Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження технологій оптимального зберігання зображень в гібридних системах з використанням блокчейн та традиційних баз даних
(тема)

Виконав:
студент 2 курсу, групи ІПЗм 22-3

Мянд Д. Ю.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова

Керівник доц. Кириченко І. В.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Мянду Денису Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження технологій оптимального зберігання зображень в гібридних системах з використанням блокчейн та традиційних баз даних»
Затверджена наказом по університету від 29.03.2024р № 250 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 13.06.2024
3. Вихідні дані до роботи опис досліджуваних систем зберігання даних та технологій, які вони використовують, вимоги до розробки схеми бази даних для збору даних для експерименту.
4. Перелік питань, що потрібно опрацювати в роботі
аналіз та порівняння існуючих технологій для систем зберігання даних, вибір підходящих систем для дослідження, проведення експериментів та аналіз отриманих результатів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі та постановка задачі	01.03 – 12.04.24	*виконано*
2	Аналіз та вибір досліджуваних систем зберігання даних	13.04 – 21.04.24	*виконано*
3	Аналіз та моделювання предметної області	22.04 – 05.05.24	*виконано*
4	Планування експериментів	06.05 – 12.05.24	*виконано*
5	Збір даних для експериментів	13.05 – 16.05.24	*виконано*
6	Експериментальні дослідження	17.05 – 23.05.24	*виконано*
7	Аналіз результатів експериментальних досліджень та розробка рекомендацій	24.05 – 01.06.24	*виконано*
8	Підготовка пояснювальної записки	02.06 – 05.06.24	*виконано*
9	Підготовка презентації та доповіді	06.06 – 09.06.24	*виконано*
10	Нормоконтроль	10.06.2024	*виконано*
11	Рецензування	10.06.2024	*виконано*
12	Занесення диплома в електронний архів	10.06.2024	*виконано*
13	Попередній захист	10.06.2024	*виконано*
14	Допуск до захисту у зав. кафедри	11.06.2024	*виконано*

Дата видачі завдання 29 березня 2024р.

Студент (ка) _____ Мянд Д.Ю.
(підпис)

Керівник роботи _____ доц. Кириченко І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка містить: 65 с., 48 рис., 3 табл., 20 джерел.

БЛОКЧЕЙН ТЕХНОЛОГІЇ, ГІБРИДНІ СИСТЕМИ ЗБЕРІГАННЯ ДАНИХ, ЗБЕРІГАННЯ ЗОБРАЖЕНЬ, ТОКЕНИ, ТРАДИЦІЙНІ БАЗИ ДАНИХ.

Об’єктом дослідження є технології зберігання зображень в гібридних системах, які комбінують використання блокчейн-технологій та традиційних баз даних для досягнення оптимального результату в зберіганні цифрових зображень.

Метою роботи є аналіз різних технічних підходів і систем зберігання, які поєднують блокчейн і традиційні бази даних для забезпечення ефективного управління та зберіганням великих обсягів зображень.

Методами розробки та проектування є аналіз предметної галузі дослідження, оцінка конкурентних систем зберігання даних за визначеними параметрами експериментальним методом.

У результаті виконання кваліфікаційної роботи було порівняно п’ять обраних систем зберігання даних. Для цього було розроблено Python-програму для визначення продуктивності зберігання і швидкості отримання доступу до зображень. В ході проведення експерименту було розраховано коефіцієнти лінійної адитивної згортки для визначення оптимальної системи зберігання даних.

BLOCKCHAIN TECHNOLOGIES, HYBRID DATA STORAGE SYSTEM, IMAGE STORING, TOKENS, TRADITIONAL DATABASES.

The object of the research is the technologies for storing images in hybrid systems that combine the use of blockchain technologies and traditional databases to achieve optimal results in the storage of digital images.

The purpose of the work is to analyze various technical approaches and storage systems that integrate blockchain and traditional databases to ensure effective management and storage of large volumes of images.

The development and design methods are the analysis of the problem area of the study, evaluation of competitive data storage systems according to defined parameters by experimental method.

As a result of the qualification work, five selected data storage systems were compared. For this, a Python program was developed to determine storage performance and image access speed. During the experiment, coefficients of linear additive convolution were calculated to determine the optimal data storage system.

Заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElArKhNURE.

Я, Мянд Денис Юрійович, студент(ка) гр. ІПЗм-22-3, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження технологій оптимального зберігання зображень в гібридних системах з використанням блокчейн та традиційних баз даних», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(на) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

Перелік скорочень 8
Вступ 9
1 Опис проблемної галузі 13
1.1 Аналіз предметної області 13
1.2 Постановка задачі 14
2 Опис прийнятих проєктних рішень 16
2.1 Аналіз конкуруючих систем 16
2.2 Визначення ключових критеріїв 16
2.3 Визначення методу аналізу 23
2.4 Обґрунтування програмного забезпечення для експерименту 24
3 Опис програмної реалізації 25
4 Опис експерементальних досліджень 34
Висновки 38
Перелік джерел посилання 39
ДОДАТОК А Перелік джерел посилання за науковими напрямами керівника та науковців кафедри програмної інженерії 42
ДОДАТОК Б Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ 43
ДОДАТОК В Слайди презентації 44
ДОДАТОК Г Апробація результатів роботи 54
ДОДАТОК Д Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015 65

ПЕРЕЛІК СКОРОЧЕНЬ

BTFS – BitTorrent File System

СУБД – Система управління базами даних

TiB – Tebibyte

AES – Advanced Encryption Standard

PoR – Proof of Retrieveability

PoRep – Proof of Replication

DDoS – Distributed Denial-of-Service

PoW – Proof of Work

API – Application Programming Interface

GiB – Gibibyte

БД – База даних

SSL/TLS – Secure Socket Layer and Transport Layer Security

IP – Internet Protocol

SSH – Secure Shell

LDAP – Lightweight Directory Access Protocol

PAM – Pluggable Authentication Modules

GSSAPI – Generic Security Service Application Program Interface

WAL – Write-Ahead Logging

PITR – Point-in-Time Recovery

TBD – To Be Decided

PIL – Python Imaging Library

ВСТУП

Сучасний цифровий світ прогресує стрімкими темпами, вимагаючи ефективних та безпечних рішень щодо зберігання та управління великими обсягами цифрових даних, зокрема цифровими зображеннями. З впровадженням новітніх технологій, таких як блокчейн, в сфері зберігання даних виникає можливість створення гібридних систем, які комбінують переваги блокчейн-технологій та традиційних баз даних для оптимального управління цифровими ресурсами.

Сучасне суспільство стикається з суттєвим збільшенням кількості цифрових зображень. Фотографії, відео, медичні знімки, масиви даних у наукових дослідженнях – всі ці дані потребують ефективного зберігання та управління. Отже, з'являється потреба в технологічних рішеннях, що забезпечать безпеку та доступність до цих обсягів даних.

Цифрові зображення стали невід'ємною складовою багатьох сфер життя, включаючи медицину, науку, мистецтво, технології. Вони використовуються у різних сферах: від діагностики у медицині до візуального впливу у маркетингу. Збереження цих зображень у безпечному, ефективному та надійному середовищі стає ключовим аспектом для подальшого розвитку цих галузей.

Збільшення кількості даних також викликає питання конфіденційності та цілісності. Забезпечення безпеки та непорушності цифрових зображень від несанкціонованого доступу або порушень стає критичним завданням у сферах, де дані мають важливе значення.

Традиційні методи зберігання даних можуть бути обмеженими у забезпеченні високої швидкодії, масштабованості та безпеки в умовах зростаючого обсягу цифрових зображень. Тому виникає потреба у нових технологіях, які можуть комбінувати найкращі аспекти різних систем для досягнення оптимального результату.

Блокчейн-технологія надає нові можливості для зберігання та управління даними, забезпечуючи безпеку, недоступність для втручання та відновлення даних. Комбінування блокчейну з традиційними базами даних у гібридних

системах може стати проривним кроком у збереженні та управлінні цифровими зображеннями.

Дослідження технологій зберігання зображень у гібридних системах з блокчейн та традиційними базами даних є актуальним через кілька проблем.

Безпека. Цифрові зображення стають об'єктом кіберзагроз, тому потрібні ефективні заходи захисту від несанкціонованого доступу.

Масштабованість. Збільшення обсягів даних потребує систем, які можуть ефективно працювати з великими обсягами інформації, не втрачаючи продуктивності.

Інтеграція. Потрібно поєднати нові технології із вже існуючими системами управління даними, а це може вимагати оптимізації процесів.

Цілісність. Важливо забезпечити, щоб дані були цілісними та відновлювалися у випадку помилок або кібератак.

Ефективне управління. Контроль за доступом до даних, їхнє індексування та пошук важливі для управління цифровими зображеннями.

Ці проблеми стають більш актуальними через зростання обсягів цифрових даних у різних сферах. Вирішення їх через гібридні системи зберігання може забезпечити нові підходи до збереження та управління цифровими зображеннями, спрямовані на забезпечення їхньої безпеки, надійності та доступності.

