіка обробки даних, генерація токенів, шифрування, дешифрування інформації, тощо.

Next.js [16] це фреймворк для створення React-додатків, який підтримує серверний рендеринг, статичну генерацію сторінок і маршрутизацію без налаштувань. У розробці служить основою для побудови форми реєстрації, налаштування профілю, доступ до сховищ і налаштування безпеки.

TypeScript [17] – мова програмування, яка є надмножиною JavaScript, що додає статичну типізацію. Вона дозволяє розробникам виявляти помилки на етапі написання коду. В практичній реалізації забезпечує надійність і зменшує кількість помилок при написанні коду, особливо важливо у криптографічних функціях, де будь-яка помилка може призвести до втрати безпеки.

PostgreSQL [18] є системою управління базами даних з відкритим кодом, яка підтримує транзакції, збережені процедури, тригери, індекси та багато іншого.

У розробці всі дані користувачів, а також метадані щодо файлів і прав доступу, зберігаються у шифрованому вигляді в PostgreSQL. Система забезпечує гнучке управління доступом та сумісність із шифруванням.

Tesseract [19] – бібліотека оптичного розпізнавання символів, яка дозволяє зчитувати текст із растрових зображень і PDF-файлів. Застосовується як додатковий інструмент для розпізнавання текстової інформації з документів, які користувачі завантажують у систему.

Далі буде описано технології інфраструктури та розробки, що були використані у реалізованому застосунку.

Першою із них є Amazon Web Services – масштабована хмарна платформа, яка надає послуги з обчислення, зберігання даних, доставки контенту та інших функцій, що допомагають бізнесам швидко розгортати застосунки. Сервери, бази даних, сховище файлів, балансувальники навантаження – усе це розміщене на AWS, що надає надійність, відмовостійкість і гнучке масштабування під час зростання кількості користувачів.

Система Docker [20] потрібна для створення, розгортання та запуску додатків у контейнерах, які є ізольованими середовищами з усім необхідним для їх функціонування. Сервер проекту , база, клієнт знаходяться в окремому контейнері, що спрощує тестування, розгортання та масштабування системи.

Інструмент Postman потрібен для тестування API, який дозволяє розробникам надсилати HTTP-запити до вебсервісів і перевіряти їхню коректність. У розробці застосовується для тестування всіх API-запитів, включно з перевіркою криптографічних функцій, реєстрації, шифрування повідомлень, перевірки токенів, тощо.

Останнім використаним інструментом є ESLint [21], що необхідний для статичного аналізу коду JavaScript, TypeScript, який виявляє проблеми та помилки, а також допомагає дотримуватися єдиного стилю кодування. Застосований для гарантування єдиного стилю написання коду та запобігання помилкам у критичних місцях, зокрема у модулях шифрування та авторизації.

Компанії-розробники програмного забезпечення пропонують широкий вибір сучасних середовищ розробки мобільних застосунків. Розглянемо найпопулярніші з них: Visual Studio Code, Eclipse, IntelliJ IDEA та Visual Studio.

У рамках створення безпечної вебплатформи, що характеризується квантово-стійкими рішеннями зберігання даних, був використаний сучасний технологічний стек, який підкреслює підвищення надійності, масштабованості, зручності для користувача та стійкості як проти поточних, так і потенційних кіберзагроз, особливо тих, що створюються квантовими обчисленнями.

Платформа була розроблена для полегшення безпечного зберігання та обміну даними між користувачами. Розробка інтерфейсів і логіки клієнтських і серверних компонентів виконана в інтегрованому середовищі Visual Studio Code.

Visual Studio Code [22] являє собою легке, швидке та високофункціональне середовище розробки, розроблене корпорацією Майкрософт. Він вміщує безліч мов програмування і може похвалитися розширюваною архітектурою, яку можна покращити за допомогою плагінів, таких як Docker, ESLint, Postgres і Git, а також безперешкодною інтеграцією з хмарними сервісами. У контексті цього проекту VS Code став найбільш ефективним інструментом завдяки всебічній підтримці стека JavaScript/TypeScript/Node.js, який є основою для більшості функціональних можливостей веб-платформи.

Крім того, середовище вміє розміщувати криптографічні бібліотеки, включаючи ML-KEM, AES та JWT, що є незамінним для реалізації постквантових заходів безпеки.

Eclipse [23]являє собою вільне та відкрите інтегроване середовище розробки (IDE), орієнтоване в основному на Java, хоча воно також вміщує C ++, Python та різні інші мови за допомогою плагінів. Він оснащений надійною системою налагодження та великою інфраструктурою проекту. Тим не менш, інтерфейс Eclipse виглядає дещо застарілим, а процес його конфігурації складніший, ніж у конкурентів.

Отже, він не оптимально підходить для розробки криптографічних веб-платформ через неадекватну інтеграцію з JavaScript/TypeScript та сучасними веб-фреймворками.

IntelliJ IDEA [24] являє собою середовище професійного розвитку, створене JetBrains, яке підтримує різноманітний набір мов програмування та фреймворків. Він пропонує зручну навігацію кодом, можливості рефакторингу, автоматичні пропозиції, інструменти управління базами даних та глибоку інтеграцію з Git.

Хоча він добре підходить для великих проектів, що використовують Java, Kotlin та Spring, він порівняно менш оптимізований для легких веб-платформ, заснованих на TypeScript/Node.js, які лежать в основі нашої системи.

Visual Studio [25] служить потужним інтегрованим середовищем розробки (IDE) для розробки NET, C++, Python і мобільних додатків. Він наділений надійними вбудованими інструментами для тестування, налагодження, безперебійної інтеграції з Azure, підтримкою баз даних та сумісністю з численними фреймворками.

Хоча Visual Studio охоплює комплексний набір інструментів для складної розробки, вона також вимагає значних ресурсів та часу для конфігурації, що не завжди виправдано в контексті веб-платформ середнього розміру. Порівняння функціональних характеристик середовищ наведено в таблиці 3.1.

Таблиця 3.1 – Таблиця порівняння функціональних характеристик середовищ програмування

Характеристика	VS Code	Eclipse	IntelliJ IDEA	Visual Studio
Підтримка веброзробки	1	1	1	1
Вбудована інтеграція з Docker, Git, Postgres, AWS	1	1	1	1
Підтримка JavaScript/TypeScript/Node.js	1	1	0	1
Розширення для криптографії AES, ML-KEM, JWT	1	0	0	1
Можливість підключення сторонніх плагінів	1	0	0	1
Легкість, швидкість запуску	1	0	0	0
Підтримка багатофакторної аутентифікації з TOTP інтеграцією	1	0	0	1
Гнучке налаштування інтерфейсу під проєкт	1	1	1	0
Загальний коефіцієнт	7	4	3	5

За таблицею, Visual Studio Code (VS Code) має вищий сумарний коефіцієнт порівняно з іншими середовищами розробки завдяки своїй гнучкості, легкості, розширюваності та підтримці великої кількості мов програмування. Це робить VS Code найкращим середовищем для розробки вебплатформи

3.2 Розробка головної сторінки платформи

При вході на веб-сайт перед користувачем відкривається головна сторінка, да описується вебплатформа, її ціль та переваги. Призначенням модуля автентифікації є створення простого та зручного інтерфейсу для входу та реєстрації користувачів у застосунку.

Процес розробки головної сторінки платформи також був дуже важливим, оскільки повинен був зацікавити користувача. Під час розробки активно використовувалися тривимірні елементи, векторна графіка та 3 кольорова гама, що додала приємного візуального ефекту для швидкої оцінки платформи (рисунок 3.1).

Під час створення модуля авторизації користувачів у систему важливо забезпечити повний захист від потенційних атак з використанням квантових комп'ютерів. З цієї причини жоден із способів входу до середовища не є вразливим до таких загроз, і жоден тип автоматизованої авторизації не прив'язаний безпосередньо до системи. Наприклад, під час спроби автоматичного входу через обліковий запис Google система переходить на протокол SSL, який, попри загальну захищеність, є потенційно вразливим до квантових атак.

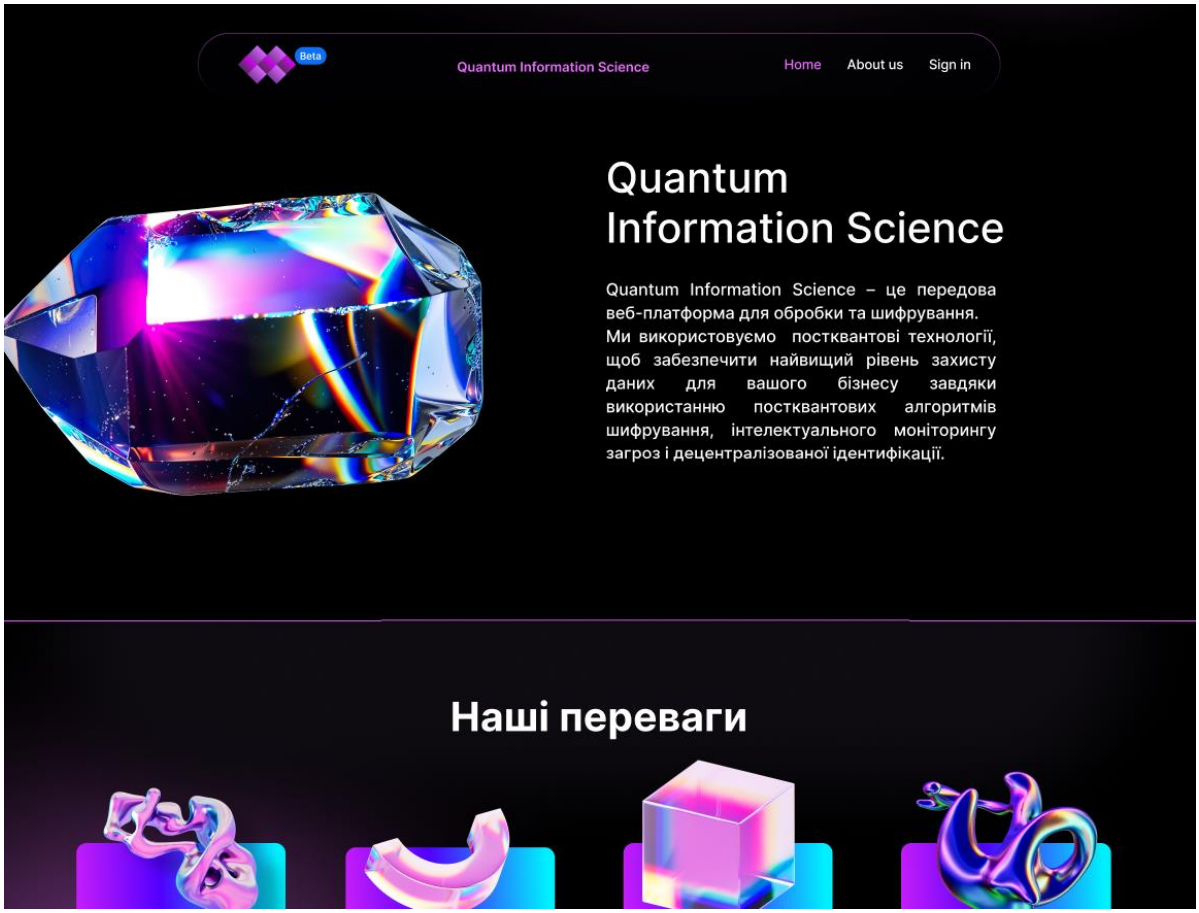


Рисунок 3.1 – Головна сторінка

3.3 Розробка сторінки реєстрації користувача

У процесі розробки модуля велика увага приділялася зручності користування та дизайну інтерфейсу. Для початку було розроблено схематичне представлення кроків реєстрації користувача, що показано на рисунку 3.2.

Рисунок 3.2 – Схема інтерфесу кроків входу користувача у систему

Нижче наведений першик крок – вибору реєстрації чи вхоу корстувача (рисунок 3.3).

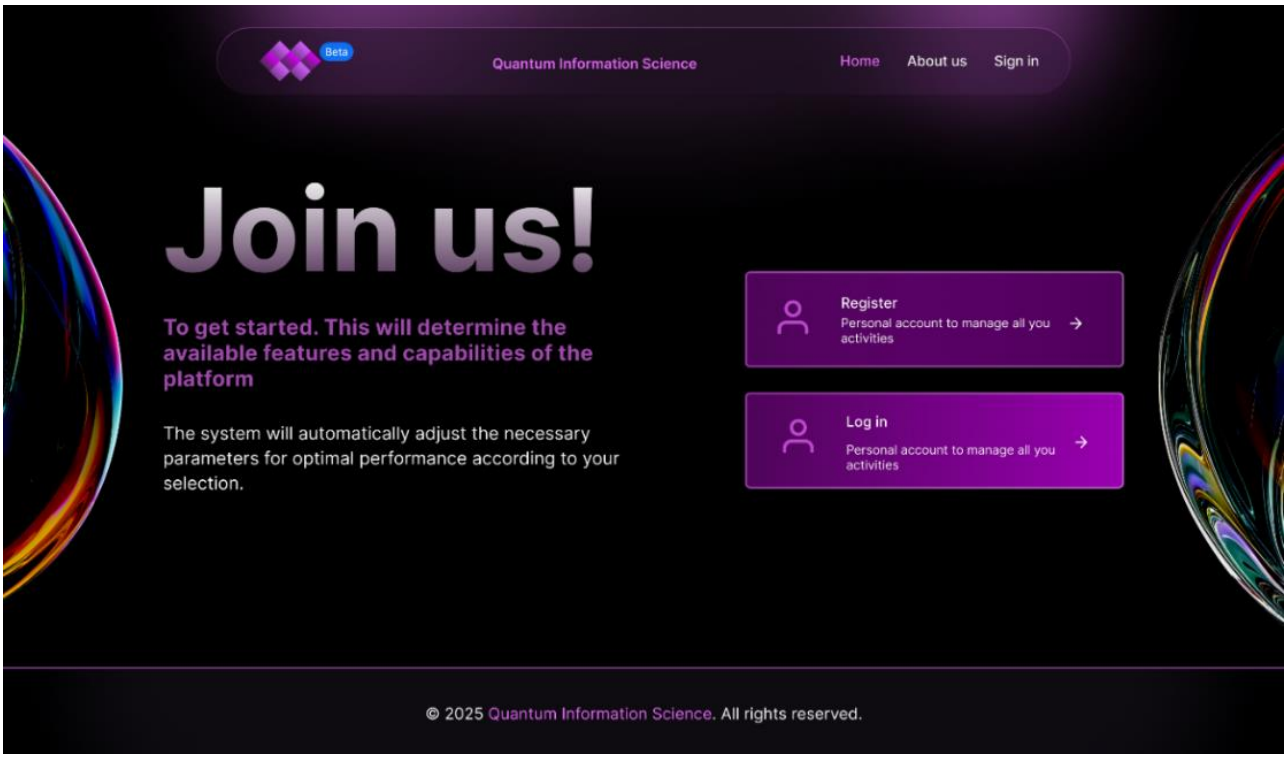


Рисунок 3.3 – Сторінка входу у систему

Якщо користувач звертається до системи вперше, йому потрібно пройти процес реєстрації. Цей механізм не тільки підтверджує особистість, а й створює індивідуальний профіль із чітко окресленими правами доступу. У процесі реєстрації генерується криптографічний контейнер з прив’язкою до конкретного ID користувача, в якому зберігаються права доступу, персональні параметри та ключі. Реєстрація також ініціює початковий запис у системі журналу аудиту.

При спробі входу, для отримання доступу до панелі управління сховищем, користувачу необхідно ввести свою електронну пошту та пароль для автентифікації у систему (рисунок 3.4).

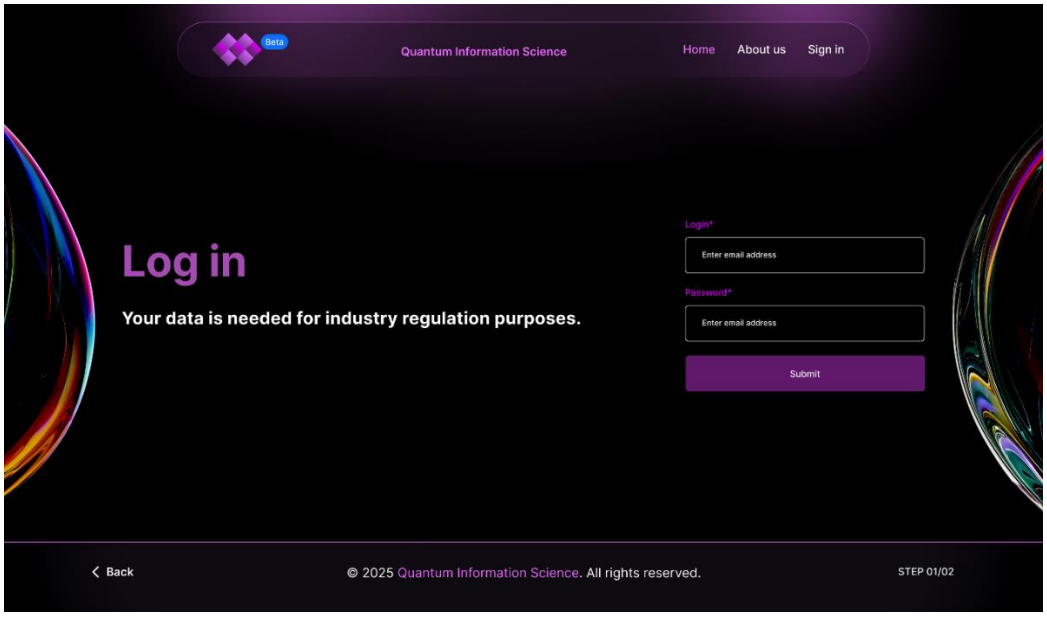


Рисунок 3.4 – Крок вводу даних користувача при вході

Важливо, що усі дані користувача на вході є захищеними за допомогою алгоритму постквантового шифрування ML-KEM, нижче на рисунку 3.5 наведено блок-схему, що покроково демонструє його роботу. Це необхідно для унеможливлення перехвату ключа користувача при його вході у систему. Далі цей алгоритм також використовуватиметься при передачі та шифруванні файлів.

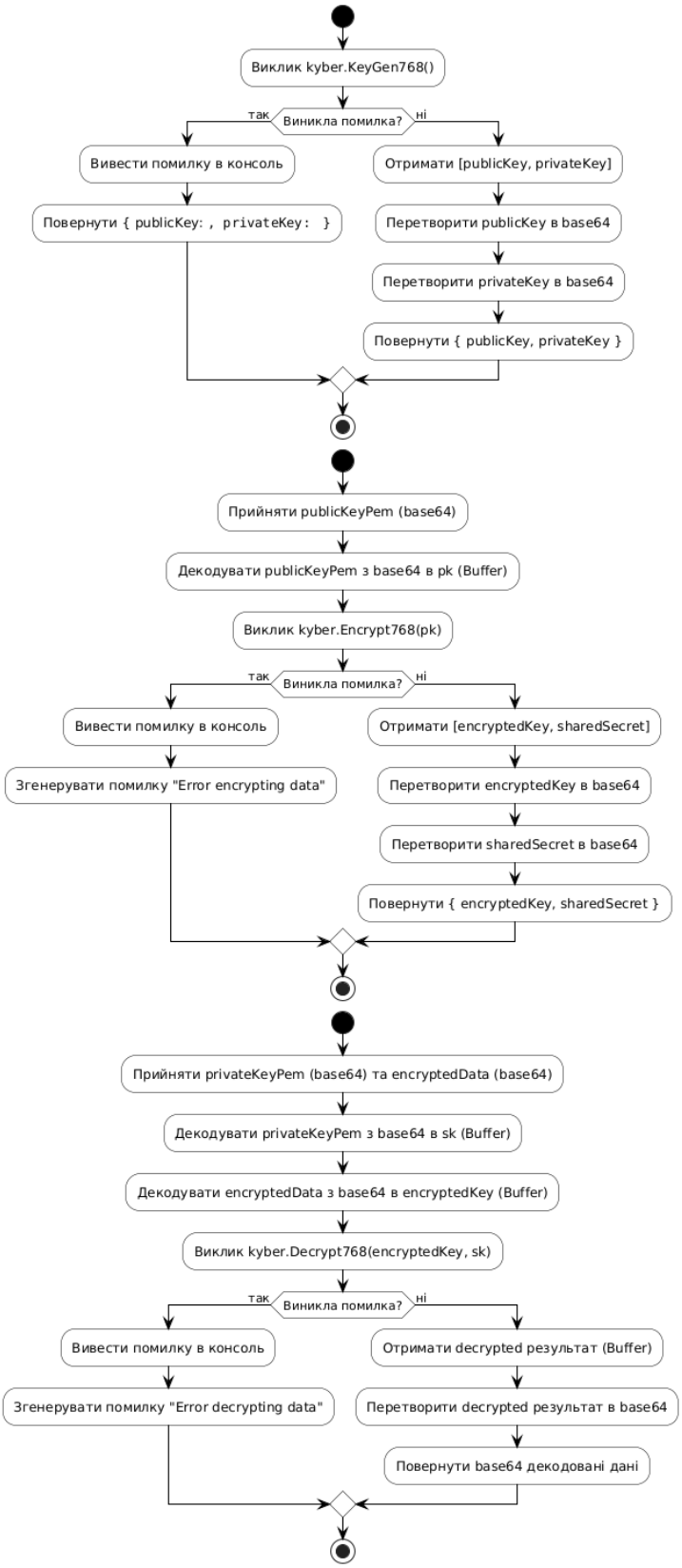


Рисунок 3.5 – Постквантовий алгоритм шифрування ML-KEM

Наступним важливим кроком для забезпечення непроникності у системі є введення TOTP ключа для мультифакторної авторизації користуваа у системі. Це тимчасовий код, що необхідний для отримання зв’язку користувача з журналом аудиту для запису актуальної сесії та перевірки користувача, як людини при спробі автентифікації (рисунок 3.6).

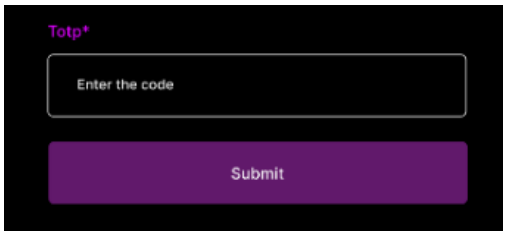


Рисунок 3.6 – Крок вводу TOTP ключа

3.3 Розробка головної панелі керування вебплатформи

Доступ до панелі керування користувача відкривається лише після введення персонального AES-256 ключа. Блок схема AES з 256-бітним ключем забезпечує захист, стійкий до атак, включаючи атаки з використанням квантових комп'ютерів (рисунок 3.7).

Компонент UserAesModal є модальним вікном, яке надає взаємодію користувача з його AES-ключем. Основна мета цього компонента – це забезпечити, щоб користувач мав активний AES-ключ для шифрування та дешифрування даних. Якщо ключ відсутній або не завантажений, модальне вікно пропонує завантажити існуючий ключ або згенерувати новий. Компонент використовує React-хуки useState, useEffect, useRef, Redux для управління станом userAesKey, userStore та взаємодії з API.

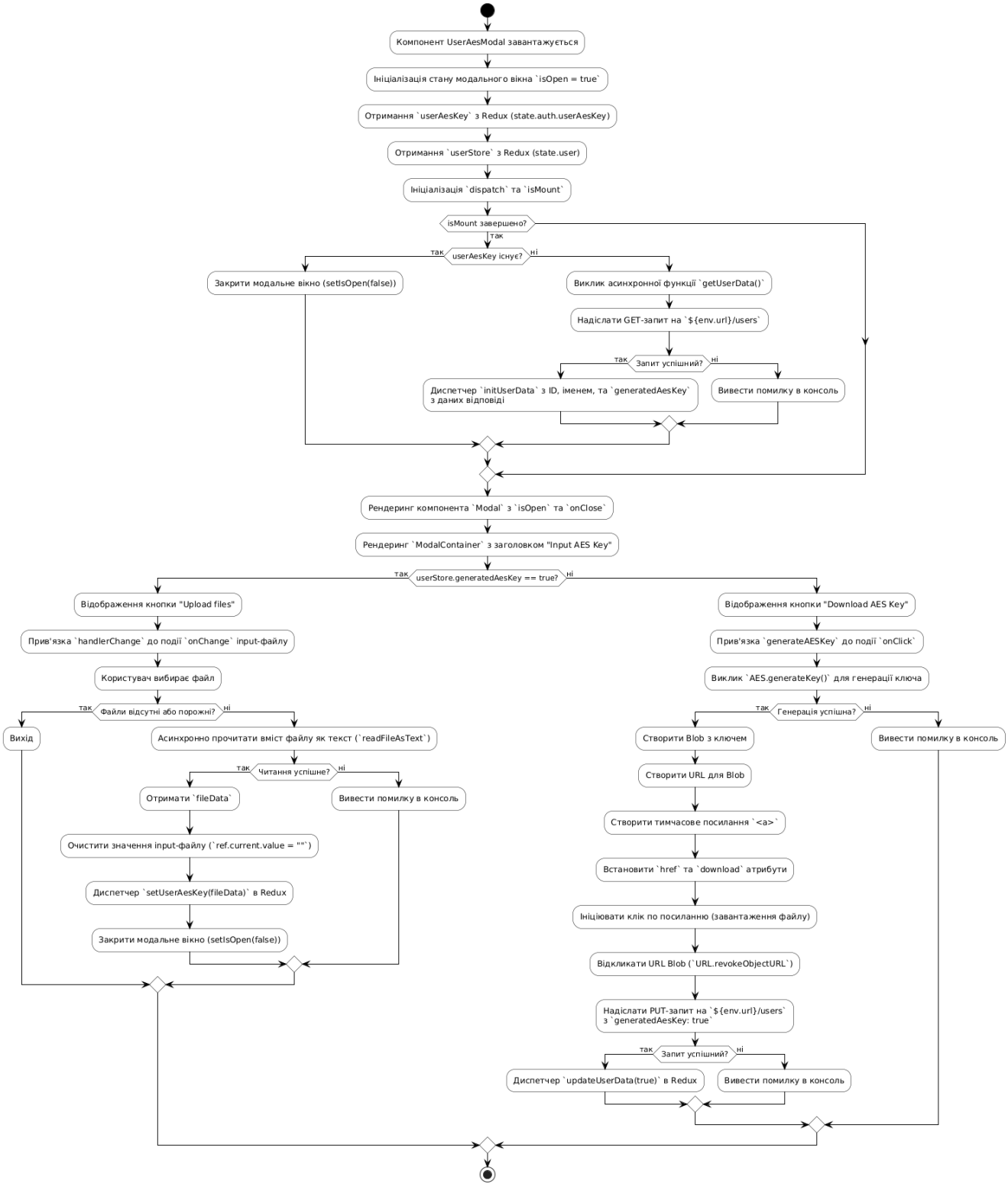


Рисунок 3.7 – Логіка роботи компонента UserAesModal

Після успішної авторизації користувач потрапляє на дешборд – головну панель керування. Візуально інтерфейс поділено на бічну панель навігації, верхню панель статусу та основне вікно контенту (рисунок 3.8).

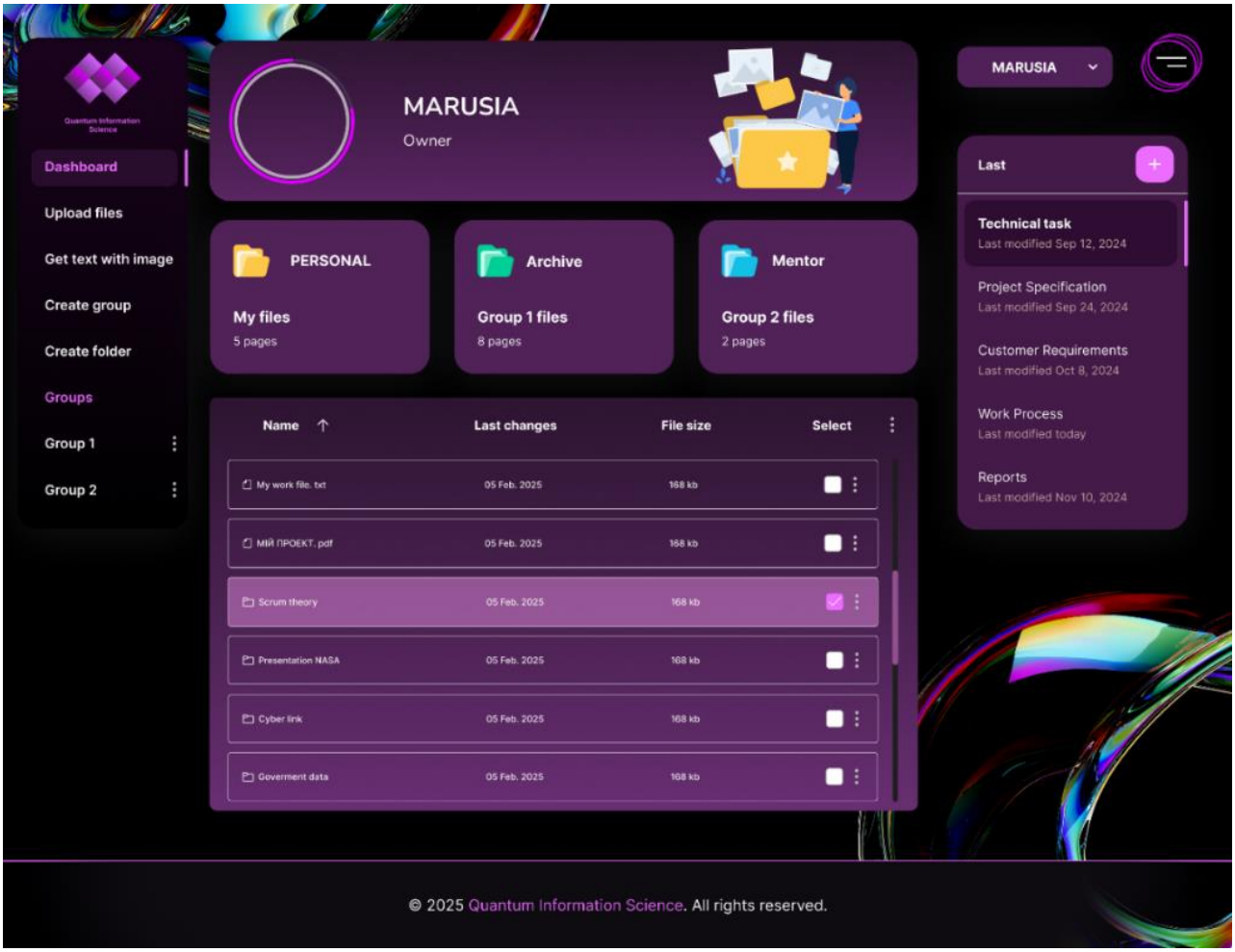


Рисунок 3.8 – Сторінка головної панелі керування

Логіка роботи дешборду детально продемонстрована за допомогою блок-схеми на рисунку 3.9.

