ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «СИСТЕМА ВПРОВАДЖЕННЯ ХМАРНИХ ТЕХНОЛОГІЙ ДЛЯ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ПІДПРИЄМСТВА»

на здобуття освітнього ступеня магістра
зі спеціальності 123 Комп'ютерної інженерії
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерні системи та мережі
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Назар БОРОДІН*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач(ка) вищої освіти гр. КСДМ-62
Назар БОРОДІН
Ім'я, ПРІЗВИЩЕ
Керівник: Ярослав ТОРОШЕНКО
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Комп'ютерної інженерії

Ступінь вищої освіти Магістр

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо-професійна програма Комп'ютерні системи та мережі

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
_____Ім'я, ПРІЗВИЩЕ
«_____» _____ 2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Бородін Назар Володимирович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: «Система впровадження хмарних технологій для підвищення ефективності підприємства»
керівник кваліфікаційної роботи Ярослав Торошенко, канд.техн.наук, доцент,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «15» жовтня 2024р. №320,
2. Строк подання кваліфікаційної роботи «29» грудня 2024р.
3. Вихідні дані до кваліфікаційної роботи: науково-технічна література з хмарних обчислень, технологічні рамки та протоколи, дизайн і архітектура систем, технічні характеристики та вимоги до інтеграції, аналіз функціональності хмарних рішень, оцінка ефективності та бенчмаркінг підприємств.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
    1. Аналіз інтерфейсів Amazon Web Services
    2. Виконання базового налаштування AWS Instances на web ресурсі
    3. Використання Terraform для підвищення швидкості налаштування AWS Instances
5. Перелік ілюстративного матеріалу:
    1. Мета, об'єкт, предмет дослідження
    2. Актуальність роботи
    3. Система впровадження класичної інфраструктури
    4. Проблеми класичної інфраструктури
    5. Система впровадження хмарної інфраструктури
    6. AWS. Створення EC2
    7. Створення EC2 за допомогою Terraform
    8. Система впровадження гібридної інфраструктури
    9. Порівняння систем впровадження інфраструктури
    10. Висновки
6. Дата видачі завдання «01» вересня 2024р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір даних	10.10-12.10.2024	Виконано
2	Обробка матеріалу	12.10-15.10.2024	Виконано
3	Розробка теоретичної частини	15.10-18.10.2024	Виконано
4	Створення AWS Instances на web-сторінці провайдера	18.10-22.10.2024	Виконано
5	Виконання входу на створений AWS Instances	22.10-26.10.2024	Виконано
6	Написання Terraform коду для створення AWS Instances	26.10-30.10.2024	Виконано
7	Висновки за результатами аналізу	30.10-31.10.2024	Виконано
8	Розробка презентації	31.10-01.11.2024	Виконано
9	Передзахист	09.12-11.12.2024	Виконано
10	Здача в деканат	20.12-29.12.2024	Виконано

Здобувач вищої освіти _____ Бородін НАЗАР
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Ярослав ТОРОШЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістр: 106 стор., 71 рис., 8 табл., 7 джерел.

Мета роботи – Налаштувати хмарні сервіси на платформі AWS від імені адміністратора, забезпечивши швидший процес впровадження завдяки використанню інструменту Terraform.

Об'єкт дослідження – Налаштування хмарного сервера за допомогою інструментів web-сторінки та Terraform.

Предмет дослідження – методи та технології впровадження хмарних обчислень для підвищення ефективності роботи підприємств, які базуються на використанні сучасних обчислювальних платформ та автоматизованих рішень.

Короткий зміст роботи: У роботі розглянуто процес впровадження хмарних технологій на підприємствах, аналіз їхньої ефективності та економічної доцільності. Досліджено провідні хмарні платформи, такі як AWS, Azure та Google Cloud, їхні основні сервіси (EC2, S3, Compute Engine, App Service тощо) та моделі (IaaS, PaaS, SaaS).

Проаналізовано сучасні тенденції впровадження хмарних технологій, їх переваги та обмеження. Хмарні обчислення забезпечують масштабованість, економічну ефективність та високу доступність IT-ресурсів. Запропонована архітектура хмарного рішення дозволяє підприємствам оптимізувати бізнес-процеси, зменшити витрати на підтримку локальної інфраструктури та швидко адаптуватися до змін у ринкових умовах.

Розроблено покрокову інструкцію для базового налаштування хмарної інфраструктури з використанням AWS EC2 та S3. Продемонстровано переваги таких рішень на прикладі написання Terraform коду.

КЛЮЧОВІ СЛОВА: ХМАРНІ ТЕХНОЛОГІЇ, AWS, AZURE, GOOGLE CLOUD, МАСШТАБОВАНІСТЬ, TERRAFIRM, IAAS, PAAS, SAAS, ОПТИМІЗАЦІЯ, ЕФЕКТИВНІСТЬ, INSTANCE

ABSTRACT

Text part of the qualification work for the degree of Master: 106 pages, 71 figures, 8 tables, 7 sources.

Purpose of the work – Configure cloud services on the AWS platform on behalf of the administrator, ensuring a faster implementation process through the use of the Terraform tool.

Object of research – Configuring a cloud server using web-page tools and Terraform.

Subject of research – methods and technologies for implementing cloud computing to improve the efficiency of enterprises based on the use of modern computing platforms and automated solutions.

Summary of the work: The work considers the process of implementing cloud technologies in enterprises, analyzes their efficiency and economic feasibility. Leading cloud platforms such as AWS, Azure and Google Cloud, their main services (EC2, S3, Compute Engine, App Service, etc.) and models (IaaS, PaaS, SaaS) are studied.

Modern trends in the implementation of cloud technologies, their advantages and limitations are analyzed. Cloud computing provides scalability, cost-effectiveness and high availability of IT resources. The proposed architecture of the cloud solution allows enterprises to optimize business processes, reduce costs for supporting local infrastructure and quickly adapt to changes in market conditions.

A step-by-step guide has been developed for basic configuration of cloud infrastructure using AWS EC2 and S3. The advantages of such solutions are demonstrated using the example of writing Terraform code.

KEYWORDS: CLOUD TECHNOLOGIES, AWS, AZURE, GOOGLE CLOUD, SCALABILITY, TERRAFIRM, IAAS, PAAS, SAAS, OPTIMIZATION, EFFICIENCY, INSTANCE

ЗМІСТ

ВСТУП

З кожним роком темпи впровадження інновацій та адаптації до змін зростають. У такій динамічній реальності компанії змушені оперативно реагувати на будь-які зміни на ринку. Якщо раніше тиждень на ухвалення рішення вважався прийнятним терміном, то зараз це неприпустима розкіш, яка може призвести до втрати конкурентоспроможності.

Сучасні технології, зокрема інтернет та гаджети, змінили стиль життя людей: вони більше подорожують, частіше змінюють місце проживання та вимагають високої якості послуг. У таких умовах застарілі рішення, наприклад, централізовані сервери у великих офісах, стають неефективними. Якщо клієнт змушений довго чекати завантаження сайту, він, швидше за все, відмовиться від покупки, що завдасть компанії фінансових втрат.

До того ж, багато стартапів не мають ресурсів для створення та обслуговування власної серверної інфраструктури. Оскільки сучасні додатки орієнтовані на глобальне використання, їм необхідні потужні обчислювальні ресурси за доступною ціною.

Усі ці виклики вирішують провайдери хмарних технологій, пропонуючи гнучкі, масштабовані й економічно вигідні рішення.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Визначення хмарних технологій, телекомунікаційних платформ для їх реалізації та їхні особливості.

Хмарні технології стають дедалі важливішими у сучасному бізнесі. Їхня актуальність зросла на фоні популяризації віддаленої роботи. Хмарні сервіси надають компаніям будь-якого розміру доступ до потужних обчислювальних ресурсів, сховищ даних та інших IT-інструментів, усуваючи потребу в придбанні та обслуговуванні власної IT-інфраструктури.

Основні переваги хмарних технологій для бізнесу:

– **Оптимізація витрат.** Використання хмарних сервісів дозволяє уникнути витрат на покупку, обслуговування та модернізацію обладнання, що особливо вигідно для малого та середнього бізнесу.

– **Гнучкість.** Хмарні рішення забезпечують швидке масштабування ресурсів відповідно до актуальних потреб, що є незамінним для галузей з високою мінливістю.

– **Підвищення ефективності.** Автоматизація процесів за допомогою хмарних сервісів сприяє підвищенню продуктивності роботи.

– **Покращення безпеки.** Провайдери хмарних технологій пропонують сучасні інструменти для захисту даних, забезпечуючи надійність та конфіденційність.

Зростання популярності віддаленої роботи значно підвищило значущість хмарних технологій. Завдяки хмарним сервісам співробітники можуть працювати з будь-якого місця, що допомагає компаніям зберігати конкурентоспроможність у глобальному ринку.

Хмарні технології надають широкий спектр можливостей для бізнесу. Ось деякі приклади їхнього використання:

– **Управління бізнес-процесами.** Хмарні сервіси сприяють автоматизації таких процесів, як управління взаєминами з клієнтами (CRM), ланцюгами поставок (SCM) та персоналом (HR).

– **Зберігання даних.** Хмара може використовуватися для збереження фінансових даних, інформації про клієнтів і даних про продукти.

– **Розробка програмного забезпечення.** Хмарні платформи підтримують створення та розгортання програмних рішень.

– **Аналіз даних.** Хмарні сервіси дозволяють виконувати обробку великих обсягів даних, аналітику та машинне навчання.

Хмарні технології є потужним засобом, який сприяє підвищенню ефективності, продуктивності та конкурентоспроможності компаній незалежно від їх розміру. Вони вже займають ключове місце у сучасному бізнесі, і їх значущість буде лише зростати в майбутньому.

Хмарні майданчики являють собою інтеграцію операційних систем і апаратного забезпечення серверів, розташованих у центрах обробки даних. Вони забезпечують можливість віддаленої та масштабованої взаємодії між програмними й апаратними продуктами.

Платформою хмарних обчислень називається процес надання різноманітних послуг через мережу Інтернет. Це поняття охоплює сервери, мережеву інфраструктуру, сховища даних, бази даних, програмне забезпечення та інші інструменти й сервіси, які використовуються для забезпечення бізнес-потреб.

Завдяки хмарним технологіям, бізнесу більше не потрібно зберігати файли на власних серверах, жорстких дисках або локальних пристроях зберігання. Хмарне середовище дозволяє розміщувати дані у віддалених базах даних. Будь-який пристрій із доступом до Інтернету може отримати доступ до необхідних даних і програм. Деякі хмарні платформи також підтримують роботу з програмами без підключення до Інтернету. Після відновлення з'єднання здійснюється синхронізація з хмарою, зберігаючи локальні зміни.

Хмарні майданчики стають дедалі популярнішими серед людей і підприємств через такі переваги:

- Економія коштів;
- Підвищення швидкості роботи;
- Зростання продуктивності;
- Покращення ефективності;
- Підвищений рівень безпеки.

Підприємства можуть орендувати комп’ютерні послуги, включаючи сховища даних, сервери, бази даних, мережі, аналітичні інструменти, програмне забезпечення та штучний інтелект. Це дозволяє уникнути витрат на придбання й обслуговування обчислювального обладнання та центрів обробки даних. Компанії сплачують лише за послуги, які їм потрібні, що робить хмарні технології вигідним і гнучким рішенням.

Для адаптації до швидко змінюваних технічних потреб були створені різні моделі, типи та служби хмарних обчислень. Кожна компанія може вибрати той варіант, який найкраще відповідає її специфічним вимогам. Існує три основні способи розгортання хмарних сервісів:

- Публічна хмара (Public cloud);
- Приватна хмара (Private cloud);
- Гібридна хмара (Hybrid cloud);

Публічна хмара — це віртуалізоване середовище, яке розширює IT-інфраструктуру підприємства, дозволяючи розміщувати її компоненти та послуги на віртуальних серверах, розташованих поза межами компанії і належних стороннім провайдерам. Управління такими ресурсами здійснюється через мережу Інтернет.

У публічній хмарі кілька компаній користуються спільним обладнанням, сховищами та мережевими пристроями. Зазвичай доступ до послуг і управління обліковим записом відбувається через веб-браузер. Такі хмари часто

використовуються для надання Інтернет-послуг, як-от електронна пошта, офісні застосунки, сховища даних і середовища для тестування чи розробки.

Переваги публічних хмар:

– Економічність: Немає потреби купувати програмне забезпечення чи обладнання.

– Відсутність витрат на обслуговування інфраструктури: Це забезпечує постачальник послуг.

– Масштабованість: Постачальник надає ресурси відповідно до потреб клієнта.

– Надійність: Розгалужена мережа серверів забезпечує стійкість до збоїв.

Серед найбільших провайдерів публічних хмарних послуг — Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), Alibaba та IBM Bluemix, які пропонують широкий спектр можливостей на різних умовах.

Приватна хмара складається з обчислювальних ресурсів, які використовуються виключно однією компанією або організацією. Це означає, що всі ресурси повністю ізольовані і не розділяються з іншими користувачами.

Розміщення та управління приватною хмарою може здійснюватися різними способами. Вона може бути фізично розташована на базі наявної інфраструктури та обладнання організації в її локальному центрі обробки даних або створена на новій інфраструктурі, запропонованій стороннім постачальником.

Переваги приватної хмари:

– Гнучкість. Організація може адаптувати своє хмарне середовище до специфічних бізнес-потреб.

– Контроль. Використання ресурсів обмежене однією компанією, що забезпечує високий рівень контролю та конфіденційності.

– Масштабованість. Приватна хмара пропонує кращу масштабованість у порівнянні з традиційною локальною інфраструктурою.

Приватні хмари значно спрощують налаштування корпоративних IT-ресурсів відповідно до унікальних потреб організації. Їх часто використовують державні

установи, фінансові організації та інші середні й великі компанії з критично важливими бізнес-операціями, які потребують високого рівня контролю та захисту даних.

Гібридна хмара — це рішення, яке об'єднує локальну інфраструктуру (приватну хмару) із публічною хмарою. Такий підхід дозволяє забезпечити обмін даними та застосунками між двома середовищами, використовуючи переваги обох типів хмар.

Багато організацій обирають гібридну хмару через її відповідність бізнес-потребам, включаючи дотримання нормативних вимог, обробку незалежних даних, підвищення рентабельності інвестицій у локальні технології та зниження часу відклику. Сучасні гібридні хмари підтримують граничні робочі навантаження, надаючи обчислювальну потужність безпосередньо на пристроях Інтернету речей (IoT). Це зменшує час передачі даних між пристроями та хмарою, знижує затримки та дозволяє пристроям працювати автономно протягом тривалого часу.

Переваги гібридної хмари:

– Контроль. Дає можливість зберігати критично важливі ресурси або робочі навантаження у приватній інфраструктурі, забезпечуючи низький час відклику.

– Гнучкість. Дозволяє використовувати ресурси публічної хмари за необхідності.

– Економічність. Користувачі сплачують лише за обчислювальну потужність, яку вони фактично використовують, завдяки масштабованості публічної хмари.

– Простота. Можливість поступового переходу на хмарне середовище шляхом послідовного перенесення робочих навантажень.

Гібридна хмара пропонує ефективне та адаптивне рішення для організацій, які прагнуть отримати переваги як від публічних, так і від приватних хмарних середовищ.

1.2 Моделі представлення хмарних сервісів

Хмарні послуги здійснили справжній прорив в обчислювальних системах завдяки моделям IaaS (Infrastructure as a Service), PaaS (Platform as a Service) та особливо SaaS (Software as a Service). Це дозволило підприємствам створювати віртуалізовану IT-інфраструктуру й надавати програмне забезпечення через хмару, незалежно від операційної системи користувача.

IaaS, PaaS та SaaS – це моделі надання хмарних сервісів. Їх часто ілюструють у вигляді піраміди з різним рівнем доступу та контролю інформації (Рисунок 1.1). На вершині – кінцевий користувач, співробітник компанії, який працює з власними даними за допомогою готових програм або сервісів із зручним інтерфейсом. Другий рівень – це програми чи сервіси, розгорнуті на технологічній платформі. Основа піраміди – інфраструктура, що охоплює віртуальні сервери, сховища, обчислювальні потужності й мережі.

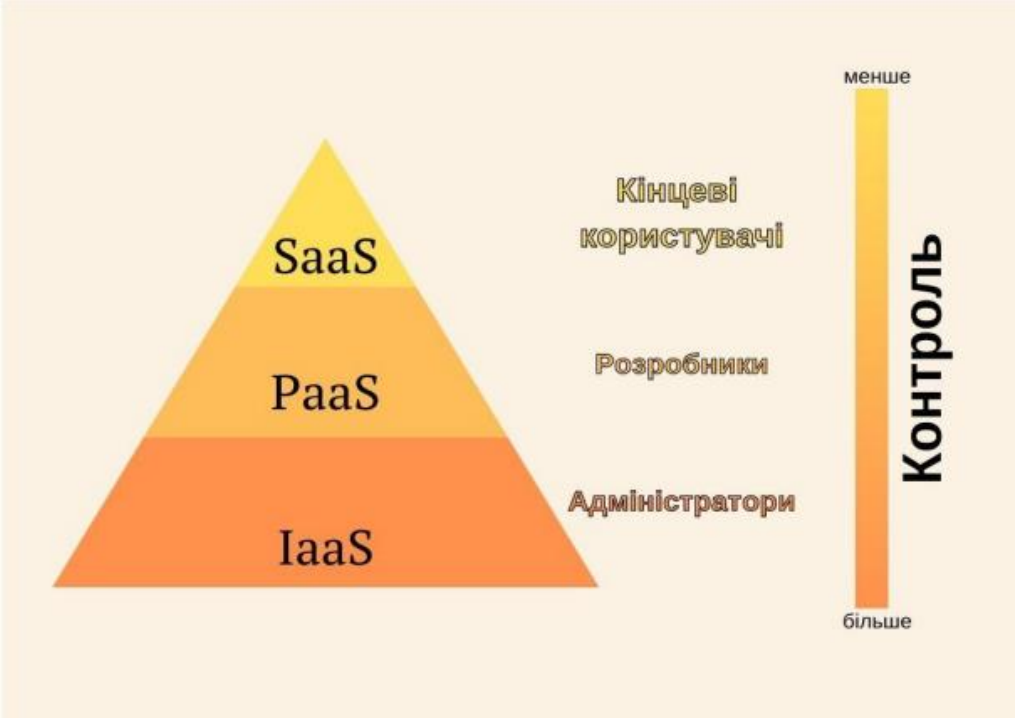


Рисунок 1.1 – Модель хмарних сервісів

SaaS (Software as a Service) — це програмне забезпечення, яке розміщується та управляється централізовано. Зазвичай воно побудоване на мультитенантній архітектурі, де одна версія програми обслуговує всіх клієнтів. За необхідності його можна масштабувати на кілька екземплярів для збереження високої продуктивності у будь-яких умовах. Оплата зазвичай здійснюється за щомісячною або річною передплатою. Постачальник SaaS відповідає за увесь стек програмного забезпечення, а користувач керує лише наданими послугами.

Головна особливість цього рішення полягає в тому, що користувач отримує доступ до створеного провайдером програмного забезпечення через Інтернет. Майже всі додатки SaaS працюють у веб-браузері та не потребують встановлення додаткових компонентів. Це суттєво економить час на впровадження і початок роботи, а також знімає з користувача тягар технічної підтримки й оновлень — вони повністю лежать на провайдері.

Прикладами SaaS-рішень є Google Apps, Dropbox, WebEx, ZenDesk.

PaaS (Platform as a Service) сьогодні є однією з найпоширеніших моделей на ринку, оскільки близько 35% компаній і корпорацій користуються нею згідно зі статистикою. Ця модель передбачає, що провайдер надає розробникам фреймворк, на основі якого вони можуть створювати власні кастомізовані додатки. У такому разі провайдер відповідає за контроль серверів, сховищ даних та мережевої інфраструктури, тоді як розробники зосереджуються на розробці та підтримці програмного забезпечення.

До прикладів цієї моделі можна віднести AWS Elastic Beanstalk, Google App Engine, OpenShift і Magento Commerce Cloud.

IaaS (Infrastructure as a Service). Постачальник хмарних послуг IaaS відповідає за запуск та управління всіма фізичними обчислювальними ресурсами й необхідним програмним забезпеченням для віртуалізації комп’ютерів. Клієнт цієї послуги може розгорнути віртуальні машини у віддалених центрах обробки даних. Хоча вони фізично розміщені деінде, споживач IaaS має повний контроль над конфігурацією та управлінням операційною системою, при цьому базова інфраструктура залишається відповідальністю постачальника хмарних обчислень.

Ця модель стає дедалі популярнішою серед корпорацій та великих IT-компаній, оскільки надає їм доступ до віртуальних серверів, а за розробку та налаштування відповідає сам користувач. Таке рішення ідеально підходить для підприємств, яким потрібна власна серверна інфраструктура з можливістю легкого масштабування та вдосконалення без значних фінансових і часових витрат.

Найпоширенішими прикладами цієї моделі є Amazon Web Services (AWS), Microsoft Azure та Google Cloud Platform (GCP).

Хмарні платформи є потужними інструментами, які суттєво спрощують процес формування корпоративної інфраструктури, підвищуючи її ефективність, масштабованість та гнучкість. У сучасних умовах використання хмарних технологій у 2022 році вже стало нагальною потребою.

1.3 Різновиди хмарних платформ

На сьогодні головними постачальниками хмарних технологій виступають три компанії: AWS (Amazon Web Services), Azure та GCP (Google Cloud Platform). Вони домінують на світовому ринку хмарних послуг, за винятком Китаю, де монопольне становище посідає Alibaba Cloud.

Ці компанії визначають основні тенденції розвитку моделі IaaS, водночас надаючи послуги й за моделями SaaS і PaaS. Загалом рішення від AWS, Azure і GCP переважно відрізняються лише візуальною реалізацією інтерфейсу для користувачів, а з погляду продуктивності вони зазвичай дуже схожі. Другим важливим критерієм вибору провайдера є вартість послуг, де між цими компаніями теж існують певні, хоча й незначні, відмінності.

