НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут телекомунікаційних систем

Кафедра телекомунікацій

До захисту допущено:

В.о.завідувача кафедри

_____ Сергій КРАВЧУК

«___»_____2026 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інженерія та програмування інфокомунікацій»

спеціальності 172 «Телекомунікації та радіотехніка»

на тему: «Проєктування файлового серверу з використанням платформи Nextcloud»

Виконав:
студент IV курсу, групи ТЗ-21
Гераськін Олександр Сергійович _____

Керівник:
Доцент кафедри ТК НН ІТС, к.т.н., доцент
Трубаров І. В. _____

Консультант:
Старший викладач кафедри ТК НН ІТС, к.ф.-м.н.
Руренко О. Г. _____

Рецензент:
Доцент кафедри ІТТ НН ІТС, к.т.н., доцент
Правило В. В. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2026 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут телекомунікаційних систем
Кафедра телекомунікацій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інженерія та програмування інфокомунікацій»

ЗАТВЕРДЖУЮ
В.о.завідувача кафедри
_____ Сергій КРАВЧУК
«___»_____2026 р.

ЗАВДАННЯ
на дипломну роботу студенту
Гераськіну Олександру Сергійовичу

1. Тема роботи «Проєктування файлового серверу з використанням платформи Nextcloud», керівник роботи Трубаров Ігор Володимирович, доцент, к.т.н., затверджені наказом по університету від «26» травня 2026 р. № 1912-с.

2. Термін подання студентом роботи 10 червня 2026 р.

3. Вихідні дані до роботи: платформа реалізації файлового сервера Nextcloud; програмне середовище розгортання LAMP-стек на базі операційної системи Ubuntu 24.04; локальний файловий сервер на власній інфраструктурі.

4. Зміст роботи: аналіз призначення, функцій та архітектури файлових серверів; дослідження платформ спільного доступу і синхронізації та програмного середовища розгортання сервера; реалізація локального файлового сервера на основі LAMP-стека з використанням платформи Nextcloud; адміністрування користувачів сервера.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

Слайд №1 Тема, мета та завдання роботи

Слайд №2 Файловий сервер: призначення та архітектура

Слайд №3 Вимоги та критерії проєктування файлового сервера

Слайд №4 Платформа Nextcloud, її особливості та переваги

Слайд №5 Програмне середовище LAMP, взаємодія з Nextcloud

Слайди №6, №7 Розгортання файлового сервера: інсталяція і налаштування LAMP і Nextcloud

Слайд №8 Мережеве налаштування сервера

Слайд №9 Адміністрування користувачів, розмежування доступу

Слайд №10 Перевірка сценаріїв використання сервера

Слайд №11 Висновки.

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ №1	Старший викладач Руренко О. Г.	16.01.2026	25.02.2026
Розділ №2	Старший викладач Руренко О. Г.	25.02.2026	30.03.2026
Розділ №3	Старший викладач Руренко О. Г.	30.03.2026	03.06.2026

7. Дата видачі завдання 06.11.2025р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної області, вибір та обґрунтування теми дипломної роботи.	06.11.2025 – 10.12.2025	Виконано
2	Опрацювання літературних джерел і добір матеріалів за темою.	10.12.2025 – 16.01.2026	Виконано
3	Дослідження архітектури, вимог та практик проєктування файлових серверів. Підготовка розділу №1.	16.01.2026 – 25.02.2026	Виконано
4	Аналіз платформи Nextcloud і засобів розгортання локального файлового сервера. Підготовка розділу №2.	25.02.2026 – 30.03.2026	Виконано
5	Інсталяція та налаштування програмного середовища.	30.03.2026 – 17.04.2026	Виконано
6	Розгортання платформи Nextcloud і її налаштування.	17.04.2026 – 13.05.2026	Виконано
7	Адміністрування користувачів та тестування роботи файлового сервера. Підготовка розділу №3.	13.05.2026 – 03.06.2026	Виконано
8	Формулювання висновків.	03.06.2025 – 05.06.2026	Виконано
9	Підготовка пояснювальної записки,	05.06.2026 – 09.06.2026	Виконано

	презентації та доповіді на захист.		

Студент Олександр ГЕРАСЬКІН

Керівник Ігор ТРУБАРОВ

РЕФЕРАТ

Текстова частина бакалаврської дипломної роботи містить: 59 сторінок, 27 рисунків, 1 таблицю та 31 джерело.

У роботі розглядаються архітектура файлових серверів, моделі організації сховища та розгортання, платформа Nextcloud і стек LAMP, які застосовуються для проєктування та практичної реалізації локального файлового сервера із забезпеченням спільного доступу до даних, розмежування прав користувачів і керування версіями файлів.

Метою роботи є проєктування та реалізація локального файлового сервера на основі платформи Nextcloud зі стеком LAMP із забезпеченням спільного доступу до даних і розмежування прав користувачів.

Ключові слова: файловий сервер, Nextcloud, LAMP, права доступу.

ABSTRACT

The text part of the bachelor's thesis contains: 59 pages, 27 figures, 1 tables and 31 references.

The thesis examines the architecture of file servers, storage organization and deployment models, the Nextcloud platform and the LAMP stack, which are applied to the design and practical implementation of a local file server providing shared access to data, differentiation of user access rights, and file version management.

The aim of the work is to design and implement a local file server based on the Nextcloud platform with the LAMP stack, providing shared access to data and differentiation of user access rights.

Keywords: file server, Nextcloud, LAMP, access rights.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

SMB	Server Message Block
NFS	Network File System
RAID	Redundant Array of Independent Disks
WebDAV	Web Distributed Authoring and Versioning
HTTP/HTTPS	HyperText Transfer Protocol / HyperText Transfer Protocol Secure
PHP	PHP: Hypertext Preprocessor
LDAP	Lightweight Directory Access Protocol
IP	Internet Protocol
ПК	Персональний комп’ютер
ОС	Операційна система
ПЗ	Програмне забезпечення
СКБД	Система керування базами даних

ВСТУП

В умовах стрімкого зростання обсягів цифрових даних та поширення командної форми роботи, питання надійного зберігання інформації й організації спільного доступу до неї набуває особливого значення для організацій будь-якого масштабу. Документи, мультимедіа та робочі матеріали дедалі частіше потребують єдиного впорядкованого сховища, доступного для кількох користувачів одночасно, із розмежуванням прав доступу та контролем версій файлів. Розв'язання цього завдання традиційно покладається на файловий сервер – централізований вузол комп'ютерної мережі, призначений для зберігання даних і надання до них спільного доступу.

Файловий сервер, на відміну від сервера застосунків, зазвичай не виконує обчислювальних завдань, а зосереджується на зберіганні файлів та керуванні доступом до них. Історично такі сервери реалізовували переважно апаратними засобами та пропрієтарним програмним забезпеченням, що потребувало значних витрат і прив'язувало організацію до конкретного постачальника. Поширення комерційних хмарних сервісів спростило доступ до даних, проте поставило під сумнів контроль власника над інформацією, оскільки дані зберігаються в інфраструктурі стороннього провайдера, а розширення можливостей потребує постійної підписки. Як відповідь на ці обмеження сформувався клас самостійно розгортуваних платформ синхронізації та спільного доступу до файлів з відкритим кодом, що поєднують зручність веб-орієнтованих рішень із повним контролем над даними на власній інфраструктурі. Однією з найпоширеніших платформ цього класу є Nextcloud.

Актуальність роботи. Зберігання даних на власній контрольованій інфраструктурі залишається затребуваним для широкого кола організацій, які з міркувань безпеки, приватності або суверенності даних не готові повністю покладатися на зовнішні хмарні сервіси. Для малих і середніх організацій, навчальних закладів та окремих робочих груп особливо важливою є можливість

розгорнути власне сховище без значних капітальних витрат і ліцензійних відрахувань.

Платформа Nextcloud разом зі стеком LAMP дозволяє побудувати таке рішення на основі вільного програмного забезпечення з відкритим кодом і доступного апаратного забезпечення, забезпечуючи водночас спільний доступ до даних, розмежування прав користувачів та керування версіями файлів. Попри поширеність платформи, її практичне впровадження потребує узгодження низки архітектурних і конфігураційних рішень, тому систематизований опис проєктування та розгортання локального файлового сервера на її основі є актуальним як із практичного, так і з навчально-методичного погляду.

Об'єктом дослідження є процес зберігання даних та організації спільного доступу до них у межах локальної мережі.

Предметом дослідження є методи та засоби проєктування й реалізації локального файлового сервера на основі платформи Nextcloud зі стеком LAMP.

Метою роботи є проєктування та реалізація локального файлового сервера на основі платформи Nextcloud зі стеком LAMP, що забезпечує спільний доступ до даних, розмежування прав користувачів і керування версіями файлів на власній інфраструктурі без залучення стороннього хмарного провайдера.

Для досягнення поставленої мети дослідження було поставлено такі основні завдання:

1) проаналізувати призначення, функції та архітектуру файлових серверів, моделі організації сховища й розгортання та сформулювати систему вимог і критеріїв до файлового сервера;
2) обґрунтувати вибір платформи Nextcloud і програмного середовища LAMP, оцінивши їх за визначеними критеріями проєктування файлового сервера;
3) спроєктувати та практично реалізувати локальний файловий сервер: розгорнути стек LAMP і платформу Nextcloud, налаштувати доступ

у локальній мережі, адміністрування користувачів та розмежування прав доступу;

4) перевірити працездатність створеного файлового сервера шляхом тестування основних сценаріїв його роботи.

Використана методика дослідження: у роботі використано аналіз і порівняння наявних рішень за визначеними критеріями, методи системного проєктування, а також експериментальне розгортання й тестування створеної системи. Теоретичні положення спираються на наукові джерела та офіційну технічну документацію використаних програмних засобів.

Практична цінність роботи:

Отримане рішення є робочим локальним файловим сервером, розгорнутим на доступному обладнанні з використанням вільного програмного забезпечення без ліцензійних витрат. Запропоновану послідовність проєктування, розгортання й налаштування може бути використано як практичний орієнтир для невеликих організацій та навчальних закладів, що потребують власного контрольованого сховища даних зі спільним доступом. Реалізоване рішення задає міцну основу для подальшого масштабування. Завдяки модульній архітектурі та екосистемі застосунків Nextcloud, базову функціональність файлового сервера можна розширювати засобами спільного редагування документів, керування календарями й контактами, комунікації та іншими компонентами, нарощуючи можливості системи відповідно до зростання потреб без зміни її ядра.

РОЗДІЛ 1
ФАЙЛОВІ СЕРВЕРИ: АРХІТЕКТУРА ТА ПРАКТИКИ РЕАЛІЗАЦІЇ

1.1. Призначення, функції та класифікація файлових серверів

Сервер у загальному розумінні є апаратно-програмним комплексом або окремою програмою, що надає послуги іншим програмам чи пристроям-клієнтам у межах комп'ютерної мережі. Серед різновидів серверів, файловий сервер посідає окреме місце, оскільки його основним завданням є централізоване зберігання даних та організація доступу до них.

Файловий сервер – це центральний вузол комп'ютерної мережі, який надає під'єднаним клієнтам спільний доступ до файлової системи або її частини, тобто слугує місцем зберігання файлів, доступним для робочих станцій мережі [1]. Це поняття охоплює як апаратну, так і програмну складову, необхідну для реалізації сервера, а авторизовані користувачі можуть відкривати, читати, змінювати, видаляти та завантажувати файли на нього [2]. Принциповою рисою файлового сервера є те, що, на відміну від сервера застосунків, він зазвичай не виконує обчислювальних завдань і не запускає програми від імені клієнтів, а лише забезпечує доступ до збережених даних [1].

Історично файлові сервери постали як відповідь на потребу ефективного спільного використання даних в організаціях. До їх появи дані зберігалися розрізнено на окремих комп'ютерах, що ускладнювало централізацію та керування ними. Поширення локальних мереж (Local Area Network) у 1970-1980-х роках уможливило винесення сховища в окремий спеціалізований вузол [3].

До основних функцій файлового сервера належать:

- централізоване зберігання та структуроване впорядкування файлів, що робить процеси пошуку та роботи з файлами послідовними та ефективними [4];
- спільний доступ і спільна діяльність, за яких користувачі однієї мережі обмінюються файлами без потреби переносити їх через зовнішні носії [5];
- керування доступом і розмежування прав. Серверне програмне забезпечення обробляє запити та застосовує політики контролю доступу [3];

– спрощення резервного копіювання, оновлень безпеки та перевірок цілісності даних [4].

З погляду будови файловий сервер поєднує кілька складових: апаратне забезпечення зі сховищем на основі жорстких дисків (Hard Disk Drive) або твердотільних накопичувачів (Solid-State Drive); операційну систему та серверне ПЗ; файлову систему, що організовує дані в каталоги; засоби мережевого підключення.

Файлові сервери, за архітектурою організації сховища, поділяють на три базові моделі [6]:

– DAS (Direct Attached Storage) – сховище, під’єднане безпосередньо до сервера без використання мережі. Фактично це внутрішні або зовнішні диски конкретної машини, наприклад, жорсткий диск всередині ПК доступний лише цій машині й не призначений для використання іншими вузлами мережі [7]. Така архітектура проста, економна й забезпечує низьку затримку, оскільки не потребує мережевого обміну [7], проте не дозволяє спільно використовувати чи гнучко масштабувати сховище.
– NAS (Network Attached Storage) – окремий спеціалізований пристрій із власним апаратним забезпеченням і полегшеною ОС, оптимізованою для зберігання та надання файлів. Як правило, це автономний пристрій, що містить власні диски, процесор і модулі пам’яті [8]. Він під’єднується до локальної мережі і працює на файловому рівні: клієнти звертаються до цілих файлів і каталогів за допомогою протоколів спільного доступу SMB та NFS, отримуючи єдину спільну точку доступу до сховища [6]. NAS порівняно простий у налаштуванні й керуванні, добре масштабується за обсягом і є зручним рішенням для спільної роботи з файлами.
– SAN (Storage Area Network) – окрема високошвидкісна мережа, що поєднує сервери з пристроями зберігання й виділена винятково для операцій зі сховищем. На відміну від NAS, SAN працює на блоковому рівні: він надає серверам не готові файли, а блоки даних у вигляді логічних томів (Logical Unit Number), які операційна система сервера сприймає як локальні диски, а обмін відбувається протоколами FC (Fibre Channel) або iSCSI (internet Small Computer Systems Interface) [5]. Завдяки цьому, SAN забезпечує високу продуктивність, низьку затримку й високу масштабованість, тому його застосовують для критично важливих корпоративних навантажень. Водночас, він складніший у проєктуванні та

обслуговуванні й коштує суттєво дорожче за NAS, оскільки потребує спеціалізованого обладнання та кваліфікованого супроводу [6].

До файлового сервера, незалежно від способу його реалізації, висувають низку базових вимог, що визначають його придатність до експлуатації:

- Безпека.
  Сервер має гарантувати конфіденційність, цілісність і доступність даних – три основні складові інформаційної безпеки [9]. На практиці це досягається автентифікацією користувачів, розмежуванням прав доступу до файлів і каталогів, шифруванням даних, а також контролем цілісності та захистом конфіденційної інформації від несанкціонованого доступу через надання дозволів [10].
- Надійність та відмовостійкість.
  Сучасні файлові сервери оснащують механізмами резервування та відмовостійкості, що підтримують доступність даних навіть у разі апаратного збою: дубльованими джерелами живлення, RAID-масивами з дублюванням даних на кількох дисках і кластеризацією з автоматичним перемиканням між серверами [10].
- Резервне копіювання та відновлення.
  Можливості резервного копіювання й відновлення даних є фундаментальними складовими сучасного файлового сервера та запорукою цілісності даних [10].
- Продуктивність і масштабованість.
  Сервер повинен витримувати пікові навантаження без погіршення обслуговування, а також мати змогу нарощувати ресурси відповідно до потреб. Різні типи серверів відрізняються за масштабованістю, продуктивністю та складністю керування [11].