Рисунок 3.9 – Блок-схема роботи головної панелі керування

Профіль користувача включає особисту інформацію, а також дозволяє оновити пароль, змінити TOTP ключ, переглянути IP-адреси входу та керувати сесіями. Усі дані, які відображаються, витягуються із зашифрованої бази даних і дешифруються безпосередньо у браузері користувача (рисунок 3.10).

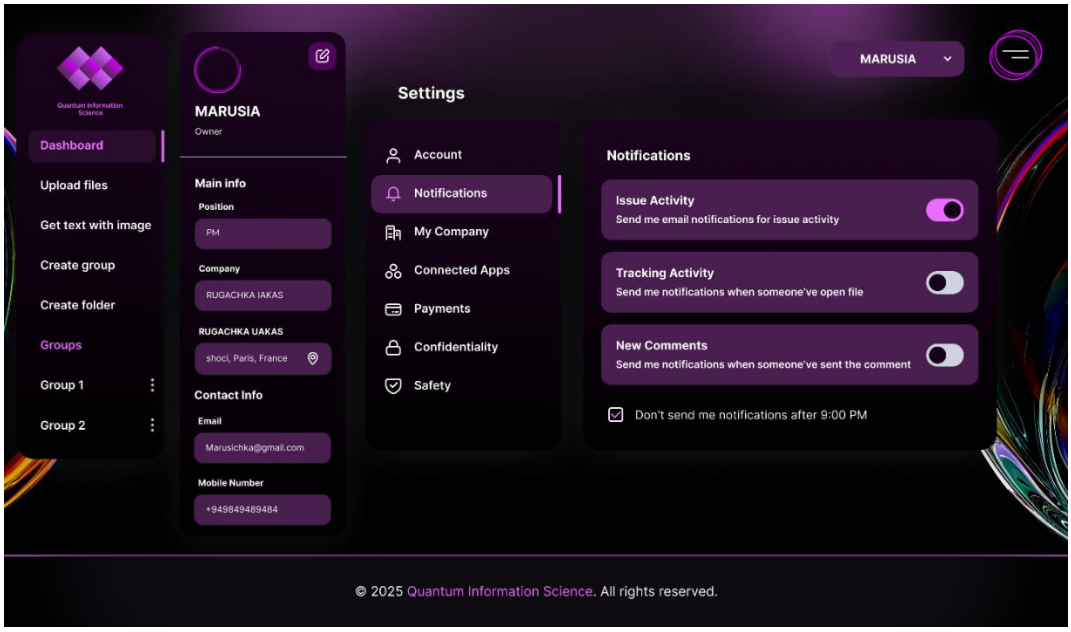


Рисунок 3.10 – Сторінка профілю користувача

У розділі «Групи» користувач бачить список усіх груп, до яких він має доступ. Є кнопка «Створити групу» для користувачів з відповідними правами. Під час перегляду конкретної групи відкривається її структура (рисунок 3.11).

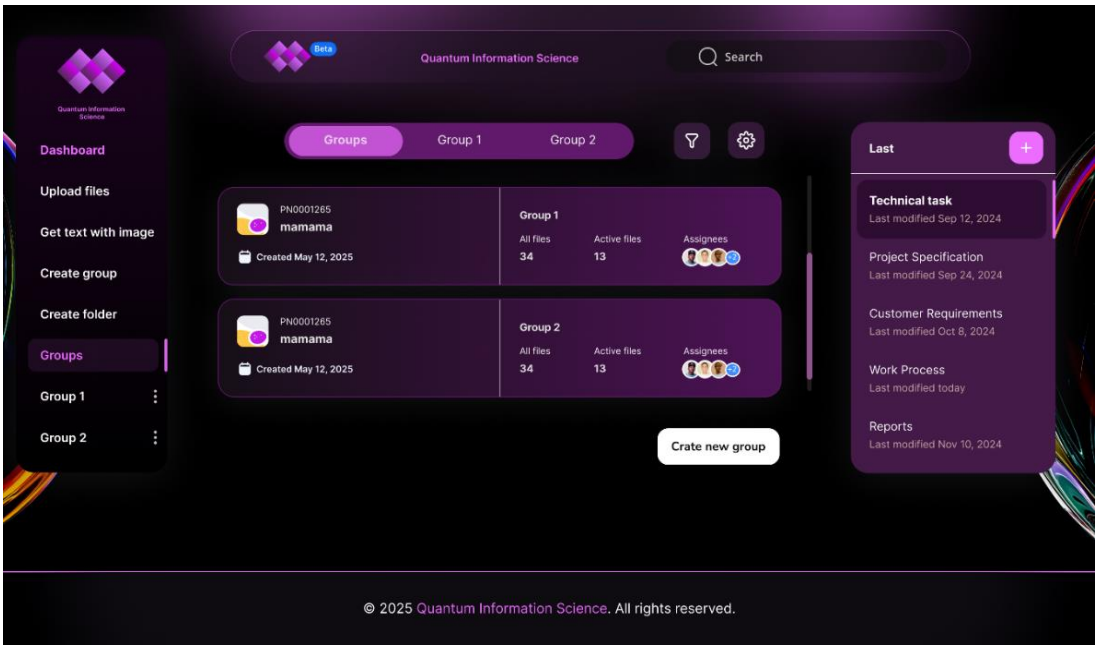


Рисунок 3.11 – Сторінка з групами та маніпуляція ними

Користувач може змінювати структуру групи, налаштовувати права доступу, додавати нових учасників або змінювати ролі існуючих. Крім того, він має змогу налаштовувати внутрішню організацію, перейменовувати групу, видаляти непотрібні елементи або змінювати порядок вкладених компонентів для зручності користування (рисунок 3.12).

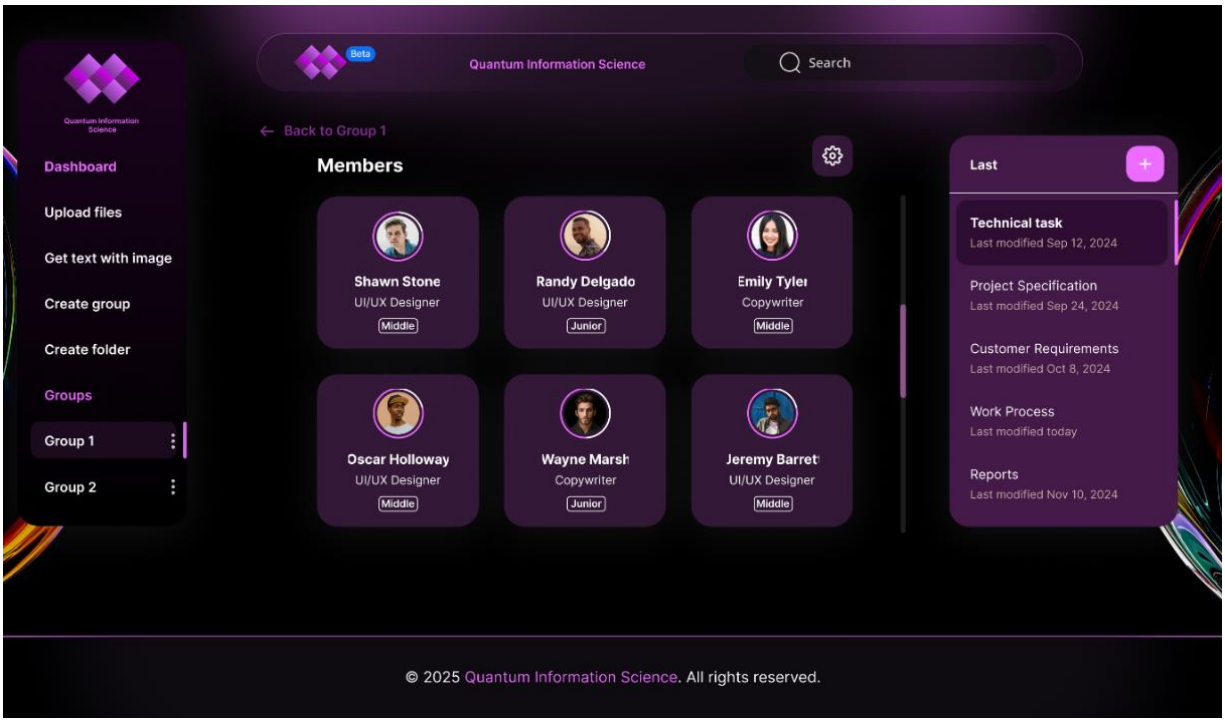


Рисунок 3.12 – Сторінка учасників групи

Також користувач має змогу переглядати матеріял групи. Це охоплює доступ до даних, а саме назву групи, її опис, список учасників та папок, файлів, підгруп. Натискаючи на групу, користувач має можливість побачити загальний вигляд, переходити між елементами, шукати необхідні дані або відкривати конкретні файли, такі як окрему папку або документ (рисунок 3.13).

Реалізована можливість навігації між елементами групи, пошуку за назвою чи тегами, відкриття окремих документів або вкладених папок для перегляду вмісту. Доступ до кожного елементу визначається відповідними рівнями прав.

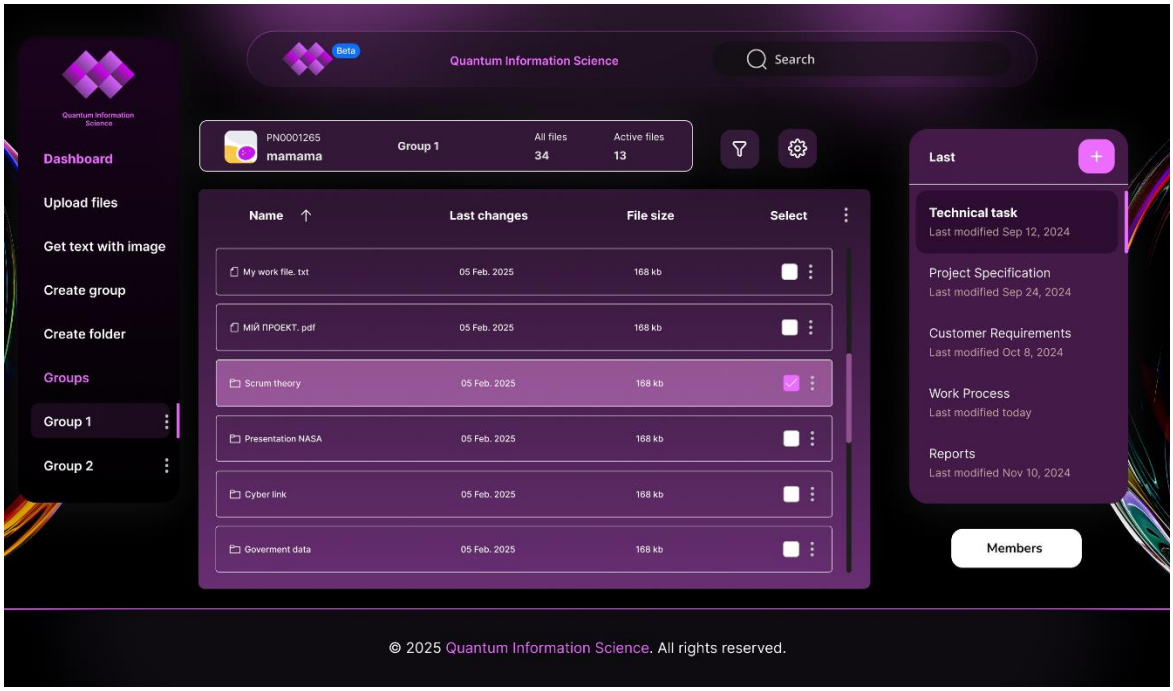


Рисунок 3.13 – Перегляд вмісту сховища групи та можливість змін

Після створення групи можна створювати вкладені папки, що впорядковують файли. Інтерфейс підтримує drag-and-drop, сортування, ієрархію папок і контроль доступу до кожного об’єкта (рисунок 3.14).

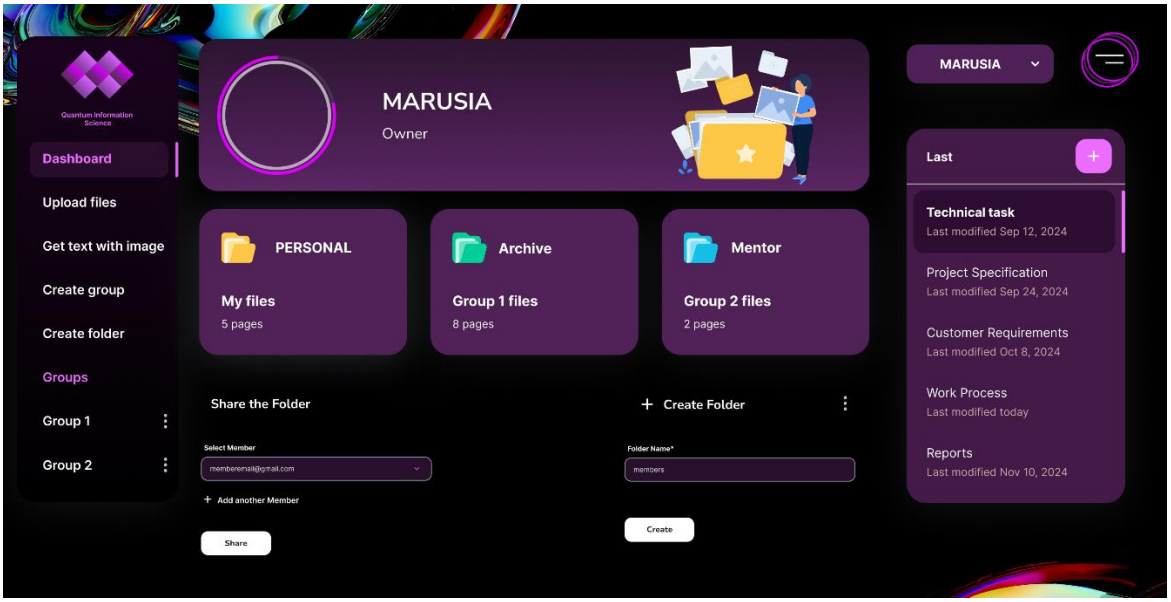


Рисунок 3.14 – Перегляд вмісту сховища групи та можливість змін

Наведена діаграма активності, нижче, в першому блоці ніціалізації та отримання даних, на рисунку 3.15, ілюструє логіку роботи React-компонента GroupUserModal.

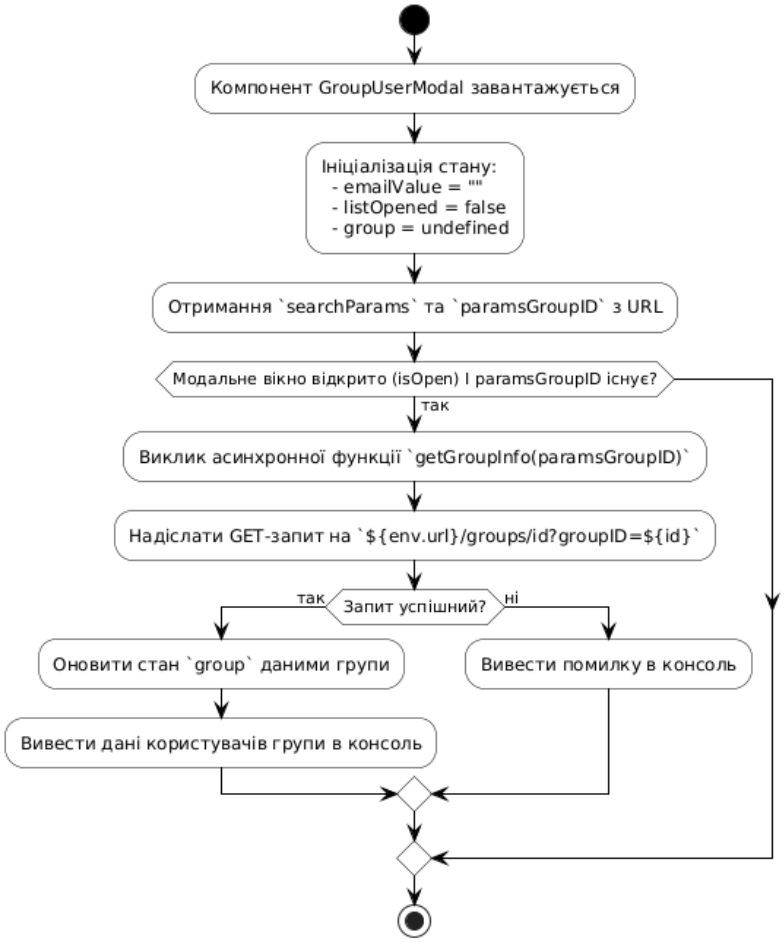


Рисунок 3.15 – Ініціалізація та отримання даних

Цей компонент є модальним вікном, призначеним для управління доступом користувачів до певної групи: додавання нових користувачів за електронною поштою та видалення існуючих учасників групи. На рисунку 3.16 показано процес рендерингу та взаємодії. Ця діаграма продовжує логіку компонента GroupUserModal, демонструючи процес рендерингу інтерфейсу та обробку взаємодій користувача для додавання та видалення учасників групи.

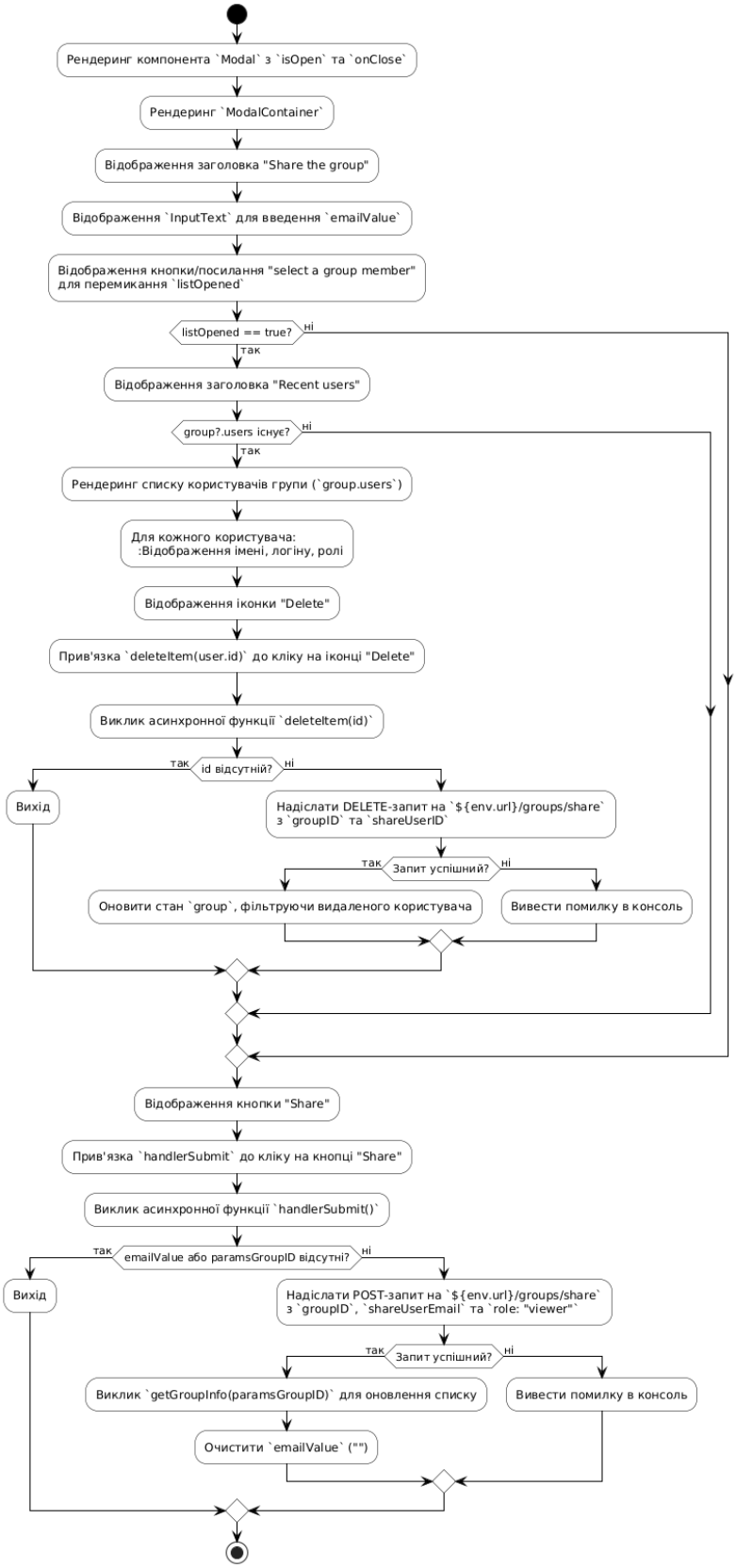


Рисунок 3.16 – Ініціалізація та отримання даних

Форма створення групи передбачає введення назви, короткого опису, визначення типу доступу, а також призначення перших учасників. Інтерфейс дозволяє додавати користувачівза ролями адміністратора чи учасника. Після створення групи автоматично формується базова структура з кореневою папкою та можливістю додаткової організації (рисунок 3.17).

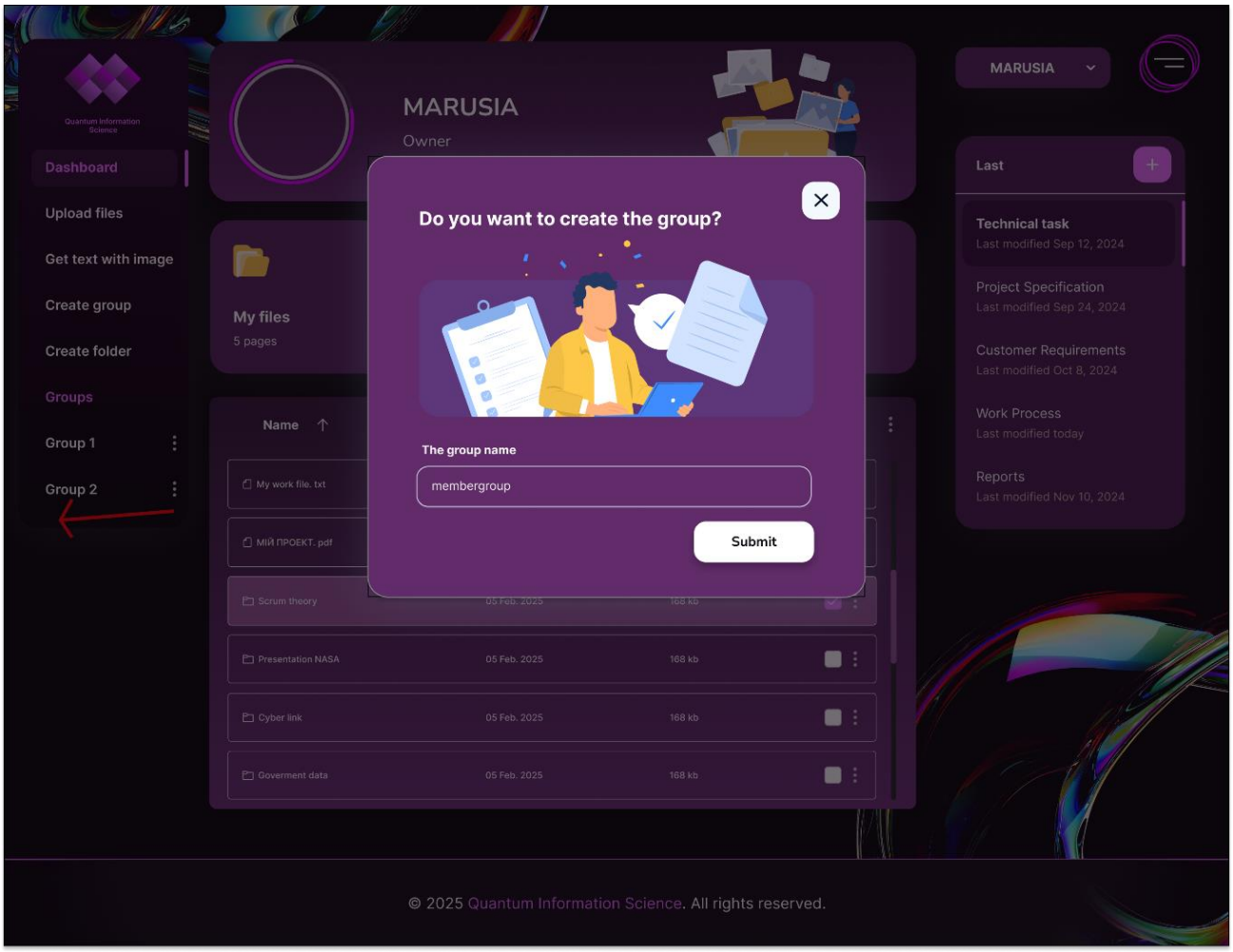


Рисунок 3.17 – Створення групи

Усі технології були ретельно підібрані з урахуванням вимог до сучасної безпеки, захисту персональних і корпоративних даних, а також ризиків найближчого майбутнього, пов’язаних із поширенням квантових обчислень.

3.4 Розробка модулю роботи з постквантовим шифруванням файлів

Функціональність завантаження побудована як покроковий процес із прозорим інтерфейсом. Спочатку користувач натискає кнопку «Завантажити файл». Відкривається форма вибору файлу, після чого файл тимчасово зберігається в оперативній пам’яті клієнтської частини (рисунок 3.18).

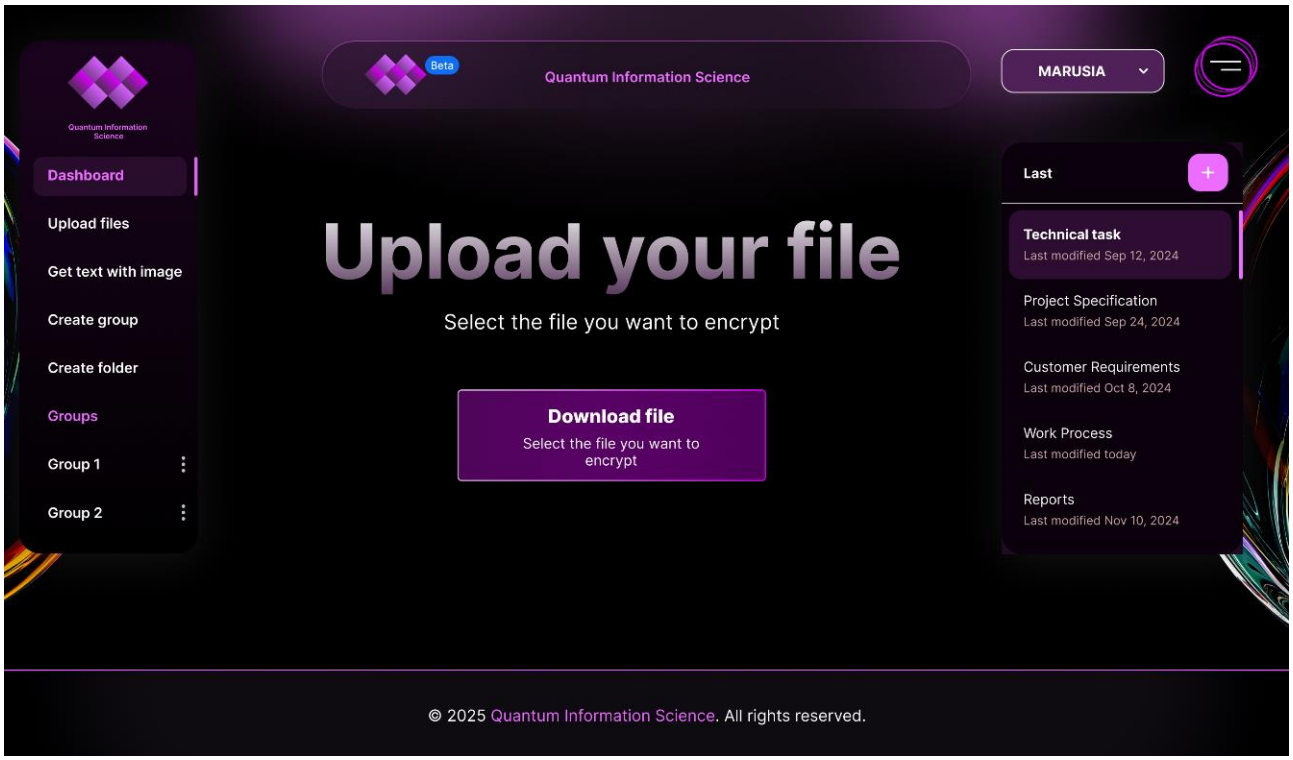


Рисунок 3.18 – Завантаження файлу у систему

Процес шифрування файлу передбачає створення симетричного ключа AES-256, який генерується безпосередньо на клієнтській стороні браузера. Після шифрування вмісту, цей ключ кодується з використанням постквантового алгоритму ML-KEM для безпечного зберігання і пересилання.