Таблиця 1.1 – Порівняння Azure, AWS та GCP

Features	AWS	Azure	GCP
Storage Services	Simple Storage Service (S3) Elastic Block Storage (EBS) Elastic File System (EFS) Storage Gateway Snowball Snowball Edge Snowmobile	Blob Storage Queue Storage File Storage Disk Storage Data Lake Store	Cloud Storage Persistent Disk Transfer Appliance Transfer Service
Compute Services	AWS Beanstalk Amazon EC2 Amazon EC2 Auto-Scaling Amazon Elastic Container Registry Amazon Elastic Kubernetes Service Amazon Lightsail AWS Serverless Application Repository VMware Cloud for AWS AWS Batch AWS Fargate AWS Lambda AWS Outposts Elastic Load Balancing	Platform-as-a-service (PaaS) Function-as-a-service (FaaS) Service Fabric Azure Batch Cloud Services Container Instances Batch Azure Container Service (AKS) Virtual Machines Compute Engine Virtual Machine Scale Sets	App Engine Docker Container Registry Instant Groups Compute Engine Graphics Processing Unit (GPU) Knative Kubernetes Functions
Storage Services	Simple Storage Service (S3) Elastic Block Storage (EBS) Elastic File System (EFS) Storage Gateway Snowball	Blob Storage Queue Storage File Storage Disk Storage Data Lake Store	Cloud Storage Persistent Disk Transfer Appliance Transfer Service

Продовження таблиці 1.1 – Порівняння Azure, AWS та GCP

Features	AWS	Azure	GCP
Backup Services	Glacier	Archive Storage Backup Site Recovery	Nearline (frequently accessed data) Coldline (infrequently accessed data)
Caching	Elastic Cache	Redis Cache	Cloud CDN
File Storage	EFS	Azure Files	ZFS and Avere
AI/ML	SageMaker Comprehend Lex Polly Rekognition Machine Learning Translate Transcribe DeepLens Deep Learning AMIs Apache MXNet on AWS TensorFlow on AWS	Machine Learning Azure Bot Service Cognitive Services	Cloud Machine Learning Engine Dialogflow Enterprise Edition Cloud Natural Language Cloud Speech API Cloud Translation API Cloud Video Intelligence Cloud Job Discovery (Private Beta)
Database Services	Aurora RDS DynamoDB ElastiCache Redshift Neptune Database Migration Service	SQL Database Database for MySQL Database for PostgreSQL Data Warehouse Server Stretch Database Cosmos DB Table Storage Redis Cache Data Factory	Cloud SQL Cloud Bigtable Cloud Spanner Cloud Datastore

Продовження таблиці 1.1 – Порівняння Azure, AWS та GCP

Features	AWS	Azure	GCP
Serverless computing	Lambda Serverless Application Repository	Functions	Google Cloud Functions
Networking	Amazon Virtual Private Cloud (VPC)	Azure Virtual Network (VNET)	Cloud Virtual Network
Security	AWS Security Hub	Azure Security Center	Cloud Security Command Center
Strengths	Dominant market position Extensive, mature offerings Support for large organizations Global reach Flexibility and a wider range of services	Second largest provider Integration with Microsoft tools and software Broad feature set Hybrid cloud Support for open source Ideal for startups and developers	Designed for cloud-native businesses Commitment to open source and portability Flexible contracts DevOps expertise Complete container-based model Most cost-efficient
Location	77 availability zones within 24 geographic regions	Presence in 60+ regions across the world	Presence in 24 regions and 73 zones. Available in 200+ countries and territories
Documentation	Best in class	High quality	High quality
DNS Services	Amazon Route 53	Azure Traffic Manager	Cloud DNS
Notifications	Amazon Simple Notification Service (SNS)	Azure Notification Hub	None
Load Balancing	Elastic Load Balancing	Load Balancing for Azure	Cloud Load Balancing
Automation	AWS Opsworks	Azure Automation	Compute Engine Management
Compliance	AWS CloudHSM	Azure Trust Center	Google Cloud Platform Security

Згадані компанії пропонують майже однакову кількість сервісів, проте, проаналізувавши таблицю 1.1, можна помітити, що на даний момент найкращим вибором є AWS. Це зумовлено тим, що рішення від цієї компанії має більше альтернатив і супроводжується найкращою на ринку документацією. При виборі хмарного провайдера одним із ключових факторів є географічне розташування, адже компанія, її клієнти та співробітники можуть бути розміщені в різних куточках світу. У цій категорії перевага належить Google, який завдяки багаторічному досвіду і наявності розгалуженої мережі центрів обробки даних забезпечує оптимальне географічне покриття. Проте, Amazon активно розвиває свою інфраструктуру та будує нові центри, тому ситуація у перспективі може змінитися.

Однією з істотних переваг цих трьох рішень є можливість розгортати програми в різних центрах обробки даних по всьому світу. Обраний регіон впливає на продуктивність програм: чим ближче центр обробки даних до більшості клієнтів, тим меншими будуть затримки у передачі даних. Така гнучкість дозволяє дотримуватись регіональних законодавчих вимог щодо розповсюдження програм, оскільки можна обрати центр відповідно до специфічних правил кожної країни чи регіону.

1.3.1 Хмарна платформа від Google – GCP

Для початку використання GCP потрібно спочатку перейти на офіційну сторінку цього сервісу та увійти в свій обліковий запис Google. Далі слід підключити Billing Account, де реєструється платіжна картка або рахунок, з якого буде стягуватися плата за надані послуги.

Після завершення реєстрації користувач отримує повноцінний доступ до всього спектра послуг, які пропонує провайдер хмарних технологій. Важливою особливістю GCP є те, що на початковому етапі кожен користувач отримує бонус у розмірі 300 доларів, які вже зараховані на рахунок. Це дає змогу вільно

розпоряджатися наданими коштами, обираючи потрібні сервіси для тестування та експериментів на платформі.

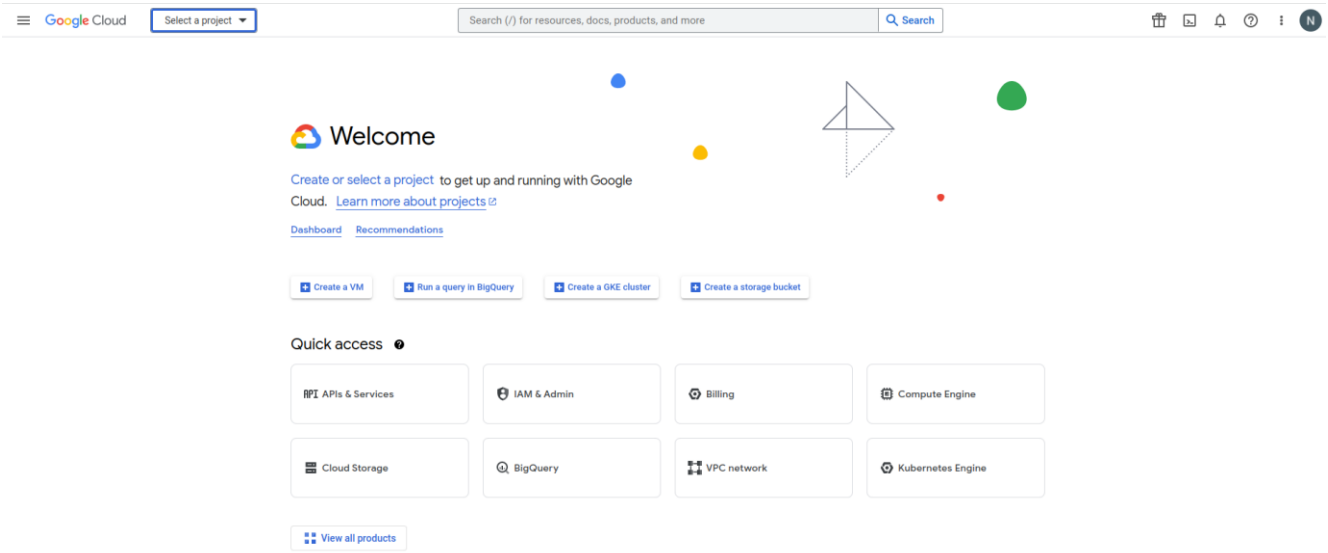


Рисунок 1.2 – Основний інтерфейс платформи GCP

На рисунку 1.2 зображено головну сторінку GCP, де можна вибрати потрібний сервіс із переліку послуг, які надає провайдер хмарних технологій. Повний список доступних сервісів GCP наведено у таблиці 1.1.

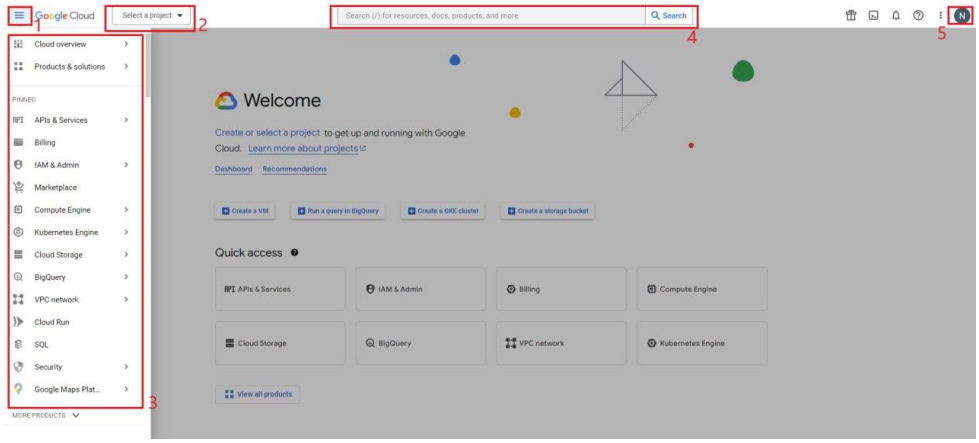


Рисунок 1.3 – Головне меню GCP із позначеними ключовими елементами

На рисунку 1.3 продемонстровано ключові елементи головної сторінки GCP. У верхньому лівому куті міститься кнопка спливаючого меню (позначена цифрою 1), через яку можна відкрити головний список доступних сервісів.

Головне меню, що містить усі послуги GCP, позначене цифрою 3. На рисунку показано лише першу частину цього списку, але насправді він значно ширший: багато пунктів розгортаються, надаючи доступ до більшого спектра сервісів.

У великих компаніях нередко працюють над кількома проектами одночасно. Для цього в GCP існує можливість розділяти та обирати потрібний проект. Список проектів, за допомогою якого можна це зробити, позначений цифрою 2 на рисунку 1.3.

Для пришвидшення пошуку потрібних ресурсів передбачено пошуковий рядок, позначений цифрою 4. Знаючи назву потрібного сервісу, користувач може оперативно його знайти.

Оскільки GCP працює через Google-акаунт, користувач може перемикатися між різними обліковими записами безпосередньо на веб-сторінці. Для цього потрібно скористатися розташованою у верхньому правому куті (позначено цифрою 5) опцією входу до відповідного акаунта.

1.3.2 Хмарна платформа від Microsoft – Azure

Azure – це глобальна хмарна платформа, доступна у багатьох регіонах по всьому світу. Під час налаштування служби, застосунку або віртуальної машини в Azure необхідно обрати регіон, який відповідає певному центру обробки даних, де буде виконуватись ваша програма.

Портал Azure – це веб-застосунок, за допомогою якого можна створювати, видаляти та керувати ресурсами й службами в Azure. До порталу можна отримати доступ за адресою portal.azure.com. Він пропонує настроювану панель моніторингу та інструменти для ефективного управління ресурсами Azure.

Ресурси Azure – це окремі елементи, такі як обчислювальні потужності, мережеві служби, інструменти для обробки даних чи хостингу застосунків,

розгорнуті в межах підписки Azure. Прикладами ресурсів можуть бути віртуальні машини, облікові записи сховищ або бази даних SQL. Часто служби Azure складаються із сукупності взаємопов'язаних ресурсів. Наприклад, віртуальна машина Azure може охоплювати саму машину, обліковий запис зберігання, мережевий адаптер і загальнодоступну IP-адресу. Ці ресурси можна створювати, видаляти та адмініструвати окремо або групами. Подальша інформація про ресурси Azure буде розглянута у цьому посібнику.[1]

1.3.3 Хмарна платформа від Amazon – AWS

Amazon Web Services (AWS) – це всеосяжна хмарна платформа, яка містить в собі як інфраструктуру як послугу (IaaS), так і платформу як послугу (PaaS). Сервіси AWS забезпечують масштабовані можливості для обчислень, зберігання даних, керування базами даних, аналітики та багато інших напрямів.[2]

AWS утримує лідерські позиції серед провайдерів публічних хмарних послуг, охоплюючи широкий спектр інфраструктурних технологій — від обчислювальних ресурсів, сховищ даних та баз даних до передових інновацій, таких як машинне навчання, штучний інтелект та Інтернет речей. Завдяки сервісам AWS компанії можуть швидко, просто та економно перенести свої існуючі додатки в хмарне середовище й реалізувати будь-які проєкти без додаткових фінансових вкладень.

AWS також випереджає конкурентів за кількістю та різноманітністю функцій, які вона пропонує у своїх сервісах. Наприклад, клієнти можуть обирати серед численних баз даних, кожна з яких спеціально розроблена для певного типу застосунків, що дозволяє підібрати оптимальний інструмент для ефективної та економної роботи.[3]

[1] https://learn.microsoft.com/ru-ru/azure/guides/operations/azure-operations-guide

[2] https://aws.amazon.com/ru/getting-started/

[3] https://aws.amazon.com/ru/what-is-aws/?nc1=f_cc

Рисунок 1.4 – Карта географічного розташування дата-центрів AWS

AWS володіє найбільшою хмарною інфраструктурою у світі. Як показано на рисунку 1.4, що відображає карту розташування дата-центрів AWS, їхні центри присутні на всіх континентах, охоплюючи значні площі. Така модель розміщення регіонів та зон доступності була рекомендована компанією Gartner, яка спеціалізується на запуску корпоративних застосунків із високими вимогами до доступності.[4]

1.4 Порівняльний аналіз сучасних хмарних платформ

Загалом, хмарні сервіси відкривають безпрецедентний потенціал для підвищення ефективності бізнесу та зростання прибутку. Для аналізу було обрано три найбільші телекомунікаційні платформи хмарних технологій: Amazon Web Services (AWS), Google Cloud Platform (GCP) та Microsoft Azure.

[4] https://aws.amazon.com/ru/what-is-aws/?nc1=f_cc

Amazon Web Services – це хмарна платформа, яка дає змогу створювати бізнес-рішення з використанням інтегрованих веб-сервісів. AWS пропонує широкий набір послуг моделей IaaS та PaaS. Зокрема:

– Elastic Cloud Compute (EC2) – веб-сервіс, що надає хмарні обчислювальні ресурси.

– Simple Storage Service (S3) – хмарне сховище даних, яке забезпечує зберігання й отримання інформації будь-якого обсягу.

– Relational Database Service (RDS) – сервіс реляційних баз даних.

AWS працює з 2006 року, що дає їй статус «першого лідера у сфері публічних хмарних обчислень». Вона стала піонером, запропонувавши інфраструктуру як послугу в 2008 році. Компанія продовжує запускати нові сервіси та розвивати власний обчислювальний стек, прагнучи зробити його ще ефективнішим.

AWS надає широкі можливості для адміністраторів за допомогою веб-клієнта, через який користувачі можуть виконувати низку завдань, включно зі створенням та аудитом ключів шифрування. Платформа також дозволяє налаштовувати інфраструктуру згідно з вимогами, і це зазвичай значно дешевше, ніж розгортання власних рішень на місці.

Microsoft Azure (часто просто Azure) — це хмарна платформа, орієнтована на розробників хмарних застосунків. Вона надає послуги SaaS, PaaS, IaaS, а також підтримує велику кількість мов програмування, інструментів та фреймворків. Перша версія, відома як Windows Azure, вийшла у лютому 2010 року, а у березні 2014 року платформу перейменували на Microsoft Azure. Azure застосовує масштабну віртуалізацію у глобальних дата-центрах і пропонує понад 100 сервісів, зокрема:

– Microsoft Azure PaaS – повноцінне середовище для розробки та розгортання застосунків у хмарі.

– Microsoft Azure Storage – сховище для даних у вигляді Tables, Blobs, Queues.

– Azure Cosmos DB – база даних типу NoSQL.

Керування платформою здійснюється через веб-клієнт, а оплата стягується лише за фактично використані ресурси.

Google Cloud Platform (GCP) — хмарна платформа для розробки та розгортання додатків у керованих Google дата-центрах. Вона підтримує моделі PaaS, IaaS і безсерверні обчислювальні середовища.

У 2011 році платформа GCP стала доступною для користувачів. Вона реалізує автоматичне масштабування: коли зростає кількість запитів до застосунків, система автоматично виділяє додаткові ресурси. Серед найпопулярніших рішень GCP варто виділити:

– App Engine – сервіс для хостингу сайтів та веб-застосунків на серверах Google.

– Cloud Storage – служба для зберігання й отримання файлів через REST-запити.

– Cloud Database – NoSQL база даних для веб- і мобільних застосунків.

Доступ до GCP здійснюється через веб-клієнт. За додаткове зберігання, використання ресурсів та пропускну здатність стягується плата.

Таким чином, для вибору платформи необхідно виконати поглиблений аналіз. Розглянемо розташування дата-центрів. AWS була першою платформою, тому в неї було більше часу для розширення мережі по всьому світу. Azure та GCP теж активно нарощують інфраструктуру, але на цей момент зона покриття AWS є ширшою.

Якщо поглянути на цифри:

– AWS має 66 зон доступності та ще 12 готуються до запуску.

– Azure охоплює 54 регіони по всьому світу та доступна в 140 країнах.

– Google Cloud Platform доступна у 20 регіонах і планує ще три.

Наступним суттєвим фактором є частка ринку та темпи зростання хмарних провайдерів. Збільшення кількості клієнтів дає змогу покращувати комунікацію з питань використання платформи, а темпи зростання свідчать про її актуальність у майбутньому.

Згідно з квартальними звітами про доходи за 2021 рік (Рисунок 1.2), Azure випереджає конкурентів настільки, що дві інші платформи заробляють менше.

Незважаючи на те, що Amazon AWS була першою та продовжує активно зростати, в останні роки Microsoft Azure випереджає суперників за доходами від хмарних послуг. Так, Azure зафіксувала дохід у розмірі 17,7 млрд доларів США (приріст на 50% порівняно з попереднім кварталом), тоді як Amazon повідомила про 13,5 млрд доларів США (приріст 32%). У Google цей показник становив 4,05 млрд доларів США, що значно скромніше за результати конкурентів.

Рисунок 1.5 – Прибуток провідних платформ

Відповідно до аналізу звітів Canalys за 2022 рік (Рисунок 1.3), частка AWS на світовому ринку становить 33%, Azure – 21%, а Google Cloud – 8%.

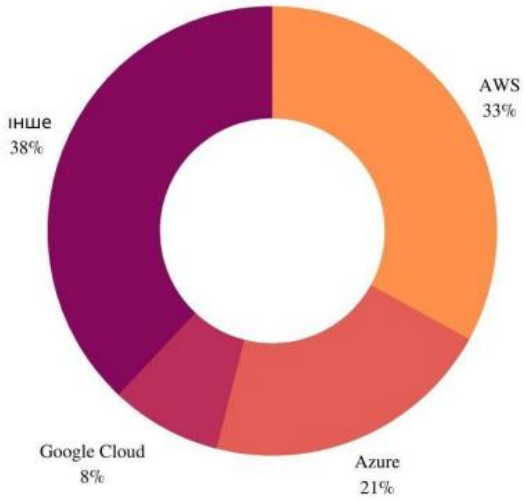


Рисунок 1.6 – Частка ринку платформ

Оскільки AWS є найдовше присутнім на ринку хмарних технологій, він має відчутну перевагу у вигляді ширшої підтримки спільноти та більш розвиненої клієнтської бази. Серед відомих клієнтів AWS — Netflix, Airbnb, Unilever, BMW, Samsung, MI, Zynga тощо.

Azure з часом також здобуває значний перелік відомих клієнтів. Нині майже 80% компаній зі списку Fortune 500 користуються послугами Azure. До ключових клієнтів належать Johnson Controls, Polycom, Fujifilm, HP, Honeywell, Apple та інші.

Google Cloud користується тією ж інфраструктурою, що й пошукова система Google та YouTube. Це приваблює чимало великих компаній, які довіряють надійності платформи. Серед основних клієнтів Google Cloud — HSBC, PayPal, 20th Century Fox, Bloomberg, Dominos тощо.

Усі ці хмарні постачальники пропонують широкий спектр хмарних обчислювальних послуг, необхідних для основних бізнес-потреб. Основні відмінності полягають у кількості доступних сервісів. Відтак, переходячи до детальнішого аналізу Azure, AWS та Google Cloud, варто звернути увагу на їхні пропозиції послуг.

Завдяки перевазі у п’ять років розвитку AWS пропонує найзріліші та найфункціональніші обчислювальні послуги — близько 200+. Azure, у свою чергу, надає понад 100+ послуг, тоді як Google Cloud наздоганяє своїх конкурентів, пропонуючи більш ніж 60 сервісів.

Нижче у таблицях 1.2, 1.3, 1.4 та 1.5 наведено порівняння сервісів, що належать до категорій обчислень, баз даних, зберігання даних та мережевих рішень для AWS, Azure та GCP.

Таблиця 1.2 – Обчислювальні сервіси

Сервіси	**Azure**	**AWS**	**GCP**
IaaS	Virtual Machines	Amazon Elastic Compute Cloud	Google Compute Engine
PaaS	App Service and Cloud Services	AWS Elastic Beanstalk	Google App Engine
Контейнери	Azure Kubernetes Service (AKS)	Amazon Elastic Compute Cloud Container Service	Google Kubernetes Engine
Безсерверні функції	Azure Functions	AWS Lambda	Google Cloud Functions

Таблиця 1.3 – Сервіси баз даних

Категорія	**Azure**	**AWS**	**GCP**
RDBMS	SQL Database	Amazon Relational Database Service (RDS)	Google Cloud SQL
NoSQL: Ключ–Значення	Table Storage	Amazon DynamoDB	Google Cloud Datastore, Google Cloud Bigtable
NoSQL: Індекс	Azure Cosmos DB	Amazon SimpleDB	Google Cloud Datastore

Таблиця 1.4 – Сервіси сховищ

Сервіс	**Azure**	**AWS**	**GCP**
Object Storage	Blob Storage	Amazon Simple Storage Service	Google Cloud Storage
Virtual Server Disks	Managed Disks	Amazon Elastic Block Store	Google Compute Engine Persistent Disks
Cold Storage	Azure Archive Blob Storage	Amazon Glacier	Google Cloud Storage Nearline
File Storage	Azure File Storage	Amazon Elastic File	ZFS/Avere

Таблиця 1.5 – Сервіси мереж

Сервіс	**AWS**	**Azure**	**GCP**
Віртуальна мережа	Virtual Networks (VNets)	Amazon Virtual Private Cloud (VPC)	Virtual Private Cloud
Еластичний балансувальник навантаження	Load Balancer	Elastic Load Balancer	Google Cloud Load Balancing
Peering	ExpressRoute	Direct Connect	Google Cloud Interconnect
DNS	Azure DNS	Amazon Route 53	Google Cloud DNS

Зараз між трьома провайдерами хмарних послуг триває доволі інтенсивна конкуренція. Вони орієнтуються на останні тенденції та запити клієнтів, тому всі троє вже пропонують подібні послуги та ймовірно розширятимуть їх надалі. Поки що постачальники не надто активно використовують гібридні та мультихмарні рішення, але надають клієнтам сучасні інструменти для більшої гнучкості.

AWS пропонує:

– AWS Snowball
– AWS Snowcone
– AWS Outposts
– AWS Local Zones
– VMware Cloud on AWS
– AWS Wavelength
– Amazon ECS Anywhere
– Amazon EKS Anywhere

Azure надає:

– Azure Arc
– Azure Backup
– Azure Active Directory
– Azure Security Center
– Azure Blob Storage
– Azure Stack
– Azure Sentinel

Google Cloud включає:

– Anthos
– Traffic Director
– Looker
– Cloud Build
– Operations
– Cloud Run for Anthos

Ще одним важливим аспектом порівняння є робота вбудованих архіваторів, оскільки сучасні компанії зберігають гігабайти, а часом і терабайти даних. Для оцінки продуктивності архіваторів було створено чисті акаунти в Azure, AWS та GCP з використанням однакового обсягу даних та регіонів, розташованих

максимально близько один до одного. На діаграмі (Рисунок 1.7) відображено загальний рейтинг у системі MIPS.