1.2. Локальні, хмарні та гібридні рішення щодо реалізації файлового сервера

Спосіб розгортання файлового сервера є одним із ключових архітектурних рішень, оскільки визначає рівень контролю над даними, вартість володіння, продуктивність і безпеку системи. Залежно від того, де фізично розміщується сховище і хто здійснює керування ним, розрізняють три основні

моделі реалізації – локальну, хмарну та гібридну. Кожна з моделей має переваги, обмеження й оптимальні сценарії застосування.

Локальне рішення передбачає розгортання файлового сервера на власному обладнанні в межах інфраструктури користувача. Його головною перевагою є повний контроль над даними. Локальний сервер забезпечує безпосереднє керування безпекою та приватністю, вищі швидкості доступу, фізичний контроль над носіями й відсутність питань щодо суверенності даних [12,13]. Водночас, ця модель потребує самостійного обслуговування та ретельного планування ємності, щоб уникнути як надлишкового, так і недостатнього резервування ресурсів [13,14].

Хмарне рішення передбачає зберігання даних у зовнішнього постачальника послуг з доступом через інтернет. Така модель пропонує неперевершену масштабованість і гнучкість за мінімальних початкових витрат та глобальної доступності, що робить її зручною для резервного копіювання, аварійного відновлення й віддаленої співпраці [15]. Її недоліками є залежність від якості інтернет-з’єднання та можливі затримки, періодичні витрати, що зростають зі збільшенням обсягу даних і трафіку, а також відсутність власності на сховище, через що збої зв’язку безпосередньо впливають на продуктивність [15].

Гібридне рішення поєднує локальну інфраструктуру з хмарними сервісами. Воно дозволяє, наприклад, зберігати чутливі або критичні дані локально, водночас використовуючи хмару для менш конфіденційних навантажень чи додаткової ємності [13]. Таке поєднання підвищує гнучкість і відмовостійкість, за відмови одного середовища роботу підтримує інше, забезпечуючи безперервну діяльність [16]. Платою за ці переваги є підвищена складність розгортання та налаштування, потреба в ретельному плануванні та керуванні обома середовищами [13].

Таким чином, локальна, хмарна та гібридна моделі розгортання реалізують різні компроміси між контролем, вартістю, масштабованістю та зручністю доступу. Вибір конкретної моделі не є універсальним і визначається пріоритетами.

1.3. Архітектура та практики проєктування файлових серверів

Проєктування файлового сервера полягає в узгодженні низки архітектурних рішень, які разом визначають у який спосіб клієнти отримують доступ до даних, як ці дані зберігаються та захищаються та наскільки система здатна масштабуватися. Якщо класифікація і модель розгортання визначають загальний тип сервера, то його внутрішня архітектура та практики проєктування формують конкретну реалізацію придатну до експлуатації.

Багаторівнева архітектура та взаємодія компонентів. Незалежно від апаратної основи, файловий сервер будується за клієнт-серверною моделлю з логічним поділом на рівні, у якій обробку запитів відокремлено від фізичного зберігання даних. Така багаторівнева організація є усталеним архітектурним підходом. Більшість сучасних веб-застосунків ґрунтуються саме на клієнт-серверній архітектурі, де клієнт надсилає запит, а сервер відповідає на нього [17]. Поділ системи на рівні реалізує принцип розмежування відповідальності й забезпечує змінюваність та переносимість. Доки інтерфейс між рівнями лишається незмінним, окремий рівень можна замінити іншим, еквівалентним йому [17]. У типовій реалізації виокремлюють чотири рівні: рівень клієнтів, рівень мережевого доступу, рівень серверних служб, рівень зберігання даних. Взаємодія компонентів відбувається за такою загальною схемою: клієнт формує запит на доступ до файлу → запит передається до серверних служб, де користувача автентифікують і перевіряють його права → за наявності дозволу сервер звертається до рівня зберігання → рівень зберігання отримує дані і передає результат серверу → сервер формує відповідь і повертає клієнтові (рис. 1.1).

Рис. 1.1 – Багаторівнева архітектура файлового сервера

Протоколи доступу до файлів. Першим критерієм проєктування є вибір протоколу, за яким клієнти взаємодіють зі сховищем. Найпоширенішими є SMB, NFS, FTP/SFTP (File Transfer Protocol/SSH File Transfer Protocol) та WebDAV. Вибір визначається складом клієнтських пристроїв і вимогами до сумісності. SMB завдяки інтеграції з AD (Active Directory) зручний для Windows-середовищ; NFS орієнтований на Unix-системи; WebDAV на основі HTTP/HTTPS забезпечує найкращу кросплатформну сумісність, для клієнтів Windows, macOS та Linux [18]. При цьому, жоден протокол не є універсально найкращим. Вибір має відповідати конкретним вимогам застосунку, типу навантаження та середовищу розгортання [19].

Практики проєктування. Придатність сервера до експлуатації забезпечується дотриманням інженерних практик, що відповідають вимогам і слугують критеріями оцінювання конкретного рішення. Для надійності та цілісності даних застосовують надлишкові дискові масиви. Технологія RAID об’єднує кілька дисків і за рахунок резервування забезпечує надійність масиву, що значно перевищує надійність окремого диска [20]. Організація незалежних дисків у єдиний логічний диск є природним рішенням проблеми обмеженої надійності одиничних накопичувачів [21]. Розмежування прав доступу будують на основі облікових записів і груп користувачів зі списками контролю доступу та рольовими моделями. У моделі RBAC (Role-Based Access Control) права надаються не безпосередньо користувачам, а ролям, що відповідають службовим функціям, після чого користувачів призначають на ролі, чим спрощується адміністрування та підтримуються принципи найменших привілеїв [22,23]. Захист даних забезпечують шифруванням під час зберігання й передавання інформації та мережевою ізоляцією. Резервне копіювання – регулярним створенням копій на різних носіях із принципом надлишковості. Продуктивність і масштабованість – плануванням ємності, кешуванням і можливістю нарощування ресурсів. Окремим практичним критерієм для невеликих мереж є економічний чинник – вартість, модель ліцензування та

простота розгортання. Через економічний чинник перевагу часто надають програмним рішенням із відкритим кодом.

Висновки

У першому розділі розглянуто файловий сервер як централізований вузол мережі для зберігання даних і спільного доступу до них, визначено його основні функції та схарактеризовано три моделі організації сховища. Сформульовано базові вимоги до сервера: безпеку, надійність і відмовостійкість, резервне копіювання, продуктивність і масштабованість, а також розглянуто моделі розгортання, внутрішню багаторівневу архітектуру, протоколи доступу та інженерні практики проєктування. Сукупно ці вимоги та практики формують систему критеріїв для оцінювання придатності конкретної платформи, що визначає завдання наступного розділу.

РОЗДІЛ 2
ПЛАТФОРМА NEXTCLOUD ТА ЗАСОБИ РОЗГОРТАННЯ ЛОКАЛЬНОГО ФАЙЛОВОГО СЕРВЕРА

2.1. Платформа Nextcloud: загальний огляд та принцип роботи

Головною метою використання програмних рішень для організації файлового сервера є спільний доступ до даних, що зберігаються в одному місці. Практично для будь-якої організації, групи людей або для роботи з одними файлами з різних робочих станцій використання єдиного електронного сховища даних є надзвичайно корисним [24]. Сучасне програмне забезпечення для синхронізації та спільного користування файлами забезпечує доступ до файлів з різних пристроїв, можливе зберігання даних у хмарному середовищі та інші різноманітні можливості в залежності від конкретного ПЗ.

Серед таких рішень виокремлюють клас самостійно розгортуваних платформ синхронізації та спільного доступу до файлів (self-hosted file sync and share), до яких належать Nextcloud, ownCloud та Seafile. Це програмні продукти з відкритим кодом, що розгортаються на власній інфраструктурі організації й надають доступ до файлів як через веб-браузер, так і через клієнтські застосунки. На відміну від комерційних хмарних сервісів, вони зберігають дані на підконтрольній власникові інфраструктурі, забезпечуючи контроль над даними без залучення стороннього хмарного провайдера [25].

Nextcloud – це програмне забезпечення з відкритим вихідним кодом для синхронізації файлів і їх спільного використання. Платформа поєднує функціональність файлового сервера з перевагами веб-орієнтованих рішень, серверна частина системи встановлюється та адмініструється на інфраструктурі користувача, без залежності від централізованого хмарного провайдера. Система складається з трьох основних рівнів: серверного ПЗ, рівня зберігання даних та клієнтського доступу [5,26].

Серверний рівень відповідає за обробку запитів користувачів, автентифікацію, авторизацію та керування логікою доступу до ресурсів.

Реалізується у вигляді веб-додатку, який працює за допомогою стандартного стеку веб-технологій [5,26].

За розміщення та зберігання даних відповідає рівень зберігання даних, який базується на файловій системі операційної системи та використовує базу даних для зберігання службової інформації [5,26].

Клієнтський рівень представлений веб-інтерфейсом та застосунками Nextcloud, які взаємодіють із сервером. Виконує зв'язок між сервером і інтерфейсом користувача, забезпечує доступ до даних з різних пристроїв [5,26].

Ключовою особливістю архітектури є розділення логіки керування даними та фізичного їх зберігання (рис. 2.1). Nextcloud виконує роль проміжного програмного шару, який не є сховищем даних у прямому сенсі, а забезпечує взаємодію клієнта з сервером.

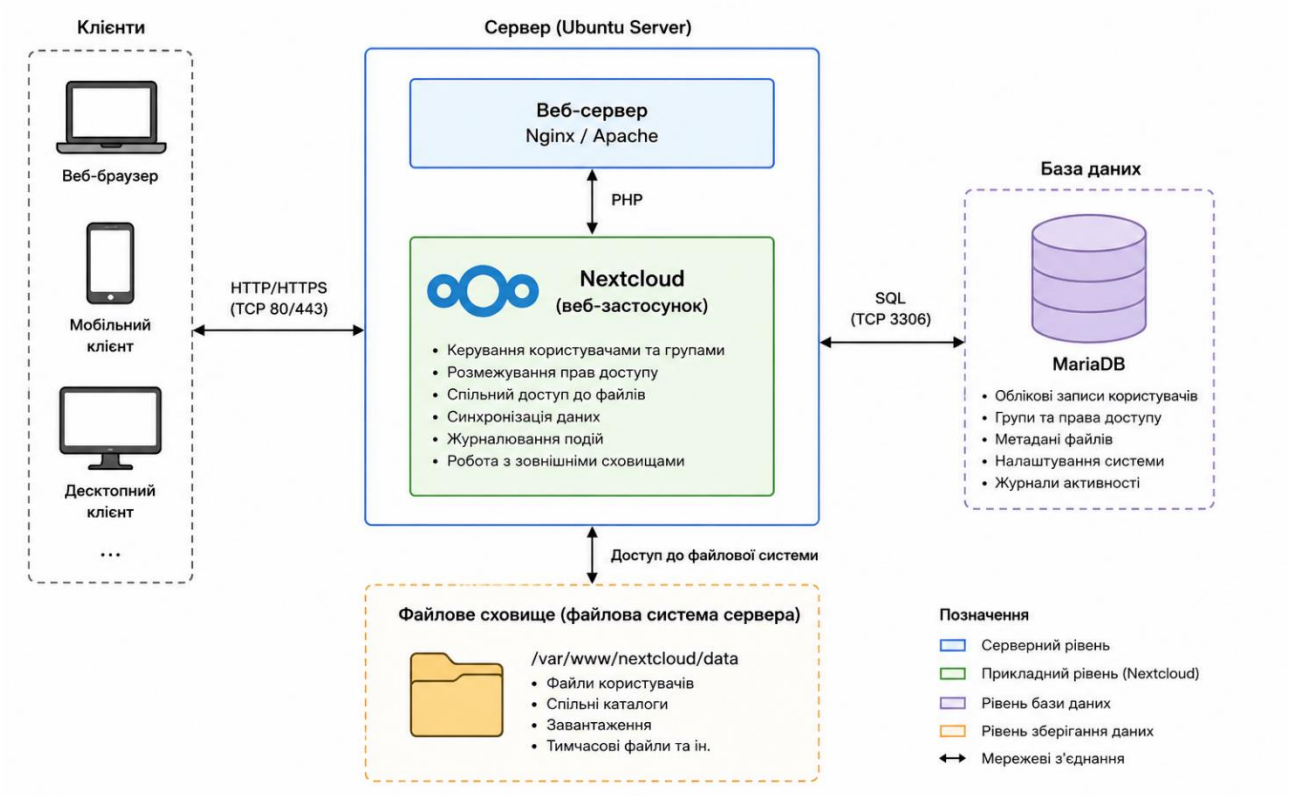

Рис. 2.1 – Архітектура файлового сервера з використанням платформи Nextcloud

Важливою характеристикою Nextcloud є модульність та розширюваність. Базову функціональність файлового сервера можна доповнювати окремими

застосунками. Платформу можна розширювати новою функціональністю у вигляді застосунку Nextcloud, що розробляється засобами веб-фреймворку мовою PHP [26]. Завдяки цьому до файлового сервера за потреби додають засоби спільного редагування документів, керування календарями й контактами, комунікації та інші компоненти, не змінюючи ядро системи [26]. Така модульна архітектура дозволяє адаптувати платформу під конкретні потреби та вільно масштабувати.

Платформа Nextcloud не є єдиним рішенням для організації файлового сервера. Існує низка конкурентів, які умовно поділяються на два класи. До першого належать комерційні хмарні сервіси, передусім Google Drive та Dropbox, що надаються за моделлю SaaS (Software as a Service). Вони не потребують власного сервера, зберігають дані в інфраструктурі постачальника, надають доступ за підпискою та мають закритий вихідний код. Другий клас утворюють самостійно розгортувані рішення з відкритим кодом, що працюють на власній інфраструктурі користувача. Насамперед ownCloud та Seafile, найближчим аналогом є ownCloud, від якого у 2016 році відділився Nextcloud. Seafile орієнтований передусім на швидку синхронізацію файлів і вирізняється ощадливістю до ресурсів, поступаючись Nextcloud за широтою засобів співпраці та інтеграцій. Зіставлення наведених вище платформ наведено в таблиці 2.1.