Отриманий зашифрований ключ зберігається разом із метаданими файлу, зокрема – часом створення, власником, групою та правами доступу (рисунок 3.19).

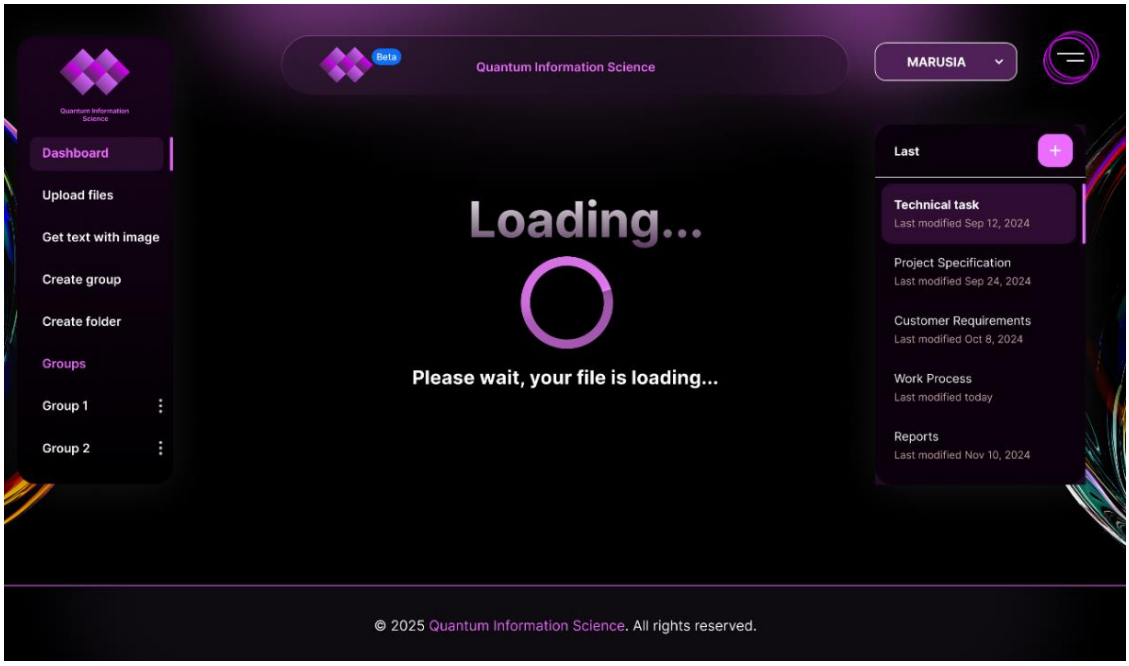


Рисунок 3.19 – Шифрування файлу

Отриманий зашифрований файл надсилається на сервер і завантажується у захищене сховище AWS S3. У базу даних записуються хеші, мітки часу, автор, група та шифровані ключі (рисунок 3.20).

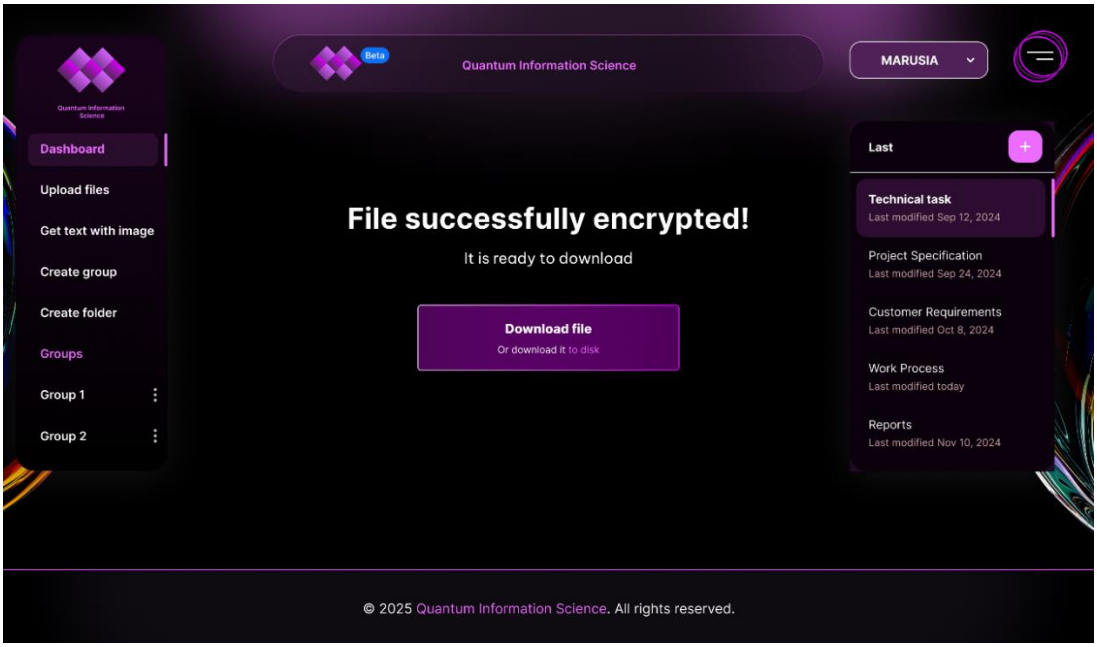


Рисунок 3.20 –Завантаження зашифрованого файлу

3.6 Висновки

У третьому розділі було здійснено аналіз і вибір інструментів для реалізації вебплатформи з квантово стійким захистом, зокрема інтегрованого середовища розробки, мови програмування та супутніх технологій. Було обґрунтовано доцільність використання мови TypeScript у поєднанні з платформою Node.js і фреймворком Next.js для побудови архітектури платформи. В якості середовища розробки було обрано Visual Studio Code як найбільш оптимальне для швидкої, масштабованої та захищеної веброзробки.

Також у цьому розділі було наведено порівняльний аналіз сучасних IDE, з урахуванням підтримки інструментів криптографії, гнучкості налаштувань, сумісності з технологіями веброзробки та рівня інтеграції з інфраструктурою безпеки. Крім того, було описано основні інтерфейси платформи, розроблено блок-схеми, моделі переходів користувача між модулями, зокрема: модуль авторизації, перегляд профілю, управління групами, завантаження і шифрування файлів, створення папок і груп. Кожен етап супроводжується відповідним рівнем безпеки та криптографічного захисту. Такий підхід дозволив сформувати цілісну структуру вебплатформи, яка забезпечує надійний рівень конфіденційності і зручність у користуванні.

4 ТЕСТУВАННЯ ПРОГРАМИ

4.1 Аналіз методів тестування програмного забезпечення

Тестування програмного забезпечення є основним компонентом життєвого циклу розробки програмного забезпечення, служить для підтвердження якості, надійності та відповідності кінцевого продукту вимогам користувачів. Він охоплює безліч методологій та методів, спрямованих на виявлення та виправлення дефектів на різних фазах життєвого циклу програмного забезпечення. Основними цілями тестування є перевірка точної функціональності програми, підвищення її операційної ефективності та зменшення ймовірності серйозних помилок під час її використання [26].

Методології тестування програмного забезпечення можна класифікувати за різними критеріями, такими як рівень тестування та методології, що використовуються в процесі тестування. Що стосується рівнів тестування, визначаються окремі категорії, включаючи модульне тестування, яке передбачає ізольоване вивчення окремих модулів або компонентів системи; інтеграційне тестування, яке зосереджується на оцінці взаємодії між взаємопов'язаними модулями; тестування системи, яке передбачає оцінку всієї системи як цілісного блоку для підтвердження її відповідності визначеним вимогам; та тестування прийняття користувачів, проведене кінцевими користувачами або клієнтами для забезпечення того, щоб система відповідала їхнім очікуванням та вимогам.

Статичні методології тестування включають перевірку коду, що передбачає перевірку коду експертними розробниками для виявлення помилок та недоліків; статичний аналіз, який використовує спеціалізовані інструменти для оцінки коду без його виконання; та перевірку та перевірку, які є формалізованими процедурами, спрямованими на перевірку коду, проектної документації та інших артефактів.

Динамічні методи тестування включають тестування у білій коробці, яке зосереджується на дослідженні внутрішньої структури або функціональності програми, коли тестер має доступ до вихідного коду; тестування чорної скриньки, яке оцінює функціональність програми без розуміння її внутрішньої роботи, тим самим полегшуючи оцінку системи з точки зору користувача, забезпечуючи відповідність встановленим вимогам та очікуванням; і тестування сірої коробки, яке представляє гібридний підхід, який об'єднує елементи обох попередніх методологій, де тестер частково знає внутрішню архітектуру системи.

Тестування програмного забезпечення стикається з численними проблемами, включаючи виявлення прихованих дефектів, які можуть проявлятися виключно за певних умов, а також значні витрати часу та ресурсів, враховуючи, що тестування може бути надзвичайно трудомістким та ресурсним. Для пом'якшення цих проблем можуть бути використані різні стратегії, такі як формулювання комплексних стратегій тестування, які інтегрують різноманітні методи тестування для оптимізації покриття; впровадження автоматизованих інструментів для підвищення ефективності, тим самим скорочуючи як час, так і витрати.

Дослідження методологій тестування програмного забезпечення показує, що ефективне тестування має першорядне значення для гарантування якості та надійності програмних продуктів. Розумний вибір відповідних методологій, таких як тестування у чорній коробці, полегшує оптимізацію процесів тестування та підвищує ймовірність виявлення більшості дефектів до доставки продукту кінцевому споживачеві.

4.2 Тестування розробленого програмного застосунку

Для тестування продуктивності веб-додатків існує різноманітні інструменти, і одним з найефективніших сервісів є WebPageTest.org. Цей вільний онлайн інструмент дозволяє проводити детальний аналіз продуктивності веб-сторінок.

Основні переваги використання ресурсу включають можливість побудови графіків завантаження даних, що надає високий рівень деталізації та дозволяє точно аналізувати характеристики завантаження сторінок. Інструмент також допомагає виявляти проблеми з продуктивністю та надає рекомендації для їх вирішення.

При тестуванні вебплатформи на продуктивність було використано сервіс WebPageTest.org було обрано розташування Virginia USA та браузер Chrome v119(рисунок 4.1).

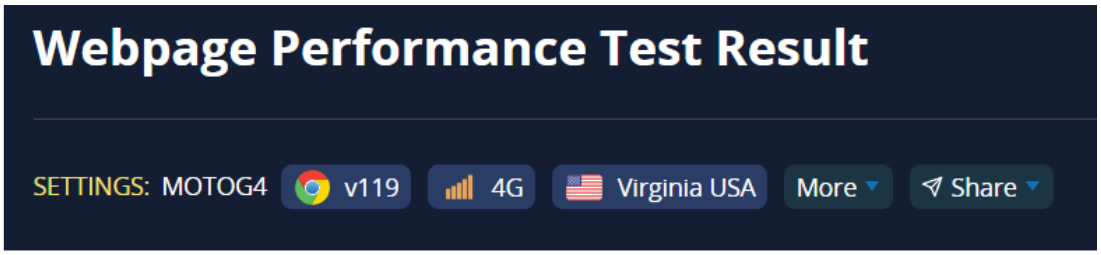


Рисунок 4.1 – Умови тестування

В результаті аналізу всіх результатів оцінки продуктивності сайту можна зробити висновок, що цей сайт зайняв небагато часу для підключення та доставки початкового коду. Він почав відтворювати вміст із невеликою затримкою. А його найбільший вміст відтворюється пізніше, ніж ідеально.

Нижче, на рисунку 4.2 детально зображено показники ефективності сховища.

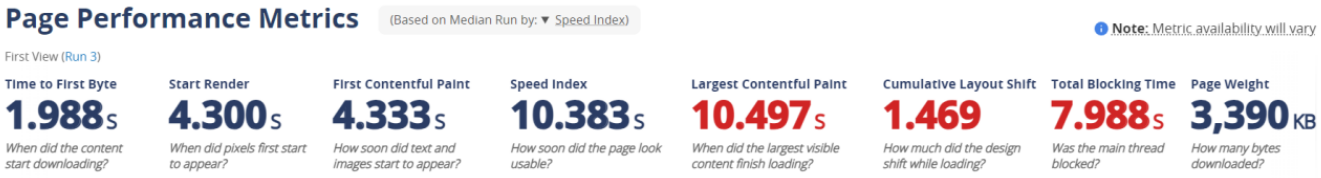


Рисунок 4.2 – Показники ефективності сховища

Час до першого байту вказує на час, який сервер потребує для відповіді на запит. 1,988 с - це досить швидкий час, що вказує на ефективну роботу сервера.

Запуск візуалізації вказує на момент, коли браузер починає відображати контент. 4,3 с - це прийнятний результат, але можливість покращення завжди існує.

Індекс швидкості визначає, наскільки швидко відбувається відображення контенту. 10,383 пд - це досить низьке значення, що свідчить про ефективність завантаження.

Наступинй показник вказує на час завантаження основного видимого контенту. 10,497 с - це прийнятно, але, можливо, варто розглянути оптимізаційні заходи.

Зміни в конструкції вказують на те, що структура сторінки динамічно змінюється під час завантаження, що може включати асинхронне завантаження ресурсів або динамічні модифікації DOM. Результат 1,469 вказує на помірно низький рівень змін в конструкції. З одного боку, це може вказувати на певну оптимізацію, але з іншого – може виникати питання щодо плавності та стабільності відображення для користувачів.

При аналізі загального часу блокування була отримана відчутна затримка. Результат 1,469 вказує на помірно низький рівень змін в конструкції. З одного боку, це може вказувати на певну оптимізацію, але з іншого – може виникати питання щодо плавності та стабільності відображення для користувачів.

Вага сторінки вказує на обсяг ресурсів, які потрібно завантажити. 3390 Кб - це значна вага, і може бути доцільним зменшити її оптимізацією зображень, стисненням файлів тощо. Це є прийнятним результатом.

Завдяки інструменту WebPageTest.org вдалося виявляти проблеми з продуктивністю та врахувати його рекомендації при їх усунені.

4.3 Тестування роботи квантової моделі машинного навчання

Виконаємо тестування роботи квантової моделі для виявлення аномалій у поведінці користувачів. Показники глибини нейронної мережі є дуже важливим

елементом для її подальшого навчання. Виконаємо дослідження роботи при K = 2, 4, 6, 8. Продуктивність квантової машинної моделі показана за допомогою середніх функцій перехресної ентропії та втрат (рисунок 4.3)

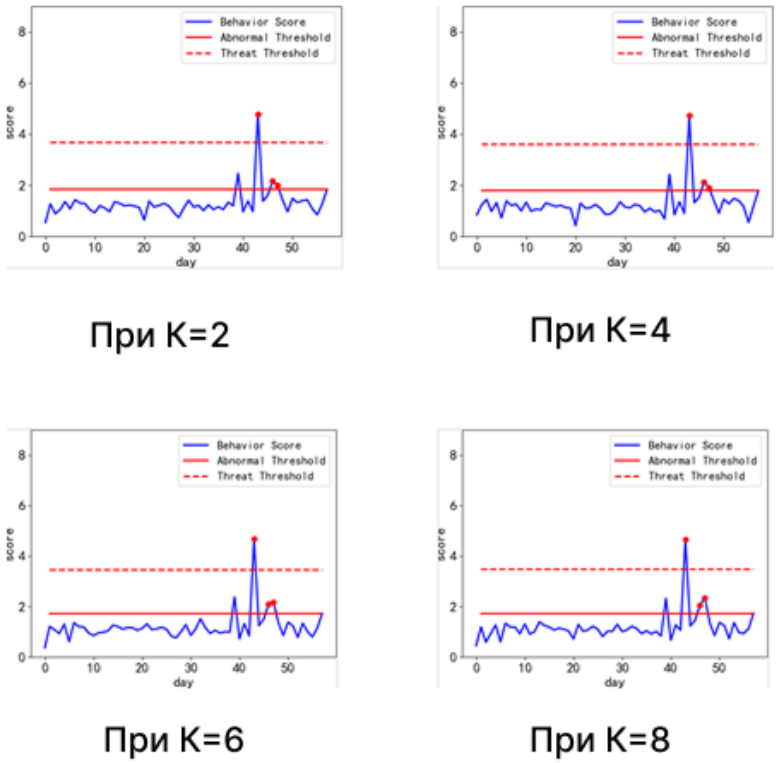


Рисунок 4.3 – Тестування глибини нейронної мережі на оцінку загроз

Результати показують, що в міру прогресування епох перехресна ентропія квантового генератора спочатку швидко зменшується, лише зменшуючись після досягнення певної епохи, що означає тенденцію до конвергенції оптимізації квантового генератора. Конвергенція залишається нижчою для K = 2,4, але надзвичайно приємною для K = 6,8. Одночасно стає очевидним, що швидкість зменшення перехресної ентропії квантового генератора прискорюється зі

збільшенням шарів K. Більші глибини сприяють швидшій та ефективнішій конвергенції для квантового генератора.

Втрати навчального генератора спочатку зростають, перш ніж різко впасти. І навпаки, втрати навчального генератора демонструють тенденцію спочатку зниження, а потім зростання. Зрештою, і навчального генератора, і навчального дешифратора вирівнюються до подібних значень і стабілізуються з часом, що свідчить про те, що зразки, створені квантового генератора, тепер схожі на автентичні зразки. Більше того, збільшення шарів корелює з більш швидким підйомом як для навчального генератора, так і для навчального дешифратора.

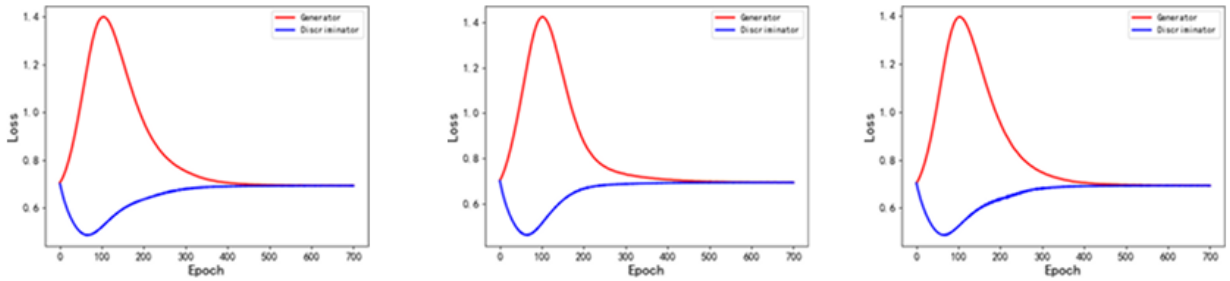
У цьому дослідженні були вивчені результати функціонування системи для трьох різних користувачів з різними обсягами вхідних даних. Зокрема, перший користувач має 300 днів спостережень, з яких 200 днів були використані для навчального етапу, а 100 – для тестування. Другий користувач представлений колекцією з 160 днів, де 100 днів використовувалися для навчання, а інші 60 – для тестування. Для третього користувача було доступно 175 днів даних, з яких 100 днів виділено для навчання моделі, а 75 – для оцінки її продуктивності. У квантовій моделі було встановлено кількість квантових шарів параметричної квантової схеми дорівнює K = 8. Результати оцінки поведінки тестових даних показані на рисунку 4.4.

	Користувач 1	Користувач 2	Користувач 3
К-сть днів для навчання	200	100	100
К-сть днів тестування	100	60	75
Точність виявлення аномалій	97,98%.	98,28%	97,30%.

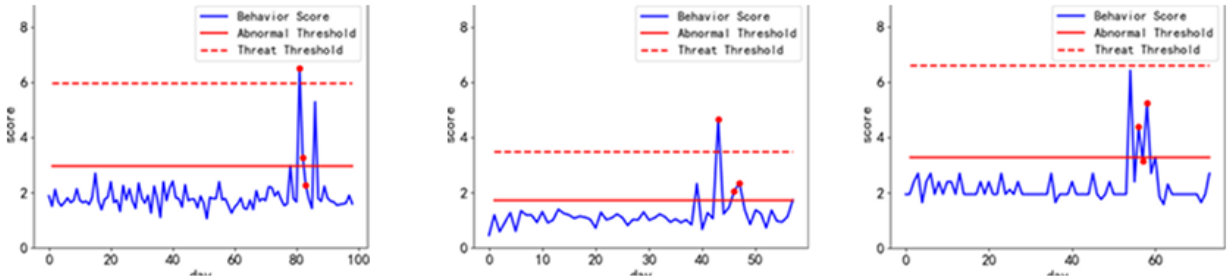
Рисунок 4.4 – Табличне представленнчя умов тестування та результатів точності

Дослідження виявило, що втрати навчальний генератор і дискримінатор демонструють подібну ситуацію у всіх трьох користувачів, що вказує на узгодженість навчання моделі.

Згідно з результатами тестування, можна зазначити, що система показує високу ефективність у визначенні аномальної активності. Практично всі аномалії в поведінці користувачів були точно визначені. Для першого, другого та третього користувача, точність моделі за показником квантової машинної моделі для оцінки поведінки користувачів дорівнювала відповідно 97,98%, 98,28% та 97,30% (рисунок 4.5).



Втрати квантового генератора і дешифратора в порівнянні з епохою для різних користувачів.



Оцінка загрози у порівнянні з днями для різних користувачів. Червоні точки символізують реальну ненормальну поведінку, яка відбувається в цей день.

Рисунок 4.5 – Тестування на рівні епох та рівня небезпеки на прикладах трьох різних користувачів

Ці дані підтверджують можливість комбінування існуючої змагальної мережі і розробленої моделі для ефективної генерації несправжніх даних, що потрібні для

створення узагальненої моделі нормальної поведінки. Така модель використовується для ідентифікації внутрішніх аномалій у поведінці користувачів з високою точністю та надійністю

4.4. Тестування процесу шифрування файлів

Результати тестування процесу шифрування файлів уже в робочій системі проілюстровано на рисунку 4.6. Відображеннчя ключів до шифрування та після свідчать про успішну обробку та хеш значення згенерованого об’єкта підтверджують коректну роботу модуля.

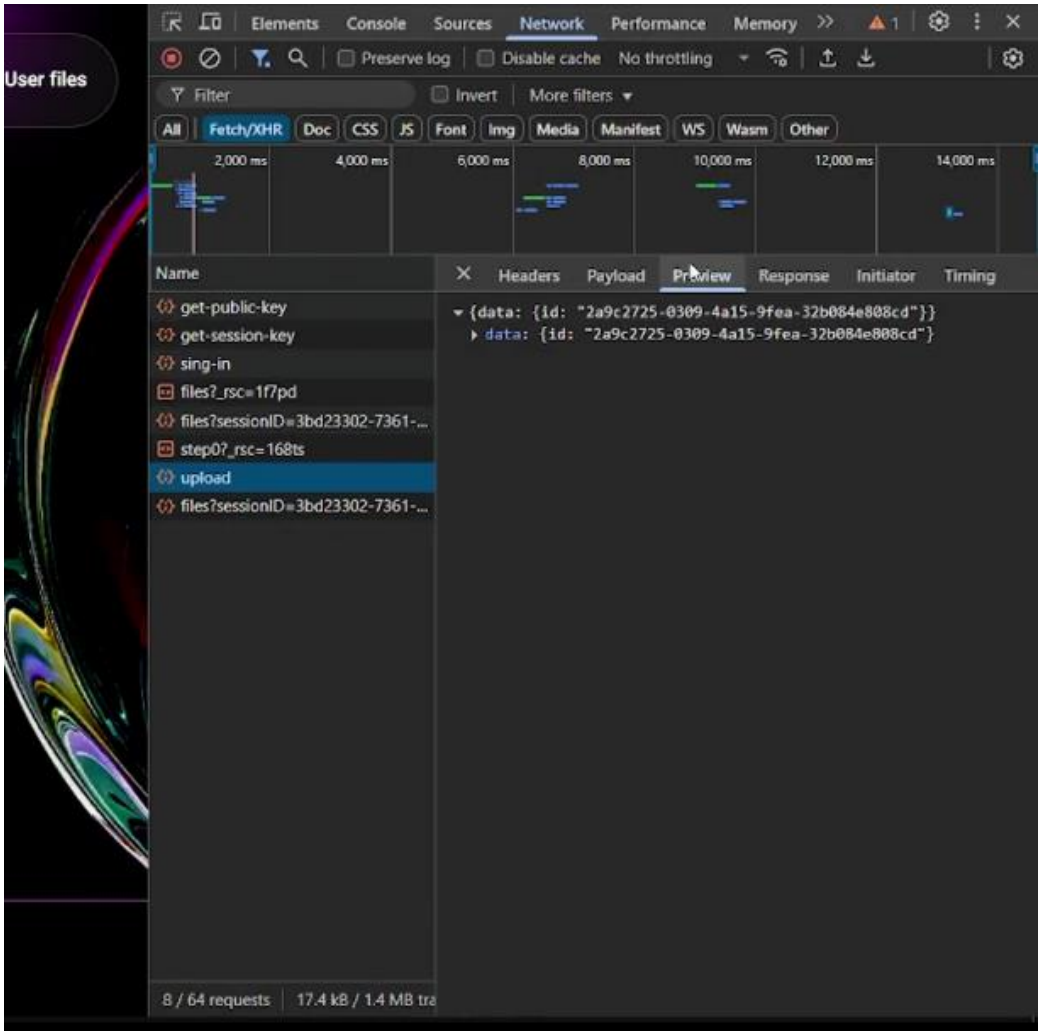


Рисунок 4.6 – Результат шифрування файлів

Нижче, на рисунку 4.7 показано постквантовий захист сховища після завантаження файлів. Зображення демонструє стан захищеного сховища після вивантаження файлу. Було перевірено, що файл потрапив до призначеного каталогу, зберігаючи зашифровану структуру та унеможливлюючи несанкціоноване відкриття без розшифрувального ключа.

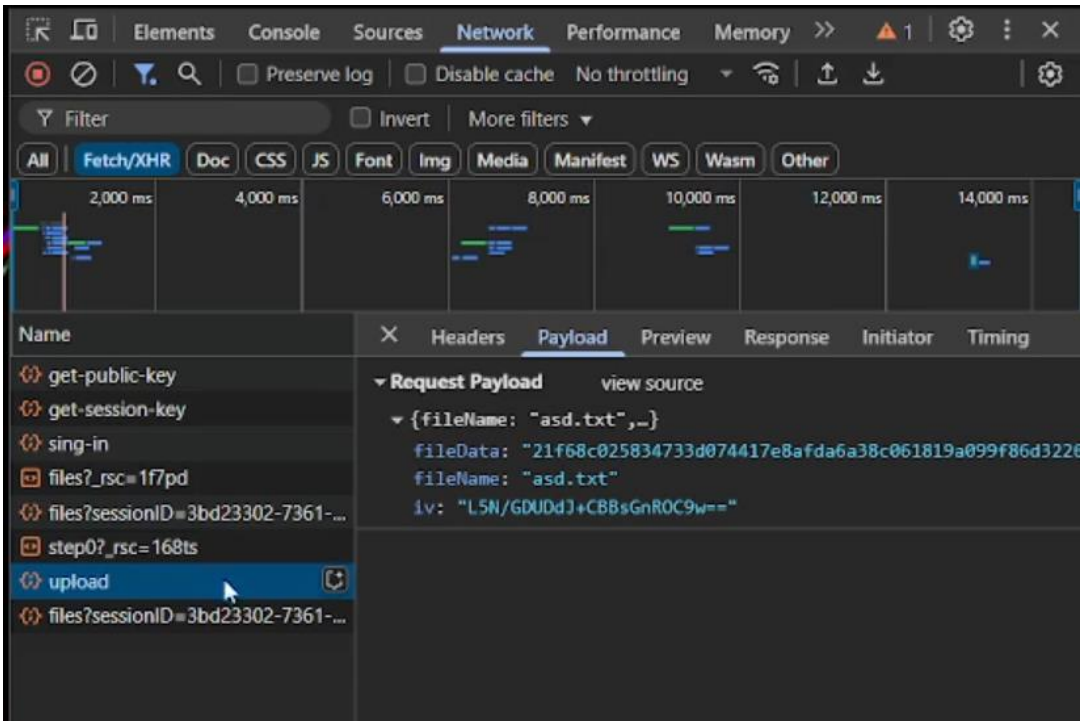


Рисунок 4.7 – Результат захисту сховища після вивантаження файлу

Нижче, на рисунку 4.8 показано як відбувається процес шифрування, також представлено архітектуру взаємодії між клієнтом і сервером, використані криптографічні методи AES-256, ML-KEM та мережевий протокол HTTP(S), через який забезпечено транспортування захищених даних. Хоча HTTP сам по собі не гарантує шифрування, у реалізації всі передані дані проходять через TLS-обгортку, що забезпечує непроникність каналу. Тут відбувається демонстрація передачі даних, оскільки з’єднання з сервером відбувається через http протокол а не https, ми можемо переглянути вміст повідомлення, тіло запиту є зашифрованим.