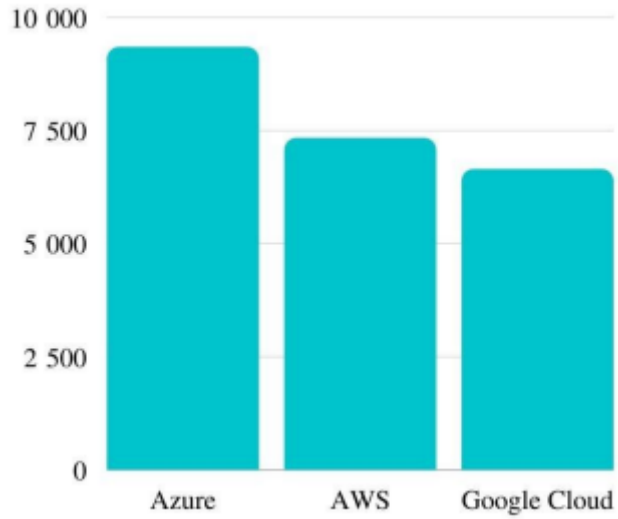


Рисунок 1.7 – Зведений рейтинг MIPS провідних хмарних платформ

Для оцінки продуктивності дискової підсистеми було вирішено застосувати CrystalDiskMark (Рисунок 1.8). Тестування виконували за такими параметрами: послідовне читання, читання блоками по 4K, читання блоками по 4K із чергою 32, послідовний запис, запис блоками по 4K та запис блоками по 4K із чергою 32. Результати тестів наведено на Рисунку 1.8.

Рисунок 1.8 – Показники швидкодії найбільших платформ

Azure та AWS працюють досить швидко. Особливо Azure помітно перевершує конкурентів у процесах, що пов’язані з блоковими операціями, тоді як Google Cloud значно відстає, ймовірно через обмеження на рівні 25 Мб/с.

З огляду на вартість ресурсів, такі результати не дивують. Найнижчу ціну пропонує Google Cloud, особливо якщо врахувати 30% знижку за умови тривалого використання віртуальної машини протягом більшої частини місяця. З урахуванням знижки вартість становитиме приблизно 305$.

Друге місце за ціною посідає AWS. Оскільки сховище тут доведеться купувати окремо, загальна вартість сягатиме близько 346$.

Ціна Azure формально найвища (приблизно 356$). Проте вона не набагато перевищує вартість AWS. До того ж, при реєстрації платформа надає бонус у розмірі 200$, що знижує фактичні витрати та робить ціну на Azure більш збалансованою у довгостроковій перспективі.

Підсумовуючи, можна визначити, для кого підходить кожна платформа. Якщо компанія обмежена в бюджеті та планує активно інтегруватися з Google-сервісами, а великий вибір послуг не є критичним, найкращим вибором стане Google Cloud Platform. Якщо ж головним пріоритетом є широкий спектр

доступних сервісів, тоді варто звернутися до Amazon Web Services. А якщо вам потрібні потужні обчислювальні ресурси – однозначно Microsoft Azure.

Саме тому, у подальшому дослідженні доцільно використовувати Azure. Ця платформа демонструє хороші темпи зростання, що гарантує її актуальність у найближчому майбутньому, а наданий бонус при реєстрації частково компенсує вищу вартість.

РОЗДІЛ 2 ОЗНАЙОМЛЕННЯ З WEB-СТОРІНКОЮ AWS

2.1 Головне меню

Після створення облікового запису AWS та авторизації користувач потрапляє на інформаційну панель консолі, яка є відправною точкою для роботи з різноманітними сервісами AWS. У верхній частині панелі розташована панель навігації, а в основній частині сторінки знаходиться низка віджетів, які можна налаштовувати та змінювати за потреби.[5]

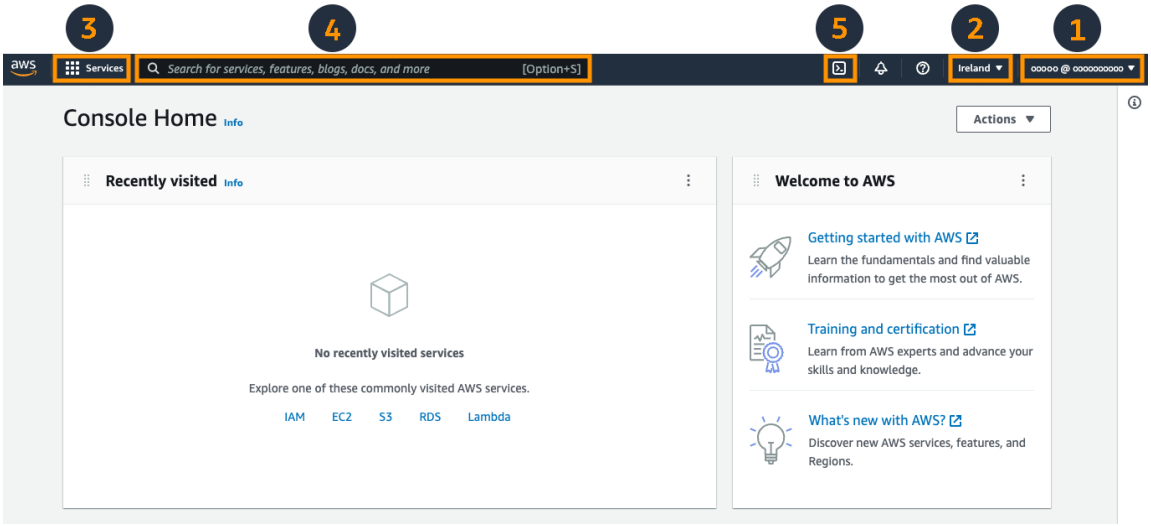


Рисунок 2.1 – Головне меню AWS

На рисунку 2.1 можна виділити п’ять ключових елементів керування на панелі навігації:

1. Інформація про обліковий запис
2. Вибір регіону
3. Селектор послуг
4. Поле пошуку
5. AWS CloudShell

[5] https://aws.amazon.com/ru/getting-started/hands-on/getting-started-with-aws-management-console/?pg=gs&sec=gtkaws#

2.2 Інформація про обліковий запис

Перше виділене меню, з яким знайомиться користувач, містить відомості та посилання щодо його облікового запису. У ньому можна побачити ідентифікатор облікового запису AWS і дані про поточного користувача, який увійшов до консолі. Як правило, користувач доволі часто звертається до цього меню та посилань у ньому.[6]

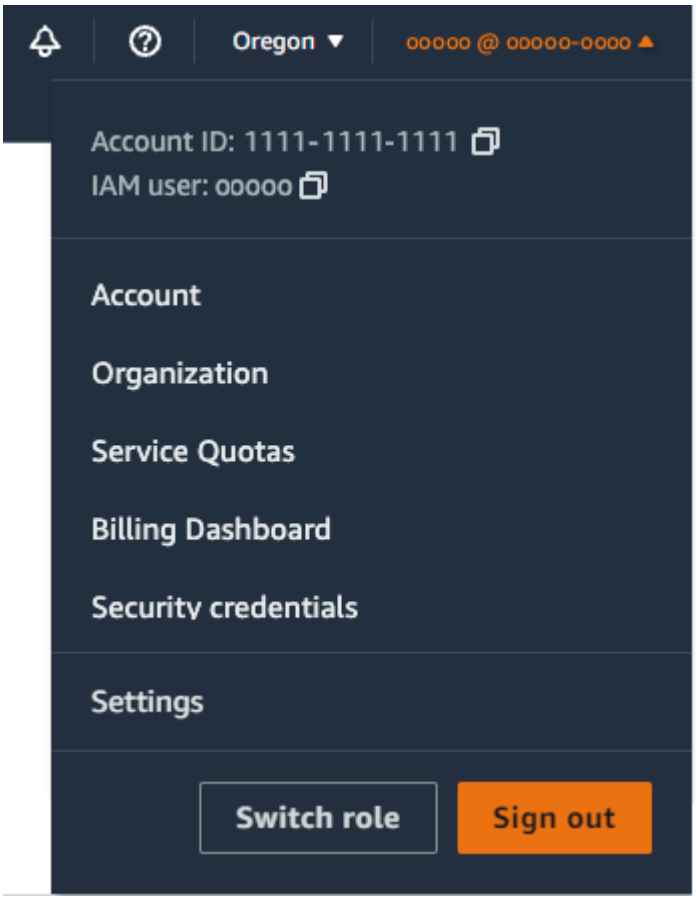


Рисунок 2.2 – Меню відомостей про обліковий запис

Як видно з рисунку 2.2, у меню користувача присутні кілька посилань, якими часто користуються під час взаємодії з різними сервісами AWS. Розділ Account надає розгорнуту інформацію про обліковий запис, включаючи адресу, контактні дані, налаштування платежів тощо.

[6] https://aws.amazon.com/ru/getting-started/hands-on/getting-started-with-aws-management-console/?pg=gs&sec=gtkaws

За посиланням Organization знаходиться служба контролю облікових записів, яка дозволяє об’єднувати кількох користувачів в одну організацію для спрощення робочого процесу.

Service Quotas — це квоти (також відомі як обмеження) у службах AWS, які визначають максимально допустимі значення для ресурсів, дій та елементів вашого облікового запису. Після створення нового облікового запису діють стандартні ліміти, наприклад, призначення п’яти еластичних IP-адрес. За потреби ці обмеження можна збільшити, надіславши відповідний запит до служби підтримки.[7]

При взаємодії з будь-яким бізнес-сервісом завжди важливо враховувати фінансовий аспект. Для полегшення оцінки витрат у меню користувача передбачено розділ Billing Dashboard, де можна ознайомитися зі сторінкою платіжної консолі AWS та отримати загальне уявлення про поточні витрати на AWS.

Оскільки послугами AWS користуються різні компанії, а безпека для більшості з них надзвичайно важлива, у меню користувача доступне посилання Security Credentials. Воно дозволяє перейти до налаштувань безпеки у консолі IAM, де можна змінити пароль, увімкнути двофакторну автентифікацію, створити ключі AWS API тощо.

Для зручності під час роботи з сервісами AWS у меню користувача також є посилання на налаштування. Тут можна керувати глобальними параметрами консолі, зокрема вибором мови та регіону за замовчуванням, а також налаштуваннями, які допомагають оптимізувати відображення консолі.[8]

2.3 Регіони AWS, зони доступності та локальні зони.

В AWS використовується поняття "регіон" – це фізичне місце розташування, де функціонують центри обробки та передачі даних. Група взаємопов’язаних центрів обробки даних, об’єднаних між собою, називається "зоною доступності".

[7] https://aws.amazon.com/ru/getting-started/hands-on/getting-started-with-aws-management-console/?pg=gs&sec=gtkaws

[8] https://aws.amazon.com/ru/getting-started/hands-on/getting-started-with-aws-management-console/?pg=gs&sec=gtkaws

Кожен регіон AWS містить щонайменше три ізольовані й фізично відокремлені зони доступності в межах однієї географічної області. На відміну від інших постачальників хмарних послуг, які часто вважають регіон окремим дата-центром, підхід AWS з використанням кількох зон доступності в одному регіоні має низку переваг:

– Кожна зона доступності має власні системи живлення та охолодження, надійну охорону та підключення до резервних наднизьколатентних мереж.

– Клієнти, для яких важлива безперебійна робота додатків, можуть розробляти та розгортати свої рішення одразу в декількох зонах доступності, що підвищує стійкість до можливих збоїв.

– Інфраструктура регіонів AWS відповідає найвищим вимогам безпеки, галузевим стандартам і нормам захисту даних.

AWS має найбільш розгалужену глобальну інфраструктуру серед провайдерів хмарних рішень та активно розширює її, забезпечуючи доступність у різних куточках світу. Завдяки цьому клієнти можуть обирати серед багатьох географічних регіонів, включно з Північною та Південною Америкою, Європою, Китаєм, Азійсько-Тихоокеанським регіоном, Південною Африкою та Близьким Сходом.

Зона доступності — це один або декілька центрів обробки даних, обладнаних резервними джерелами живлення, оптимізованою мережевою конфігурацією та підключенням у межах регіону AWS. Така архітектура забезпечує підвищену доступність, відмовостійкість і можливості масштабування для застосунків та баз даних у робочому середовищі, порівняно з розгортанням на одному ЦОД.

Усі зони доступності в регіоні AWS поєднані у високошвидкісну резервовану мережу на базі метроволокон, яка забезпечує велику пропускну здатність та низькі затримки передачі даних між ними. Увесь трафік між зонами доступності проходить у зашифрованому вигляді. Продуктивність мережі настільки висока, що дозволяє здійснювати синхронну реплікацію між зонами доступності.

Завдяки використанню зон доступності можна розподілити компоненти застосунків таким чином, щоб забезпечити їхню стійкість до відмов і безперервний доступ до сервісів. Розташування частин застосунку у різних зонах доступності підвищує рівень захисту даних та дає змогу протидіяти негативним наслідкам відключення електроенергії або природних катастроф, таких як удари блискавки, торнадо чи землетруси.

Кожна зона доступності фізично віддалена на значну відстань від інших зон (декілька кілометрів), але загалом усі вони розміщені в межах 100 км (60 миль) одна від одної.[9]

У AWS також існують локальні зони, завдяки яким сервіси можна розгортати ближче до кінцевого користувача. Ці локальні зони розширюють обраний регіон AWS, розташований географічно недалеко від користувача, та дають змогу запускати додатки, чутливі до затримок, використовуючи такі сервіси, як Amazon Elastic Compute Cloud, Amazon Elastic Block Store, Amazon Virtual Private Cloud, Amazon File Storage та Amazon Elastic Load Balancing.

Локальні зони AWS забезпечують безпечний широкосмуговий канал зв’язку між локальними ресурсами та процесами, що виконуються в регіоні AWS. Завдяки цьому можна безперешкодно підключитися до всіх сервісів регіону через наявні API та інструментарій.[10]

[9] https://aws.amazon.com/ru/about-aws/global-infrastructure/regions_az/

[10] https://aws.amazon.com/ru/about-aws/global-infrastructure/regions_az/

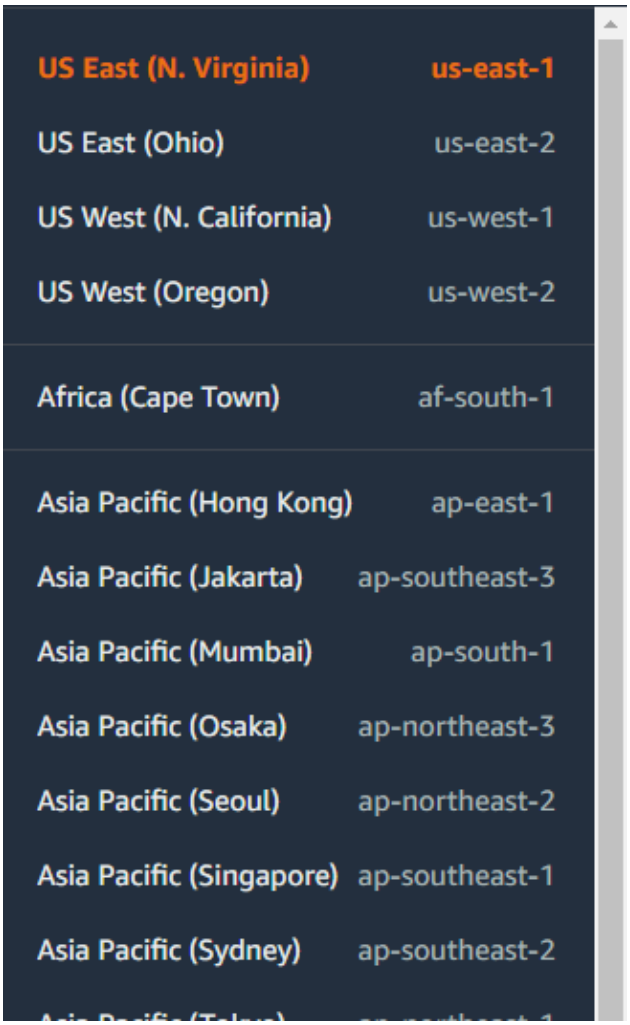


Рисунок 2.3 – Меню регіонів AWS

Щоб працювати в певному регіоні, необхідно його вибрати в головному меню. Регіони обираються у розділі номер 2 на рисунку 2.1, після чого користувач побачить меню з усіма регіонами AWS, як зображено на рисунку 2.3. У цьому меню перелічені всі доступні сервісні центри, де можна розмістити свою інфраструктуру.

2.4 Меню вибору служб AWS

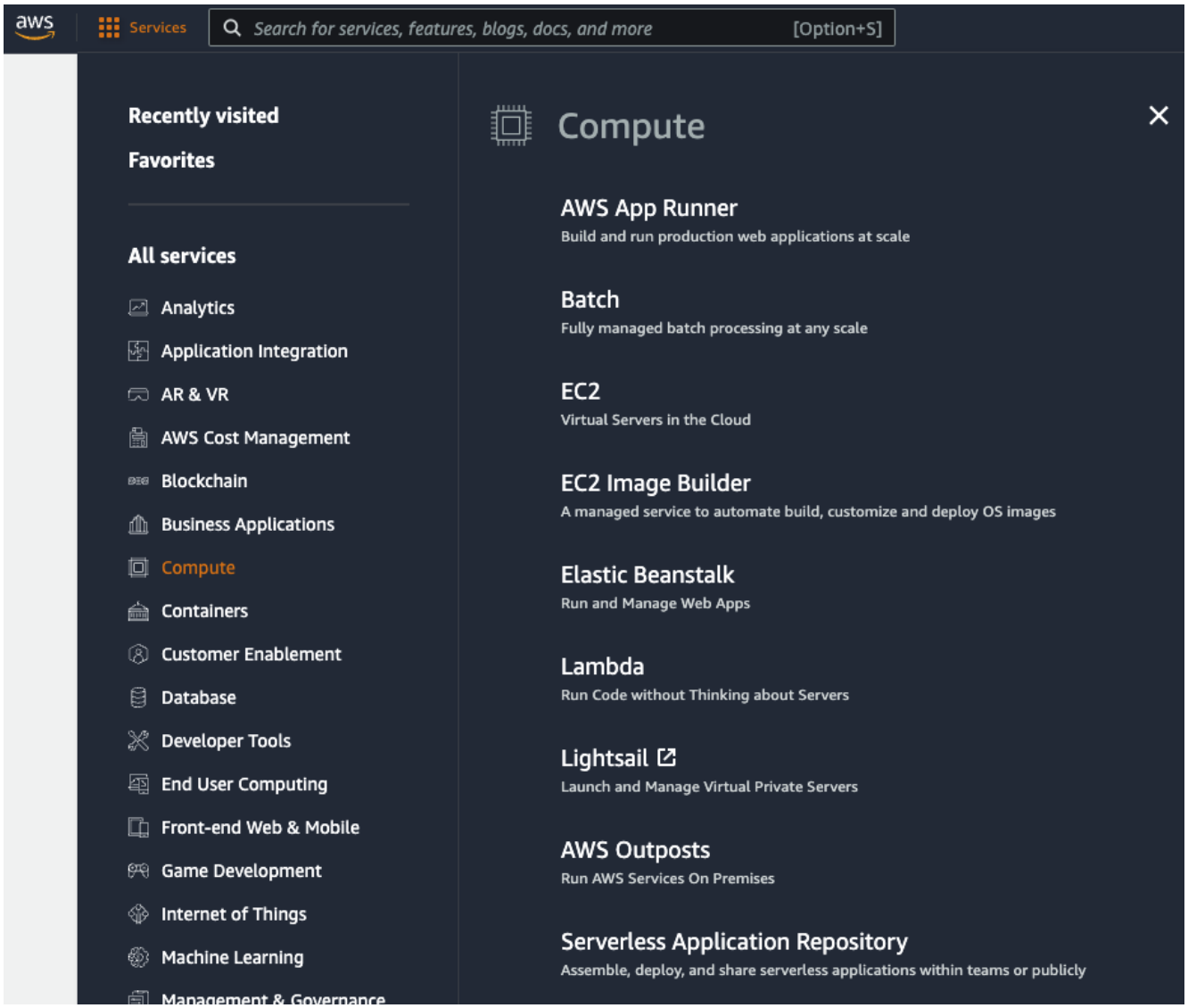


Рисунок 2.4 – Селектор сервісів AWS

На рисунку 2.1 у розділі 3 розміщено селектор послуг, які надає компанія AWS. Цей селектор використовується для переміщення між різними сервісами, організовуючи їх за окремими категоріями для зручності роботи з ними. Приклад вигляду цього меню можна побачити на рисунку 2.6.

РОЗДІЛ 3 ВИКОНАННЯ БАЗОВОГО НАЛАШТУВАННЯ ХМАРИ ЗА ДОПОМОГОЮ WEB-ПОРТАЛУ AWS

3.1 Основна сторінка Elastic Compute Cloud (EC2)

Elastic Compute Cloud (EC2) — це сервіс AWS, що належить до категорії «Інфраструктура як послуга». Він пропонує широкий вибір віртуальних машин, придатних як для загальних, так і для спеціалізованих обчислювальних задач на вимогу. Для початку налаштування EC2 необхідно відкрити головний інтерфейс AWS EC2. Це можна зробити, перейшовши за посиланням, яке розташоване у верхній лівій частині головного меню.

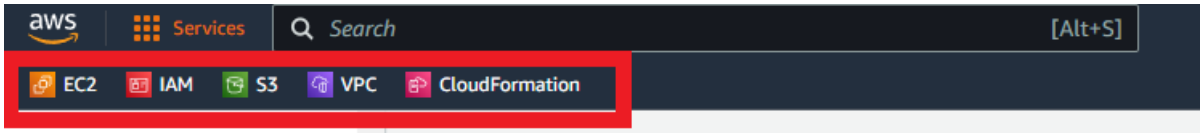


Рисунок 3.1 – Основне меню AWS з посиланнями на ключові сервіси.

Після переходу до EC2 користувач потрапляє на головне меню, яке дозволяє здійснювати налаштування та моніторинг усіх своїх інстансів. Тут доступні різноманітні функції для керування ресурсами, включаючи запуск нових інстансів, зміну їх конфігурації, налаштування параметрів безпеки, а також відстеження продуктивності та використання ресурсів у реальному часі.

3.1.1 Виконання створення EC2 Instance через web-сторінку AWS

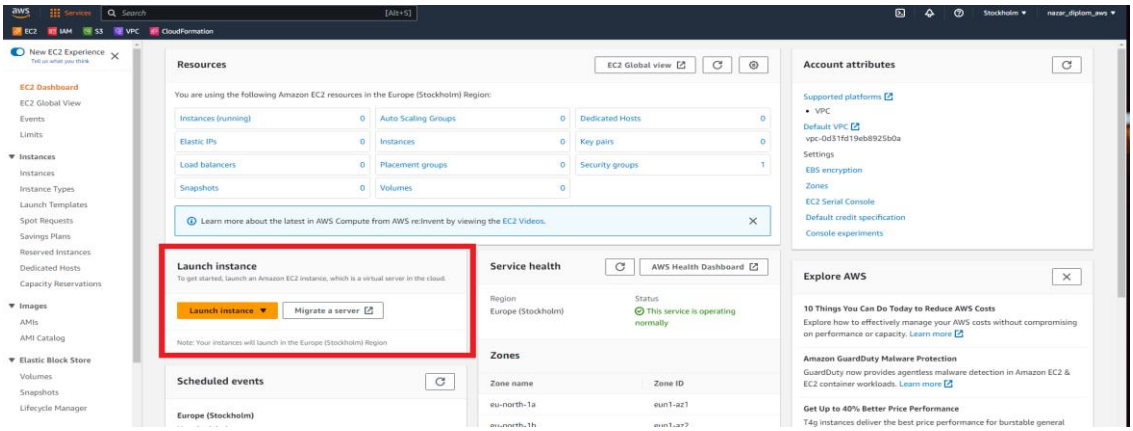


Рисунок 3.2 – Сторінка моніторингу та налаштування інстансів — Перехід до створення нового інстанса.