Таблиця 2.1

Порівняння платформ синхронізації та спільного доступу до файлів

Характеристика	Google Drive	Dropbox	Nextcloud	ownCloud	Seafile
Клас рішення	Комерційний SaaS	Комерційний SaaS	Self-hosted, відкрите	Self-hosted, відкрите	Self-hosted, відкрите
Розміщення даних	Хмара постачальника	Хмара постачальника	Власна інфраструктура	Власна інфраструктура	Власна інфраструктура
Вихідний код	Закритий	Закритий	Відкритий	Відкритий	Відкритий
Модель вартості	Підписка за розширення	Підписка за розширення	Безкоштовно	Безкоштовно + платні	Безкоштовно + платні
Доступ через браузер	Так	Так	Так	Так	Так
Нативна підтримка WebDAV	Ні	Ні	Так	Так	Так
Розмежування прав доступу	Так	Так	Так + LDAP/AD	Так + LDAP/AD	Так + LDAP/AD
Розширюваність (застосунки)	Закритий маркетплейс	Закритий маркетплейс	Висока (екосистема застосунків)	Помірна	Обмежена (фокус на синхронізації)

Комерційні сервіси виграють у простоті використання, оскільки не потребують власного сервера й адміністрування, проте, мають обмеження. Дані перебувають поза контролем власника, безкоштовний доступ обмежений і його розширення потребує підписки, код закритий, а стандартний протокол WebDAV нативно не підтримується. Самостійно розгортувані платформи усувають ці обмеження. Дані залишаються на підконтрольній власникові інфраструктурі [25], програмне забезпечення поширюється безкоштовно з відкритим кодом і нативно підтримує WebDAV та інтеграцію зі службами каталогів. Серед них Nextcloud вирізняється найширшою екосистемою застосунків і активною спільнотою [26].

2.2. Оцінювання платформи Nextcloud за критеріями проєктування файлового сервера

Архітектурна відповідність. Архітектура Nextcloud прямо відповідає моделі клієнт-сервер. Платформа реалізована як веб-застосунок, що поєднує серверний рівень обробки запитів, рівень зберігання на основі файлової системи та бази даних, клієнтський рівень доступу через браузер і застосунки. Ключовою рисою є те, що Nextcloud виступає проміжним програмним шаром і не є сховищем у прямому сенсі, а лише організовує взаємодію клієнта зі сховищем. Це узгоджується з принципом розмежування відповідальності між рівнями.

Безпека: автентифікація, розмежування прав доступу та шифрування. Nextcloud реалізує автентифікацію користувачів, шифрування даних та розмежування прав на основі облікових записів і груп. Розмежування прав будується на користувачах, групах і списках контролю доступу, де групи фактично виконують роль ролей у розумінні моделі RBAC, наближаючи систему до принципу найменших привілеїв [22,23]. Для централізованого керування обліковими записами платформа інтегрується зі службами каталогів LDAP та AD, що дозволяє узгоджувати права доступу з наявною

інфраструктурою організації. Захист даних забезпечується двома механізмами шифрування: серверним, що шифрує вміст файлів на боці сервера, та наскрізним, за якого сервер не отримує доступ до незашифрованих даних і ключів [25,26]. Слід зважати на обмеження, серверне шифрування не приховує імен файлів та структури каталогів і не захищає дані від скомпрометованого сервера чи зловмисного адміністратора. Тоді як наскрізне шифрування, забезпечує найвищий рівень конфіденційності, накладаючи експлуатаційні обмеження на роботу з файлами [26].

Надійність, відмовостійкість і резервне копіювання. Апаратна відмовостійкість забезпечується не самою платформою Nextcloud, а безпосередньо апаратним забезпеченням, операційною системою та системою керування базами даних. Nextcloud покладається на ці механізми й доповнює їх власними засобами цілісності даних. Версіонування файлів, кошик для відновлення вилучених об’єктів та можливість резервного копіювання каталогу даних і бази даних є основними засобами платформи [25].

Продуктивність і масштабованість. Модульна архітектура Nextcloud дозволяє масштабувати систему у різних аспектах. Продуктивність підвищують засобами кешування та винесенням службових даних у спеціалізовані сховища. Ємність підвищують підключенням зовнішніх сховищ. Інтеграція зі службами каталогів дає змогу обслуговувати велику кількість облікових записів без створення окремої бази користувачів [25]. Водночас фактична продуктивність істотно залежить від конфігурації середовища та апаратного забезпечення.

Протоколи доступу та сумісність. Nextcloud нативно підтримує протокол WebDAV на основі HTTP/HTTPS, який забезпечує найкращу кросплатформну сумісність для клієнтів Windows, macOS та Linux, а доступ до даних можливий як через веб-браузер так і через клієнтські застосунки. Додатково платформа підтримує підключення зовнішніх сховищ за протоколами спільного доступу, зокрема SMB та NFS, що дозволяє інтегрувати наявні мережеві ресурси в єдину точку доступу.

Економічність і простота розгортання. Платформа поширюється безкоштовно з відкритим вихідним кодом, тож її використання не потребує ліцензійних відрахувань чи передплати за кількість користувачів, на відміну від комерційних SaaS-сервісів. Оскільки сервер розгортається на власній інфраструктурі, відсутні й періодичні витрати, що зростають зі збільшенням обсягу даних і властиві хмарним сервісам, а вартість володіння зводиться переважно до наявного обладнання та його обслуговування [26]. Простота розгортання є помірною. Платформа потребує попереднього налаштування серверного середовища та адміністрування, проте її модульність дозволяє обмежитися лише потрібним набором компонентів, а докладна документація й активна спільнота знижують витрати на супровід. Сукупно це робить Nextcloud придатним для розгортання власними силами, без залучення стороннього провайдера.

2.3. Програмне середовище розгортання файлового сервера на основі стека LAMP

Платформа Nextcloud у своїй серверній частині є веб-застосунком, написаним мовою PHP [26], вона не здатна працювати самостійно. Для її функціонування потрібне програмне середовище, яке надає операційну систему, веб-сервер, систему керування базами даних та інтерпретатор PHP. Саме таку сукупність компонентів утворює стек LAMP.

LAMP-стек – це набір із програмних компонентів з відкритим кодом, що спільно утворюють середовище для роботи веб-сайтів і веб-застосунків. Назва є абревіатурою його складових: операційна система Linux, веб-сервер Apache, система керування базами даних MySQL або MariaDB, мова програмування PHP або Perl чи Python [27] (рис.2.2). Кожен компонент відповідає за окремий рівень роботи веб-застосунку, від базової операційної системи до обробки прикладної логіки. Їх поєднання забезпечує повний цикл приймання, обробки та повернення запитів користувача.

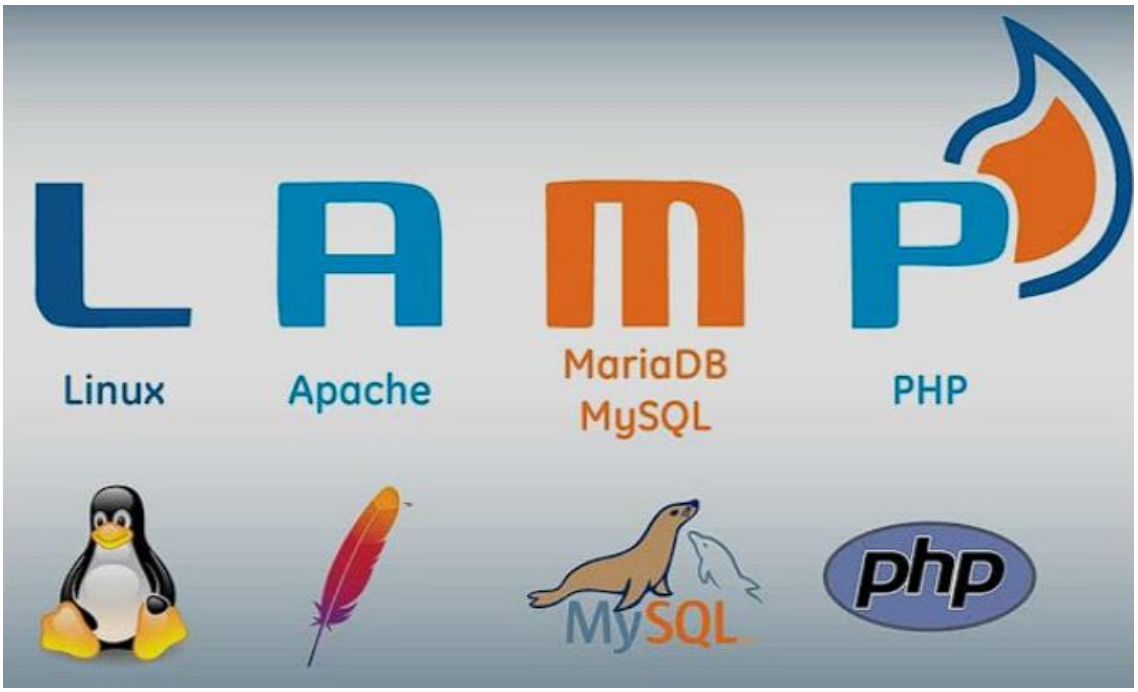

Рис. 2.2 – Компоненти LAMP-стека

Основою стека є операційна система Linux, яка є базою для решти компонентів. У межах роботи використано дистрибутив Ubuntu, який базується на ядрі Linux і широко застосовується для розгортання серверних служб та мережевих застосунків. Вибір Ubuntu зумовлений зручністю адміністрування, гнучким налаштуванням мережевих служб та наявністю в репозиторіях усіх програмних компонентів, необхідних для роботи стека [28]. Додатковим чинником стали попередній досвід роботи з цим дистрибутивом та його наявність на особистому комп’ютері.

Веб-сервер Apache відповідає за приймання й обробку HTTP-запитів від клієнтів та повернення їм відповідей. Він є одним із найпоширеніших веб-серверів завдяки стабільності, докладній документації та підтримці додаткових модулів, які розширюють його можливості. У складі стека, Apache виступає точкою входу, через яку користувач звертається до веб-застосунку з браузера [27,29].

Зберігання даних забезпечує система керування базами даних MariaDB – відгалуження MySQL з відкритим кодом, сумісне з ним за командами та

інтерфейсами. Вона зберігає дані застосунку в структурованому вигляді й надає доступ до них мовою запитів SQL (Structured Query Language). MariaDB поширена в Linux-середовищі, сумісна з PHP та вирізняється високою продуктивністю й відкритим програмним кодом [27,30].

Мова програмування PHP виконує прикладну логіку веб-застосунку на боці сервера. Відповідає за обробку запитів, взаємодію з базою даних та формування сторінки HTML (HyperText Markup Language). PHP є інтерпретованою мовою, орієнтованою на веб-розробку, а її функціональність розширюється підключенням окремих модулів, що додають підтримку додаткових форматів даних і операцій [31].

Робота стека полягає в узгодженій взаємодії його компонентів під час обробки запиту від браузера. Apache та MariaDB працюють під керуванням Linux і обмінюються даними за допомогою PHP. Послідовність обробки запиту виглядає наступним чином: веб-сервер Apache приймає запит від браузера → запит на статичний вміст Apache обслуговує самостійно, а запит на динамічний вміст передає інтерпретаторові PHP → PHP виконує необхідний скрипт, за потреби звертається до бази даних MariaDB та формує результат у форматі HTML → сформовану сторінку Apache повертає у браузер користувача [27].

Розгортання локального файлового сервера можливе й за допомогою інших рішень. Зокрема Windows Server, контейнеризованих середовищ на основі Docker або стеків MEAN (MongoDB Express.js Angular Node.js), WAMP (Windows Apache MySQL PHP) та MAMP (MacOS Apache MySQL PHP). Порівняно з ними, поєднання LAMP з Ubuntu забезпечує реалізацію файлового сервера з помірною складністю налаштування й адміністрування за відсутності ліцензійних витрат. Для локального використання в невеликих мережах такий вибір є оптимальним.

2.4. Взаємодія платформи Nextcloud з компонентами LAMP-стека

Платформа Nextcloud не керує компонентами стека безпосередньо, а взаємодіє з ними через усталені інтерфейси. З веб-сервером взаємодія відбувається за протоколом HTTP, з базою даних – мовою запитів SQL, а зі сховищем – через файлову систему операційної системи [26]. Ці точки взаємодії визначають роботу системи як цілого.

Передавання запиту від Apache до застосунку. Взаємодія з веб-сервером побудована за моделлю єдиної точки входу. Apache спрямовує всі запити до прикладної логіки до визначених точок входу Nextcloud – сценарію index.php для веб-інтерфейсу та remote.php для доступу за протоколом WebDAV і синхронізації. Доступ до файлів за протоколом WebDAV, застосовуючи власну базову автентифікацію, Nextcloud обслуговує самостійно [26,27]. Перенаправлення запитів до цих точок входу забезпечує механізм перезапису URL на боці Apache, таким чином вся платформа має вигляд єдиного ресурсу для зовнішнього клієнта.

Доступ до даних через СКБД. З базою даних Nextcloud взаємодіє як клієнт СКБД. Під час виконання, платформа встановлює з’єднання з MariaDB за параметрами, заданими в її конфігурації, і звертається до даних мовою SQL. На відміну від зберігання вмісту, у базі даних зберігаються облікові записи користувачів, метадані файлів, права спільного доступу та конфігурації застосунків [26]. Майже кожна операція над файлами супроводжується зверненням до бази даних, оскільки Nextcloud веде в ній окремий індекс файлів. Цей індекс слугує рівнем абстракції для зберігання метаданих, адже звернення до бази даних швидше за звернення безпосередньо до сховища [26]. Завдяки цьому платформа виконує операції над структурою каталогів і правами доступу засобами СКБД, не перечитуючи самі файли.

Звернення до файлової системи. Nextcloud зберігає і читає вміст файлів через файлову систему операційної системи, звертаючись до виділеного каталогу даних. Доступ до сховища реалізується через власний рівень абстракції, не напряму. Для кожного сховища підтримується окремий кеш-індекс метаданих, що дозволяє читати їх, не звертаючись щоразу до потенційно

повільного сховища [26]. Такий проміжний рівень уніфікує доступ і дає змогу підключати різні типи сховищ без зміни прикладної логіки платформи.

Розмежування метаданих і вмісту. Ключовим принципом взаємодії Nextcloud з базою даних і файловою системою є розподіл ролей між ними. У базі даних платформа веде індекс файлів з метаданими (іменами, шляхами, розмірами та іншими атрибутами). Він слугує рівнем абстракції для швидкого доступу до структури каталогів і прав без звернення до сховища. Сам вміст файлів зберігається у файловій системі у виділеному каталозі даних. За використання локального сховища, файли зберігаються зі своїми іменами та структурою каталогів, а база даних виконує роль швидкого індексу. Проте за під'єднання об'єктного сховища, вміст зберігається за унікальними ідентифікаторами, а імена та структура існують лише в базі даних [26]. У будь-якому разі, база даних і файлова система є взаємодоповнювальними. Без індексу метаданих платформа не здатна ефективно відтворити структуру збережених даних, а без файлової системи метадані не мають вмісту, на який вони посилаються.

Життєвий цикл запиту. Описані точки взаємодії об'єднує типовий сценарій звернення до файлу (рис. 2.3). Клієнт надсилає HTTP-запит, його приймає Apache та передає інтерпретаторові PHP до точки входу в Nextcloud. Платформа автентифікує користувача й перевіряє його права, звертаючись до MariaDB, за тим самим запитом до бази даних визначає розташування потрібного об'єкта. За наявності дозволу, Nextcloud зчитує або записує вміст файлу у файловій системі, після чого формує відповідь, яку Apache повертає клієнтові.