Об'єктом дослідження є технології зберігання зображень у гібридних системах, які поєднують блокчейн та традиційні бази даних. Мета роботи – дослідження, спрямоване на вивчення ефективних методів зберігання, управління та захисту цифрових зображень у таких гібридних системах.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– аналіз технічних можливостей гібридних систем: огляд та аналіз можливостей систем, які поєднують блокчейн-технології та традиційні бази даних для зберігання зображень. Це включає вивчення архітектури систем, їхніх можливостей у збереженні та управлінні обсягами цифрових даних;

- оцінка безпеки та конфіденційності даних: визначення рівня безпеки, яку можуть забезпечити гібридні системи зберігання зображень. Особлива увага приділяється заходам захисту даних від несанкціонованого доступу та збереженню конфіденційності;
- масштабованість та ефективність: оцінка можливостей системи працювати з великими обсягами цифрових зображень без втрати продуктивності та якості обслуговування;
- відновлення та цілісність даних: визначення можливостей системи забезпечувати відновлення та цілісність цифрових зображень у випадку можливих помилок або атак на дані;

Об'єктом дослідження є технології зберігання зображень у гібридних системах, які поєднують використання блокчейн-технологій і традиційних баз даних для досягнення оптимальних результатів у зберіганні цифрових зображень.

Метою роботи є аналіз різних технічних підходів і систем зберігання, що об'єднують блокчейн і традиційні бази даних для забезпечення ефективного управління та зберігання великих обсягів зображень.

Методами розробки та проектування є аналіз предметної області дослідження, оцінка конкурентних систем зберігання даних за визначеними параметрами експериментальним шляхом.

Це дослідження спрямоване на вирішення практичних завдань у зберіганні та управлінні цифровими зображеннями, що є важливим елементом сучасного цифрового середовища.

Дане дослідження відповідає на термінові завдання управління та збереження цифрових зображень, що виходять за межі стандартних підходів та вимагають новаторських рішень для забезпечення безпеки, ефективності та надійності у цифровому віці.

Отже, у контексті стрімкого зростання обсягів цифрових зображень та загроз кібербезпеки, дослідження технологій зберігання у гібридних системах з блокчейн та традиційними базами даних набуває великого значення. Це дослідження актуальне, оскільки воно спрямоване на вирішення проблем

ефективності, безпеки та масштабованості в управлінні цифровими зображеннями, що є важливим компонентом в сучасному цифровому середовищі.

Отримані результати дослідження можуть бути використані при бізнес плануванні для проєктів, які використовують зображення у своїх робочіх процесах.

Результати даної кваліфікаційної роботи було представлено на науковій конференції 8th International Conference on Computational Linguistics and Intelligent Systems (Scopus), April 20–21, 2024, Lviv, Ukraine [1].

1 ОПИС ПРОБЛЕМНОЇ ГАЛУЗІ

1.1 Аналіз предметної області

Для аналізу предметної області необхідно розглянути стан сучасних цифрових технологій та їхній вплив на збереження та управління цифровими зображеннями. Дослідження даної області включає в себе аналіз технічних можливостей, проблем безпеки, аспектів масштабованості та вивчення практичних застосувань гібридних систем у цілях оптимізації управління цифровими даними. Нижче приведене більш детальне пояснення кожного з названих пунктів:

Зростання обсягів цифрових зображень: сучасний світ переживає вибуховий ріст цифрових зображень у різних сферах – від медицини та науки до мистецтва та бізнесу. Це призводить до потреби ефективного та безпечного зберігання величезних обсягів даних.

Проблеми безпеки і конфіденційності: цифрові зображення стають об'єктом загроз кібербезпеки [2], які можуть включати несанкціонований доступ, порушення конфіденційності та можливість маніпуляції даними. Забезпечення високого рівня безпеки та захисту цілісності даних – критичні завдання [3];

Важливість ефективного управління даними: управління метаданими, пошук, індексація та доступ до цифрових зображень є ключовими факторами для забезпечення доступності та ефективного використання цих даних.

Масштабованість систем зберігання: збільшення обсягів даних вимагає систем, здатних працювати з великими обсягами інформації, забезпечуючи швидкість та продуктивність.

Інноваційність гібридних систем зберігання: поєднання блокчейн-технологій з традиційними базами даних у гібридних системах відкриває нові можливості для збереження та управління даними, забезпечуючи безпеку, недоступність для втручання та відновлення даних.

Вплив на різні галузі: фективне управління цифровими зображеннями має велике значення для різних сфер, включаючи медицину, науку, мистецтво,

маркетинг, аналітику даних та інші галузі, де важлива обробка та збереження великих обсягів інформації.

Детальний аналіз предметної області показує, що дослідження технологій зберігання зображень у гібридних системах з блокчейн та традиційними базами даних є ключовим для вирішення низки викликів, пов'язаних з безпекою, ефективністю та масштабованістю управління цифровими даними.

1.2 Постановка задачі

Аналіз предметної області включає визначення ключових аспектів технологій зберігання зображень у гібридних системах, які поєднують в собі блокчейн [3] та традиційні бази даних. Ця область вивчається з метою вирішення актуальних завдань управління цифровими даними, які стають складнішими через стрімке збільшення обсягів інформації та загрози кібербезпеки. Постановка завдань для аналізу цієї області передбачає оцінку ефективності, безпеки та практичності гібридних систем зберігання зображень, а також їхні можливості в зберіганні зображень. Детальне вивчення цих аспектів є ключовим для розуміння переваг та викликів, пов'язаних із впровадженням новаторських підходів у збереженні та управлінні цифровими зображеннями. Під час дослідження буде розглянуто наступні завдання:

- аналіз існуючих технологій: вивчення та порівняння різних методів зберігання зображень, включаючи традиційний підхід та рішення на основі блокчейн-технологій [4-5];
- оцінка систем безпеки: дослідження рівня безпеки пропонованих систем зберігання зображень [6], визначення їхньої стійкості до потенційних загроз кібербезпеки;
- оцінка масштабованості та продуктивності: визначення можливостей системи працювати з великими обсягами даних, не втрачаючи ефективності та продуктивності;

– управління та відновлення даних: дослідження можливостей системи забезпечувати ефективне управління даними та відновлення інформації [7] у випадку втрати або пошкодження.

2 ОПИС ПРИЙНЯТИХ ПРОЄКТНИХ РІШЕНЬ

2.1 Аналіз конкуруючих систем

Для даного дослідження було прийнято рішення обрати 3 блокчейн-сховища та 2 традиційних, аби оцінити переваги і недоліки блокчейн [8] і традиційних систем одне перед одним. Вибрані сховища:

- BitTorrent (BTFS) – протокол для обміну та розповсюдження файлів через Інтернет, який використовує децентралізовану мережу для завантаження та передавання частин файлів між користувачами;
- Sia – децентралізована платформа зберігання даних, яка використовує технологію блокчейну для створення розподіленої мережі зберігання;
- Storj – децентралізована платформа зберігання даних, яка використовує технологію блокчейну та криптографічні протоколи для забезпечення безпеки та приватності даних;
- MySQL – це відкрита реляційна система управління базами даних, яка використовує мову структурованого запиту SQL для управління даними;
- PostgreSQL – система управління об'єктно-реляційними базами даних з відкритим вихідним кодом, відома своєю надійністю, розширеними можливостями та відповідністю стандартам SQL.

2.2 Визначення ключових критеріїв

Складання плану експерименту варто почати з визначення критеріїв, важливих при зберіганні зображень.

а) ціна – відображає вартість використання конкретної системи зберігання даних. Оцінка вираховується за вартістю використання 1TiB обсягу сховища на місяць:
   1) до 0.1$ – 10;
   2) від 0.1$ до 1$ – 7;
   3) від 1$ до 10$ – 5;
   4) від 10$ до 100$ – 3;
   5) від 100$ - 1;

б) швидкість – відображає швидкість завантаження і отримання доступу до зображень в системі зберігання. Оцінки швидкості вираховуватимуться відносно результатів експерименту: найбільше середнє з-поміж усіх наборів даних значення часу, витраченого на загрузку даних буде взяте за 1 бал, найменше – за 10. Після цього оцінка вираховуватиметься по середній швидкості з-поміж усіх наборів для кожної системи;

в) безпека – відображає рівень захищеності даних [9]. Оцінка вираховується за факторами, які впливають на захищеність даних, даючи по 1 балу за кожен фактор;

г) обсяги пам'яті – відображає обсяг доступної пам'яті або простору для зберігання даних у кожній системі. Оцінка базується на тому, чим лімітуються обсяги сховища:

1) обсяги датацентру – 1;
2) кількість провайдерів сховища – 2;

д) масштабованість – відображає вплив збільшення кількості даних на працездатність системи. Оцінка базується на середній швидкості зчитування даних. Рахуватиметься вона за формулою 1.1:

$$M_{avg} = \frac{1}{n}\sum_{i=1}^{n} x_i \, , P = 10 - \frac{M_{avg} - M_{min}}{M_{max} - M_{min}} * 10 \quad (1.1)$$

де M_{avg} – середній час зчитування протягом усіх експериментів на різних типах даних для одної системи;

n – кількість експериментів для конкретної системи зберігання даних;

i – порядковий номер експерименту;

x_i – результат певного експерименту;

P – кількість балів;

M_{min} – мінімальний середній час зчитування зображень серед усіх систем зберігання даних;

M_{max} – максимальний середній час зчитування зображень серед усіх систем зберігання даних.