1) Для створення нового інстанса необхідно перейти до блоку "Launch instance" та натиснути жовту кнопку "Launch instance". Ця кнопка, як показано на рисунку 3.2, дозволяє ініціювати процес створення нового інстанса. Після натискання користувач буде спрямований до покрокового майстра налаштувань, де можна обрати тип інстанса, налаштувати конфігурацію обчислювальних ресурсів, вибрати операційну систему, мережеві параметри та інші важливі опції.

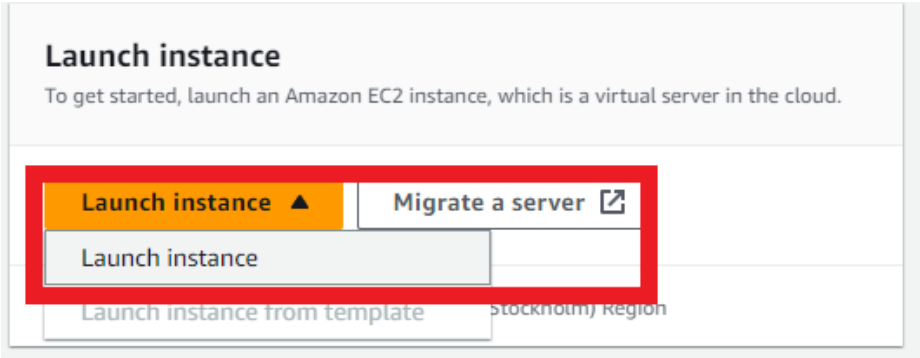


Рисунок 3.3 – Модуль Lauch instance

2) Після натискання цієї кнопки для початкового налаштування слід вибрати єдине доступне меню, яке перенаправляє на сторінку створення нового інстанса. На цій сторінці користувач зможе задати основні параметри нового інстанса,

зокрема вибрати операційну систему, тип віртуальної машини, налаштувати мережеві та безпекові параметри, а також задати інші конфігурації для належної роботи інстанса.

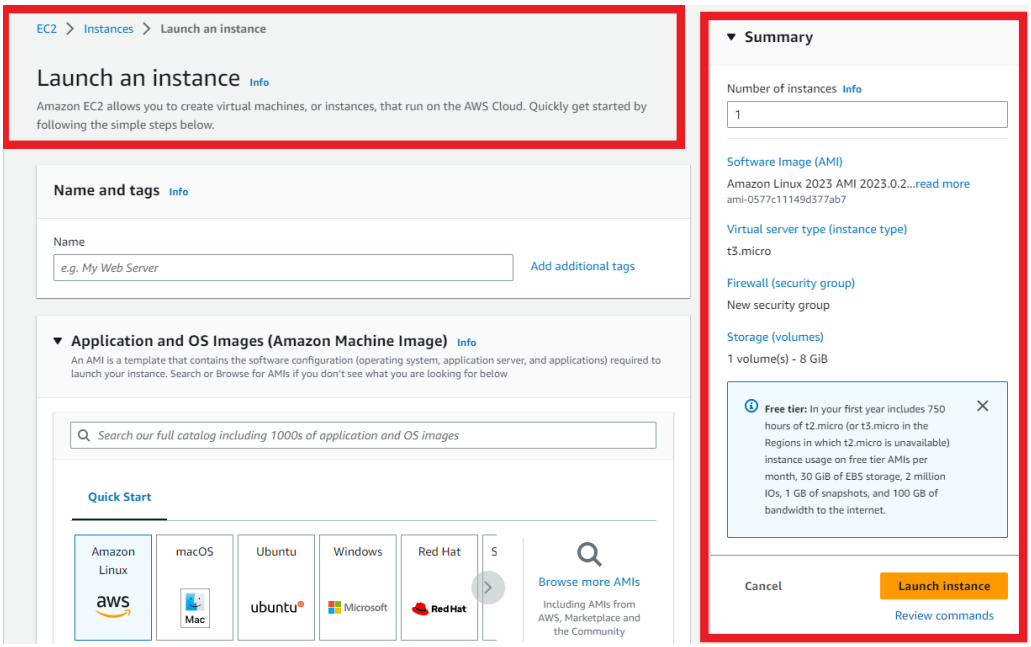


Рисунок 3.4 – Сторінка налаштування instance

3) На цій сторінці з правого боку знаходиться модуль, що відображає зведення всіх основних параметрів для нового інстанса. Ліву частину сторінки займають модулі для налаштування інстанса, де необхідно заповнити та обрати всі базові налаштування, такі як назва інстанса, тип операційної системи, обчислювальні ресурси, мережеві параметри та безпекові налаштування.

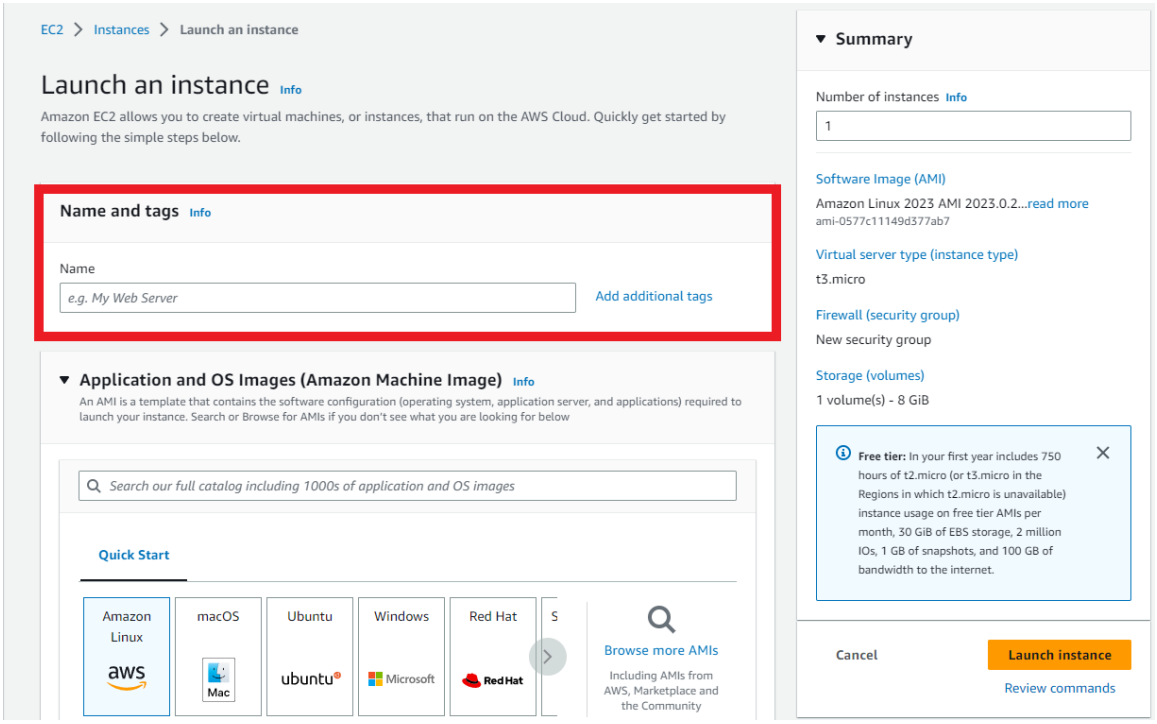


Рисунок 3.5 – Сторінка налаштування instance – модуль створення імені та тегів

4) У першому модулі "Name and tags" потрібно вказати назву інстанса, а також, за необхідності, додати кілька тегів. Це поле не є обов'язковим, однак використання імені замість ідентифікатора (ID) значно спрощує роботу, роблячи ідентифікацію інстансів зручнішою.

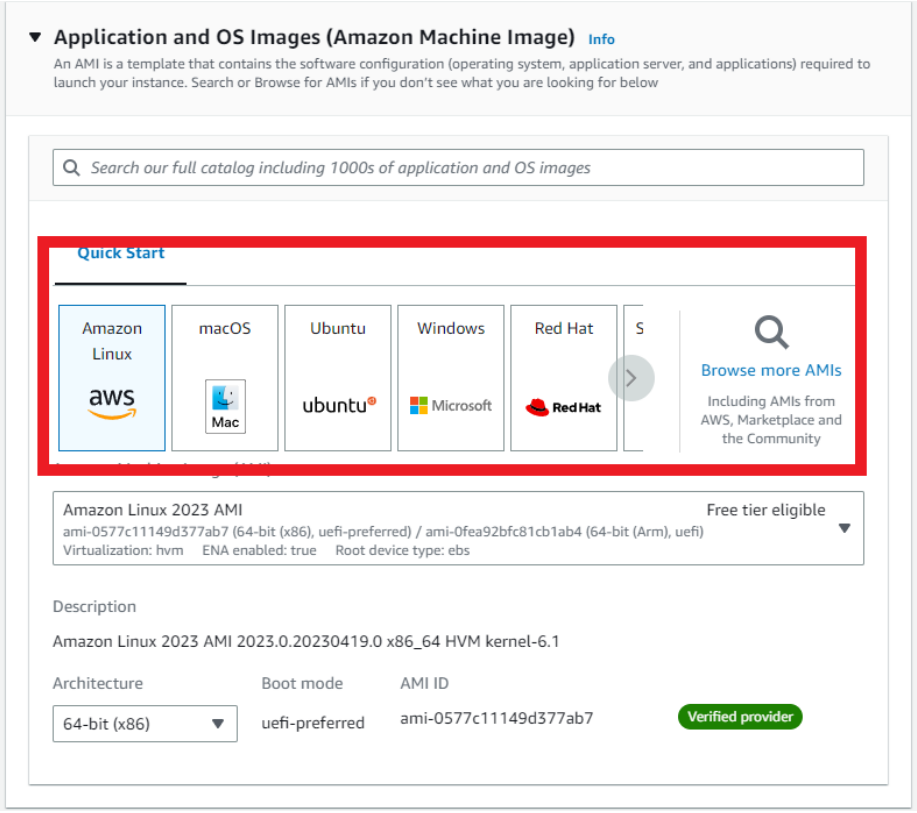


Рисунок 3.6 – модуль Application and OS Images

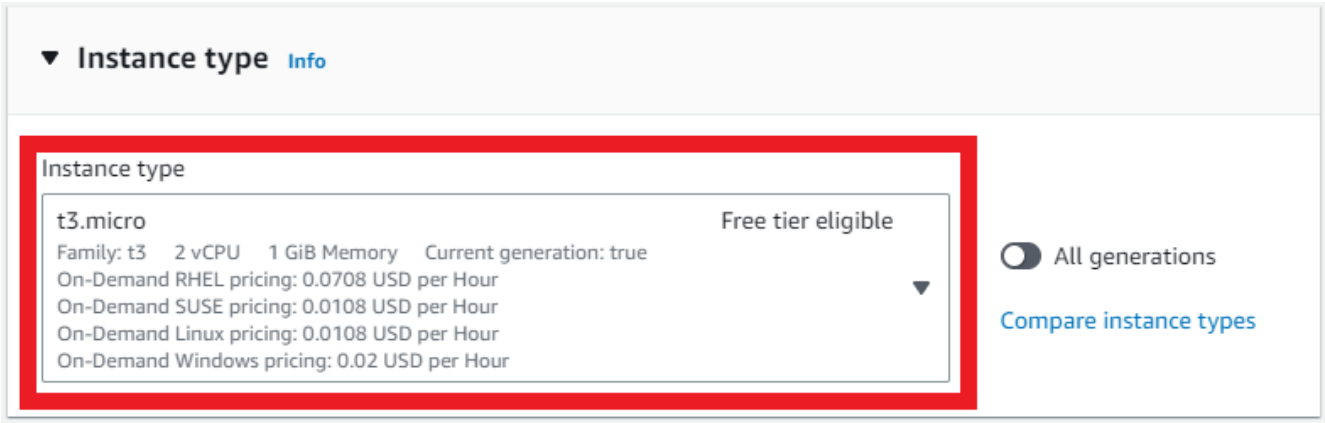


Рисунок 3.7 – модуль Application and OS Images– вибір Instance type

5) У модулі "Application and OS Images" необхідно вибрати операційну систему (Рисунок 3.6), яка буде встановлена на новому інстансі. Це дуже важливий етап, оскільки вибір операційної системи має великий вплив на функціональні можливості майбутнього сервера та його вартість. Вибір ОС визначає, які застосунки можна буде запускати на інстансі, а також який набір інструментів та функцій буде доступний для управління та обслуговування. Крім того, різні

операційні системи мають різні ліцензійні вимоги, що прямо впливає на ціну користування інстансом. Тому на цьому етапі важливо ретельно розглянути всі доступні варіанти та обрати ОС, яка найкраще відповідає потребам проєкту.

6) Не менш важливим є вибір типу інстанса, який буде використовуватися (Рисунок 3.7). Тип інстанса визначає його обчислювальні можливості, такі як кількість віртуальних процесорів, оперативна пам'ять, можливості зберігання даних і підтримка мережевих ресурсів. Від вибраного типу інстанса залежатиме, наскільки ефективно він зможе виконувати поставлені завдання. Це також впливає на вартість обслуговування — чим потужніший тип інстанса, тим дорожче коштуватиме його використання. Для базових налаштувань рекомендується вибрати безкоштовний тип інстанса, який дозволить тестувати і розгортати прості проєкти. Згодом, якщо вимоги до обчислювальних ресурсів зростуть, можна перенести інстанс на більш потужний тип, що забезпечить необхідну продуктивність для розширення або складніших задач. Це забезпечує гнучкість і дає змогу ефективно керувати витратами залежно від реальних потреб.

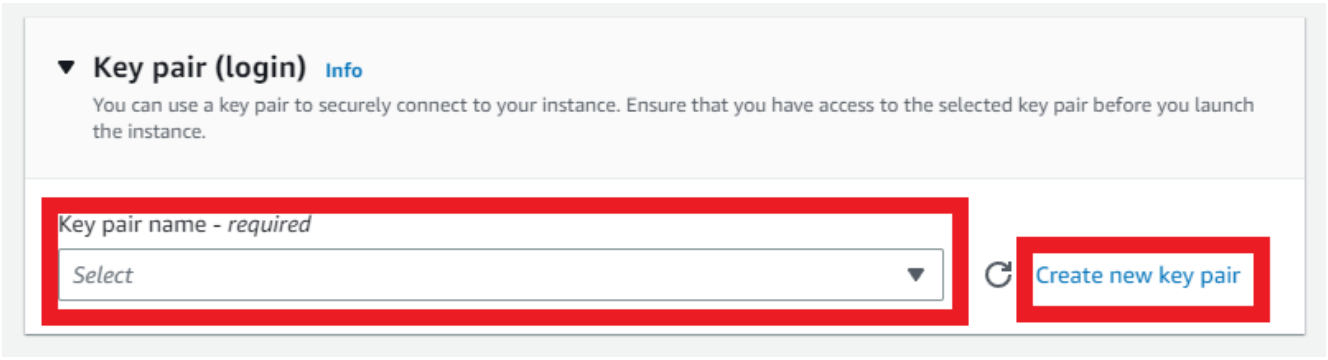


Рисунок 3.8 – модуль Key pair– створення ключа доступу

7) Для забезпечення безпеки та організації доступу до налаштувань інстанса необхідно обрати існуючу або створити нову SSH Key Pair. Ця ключова пара дозволяє встановити безпечне з'єднання з інстансом, забезпечуючи автентифікацію користувача під час доступу до сервера. SSH Key Pair складається з приватного і публічного ключів, і саме приватний ключ зберігається на вашому пристрої для подальшого використання під час підключення до інстанса. Це важливий етап,

оскільки правильна налаштування ключів доступу забезпечує захист сервера від несанкціонованого доступу.

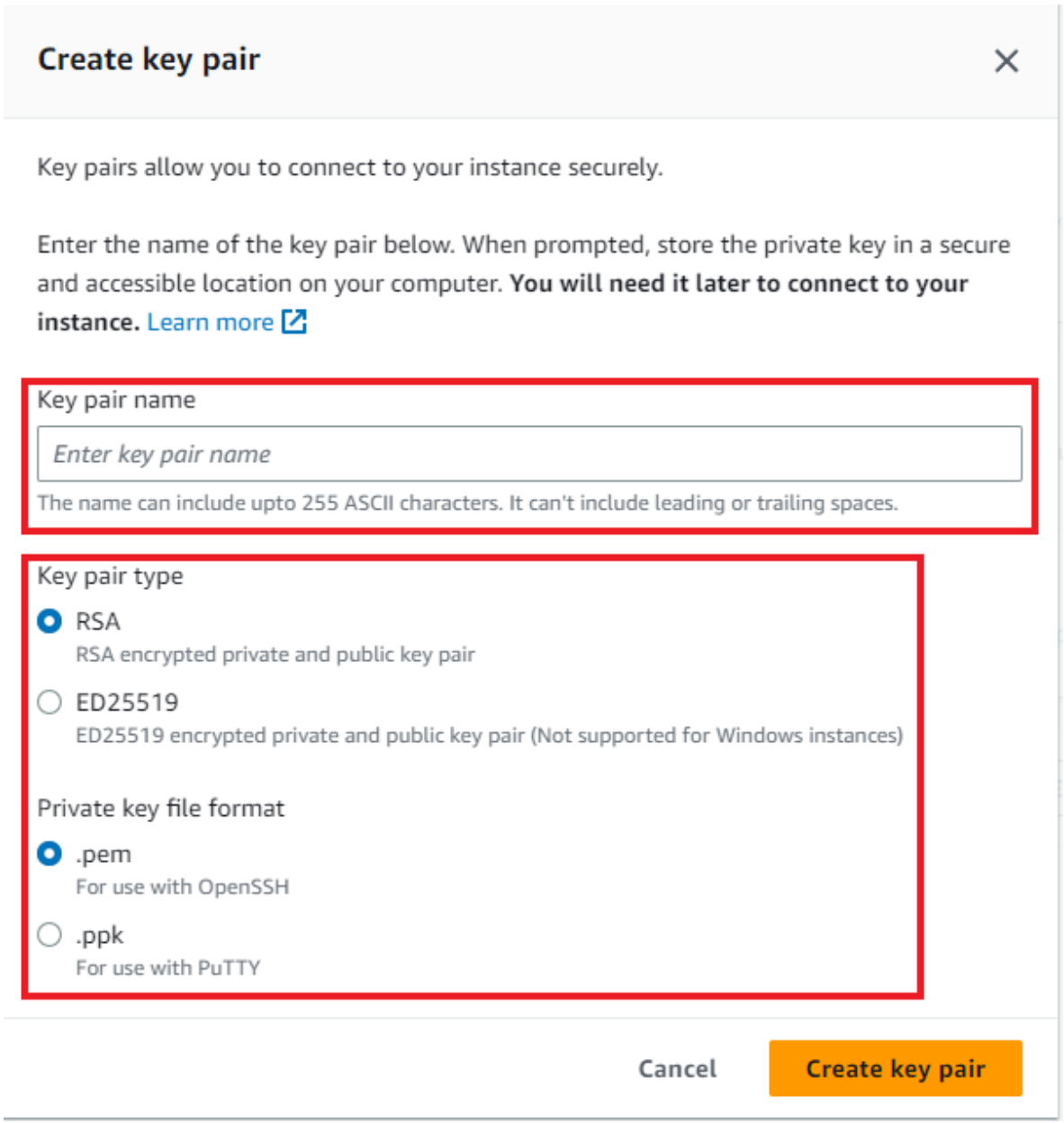


Рисунок 3.8.1 – модуль Key pair– створення ключа доступу

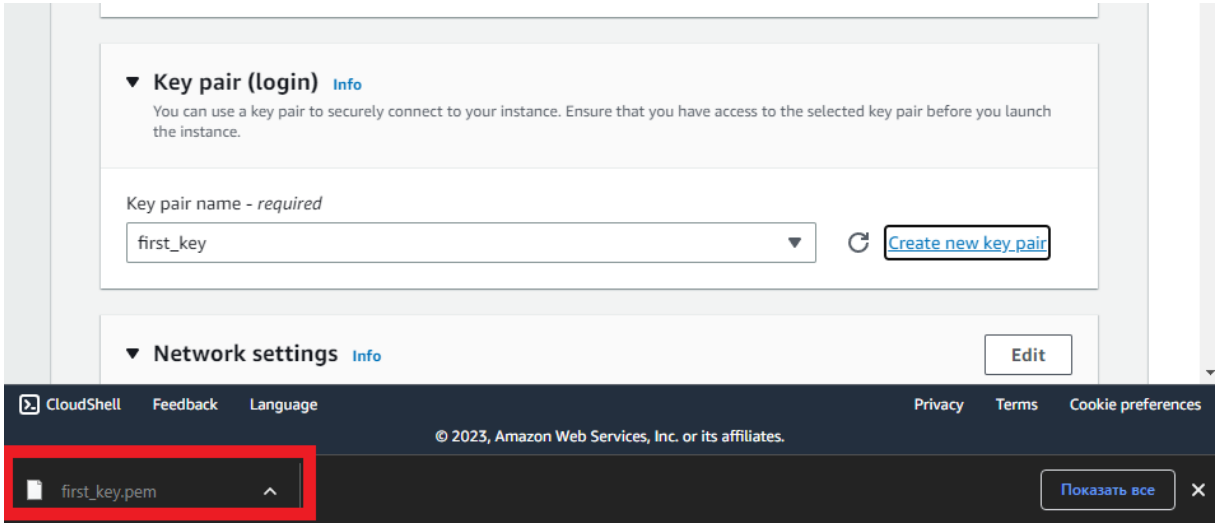


Рисунок 3.8.2 – модуль Key pair– створення ключа доступу

8) Щоб створити SSH ключову пару, потрібно натиснути на кнопку "Create new key pair" (Рисунок 3.8). Після цього з'явиться меню для створення ключа (Рисунок 3.8.1). У цьому меню слід вказати ім'я для нової ключової пари та обрати спосіб підключення до інстанса. У даному випадку було обрано формат ключа ".pem", оскільки доступ до інстанса здійснюватиметься з ПК, що працює під управлінням ОС Linux. Після натискання на кнопку "Create key pair" ключ буде автоматично завантажений на ПК (Рисунок 3.8.2).