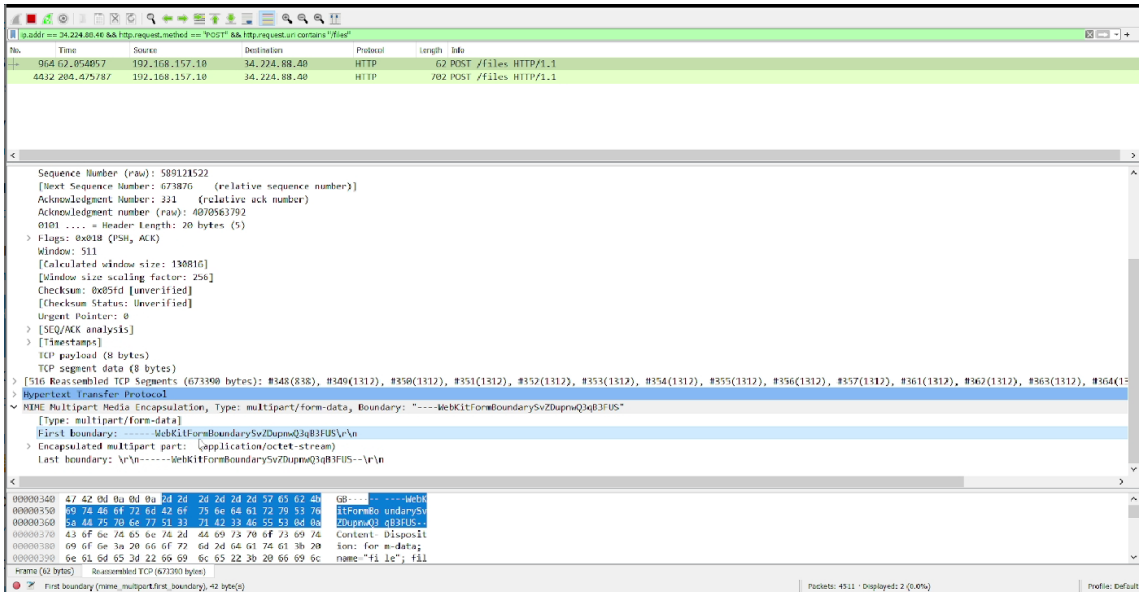


Рисунок 4.8 – Результат процесу шифрування

4.5. Тестування роботи платформи

Першим кроком клієнт робить запит на отримання налаштувань чи увімкнено з’єднання з використанням шифрування, дане з’єднання можна вимкнути тільки коли сервер знаходиться в DEV режимі (рисунок 4.9)

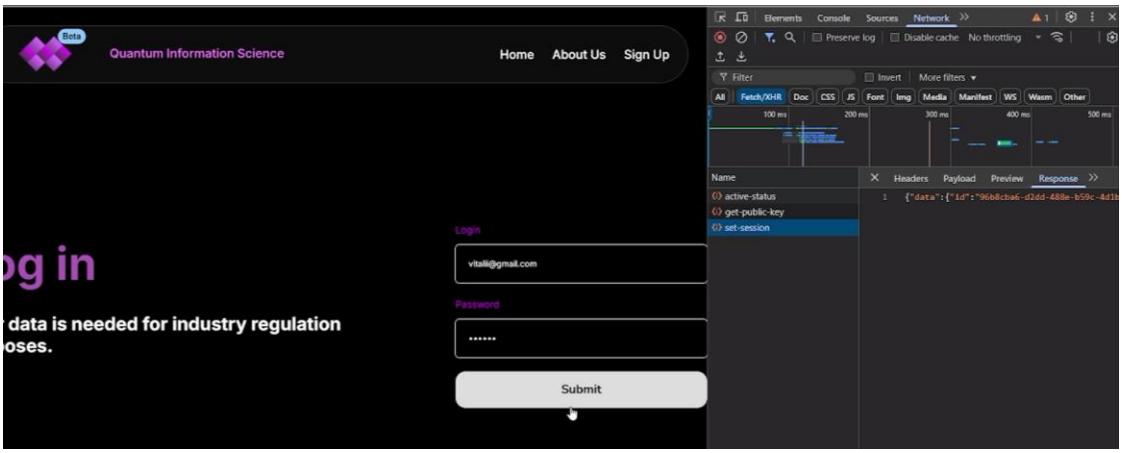


Рисунок 4.9 – Результат захисту сховища після вивантаження файлу

Після отримання налаштувань клієнт робить запит на отримання публічного ключа серверу та після встановлюється ID сесії. Коли клієнт ввів логін та пароль,

проте в нього увімкнено MFA, він має ввести Time-based one-time password та отримує токен який дозволяє серверу зрозуміти, що користувач пройшов автинтифікація. Якщо MFA не увімкнено, TOTP не вимагаєтсья (рисунок 4.10).

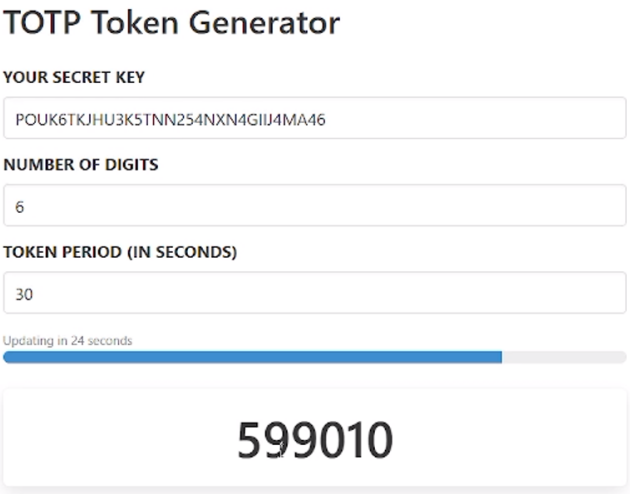


Рисунок 4.10 – Отримання у генераторі тимчавсового ключа

На рисунку 4.11 користувач вводить отриманий TOTP-код у спеціальне поле на сторінці входу. Усі передані значення додатково шифруються в процесі передачі, а перевірка відбувається на сервері з використанням алгоритмів часу життя та контрольного хешу.

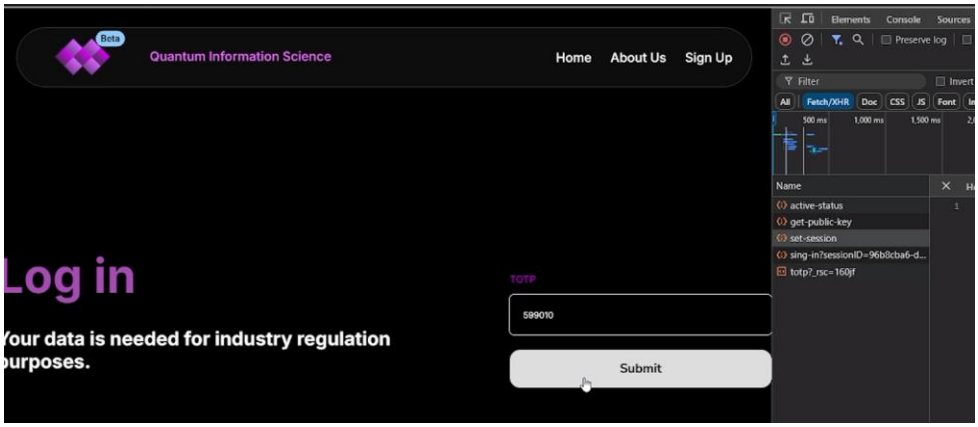


Рисунок 4.11 – Введення ключа у систему

Коли користувач отримав доступ до свого Dashboard, він повинен ввести Personal AES-256 KEY. AES це криптографічний алгоритм, за умови довжини ключа в 256 біт, він є стійким до атак за допомогою квантових комп'ютерів (рисунок 4.12).

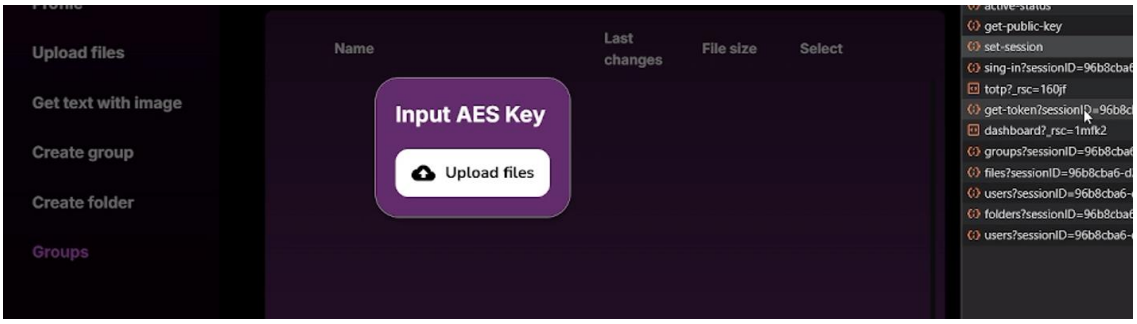


Рисунок 4.12 – Переіхд на дешборд з закликом введення Personal AES KEY

Далі користувач може завантажити персональний AES-ключ з локального сховища або скопіювати з безпечного зовнішнього джерела. Після перевірки форматування та валідності, ключ зберігається локально в середовищі браузера із застосуванням політики нульового розголошення.4.13.

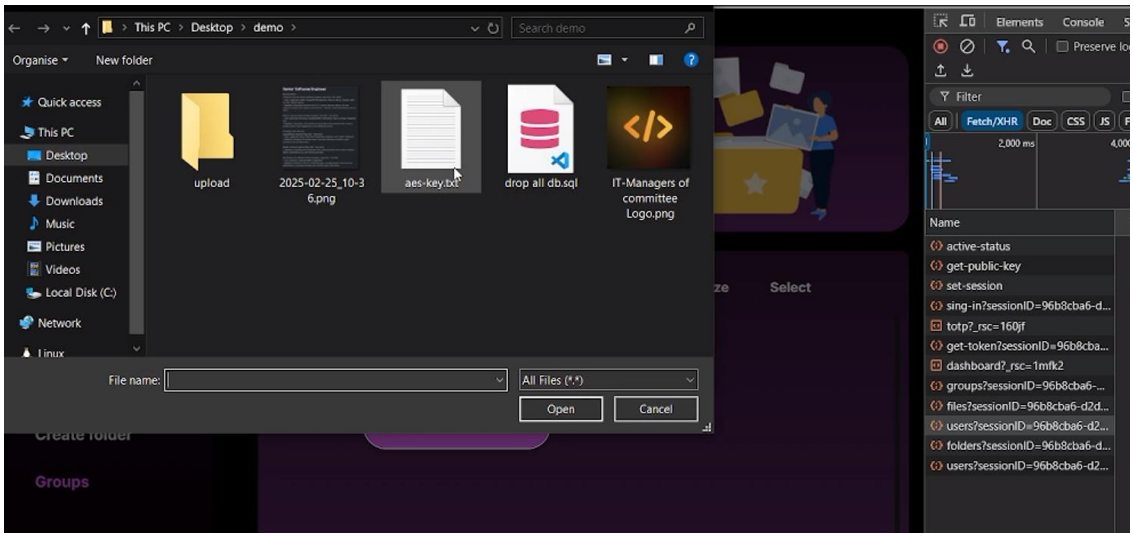


Рисунок 4.13 – Завантаження Personal AES-256 KEY

На цьому етапі користувач обирає файл для завантаження. У даному випадку тестування проводилось з PNG-зображенням, яке буде використовуватись як еталон для перевірки ефективності шифрування. (рисунок 4.14).

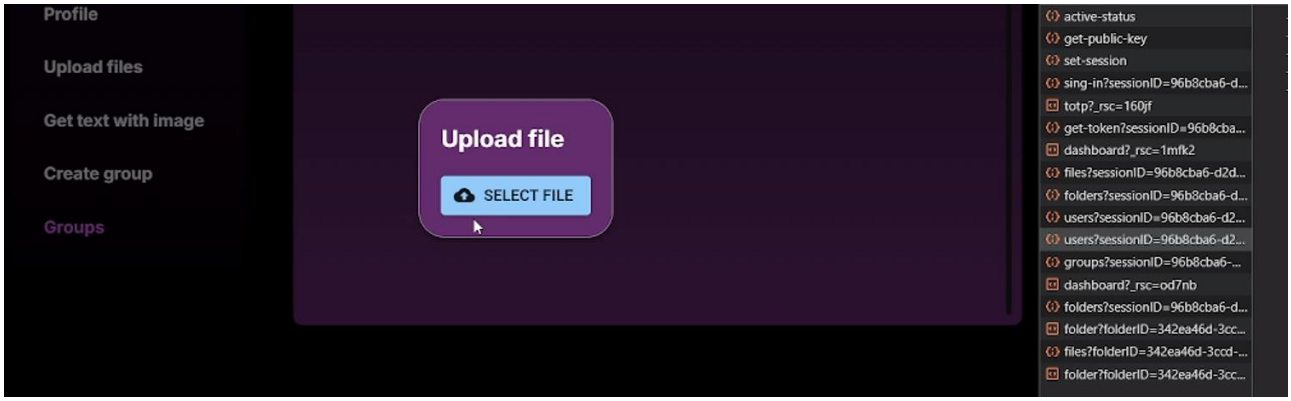


Рисунок 4.14 – Заклик до завантаження файлу PNG формату у сховище

Після завантаження файл відображається в інтерфейсі системи. У правій панелі розробника можна відстежити, як змінюються внутрішні ключі, що підтверджує запуск процесу шифрування і його завершення (рисунок 4.15).

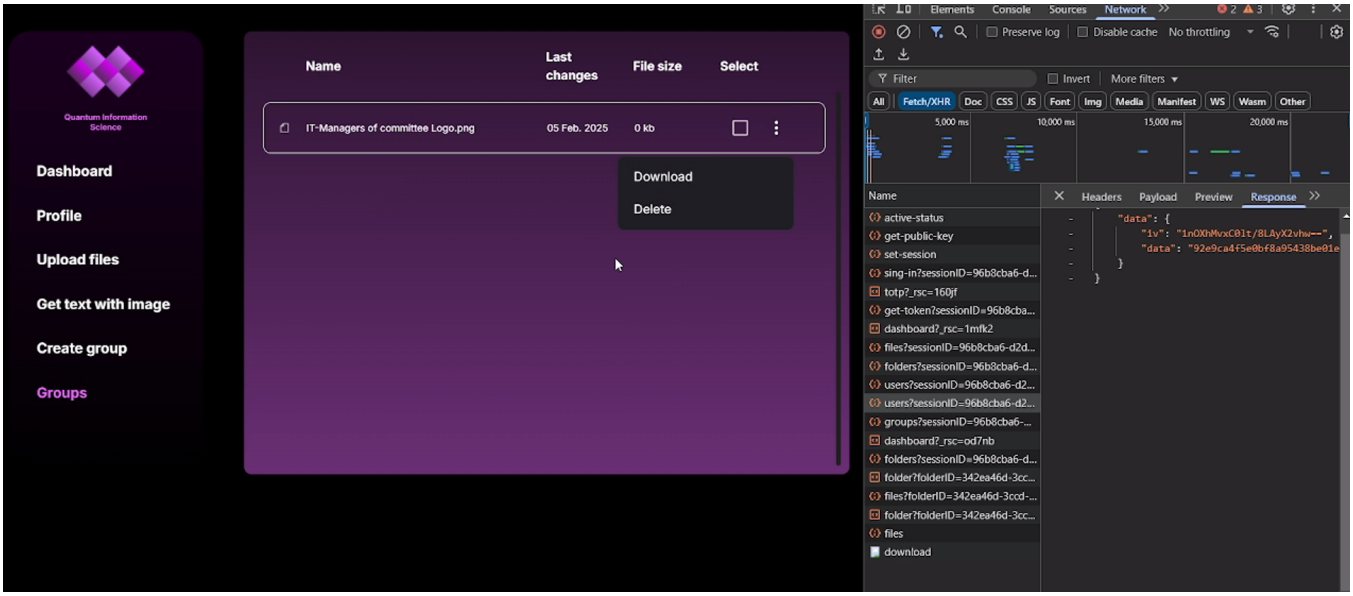


Рисунок 4.15 Завантажений файл PNG формату у сховище

На рисунку 4.16 показано підтвердження, що файл, пройшовши процес шифрування і зберігання, при розшифруванні повністю відповідає оригіналу, що доводить цілісність і надійність криптографічного процесу.

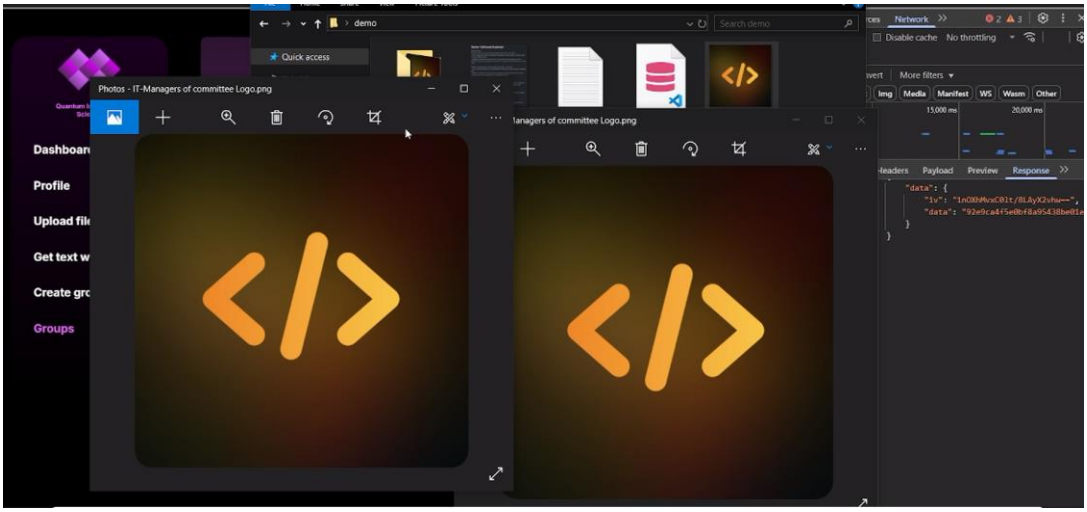


Рисунок 4.16 – Результат порівняння віхдного і вихідного зашифрованого зображення

Також користувач має змогу створити групу для спільної роботи. Створення починається з натискання кнопки, яка відкриває форму з базовими параметрами (рисунок 4.17).

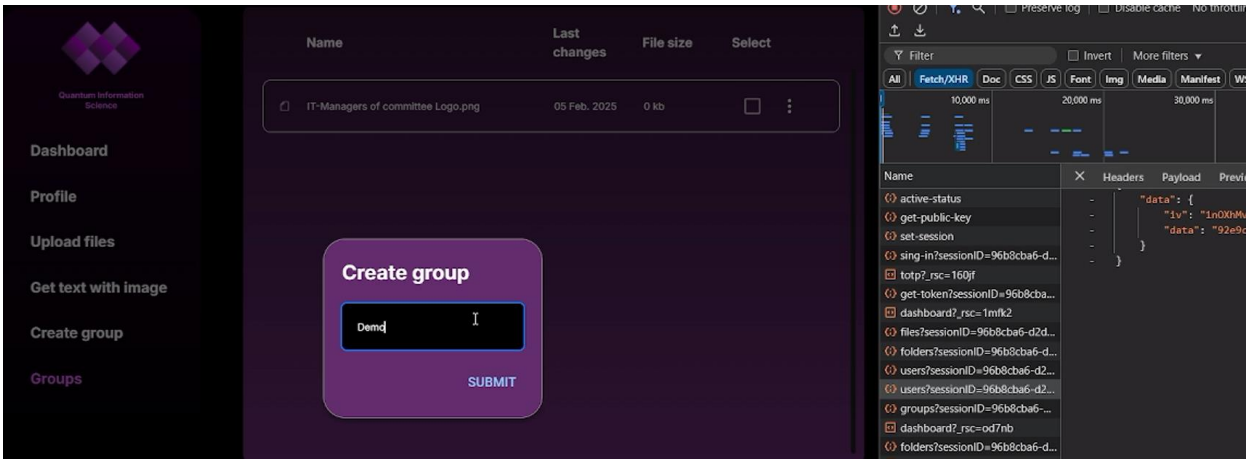


Рисунок 4.17 – Створення групи

У формі потрібно вказати назву, опис групи, рівень доступу: публічна, приватна, а також запрошення користувачів. Після підтвердження, група зберігається в базі даних платформи (рисунок 4.18).

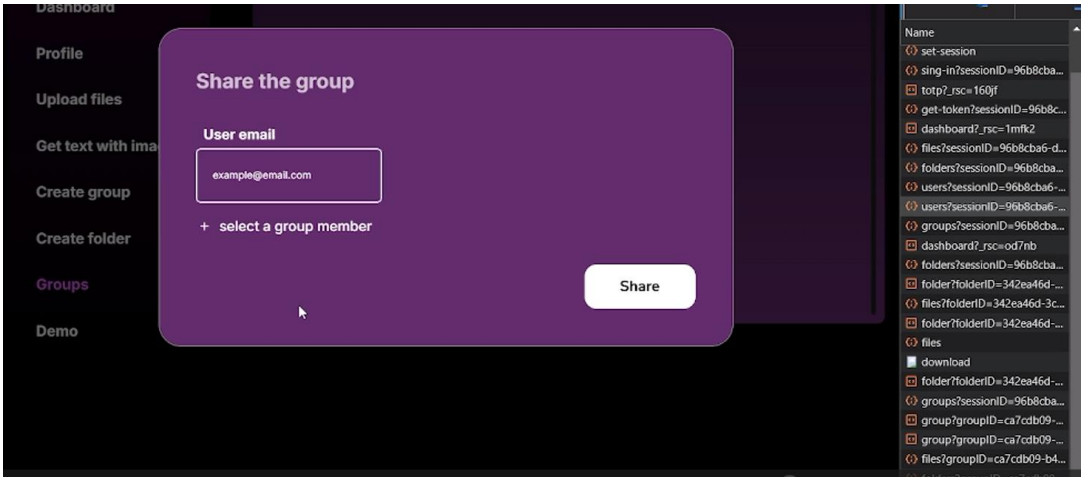


Рисунок 4.18 – Параметри які необхвдно ввести при створенні групи

Процес підтвердження і формування нової групи на стороні клієнта виконано. Супроводжується повідомленням про успішне створення та перенаправленням користувача до нової групи. А тепер отримаємо текст з картинки за допомогою бібліотеки tesseract (рисунок 4.19)

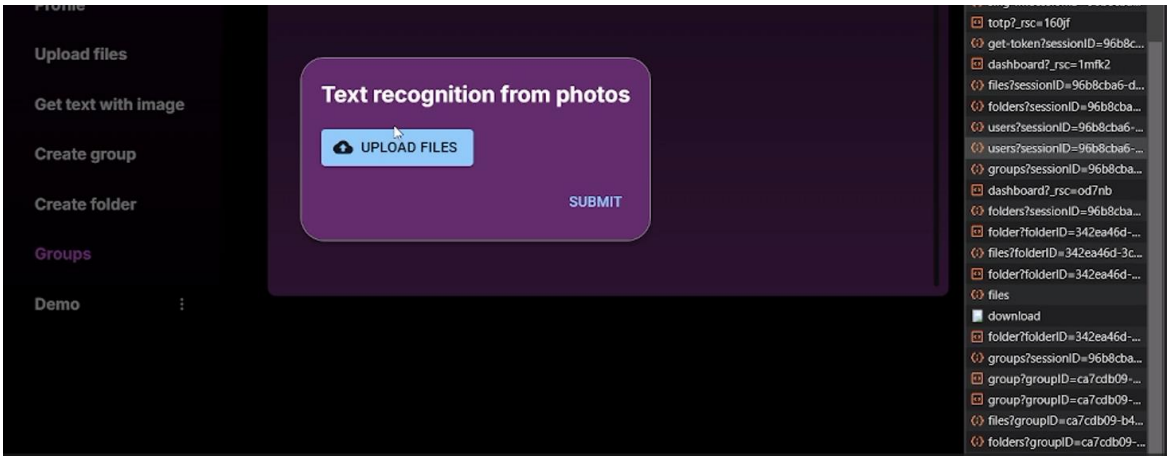


Рисунок 4.19 – Створення групи

Платформа дозволяє проводити обробку зображень за допомогою OCR-бібліотеки Tesseract. Користувач завантажує файл з графічним текстом, після чого відбувається розпізнавання та вивід результату. Було встановлено, що розпізнаний текст повністю відповідає оригіналу (рисунок 4.20).

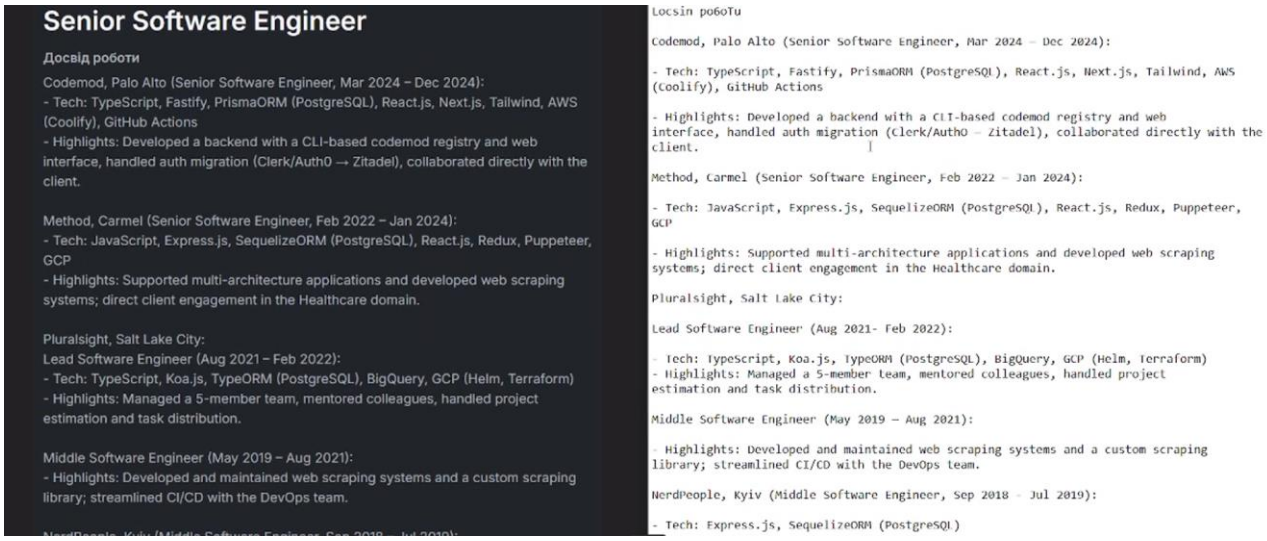


Рисунок 4.20 – Порівнняння вхідного та вихідного файлів

Також продемонструємо налаштування користувача, в яких можна змінити пароль, налаштувати MFA та змінити ім’я. Інтерфейс налаштувань дозволяє змінити пароль, активувати або вимкнути багатофакторну автентифікацію, оновити особисті дані (рисунок 4.21).

Усі зміни потребують повторної верифікації через TOTP, що підвищує рівень безпеки профілю. Цей модуль також містить функцію «аварійного відновлення доступу» через контрольне питання та резервний криптографічний токен, який генерується при реєстрації. Усі дії в налаштуваннях супроводжуються детальними повідомленнями про успішність або помилки виконання операцій, що забезпечує прозорість і зручність керування безпекою облікового запису користувача.

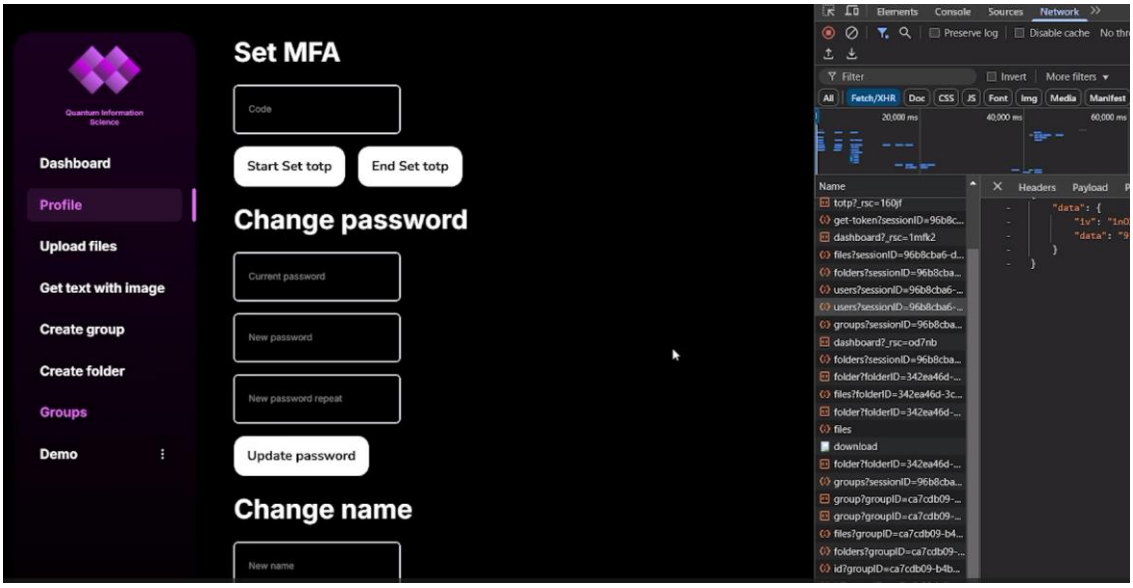


Рисунок 4.21 – Налаштування користувача

Усі перевірені сторінки платформи відобразили очікувані результати. Було отримано повну відповідність між вхідними параметрами й вихідними значеннями, включно з контрольними хешами. Жодних збоїв або відхилень від стандартної логіки роботи не виявлено.

При проведенні тестування отримана повна відповідність вхідних даних і вихідних результатів. Помилок при роботі програми не виявлено і це підтверджує нормальний режим роботи. У результаті доведено повну працездатність вебплатформи та її відповідність поставленому технічному завданню.

4.6 Розробка інструкції користувача вебплатформи

На відміну від мобільних застосунків, вебплатформа "Quantum Information Science" не потребує окремого встановлення або завантаження файлів формату .apk. Доступ до платформи здійснюється безпосередньо через веббраузер за захищеним HTTPS-з'єднанням, що забезпечує сумісність з більшістю сучасних пристроїв і операційних систем без додаткових дій з боку користувача.

Перед початком роботи користувачеві достатньо перейти за офіційною URL-адресою платформи. Система автоматично перевіряє відповідність клієнтського пристрою базовим технічним вимогам. Якщо браузер не підтримує необхідні криптографічні протоколи або має застарілу версію JavaScript-рушія, користувач отримує відповідне повідомлення.

Мінімальна та рекомендована конфігурація для доступу до вебплатформи наведена в таблицях 4.1 та 4.2.