Після визначення критеріїв для порівняння систем, можна проводити більш детальний аналіз кожної обраної системи зберігання даних і визначати значення параметрів, які можна отримати до експерименту. Можна оцінити такі параметри, як ціна, безпека та обсяг пам’яті. Швидкість і масштабованість визначатиметься протягом експерименту.

a) BitTorrent:
   1) ціна зберігання одного TiB в мережі BTFS протягом місяця коштує 3840 BTT (токен мережі) [10]. На момент дослідження ціна одного токена дорівнює 0.00000125$ [11], отже місячна плата дорівнює 0.00576. В такому разі сховище отримує 10 балів;
   2) безпека – система отримує 9 балів, адже дані в цій системі захищені 9 різними заходами безпеки:
      - шифрування – перед відсиланням на зберігання дані шифруються методом AES-256, забезпечуючи те, що тільки користувач з ключами шифрування матиме змогу отримати доступ до файлів;
      - децентралізація – файли розділені між різними провайдерами сховища, що значно знижує імовірність повної втрати даних;
      - копіювання – усі частини, на які поділені дані, копіюються і передаються на зберігання різним провайдерам;
      - інтеграція з блокчейном – історія транзакцій завжди записується і не може бути змінена. Усі операції керуються розумними контрактами, що передбачає, нагляд за умовами зберігання даних, обробку платежів і інших транзакцій;
      - управління доступом – користувач може налаштовувати умови доступу до своїх файлів для інших користувачів, використовувати публічні ключі для поширення файлів і приватні для розшифровки і отримання доступу;
      - протоколи PoR та PoRep – протокол PoR забезпечує систематичні перевірки провайдера на наявність у нього частин даних, які він повинен зберігати, а PoRep – відповідність кількості копій цих

частин даних відповідно до умов зберігання;

- механізм заохочення – використання токена мережі стимулює користувачів не вдаватися до підозрілої активності, адже це може негативно впливати на ціну токена, що робитиме подальше зберігання файлів менш вигідним;
- цілісність вмісту – наявність хеша для кожної частини даних і збереження його в блокчейні у вигляді дерев Меркла унеможливлює підробку цих даних;
- безпечні вузли – вузли BTFS використовують протокол безпеки, який блокує несанкціонований доступ і захищає від DDoS-атак і інших поширених загроз. Також ці вузли можуть бути ізольовані для мінімізації проблем з безпекою;

3) обсяг пам’яті – обсяг сховища регулюється кількістю вузлів, які надають свою пам’ять в оренду, отже система отримує 2 бали;

б) Sia:

1) ціна за зберігання 1 TiB на місяць, як вказано на сайті розробника [12], дорівнює приблизно 1-2$, тож система отримує 5 балів з цього параметру;
2) безпека – сховище також захищено дев’ятьма заходами з запровадження безпеки, тож отримує 9 балів:
   - шифрування Threefish-256;
   - децентралізація;
   - інтеграція з блокчейном;
   - управління доступом;
   - PoR та PoW протоколи – використовується протокол PoR і протокол PoW, який вимагає від провайдера надання доказів того, що він зберігає частини файлів згідно з умовами угоди;
   - механізм заохочення;
   - безпечні вузли;
   - резервування та відмовостійкість – сховище використовує

кодування зі стиранням для того, щоб забезпечити відновлюваність даних у випадку втрати частини даних або недоступності провайдера. Також вибір провайдерів відбувається за надійністю, продуктивністю та репутацією для максимізації безпеки даних та доступності;

3) обсяг пам’яті – розмір сховища залежить від кількості вузлів-провайдерів, тож система отримає 2 бали;

в) Storj:

1) ціна зберігання 1 GB протягом місяця дорівнює 0.004$ [13], що дорівнює 0.004x1099.51=4.398 за 1 TiB на місяць, тож система отримує 5 балів;
2) безпека – система захищена 10 заходами, тож отримує 10 балів:
   - шифрування AES-256;
   - децентралізація;
   - інтеграція з блокчейном;
   - управління доступом – окрім токенів для надання доступу можна використовувати API ключі;
   - протокол PoR та регулярні аудити – використовуються PoR та проводяться регулярні аудити для підтвердження цілісності і доступності даних і перевірки, чи поведінка вузла не є підозрілою;
   - механізм заохочення – користувачі, які не дотримуються своєї частини угоди можуть отримувати штрафи у вигляді втрати токенів та репутації;
   - цілісність даних;
   - безпечні вузли;
   - резервування та відмовостійкість – дані розподіляються по різним географічним регіонам для покращення стійкості до збоїв. У разі пошкодження вузла, мережа автоматично відновлює та знов відсилає постраждалі частини даних;

3) обсяг пам’яті – вмісткість сховища залежить від обсягів пам’яті провайдерів, тож система отримує 2 бали;

г) MySQL розглядатиметься в поєднанні з хостингом від Digital Ocean:

1) ціна зберігання 1 GiB даних коштує 0.1$+15$ за сам вузол на місяць [14–15], тобто зберігання 1 TiB коштує 117.4$ на місяць, тож система отримує 1 бал;
2) безпека – система захищена 7 мірами захисту, отже отримує 7 балів:
   - авторизація і автентифікація – MySQL використовує систему паролів для захисту від несанкціонованого доступу до БД;
   - шифрування SSL/TLS;
   - управління доступом – MySQL дозволяє давати доступ до даних за IP адресою;
   - SSH-ключі – Digital Ocean дозволяє авторизуватися завдяки SSH-ключу замість пароля;
   - приватні мережі – Digital Ocean дає доступ до своїх приватних мереж для більш безпечного з’єднання;
   - автозбереження та снапшоти – Digital Ocean надає можливість увімкнути автозбереження, яке періодично створюватиме резервну копію даних і робитиме знімки (снапшоти) перед внесенням великих правок для забезпечення швидкого відновлення даних;
   - моніторинг та сповіщення – Digital Ocean проводить моніторинг роботи дроплетів для виявлення несправностей і у разі виявлення надсилає сповіщення;
3) обсяг пам’яті – пам’ять обмежена 16 TiB, тож система отримує 1 бал;

д) PostgreSQL також розглядатиметься в поєднанні з хостингом від Digital Ocean:

1) ціна зберігання даних коштує 0.1$ за 1 GiB+15$ за вузол на місяць [15–16], тож загалом місяць зберігання 1 TiB даних коштує 117.4$, отже система отримує 1 бал;
2) безпека – система має 8 мір захисту, тож отримує 8 балів:

- авторизація та автентифікація – поділення користувачів за ролями з різним рівнем доступу, автентифікація, посилена LDAP, PAM та GSSAPI/Kerberos;
- шифрування SSL/TLS;
- убезпечення мережі – контроль під’єднання користувачів до мережі та детальний контроль над методами автентифікації;
- цілісність даних – технологія WAL підтримує відновлюваність бази даних, а PITR дозволяє повернути дані до їхнього стану в певний момент часу;
- аудит – є можливість вести детальний лог усієї активності бази даних;
- моніторинг та сповіщення;
- резервне копіювання і відновлення – дозволяє власноруч робити резервні копії;
- управління доступами;

3) обсяг пам’яті обмежено 1TiB, тож система отримує 1 бал.

Таблиця 2.1 – Проміжні дані для порівняння систем збереження даних (таблиця виконана самостійно)

	Ціна	Швидкість	Безпечність	Обсяг пам’яті	Масштабованість
Bittorrent (BTFS)	10	TBD	9	2	TBD
Sia	5	TBD	9	2	TBD
Storj	5	TBD	10	2	TBD
MySQL	1	TBD	7	1	TBD
PostgreSQL	1	TBD	8	1	TBD

Наявність чітко визначених і описаних критеріїв для порівняння систем зберігання даних дає змогу обрати метод, яким конкурентні сховища порівнюватимуться.

2.3 Визначення методу аналізу

Після виділення параметрів, важливих для системи збереження даних, варто з’ясувати, яким методом ці дані порівнюватимуться для визначення оптимальної технології для зберігання зображень.

Виходячи з того, що було створено критерії з вимірюваними числовими параметрами, прийнято рішення використовувати лінійну адитивну згортку з нормуючими множниками. На це було декілька причин:

- простота та зрозумілість: лінійна адитивна згортка є досить простою для розуміння та використання, що робить її привабливим методом для аналізу порівнюваних систем зберігання даних. Вона базується на простих математичних операціях, що дозволяє швидко та легко проводити розрахунки;
- універсальність: метод можна застосовувати в різних областях: від фінансів та економіки до технічних наук чи соціології, тож він є доречним будь-коли, даючи змогу аналізувати й порівнювати показники, незалежно від специфіки дослідження;
- можливість нормалізації даних: використання нормуючих множників дозволяє враховувати вагомість кожного показника у загальному аналізі. Це допомагає уникнути переваги одних показників над іншими та забезпечує об'єктивність оцінки;
- можливість врахування різноманітності показників: лінійна адитивна згортка дозволяє об'єднувати різні типи показників у єдину систему оцінювання, що робить її корисною для врахування багатофакторності в дослідженні.