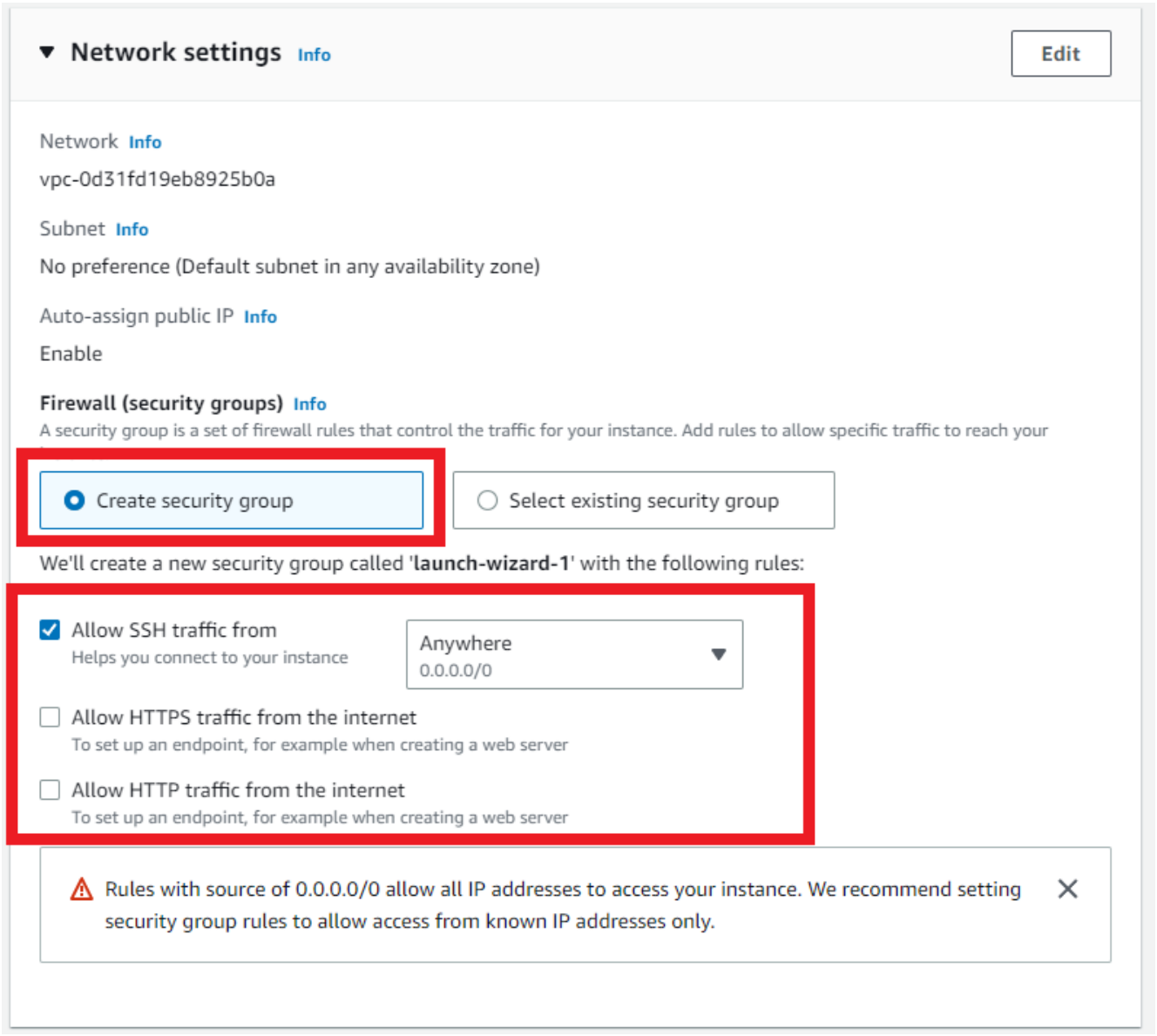


Рисунок 3.9 – модуль Network settings– створення правил доступу до мережі

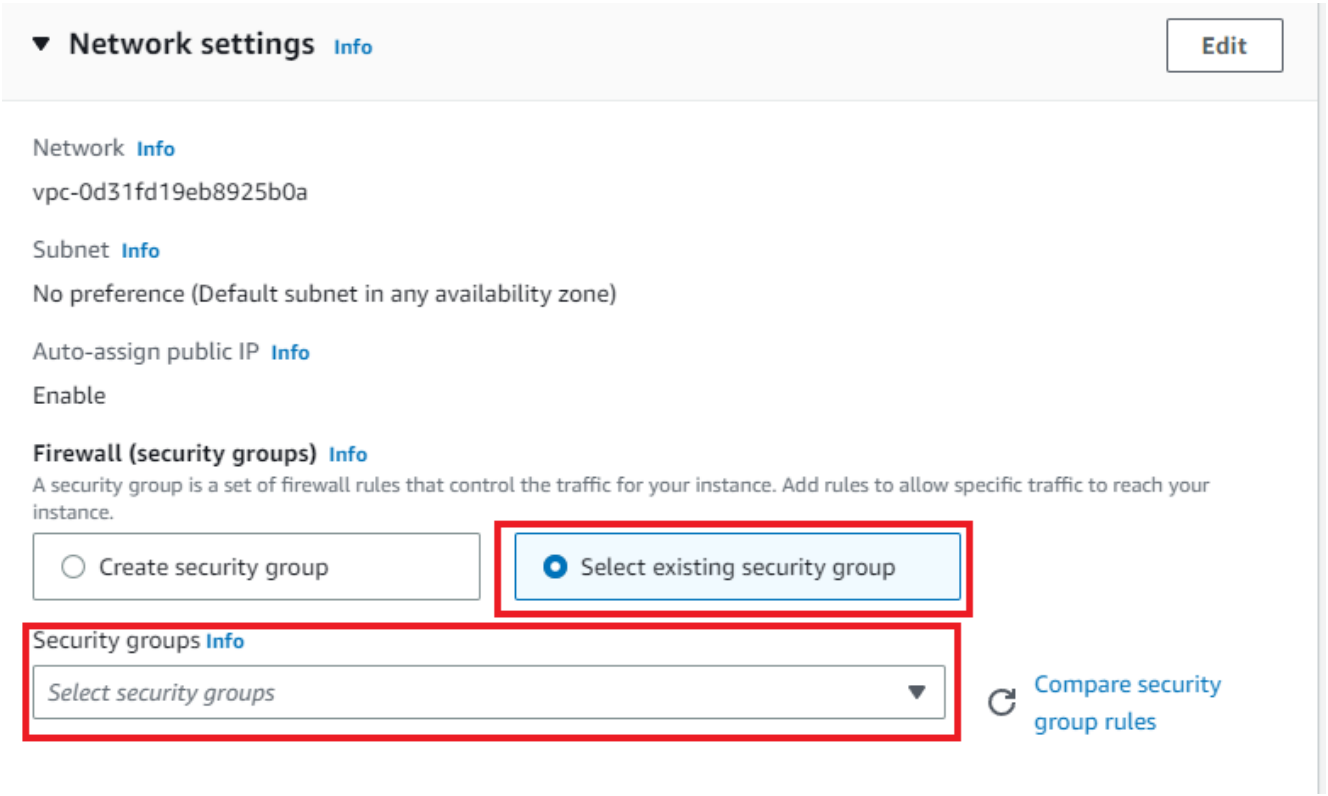


Рисунок 3.10 – модуль Network settings– вибір правил доступу мережі

9) Наступним кроком у процесі створення інстанса є вибір правил доступу до майбутнього сервера. Це може бути як публічний сервер для веб-сторінки, так і приватний сервер для обробки корпоративних даних. Подібно до вибору ключової пари доступу, правила доступу до мережі можна вибрати з уже існуючих (Рисунок 3.10) або створити нові (Рисунок 3.9). Налаштування правил доступу визначає, хто і як зможе підключатися до сервера, а також дозволяє налаштувати специфічні мережеві порти для певних типів підключень, що впливає на рівень безпеки сервера.

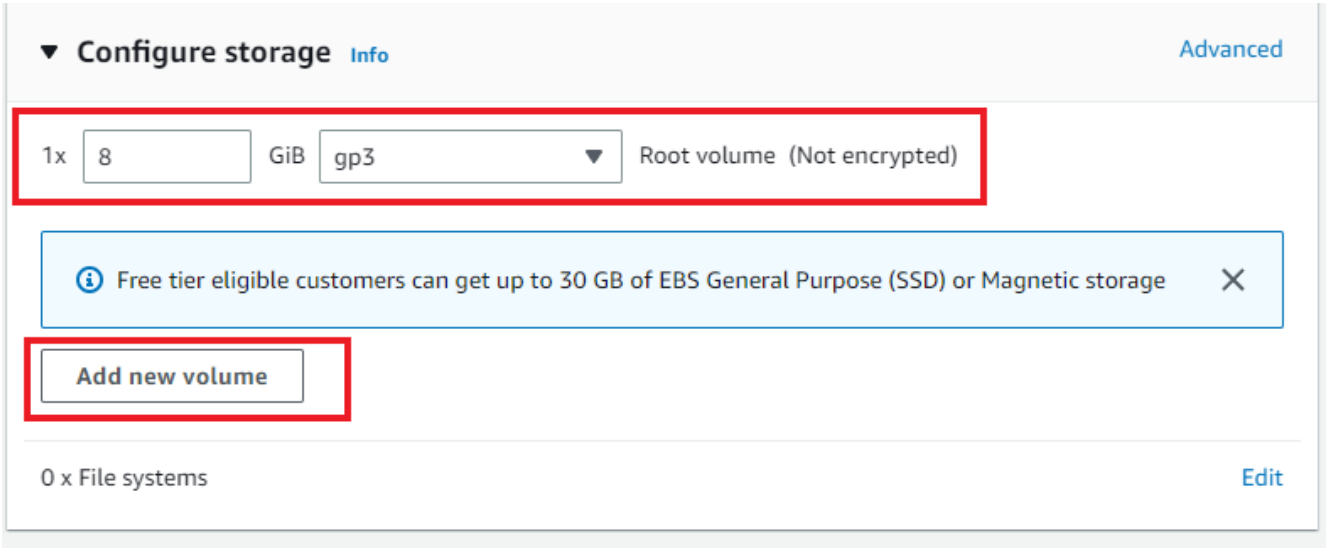


Рисунок 3.11 – модуль Configure storage

10) Останнім з основних модулів для конфігурації інстанса є модуль "Configure storage". У цьому модулі можна додати диски для зберігання даних. Під час роботи інстанса в майбутньому можна буде додавати нові диски у будь-який момент, тому немає потреби одразу додавати більше, ніж необхідно. Це важливо з точки зору оптимізації витрат, оскільки кожен додатковий диск потребує додаткових фінансових вкладень. Оптимальним рішенням буде спершу налаштувати мінімально необхідне зберігання, а потім масштабувати його у міру потреби.

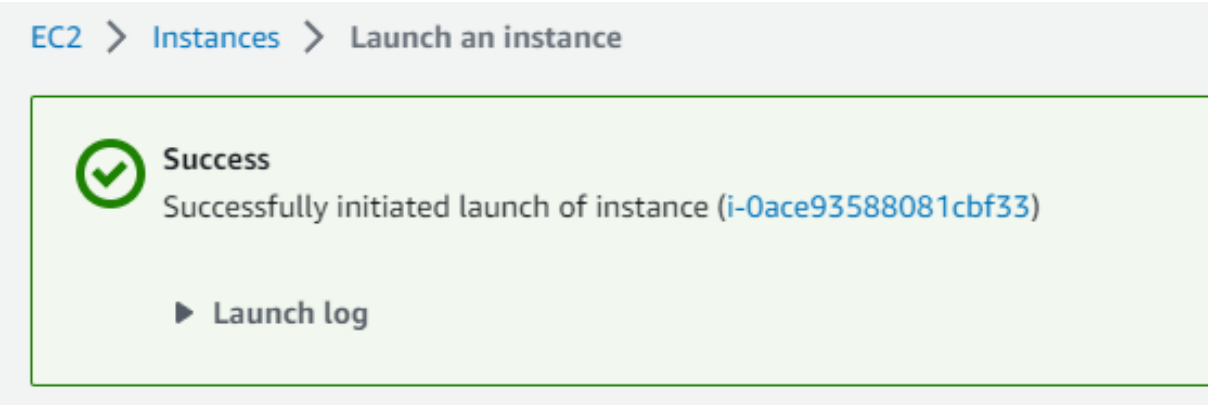


Рисунок 3.12 – Успішне створення нового instance і присвоєння йому ID

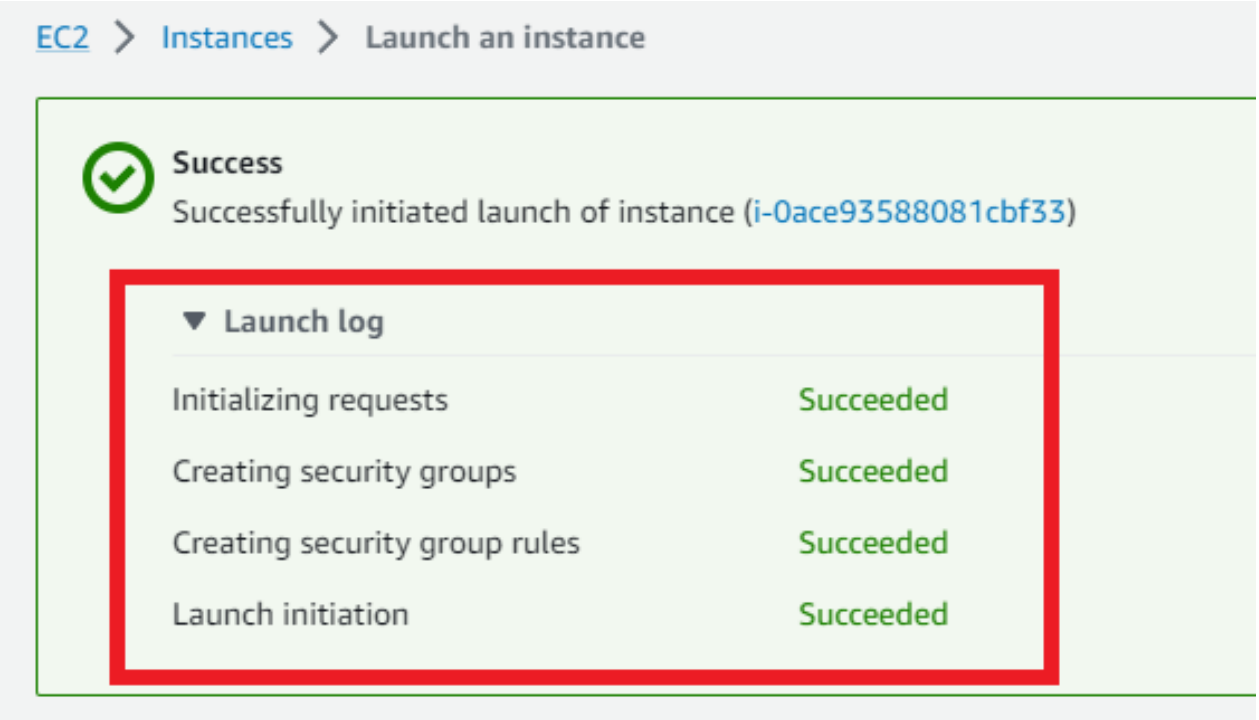


Рисунок 3.13 – Перелік логів які були під час створення нового instance

11) Після успішного створення нового інстанса він автоматично запускається, і на сторінці веб-браузера з'являється відповідне повідомлення з ідентифікатором (ID) створеного інстанса (Рисунок 3.12). Крім того, за необхідності, можна переглянути логи, що були згенеровані під час процесу створення інстанса, для отримання детальної інформації про його налаштування та перебіг процесу (Рисунок 3.13).

3.1.2 Виконання входу до EC2 Instance через командний рядок Linux

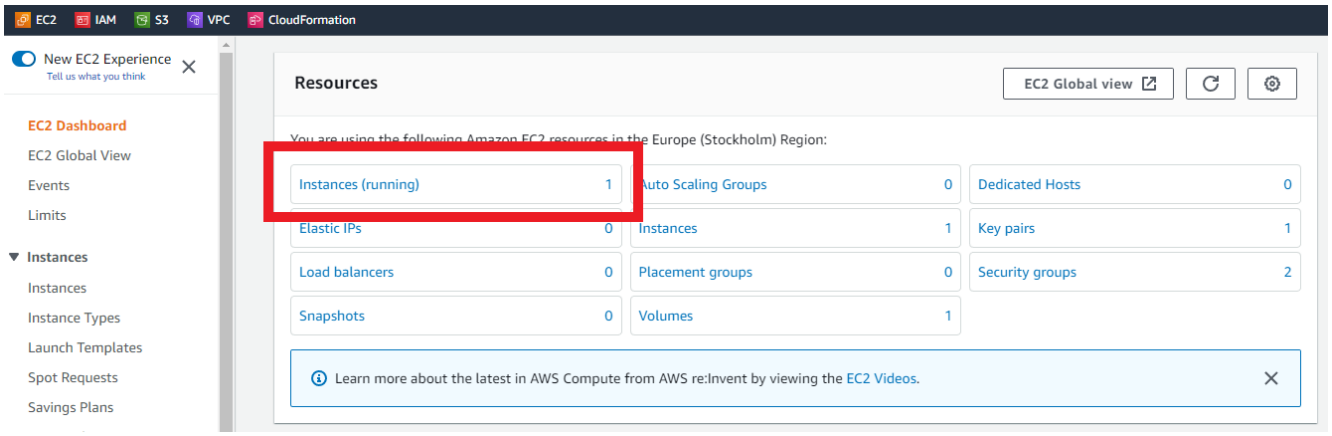


Рисунок 3.14 – EC2 Dashboard – поле instances

1) Як тільки інстанс буде створено, він відразу з’явиться на головній сторінці EC2 у розділі "Instances" (Рисунок 3.14). Щоб отримати детальну інформацію про новостворений інстанс, необхідно натиснути на відповідне поле, як показано на рисунку 3.14, і перейти на сторінку зі списком усіх інстансів. На цій сторінці можна отримати повну інформацію про інстанс, зокрема його статус, IP-адресу, тип, а також керувати його налаштуваннями та параметрами доступу.

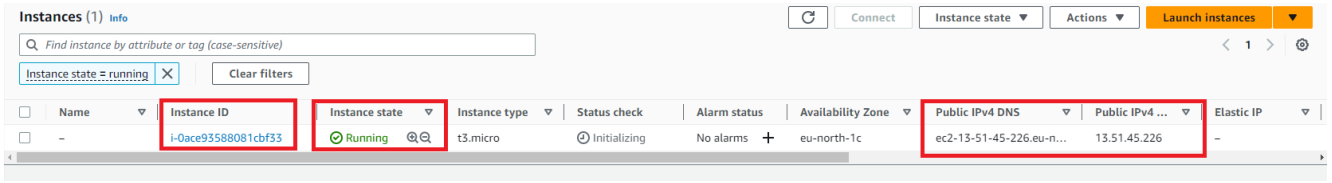


Рисунок 3.15.1 – EC2 інстанси — коротка інформація про запущені та створені інстанси

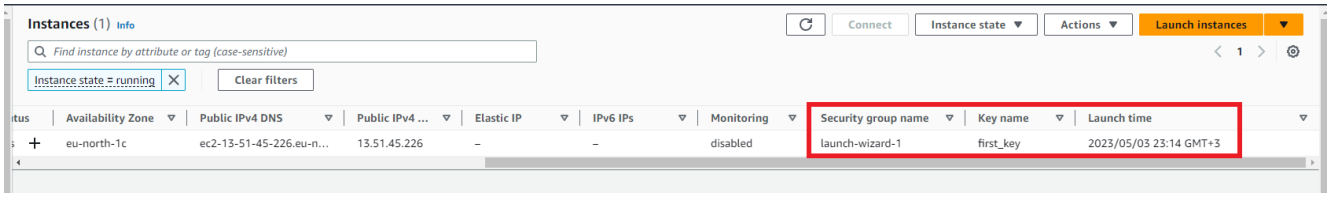


Рисунок 3.15.2 – EC2 інстанси — короткий огляд запущених та створених інстансів.

2) Після переходу за посиланням користувач може переглянути короткі відомості про створені інстанси. Основна інформація, яку можна отримати на цій сторінці, включає публічну IP-адресу інстанса, його поточний стан, унікальний ідентифікатор (ID), а також ключ доступу, що використовується для підключення до інстанса. Інша подібна інформація також відображається для зручного керування (Рисунок 3.15.1 та Рисунок 3.15.2).

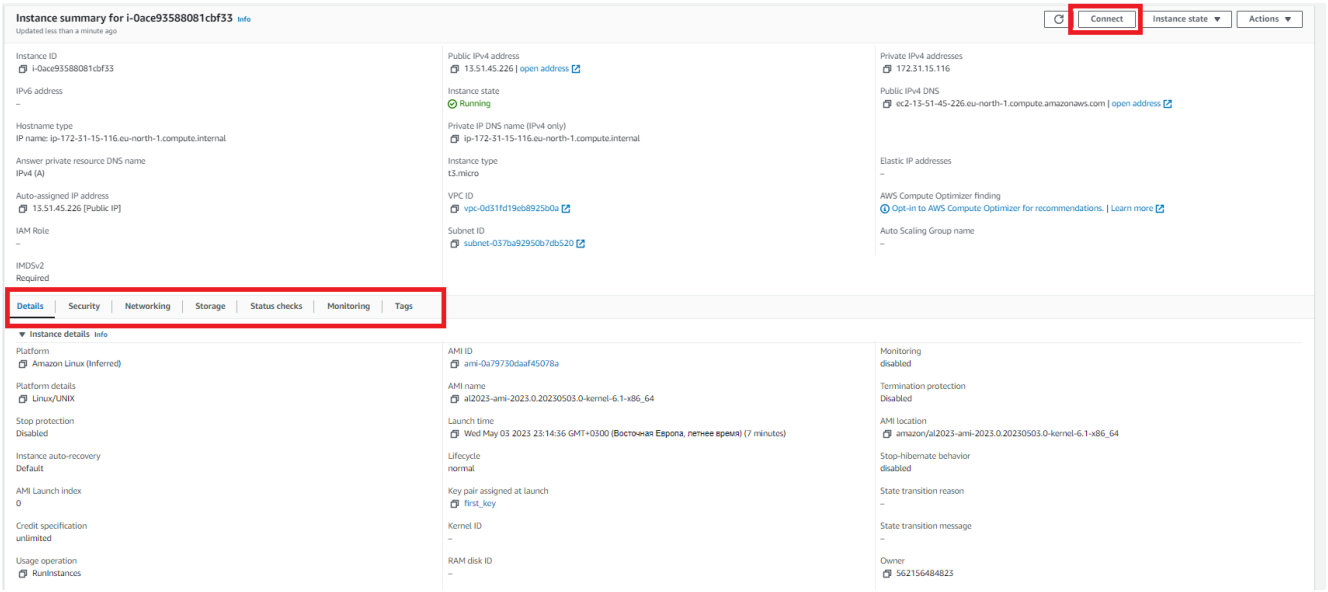


Рисунок 3.16 – EC2 instances– повна інформація по обраному instance

3) Якщо натиснути на ID інстанса (Рисунок 2.15.1), користувача перенаправить на сторінку з детальною інформацією про цей інстанс (Рисунок 2.16). На цій сторінці для доступу до інстанса необхідно натиснути кнопку "Connect", яка розташована у верхньому правому куті (Рисунок 2.16). Після цього користувач буде спрямований на сторінку, де детально описані інструкції для всіх можливих способів підключення до інстанса.