Таблиця 4.1 – Мінімальна конфігурація:

Тип процесора	Двоядерний, від 1.4 GHz
Об’єм оперативної пам’яті	2 ГБ
Браузер	Google Chrome 96+, Firefox 95+
Операційна система	Windows 10, macOS 10.14, Linux
Додаткові вимоги	Підтримка TLS 1.3, WebAssembly

Таблиця 4.2 – Рекомендована конфігурація:

Тип процесора	.4GHz Octa-core Qualcomm Snapdragon 888
Тип процесора	4-ядерний, від 2.5 GHz
Об’єм оперативної пам’яті	4 ГБ
Браузер	Google Chrome 120+, Edge 115+
Операційна система	Windows 11, macOS Ventura, Ubuntu 22.04
Додаткові вимоги	Підтримка AES-NI, HW Acceleration

Після входу на платформу система автоматично перевіряє, чи користувач авторизований. Якщо ні – відкривається сторінка автентифікації з можливістю

входу або реєстрації. Без успішної авторизації доступ до будь-якого внутрішнього функціоналу, зокрема перегляду файлів, груп, завантаження документів або взаємодії з іншими учасниками – блокується.

Після проходження автентифікації користувач переходить на головну панель керування. Звідси можна переглядати доступні групи, підключатися до них, створювати нові, а також використовувати інші функції платформи, такі як завантаження/шифрування файлів, управління правами доступу та інше.

На відміну від мобільного застосунку, користувачі не обмежені лише однією ОС, а можуть взаємодіяти з платформою з будь-якого пристрою, що підтримує сучасний веббраузер. Це забезпечує більшу універсальність, зменшує бар'єри входу та не вимагає оновлення застосунку вручну – усі зміни на платформі застосовуються одразу після їхнього релізу розробником.

4.7 Висновки

У четвертому розділі бакалаврської кваліфікаційної роботи було здійснено детальне тестування розробленого методу та програмних засобів для вебплатформи «Quantum Information Science».

Крім того, розроблено детальну інструкцію для користувачів, яка описує процес переходу та рекомендаційні вимоги. Інструкція також включає кроки реєстрації в системі, методи взаємодії з медичними фахівцями через застосунок.

Результати тестування підтвердили, що реалізовані криптографічні алгоритми, архітектурні рішення та мережеві механізми відповідають встановленим критеріям безпеки, стабільності та функціональності, передбаченим технічним завданням для вебплатформи з квантово стійким захистом. Тестування продемонструвало повну відповідність вхідних і вихідних даних, відсутність критичних помилок, а також безпечну роботу механізмів шифрування та розшифрування на клієнтському та серверному рівнях.

Таким чином, вебплатформа «Quantum Information Science» демонструє значний потенціал для практичного застосування в умовах підвищених вимог до конфіденційності даних. Її функціональність, що включає підтримку постквантових алгоритмів, гнучке управління доступом та адаптивний користувацький інтерфейс, дозволяє забезпечити безпечне зберігання, обмін та контроль даних у масштабованому середовищі, незалежно від типу пристрою користувача. Отже, платформа є цілісною для організацій та установ, які прагнуть захистити критично важливу інформацію в умовах сучасних і майбутніх загроз.

ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи було розроблено вебплатформу «Quantum Information Science», яка забезпечує постквантове шифрування даних, захищене зберігання від атак квантових комп'ютерів та модулі для виявлення аномального поведінки користувачів. Створена система забезпечує безпечне зберігання, обмін та управління доступом до конфіденційних файлів в організаціях, компаніях та державних установах.

Здійснено порівняльний аналіз аналогів, виявлено їхні недоліки та на цій основі підтверджено доцільність власної розробки.

Створено структуру вебплатформи, реалізовано модулі взаємодії користувача з сервером, систему зберігання та передачі ключів, яка захищена ML-KEM і AES. Архітектуру платформи створено за допомогою: Node.js, Next.js, TypeScript, PostgreSQL, GoogleAuth, JWT, Docker, AWS, Swagger.

Розроблено метод квантової моделі машинного навчання для виявлення аномального поведінки користувачів, що дозволяє виявляти потенційні загрози і підвищує загальний рівень безпеки середовища.

Створено інтерфейс платформи з акцентом на зручність використання, доступність для користувачів, з реалізацією багаторівневої авторизації та візуальними елементами взаємодії.

В результаті виконання завдань було спроєктовано структуру платформи, створені діаграми взаємодії, розроблені модулі безпечного зберігання файлів, обміну ключами, виявлення аномалій, розпізнавання тексту на зображеннях та його перетворення документи, реалізовано архітектуру інтерфейсу користувача та логіку управління доступом. Було проведено випробування в різних умовах використання, що засвідчило високу ефективність і функціональність платформи.

Робота оформлена з урахуванням рекомендацій [27].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sudarshan, Deeksha, et al. Resource Centric Analysis of RSA and ECC Algorithms on FPGA. ITM Web of Conferences. Vol. 56. EDP Sciences, 2023.

2. Національний координаційний центр кібербезпеки. Хакери масово атакують розробників та постачальників рішень АСУТП, щоб дістатися до критично важливих українських підприємств. [Електронний ресурс].

3. Amazon AWS: Data Protection in AWS. [Електронний ресурс].

4. Google Cloud Platform: Encryption in Transit. [Електронний ресурс].

5. IBM Quantum Experience: Documentation Overview. [Електронний ресурс].

6. Войтко В.В. Розробка вебплатформи з постквантовим шифруванням та системою виявлення аномалій у поведінці / В.В. Войтко, В.Д. Шиндирук // Матеріали Всеукраїнської науково-практичної інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2025) » [Електронний ресурс] – https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/view/24662

7. Шиндирук В. РОЗРОБКА ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ ГЕНЕРАЦІЇ БІОНІЧНИХ 3D-МОДЕЛЕЙ ПРОТЕЗІВ НА ОСНОВІ МАШИННОГО НАВЧАННЯ / Матеріали Міжнародної науково-практичної інтернет-конференції «Молодь в науці: дослідження, проблеми,перспективи (МН-2024)» : збірник доповідей. [Електронний ресурс]. – Вінниця: ВНТУ, 2024. – (PDF, 2859 с.) – С. 165-169.

8. What is a UML diagram? [Електронний ресурс] – Режим доступу до ресурсу: https://miro.com/diagramming/what-is-a-uml-diagram.

9. Bernstein, D. J., & Lange, T. (2017). Post-quantum cryptography. Nature, 549(7671), 188-194. https://doi.org/10.1038/nature23461

10. Chen, L., et al. Report on Post-Quantum Cryptography. NISTIR 8105, 2016.

11. Stallings, W. Cryptography and Network Security: Principles and Practice. 8th ed., Pearson, 2020.

12. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. Proc. 28th

13. JSON Web Token [Електронний ресурс] – Режим доступу до ресурсу: https://jwt.io/

14. Time-based One-Time Password [Електронний ресурс] – Режим доступу до ресурсу: https://totp.app/

15. Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://nodejs.org/en

16. Next.js [Електронний ресурс] – Режим доступу до ресурсу: https://nextjs.org/

17. TypeScript [Електронний ресурс] – Режим доступу до ресурсу: https://www.typescriptlang.org/

18. PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org/

19. Tesseract [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/tesseract-ocr/tesseract

20. Docker [Електронний ресурс] – Режим доступу до ресурсу: https://www.docker.com/

21. ESLint [Електронний ресурс] – Режим доступу до ресурсу: https://eslint.org/

22. Eclipse [Електронний ресурс] – Режим доступу до ресурсу: https://www.eclipse.org/

23. Visual Studio Code - Code Editing. Redefined Guide [Електронний ресурс] – Режим доступу до ресурсу: https://code.visualstudio.com/

24. Intelij IDEA [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/idea/#

25. What is Visual Studio? [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/uk-ua/visualstudio/get-started/visual-studio-ide

26. Рівні тестування [Електронний ресурс] – Режим доступу до ресурсу: https://qalearning.com.ua/theory/lectures/material/testing-levels/

27. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів студентів спеціальності 121 "Інженерія програмного забезпечення" / Уклад. О. Н. Романюк, Г. О. Черноволик – Вінниця: ВНТУ, 2022. – 52с.

ДОДАТКИ

84

Додаток А. Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О. Н.
«_21_» __березня__ 2025 р.

Технічне завдання
на бакалаврську кваліфікаційну роботу
«Розробка методу і програмних засобів вебплатформи з постквантовим шифруванням файлів та стійким сховищем до атак»
за спеціальністю 121 – Інженерія програмного забезпечення

Керівник бакалаврської кваліфікаційної роботи:
к.т.н., доц. Войтко В. В.
«_21_» __березня__ 2025 р.

Виконала:
студентка гр. 1ПІ-21б Шиндирук В. Д.
«_21_» __березня__ 2025 р.

Вінниця – 2025 року

1.

Найменування та галузь застосування

Бакалаврська кваліфікаційна робота: «Розробка методу і програмних засобів вебплатформи з постквантовим шифруванням файлів та стійким сховищем до атак».

Галузь застосування – малий, середній та великий бізнес, військові структури, медичні установи.

2. Підстава для розробки

Підставою для виконання бакалаврської квалілфікаційної роботи (БКР) є індивідуальне завдання на БКР та наказ №97 від 21 березня 2025 року ректора по ВНТУ про закріплення тем БКР.

3. Мета та призначення розробки

Метою дослідження є підвищення рівня захищеності критично важливих структур від ураження квантовими комп'ютерами шляхом розробки модулів системи, що підвищить захист передачі даних за рахунок змагальної квантової нейронної мережі.

Призначення роботи – розробка програмного продукту для захисту конфіденційних даних установ, бізнесів, структур.

4. Вихідні дані для проведення НКР

1. Bernstein, D. J., & Lange, T. (2017). Post-quantum cryptography. Nature, 549(7671), 188-194. https://doi.org/10.1038/nature23461
2. Chen, L., et al. Report on Post-Quantum Cryptography. NISTIR 8105, 2016.
3. Stallings, W. Cryptography and Network Security: Principles and Practice. 8th ed., Pearson, 2020.
4. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. Proc. 28th

5. Технічні вимоги

Комп’ютер з 64-розрядним (x64) процесором та тактовою частотою 2 ГГц. Об’єм оперативної пам’яті 8 ГБ, розмір жорсткого диску 256 ГБ. Операційна система Windows 10. З клієнтського боку: ПК з будь-якою операційною системою та будь-який браузер.

6. Конструктивні вимоги

Вебзастосунок має відповідати ергономічним та технічним вимогам. Текстова та графічна документація повинна відповідати стандартам України.

7. Перелік технічної документації, що пред’являється по закінченню робіт:

- пояснювальна записка до бакалаврської ккваліфікаційної роботи;
- технічне завдання;
- лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз стану розвитку програм з квантово захищеним сховищем	25.03.25 – 01.04.25
2	Розробка методів, моделі та алгоритмів роботи програми	02.04.25 – 14.04.25
3	Розробка програмного забезпечення	16.04.25 – 03.05.25
4	Тестування програми	06.05.25 – 17.05.25
5	Оформлення матеріалів до захисту БКР	23.05.25 – 30.05.25

10. Порядок контролю та прийняття

Всі етапи бакалаврської кваліфікаційної роботи контролюються науковим керівником згідно плану по виконанню роботи. Прийняття бакалаврської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком захисту.

88

Додаток Б. Протокол перевірки бакалаврської кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Розробка методу і програмних засобів вебплатформи з постквантовим шифруванням файлів та стійким сховищем до атак

Тип роботи: БКР

(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра програмного забезпечення, ФІТКІ , група1ПІ-21б

(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 3,2 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

_____ _____
(прізвище, ініціали, посада) (підпис)

_____ _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Черноволик Г. О.

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Войтко В. В., к.т.н., доцент кафедри ПЗ
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Шиндирук В. Д.
(підпис) (прізвище, ініціали)

Додаток В. Лістинг програми


```
    "node_modules/eslint-config-next": {
      "version": "15.2.3",
      "resolved": "https://registry.npmjs.org/eslint-config-next/-/eslint-config-next-15.2.3.tgz",
      "integrity": "sha512-VDQwbajhNMFmrhLWVyUXCqsGPN+zz5G8Ys/QwFubfsxTIrkqdx3N3x3QPW+pERz8bzGPP0IgEm8cNbZcd8PFRQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "@next/eslint-plugin-next": "15.2.3",
        "@rushstack/eslint-patch": "^1.10.3",
        "@typescript-eslint/eslint-plugin": "^5.4.2 || ^6.0.0 || ^7.0.0 || ^8.0.0",
        "@typescript-eslint/parser": "^5.4.2 || ^6.0.0 || ^7.0.0 || ^8.0.0",
        "eslint-import-resolver-node": "^0.3.6",
        "eslint-import-resolver-typescript": "^3.5.2",
        "eslint-plugin-import": "^2.31.0",
        "eslint-plugin-jsx-a11y": "^6.10.0",
        "eslint-plugin-react": "^7.37.0",
        "eslint-plugin-react-hooks": "^5.0.0"
      },
      "peerDependencies": {
        "eslint": "^7.23.0 || ^8.0.0 || ^9.0.0",
        "typescript": ">=3.3.1"
      },
      "peerDependenciesMeta": {
        "typescript": {
          "optional": true
        }
      }
    },
```

```
    "node_modules/eslint-import-resolver-node": {
      "version": "0.3.9",
      "resolved": "https://registry.npmjs.org/eslint-import-resolver-node/-/eslint-import-resolver-node-0.3.9.tgz",
      "integrity": "sha512-WFj2isz22JahUv+B788TlO3N6zL3nNJGU8CcZbPZvVEkBPaJdCV4vy5wyghty5ROFbCRnm132v8BScu5/1BQ8g==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "debug": "^3.2.7",
        "is-core-module": "^2.13.0",
        "resolve": "^1.22.4"
      }
    },
    "node_modules/eslint-import-resolver-node/node_modules/debug": {
      "version": "3.2.7",
      "resolved": "https://registry.npmjs.org/debug/-/debug-3.2.7.tgz",
      "integrity": "sha512-CFjzYYAi4ThfiQvizrFQevTTXHtnCqWfe7x1AhgEscTz6ZbLbfoLRLPugTQyBth6f8ZERVUSyWHFD/7Wu4t1XQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "ms": "^2.1.1"
      }
    },
    "node_modules/eslint-import-resoler-typescript": {
      "version": "3.10.0",
      "resolved": "https://registry.npmjs.org/eslint-import-resolver-typescript/-/eslint-import-resolver-typescript-3.10.0.tgz",
```

"integrity": "sha512-aV3/dVsT0/H9BtpNwbaqvl+0xGMRGzncLyhm793NFGvbwGGvzyAykqWZ8oZlZuGwuHkwJjhWJkG1cM3ynvd2pQ==",
"dev": true,
"license": "ISC",
"dependencies": {
"@nolyfill/is-core-module": "1.0.39",
"debug": "^4.4.0",
"get-tsconfig": "^4.10.0",
"is-bun-module": "^2.0.0",
"stable-hash": "^0.0.5",
"tinyglobby": "^0.2.12",
"unrs-resolver": "^1.3.2"
},
"engines": {
"node": "^14.18.0 || >=16.0.0"
},
"funding": {
"url": "https://opencollective.com/eslint-import-resolver-typescript"
},
"peerDependencies": {
"eslint": "*",
"eslint-plugin-import": "*",
"eslint-plugin-import-x": "*"
},
"peerDependenciesMeta": {
"eslint-plugin-import": {
"optional": true
},
"eslint-plugin-import-x": {
"optional": true
}

```
    }
  },
  "node_modules/eslint-module-utils": {
    "version": "2.12.0",
    "resolved": "https://registry.npmjs.org/eslint-module-utils/-/eslint-module-utils-2.12.0.tgz",
    "integrity": "sha512-wALZ0HFoytlyh/1+4wuZ9FJCD/leWHQzzrxJ8+rebyReSLk7LApMyd3WJaLVoN+D5+WIdJyDK1c6JnE65V4Zyg==",
    "dev": true,
    "license": "MIT",
    "dependencies": {
      "debug": "^3.2.7"
    },
    "engines": {
      "node": ">=4"
    },
    "peerDependenciesMeta": {
      "eslint": {
        "optional": true
      }
    }
  },
  "node_modules/eslint-module-utils/node_modules/debug": {
    "version": "3.2.7",
    "resolved": "https://registry.npmjs.org/debug/-/debug-3.2.7.tgz",
    "integrity": "sha512-CFjzYYAi4ThfiQvizrFQevTTXHtnCqWfe7x1AhgEscTz6ZbLbfoLRLPugTQyBth6f8ZERVUSyWHFD/7Wu4t1XQ==",
    "dev": true,
    "license": "MIT",
    "dependencies": {
      "ms": "^2.1.1"
```

```
    }
   },
   "node_modules/eslint-plugin-import": {
    "version": "2.31.0",
    "resolved": "https://registry.npmjs.org/eslint-plugin-import/-/eslint-plugin-import-2.31.0.tgz",
    "integrity": "sha512-ixmkI62Rbc2/w8Vfxyh1jQRTdRTF52VxwRVHl/ykPAmqG+Nb7/kNn+byLP0LxPgI7zWA16Jt82SybJInmMia3A==",
    "dev": true,
    "license": "MIT",
    "dependencies": {
     "@rtsao/scc": "^1.1.0",
     "array-includes": "^3.1.8",
     "array.prototype.findlastindex": "^1.2.5",
     "array.prototype.flat": "^1.3.2",
     "array.prototype.flatmap": "^1.3.2",
     "debug": "^3.2.7",
     "doctrine": "^2.1.0",
     "eslint-import-resolver-node": "^0.3.9",
     "eslint-module-utils": "^2.12.0",
     "hasown": "^2.0.2",
     "is-core-module": "^2.15.1",
     "is-glob": "^4.0.3",
     "minimatch": "^3.1.2",
     "object.fromentries": "^2.0.8",
     "object.groupby": "^1.0.3",
     "object.values": "^1.2.0",
     "semver": "^6.3.1",
     "string.prototype.trimend": "^1.0.8",
     "tsconfig-paths": "^3.15.0"
    },
```

```
      "engines": {
        "node": ">=4"
      },
      "peerDependencies": {
        "eslint": "^2 || ^3 || ^4 || ^5 || ^6 || ^7.2.0 || ^8 || ^9"
      }
    },
    "node_modules/eslint-plugin-import/node_modules/debug": {
      "version": "3.2.7",
      "resolved": "https://registry.npmjs.org/debug/-/debug-3.2.7.tgz",
      "integrity": "sha512-
CFjzYYAi4ThfiQvizrFQevTTXHtnCqWfe7x1AhgEscTz6ZbLbfoLRLPugTQyBth6f8ZERVUSyWHF
D/7Wu4t1XQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "ms": "^2.1.1"
      }
    },
    "node_modules/eslint-plugin-import/node_modules/semver": {
      "version": "6.3.1",
      "resolved": "https://registry.npmjs.org/semver/-/semver-6.3.1.tgz",
      "integrity": "sha512-
BR7VvDCVHO+q2xBEWskxS6DJE1qRnb7DxzUrogb71CWoSficBxYsiAGd+Kl0mmq/MprG9yArR
kyrQxTO6XjMzA==",
      "dev": true,
      "license": "ISC",
      "bin": {
        "semver": "bin/semver.js"
      }
    },
    "node_modules/eslint-plugin-jsx-a11y": {
```

```
      "version": "6.10.2",
      "resolved": "https://registry.npmjs.org/eslint-plugin-jsx-a11y/-/eslint-plugin-jsx-a11y-6.10.2.tgz",
      "integrity": "sha512-scB3nz4WmG75pV8+3eRUQOHZlNSUhFNq37xnpgRkCCELU3XMvXAxLk1eqWWyE22Ki4Q01Fnsw9BA3cJHDPgn2Q==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "aria-query": "^5.3.2",
        "array-includes": "^3.1.8",
        "array.prototype.flatmap": "^1.3.2",
        "ast-types-flow": "^0.0.8",
        "axe-core": "^4.10.0",
        "axobject-query": "^4.1.0",
        "damerau-levenshtein": "^1.0.8",
        "emoji-regex": "^9.2.2",
        "hasown": "^2.0.2",
        "jsx-ast-utils": "^3.3.5",
        "language-tags": "^1.0.9",
        "minimatch": "^3.1.2",
        "object.fromentries": "^2.0.8",
        "safe-regex-test": "^1.0.3",
        "string.prototype.includes": "^2.0.1"
      },
      "engines": {
        "node": ">=4.0"
      },
      "peerDependencies": {
        "eslint": "^3 || ^4 || ^5 || ^6 || ^7 || ^8 || ^9"
      }
    },
```

```
"node_modules/eslint-plugin-react": {
 "version": "7.37.5",
 "resolved": "https://registry.npmjs.org/eslint-plugin-react/-/eslint-plugin-react-7.37.5.tgz",
 "integrity": "sha512-Qteup0SqU15kdocexFNAJMvCJEfa2xUKNV4CC1xsVMrIIqEy3SQ/rqyxCWNzfrd3/ldy6HMlD2e0JDVpDg2qIA==",
 "dev": true,
 "license": "MIT",
 "dependencies": {
  "array-includes": "^3.1.8",
  "array.prototype.findlast": "^1.2.5",
  "array.prototype.flatmap": "^1.3.3",
  "array.prototype.tosorted": "^1.1.4",
  "doctrine": "^2.1.0",
  "es-iterator-helpers": "^1.2.1",
  "estraverse": "^5.3.0",
  "hasown": "^2.0.2",
  "jsx-ast-utils": "^2.4.1 || ^3.0.0",
  "minimatch": "^3.1.2",
  "object.entries": "^1.1.9",
  "object.fromentries": "^2.0.8",
  "object.values": "^1.2.1",
  "prop-types": "^15.8.1",
  "resolve": "^2.0.0-next.5",
  "semver": "^6.3.1",
  "string.prototype.matchall": "^4.0.12",
  "string.prototype.repeat": "^1.0.0"
 },
 "engines": {
  "node": ">=4"
 },
 "peerDependencies": {
```

```
      "eslint": "^3 || ^4 || ^5 || ^6 || ^7 || ^8 || ^9.7"
     }
    },
    "node_modules/eslint-plugin-react-hooks": {
     "version": "5.2.0",
     "resolved":  "https://registry.npmjs.org/eslint-plugin-react-hooks/-/eslint-plugin-react-hooks-5.2.0.tgz",
     "integrity": "sha512-+f15FfK64YQwZdJNELETdn5ibXEUQmW1DZL6KXhNnc2heoy/sg9VJJeT7n8TlMWouzWqSWavFkIhHyIbIAEapg==",
     "dev": true,
     "license": "MIT",
     "engines": {
      "node": ">=10"
     },
     "peerDependencies": {
      "eslint": "^3.0.0 || ^4.0.0 || ^5.0.0 || ^6.0.0 || ^7.0.0 || ^8.0.0-0 || ^9.0.0"
     }
    },
    "node_modules/eslint-plugin-react/node_modules/resolve": {
     "version": "2.0.0-next.5",
     "resolved": "https://registry.npmjs.org/resolve/-/resolve-2.0.0-next.5.tgz",
     "integrity": "sha512-U7WjGVG9sH8tvjW5SmGbQuui75FiyjAX72HX15DwBBwF9dNiQZRQAg9nnPhYy+TUnE0+VcrttuvNI8oSxZcocA==",
     "dev": true,
     "license": "MIT",
     "dependencies": {
      "is-core-module": "^2.13.0",
      "path-parse": "^1.0.7",
      "supports-preserve-symlinks-flag": "^1.0.0"
     },
```

```
      "bin": {
        "resolve": "bin/resolve"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/eslint-plugin-react/node_modules/semver": {
      "version": "6.3.1",
      "resolved": "https://registry.npmjs.org/semver/-/semver-6.3.1.tgz",
      "integrity": "sha512-
BR7VvDCVHO+q2xBEWskxS6DJE1qRnb7DxzUrogb71CWoSficBxYsiAGd+Kl0mmq/MprG9yArR
kyrQxTO6XjMzA==",
      "dev": true,
      "license": "ISC",
      "bin": {
        "semver": "bin/semver.js"
      }
    },
    "node_modules/eslint-scope": {
      "version": "8.3.0",
      "resolved": "https://registry.npmjs.org/eslint-scope/-/eslint-scope-8.3.0.tgz",
      "integrity": "sha512-
pUNxi75F8MJ/GdeKtVLSbYg4ZI34J6C0C7sbL4YOp2exGwen7ZsuBqKzUhXd0qMQ362yET3z+uP
wKeg/0C2XCQ==",
      "dev": true,
      "license": "BSD-2-Clause",
      "dependencies": {
        "esrecurse": "^4.3.0",
        "estraverse": "^5.2.0"
      },
      "engines": {
```

```
      "node": "^18.18.0 || ^20.9.0 || >=21.1.0"
     },
     "funding": {
      "url": "https://opencollective.com/eslint"
     }
    },
    "node_modules/eslint-visitor-keys": {
     "version": "4.2.0",
     "resolved": "https://registry.npmjs.org/eslint-visitor-keys/-/eslint-visitor-keys-4.2.0.tgz",
     "integrity": "sha512-
UyLnSehNt62FFhSwjZlHmeokpRK59rcz29j+F1/aDgbkbRTk7wIc9XzdoasMUbRNKDM0qQt/+BJ4B
rpFeABemw==",
     "dev": true,
     "license": "Apache-2.0",
     "engines": {
      "node": "^18.18.0 || ^20.9.0 || >=21.1.0"
     },
     "funding": {
      "url": "https://opencollective.com/eslint"
     }
    },
    "node_modules/espree": {
     "version": "10.3.0",
     "resolved": "https://registry.npmjs.org/espree/-/espree-10.3.0.tgz",
     "integrity": "sha512-
0QYC8b24HWY8zjRnDTL6RiHfDbAWn63qb4LMj1Z4b076A4une81+z03Kg7l7mn/48PUTqoLptSX
ez8oknU8Clg==",
     "dev": true,
     "license": "BSD-2-Clause",
     "dependencies": {
      "acorn": "^8.14.0",
      "acorn-jsx": "^5.3.2",
```

```
      "eslint-visitor-keys": "^4.2.0"
     },
     "engines": {
      "node": "^18.18.0 || ^20.9.0 || >=21.1.0"
     },
     "funding": {
      "url": "https://opencollective.com/eslint"
     }
    },
    "node_modules/esquery": {
     "version": "1.6.0",
     "resolved": "https://registry.npmjs.org/esquery/-/esquery-1.6.0.tgz",
     "integrity": "sha512-ca9pw9fomFcKPvFLXhBKUK90ZvGibiGOvRJNbjljY7s7uq/5YO4BOzcYtJqExdx99rF6aAcnRxHmcUHcz6sQsg==",
     "dev": true,
     "license": "BSD-3-Clause",
     "dependencies": {
      "estraverse": "^5.1.0"
     },
     "engines": {
      "node": ">=0.10"
     }
    },
    "node_modules/esrecurse": {
     "version": "4.3.0",
     "resolved": "https://registry.npmjs.org/esrecurse/-/esrecurse-4.3.0.tgz",
     "integrity": "sha512-KmfKL3b6G+RXvP8N1vr3Tq1kL/oCFgn2NYXEtqP8/L3pKapUA4G8cFVaoF3SU323CD4XypR/ffioHmkti6/Tag==",
     "dev": true,
     "license": "BSD-2-Clause",
```

```
      "dependencies": {
        "estraverse": "^5.2.0"
      },
      "engines": {
        "node": ">=4.0"
      }
    },
    "node_modules/estraverse": {
      "version": "5.3.0",
      "resolved": "https://registry.npmjs.org/estraverse/-/estraverse-5.3.0.tgz",
      "integrity": "sha512-
MMdARuVEQziNTeJD8DgMqmhwR11BRQ/cBP+pLtYdSTnf3MIO8fFeiINEbX36ZdNlfU/7A9f3gU
w49B3oQsvwBA==",
      "dev": true,
      "license": "BSD-2-Clause",
      "engines": {
        "node": ">=4.0"
      }
    },
    "node_modules/esutils": {
      "version": "2.0.3",
      "resolved": "https://registry.npmjs.org/esutils/-/esutils-2.0.3.tgz",
      "integrity": "sha512-
kVscqXk4OCp68SZ0dkgEKVi6/8ij300KBWTJq32P/dYeWTSwK41WyTxalN1eRmA5Z9UU/LX9D7
FWSmV9SAYx6g==",
      "dev": true,
      "license": "BSD-2-Clause",
      "engines": {
        "node": ">=0.10.0"
      }
    },
    "node_modules/fast-deep-equal": {
```

```
      "version": "3.1.3",
      "resolved": "https://registry.npmjs.org/fast-deep-equal/-/fast-deep-equal-3.1.3.tgz",
      "integrity": "sha512-
f3qQ9oQy9j2AhBe/H9VC91wLmKBCCU/gDOnKNAYG5hswO7BLKj09Hc5HYNz9cGI++xlpDCIg
DaitVs03ATR84Q==",
      "dev": true,
      "license": "MIT"
    },
    "node_modules/fast-glob": {
      "version": "3.3.1",
      "resolved": "https://registry.npmjs.org/fast-glob/-/fast-glob-3.3.1.tgz",
      "integrity": "sha512-
kNFPyjhh5cKjrUltxs+wFx+ZkbRaxxmZ+X0ZU31SOsxCEtP9VPgtq2teZw1DebupL5GmDaNQ6yK
MMVcM41iqDg==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "@nodelib/fs.stat": "^2.0.2",
        "@nodelib/fs.walk": "^1.2.3",
        "glob-parent": "^5.1.2",
        "merge2": "^1.3.0",
        "micromatch": "^4.0.4"
      },
      "engines": {
        "node": ">=8.6.0"
      }
    },
    "node_modules/fast-glob/node_modules/glob-parent": {
      "version": "5.1.2",
      "resolved": "https://registry.npmjs.org/glob-parent/-/glob-parent-5.1.2.tgz",
```

```
      "integrity": "sha512-AOIgSQCepiJYwP3ARnGx+5VnTu2HBYdzbGP45eLw1vr3zB3vZLeyed1sC9hnbcOc9/SrMyM5RPQrkGz4aS9Zow==",
      "dev": true,
      "license": "ISC",
      "dependencies": {
       "is-glob": "^4.0.1"
      },
      "engines": {
       "node": ">= 6"
      }
     },
     "node_modules/fast-json-stable-stringify": {
      "version": "2.1.0",
      "resolved": "https://registry.npmjs.org/fast-json-stable-stringify/-/fast-json-stable-stringify-2.1.0.tgz",
      "integrity": "sha512-lhd/wF+Lk98HZoTCtlVraHtfh5XYijIjalXck7saUtuanSDyLMxnHhSXEDJqHxD7msR8D0uCmqlkwjCV8xvwHw==",
      "dev": true,
      "license": "MIT"
     },
     "node_modules/fast-levenshtein": {
      "version": "2.0.6",
      "resolved": "https://registry.npmjs.org/fast-levenshtein/-/fast-levenshtein-2.0.6.tgz",
      "integrity": "sha512-DCXu6Ifhqcks7TZKY3Hxp3y6qphY5SJZmrWMDrKcERSOXWQdMhU9Ig/PYrzyw/ul9jOIyh0N4M0tbC5hodg8dw==",
      "dev": true,
      "license": "MIT"
     },
     "node_modules/fastq": {
```

```
      "version": "1.19.1",
      "resolved": "https://registry.npmjs.org/fastq/-/fastq-1.19.1.tgz",
      "integrity": "sha512-GwLTyxkCXjXbxqIhTsMI2Nui8huMPtnxg7krajPJAjnEG/iiOS7i+zCtWGZR9G0NBKbXKh6X9m9UIsYX/N6vvQ==",
      "dev": true,
      "license": "ISC",
      "dependencies": {
        "reusify": "^1.0.4"
      }
    },
    "node_modules/file-entry-cache": {
      "version": "8.0.0",
      "resolved": "https://registry.npmjs.org/file-entry-cache/-/file-entry-cache-8.0.0.tgz",
      "integrity": "sha512-XXTUwCvisa5oacNGRP9SfNtYBNAMi+RPwBFmblZEF7N7swHYQS6/Zfk7SRwx4D5j3CH211YNRco1DEMNVfZCnQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "flat-cache": "^4.0.0"
      },
      "engines": {
        "node": ">=16.0.0"
      }
    },
    "node_modules/fill-range": {
      "version": "7.1.1",
      "resolved": "https://registry.npmjs.org/fill-range/-/fill-range-7.1.1.tgz",
      "integrity": "sha512-YsGpe3WHLK8ZYi4tWDg2Jy3ebRz2rXowDxnld4bkQB00cc/1Zw9AWnC0i9ztDJitivtQvaI9KaLyKrc+hBW0yg==",
```

```
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "to-regex-range": "^5.0.1"
      },
      "engines": {
        "node": ">=8"
      }
    },
    "node_modules/find-root": {
      "version": "1.1.0",
      "resolved": "https://registry.npmjs.org/find-root/-/find-root-1.1.0.tgz",
      "integrity": "sha512-NKfW6bec6GfKc0SGx1e07QZY9PE99u0Bft/0rzSD5k3sO/vwkVUpDUKVm5Gpp5Ue3YfShPFTX2070tDs5kB9Ng==",
      "license": "MIT"
    },
    "node_modules/find-up": {
      "version": "5.0.0",
      "resolved": "https://registry.npmjs.org/find-up/-/find-up-5.0.0.tgz",
      "integrity": "sha512-78/PXT1wlLLDgTzDs7sjq9hzz0vXD+zn+7wypEe4fXQxCmdmqfGsEPQxmiCSQI3ajFV91bVSsvNtrJRiW6nGng==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "locate-path": "^6.0.0",
        "path-exists": "^4.0.0"
      },
      "engines": {
        "node": ">=10"
      },
```

```
    "funding": {
      "url": "https://github.com/sponsors/sindresorhus"
    }
  },
  "node_modules/flat-cache": {
    "version": "4.0.1",
    "resolved": "https://registry.npmjs.org/flat-cache/-/flat-cache-4.0.1.tgz",
    "integrity": "sha512-f7ccFPK3SXFHpx15UIGyRJ/FJQctuKZ0zVuN3frBo4HnK3cay9VEW0R6yPYFHC0AgqhukPzKjq22t5DmAyqGyw==",
    "dev": true,
    "license": "MIT",
    "dependencies": {
      "flatted": "^3.2.9",
      "keyv": "^4.5.4"
    },
    "engines": {
      "node": ">=16"
    }
  },
  "node_modules/flatted": {
    "version": "3.3.3",
    "resolved": "https://registry.npmjs.org/flatted/-/flatted-3.3.3.tgz",
    "integrity": "sha512-GX+ysw4PBCz0PzosHDepZGANEuFCMLrnRTiEy9McGjmkCQYwRq4A/X786G/fjM/+OjsWSU1ZrY5qyARZmO/uwg==",
    "dev": true,
    "license": "ISC"
  },
  "node_modules/follow-redirects": {
    "version": "1.15.9",
    "resolved": "https://registry.npmjs.org/follow-redirects/-/follow-redirects-1.15.9.tgz",
```