Після з’ясування методу аналізу, який буде використовуватися для порівняння конкуруючих систем зберігання даних, необхідно обрати технічні засоби, які будуть застосовані для реалізації програмного рішення [17] для збору даних для проведення експерименту.

2.4 Обґрунтування програмного забезпечення для експерименту

Для експерименту було обрано мову програмування Python [18] у зв’язку з наявністю великої кількості позитивних сторін при роботі, зокрема і з системами зберігання даних:

- легкість використання;
- велика кількість бібліотек і фреймворків;
- інтеграція зі сховищами даних – абсолютна більшість систем зберігання даних підтримують використання Python для роботи з даними;
- велика спільнота – велика кількість користувачів використовує Python для розробки, що значно підвищує імовірність знайти відповіді на питання, які можуть виникнути в процесі розробки.

Для написання програм використовуватиметься Python версії 3.12.4 на комп’ютері з 64-розрядною Windows 10, шестиядерним процесором з частотою 3.4 ГГц, та 16 GB ОЗП.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У цьому розділі буде розглядатися програмна реалізація для збору даних, які будуть використані під час проведення експерименту. Окрім середи розробки PyCharm 2024.1.2 для мови програмування Python, якою реалізована програма, в ході роботи використовувалися також системи зберігання даних, в яких розміщувалися зображення. Загальну схему даних приведено нижче (див. рис. 3.1):

Рисунок 3.1 – Схема даних БД зображень (рисунок створено самостійно)

Для програмної реалізації було створено WorkFlow діаграму (див рис 3.2).

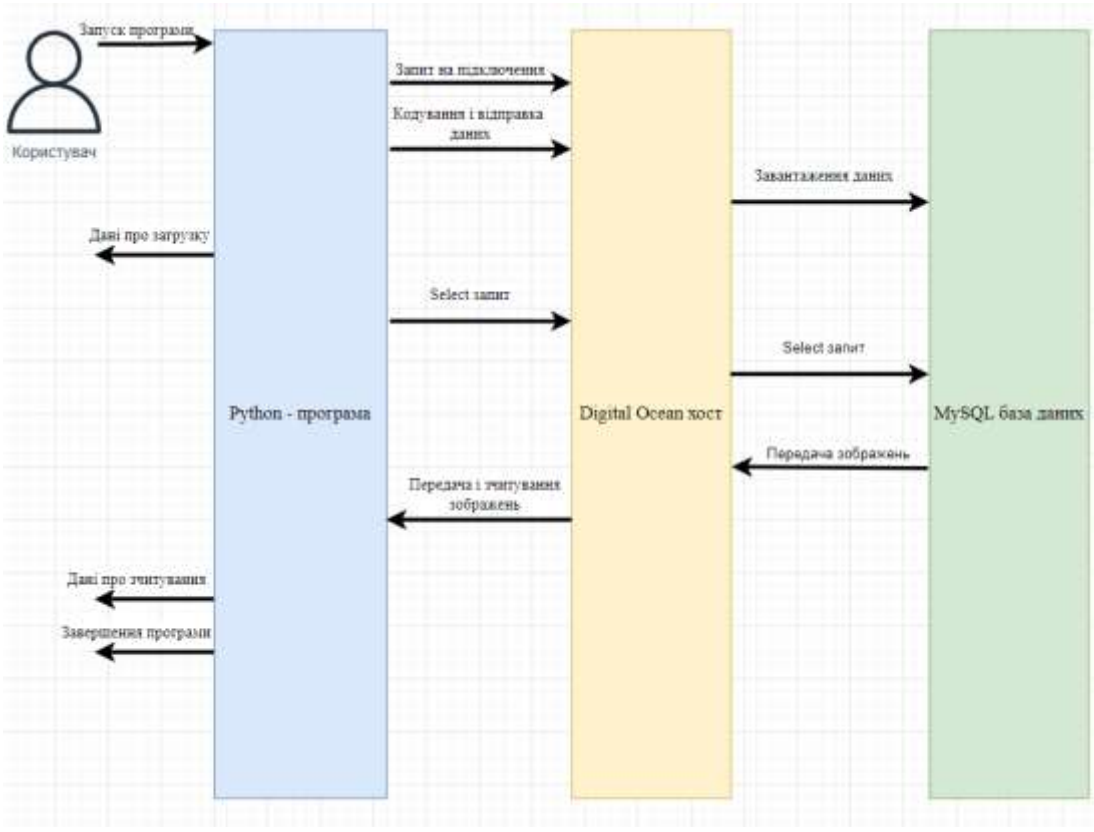

Рисунок 3.2 – WorkFlow діаграма програмної реалізації (рисунок створено самостійно)

Для створення програмної реалізації зі збору даних, зокрема швидкості загрузки зображень до сховища даних і їхнього подальшого зчитування задля порівняння методів зберігання даних в гібридних системах, було використано такі Python-бібліотеки:

- Os – модуль, який надає методи взаємодії з операційною системою;
- DotEnv – бібліотека, яка дозволяє створювати змінні середовища для уникання показування чутливої інформації;
- Time – модуль, який дозволяє відслідковувати час для подальшого його вимірювання;
- Pandas – бібліотека, яка дає змогу аналізувати дані та маніпулювати ними;
- PIL – бібліотека для роботи з зображеннями [19].

Розглянемо процес роботи програми на прикладі роботи з MySQL бази даних. Першим кроком було створення бази даних, після чого в базі даних було створено таблицю приведеним нижче запитом (див. рис. 3.3):

Рисунок 3.3 – Запит для створення таблиці для збереження зображень в базі даних MySQL (рисунок створено самостійно)

Після цього ми налаштовуємо програму, підключаючи її до створеної раніше бази даних завдяки бібліотеці mysql.connector для MySQL (див. рис. 3.4):

```
load_dotenv()

db_config = {
    'user': os.getenv('DB_USER'),
    'password': os.getenv('DB_PASSWORD'),
    'host': os.getenv('DB_HOST'),
    'database': os.getenv('DB_DATABASE')
}

connection = mysql.connector.connect(**db_config)
cursor = connection.cursor()
```

Рисунок 3.4 – Підключення до бази даних (рисунок створено самостійно)

Для збору даних було прийнято рішення розбити програму на 2 функції. В тілі першої («process_images») відбувається обробка процесу завантаження зображень у сховище і замірювання часу, який на це витрачається, з подальшим збереженням цих даних у вигляді csv-таблиці і виведенням у консоль середнього часу загрузки. Функція приймає змінну «folder_path», яка є шляхом до локально розташованої папки з зображеннями форматів .jpg та .png, змінну «dataset_name», яка використовується для надання імен таблицям з даними та виводу в консоль, змінну «db_connection» – об’єкт підключення до бази даних і змінну «connection_cursor», яка використовується для відправки запитів до БД (див. рис. 3.5).