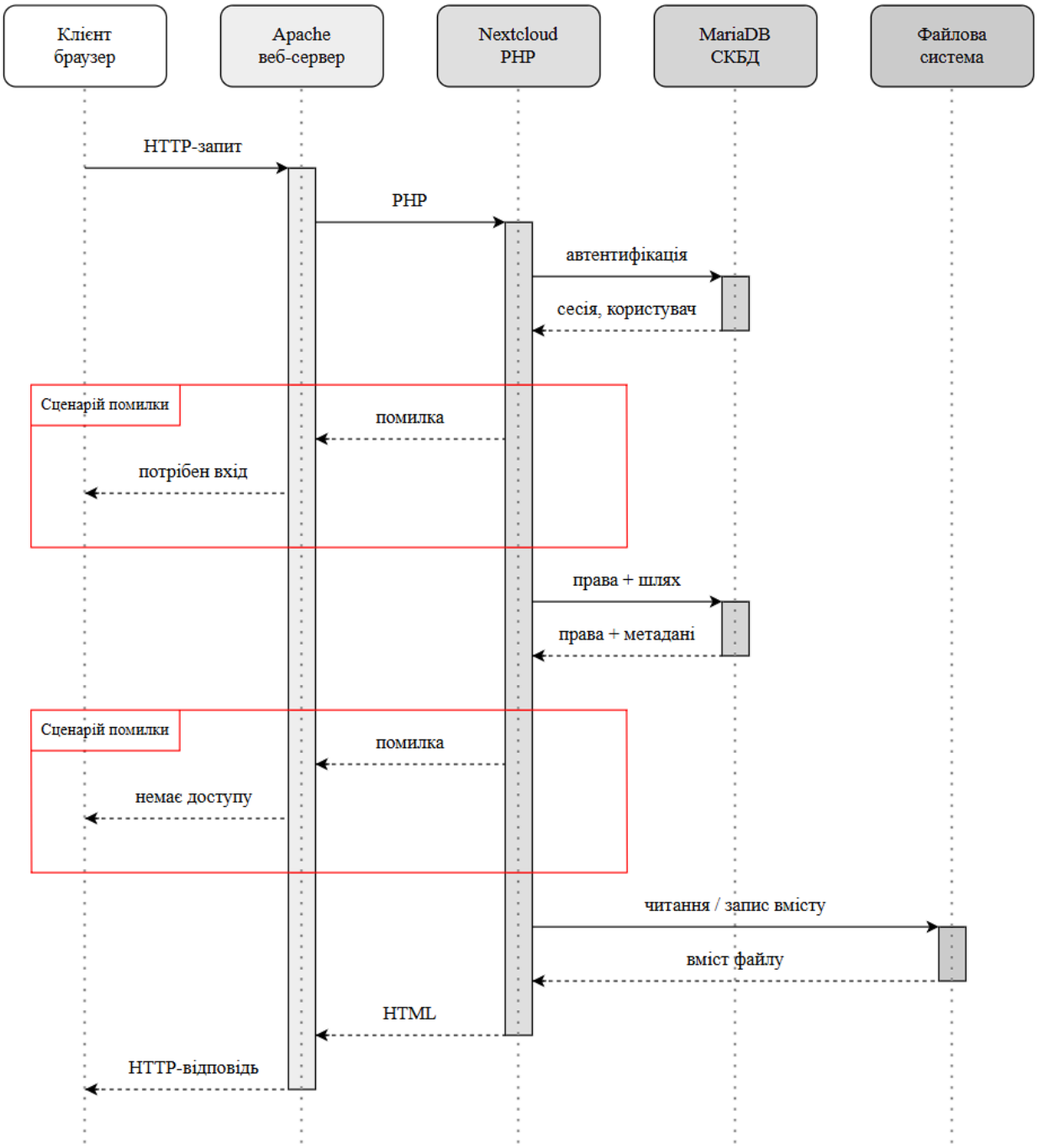

Рис. 2.3 – Діаграма послідовності обробки запиту користувача в системі Nextcloud на основі LAMP-стека

Висновки

У цьому розділі розглянуто платформу Nextcloud як самостійно розгортуване рішення з відкритим кодом для синхронізації та спільного доступу

до файлів, схарактеризовано її архітектуру й принцип роботи, а також зіставлено з альтернативними рішеннями. За сформульованими в розділі 1 критеріями, оцінено придатність Nextcloud до реалізації файлового сервера й підтверджено її відповідність вимогам щодо архітектури, безпеки, надійності, продуктивності, сумісності та економічності. Обґрунтовано вибір LAMP-стека як програмного середовища розгортання та розглянуто взаємодію Nextcloud з його компонентами. Отримані результати формують теоретичне підґрунтя для практичної реалізації файлового сервера, якій присвячено наступний розділ.

РОЗДІЛ 3
ВАРІАНТ РЕАЛІЗАЦІЇ ФАЙЛОВОГО СЕРВЕРА З ВИКОРИСТАННЯМ ПЛАТФОРМИ NEXTCLOUD

3.1. Інсталяція та налаштування LAMP-стека в ОС Ubuntu 24.04.4

Для розгортання файлового сервера було використано операційну систему Ubuntu 24.04.4. Ubuntu розгорнуто в середовищі підсистеми Windows для Linux (Windows Subsystem for Linux) на персональному комп'ютері. Такий підхід обрано як зручний спосіб перевірити роботу системи в межах навчальної роботи без виділення окремого фізичного вузла чи розгортання віртуальної машини. Усі подальші дії з інсталяції та налаштування виконуються в командному рядку (терміналі) Ubuntu, через який вводяться команди керування системою (рис. 3.1). Для промислової експлуатації файловий сервер доцільно розгортати на нативній Linux-системі, на окремому фізичному вузлі або віртуальній машині.

Рис. 3.1 – Командний рядок Ubuntu 24.04.4

На першому етапі оновлюємо системні пакети операційної системи командою:

```
sudo apt update && sudo apt upgrade -y
```

Команда складається з двох послідовних дій, поєднаних оператором `&&`, за якого друга дія виконується лише після успішного завершення першої. Перша частина `apt update` оновлює локальний індекс пакетів, зчитуючи актуальні відомості про доступні версії з репозиторіїв. Друга – `apt upgrade` встановлює оновлення для вже наявних у системі пакетів. Прапорець `-y` автоматично підтверджує встановлення, не очікуючи відповіді користувача. Оновлення системи перед розгортанням забезпечує роботу з актуальними та виправленими версіями компонентів і зменшує ризик помилок сумісності.

Після завершення оновлення системи, встановлюємо компоненти LAMP-стека та додаткові модулі PHP, потрібні для роботи Nextcloud. Для цього застосовано команду:

```
sudo apt install apache2 mariadb-server libapache2-mod-php php php-mysql php-gd php-curl php-xml php-mbstring php-zip php-intl php-bcmath php-gmp unzip -y
```

Менеджер пакетів `apt` самостійно визначив і встановив усі залежності. Крім явно зазначених компонентів, до системи було додано низку супутніх пакетів. Хід встановлення відображається в терміналі. Система виводить перелік пакетів, що завантажуються, повідомляє про вже наявні компоненти й автоматично додані залежності (рис. 3.2).

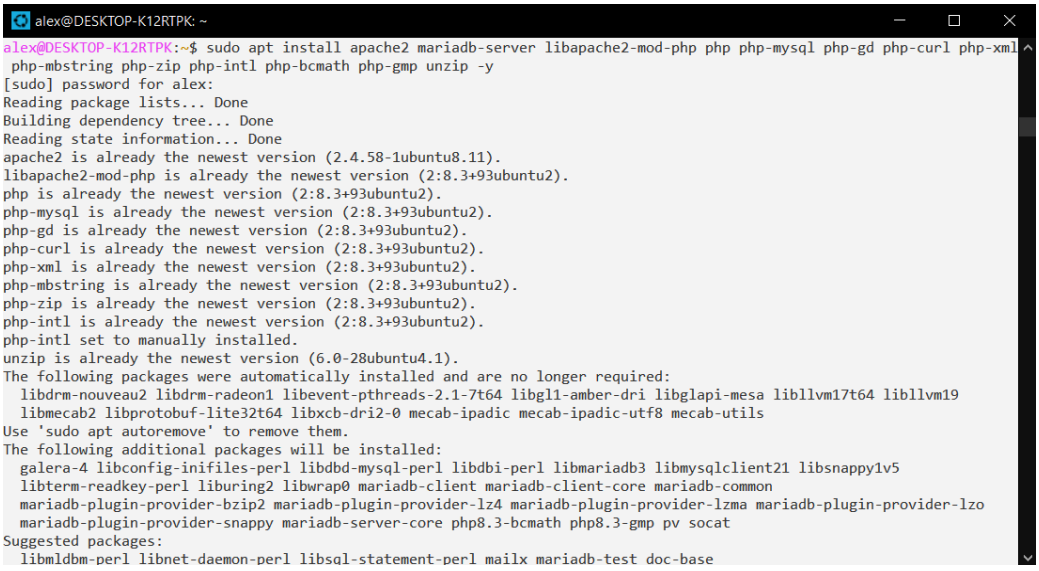

Рис. 3.2 – Хід встановлення компонентів LAMP-стека

Встановлений набір відповідає компонентам LAMP-стека – веб-сервер Apache (`apache2`), СКБД MariaDB (`mariadb-server`), та інтерпретаторові PHP. Також було встановлено пакет `libapache2-mod-php`, що інтегрує PHP безпосередньо у процес Apache, завдяки чому веб-сервер обробляє PHP-сценарії без звернення до зовнішнього обробника. Решту встановлених пакетів становлять додаткові розширення PHP, потрібні для роботи Nextcloud. Модуль `php-mysql` забезпечує взаємодію PHP із базою даних MariaDB; `php-gd` відповідає за обробку зображень і створення мініатюр; `php-curl` і `php-xml` забезпечують

обмін даними мережею та роботу зі структурами XML; `php-mbstring` потрібен для коректної обробки багатобайтового тексту й підтримки Unicode; `php-zip` забезпечує роботу з ZIP-архівами; `php-intl` додає підтримку локалізації та міжнародних форматів даних; `php-bcmath` і `php-gmp` виконують операції з числами довільної точності, потрібні окремим компонентам платформи.

Отже, усі компоненти LAMP-стека встановлено, переходимо до їх налаштування. Починаємо з налаштування СКБД MariaDB, командою:

```
sudo mysql_secure_installation
```

Команда запускає інтерактивний сценарій початкового налаштування MariaDB. Під час налаштування задано пароль адміністратора бази даних. Відхилено пропозицію переходу на автентифікацію через `unix_socket`, який являє собою механізм входу що ототожнює користувача бази даних з обліковим записом операційної системи замість окремого пароля. Також вилучено анонімних користувачів і тестову базу даних, і віддалений вхід для адміністратора заборонено, що зменшує площу можливої атаки та ризик несанкціонованого доступу до сервера. Покрокові запити цього сценарію та відповіді на них у вікні термінала наведено на рис. 3.3.

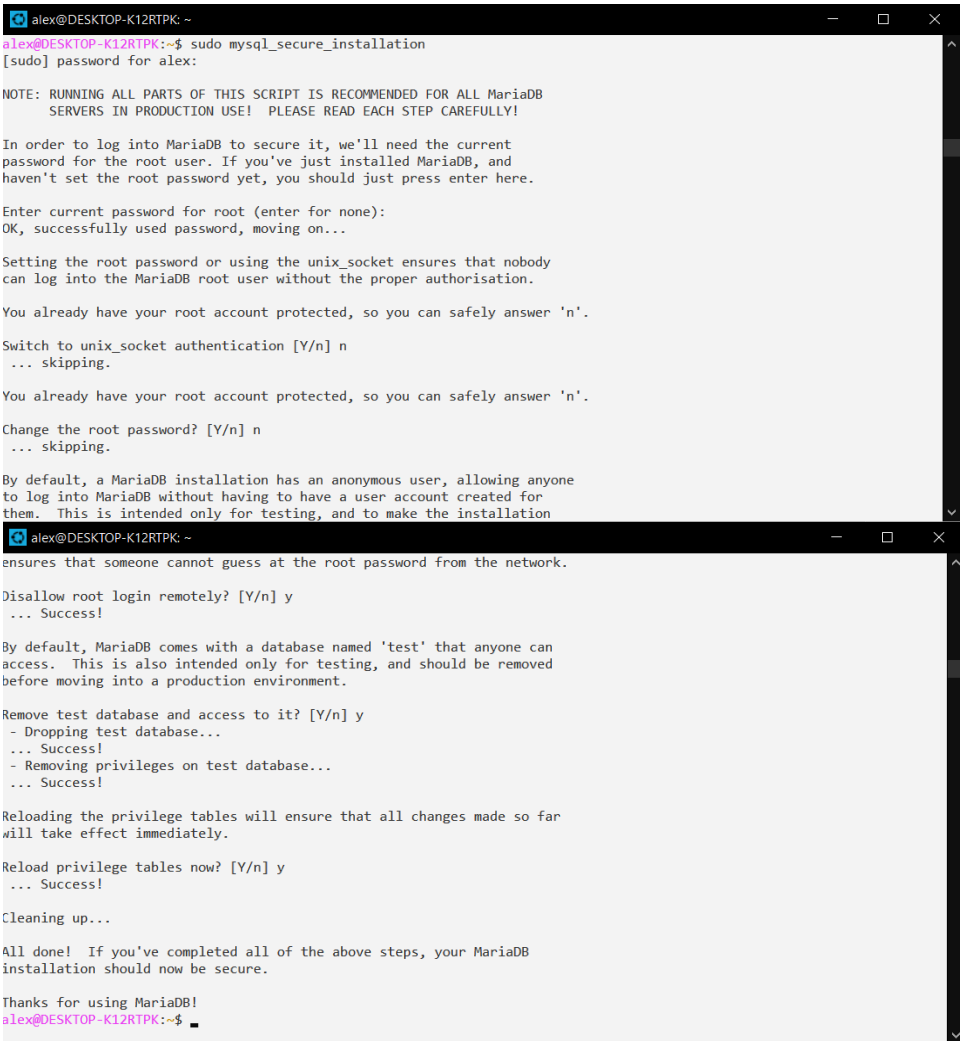

Рис. 3.3 – Інтерактивний сценарій налаштування СКБД MariaDB

Далі необхідно створити окрему базу даних і користувача для роботи платформи Nextcloud. Для цього командою `sudo mysql -u root -p` виконуємо вхід до консолі MariaDB від імені адміністратора, з паролем заданим на попередньому кроці. Після чого послідовно виконуємо SQL-інструкції:

```
CREATE DATABASE nextcloud;
CREATE USER 'alex'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON nextcloud.* TO 'alex'@'localhost';
FLUSH PRIVILEGES;
EXIT
```

Інструкція `CREATE DATABASE nextcloud;` створює базу даних з назвою ‘nextcloud’. Інструкція `CREATE USER 'alex'@'localhost' IDENTIFIED BY 'password';` створює користувача ‘alex’, якому дозволено під’єднання лише з локального вузла ‘localhost’, із паролем для автентифікації. Інструкція `GRANT`

`ALL PRIVILEGES ON nextcloud.* TO 'alex'@'localhost';` надає користувачу ‘alex’ повний набір прав у межах бази даних ‘nextcloud’. Далі, командою `FLUSH PRIVILEGES;` оновлюємо таблиці прав, таким чином попередні зміни набувають чинності. Командою `EXIT` завершуємо сеанс роботи з консоллю MariaDB. Повний перебіг цих дій у консолі, від під’єднання до виходу, показано на рис. 3.4.