"integrity": "sha512-gew4GsXizNgdoRyqmyfMHyAmXsZDk6mHkSxZFCzW9gwlbtOW44CDtYavM+y+72qD/Vq2l550kMF52DT8fOLJqQ==",
"funding": [
{
"type": "individual",
"url": "https://github.com/sponsors/RubenVerborgh"
}
],
"license": "MIT",
"engines": {
"node": ">=4.0"
},
"peerDependenciesMeta": {
"debug": {
"optional": true
}
}
},
"node_modules/for-each": {
"version": "0.3.5",
"resolved": "https://registry.npmjs.org/for-each/-/for-each-0.3.5.tgz",
"integrity": "sha512-dKx12eRCVIzqCxFGplyFKJMPvLEWgmNtUrpTiJIR5u97zEhRG8ySrtboPHZXx7daLxQVrl643cTzbab2tkQjxg==",
"dev": true,
"license": "MIT",
"dependencies": {
"is-callable": "^1.2.7"
},
"engines": {
"node": ">= 0.4"

```
    },
    "funding": {
      "url": "https://github.com/sponsors/ljharb"
    }
  },
  "node_modules/form-data": {
    "version": "4.0.2",
    "resolved": "https://registry.npmjs.org/form-data/-/form-data-4.0.2.tgz",
    "integrity": "sha512-
hGfm/slu0ZabnNt4oaRZ6uREyfCj6P4fT/n6A1rGV+Z0VdGXjfOhVUpkn6qVQONHGIFwmveGXyD
s75+nr6FM8w==",
    "license": "MIT",
    "dependencies": {
      "asynckit": "^0.4.0",
      "combined-stream": "^1.0.8",
      "es-set-tostringtag": "^2.1.0",
      "mime-types": "^2.1.12"
    },
    "engines": {
      "node": ">= 6"
    }
  },
  "node_modules/function-bind": {
    "version": "1.1.2",
    "resolved": "https://registry.npmjs.org/function-bind/-/function-bind-1.1.2.tgz",
    "integrity": "sha512-
7XHNxH7qX9xG5mIwxkhumTox/MIRNcOgDrxWsMt2pAr23WHp6MrRlN7FBSFpCpr+oVO0F744i
UgR82nJMfG2SA==",
    "license": "MIT",
    "funding": {
      "url": "https://github.com/sponsors/ljharb"
    }
```

```
    },
    "node_modules/function.prototype.name": {
      "version": "1.1.8",
      "resolved": "https://registry.npmjs.org/function.prototype.name/-/function.prototype.name-1.1.8.tgz",
      "integrity": "sha512-e5iwyodOHhbMr/yNrc7fDYG4qlbIvI5gajyzPnb5TCwyhjApznQh1BMFou9b30SevY43gCJKXycoCBjMbsuW0Q==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "call-bind": "^1.0.8",
        "call-bound": "^1.0.3",
        "define-properties": "^1.2.1",
        "functions-have-names": "^1.2.3",
        "hasown": "^2.0.2",
        "is-callable": "^1.2.7"
      },
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/functions-have-names": {
      "version": "1.2.3",
      "resolved": "https://registry.npmjs.org/functions-have-names/-/functions-have-names-1.2.3.tgz",
      "integrity": "sha512-xckBUXyTIqT97tq2x2AMb+g163b5JFysYk0x4qxNFwbfQkmNZoiRHb6sPzI9/QV33WeuvVYBUIiD4NzNIyqaRQ==",
```

```
      "dev": true,
      "license": "MIT",
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/get-intrinsic": {
      "version": "1.3.0",
      "resolved": "https://registry.npmjs.org/get-intrinsic/-/get-intrinsic-1.3.0.tgz",
      "integrity": "sha512-
9fSjSaos/fRIVIp+xSJlE6lfwhES7LNtKaCBIamHsjr2na1BiABJPo0mOjjz8GJDURarmCPGqaiVg5mfj
b98CQ==",
      "license": "MIT",
      "dependencies": {
        "call-bind-apply-helpers": "^1.0.2",
        "es-define-property": "^1.0.1",
        "es-errors": "^1.3.0",
        "es-object-atoms": "^1.1.1",
        "function-bind": "^1.1.2",
        "get-proto": "^1.0.1",
        "gopd": "^1.2.0",
        "has-symbols": "^1.1.0",
        "hasown": "^2.0.2",
        "math-intrinsics": "^1.1.0"
      },
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
```

```
        "node_modules/get-proto": {
          "version": "1.0.1",
          "resolved": "https://registry.npmjs.org/get-proto/-/get-proto-1.0.1.tgz",
          "integrity": "sha512-
sTSfBjoXBp89JvIKIefqw7U2CCebsc74kiY6awiGogKtoSGbgjYE/G/+l9sF3MWFPNc9IcoOC4ODfK
HfxFmp0g==",
          "license": "MIT",
          "dependencies": {
            "dunder-proto": "^1.0.1",
            "es-object-atoms": "^1.0.0"
          },
          "engines": {
            "node": ">= 0.4"
          }
        },
        "node_modules/get-symbol-description": {
          "version": "1.1.0",
          "resolved": "https://registry.npmjs.org/get-symbol-description/-/get-symbol-description-
1.1.0.tgz",
          "integrity": "sha512-
w9UMqWwJxHNOvoNzSJ2oPF5wvYcvP7jUvYzhp67yEhTi17ZDBBC1z9pTdGuzjD+EFIqLSYRwe
ZjqfiPzQ06Ebg==",
          "dev": true,
          "license": "MIT",
          "dependencies": {
            "call-bound": "^1.0.3",
            "es-errors": "^1.3.0",
            "get-intrinsic": "^1.2.6"
          },
          "engines": {
            "node": ">= 0.4"
          },
```

```
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/get-tsconfig": {
      "version": "4.10.0",
      "resolved": "https://registry.npmjs.org/get-tsconfig/-/get-tsconfig-4.10.0.tgz",
      "integrity": "sha512-kGzZ3LWWQcGIAmg6iWvXn0ei6WDtV26wzHRMwDSzmAbcXrTEXxHy6IehI6/4eT6VRKyMP1eF1VqwrVUmE/LR7A==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "resolve-pkg-maps": "^1.0.0"
      },
      "funding": {
        "url": "https://github.com/privatenumber/get-tsconfig?sponsor=1"
      }
    },
    "node_modules/glob-parent": {
      "version": "6.0.2",
      "resolved": "https://registry.npmjs.org/glob-parent/-/glob-parent-6.0.2.tgz",
      "integrity": "sha512-XxwI8EOhVQgWp6iDL+3b0r86f4d6AX6zSU55HfB4ydCEuXLXc5FcYeOu+nnGftS4TEju/11rt4KJPTMgbfmv4A==",
      "dev": true,
      "license": "ISC",
      "dependencies": {
        "is-glob": "^4.0.3"
      },
      "engines": {
        "node": ">=10.13.0"
```

```
    }
  },
  "node_modules/globals": {
    "version": "11.12.0",
    "resolved": "https://registry.npmjs.org/globals/-/globals-11.12.0.tgz",
    "integrity": "sha512-
WOBp/EEGUiIsJSp7wcv/y6MO+lV9UoncWqxuFfm8eBwzWNgyfBd6Gz+IeKQ9jCmyhoH99g15M3
T+QaVHFjizVA==",
    "license": "MIT",
    "engines": {
      "node": ">=4"
    }
  },
  "node_modules/globalthis": {
    "version": "1.0.4",
    "resolved": "https://registry.npmjs.org/globalthis/-/globalthis-1.0.4.tgz",
    "integrity": "sha512-
DpLKbNU4WylpxJykQujfCcwYWiV/Jhm50Goo0wrVILAv5jOr9d+H+UR3PhSCD2rCCEIg0uc+G+
muBTwD54JhDQ==",
    "dev": true,
    "license": "MIT",
    "dependencies": {
      "define-properties": "^1.2.1",
      "gopd": "^1.0.1"
    },
    "engines": {
      "node": ">= 0.4"
    },
    "funding": {
      "url": "https://github.com/sponsors/ljharb"
    }
  },
```

```
    "node_modules/gopd": {
      "version": "1.2.0",
      "resolved": "https://registry.npmjs.org/gopd/-/gopd-1.2.0.tgz",
      "integrity": "sha512-
ZUKRh6/kUFoAiTAtTYPZJ3hw9wNxx+BIBOijnlG9PnrJsCcSjs1wyyD6vJpaYtgnzDrKYRSqf3OO6
Rfa93xsRg==",
      "license": "MIT",
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/graceful-fs": {
      "version": "4.2.11",
      "resolved": "https://registry.npmjs.org/graceful-fs/-/graceful-fs-4.2.11.tgz",
      "integrity": "sha512-
RbJ5/jmFcNNCcDV5o9eTnBLJ/HszWV0P73bc+Ff4nS/rJj+YaS6IGyiOL0VoBYX+l1Wrl3k63h/KrH
+nhJ0XvQ==",
      "dev": true,
      "license": "ISC"
    },
    "node_modules/graphemer": {
      "version": "1.4.0",
      "resolved": "https://registry.npmjs.org/graphemer/-/graphemer-1.4.0.tgz",
      "integrity": "sha512-
EtKwoO6kxCL9WO5xipiHTZlSzBm7WLT627TqC/uVRd0HKmq8NXyebnNYxDoBi7wt8eTWrUrK
XCOVaFq9x1kgag==",
      "dev": true,
      "license": "MIT"
    },
```

```
    "node_modules/has-bigints": {
      "version": "1.1.0",
      "resolved": "https://registry.npmjs.org/has-bigints/-/has-bigints-1.1.0.tgz",
      "integrity": "sha512-
R3pbpkcIqv2Pm3dUwgjclDRVmWpTJW2DcMzcIhEXEx1oh/CEMObMm3KLmRJOdvhM7o4uQBn
wr8pzRK2sJWIqfg==",
      "dev": true,
      "license": "MIT",
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/has-flag": {
      "version": "4.0.0",
      "resolved": "https://registry.npmjs.org/has-flag/-/has-flag-4.0.0.tgz",
      "integrity": "sha512-
EykJT/Q1KjTWctppgIAgfSO0tKVuZUjhgMr17kqTumMl6Afv3EISleU7qZUzoXDFTAHTDC4NOo
G/ZxU3EvlMPQ==",
      "dev": true,
      "license": "MIT",
      "engines": {
        "node": ">=8"
      }
    },
    "node_modules/has-property-descriptors": {
      "version": "1.0.2",
      "resolved": "https://registry.npmjs.org/has-property-descriptors/-/has-property-descriptors-
1.0.2.tgz",
```

```
      "integrity": "sha512-55JNKuIW+vq4Ke1BjOTjM2YctQIvCT7GFzHwmfZPGo5wnrgkid0YQtnAleFSqumZm4az3n2BS+erby5ipJdgrg==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "es-define-property": "^1.0.0"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/has-proto": {
      "version": "1.2.0",
      "resolved": "https://registry.npmjs.org/has-proto/-/has-proto-1.2.0.tgz",
      "integrity": "sha512-KIL7eQPfHQRC8+XluaIw7BHUwwqL19bQn4hzNgdr+1wXoU0KKj6rufu47lhY7KbJR2C6T6+PfyN0Ea7wkSS+qQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "dunder-proto": "^1.0.0"
      },
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/has-symbols": {
      "version": "1.1.0",
```

```
      "resolved": "https://registry.npmjs.org/has-symbols/-/has-symbols-1.1.0.tgz",
      "integrity": "sha512-1cDNdwJ2Jaohmb3sg4OmKaMBwuC48sYni5HUw2DvsC8LjGTLK9h+eb1X6RyuOHe4hT0ULCW68iomhjUoKUqlPQ==",
      "license": "MIT",
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/has-tostringtag": {
      "version": "1.0.2",
      "resolved": "https://registry.npmjs.org/has-tostringtag/-/has-tostringtag-1.0.2.tgz",
      "integrity": "sha512-NqADB8VjPFLM2V0VvHUewwwsw0ZWBaIdgo+ieHtK3hasLz4qeCRjYcqfB6AQrBggRKppKF8L52/VqdVsO47Dlw==",
      "license": "MIT",
      "dependencies": {
        "has-symbols": "^1.0.3"
      },
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/hasown": {
      "version": "2.0.2",
      "resolved": "https://registry.npmjs.org/hasown/-/hasown-2.0.2.tgz",
```

"integrity": "sha512-0hJU9SCPvmMzIBdZFqNPXWa6dqh7WdH0cII9y+CyS8rG3nL48Bclra9HmKhVVUHyPWNH5Y7xDwAB7bfgSjkUMQ==",
"license": "MIT",
"dependencies": {
"function-bind": "^1.1.2"
},
"engines": {
"node": ">= 0.4"
}
},
"node_modules/hoist-non-react-statics": {
"version": "3.3.2",
"resolved": "https://registry.npmjs.org/hoist-non-react-statics/-/hoist-non-react-statics-3.3.2.tgz",
"integrity": "sha512-/gGivxi8JPKWNm/W0jSmzcMPpfpPLc3dY/6GxhX2hQ9iGj3aDfklV4ET7NjKpSinLpJ5vafa9iiGIEZg10SfBw==",
"license": "BSD-3-Clause",
"dependencies": {
"react-is": "^16.7.0"
}
},
"node_modules/hoist-non-react-statics/node_modules/react-is": {
"version": "16.13.1",
"resolved": "https://registry.npmjs.org/react-is/-/react-is-16.13.1.tgz",
"integrity": "sha512-24e6ynE2H+OKt4kqsOvNd8kBpV65zoxbA4BVsEOB3ARVWQki/DHzaUoC5KuON/BiccDaCCTZBuOcfZs70kR8bQ==",
"license": "MIT"
},
"node_modules/ignore": {

```
      "version": "5.3.2",
      "resolved": "https://registry.npmjs.org/ignore/-/ignore-5.3.2.tgz",
      "integrity": "sha512-hsBTNUqQTDwkWtcdYI2i06Y/nUBEsNEDJKjWdigLvegy8kDuJAS8uRlpkkcQpyEXL0Z/pjDy5HBmMjRCJ2gq+g==",
      "dev": true,
      "license": "MIT",
      "engines": {
        "node": ">= 4"
      }
    },
    "node_modules/immer": {
      "version": "10.1.1",
      "resolved": "https://registry.npmjs.org/immer/-/immer-10.1.1.tgz",
      "integrity": "sha512-s2MPrmjovJcoMaHtx6K11Ra7oD05NT97w1IC5zpMkT6Atjr7H8LjaDd81iIxUYpMKSRRNMJE703M1Fhr/TctHw==",
      "license": "MIT",
      "funding": {
        "type": "opencollective",
        "url": "https://opencollective.com/immer"
      }
    },
    "node_modules/import-fresh": {
      "version": "3.3.1",
      "resolved": "https://registry.npmjs.org/import-fresh/-/import-fresh-3.3.1.tgz",
      "integrity": "sha512-TR3KfrTZTYLPB6jUjfx6MF9WcWrHL9su5TObK4ZkYgBdWKPOFoSoQIdEuTuR82pmtxH2spWG9h6etwfr1pLBqQ==",
      "license": "MIT",
      "dependencies": {
        "parent-module": "^1.0.0",
```

```
      "resolve-from": "^4.0.0"
     },
     "engines": {
      "node": ">=6"
     },
     "funding": {
      "url": "https://github.com/sponsors/sindresorhus"
     }
    },
    "node_modules/imurmurhash": {
     "version": "0.1.4",
     "resolved": "https://registry.npmjs.org/imurmurhash/-/imurmurhash-0.1.4.tgz",
     "integrity": "sha512-
JmXMZ6wuvDmLiHEml9ykzqO6lwFbof0GG4IkcGaENdCRDDmMVnny7s5HsIgHCbaq0w2MyPhD
qkhTUgS2LU2PHA==",
     "dev": true,
     "license": "MIT",
     "engines": {
      "node": ">=0.8.19"
     }
    },
    "node_modules/internal-slot": {
     "version": "1.1.0",
     "resolved": "https://registry.npmjs.org/internal-slot/-/internal-slot-1.1.0.tgz",
     "integrity": "sha512-
4gd7VpWNQNB4UKKCFFVcp1AVv+FMOgs9NKzjHKusc8jTMhd5eL1NqQqOpE0KzMds804/yHlg
lp3uxgluOqAPLw==",
     "dev": true,
     "license": "MIT",
     "dependencies": {
      "es-errors": "^1.3.0",
      "hasown": "^2.0.2",
```

```
      "side-channel": "^1.1.0"
     },
     "engines": {
      "node": ">= 0.4"
     }
    },
    "node_modules/is-array-buffer": {
     "version": "3.0.5",
     "resolved": "https://registry.npmjs.org/is-array-buffer/-/is-array-buffer-3.0.5.tgz",
     "integrity": "sha512-
DDfANUiiG2wC1qawP66qlTugJeL5HyzMpfr8lLK+jMQirGzNod0B12cFB/9q838Ru27sBwfw78/rdo
U7RERz6A==",
     "dev": true,
     "license": "MIT",
     "dependencies": {
      "call-bind": "^1.0.8",
      "call-bound": "^1.0.3",
      "get-intrinsic": "^1.2.6"
     },
     "engines": {
      "node": ">= 0.4"
     },
     "funding": {
      "url": "https://github.com/sponsors/ljharb"
     }
    },
    "node_modules/is-arrayish": {
     "version": "0.2.1",
     "resolved": "https://registry.npmjs.org/is-arrayish/-/is-arrayish-0.2.1.tgz",
     "integrity": "sha512-
zz06S8t0ozoDXMG+ube26zeCTNXcKIPJZJi8hBrF4idCLms4CG9QtK7qBl1boi5ODzFpjswb5JPmH
CbMpjaYzg==",
```

```
      "license": "MIT"
    },
    "node_modules/is-async-function": {
      "version": "2.1.1",
      "resolved": "https://registry.npmjs.org/is-async-function/-/is-async-function-2.1.1.tgz",
      "integrity": "sha512-9dgM/cZBnNvjzaMYHVoxxfPj2QXt22Ev7SuuPrs+xav0ukGB0S6d4ydZdEiM48kLx5kDV+QBPrpVnFyefL8kkQ==",
      "dev": true,
      "license": "MIT",
      "dependencies": {
        "async-function": "^1.0.0",
        "call-bound": "^1.0.3",
        "get-proto": "^1.0.1",
        "has-tostringtag": "^1.0.2",
        "safe-regex-test": "^1.1.0"
      },
      "engines": {
        "node": ">= 0.4"
      },
      "funding": {
        "url": "https://github.com/sponsors/ljharb"
      }
    },
    "node_modules/is-bigint": {
      "version": "1.1.0",
      "resolved": "https://registry.npmjs.org/is-bigint/-/is-bigint-1.1.0.tgz",
      "integrity": "sha512-n4ZT37wG78iz03xPRKJrHTdZbe3IicyucEtdRsV5yglwc3GyUfbAfpSeD0FJ41NbUNSt5wbhqfp1fS+BgnvDFQ==",
      "dev": true,
      "license": "MIT",
```

```
  "dependencies": {
    "has-bigints": "^1.0.2"
  },
  "engines": {
    "node": ">= 0.4"
  },
  "funding": {
    "url": "https://github.com/sponsors/ljharb"
  }
},
```