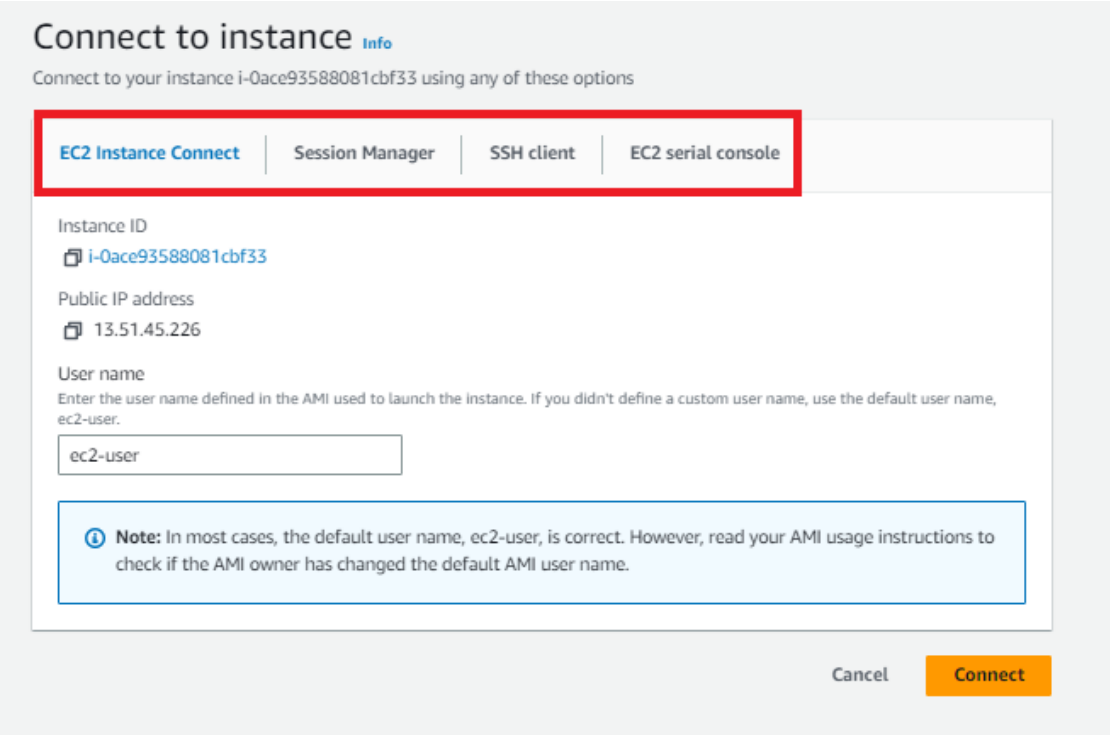


Рисунок 3.17 – Connect to instance

4) Щоб підключитися до інстанса за допомогою SSH через командний рядок у Linux, необхідно виконати кілька кроків. Спершу слід перейти до розділу "SSH client" на сторінці підключення (Рисунок 3.17). У цьому розділі наведено покрокову інструкцію для встановлення з'єднання з інстансом.

На цій сторінці ви знайдете команду для підключення, яка виглядає приблизно так:

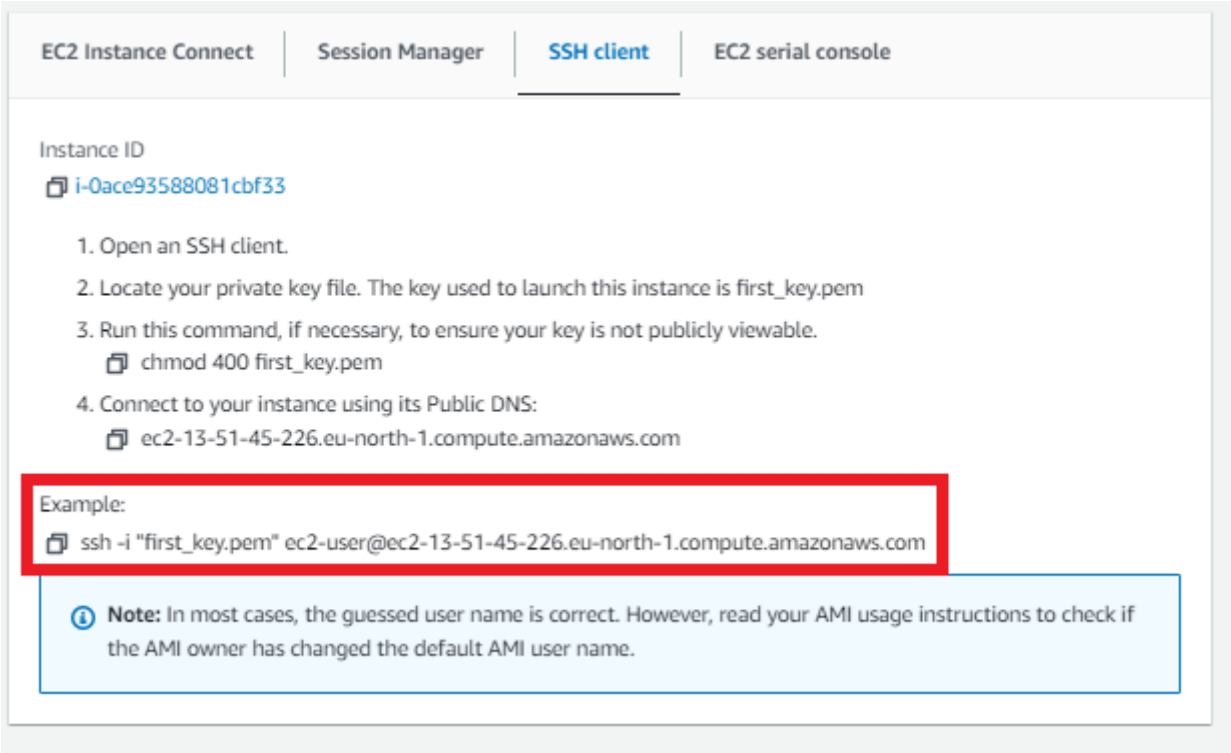


Рисунок 3.18 – Connect to instance – SSH client

5) Виконавши всі інструкції, наведені в розділі "SSH client" (Рисунок 3.18), користувач зможе підключитися до свого інстанса за допомогою команди, яка виділена на рисунку 3.18. Ця команда дозволяє встановити захищене з'єднання з інстансом, надаючи можливість керувати сервером через командний рядок, виконувати команди, налаштовувати програми та здійснювати інші адміністративні дії.

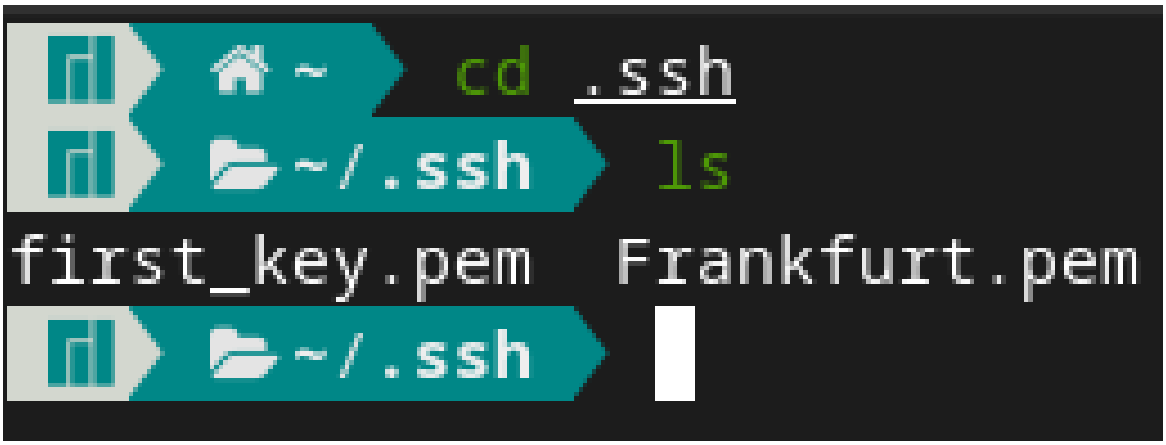


Рисунок 3.20 – Командний рядок Linux – перехід в директорію з ключем доступу

Рисунок 3.21 – Командний рядок Linux – виконання команди які прописані в Connect to instance

6) Щоб виконати вхід на інстанс, спочатку необхідно відкрити командний рядок у Linux. Далі, за допомогою команди cd, потрібно перейти до тієї директорії, в якій зберігається ключ (ключ був створений і завантажений, як показано на рисунках 3.8.1 і 3.8.2). Після цього слід перевірити, чи присутній цей ключ у директорії, використовуючи команду ls (Рисунок 3.20).

Після підтвердження наявності ключа, потрібно виконати команду, яка наведена в інструкціях до підключення (Рисунок 3.21). Ця команда встановлює права на читання для файлу з ключем, що необхідно, щоб забезпечити доступність ключа для програми під час підключення до сервера. Правильна налаштування прав доступу до файлу є важливим для того, щоб ключ міг бути використаний для безпечного встановлення з'єднання з інстансом через SSH.

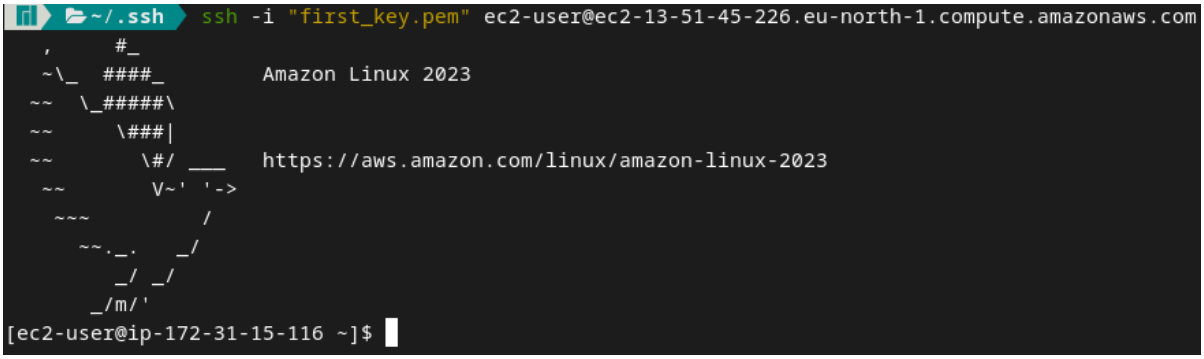


Рисунок 3.22 – Командний рядок Linux – виконання команди ssh для переходу на instance

7) Після виконання останньої команди, якщо всі попередні кроки були виконані правильно, користувач потрапить на створений сервер. На цьому сервері він зможе провести всі необхідні налаштування для належного виконання завдань, призначених для цього сервера. Це може включати встановлення додаткового програмного забезпечення, налаштування системних параметрів, конфігурацію

безпеки та будь-які інші кроки, необхідні для забезпечення роботи сервера відповідно до потреб проєкту.

3.2 S3 bucket. Створення. Використання

S3 Bucket — це сервіс AWS, призначений для зберігання великих обсягів даних за дуже низькою ціною. Відомим аналогом такого рішення є Google Drive, однак, на відміну від нього, AWS S3 Bucket дозволяє зберігати набагато більше інформації за меншу вартість. Цей сервіс не призначений для використання як жорсткий диск на EC2 інстансі, оскільки ресурси, на яких працює S3, є значно повільнішими. Проте їхня вартість суттєво нижча, що робить S3 економічно вигідним рішенням.

Зазвичай цей сервіс використовується для зберігання документів, зображень, відеофайлів, матеріалів для проєктів тощо. Він є необхідним для бізнесу, оскільки певні документи потрібно зберігати протягом десятиліть — 10, 15, а іноді й 100 років. Використання жорстких дисків для таких завдань не є надійним рішенням через ризик втрати даних, тоді як S3 забезпечує довготривале та безпечне зберігання.

Для початку роботи з цим сервісом необхідно перейти на головну сторінку S3, скориставшись посиланням у верхньому рядку головного меню AWS (Рисунок 3.1).

3.2.1 Створення S3 bucket

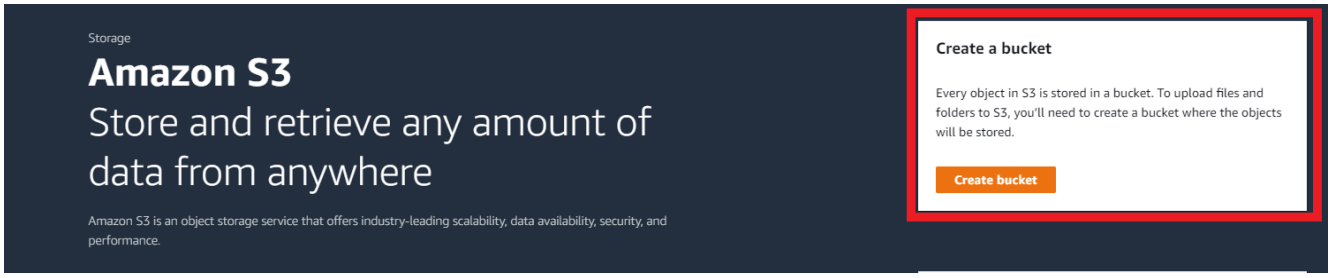


Рисунок 3.23 – Головне меню S3

1) Після переходу на головну сторінку роботи з S3, у правій частині екрана буде розташований модуль "Create a Bucket" (Рисунок 3.23). Цей модуль дозволяє почати створення нового бакету для зберігання даних.

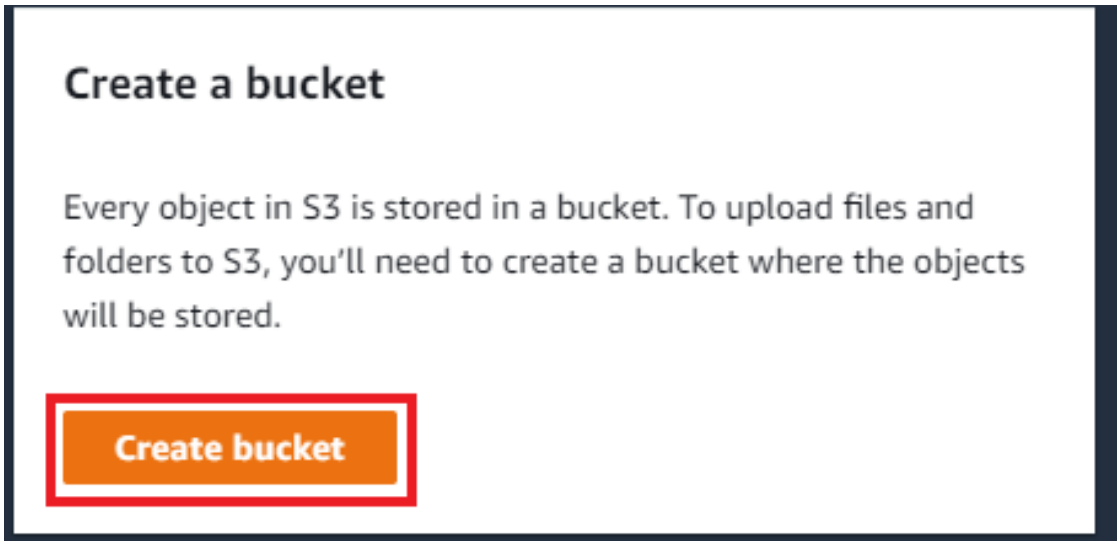


Рисунок 3.24 – Головне меню S3 – Модуль Create a bucket

2) Для створення нового сховища даних потрібно натиснути на кнопку, позначену на рисунку 3.24. Після цього користувач буде перенаправлений на сторінку створення нового бакету, де можна задати основні параметри та налаштування для зберігання даних.

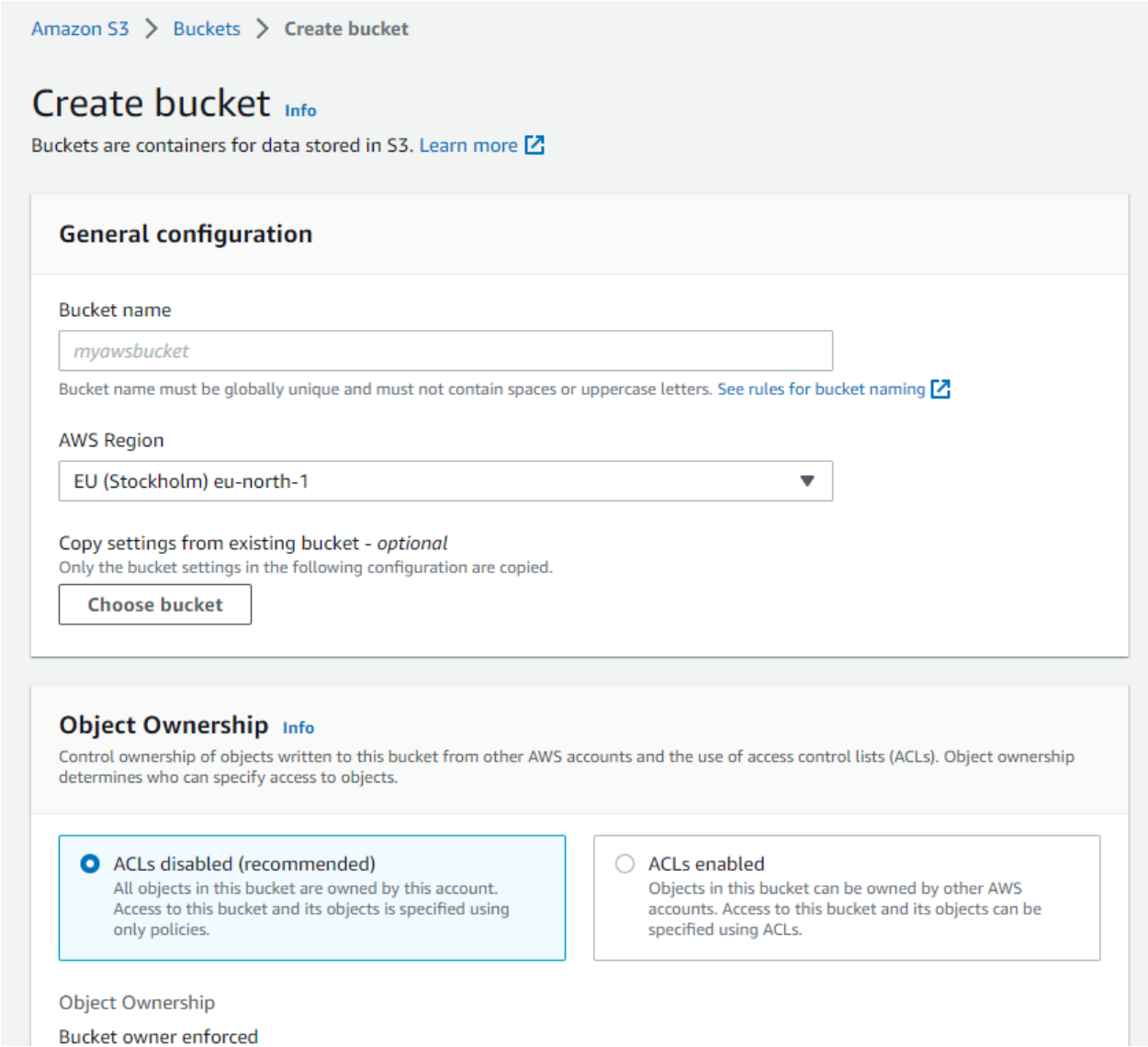


Рисунок 3.25 – Меню створення нового bucket

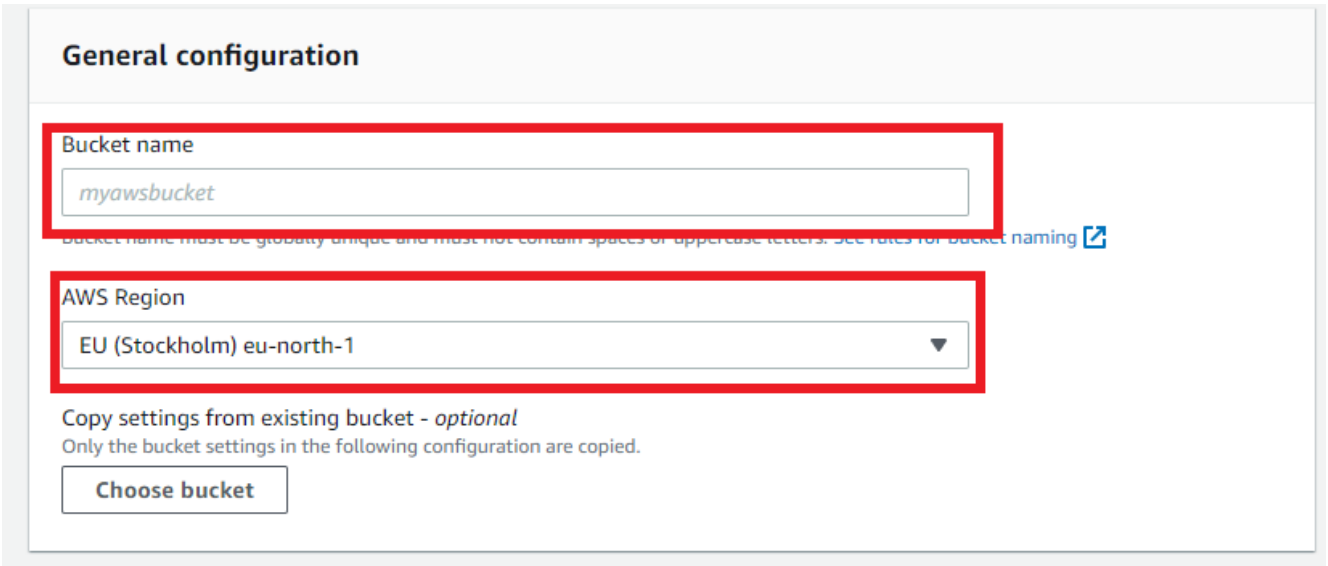


Рисунок 3.26 – Меню створення нового bucket – General configuration

3) Першим етапом створення нового бакета є визначення його унікального імені та вибір регіону, де він буде розташований (Рисунок 3.25). Ім'я бакета повинно бути унікальним у всьому світі, що є важливою умовою для його створення. Зазвичай, для імені використовуються ініціали користувача, назва компанії або будь-яке інше унікальне позначення, яке має відношення до власника.

Заповнення поля з назвою бакета та вибір регіону, де буде розташоване сховище, є першим кроком у процесі створення нового бакета. Ці налаштування дозволяють забезпечити зручне управління даними та оптимальну доступність сховища залежно від географічного розташування (Рисунок 3.26).

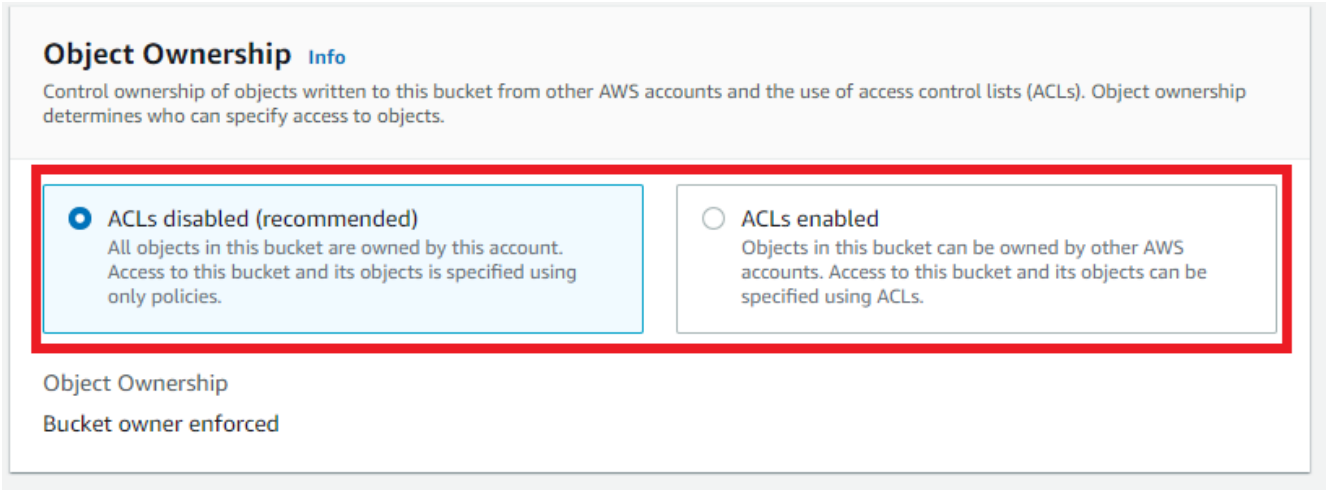


Рисунок 3.27 – Меню створення нового bucket – Object Ownership

4) Наступним кроком є вибір між публічним сховищем або сховищем з обмеженим доступом. Зазвичай S3 bucket використовують для зберігання конфіденційної інформації, тому за замовчуванням обирається варіант з обмеженим доступом (Рисунок 3.27). Цей вибір забезпечує захист даних від несанкціонованого доступу.