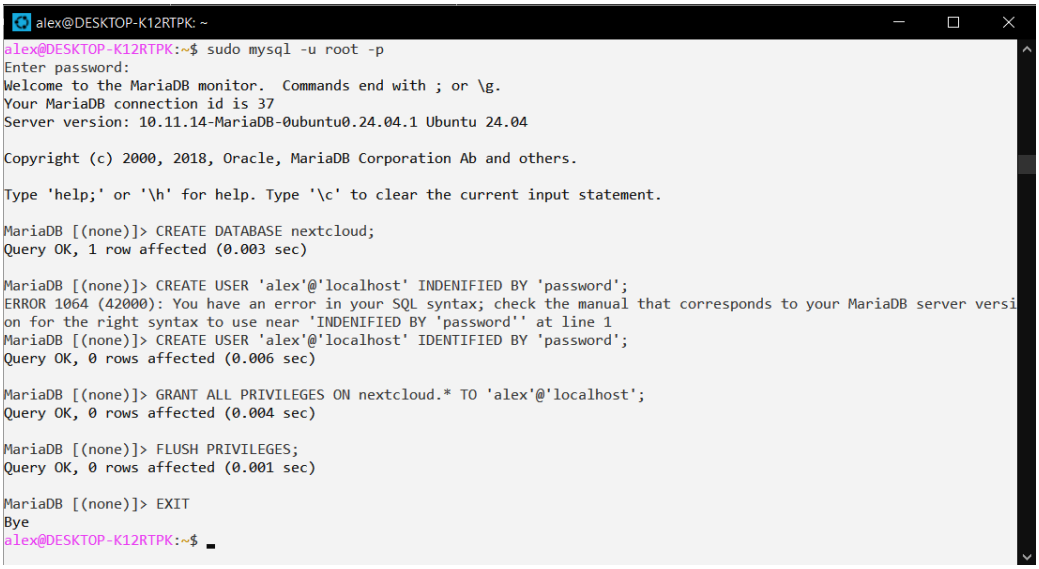

Рис. 3.4 – Створення бази даних, користувача та надання привілеїв у консолі MariaDB

Створення окремої бази даних ізолює службові дані платформи від інших сервісів сервера, а окремий обліковий запис із визначеним набором прав підвищує безпеку та спрощує адміністрування. Обмеження під’єднання адресою ‘localhost’ забороняє доступ до бази даних із зовнішніх вузлів. Користувачеві ‘alex’ надано повні привілеї у межах бази даних ‘nextcloud’, за потреби суворішого розмежування замість `ALL PRIVILEGES` можна надати лише окремі права, наприклад `SELECT`, `INSERT`, `UPDATE`, `DELETE`.

На цьому етапі попереднього налаштування потребує лише СКБД. Базу даних і обліковий запис для неї має бути створено до встановлення Nextcloud, оскільки майстер встановлення платформи звертається до них під час першого запуску. Решта компонентів стека додаткового налаштування зараз не

потребують. Ubuntu готова до роботи після оновлення пакетів, а інтерпретатор PHP встановлюється з типовими параметрами, достатніми для коректної роботи платформи в межах локального сервера. Веб-сервер Apache починає працювати з типовою конфігурацією, проте для обслуговування Nextcloud його потрібно налаштовувати окремо після завантаження платформи.

3.2. Розгортання платформи Nextcloud на основі LAMP-стека

Коли серверне середовище з компонентами LAMP-стека підготовлено, переходимо до встановлення платформи Nextcloud. Спершу здійснюємо перехід до кореневого каталогу веб-сервера командою `cd /var/www/` та завантажуємо туди архів з актуальною версією платформи за допомогою команди:

```
sudo wget https://download.nextcloud.com/server/releases/latest.zip
```

Каталог `/var/www/` використовується Apache для розміщення веб-застосунків і файлів, до яких надається доступ через браузер. Утиліта `wget` завантажує файл за вказаним посиланням. Використовуємо архів ‘latest.zip’ із офіційного сайту Nextcloud, чим гарантуємо актуальну та стабільну версію.

Завантажений архів розпаковуємо командою `sudo unzip latest.zip`, унаслідок чого створено каталог ‘nextcloud’ із системними файлами платформи, сценаріями PHP, конфігураційними файлами та веб-інтерфейсом. Для коректної роботи веб-сервера налаштовуємо права доступу до файлів, послідовно виконуючи команди:

```
sudo chown -R www-data:www-data nextcloud
sudo chmod -R 755 nextcloud
```

Команда `chown` змінює власника файлів і каталогів. Власником призначено системного користувача ‘www-data’, від імені якого працює Apache. Зміна власника дозволяє серверу читати файли платформи та виконувати дії з ними. Команда `chmod` встановлює права доступу, а значення ‘755’ надає власникові дозвіл на читання, запис та виконання, а решті користувачів – читання та виконання. Прапорець `-R` застосовує зміни рекурсивно до всіх вкладених об’єктів. Можна використати інші права доступу, наприклад, для каталогу

даних користувачів, із міркувань безпеки доцільні суворіші права, наприклад, значення ‘750’, що приховає його вміст від сторонніх облікових записів системи [34].

Щоб Apache обслуговував платформу як окремий ресурс, необхідно створити для неї конфігураційний файл віртуального хоста. Команда `sudo nano /etc/apache2/sites-available/nextcloud.conf` створює файл ‘nextcloud.conf’ і відкриває його в текстовому редакторі ‘nano’, до якого вносимо наступні параметри:

```
<VirtualHost *:80>
      ServerName localhost
      DocumentRoot /var/www/nextcloud

      <Directory /var/www/nextcloud/>
                Require all granted
                AllowOverride All
                Options FollowSymLinks MultiViews
      </Directory>
</VirtualHost>
```

Конфігураційний файл ‘nextcloud.conf’ із наведеними параметрами у вікні текстового редактора ‘nano’ продемонстровано на рис.3.5.

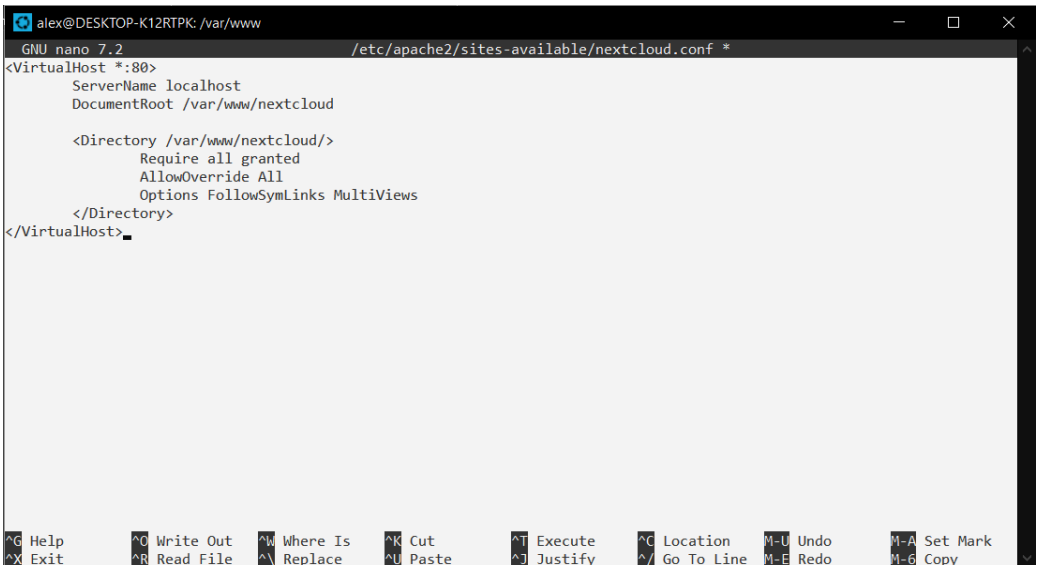

Рис. 3.5 – Конфігураційний файл віртуального хоста ‘nextcloud.conf’

Директива `VirtualHost` описує окрему конфігурацію веб-застосунку, що обслуговується на порту ‘80’. Параметр `ServerName` задає ім’я, за яким Apache розпізнає запити до хоста, а `DocumentRoot` визначає каталог із файлами веб-застосунку. Блок `Directory` містить правила доступу до цього каталогу: директива `Require all granted` дозволяє доступ до веб-застосунку через браузер; `AllowOverride All` дає змогу застосовувати локальні правила з файлів ‘.htaccess’, зокрема правила перезапису URL, які потрібні для Nextcloud; `Options FollowSymLinks MultiViews` керує переходом за символічними посиланнями та узгодженням вмісту. Замість редактора ‘nano’ можна скористатися будь-яким іншим текстовим редактором, наприклад, ‘vim’ або ‘gedit’.

Аби конфігурація набула чинності, активуємо новий віртуальний хост і потрібні модулі Apache, після виконання перезапускаємо веб-сервер. Послідовно виконуємо команди:

```
sudo a2ensite nextcloud.conf
sudo a2enmod rewrite headers env dir mime
sudo systemctl restart apache2
```

Команда `a2ensite` додає створену конфігурацію до активних сайтів Apache, створюючи символічне посилання в каталозі ‘sites-enabled’. У такий самий спосіб, командою `a2enmod` вмикаються модулі сервера. Модуль `rewrite` забезпечує механізм перезапису URL-адрес і коректну маршрутизацію Nextcloud. Модулі `headers` і `env` керують службовими HTTP-заголовками та змінними середовища. `dir` і `mime` відповідають за обробку типів файлів і роботу з індексними сторінками каталогів. Перезапуск служби командою `sudo systemctl restart apache2` застосовує внесені зміни. За потреби уникнути розриву наявних з’єднань використовують команду `systemctl reload apache2`.

Результат активації віртуального хоста й модулів, а також повідомлення про успішний перезапуск Apache наведено на рис. 3.6.

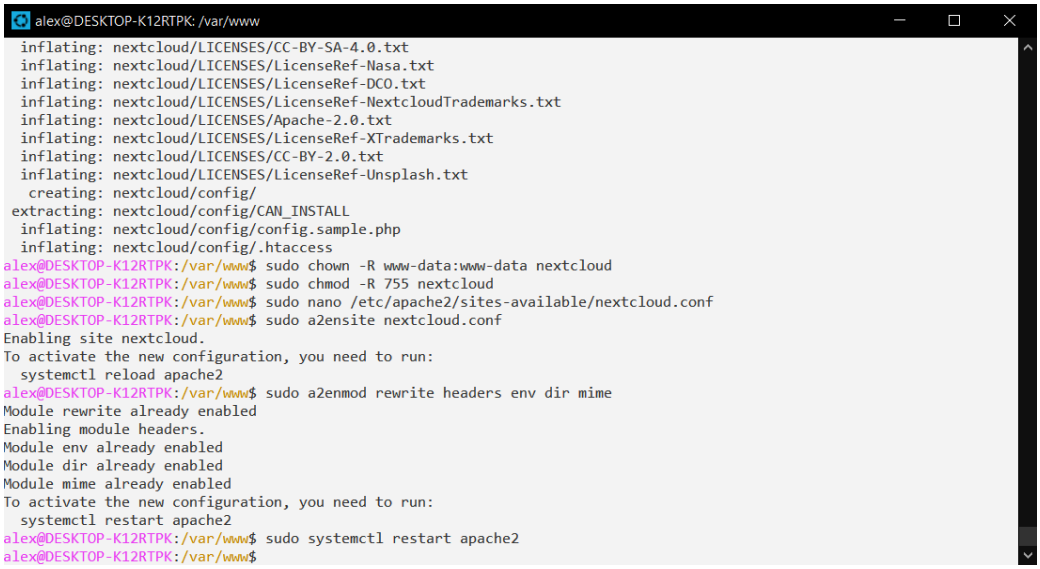

Рис. 3.6 – Активація віртуального хоста й модулів та перезапуск веб-сервера Apache

На цьому налаштування серверної частини завершено. Веб-сервер обслуговує каталог із файлами платформи, тож подальші дії з платформою, за можливості, відбуваються через веб-інтерфейс. Для цього у браузері Google Chrome відкриваємо веб-інтерфейс Nextcloud за адресою ‘localhost’. На сторінці ініціалізації, наведеній на рис. 3.7, створюємо обліковий запис адміністратора з іменем ‘OleksandrGeraskin’ та паролем. Також задаємо параметри під’єднання до бази даних: ім’я бази даних ‘nextcloud’, користувач ‘alex’ та пароль, створені під час налаштування MariaDB. Після під’єднання до бази даних, Nextcloud створює структуру бази й фактично стає готовим до використання.

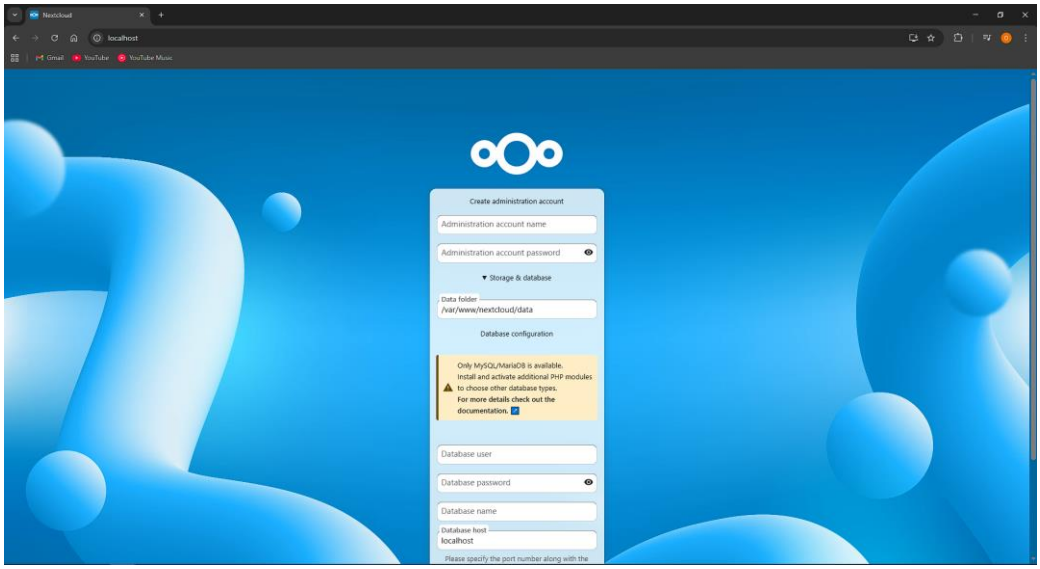

Рис. 3.7 – Сторінка ініціалізації Nextcloud за адресою ‘localhost’

Завершення початкової конфігурації робить платформу Nextcloud доступною для подальшого використання й налаштування. Головну сторінку робочого веб-інтерфейсу продемонстровано на рис. 3.8.