Додаток Г. Ілюстративна частина

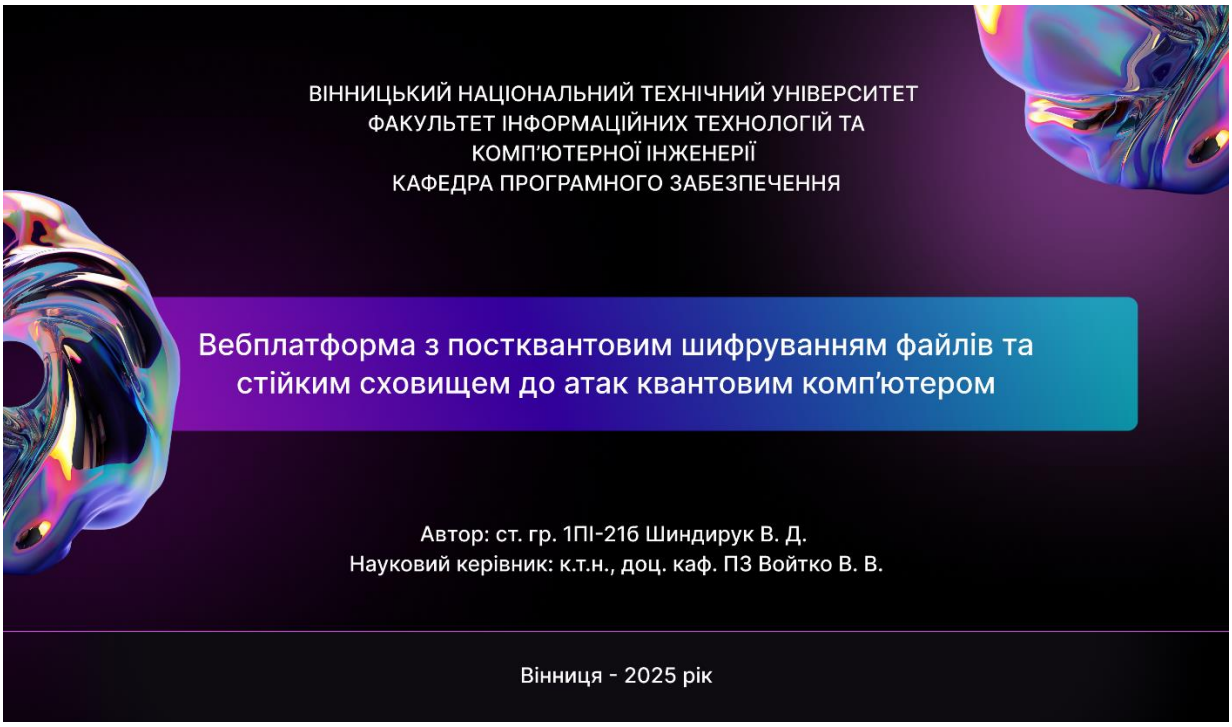


Рисунок Г.1 – Титульний слайд

Рисунок Г.2 – Актуальність розробки

Рисунок Г.3 – Мета, об'єкт, предмет дослідження

Рисунок Г.4 –Задачі розробки

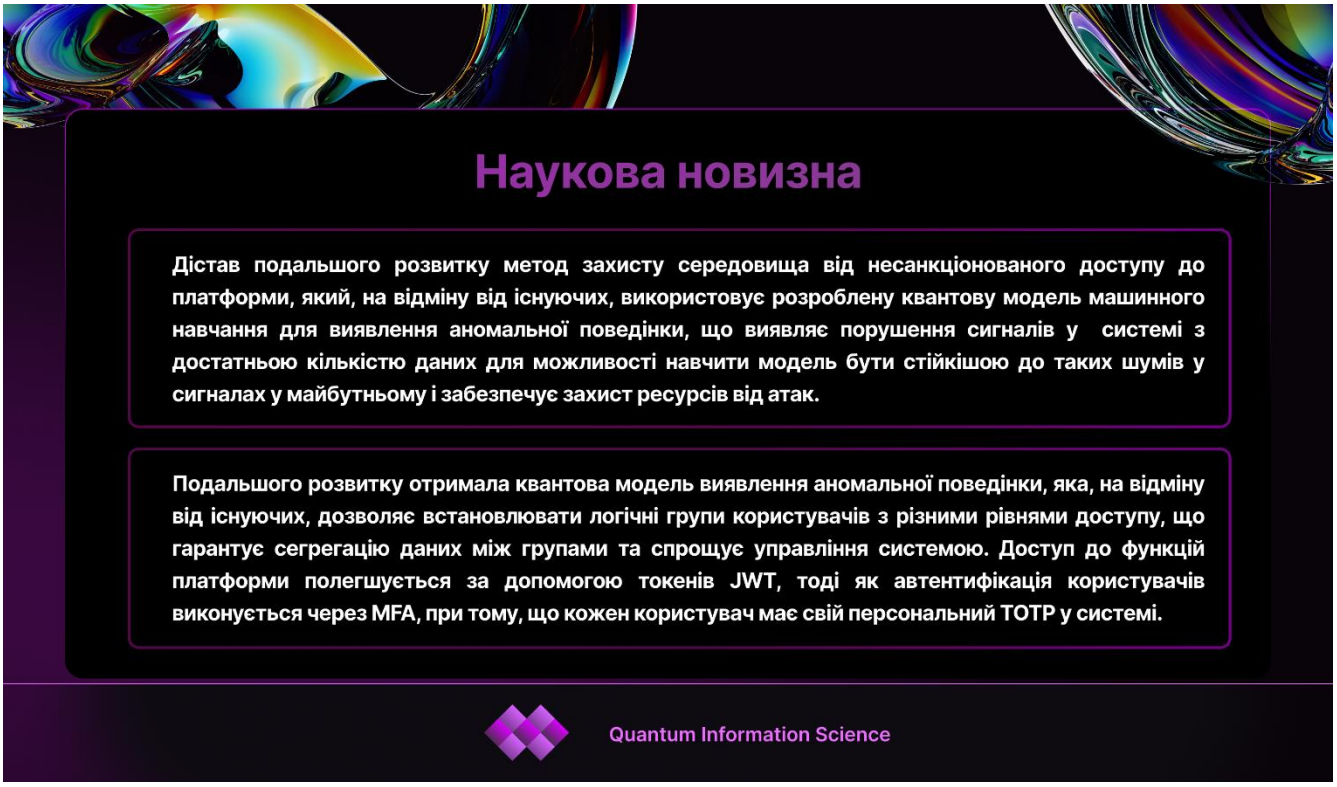


Рисунок Г.5 –Наукова новизна

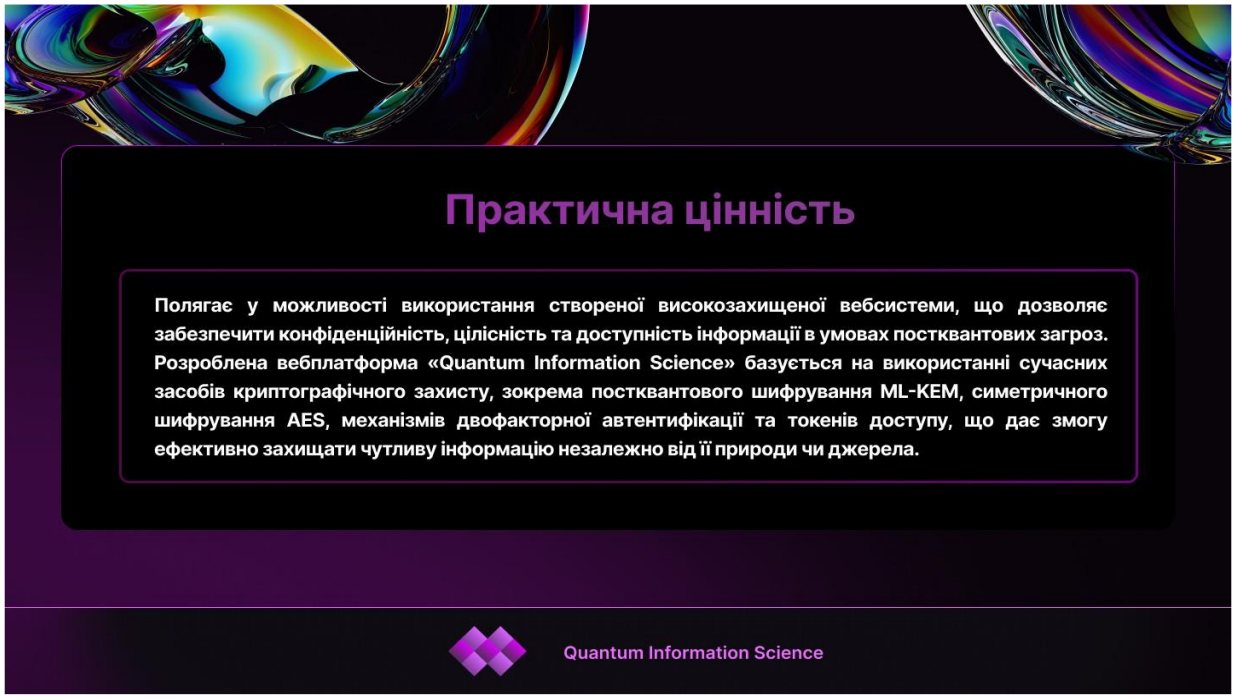


Рисунок Г.6 –Практична цінність

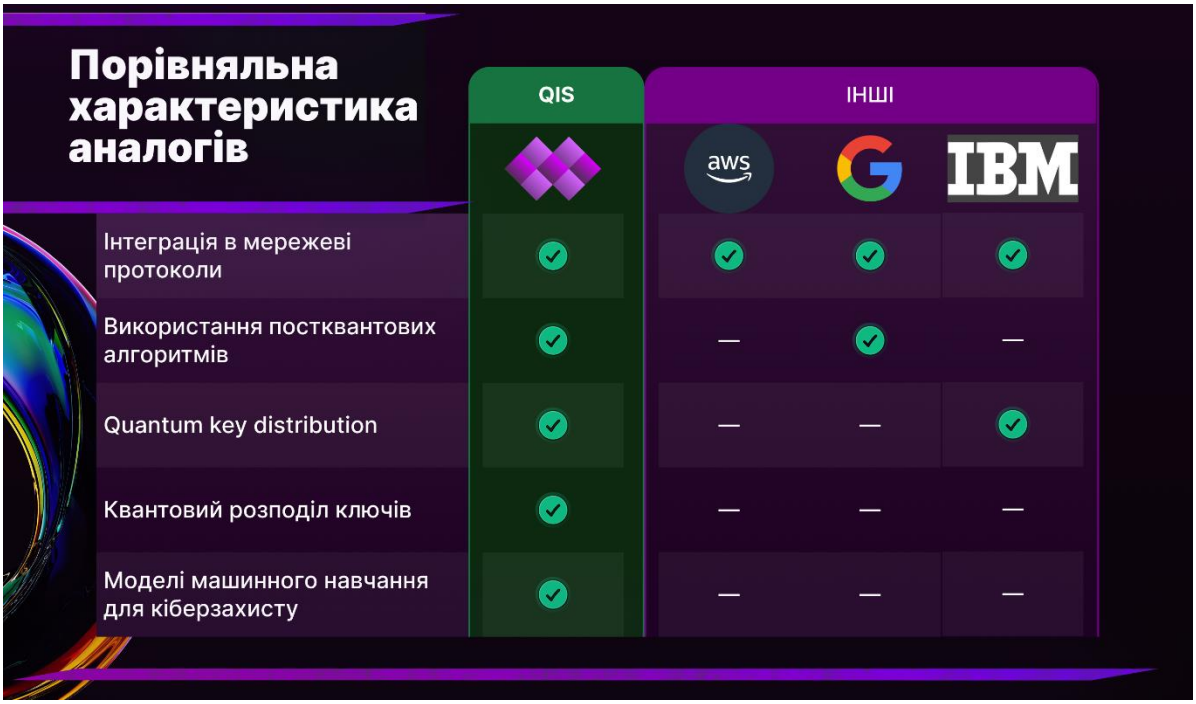


Рисунок Г.7 –Порівняльна характеристика аналогів

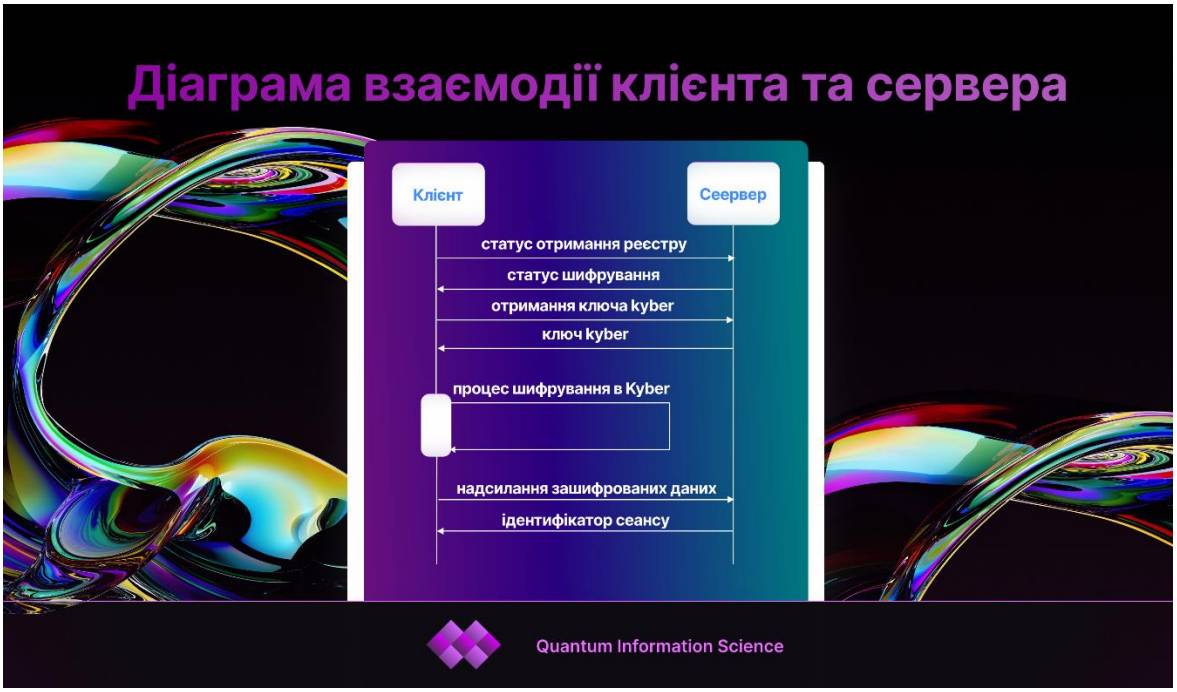


Рисунок Г.8 – Діаграма взаємодії клієнта та сервера

Рисунок Г.9 – Загальна модель роботи програми

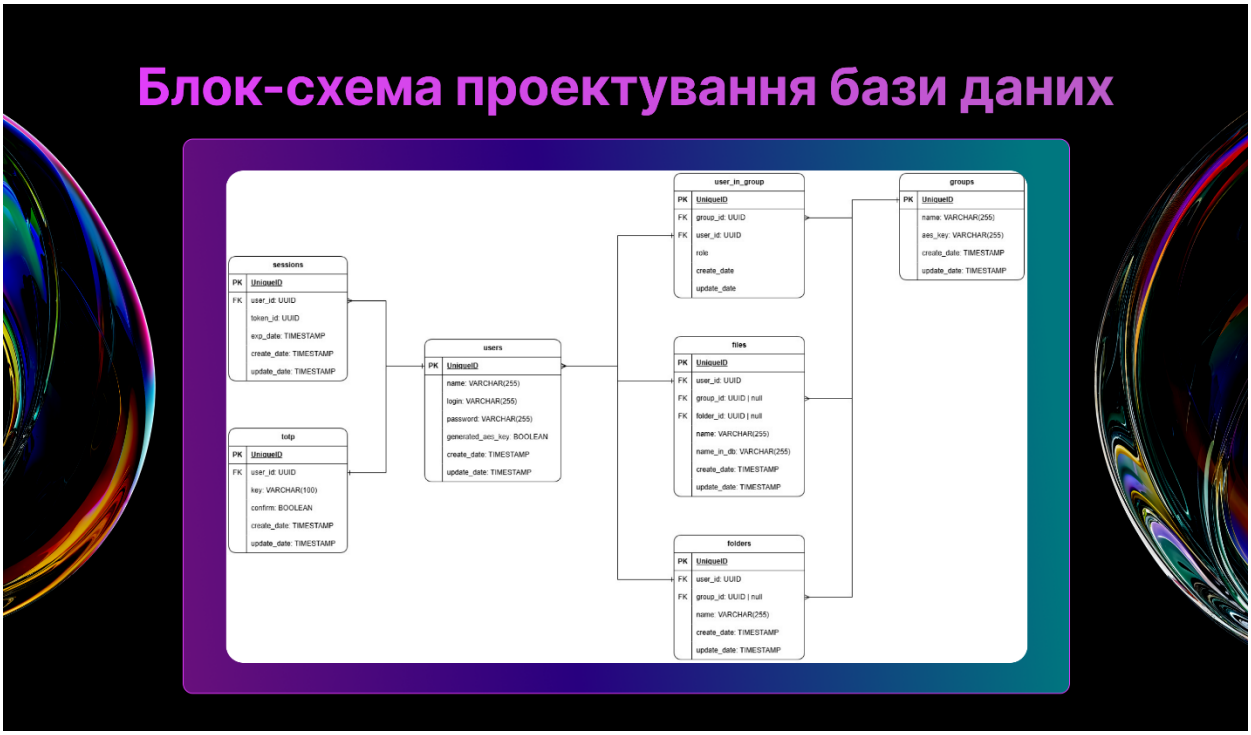


Рисунок Г.10 – Блок-схема проєктування бази даних

Рисунок Г.11 – Алгоритм шифрування AES

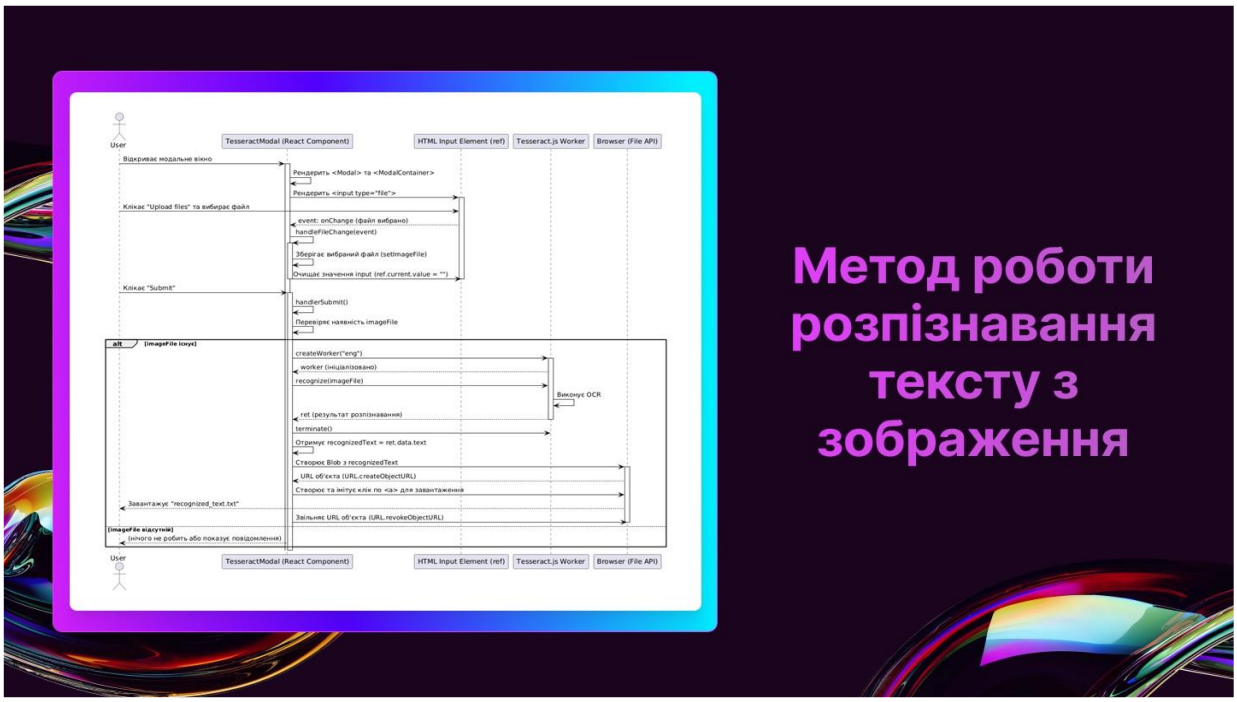


Рисунок Г.12 – Метод роботи розпізнавання тексту з зображення

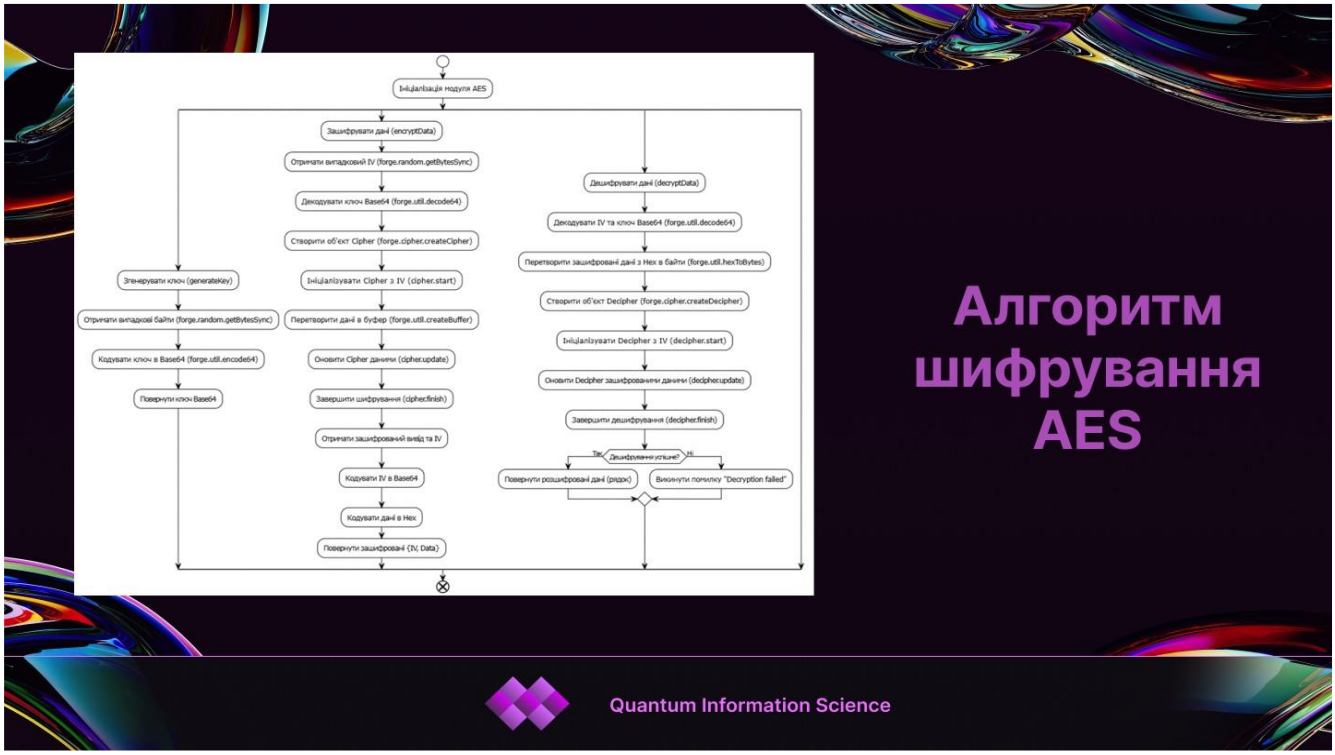


Рисунок Г.13 – Алгоритм шифрування AES

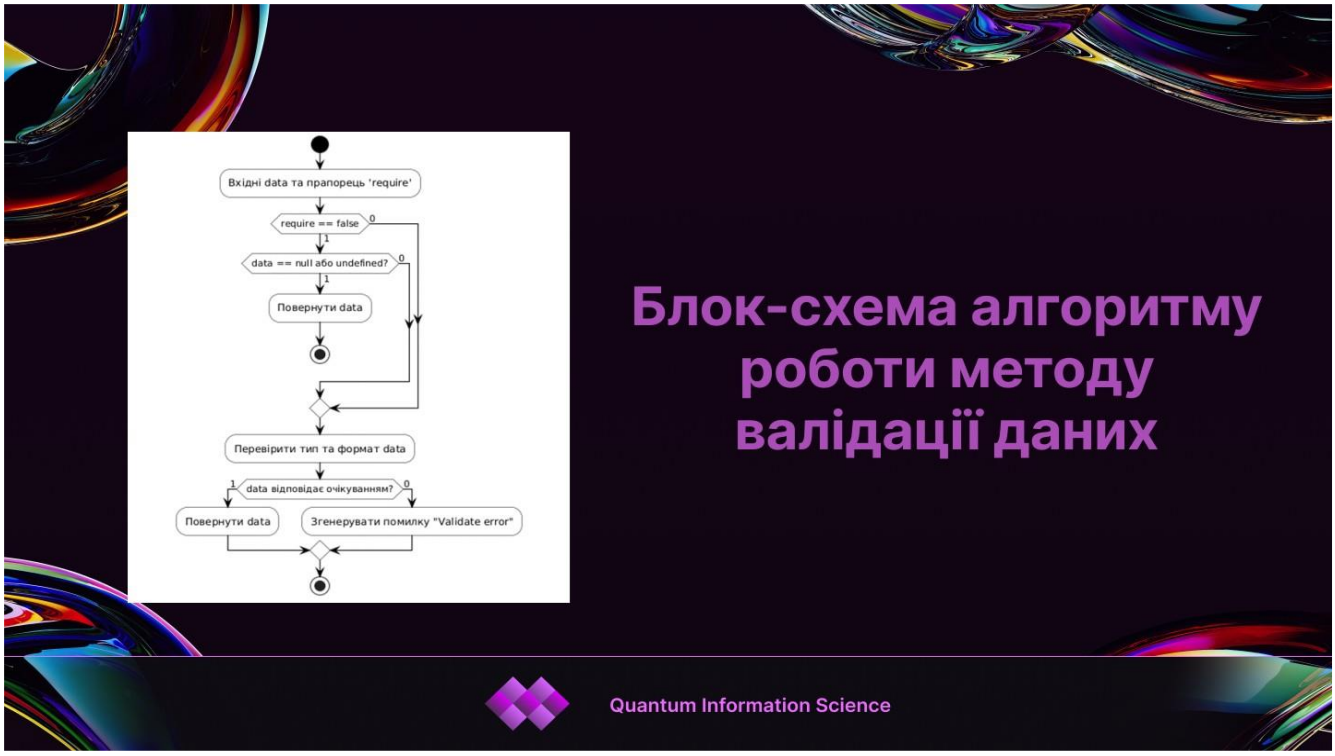


Рисунок Г.14 – Блок-схема алгоритму роботи методу валідації даних

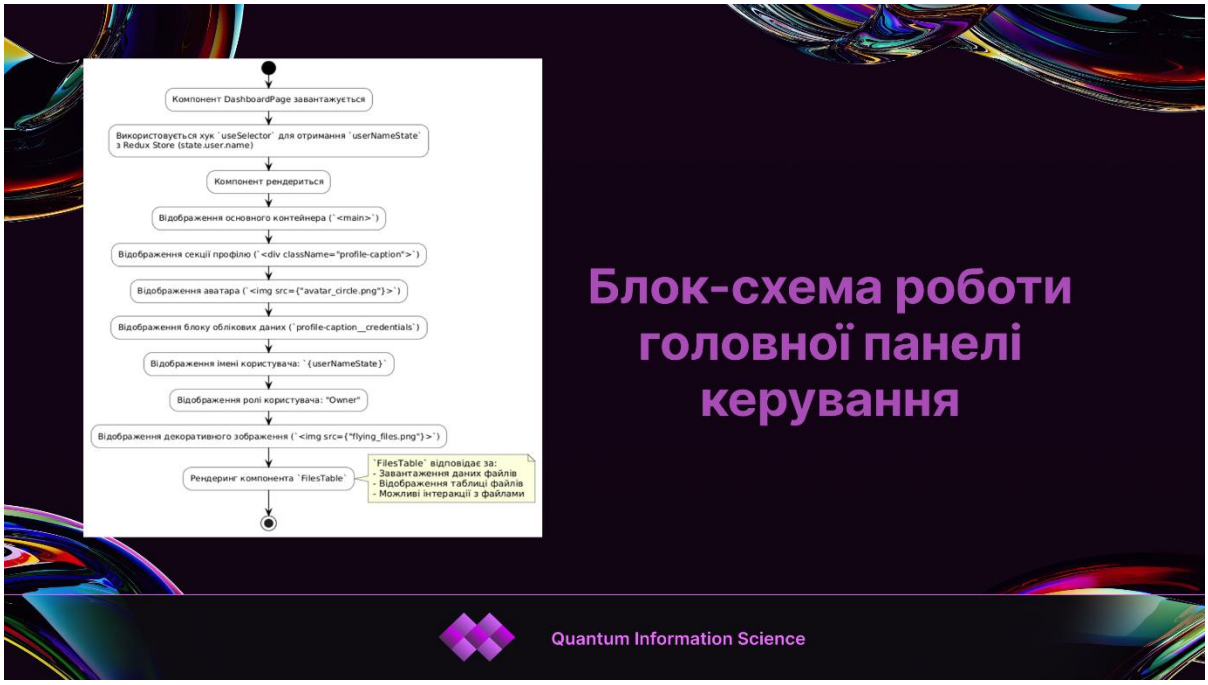