Однак, у випадках, коли необхідно розмістити великий обсяг даних у публічному просторі для загального доступу, наприклад, для спільного використання або публічних проєктів, слід вибрати варіант створення публічного сховища. Цей варіант дозволяє зробити дані доступними для всіх користувачів без потреби у спеціальній автентифікації..

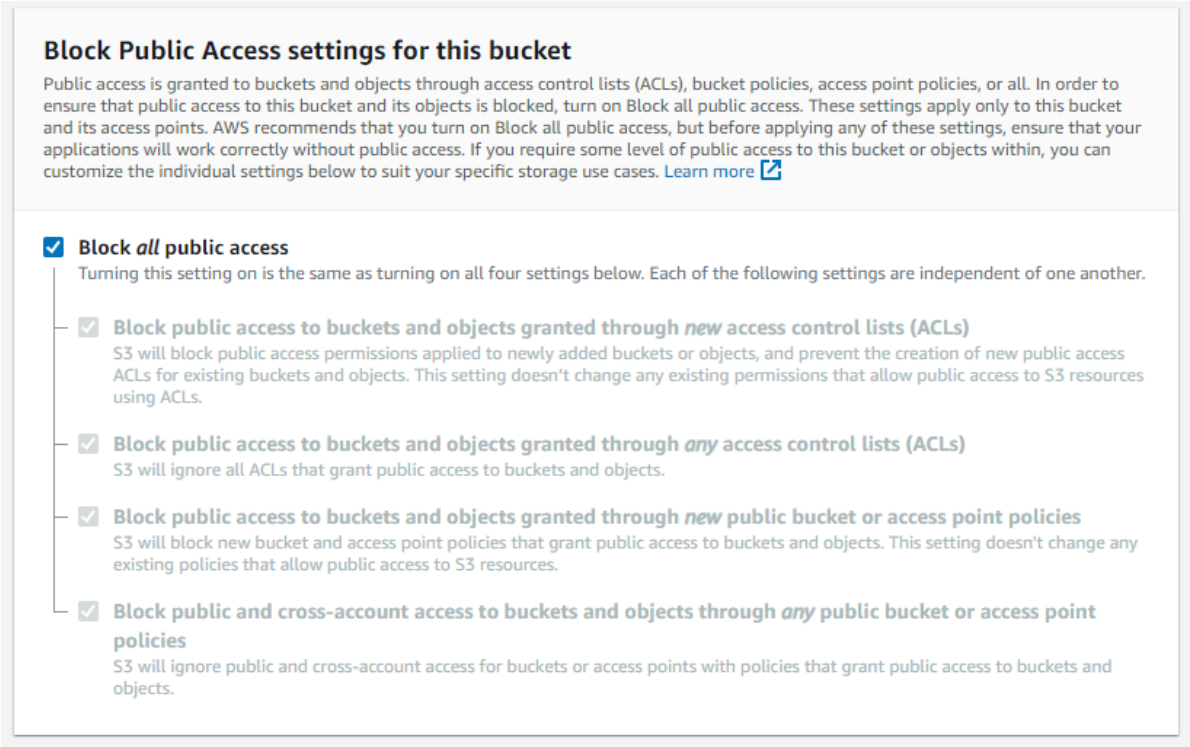


Рисунок 3.28 – Меню створення нового bucket – Block Public Access setting for this bucket

5) Наступний модуль, під назвою "Block Public Access setting for this bucket", відповідає за налаштування публічного доступу до сховища. У деяких ситуаціях може виникнути потреба у створенні частково публічного бакета, наприклад, для надання доступу до певних даних або ресурсів. У таких випадках цей модуль дозволяє налаштувати доступ відповідно до вимог.

Однак, у більшості випадків налаштування цього модуля залишаються за замовчуванням, щоб забезпечити максимальний рівень захисту даних і запобігти несанкціонованому доступу. Цей підхід є оптимальним для більшості сценаріїв використання S3 bucket.

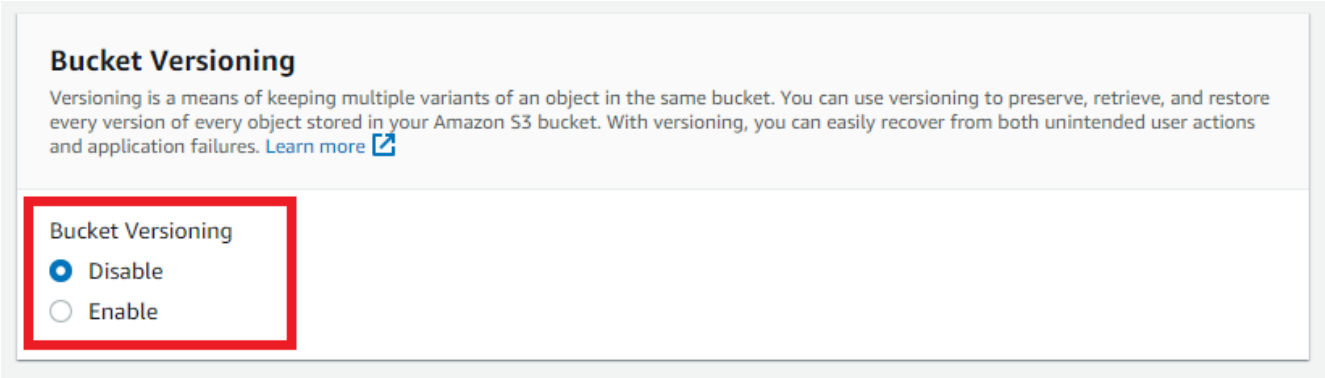


Рисунок 3.29 – Меню створення нового bucket – Block Versioning

6) Шостий модуль відповідає за контроль версій у створеному сховищі. Це налаштування є корисним у випадках, коли у бакеті зберігатимуться документи або дані, які можуть змінюватися з часом. Завдяки цій функції, при пошкодженні файлу або внесенні небажаних змін завжди можна буде повернутися до попередньої версії документа та відновити його початковий стан.

Однак варто зазначити, що увімкнення контролю версій збільшує вартість зберігання, оскільки зберігаються всі попередні версії файлів. Тому це рішення доцільно використовувати лише у випадках, коли дані часто змінюються або потребують відстеження історії змін. Для файлів, які не планується оновлювати регулярно, ця функція може бути зайвою та економічно недоцільною..

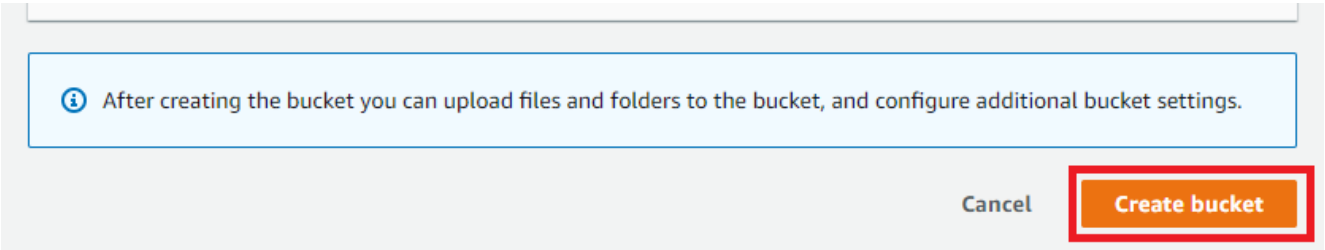


Рисунок 3.30 – Меню створення нового bucket – Create bucket

7) Після завершення всіх основних налаштувань, у нижній частині сторінки знаходиться кнопка "Create bucket". Натискання цієї кнопки автоматично створює бакет з усіма зазначеними раніше параметрами та налаштуваннями (Рисунок 3.30).

Це завершує процес створення нового сховища, яке тепер готове до використання для зберігання даних відповідно до заданих вимог.

3.2.2 Використання S3 bucket

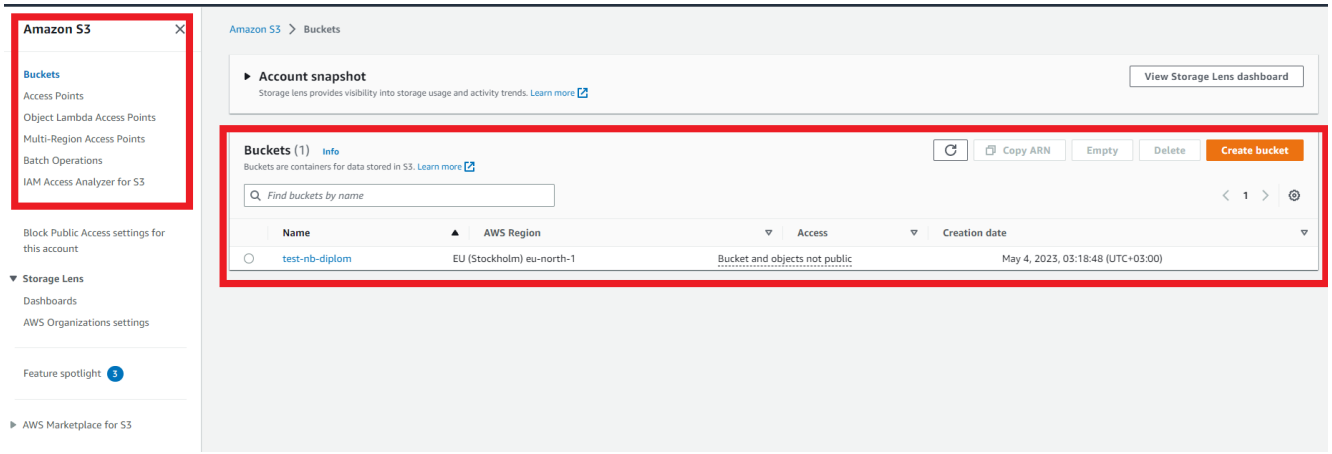


Рисунок 3.31 – Головна сторінка AWS S3

Після створення першого бакета, головна сторінка служби S3 змінює свій вигляд. У лівій частині екрану з’являється меню з різними пунктами, кожен з яких відповідає за певний сервіс або налаштування, доступні в S3. У центральній частині сторінки відображається список усіх створених бакетів, де можна переглядати їхній статус, ім'я, регіон та іншу основну інформацію (Рисунок 3.31). Це забезпечує зручний доступ до всіх існуючих сховищ і дозволяє швидко перемикатися між ними для управління та роботи з даними.

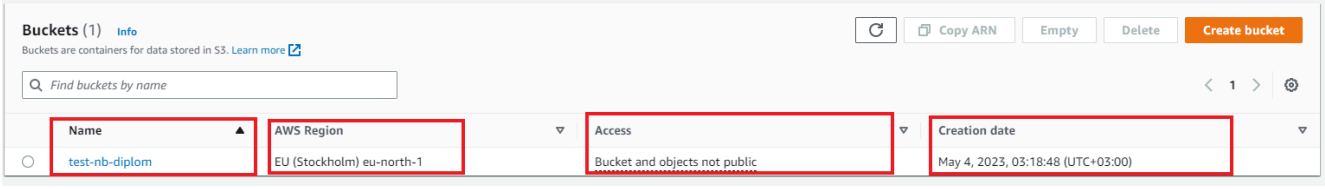


Рисунок 3.32 – Головна сторінка AWS S3 – Список buckets

У цьому списку відображаються всі бакети, створені користувачем. Тут також представлена основна інформація про існуючі бакети, яка включає їх назву, регіон розташування, рівень доступності та дату створення (Рисунок 3.32). Ця

інформація допомагає користувачу швидко орієнтуватися серед своїх бакетів і отримувати ключові дані для подальшого управління ними.

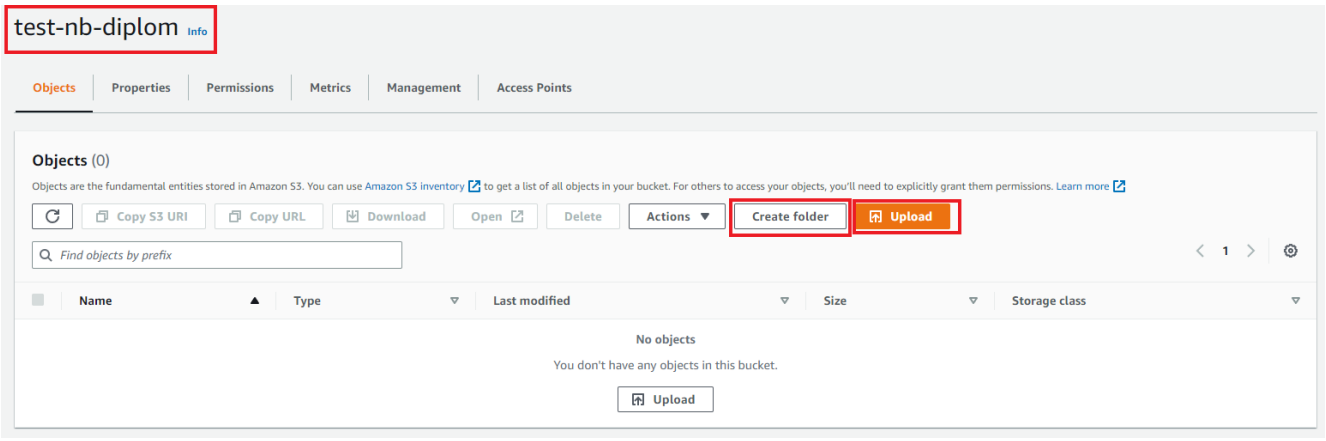


Рисунок 3.33 – Середина не наповненого bucket

При натисканні на ім’я бакета, користувач переходить до його вмісту і отримує можливість переглядати всі файли та дані, які там зберігаються.

Для зручності організації роботи з бакетом користувач може створювати директорії, щоб розподілити файли за відповідними категоріями. Для цього потрібно натиснути кнопку "Create folder", яка виділена на Рисунку 3.33, і задати ім’я для нової директорії. Це дозволяє ефективно структурувати дані в бакеті.

Щоб завантажити певні файли до сховища, слід натиснути кнопку "Upload", також позначену на Рисунку 3.33. Після цього користувач буде перенаправлений на сторінку завантаження даних, де зможе вибрати файли з локального пристрою, налаштувати параметри завантаження і завантажити їх у бакет.

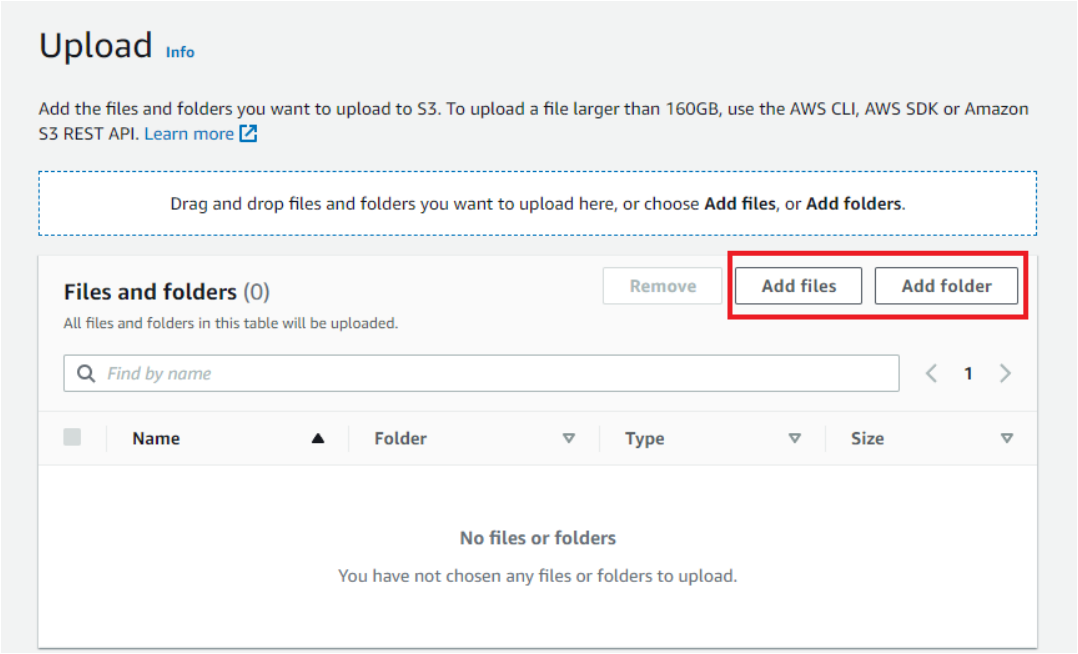


Рисунок 3.34 – Додавання файлів до сховища

Щоб завантажити файли з локального комп’ютера до бакета, необхідно виконати кілька простих кроків. Спочатку на сторінці завантаження слід натиснути кнопку "Add files", яка чітко позначена на рисунку 3.34. Після цього відкриється вікно провідника файлів вашої операційної системи.

У провіднику можна переглянути всі доступні файли та папки на вашому комп’ютері. Для завантаження потрібно вибрати один або кілька файлів, які ви хочете додати до сховища. Це можна зробити, виділяючи файли мишею або використовуючи клавішу Ctrl (або Command на Mac) для вибору декількох файлів одночасно.

Після вибору необхідних файлів натисніть кнопку "Open" (або "Відкрити" залежно від мови вашої операційної системи) у вікні провідника. Файли автоматично з’являться у списку завантажень на сторінці S3.

Цей інтуїтивний процес дозволяє легко додавати дані до бакета для їх подальшого зберігання, організації або обробки. Усі обрані файли будуть готові до наступного кроку, де можна налаштувати додаткові параметри завантаження, якщо це потрібно..

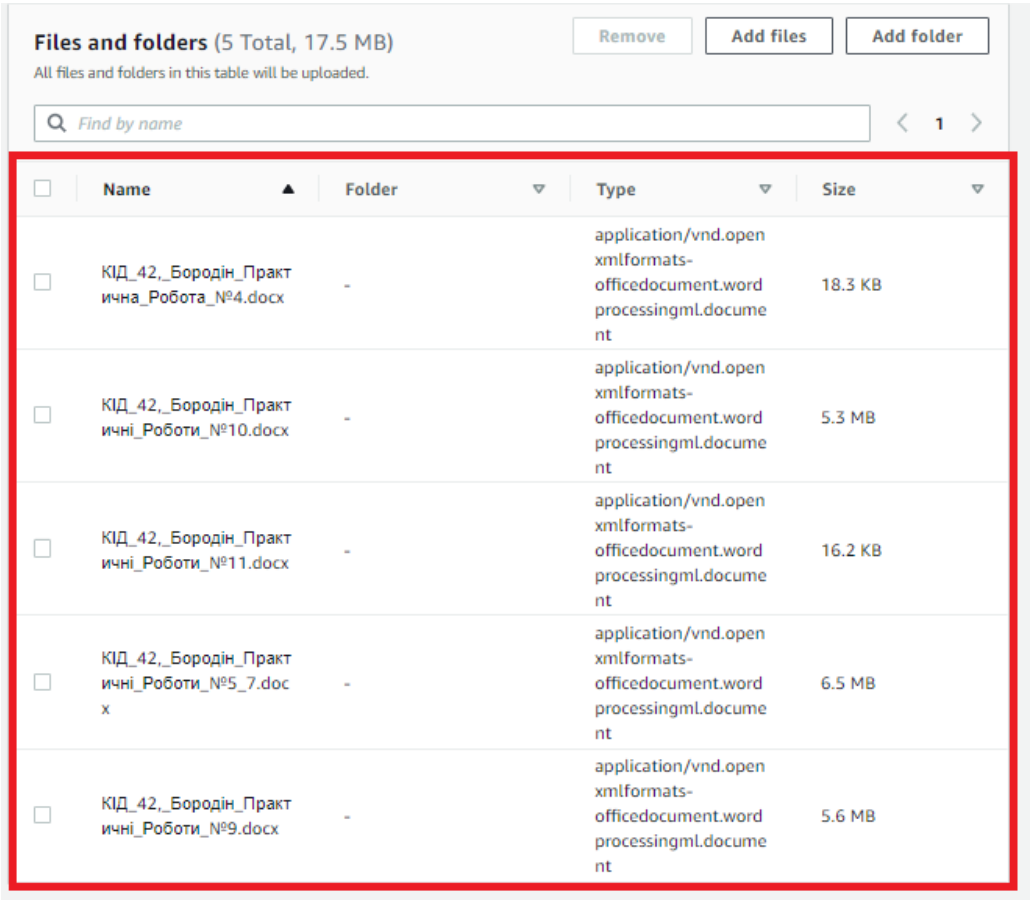


Рисунок 3.35 – Файли готові до завантаження

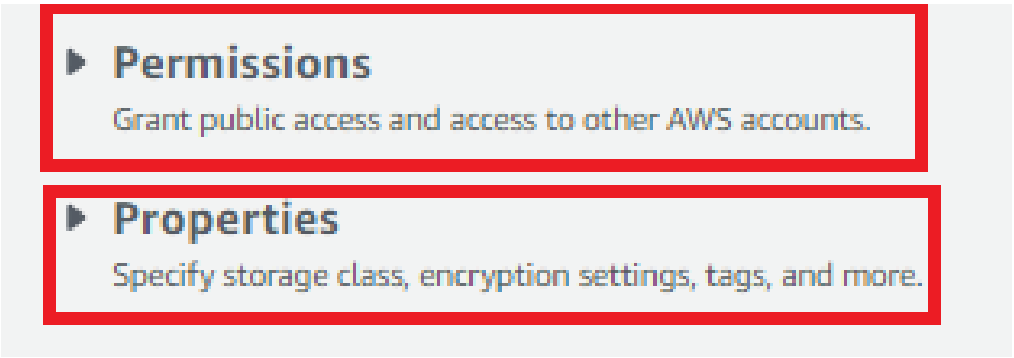


Рисунок 3.36 – Додаткові параметри при завантажені

Після вибору файлів у провіднику, вони автоматично з’являться у відповідній частині вікна завантаження (Рисунок 3.35). Тут ви зможете переглянути список вибраних файлів, включаючи їх назви, розмір та іншу основну інформацію.

Нижче на цій сторінці розташовані додаткові параметри налаштування завантаження (Рисунок 3.36). Ці параметри дозволяють визначити права доступу до файлів, вибрати тип зберігання, додати теги для організації чи ідентифікації

файлів, а також задати шифрування для підвищення безпеки даних. Використання цих опцій дозволяє адаптувати завантаження під конкретні потреби і забезпечити зручність роботи з файлами у бакеті.