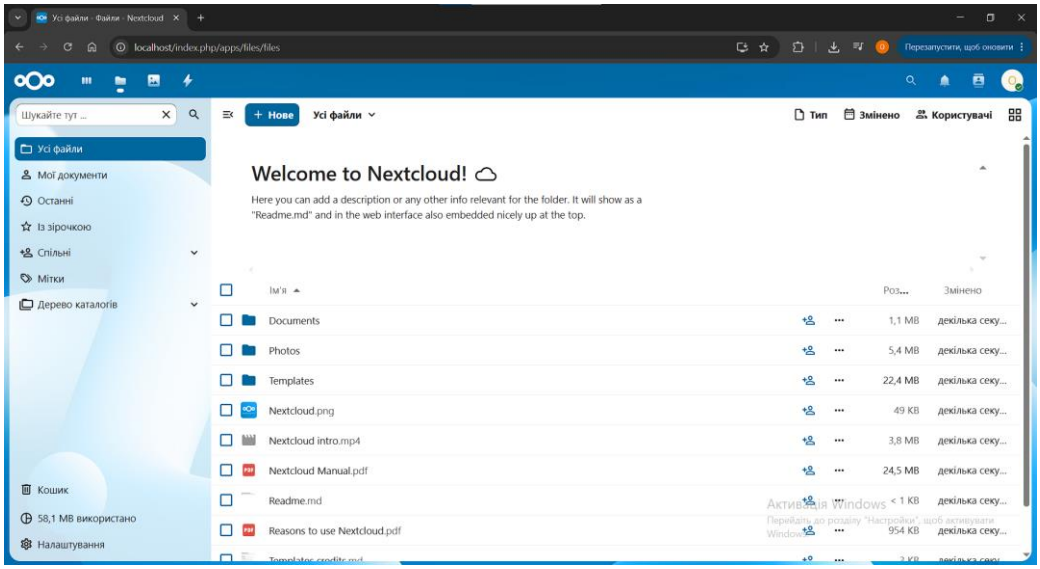

Рис. 3.8 – Робочий веб-інтерфейс платформи Nextcloud

3.3. Налаштування доступу до сервера в локальній мережі

Щоб до сервера могли звертатися інші пристрої локальної мережі, необхідно налаштувати його доступність. Оскільки Ubuntu розгорнуто в

середовищі WSL, спершу визначаємо локальну IP-адресу хоста. Саме за нею сервер буде доступний у мережі. Адресу отримаємо засобами Windows командою `ipconfig`. У відповіді знаходимо потрібну IPv4-адресу комп'ютера в локальній бездротовій мережі – '192.168.0.136'. Результат виконання команди наведено на рис. 3.9.

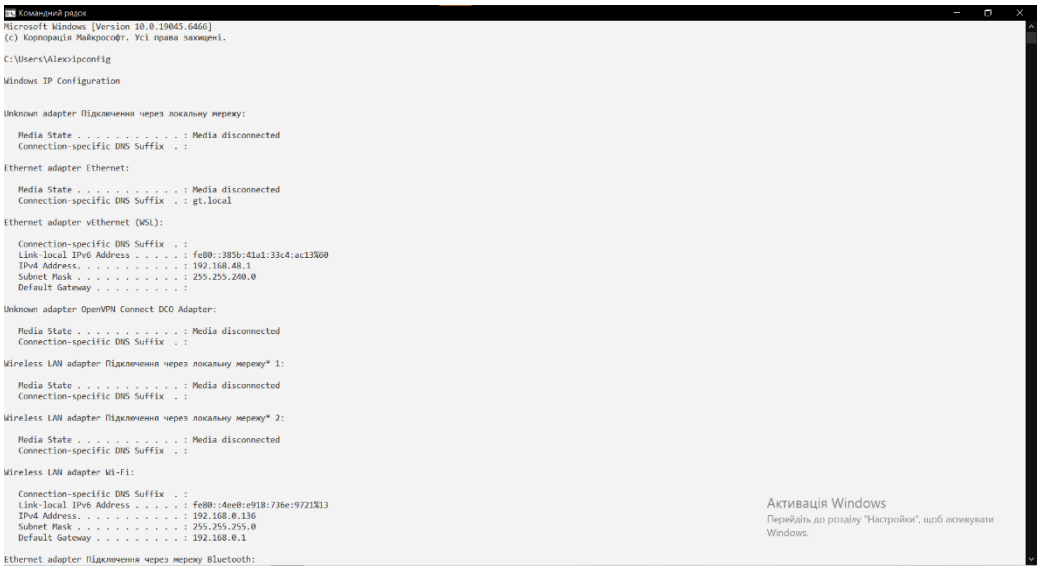

Рис. 3.9 – Визначення локальної IP-адреси хоста командою `ipconfig`

WSL за замовчуванням використовує ізольовану віртуальну мережу з трансляцією адрес. Тому платформа Nextcloud отримує внутрішню IP-адресу WSL і напряму недоступна іншим пристроям мережі. Щоб зробити сервер доступним за локальною адресою хоста, на боці Windows у консолі PowerShell, запущеній з правами адміністратора, налаштовуємо переадресацію портів засобами `netsh`. Внутрішню адресу WSL попередньо отримуємо в терміналі Ubuntu командою `hostname -I`, у відповідь отримано '192.168.59.174'. Після чого у PowerShell виконуємо команду:

```
netsh interface portproxy add v4tov4 listenaddress=192.168.0.136
listenport=80 connectaddress=192.168.59.174 connectport=80
```

Параметр `listenaddress` задає локальну адресу хоста, на якій приймаються вхідні запити; `listenport` – номер порту, на якому хост очікує ці запити. Параметри `connectaddress` і `connectport` визначають внутрішню адресу

та порт WSL, на які запити перенаправляються. Унаслідок цього запити, що надходять на порт 80 хоста, спрямовуються на відповідний порт WSL.

Отриману командою `ipconfig` адресу вносимо до конфігураційного файлу платформи. Для цього відкриваємо файл ‘config.php’, використовуючи команду:

```
sudo nano /var/www/nextcloud/config/config.php
```

У файлі до параметра `trusted_domains` додаємо перелік адрес, через які дозволено доступ до системи:

```
'trusted_domains' =>
array (
      0 => 'localhost',
      1 => '127.0.0.1',
      2 => '192.168.0.136',
),
```

Вигляд конфігураційного файлу ‘config.php’ з доданим `trusted_domains` переліком зображено на рис. 3.10.

Рис. 3.10 – Конфігураційний файл ‘config.php’ із зміненим параметром `trusted_domains`

Механізм `trusted_domains` захищає платформу від несанкціонованого доступу та підміни доменних імен. Якщо адреса, через яку виконується під’єднання, відсутня в переліку, система блокує доступ до веб-інтерфейсу. Адресу ‘localhost’ було внесено автоматично під час встановлення платформи,

тому до переліку додано дві адреси – ‘127.0.0.1’ та ‘192.168.0.136’. Адреса ‘localhost’ – символьне позначення поточного комп’ютера, яке система перетворює на петлеву (loopback) IP-адресу ‘127.0.0.1’. Обидва записи відповідають зверненню комп’ютера до самого себе й забезпечують локальний доступ безпосередньо із серверного комп’ютера. Різниця полягає лише у формі, числова або іменна адреса. Третя адреса, ‘192.168.0.136’, шлях для під’єднання інших пристроїв локальної мережі.

Описане налаштування забезпечує доступ до сервера в межах локальної мережі. За потреби, платформу можна зробити доступною із глобальної мережі. Для цього серверу надають публічну IP-адресу або налаштовують переадресацію портів на маршрутизаторі, реєструють доменне ім’я та додають його до переліку `trusted_domains`. При глобальному доступі, обмін даними обов’язково захищають шифруванням за протоколом HTTPS із використанням сертифіката TLS (Transport Layer Security). Глобальний доступ розширює можливості використання сервера, проте підвищує вимоги до захисту [18].

Застосування нових параметрів вимагає перезапуску веб-сервера, перезапускаємо Apache командою `sudo systemctl restart apache2`. Далі, доцільно перевірити доступність платформи через усі дозволені адреси. Перевіряємо доступність через браузер на серверному комп’ютері, а також з іншого пристрою в локальній мережі. У результаті, підтверджено коректну роботу веб-інтерфейсу та можливість під’єднання до файлового сервера через локальну мережу. Вікна браузера з відкритою платформою за адресами ‘localhost’, ‘127.0.0.1’ і ‘192.168.0.136’, за останньою адресою перевірка здійснена з серверного комп’ютера та з особистого смартфона, продемонстровані на рис. 3.11 – 3.14. Доступ із зовнішніх пристроїв може блокуватися, в такому випадку додатково перевіряють правила брандмауера та відкривають потрібний порт, наприклад, командою `sudo ufw allow 80`.

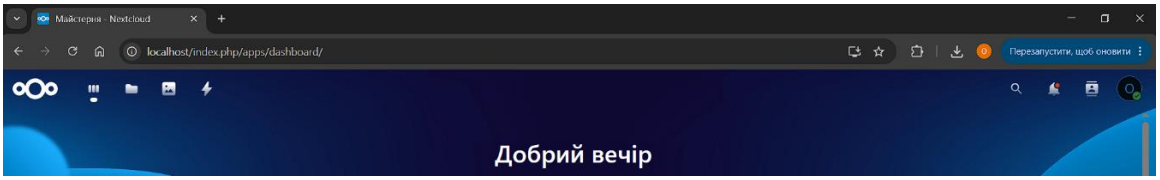

Рис. 3.11 – Вигляд веб-інтерфейсу Nextcloud за адресою ‘localhost’

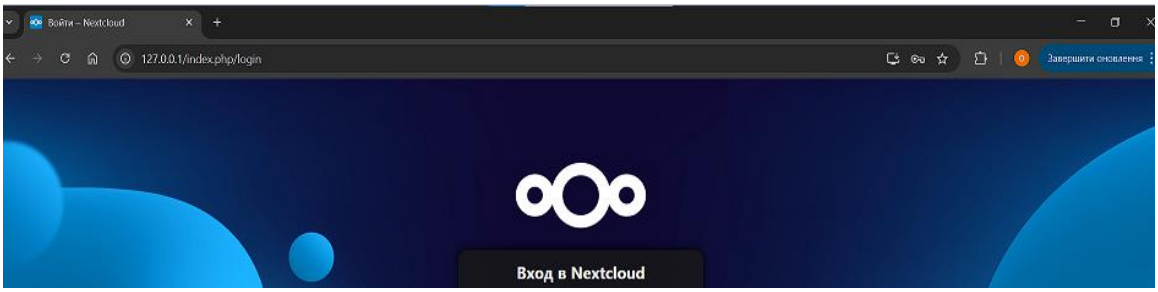

Рис. 3.12 – Вигляд веб-інтерфейсу Nextcloud за адресою ‘127.0.0.1’

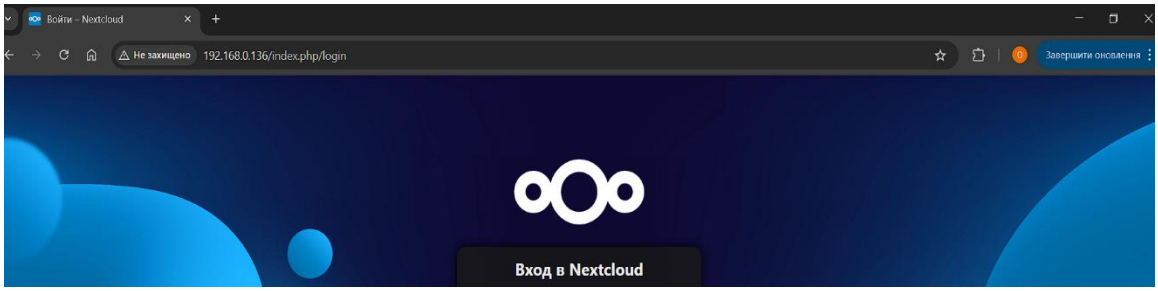

Рис. 3.13 – Вигляд веб-інтерфейсу Nextcloud за адресою ‘192.168.0.136’ з серверного комп’ютера

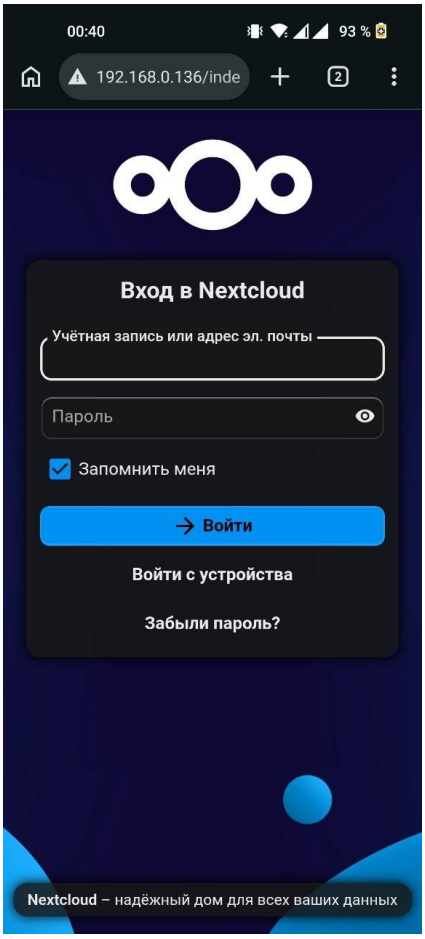

Рис. 3.14 – Вигляд веб-інтерфейсу Nextcloud за адресою ‘192.168.0.136’ з особистого смартфона

3.4. Адміністрування користувачів Nextcloud та розмежування прав доступу

Окремим етапом роботи, є створення користувачів та розмежування прав доступу. Керування обліковими записами зосереджено в адміністративному розділі "Облікові записи". Адміністратор може створювати й вилучати користувачів, задавати ім’я та пароль для входу, відображуване ім’я, тимчасово блокувати облікові записи, а також встановлювати для кожного користувача обмеження на обсяг даних, які дозволено зберігати на сервері. Для демонстрації роботи механізмів розмежування доступу, створено два облікові записи ‘User1’ та ‘User2’ (рис.3.15). Облікові записи часто об’єднують у групи, наприклад, за відділами чи ролями, і надають права не кожному користувачеві окремо, а групі

загалом. Такий підхід зручний для роботи з великою кількістю користувачів, щоб змінити права відразу для певної кількості облікових записів, достатньо змінити налаштування групи. Окремим користувачам можна також делегувати роль адміністратора групи.

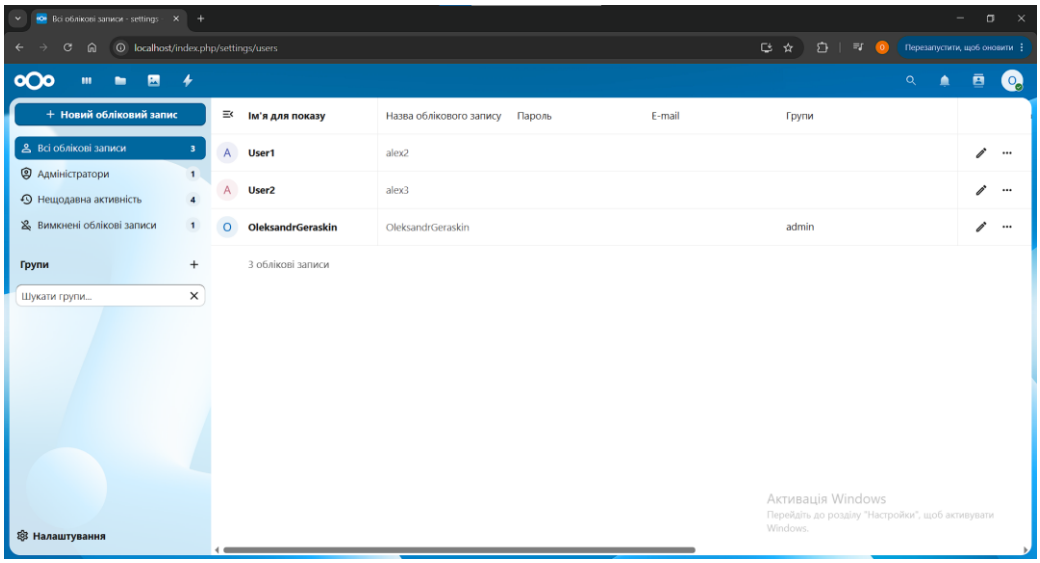

Рис. 3.15 – Розділ Nextcloud "Облікові записи" з доданими користувачами

Маючи створених користувачів, від імені адміністратора створено окремий каталог "Тестовий каталог" і завантажено до нього три файли різних форматів: текстовий документ ‘txt_test.txt’; документ PDF ‘pdf_test.pdf’; документ Word ‘docx_test.docx’ (рис. 3.16). Різні формати файлів обрано щоб перевірити засоби перегляду й редагування для кожного типу.