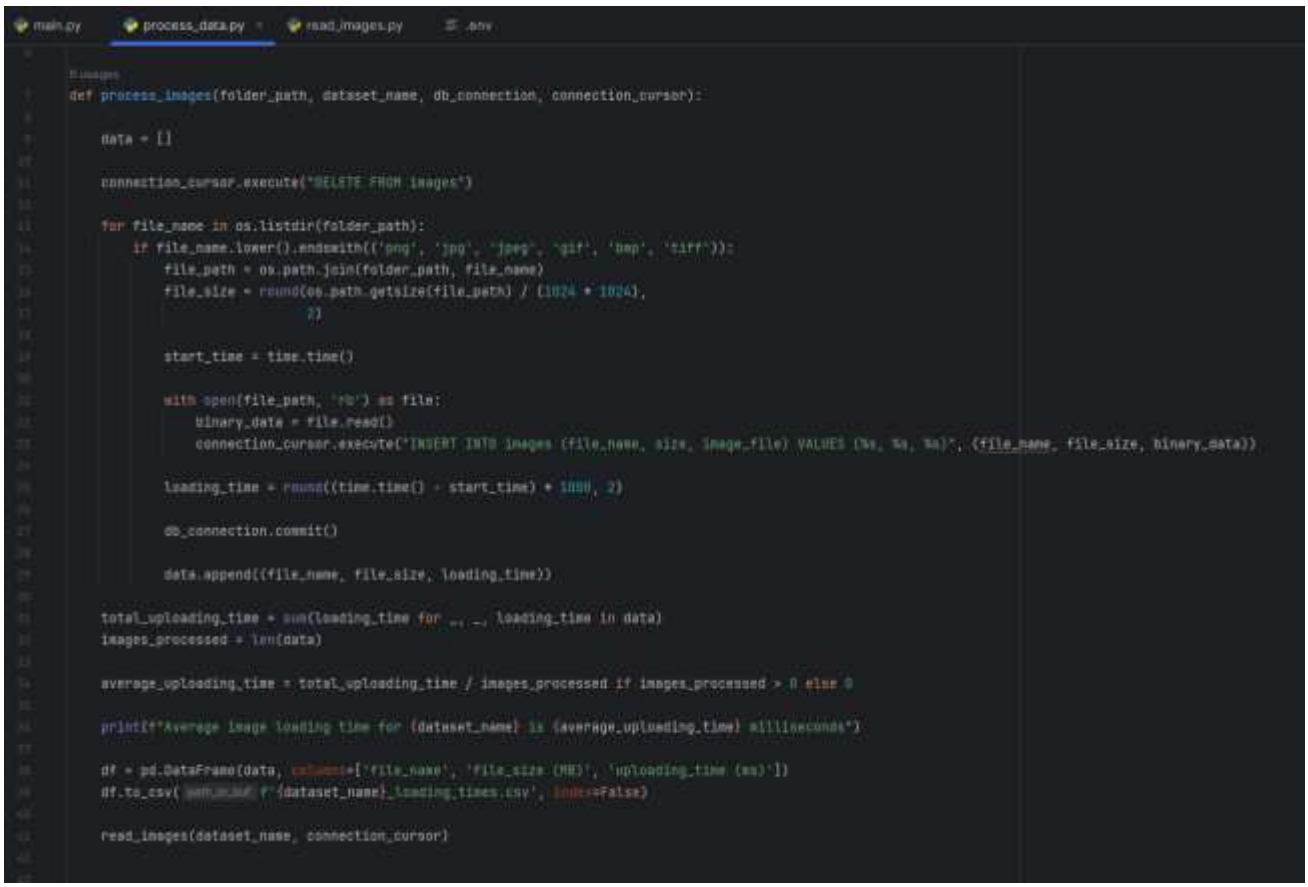

Рисунок 3.5 – Функція «process_images» (рисунок створено самостійно)

Друга ж функція («read_images») викликається всередині першої, після того, як вона завантажить дані в базу даних, і відповідає за збір даних про час, який йде на те, щоб отримати зображення з бази даних і отримати можливість його зчитати. Отримавши необхідну інформацію, функція виводить середній час на зчитування зображення і генерує таблицю зі значеннями часу зчитування кожного зображення. Функція приймає змінні «dataset_name» та «connection_cursor» (див. рис. 3.6).

main.py process_data.py read_images.py .env

```python
def read_images(dataset_name, connection_cursor):
    retrieval_start = time.time()
    connection_cursor.execute("SELECT file_name, size, image_file FROM images")
    retrieval_time = (time.time() - retrieval_start) * 1000
    data = connection_cursor.fetchall()

    processed_data = []

    total_images_size = float(sum(row[1] for row in data))

    for row in data:
        file_name, file_size, image_data = row

        fetch_start_time = time.time()

        image_data = io.BytesIO(image_data)

        file_retrieval_time = retrieval_time / total_images_size * float(file_size)

        fetch_time = (time.time() - fetch_start_time) * 1000 + file_retrieval_time

        start_time = time.time()

        with Image.open(image_data) as image:
            pass

        reading_time = round((time.time() - start_time) * 1000, 2)

        total_reading_time = float(fetch_time + reading_time)

        processed_data.append((file_name, file_size, total_reading_time))

    dataset_total_reading_time = sum(loading_time for _, _, loading_time in processed_data)
    images_read = len(data)

    average_reading_time = dataset_total_reading_time / images_read if images_read > 0 else 0

    print(f"Average image reading time for {dataset_name} is {average_reading_time} milliseconds")
    df = pd.DataFrame(processed_data, columns=['file_name', 'file_size', 'reading_time'])
    df.to_csv( path_or_buf: f'{dataset_name}_reading_times.csv', index=False)
    print(f"Data has been processed and saved to {dataset_name}_reading_times.csv")
```

Рисунок 3.6 – Функція «read_images» (рисунок створено самостійно)

Після завершення роботи програми від’єднуємось від бази даних, вимикаючи курсор та підключення (див. рис. 3.7).

```python
cursor.close()
connection.close()
```

Рисунок 3.7 – Відключення від бази даних (рисунок створено самостійно)

Після того, як програма повністю готова, можна приступити до збору даних на наборах зображень різного розміру. Для чистоти експерименту було прийнято

рішення сформувати директорію, наповнену зображеннями, і ще 4 директорії з таким самим набором зображень, продубльованим 3, 5, 7 і 10 разів. Для того, щоб отримати результати, викличемо функцію «process_data» на кожному з наборів зображень (див. рис. 3.8) і почекаємо поки програма завершиться.

main.py × process_data.py read_images.py .env

```
import os
import mysql.connector
from dotenv import load_dotenv
from process_data import process_images

load_dotenv()

db_config = {
    'user': os.getenv('DB_USER'),
    'password': os.getenv('DB_PASSWORD'),
    'host': os.getenv('DB_HOST'),
    'database': os.getenv('DB_DATABASE')
}

connection = mysql.connector.connect(**db_config)
cursor = connection.cursor()

default_images_pack_path = os.getenv('DEFAULT_IMAGES_PACK_PATH')
x3_images_pack_path = os.getenv("X3_IMAGES_PACK_PATH")
x5_images_pack_path = os.getenv('X5_IMAGES_PACK_PATH')
x7_images_pack_path = os.getenv('X7_IMAGES_PACK_PATH')
x10_images_pack_path = os.getenv("X10_IMAGES_PACK_PATH")

process_images(default_images_pack_path, dataset_name: "default_set", connection, cursor)
process_images(x3_images_pack_path, dataset_name: "x3_set", connection, cursor)
process_images(x5_images_pack_path, dataset_name: "x5_set", connection, cursor)
process_images(x7_images_pack_path, dataset_name: "x7_set", connection, cursor)
process_images(x10_images_pack_path, dataset_name: "x10_set", connection, cursor)

cursor.close()
connection.close()
```

Рисунок 3.8 – Фінальний варіант основного тіла програми (рисунок створено самостійно)

Після виконання програми у головній дерикторії проєкта з’являться 10 csv-файлів, які будуть заповнені даними зображень, зокрема про час завантаження і зчитування цих зображень (див. рис. 3.9) та в консолі з’явиться інформація про середні час загрузки і зчитування зображень на певному наборі зображень, розділені повідомленням про генерацію останнього з двох файлів для кожного набору зображень (див. рис. 3.10).

Рисунок 3.9 – Згенеровані програмою таблиці з даними (рисунок створено самостійно)

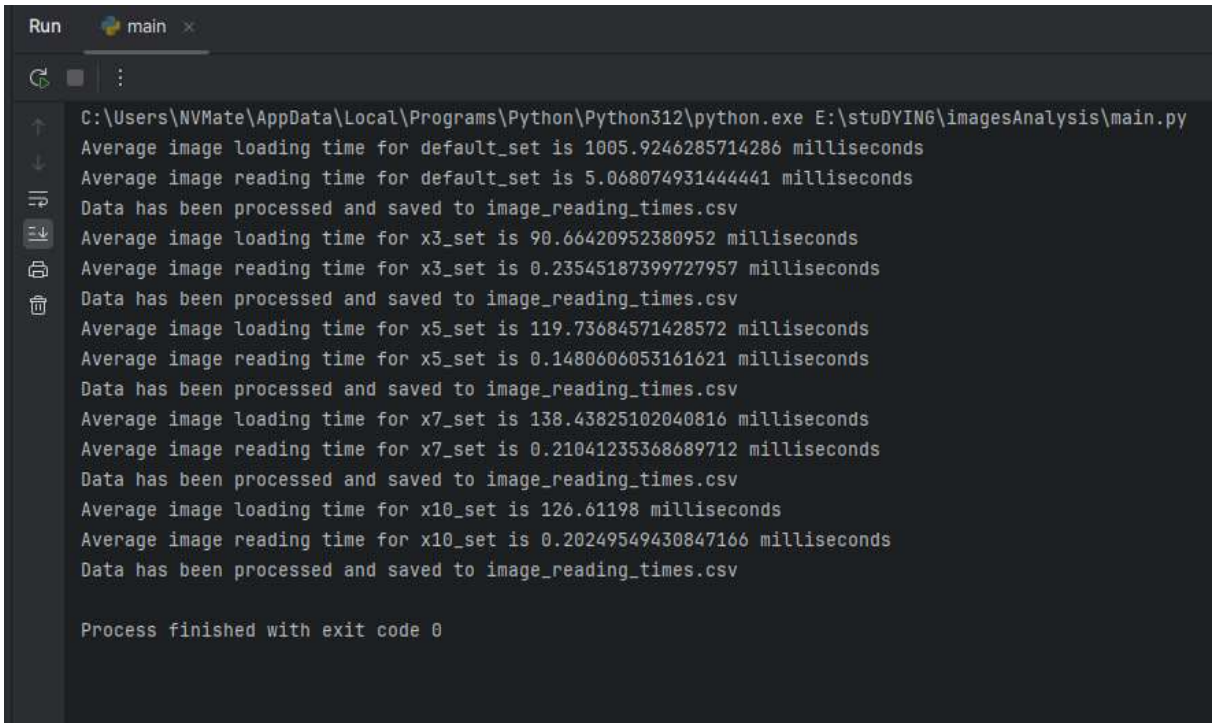

Рисунок 3.10 – Результат роботи програми для (рисунок створено самостійно)

Після того, як дані було успішно отримано, можна переходити до застосування програми на інших системах збереження даних. Результати виконання програми для них будуть приведені нижче (див. рис. 3.11 – 3.14).