Рисунок Г.15 – Блок-схема роботи головної панелі керування

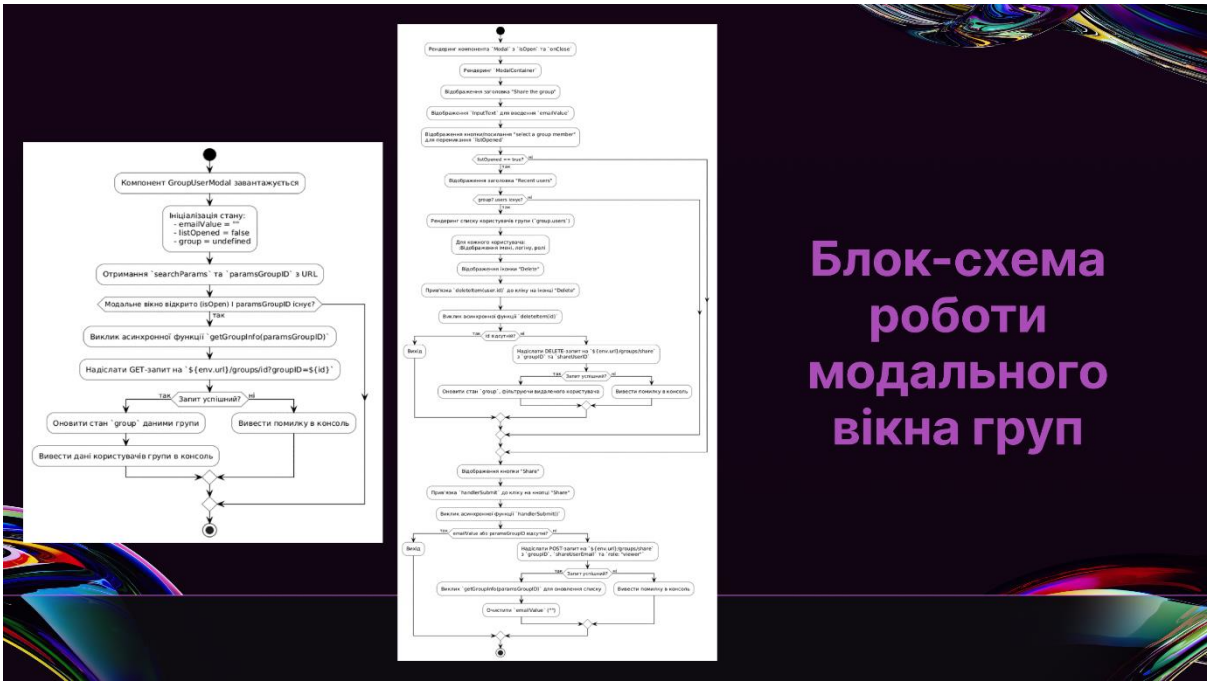


Рисунок Г.16 – Блок-схема роботи модального вікна груп

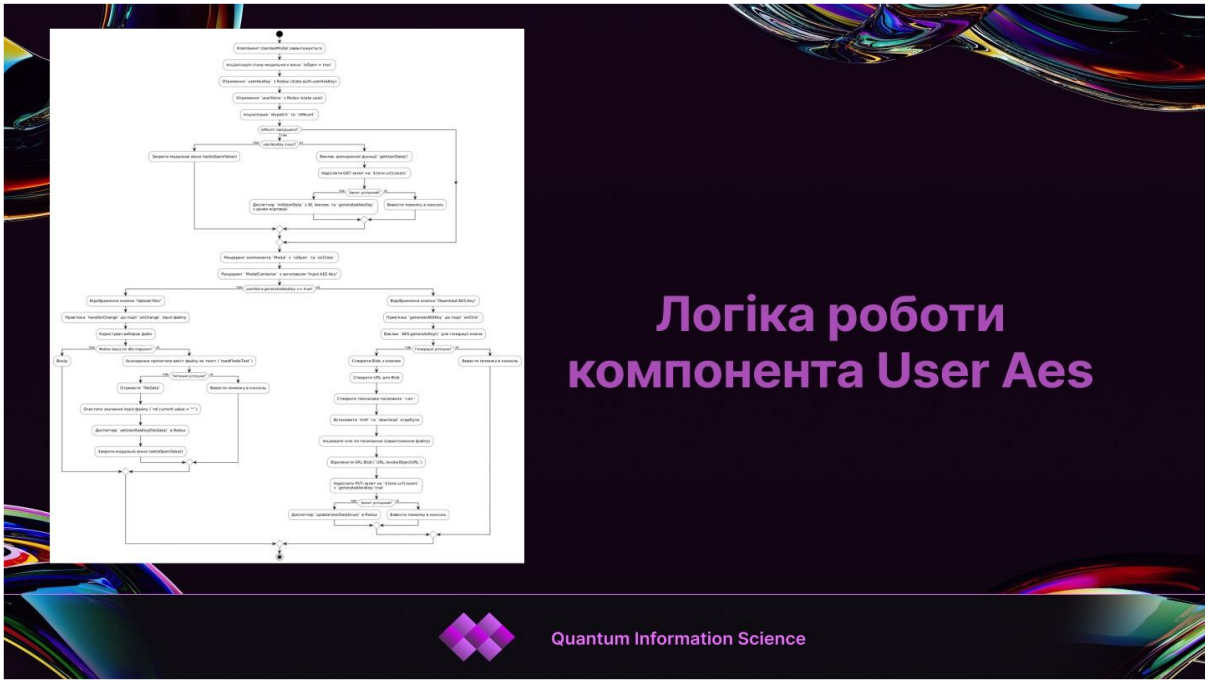


Рисунок Г.17 – Логіка роботи компонента User Aes

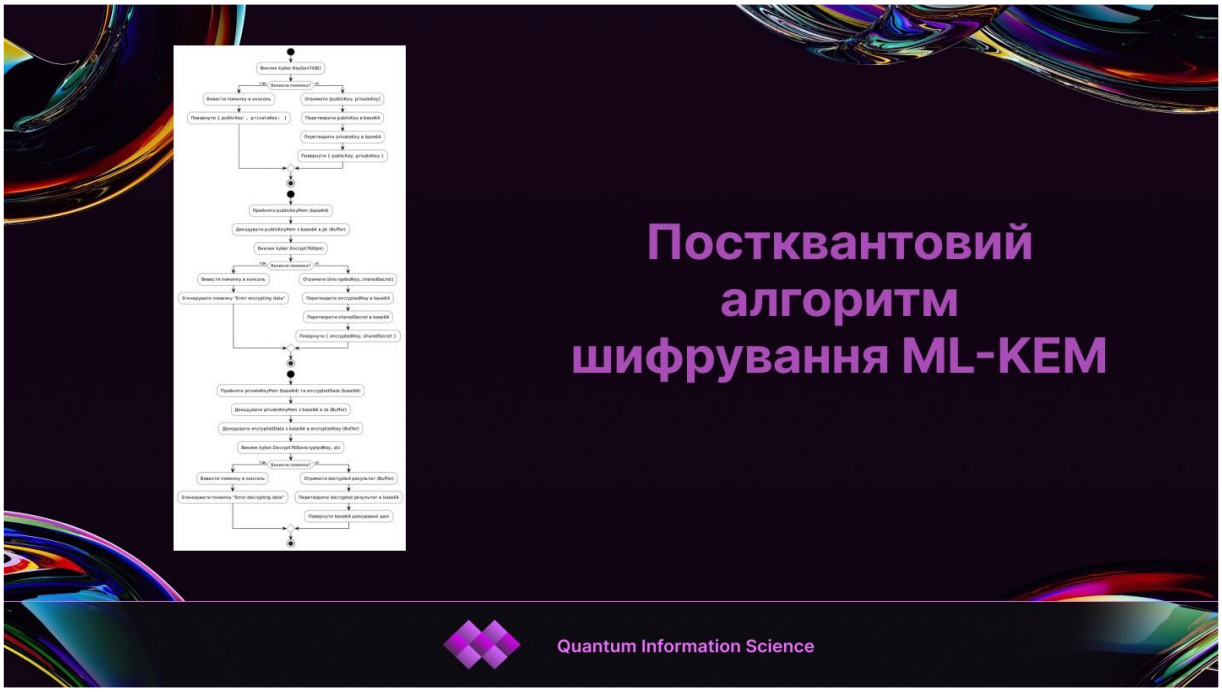


Рисунок Г.18 – Постквантовий алгоритм шифрування ML-KEM

Рисунок Г.19 – Модель квантового машинного навчання

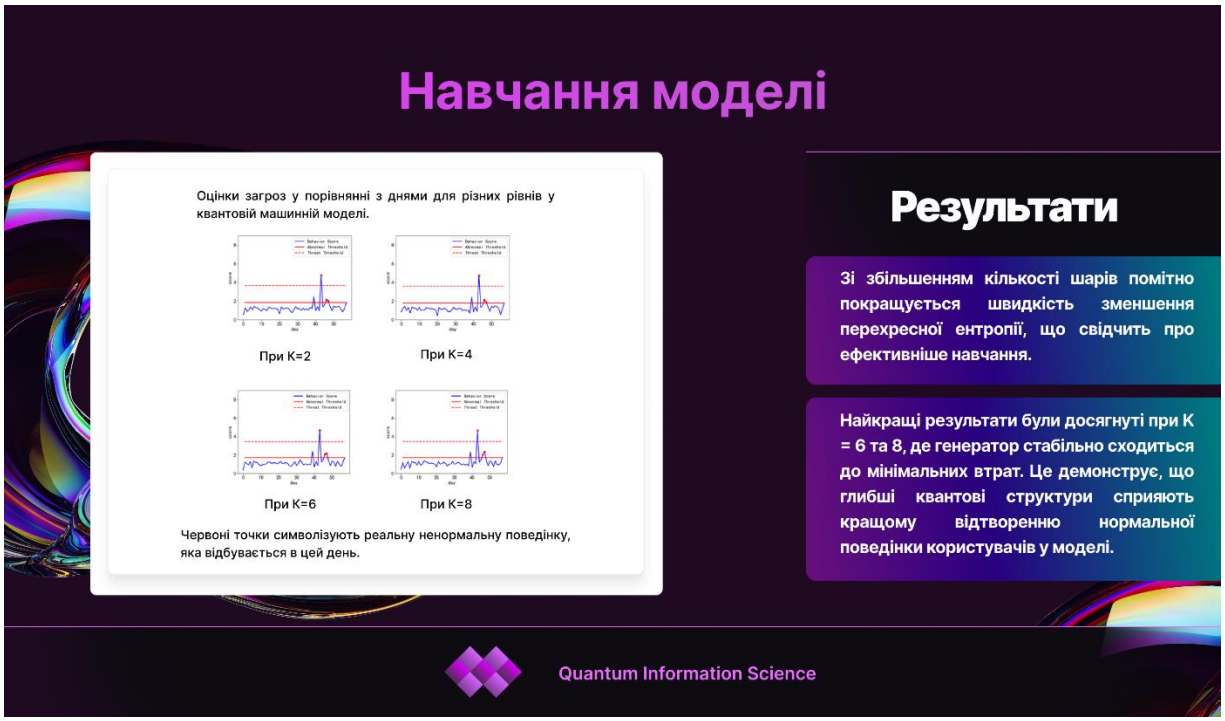


Рисунок Г.20 – Навчання моделі

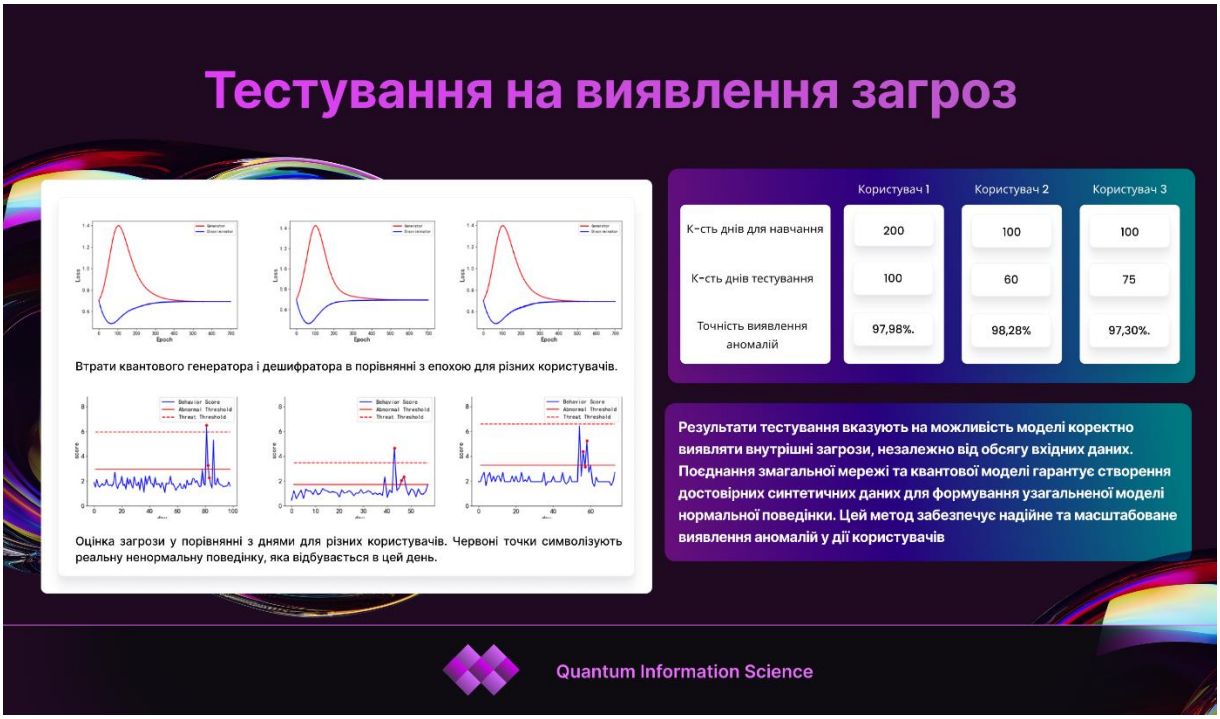


Рисунок Г.21 – Тестування на виявлення загрозі

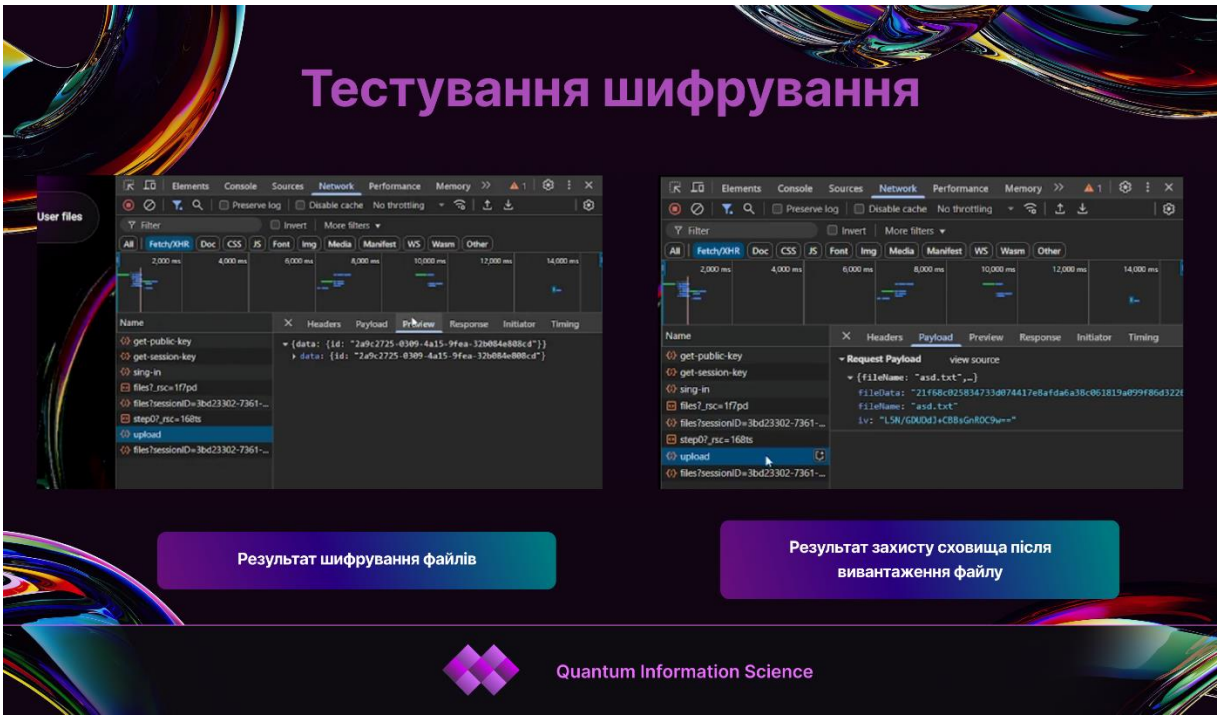


Рисунок Г.22 – Тестування шифрування

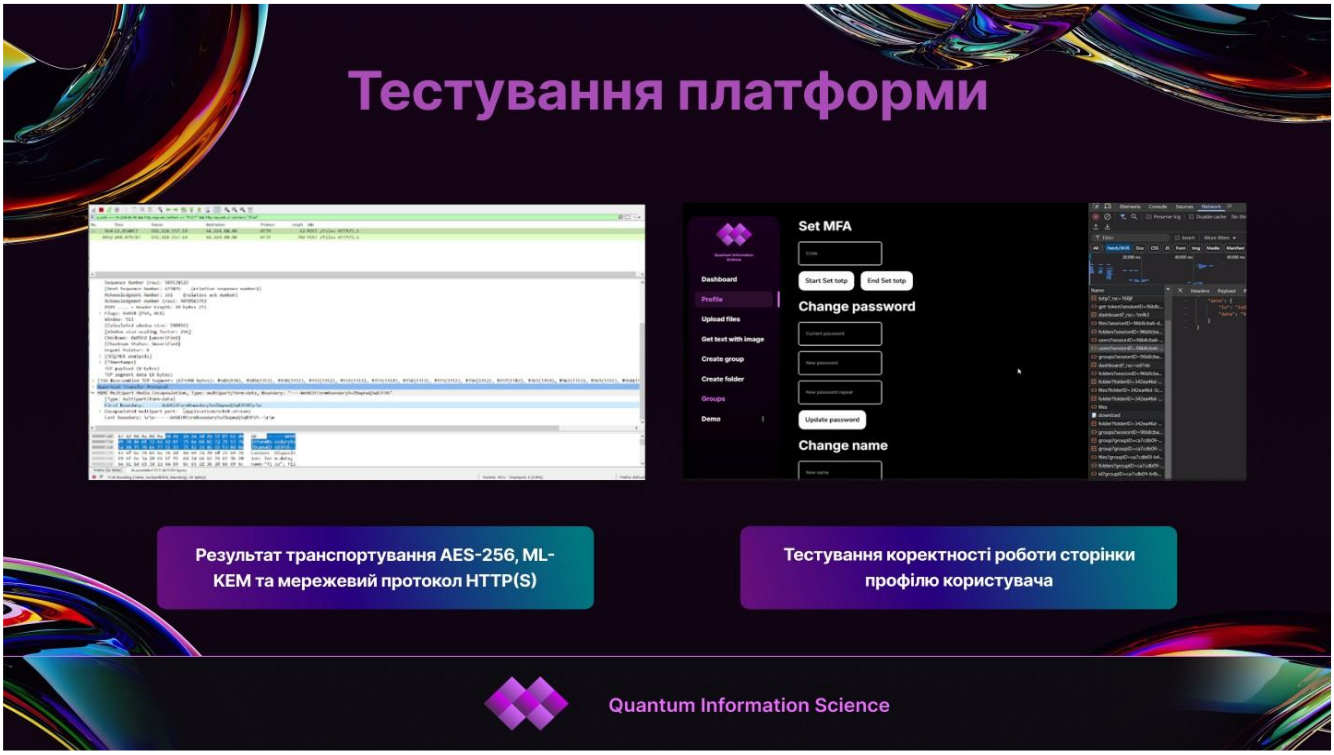


Рисунок Г.23 – Тестування платформи

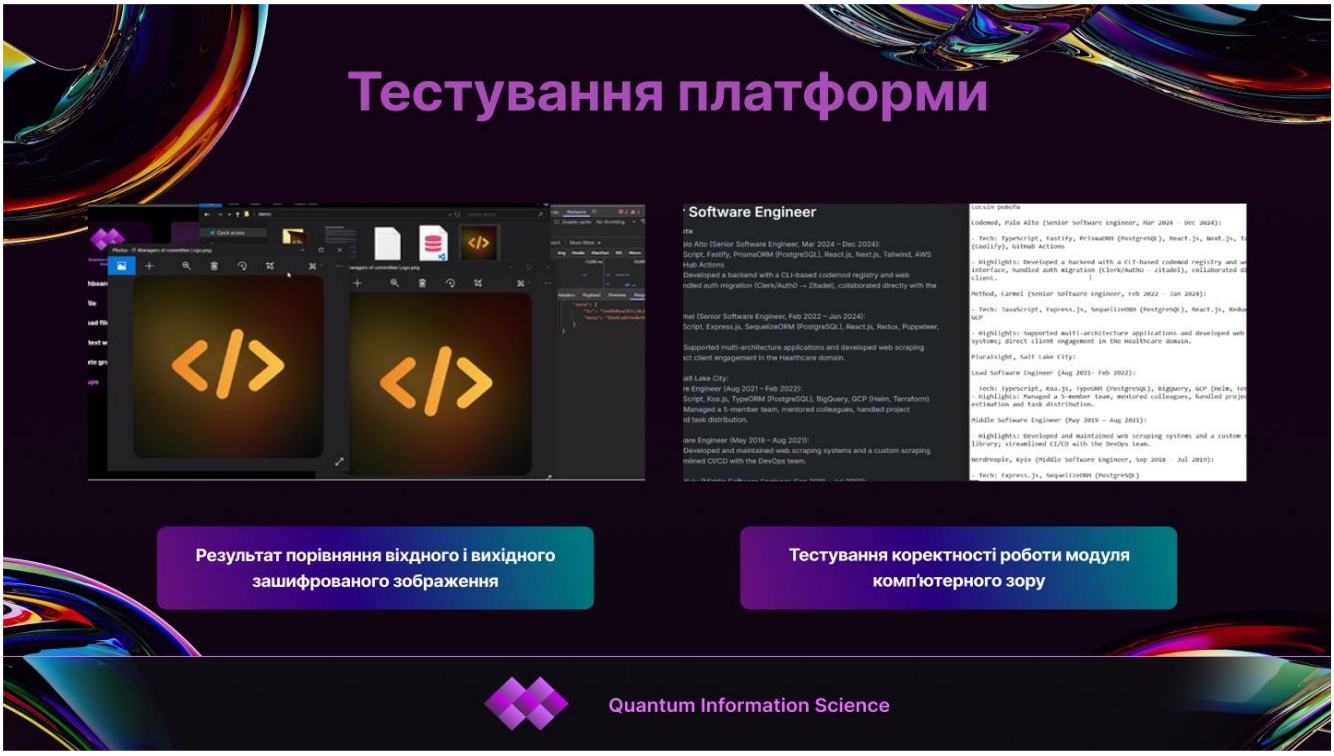


Рисунок Г.24 – Тестування платформи

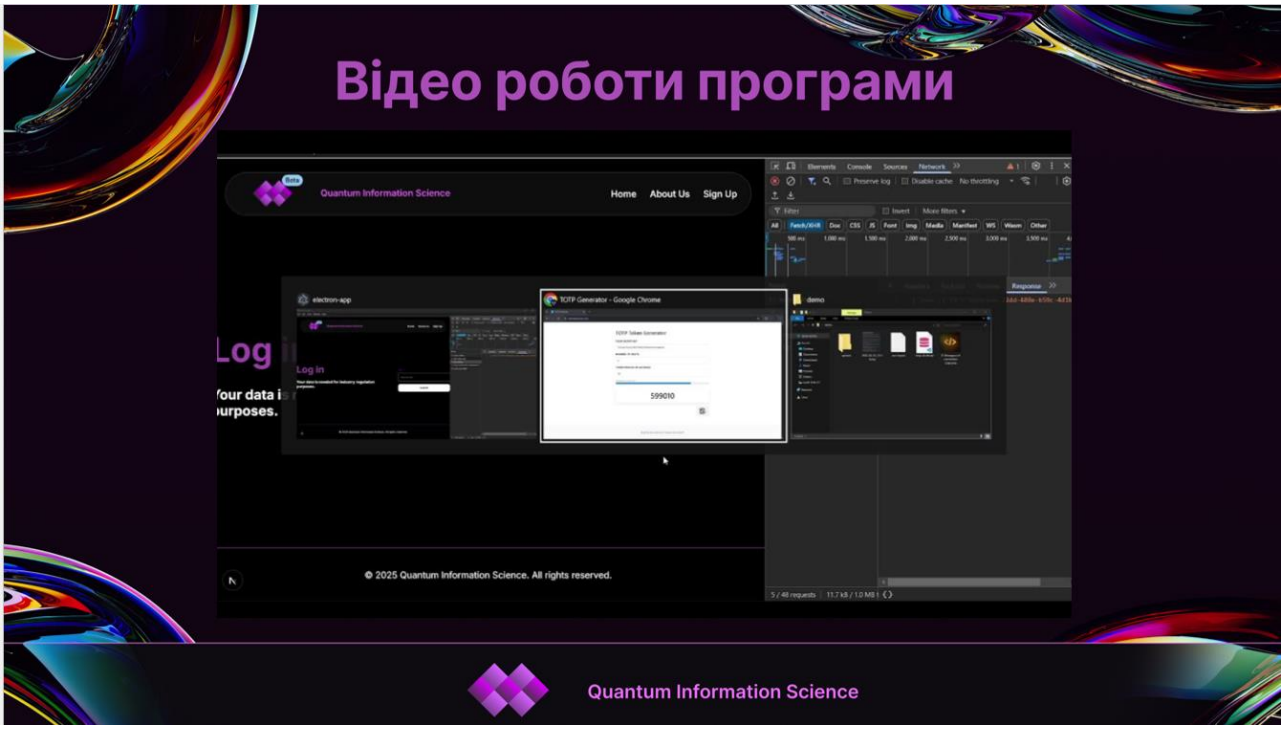


Рисунок Г.25 – Відео роботи платформи

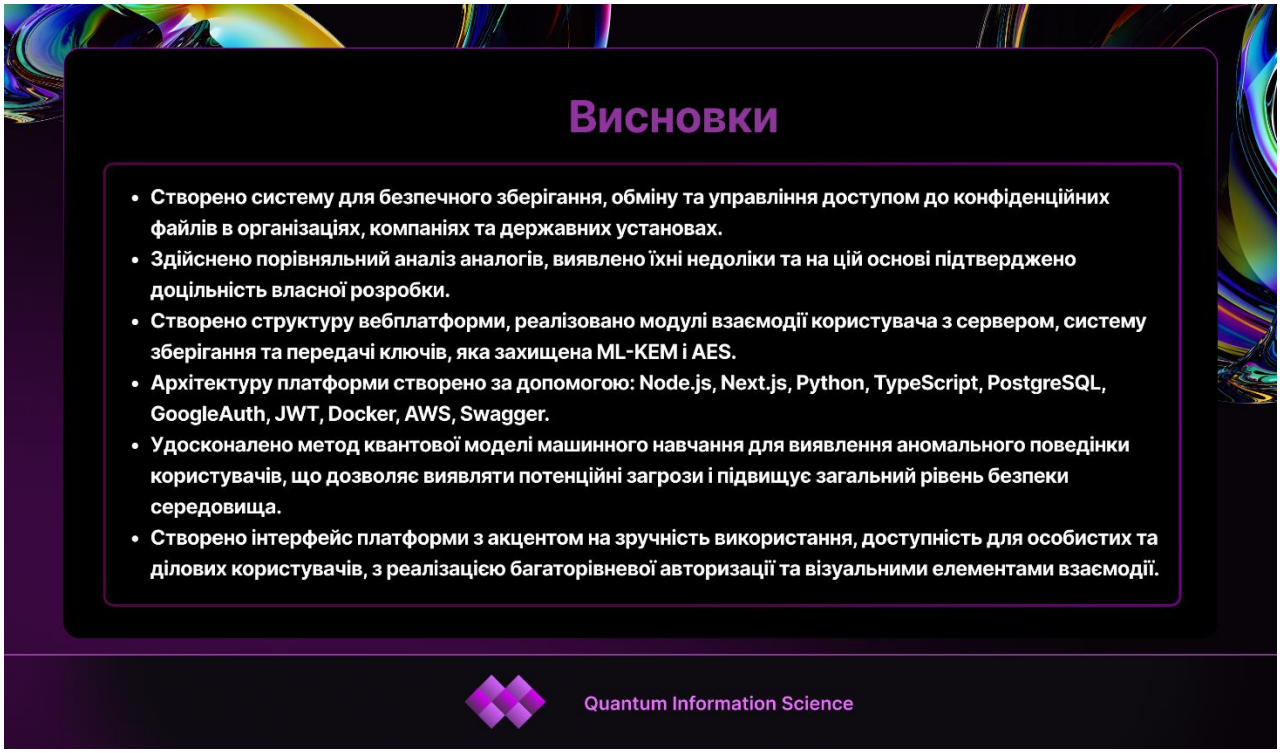


Рисунок Г.26 – Висновки

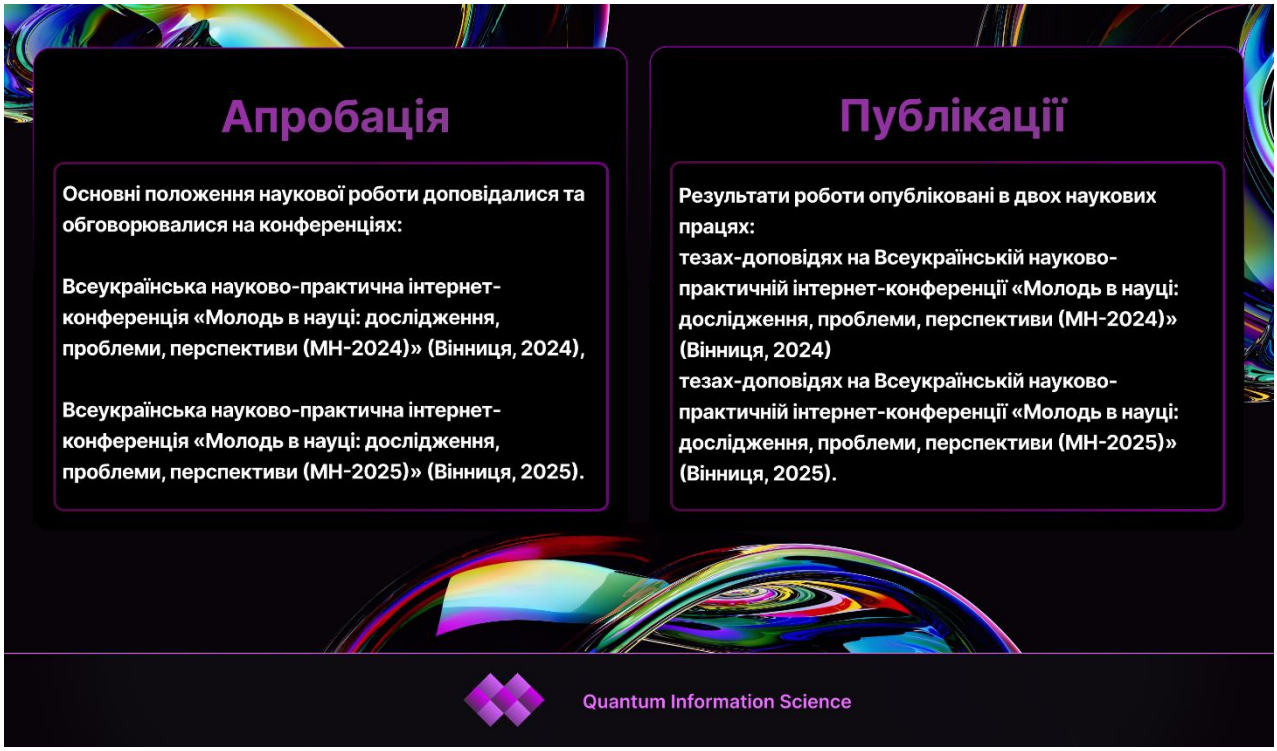


Рисунок Г.27 – Апробація та публікації

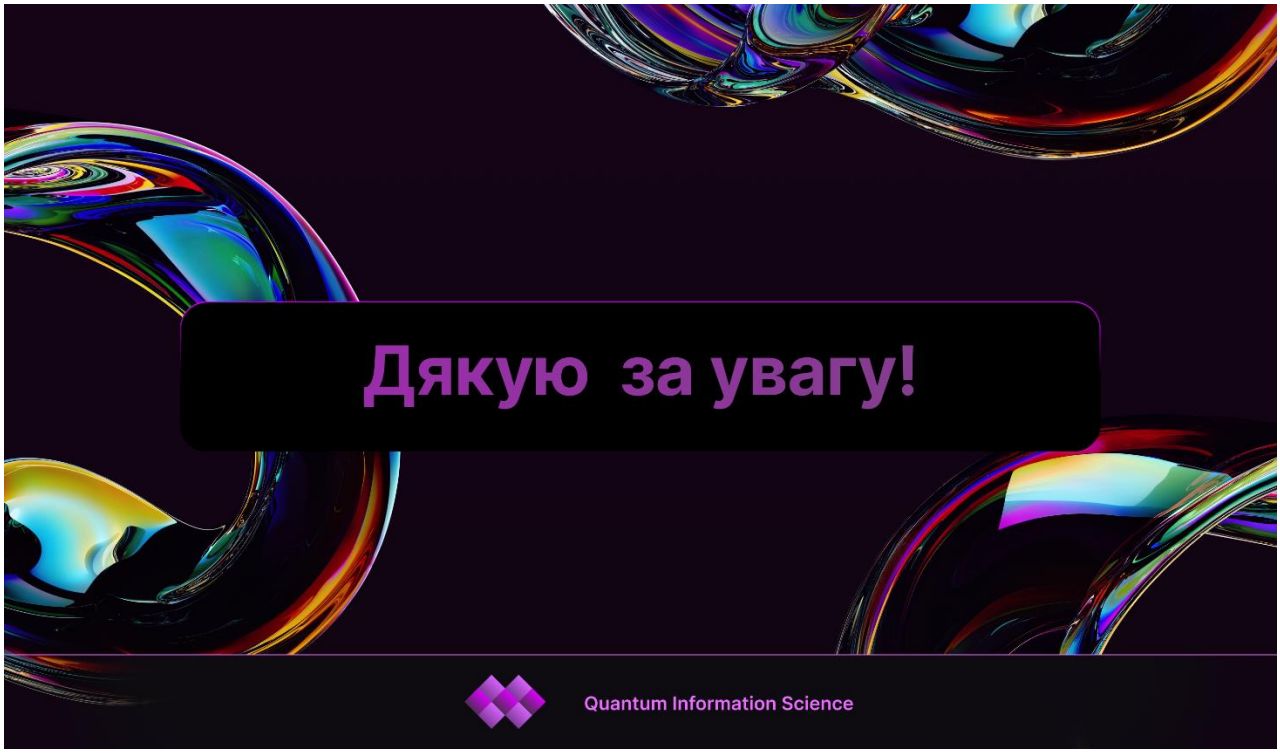


Рисунок Г.28 – Фінальний слайд