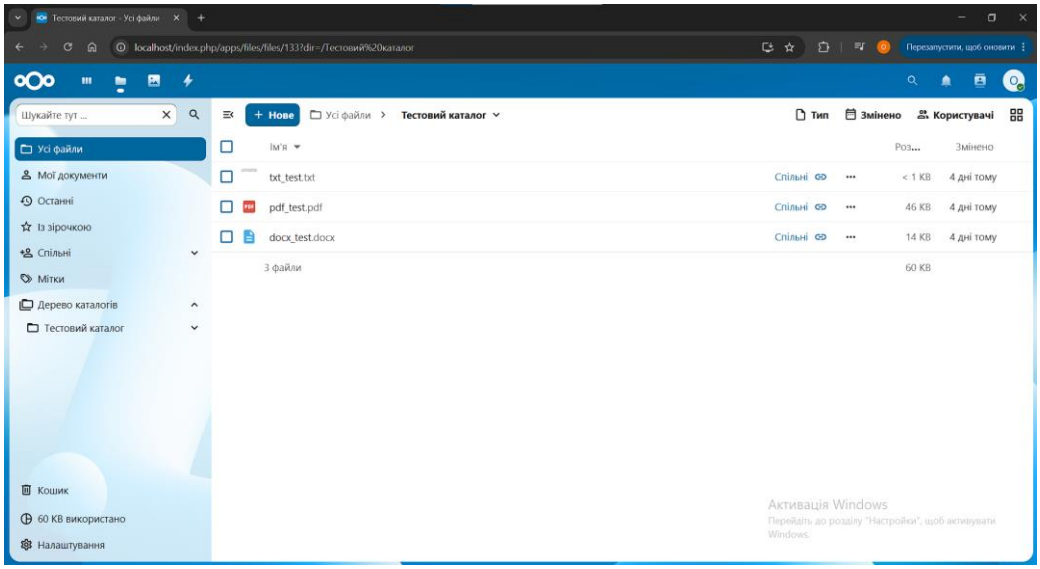

Рис. 3.16 – Каталог "Тестовий каталог" із трьома завантаженими файлами

Адміністратор поділився вмістом каталогу з користувачами ‘User1’ і ‘User2’, також задав кожному різний набір дозволів. Користувач ‘User1’ має право на перегляд та редагування файлів, а користувач ‘User2’ – лише перегляд, без можливості внесення змін (рис. 3.17). Платформа Nextcloud дозволяє гнучко комбінувати окремі дозволи спільного доступу, такі як перегляд, редагування, створення, вилучення та повторний обмін.

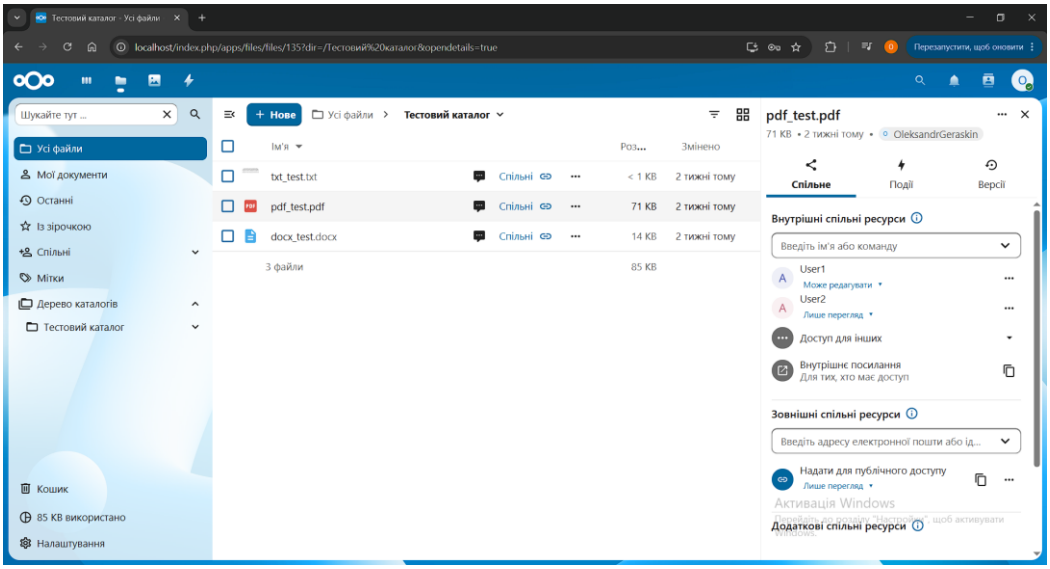

Рис. 3.17 – Налаштування спільного доступу та прав користувачів у веб-інтерфейсі Nextcloud

Надання користувачам різних прав доступу реалізує механізм розмежування прав, який є одним із ключових елементів безпеки файлового сервера. Гнучкі дозволи спільного доступу разом із груповим керуванням відповідають принципам рольового керування доступом і найменших привілеїв. Кожен користувач отримує рівно стільки прав, скільки потрібно для його завдань, що обмежує ризик несанкціонованої зміни чи втрати даних.

3.5. Тестування роботи файлового сервера

Заключним етапом стало тестування роботи розгорнутого файлового сервера. Тестування проводилося за сценарієм типової спільної роботи: користувач ‘User1’ з правами перегляду й редагування та користувач ‘User2’ з правом лише перегляду, по черзі звертаються до спільних файлів, а адміністратор ‘OleksandrGeraskin’ контролює внесені зміни та керує версіями файлів та правами доступу. Для перевірки використано три файли різних форматів розміщені у каталозі "Тестовий каталог".

Для перевірки прав доступу користувача ‘User1’, здійснено вхід до платформи Nextcloud за відповідним ім’ям та паролем. У розділі "Спільні", відображаються файли, до яких адміністратор ‘OleksandrGeraskin’ надав доступ (рис. 3.18).

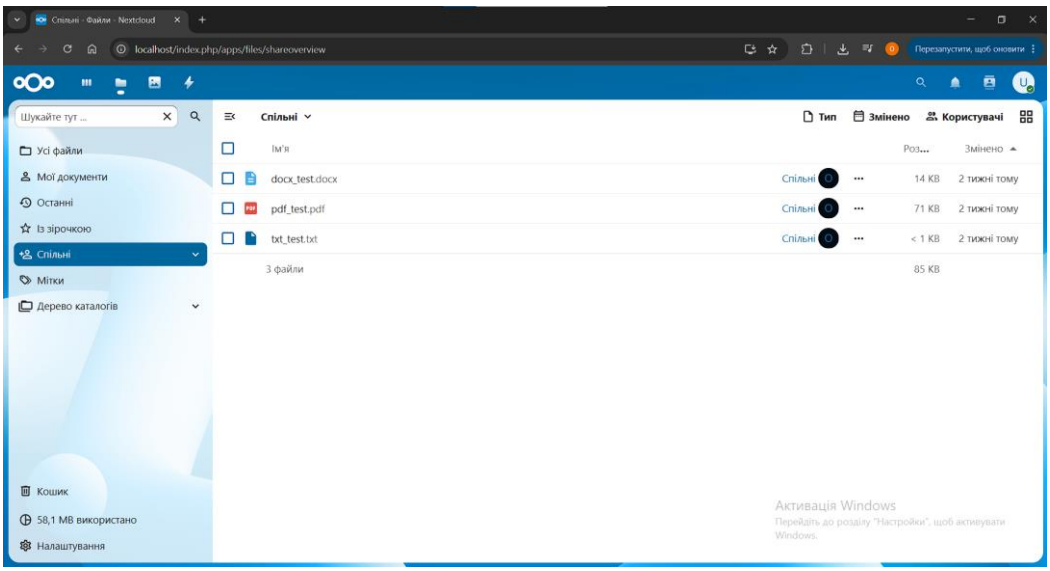

Рис. 3.18 – Розділ "Спільні" користувача ‘User1’

Від імені користувача ‘User1’ відкрито текстовий файл ‘txt_test.txt’. Файл відкривається у вбудованому в платформу текстовому редакторі. Через наявність відповідних прав, користувач вніс зміни та зберіг нову версію файлу. Після внесення змін було додано коментар із поясненням внесених правок (рис. 3.19). Важливо зауважити, коментарі, версії та інформацію про інші дії з файлом доступні до перегляду всім користувачам з доступом до файлу.

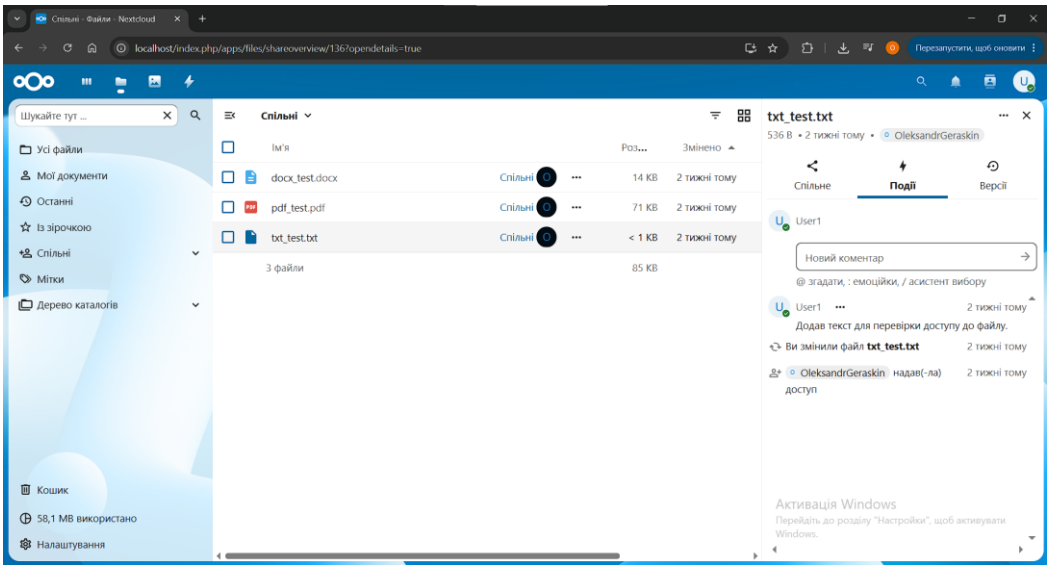

Рис. 3.19 – Інформаційне поле файлу ‘txt_test.txt’ з повідомленнями про зміну файлу та доданим коментарем

Документ формату .docx, на відміну від текстового файлу, у базовій конфігурації платформи, без встановленого офісного застосунку Collabora Online чи OnlyOffice, не редагується безпосередньо у вбудованому редакторі. Тому дії з файлом ‘docx_test.docx’ виконуються після його завантаження на комп’ютер. Користувач ‘User1’ завантажив файл і вніс зміни використовуючи застосунок Microsoft Word. Під час завантаження на сервер, платформа виявила конфлікт версій і запропонувала варіанти їх узгодження, було обрано збереження нової версії файлу, замість зміни імені нового файлу (рис. 3.20).

Після завантаження файлу, на сервері з’явилась нова версія, яка стала поточною.

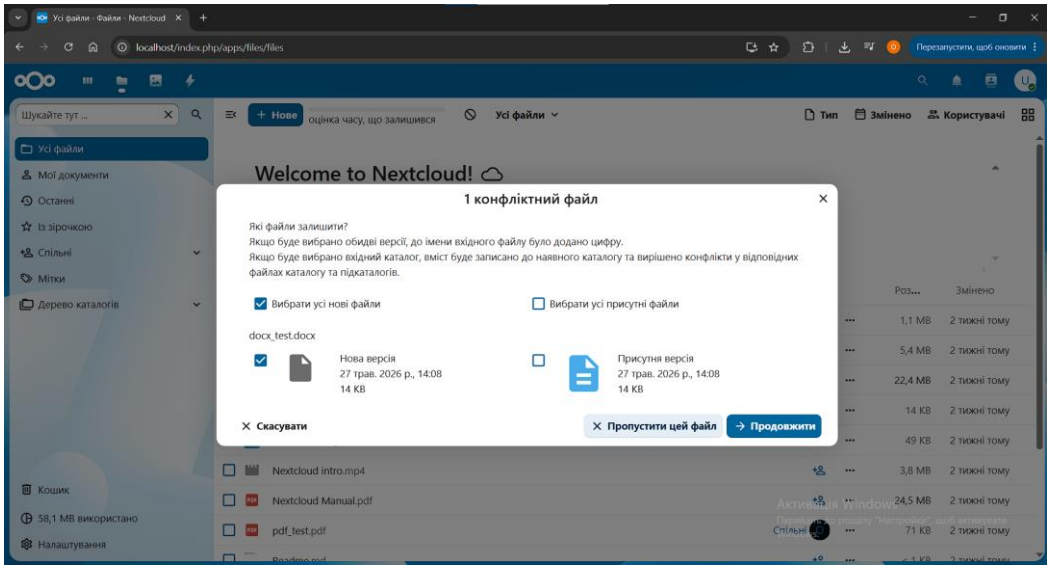

Рис. 3.20 – Узгодження конфлікту версій файлу під час завантаження ‘docx_test.docx’ на сервер

Для документів формату .pdf, платформа має вбудований засіб перегляду з можливістю додавання текстових анотацій та графічних позначок. Користувач ‘User1’ відкрив файл ‘pdf_test.pdf’ і вніс текстові та графічні зміни, зберіг їх і залишив коментар. Таким чином підтверджено, що користувач ‘User1’ із правами перегляду та редагування може змінювати спільні файли.

Під час перевірки прав доступу користувача ‘User2’, виконуємо таку саму послідовність дій. Спільні файли дозволено переглядати та завантажувати. Під час перегляду файлів ‘txt_test.txt’ та ‘pdf_test.pdf’ можливість вносити зміни відсутня, що підтверджує відсутність відповідних прав. Для дій з файлом ‘docx_test.docx’, користувач завантажив його та вніс зміни у застосунку Microsoft Word. Під час спроби завантажити змінений файл назад на сервер, платформа заблокувала операцію через недостатні права доступу, повідомивши про це (рис. 3.21). Це підтверджує, що права доступу користувача ‘User2’ працюють коректно й не дозволяють зміну файлу.