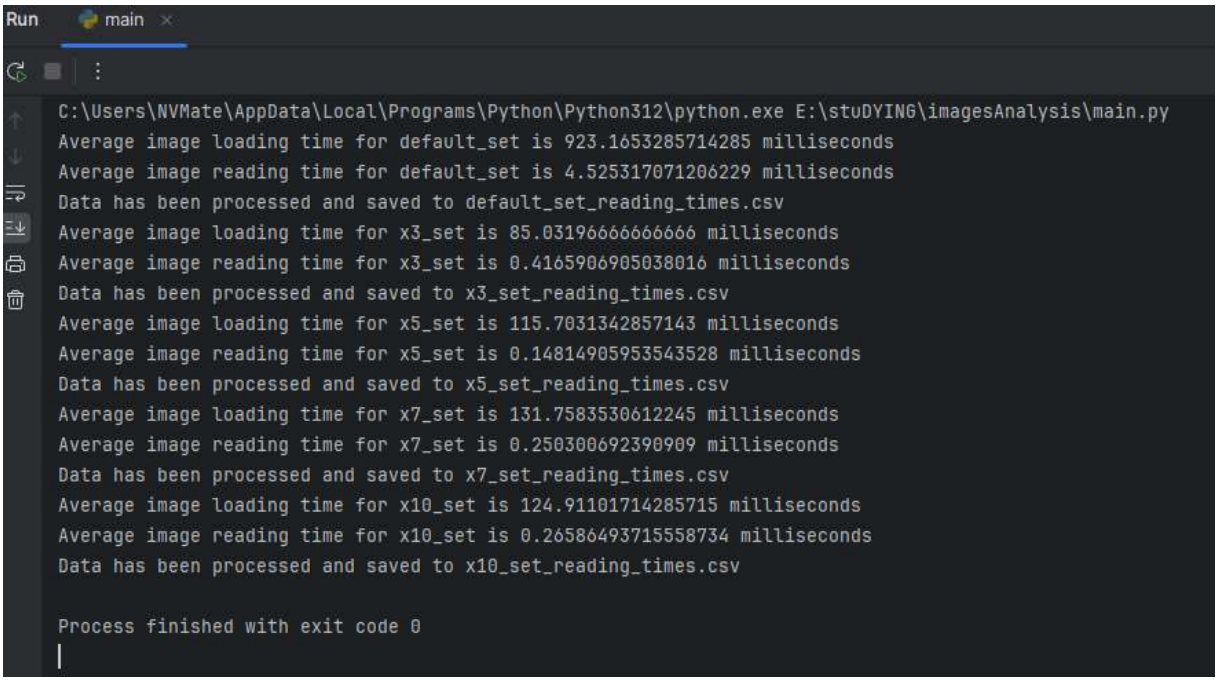

Рисунок 3.11 – Результат роботи програми для PostgreSQL (рисунок створено самостійно)

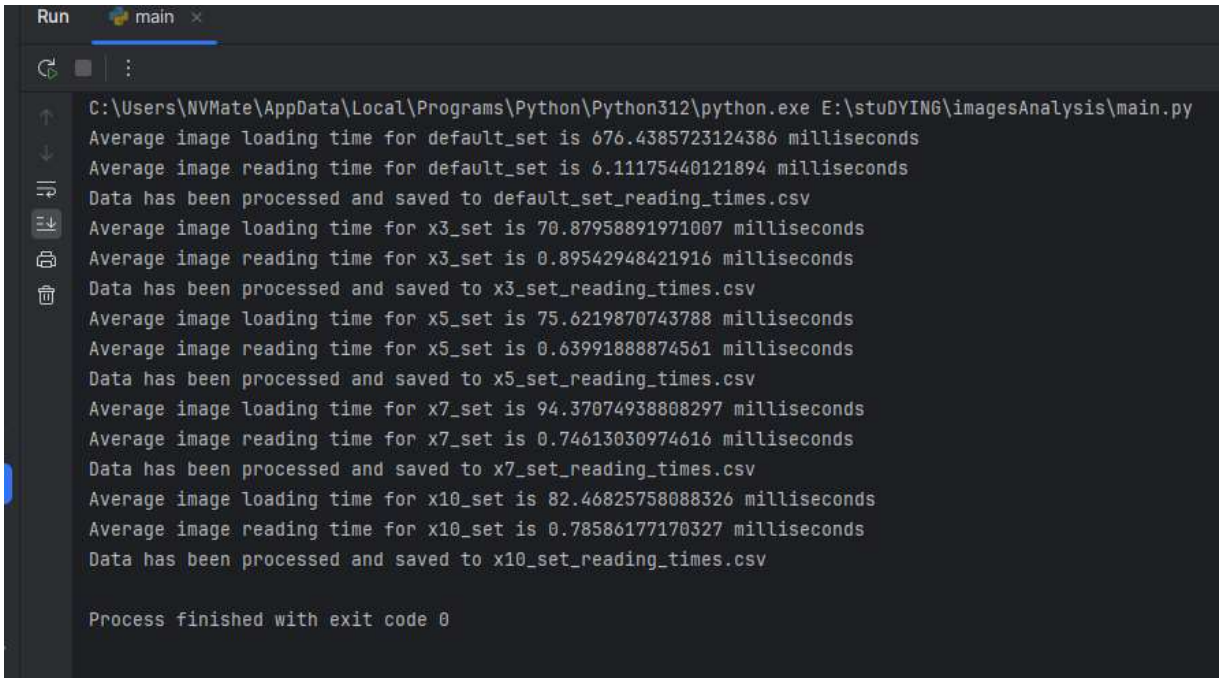

Рисунок 3.12 – Результат роботи програми для BitTorrent (рисунок створено самостійно)

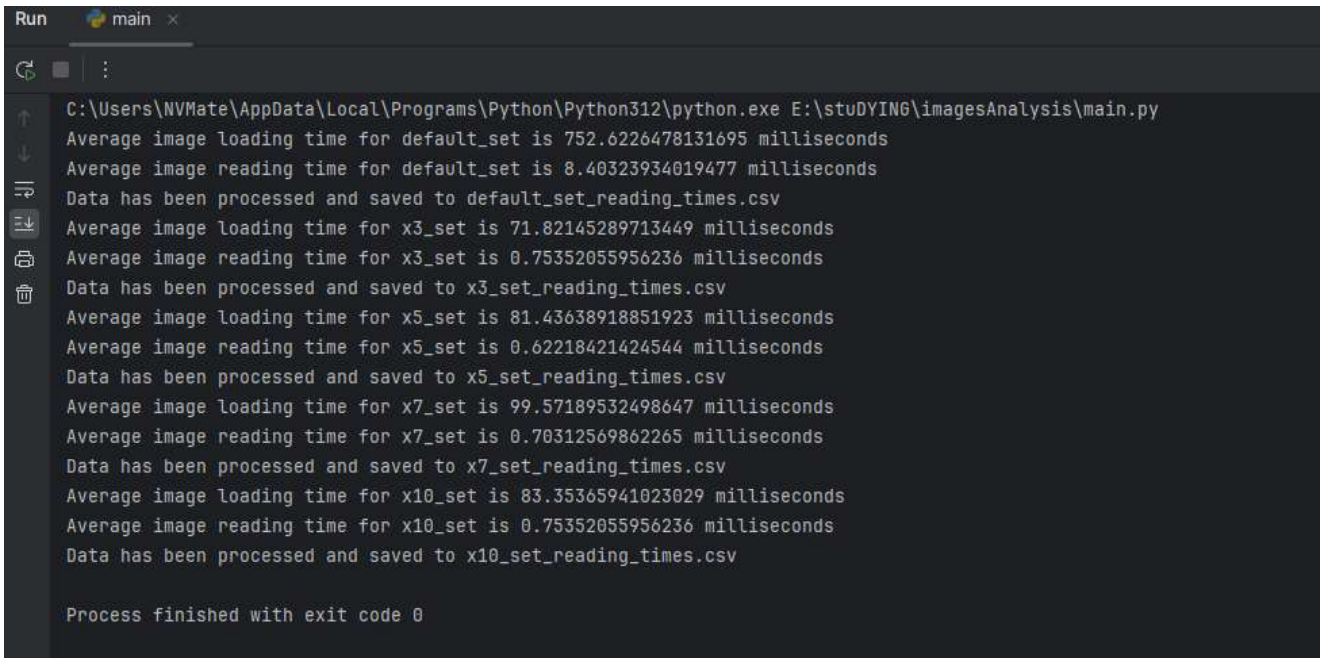

Рисунок 3.13 – Результат роботи програми для Sia (рисунок створено самостійно)