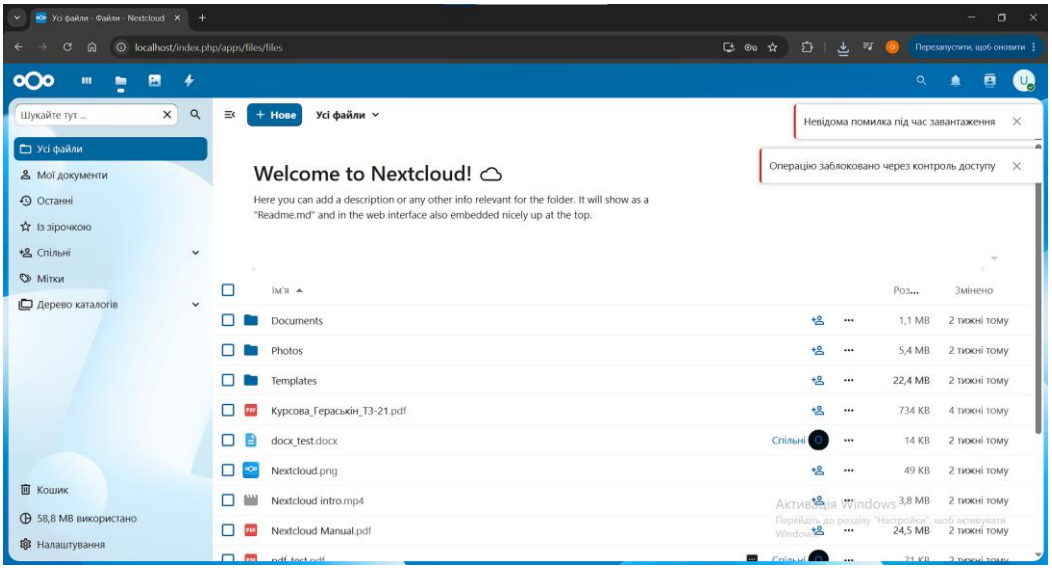

Рис. 3.21 – Блокування завантаження файлу на сервер через недостатні права доступу користувача ‘User2’

Оскільки користувачу ‘User2’ внести зміни не вдалося, він залишає коментар із проханням до адміністратора скасувати попередні правки та надати йому право редагування файлу. Від імені адміністратора ‘OleksandrGeraskin’, внесено зміни прав доступу користувачів. Користувачу ‘User2’ надано дозвіл змінювати файл ‘docx_test.docx’, а у користувача ‘User1’ вилучено право редагування цього файлу (рис. 3.22).

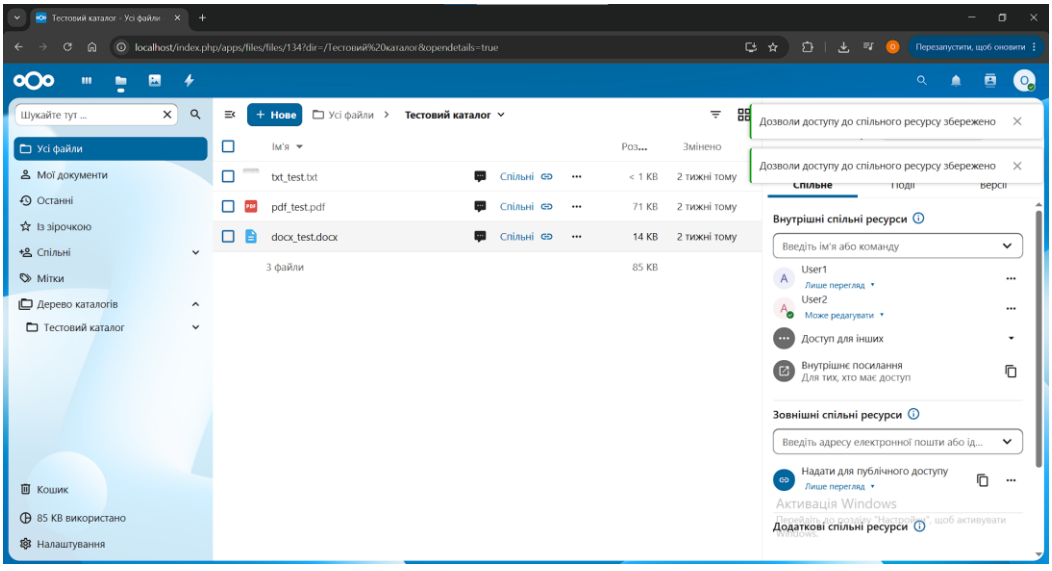

Рис. 3.22 – Зміна прав доступу користувачів від імені адміністратора ‘OleksandrGeraskin’

Адміністратор, крім перегляду версій файлу, має змогу вилучати попередні версії. Видаленню не підлягає версія зі станом "Поточна версія", платформа блокує спробу видалення поточної версії файлу. Адміністратор має можливість надавати версіям стан "Поточна версія" (рис. 3.23). За потреби, адміністратор може надавати окреме право на вилучення версій користувачам.

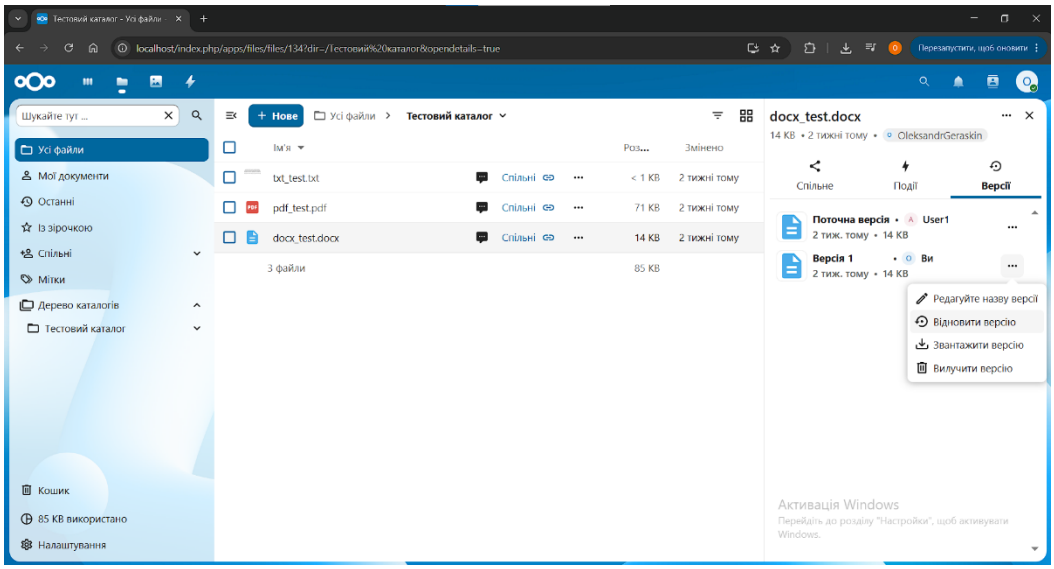

Рис. 3.23 – Можливі дії адміністратора з версіями файлів

Висновки

У третьому розділі, реалізовано файловий сервер на основі платформи Nextcloud і стека LAMP у середовищі Ubuntu 24.04.4. Описано інсталяцію та налаштування компонентів LAMP, розгортання Nextcloud з конфігуруванням віртуального хоста Apache та потрібних модулів, а також налаштування доступу до сервера в локальній мережі. Виконано адміністрування користувачів із розмежуванням прав доступу на основі гнучких дозволів спільного доступу та групового керування, що відповідає принципам RBAC і найменших привілеїв. Проведене тестування за сценарієм типової спільної роботи з файлами різних форматів підтвердило коректну роботу розгорнутого сервера: спільний доступ, керування версіями та розмежування прав, чим засвідчено практичну придатність обраного рішення до експлуатації.

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

У результаті проведеного дослідження було досягнуто поставлену мету, а саме: спроєктовано та практично реалізовано локальний файловий сервер на основі платформи Nextcloud зі стеком LAMP, що забезпечує спільний доступ до даних, розмежування прав користувачів і керування версіями файлів на власній контрольованій інфраструктурі без залучення стороннього хмарного провайдера.

Для досягнення мети, було вирішено ряд завдань:

1. Проаналізовано призначення, функції та архітектуру файлових серверів, розглянуто моделі організації сховища і моделі розгортання. На цій основі сформульовано систему вимог і критеріїв, яким має відповідати придатний до експлуатації файловий сервер.
2. Обґрунтовано вибір платформи Nextcloud і програмного середовища LAMP: платформу схарактеризовано та зіставлено з альтернативними рішеннями, а її придатність підтверджено оцінюванням за визначеними раніше критеріями.
3. Спроєктовано та практично реалізовано локальний файловий сервер у середовищі Ubuntu 24.04.4: розгорнуто стек LAMP і платформу Nextcloud, налаштовано доступ до сервера в локальній мережі, виконано адміністрування користувачів і розмежування прав доступу на основі гнучких дозволів спільного доступу.
4. Перевірено працездатність створеного файлового сервера шляхом тестування основних сценаріїв його роботи із залученням користувачів з різними наборами прав та файлів різних форматів.

Аналіз отриманих результатів засвідчив практичну придатність обраного рішення до експлуатації. Запропоновану послідовність проєктування, розгортання й налаштування може бути використано як практичний орієнтир для невеликих організацій та навчальних закладів, що потребують власного

контрольованого сховища даних зі спільним доступом за відсутності ліцензійних витрат.

Крім того, результати дослідження показали, що застосування гнучких дозволів спільного доступу разом із груповим керуванням узгоджується з принципами рольового керування доступом і найменших привілеїв, а модульна архітектура Nextcloud дає змогу в подальшому розширювати базову функціональність файлового сервера засобами спільного редагування документів, керування календарями й контактами та іншими компонентами без зміни ядра системи, відповідно до зростання потреб.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. File server. Wikipedia. URL: https://en.wikipedia.org/wiki/File_server (Last accessed: 20.01.2026)
2. File server: what is a file server and how does it work? IONOS Digital Guide. URL: https://www.ionos.com/digitalguide/server/know-how/file-server/ (Last accessed: 22.01.2026)
3. What is a file server? BroadbandSearch. URL: https://www.broadbandsearch.net/definitions/file-server (Last accessed: 27.01.2026)
4. File server. phoenixNAP IT Glossary. URL: https://phoenixnap.com/glossary/file-server (Last accessed: 31.01.2026)
5. What is SAN (storage area network)? NetApp. URL: https://www.netapp.com/data-storage/what-is-san-storage-area-network/ (Last accessed: 03.02.2026)
6. The difference between SAN and NAS. TechTarget. URL: https://www.techtarget.com/searchstorage/answer/The-difference-between-SAN-and-NAS (Last accessed: 03.02.2026)
7. Storage on demand – the storage architecture DAS, NAS, SAN. first colo. URL: https://firstcolo.net/en/storage-on-demand-the-storage-architecture-das-nas-san/ (Last accessed: 09.02.2026)
8. What is a file server? UGREEN. URL: https://nas.ugreen.com/blogs/knowledge/what-is-a-file-server (Last accessed: 11.02.2026)
9. Azure Files. Azure Well-Architected Framework. Microsoft Learn. URL: https://learn.microsoft.com/en-us/azure/well-architected/service-guides/azure-files (Last accessed: 16.02.2026)
10. File server. RS Inc. URL: https://www.rsinc.com/file-server.php (Last accessed: 18.02.2026)

11. What is a file server? MyWorkDrive. URL: https://www.myworkdrive.com/blog/file-server (Last accessed: 20.02.2026)
12. Hybrid cloud storage vs on-prem: what data goes where? ComputerWeekly. URL: https://www.computerweekly.com/feature/Hybrid-cloud-storage-vs-on-prem-What-data-goes-where (Last accessed: 21.02.2026)
13. On-premise data centers. Red River. URL: https://redriver.com/data-center/on-premise-data-centers (Last accessed: 23.02.2026)
14. Cloud storage vs on-premise storage. MyWorkDrive. URL: https://www.myworkdrive.com/blog/cloud-storage-vs-on-premise-storage (Last accessed: 23.02.2026)
15. On-premises vs cloud vs hybrid storage. Exxact. URL: https://www.exxactcorp.com/blog/storage/on-premises-vs-cloud-vs-hybrid-storage (Last accessed: 23.02.2026)
16. How to choose between on-premises, hybrid, and cloud storage models. One Data Software. URL: https://www.onedatasoftware.com/blog/choose-between-on-premises-hybrid-and-cloud-storage-models (Last accessed: 23.02.2026)
17. Piantadosi G., Marrone S., Sansone M., Sansone C. A secure, scalable and versatile multi-layer client–server architecture for remote intelligent data processing //Journal of Reliable Intelligent Environments Volume 1, 2015, pp.173-187. – Режим доступу: https://doi.org/10.1007/s40860-015-0007-1
18. Seth V. Analyzing and Comparing the Performance of SMB and NFS Protocols for Efficient File Sharing in Linux Environments //International Journal of Scientific Research & Engineering Trends (IJSRET) Volume 9, Issue 1, Jan-Feb 2023, ISSN 2395-566X. – Режим доступу: https://ijsret.com/wp-content/uploads/IJSRET_V9_issue1_138.pdf
19. Kumar B., Banerjee S. A Theoretical Study on the Performance Characteristics of Different Cloud Storage Access Protocols //International Journal for Multidisciplinary Research (IJFMR) Volume 7, Issue 4, July-August 2025, E-ISSN 2582-2160, paper ID IJFMR250452726. – Режим доступу: https://www.ijfmr.com/papers/2025/4/52726.pdf

20. Patterson D.A., Gibson G., Katz R.H. A Case for Redundant Arrays of Inexpensive Disks (RAID) //Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data 1988, pp.109-116.
21. Chen P.M., Lee E.K., Gibson G.A., Katz R.H., Patterson D.A. RAID: High-Performance, Reliable Secondary Storage //ACM Computing Surveys Volume 26, Issue 2, 1994, pp.145-185.
22. Sandhu R.S., Coyne E.J., Feinstein H.L., Youman C.E. Role-Based Access Control Models //IEEE Computer Volume 29, Issue 2, 1996, pp.38-47.
23. Ferraiolo D.F., Sandhu R., Gavrila S., Kuhn D.R., Chandramouli R. Proposed NIST Standard for Role-Based Access Control //ACM Transactions on Information and System Security Volume 4, Issue 3, 2001, pp.224-274.
24. Програмне забезпечення : навчальні матеріали [Електронний ресурс] // StudFile. – Режим доступу: https://studfile.net/preview/9410503/page:5/ (Last accessed: 27.02.2026).
25. Aryan P., Shetty S.D. Designing a Secure, Scalable, and Cost-Effective Cloud Storage Solution: A Novel Approach to Data Management using NextCloud, TrueNAS, and QEMU/KVM //arXiv preprint arXiv:2412.05091 [cs.DB] 2024. – Режим доступу: https://arxiv.org/abs/2412.05091
26. Nextcloud user manual [Електронний ресурс] // Nextcloud Documentation. – Режим доступу: https://docs.nextcloud.com/server/latest/user_manual/en/ (Last accessed: 30.03.2026)
27. What is a LAMP stack? [Електронний ресурс] // Amazon Web Services. – Режим доступу: https://aws.amazon.com/what-is/lamp-stack/ (Last accessed: 15.03.2026)
28. Official Ubuntu documentation [Електронний ресурс] // Ubuntu. – Режим доступу: https://help.ubuntu.com/ (Last accessed: 19.03.2026)
29. Apache HTTP Server documentation [Електронний ресурс] // The Apache Software Foundation. – Режим доступу: https://httpd.apache.org/docs/ (Last accessed: 19.03.2026)

30. MariaDB documentation [Електронний ресурс] // MariaDB. – Режим доступу: https://mariadb.com/docs (Last accessed: 19.03.2026)
31. PHP: розширення (Extensions) [Електронний ресурс] // PHP Manual. – Режим доступу: https://www.php.net/manual/uk/extensions.php (Last accessed: 25.03.2026)