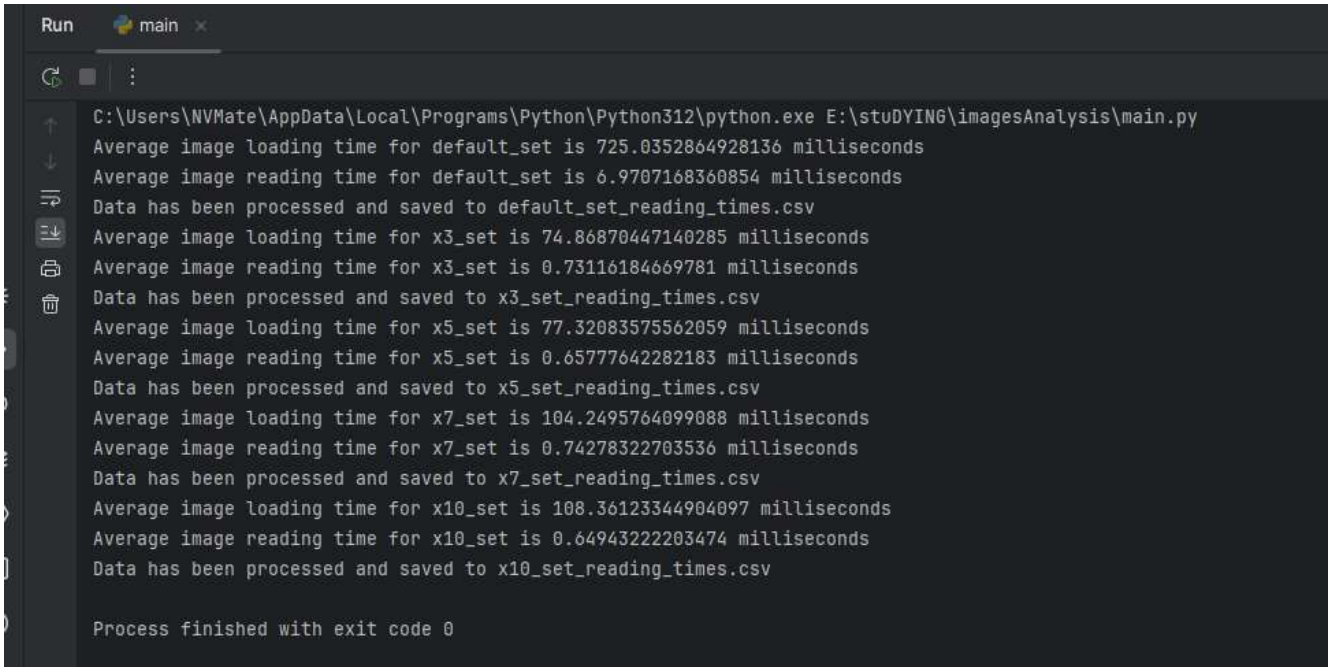

Рисунок 3.14 – Результат роботи програми для Storj (рисунок створено самостійно)

Як видно на зображеннях, програми вдало виконались для усіх систем зберігання даних і повернули результати обчислень, згенерувавши таблиці з даними і вирахувавши середні значення часу обробки зображень та часу зчитування зображень для кожного з наборів даних.

4 ОПИС ЕКСПЕРЕМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

В результаті проведених експериментів, коли дані з усіх систем зберігання даних зібрано, можна переходити до визначення результатів експерименту. Порівняльні графіки для середнього часу загрузки (див. рис. 4.1) та зчитування (див. рис. 4.2) зображення представлено нижче.

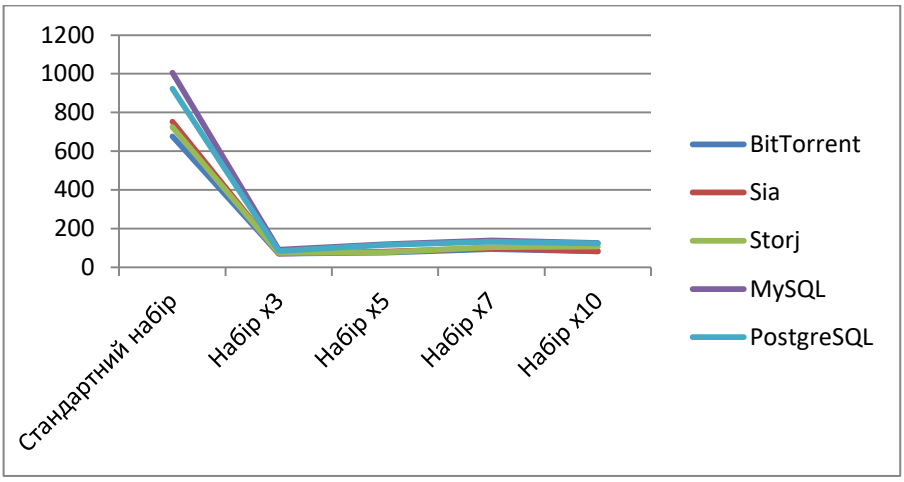

Рисунок 4.1 – Порівняльний графік часу загрузки зображень (рисунок створено самостійно)

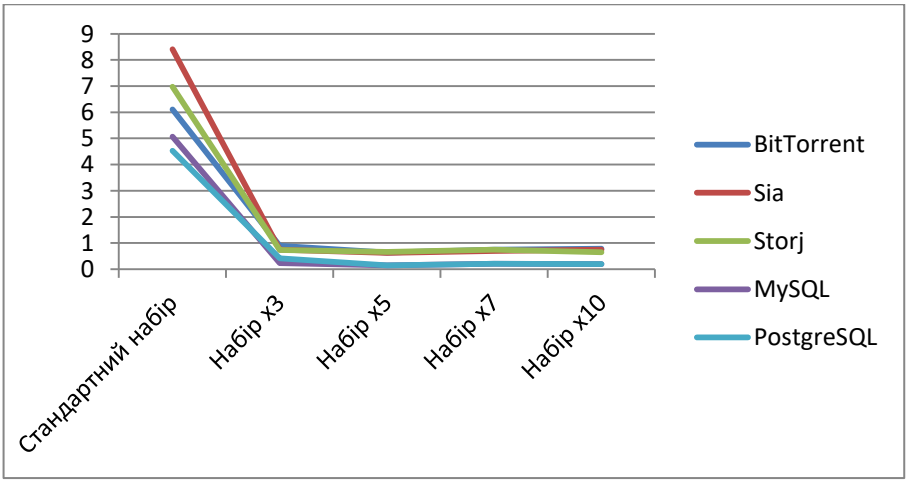

Рисунок 4.2 – Порівняльний графік часу зчитування зображень (рисунок створено самостійно)

Вирахуємо середні швидкості і згенеруємо оцінки:

– середня швидкість завантаження BitTorrent = $\frac{676.44+70.88+75.62+94.37+82.47}{5} = \frac{999.78}{5} = 199.95$;

– середня швидкість завантаження Sia = $\frac{752.62+71.82+81.44+99.58+83.35}{5} = \frac{1088.81}{5} = 217.76$;

– середня швидкість завантаження Storj = $\frac{725.03+74.65+77.32+104.25+108.36}{5} = \frac{1089.61}{5} = 217.92$;

– середня швидкість завантаження MySQL = $\frac{1005.92+90.66+119.74+138.44+126.61}{5} = \frac{1481.37}{5} = 296.27$;

– середня швидкість PostgreSQL = $\frac{923.16+85.03+115.70+131.76+126.91}{5} = \frac{1382.56}{5} = 276.51$.

Отже, найвища оцінка зі швидкості завантаження у BitTorrent = 199.95, а у MySQL найнижча = 296.27. Отже, BitTorrent отримує 10 балів, Sia отримує 7.35 балів, Storj отримує 7.32 балів, MySQL отримує 1 бал, і PostgreSQL отримує 1.84 балів.

Порахуємо середній час зчитування:

– для BitTorrent час зчитування = $\frac{6.11+0.89+0.64+0.75+0.79}{5} = \frac{9.18}{5} = 1.84$;

– для Sia час зчитування = $\frac{8.4+0.75+0.62+0.7+0.75}{5} = \frac{11.22}{5} = 2.24$;

– для Storj час зчитування = $\frac{6.97+0.73+0.66+0.74+0.65}{5} = \frac{9.75}{5} = 1.95$;

– для MySQL час зчитування = $\frac{5.07+0.23+0.15+0.21+0.2}{5} = \frac{5.86}{5} = 1.17$;

– для PostgreSQL час зчитування = $\frac{4.52+0.42+0.15+0.21+0.2}{5} = \frac{5.5}{5} = 1.1$.

Тоді, значення масштабованості виходять такі:

– BitTorrent – оцінка 3.16;

– Sia – оцінка 1;

– Storj – оцінка 2.29;

– MySQL – 8.45;

– PostgreSQL – 10.

Тепер, коли у нас є всі оцінки, можна скласти таблицю оцінок (див. табл.4.1).

Таблиця 4.1 – Фінальні дані для порівняння систем збереження даних (таблиця виконана самостійно)

	Ціна	Швидкість	Безпечність	Обсяг пам’яті	Масштабованість
Bittorrent (BTFS)	10	10	9	2	3.16
Sia	5	7.35	9	2	1
Storj	5	7.32	10	2	2.29
MySQL	1	1	7	1	8.45
PostgreSQL	1	1.84	8	1	10

Розраховуватимемо згортку за формулою 4.1:

$$Z *= max \sum_{j=1}^{n} \alpha_j \beta_j a_j, \quad (4.1)$$

де α_j – нормуючі множники,

β_j– вагові коефіцієнти, що відображають відносний внесок окремих критеріїв до загального критерію.

Спочатку розрахуємо нормуючі множники для кожного з параметрів:

- ціна – $\frac{1}{10+5+5+1+1} = 0.045;$
- швидкість – $\frac{1}{10+7.35+7.32+1+1.84} = 0.036;$
- безпечність – $\frac{1}{9+9+10+7+8} = 0.023;$
- обсяг пам’яті – $\frac{1}{2+2+2+1+1} = 0.125;$
- масштабованість – $\frac{1}{3.16+1+2.29+8.45+10} = 0.04.$

Після визначення вагових коефіцієнтів ми можемо розрахувати показники згортки для обраних систем:

- BitTorrent – $Z = 0.045 * 10 + 0.036 * 10 + 0.023 * 9 + 0.125 * 2 + 0.04 * 3.16 = 1.393$;
- Sia – $Z = 0.045 * 5 + 0.036 * 7.35 + 0.023 * 9 + 0.125 * 2 + 0.04 * 1 = 0.987$;
- Storj – $Z = 0.045 * 5 + 0.036 * 7.32 + 0.023 * 10 + 0.125 * 2 + 0.04 * 2.29 = 1.060$;
- MySQL – $Z = 0.045 * 1 + 0.036 * 1 + 0.023 * 7 + 0.125 * 1 + 0.04 * 8.45 = 0.705$;
- PostgreSQL – $Z = 0.045 * 1 + 0.036 * 1.84 + 0.023 * 8 + 0.125 * 1 + 0.04 * 10 = 0.820$.

Винесемо фінальний результат згортки в таблицю 4.2:

Таблиця 4.2 – Кінцеві результати згортки (таблиця виконана самостійно)

Система	**Результати згортки**
BitTorrent (BTFS)	1.393
Sia	0.987
Storj	1.060
MySQL	0.705
PostgreSQL	0.820

З результатів можна побачити, що найбільш оптимальні технології використовуються сховищем BitTorrent (BTFS), другим кращим результатом є Storj, третім – Sia, четвертим – MySQL, і останнім – PostgreSQL.

Як видно з результатів, сховища, які використовують блокчейн технології для зберігання даних, є кращим вибором, якщо визначені параметри є рівноважливими при виборі сховища для зображень [20]. Однак, велику частину різниці між блокчейн і традиційними сховищами створює різниця в ціні між ними, тож блокчейн сховища є більш підходящими для бюджетних проєктів, а традиційні сховища даних дозволяють отримати високу продуктивність при збільшенні вкладень, що краще підходить великим проєктам.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було проаналізовано предметну галузь, розглянуто декілька варіантів систем, підходящих для зберігання зображень – дві традиційних і три, які використовують блокчейн технології для зберігання даних. Для усіх систем було визначено основні критерії для порівняння їх одне з одним. Для кожної системи проведено детальний розгляд кожного параметра кожної окремо взятої системи зберігання даних.

У процесі роботи проведено експерименти зі збором параметрів, значення яких мали бути з’ясовані практичним методом. Для цього було створено Python-проєкт, який завантажував в системи зберігання даних зображення, вимірюючи час завантаження та створюючи таблиці-звіти із виміряними показниками. Також проєкт вимірював і генерував звіти з часом зчитування зображень.

В ході порівняння систем зберігання даних було з’ясовано, що для використання в малобюджетних проєктах варто звертатися до блокчейн-сховищ, бо за свої гроші вони дають гарні показники, але для великих проєктів краще підійдуть традиційні системи зберігання даних. Оптимальним варіантом при рівній важливості кожного з визначених критеріїв є використання системи зберігання BitTorrent (BTFS).

Результати цього дослідження можуть бути використані для прийняття рішення про те, яку систему зберігання даних варто обрати для проєкта на стадії бізнес планування, виходячи з потреб і пріоритетів компанії.

Кваліфікаційна робота пройшла апробацію на міжнародній конференції, що індексується SCOPUS:

– 8th International Conference on Computational Linguistics and Intelligent Systems (Scopus) April 12-13, 2024, Lviv, Ukraine (матеріали статті наведені у додатку Г).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Cherednichenko O., Kyrychenko I., Tereshchenko G., Myand D., Pylypenko S., Comparison of Blockchain–Based Data Storage Systems, COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine, vol. 3688, p. 10, https://ceur-ws.org/Vol-3688/paper10.pdf.

2. Кириченко І. В., Терещенко Г. Ю., Груздо І. В., Застосування симетричних алгоритмів в блокчейні / І. В. Кириченко, Г. Ю. Терещенко, І. В. Груздо // Біоніка інтелекту. – 2020. – № 1(94). – С. 71–75.

3. Kyrychenko I., Shyshlo O., Shanidze N., Minimizing Security Risks and Improving System Reliability in Blockchain Applications: a Testing Method Analysis, CEUR Workshop Proceedings, 2023, 3403, - C423–433, https://ceur-ws.org/Vol-3403/paper33.pdf.

4. A. Brickman, Y. Yigit, M. Carabaño, S. M. Hacking, Whole slide images as non-fungible tokens: A decentralized approach to secure, scalable data storage and access, Elsevier B.V., DOI: 10.1016/j.jpi.2023.100350.

5. S. Inam, S. Kanwal, R. Firdous, F. Hajjej, Blockchain based medical image encryption using Arnold’s cat map in a cloud environment, Nature Research, 2024, DOI: https://doi.org/10.1038/s41598-024-56364-z.

6. J. Noor, R. H. Ratul, R. Reaz, Secure Processing-aware Media Storage and Archival (SPMSA), Elsevier B.V., 2024, DOI: 10.1016/j.future.2024.05.010.

7. Wen, H., Yang, L., Bai, C., Hu, Y., He, D., Exploiting high-quality reconstruction image encryption strategy by optimized orthogonal compressive sensing, Scientific Reports, 14(1), 8805, 2024, DOI: 10.1038/s41598-024-59277-z.

8. J. Fior, INNBC DApp, a decentralized application to permanently store biomedical data on a modern, proof-of-stake (POS), blockchain such as BNB Smart Chain, BMC Medical Informatics and Decision Making, 24(1), 109, 2024, DOI: 10.1186/s12911-024-02498-z.

9. Noor, J., Ratul, R.H., Basher, M.S., Chellappan, S., Islam, A.B.M.A.A., Secure Processing-aware Media Storage and Archival (SPMSA), Future Generation Computer Systems, 159, pp. 290–306, 2024, DOI: 10.1016/j.future.2024.05.010.

10.BTSF FAQ. URL: https://docs.btfs.io/v1.0/docs/summary-of-btfs-faq (дата звернення: 05.05.2024).

11.BTT token price. URL: https://www.coingecko.com/en/coins/bittorrent (дата звернення: 05.05.2024).

12.Why projects choose Sia. URL: https://sia.tech/?ref=layers.cloud#why-projects-choose-sia (дата звернення: 05.05.2024).

13.Storj pricing. URL: https://docs.storj.io/dcs/pricing (дата звернення: 05.05.2024).

14.Digital ocean pricing for MySQL hosting. URL: https://docs.digitalocean.com/products/databases/mysql/details/pricing/ (дата звернення: 05.05.2024).

15.Digital ocean pricing for data volumes. URL: https://docs.digitalocean.com/products/volumes/details/pricing/ (дата звернення: 05.05.2024).

16. Digital ocean pricing for PostgreSQL hosting. URL: https://docs.digitalocean.com/products/databases/postgresql/details/pricing/ (дата звернення: 05.05.2024).

17. Tereshchenko Glib, Chetverykov Grigori, Overview of image storage models in Big Data conditions, Комп’ютерна математика в науці, інженерії та освіті (CMSEE-2020). – Матеріали V Всеукраїнської науково-технічної конференції – Полтава, 27 листопада 2020 р. – С. 18–21. УДК 00.89:004.043, http://kist.ntu.edu.ua/konferencii/27_konf_2020.pdf.

18. Reasons Why You Should Learn Python (2024). URL: https://www.geeksforgeeks.org/reasons-why-you-should-learn-python/ (дата звернення: 12.05.2024).

19. Image Processing With the Python Pillow Library URL: https://realpython.com/image-processing-with-the-python-pillow-library/ (дата звернення: 12.05.2024).

20. Liu, H., Wu, Z., Yin, M., Zhu, X., Lou, J., An improved deep hashing model for image retrieval with binary code similarities, Journal of Big Data, 11(1), 54, 2024, DOI: 10.1186/s40537-024-00919-4.