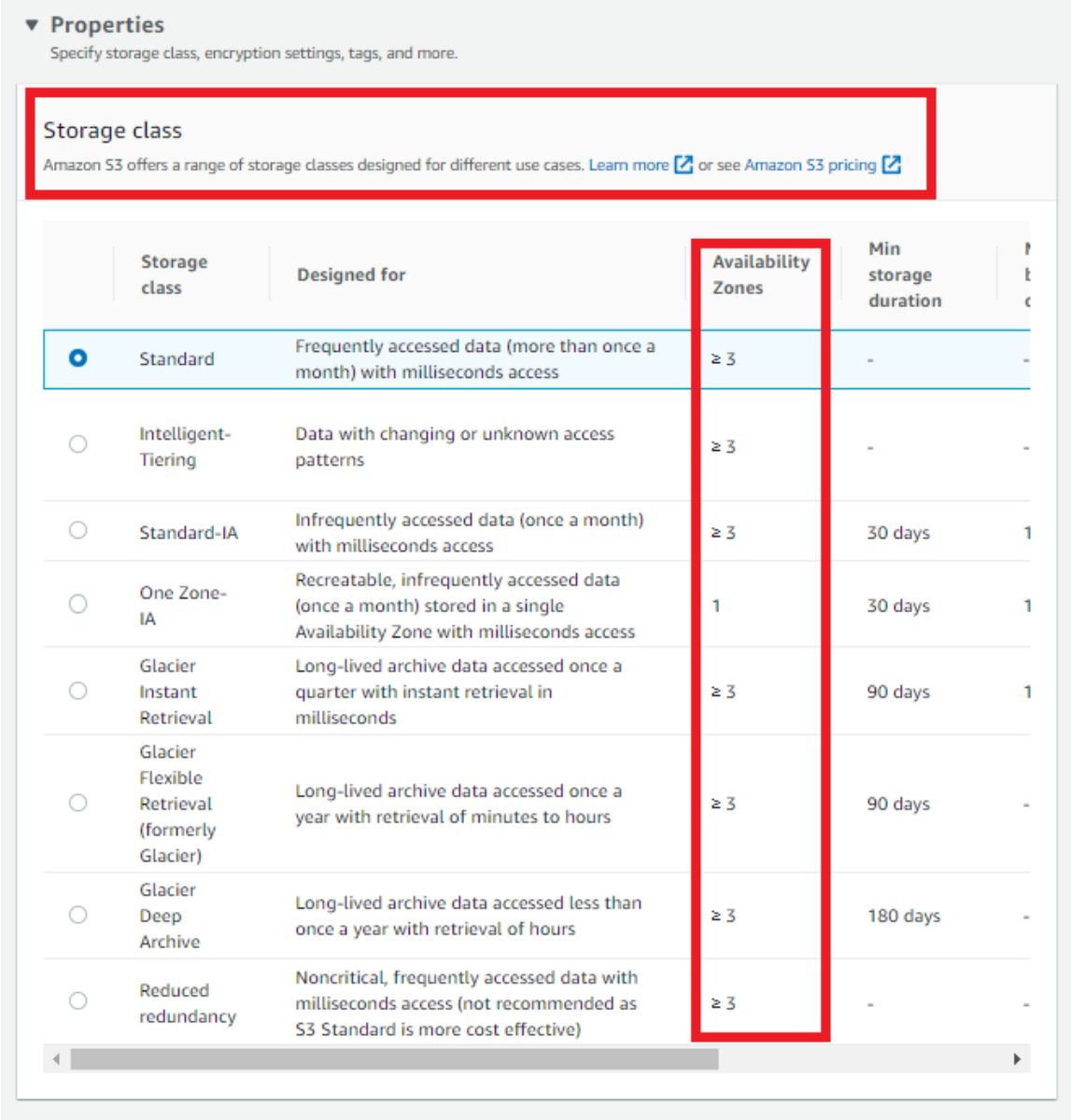


Рисунок 3.37 – Налаштування зберігання обраних файлів

При необхідності, для забезпечення додаткової безпеки, можна скористатися додатковими параметрами зберігання файлів, які показані на рисунку 3.37. У цьому полі є можливість вибрати клас зберігання, який визначає, як саме будуть зберігатися ваші файли.

Клас зберігання впливає на кількість створюваних резервних копій, регіони, в яких вони зберігатимуться, а також тривалість їхнього зберігання. Наприклад, деякі класи забезпечують високу доступність і зберігають дані в декількох регіонах, тоді як інші оптимізовані для тривалого архівного зберігання з меншою вартістю, але повільнішим доступом.

Ці параметри не лише підвищують рівень безпеки та надійності зберігання даних, але й впливають на ціну використання сервісу. Тому важливо обрати клас зберігання, який найбільше відповідає вашим потребам та бюджету.

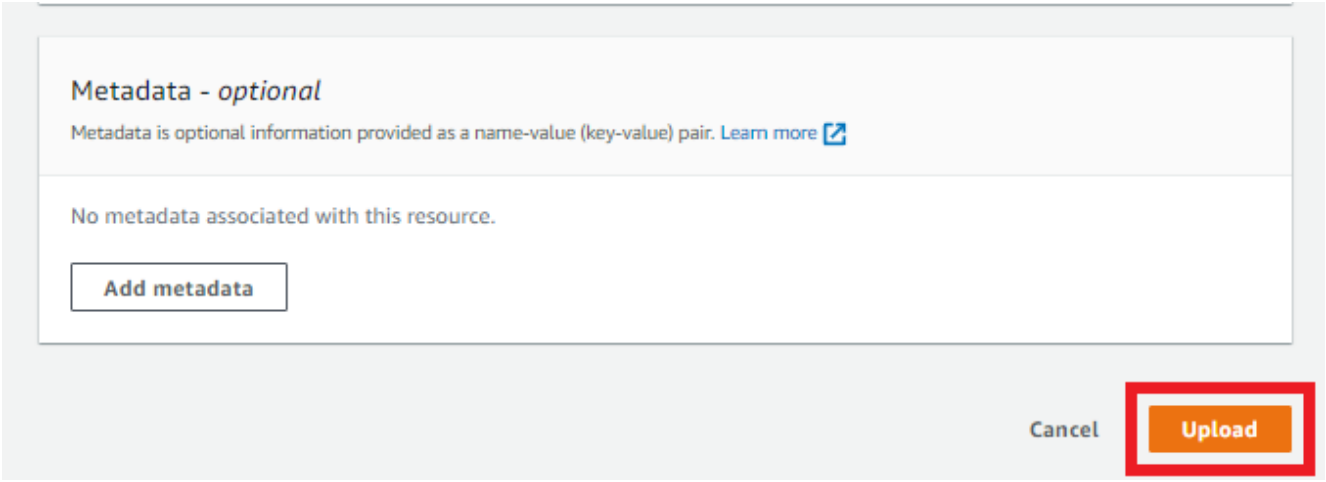


Рисунок 3.38 – Завершення додавання файлів в сховище

Щоб завершити процес завантаження даних у вибраний бакет, потрібно внизу сторінки натиснути кнопку "Upload". Після цього файли будуть завантажені у бакет відповідно до обраних налаштувань і стануть доступними для подальшого використання або обробки.

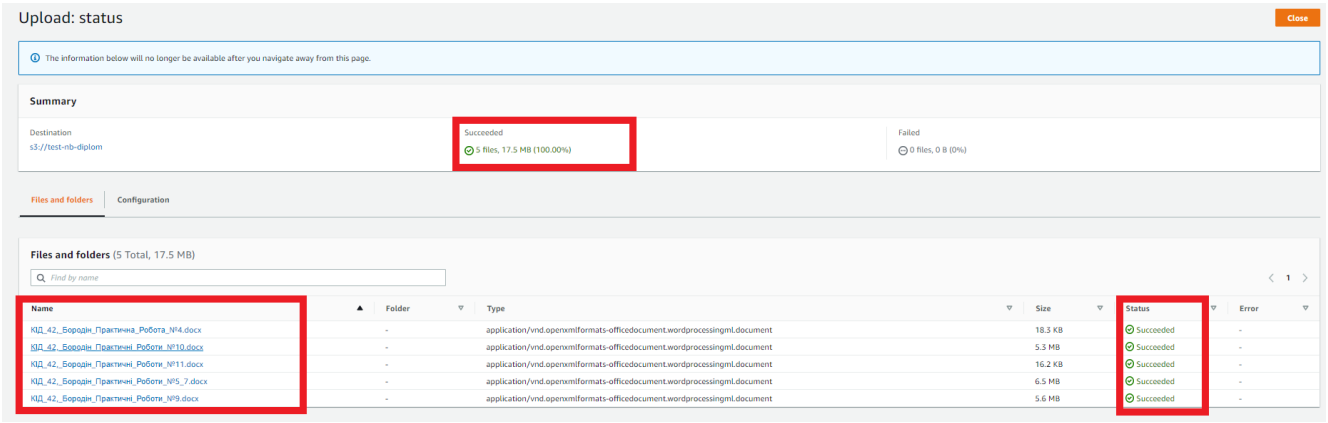


Рисунок 3.38 – Логи файлів після їх завантаження

Після завершення завантаження всіх файлів, на екрані буде відображено інформацію про статус завантаження. У цій інформації можна побачити, чи всі файли були успішно завантажені, а також загальний обсяг дискового простору, який зайняли ці файли (Рисунок 3.35). Це дозволяє переконатися, що завантаження виконано коректно, і оцінити використання доступного простору в бакеті.

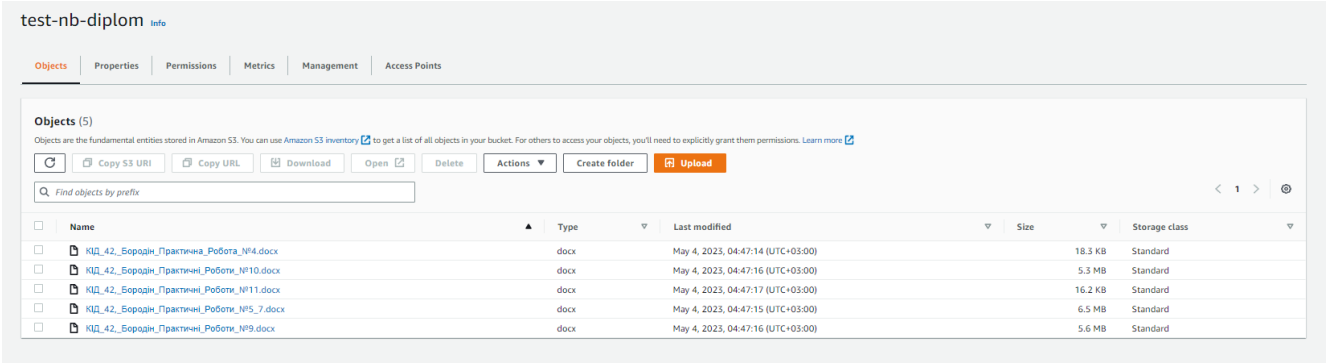


Рисунок 3.39 – Кінцевий результат залиття даних на S3 bucket

4 ВИКОНАННЯ БАЗОВОГО НАЛАШТУВАННЯ ХМАРИ ЗА ДОПОМОГОЮ TERRAFORM

4.1 Terraform – як інструмент налаштування хмари

Terraform — це сучасний інструмент для управління інфраструктурою як кодом (Infrastructure as Code, IaC), створений компанією HashiCorp. Він дозволяє автоматизувати створення, зміну, видалення та підтримку інфраструктурних ресурсів за допомогою декларативного опису в текстових файлах.

Terraform використовує декларативний підхід до управління інфраструктурою. Це означає, що користувач описує бажаний стан інфраструктури, а Terraform самостійно виконує необхідні дії для досягнення цього стану. Наприклад, якщо в конфігурації описано створення серверу з певними параметрами, Terraform забезпечить його створення, навіть якщо це вимагає кількох кроків.

Основні термінології Terraform

– Провайдери — це модулі, які дозволяють Terraform працювати з різними платформами. Кожен провайдер — це набір інструментів для створення та управління ресурсами на певній платформі (наприклад, AWS, Azure, Google Cloud, Kubernetes, VMware тощо).

– Ресурс — це базовий елемент інфраструктури, яким управляє Terraform. Наприклад, EC2-інстанс, база даних, мережевий інтерфейс або балансувальник навантаження. Ресурси описуються в файлах конфігурації Terraform.

– Декларативний синтаксис — Terraform використовує мову конфігурації HCL (HashiCorp Configuration Language), яка дозволяє зрозуміло описувати інфраструктурні ресурси.

– Стан (State) — Terraform зберігає поточний стан інфраструктури у файлі terraform.tfstate. Це дозволяє відслідковувати зміни та забезпечує ідпотентність (виконання однієї і тієї ж операції не призведе до додаткових змін).

Етапи роботи Terraform:

– **Ініціалізація проекту**. Команда terraform init завантажує плагіни провайдерів та налаштовує робоче середовище для подальшої роботи.

– **Планування змін**. Команда terraform plan дозволяє переглянути, які зміни будуть застосовані до інфраструктури. Це зручно для перевірки перед внесенням змін.

– **Застосування змін**. Команда terraform apply виконує реальні зміни в інфраструктурі, створюючи, змінюючи або видаляючи ресурси відповідно до опису в конфігураціях.

– **Знищення інфраструктури**. Команда terraform destroy дозволяє повністю видалити ресурси, створені Terraform.

Основні переваги роботи з Terraform:

– **Мультихмарність**. Terraform підтримує понад 100 провайдерів, дозволяючи працювати з різними хмарними платформами та локальними середовищами одночасно.

– **Ідпотентність**. Один і той же код завжди створює однаковий результат, незалежно від того, скільки разів він виконується.

– **Масштабованість**. Terraform підходить як для невеликих проектів, так і для великих інфраструктур.

– **Автоматизація**. Використання Terraform зменшує кількість ручних помилок і спрощує процес розгортання.

– **Контроль версій**. Terraform-код можна зберігати в системах контролю версій (наприклад, Git), що дозволяє зберігати історію змін і повертатися до попередніх станів.

Завдяки своїй універсальності, гнучкому підходу до управління інфраструктурою та підтримці мультихмарних середовищ, Terraform став одним із провідних інструментів для налаштування, автоматизації та управління хмарними сервісами. Його популярність зростає завдяки здатності ефективно працювати з різними хмарними платформами, такими як AWS, Azure, Google Cloud, а також з

локальними середовищами та спеціалізованими інструментами, такими як Kubernetes.

Terraform активно застосовується в різних сферах, пов'язаних із хмарними обчисленнями та DevOps-практиками. Його головні переваги, такі як декларативний підхід, ідпотентність та автоматизація процесів, роблять його незамінним для багатьох задач, пов'язаних з інфраструктурою як кодом (IaC). Сфери застосування Terraform включають:

– **Хмарна інфраструктура** – розгортання серверів, баз даних, балансувальників навантаження та інших ресурсів у AWS, Azure чи GCP.

– **Оркестрація контейнерів** – налаштування кластерів Kubernetes або Docker Swarm.

– **Мережеві налаштування** – автоматизація створення VPN, мережевих інтерфейсів, фаєрволів тощо.

– **Розгортання додатків** – налаштування серверів для розгортання додатків у тестових або продакшн-середовищах.

4.2 Початок роботи з Terraform

Перед тим як розпочати процес налаштування хмарних ресурсів за допомогою Terraform, необхідно виконати низку підготовчих дій, які включають встановлення цього інструменту та налаштування його базових конфігурацій.

Terraform є потужним інструментом для управління інфраструктурою, який дозволяє автоматизувати створення, зміну та видалення ресурсів у хмарному середовищі. Щоб ефективно використовувати його можливості, спершу слід завантажити та встановити Terraform на ваш комп’ютер або сервер. Цей процес залежить від операційної системи, на якій ви працюєте, тому потрібно дотримуватися відповідних інструкцій для Windows, macOS або Linux.

```
Terraform v1.5.6
on darwin_amd64
+ provider registry.terraform.io/hashicorp/aws v4.68.0
```

Рисунок 4.1 – Приклад виконання команди terraform -v

Після встановлення необхідно перевірити, чи правильно налаштоване середовище, виконавши команду terraform -v в командному рядку (Рисунок 4.1). Це допоможе переконатися, що Terraform встановлено коректно та готове до роботи.

Наступним етапом є налаштування базових конфігурацій. Це включає створення файлу конфігурації з розширенням .tf, де визначаються ресурси, які ви хочете створити або змінити. Також потрібно налаштувати провайдер, через який Terraform буде взаємодіяти з вашим хмарним середовищем (наприклад, AWS, Azure чи GCP). Для цього необхідно вказати облікові дані, регіони та інші параметри, що забезпечать доступ до вашої хмарної інфраструктури.

Ці підготовчі дії є ключовими для подальшої успішної роботи з Terraform і дозволяють уникнути можливих помилок у процесі налаштування хмарної інфраструктури

4.3 Виконання створення EC2 Instance через Terraform

Для створення EC2 Instance через Terraform необхідно створити файл з розширенням .tf, який міститиме всі необхідні налаштування для ініціалізації та запуску інстанса. Terraform використовує цей файл як інфраструктурний код, що дозволяє автоматизувати процес створення, конфігурації та управління хмарними ресурсами.

```
# Визначення провайдера AWS
provider "aws" {
  region = "us-west-2" # Вкажіть потрібний регіон AWS
}
```

Рисунок 4.2 – Оголошення провайдера

В самому початку написання Terraform файлу необхідно на якому саме хмарному провайдері буде будуватись нова інфраструктура. Для цього необхідно прописати відповідні команди що відображені на рисунку 4.2.

```
resource "aws_instance" "example" {
  ami           = "ami-0c55b159cbfafe1f0" # Ідентифікатор AMI
  instance_type = "t2.micro"              # Тип інстанса
}
```

Рисунок 4.3 – Створення нового Instance

Наступним етапом є створення нового EC2 Instance, код для його створення відображено на рисунку 4.3. В даному коді демонструється мінімально необхідні налаштування, за необхідності їх можливо доповнити. На рисунку 4.3 відбувається оголошується ресурс типу aws_instance, де вказуються основні параметри:

– Ім'я AMI (Amazon Machine Image), яке визначає операційну систему.
– Тип інстанса (наприклад, t2.micro для безкоштовного рівня AWS).
–

```
key_name = "my-key-pair" # Ім'я ключової пари, створеної в AWS
```

Рисунок 4.4 – Налаштування ключа доступу (Key Pair)

За для безпеки необхідно також прописати створення ключа доступу. Даний ключ буде забезпечувати безпеку при підключенні до створеного інстансу. Код для створення даного ключа відображений на рисунку 4.4.

```
resource "aws_security_group" "example" {
  name_prefix = "example-security-group"

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Доступ через SSH
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # HTTP-доступ
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"] # Дозвіл на вихідний трафік
  }
}
```

Рисунок 4.5 – Додавання правил безпеки (Security Group)

```
vpc_security_group_ids = [aws_security_group.example.id]
```

Рисунок 4.6 – Підключення інстансу до Security Group

Невід’ємною частиною створення EC2 Instance є налаштування правил Security Group. Додавання правил безпеки (Security Group) є ключовим етапом під час налаштування EC2 Instance або будь-якого іншого хмарного ресурсу в AWS. Security Group виконує роль брандмауера на рівні мережі, який контролює вхідний та вихідний трафік до і від інстанса. Код створення цих правил можна побачити на рисунку 4.5. Після створення правил, необхідно підключити сам інстанс до відповідної групи, це робиться за допомогою команди яка відображена на рисунку 4.5.

```hcl
# Визначення провайдера AWS
provider "aws" {
  region = "us-west-2" # Вкажіть потрібний регіон AWS
}

# Ресурс для створення EC2 Instance
resource "aws_instance" "example" {
  ami           = "ami-0c02fb55956c7d316" # Вкажіть AMI ID, актуальний для вашого регіону
  instance_type = "t2.micro"              # Тип EC2 Instance

  # Підключення ключа SSH
  key_name = "my-key-pair"                # Вкажіть ім'я вашого ключа SSH

  # Теги для ідентифікації
  tags = {
    Name = "TerraformInstance"           # Ім'я EC2 Instance
  }
}

# Ресурс для створення Security Group
resource "aws_security_group" "allow_ssh" {
  name        = "allow_ssh"
  description = "Security group to allow SSH access"

  # Правило для дозволу SSH
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]          # Відкрити доступ для всіх IP-адрес (змінюйте для більшої безпеки)
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

# Додавання Security Group до EC2 Instance
resource "aws_instance" "example_with_sg" {
  ami           = "ami-0c02fb55956c7d316" # Вкажіть AMI ID
  instance_type = "t2.micro"              # Тип EC2 Instance
  key_name      = "my-key-pair"           # Ім'я ключа SSH

  # Підключення Security Group
  vpc_security_group_ids = [aws_security_group.allow_ssh.id]

  tags = {
    Name = "TerraformInstanceWithSG"     # Ім'я EC2 Instance
  }
}
```

Рисунок 4.7 – Увесь код Terraform по створенню нового EC2 Instance

```
terraform init
```

Рисунок 4.8 – Ініціалізація Terraform

Рисунок 4.9 – Перегляд запланованих змін

Рисунок 4.10 – Запуск виконання змін

Після того як всі необхідні модулі та параметри були детально прописані у файлі з налаштуваннями (рисунок 4.5), настає момент переходу до наступного етапу роботи з Terraform — збереження файлу та його виконання. Це важливий крок, який дозволяє реалізувати всю створену інфраструктуру у хмарному середовищі AWS.

Файл з розширенням .tf, у якому міститься інфраструктурний код, необхідно зберегти у робочому каталозі, де буде виконуватися подальша робота з Terraform. Він є основою для автоматизації, оскільки містить усі вказівки щодо створення, конфігурації та управління хмарними ресурсами.

Після збереження файлу можна переходити до його виконання. Для цього слід скористатися командним рядком і виконати ряд команд, які забезпечать підготовку середовища Terraform та створення ресурсів:

1. Ініціалізація середовища (рисунок 4.8). Команда terraform init підготує робочий каталог для використання Terraform, завантажуючи необхідні плагіни для роботи з провайдером AWS. Це обов’язковий крок перед виконанням будь-яких інших операцій.
2. Перевірка плану створення ресурсів (рисунок 4.9). Використовуючи команду terraform plan, можна переглянути, які саме ресурси будуть створені, модифіковані чи видалені на основі конфігурації у файлі .tf. Це дозволяє перевірити правильність налаштувань перед їхнім фактичним застосуванням.

3. Створення інфраструктури (рисунок 4.10). За допомогою команди terraform apply запускається процес створення інфраструктури у хмарному середовищі AWS. Ця команда ініціює виконання всіх вказівок з файлу .tf, і після підтвердження система автоматично створить EC2 Instance та інші ресурси відповідно до налаштувань.

Важливим моментом є те, що після запуску файлу всі створені ресурси повністю відповідають заданим параметрам. Крім того, цей підхід забезпечує можливість повторного використання файлу .tf для масштабування чи оновлення інфраструктури в майбутньому.

Таким чином, збереження файлу з налаштуваннями та його запуск є завершальним кроком у процесі автоматизованого налаштування хмарної інфраструктури, що відкриває нові можливості для ефективного управління ресурсами та спрощує подальшу роботу з ними.

РОЗДІЛ 5 ПОРІВНЯННЯ ТРАДИЦІЙНИХ ТА ХМАРНИХ СИСТЕМ ВПРОВАДЖЕННЯ ІНФРАСТРУКТУРИ

5.1 Традиційна система впровадження інфраструктури

На сьогоднішній день все ще актуальною залишається так звана традиційна інфраструктура, яка базується на використанні фізичного серверного обладнання та локально розташованих систем зберігання і обробки даних. Ця модель впровадження включає в себе всі етапи побудови IT-інфраструктури, починаючи від оцінки потреб бізнесу, закупівлі необхідного обладнання та програмного забезпечення, встановлення і налаштування серверів, мережевих пристроїв, систем резервного копіювання, закінчуючи моніторингом, технічним обслуговуванням і поступовим масштабуванням.

Традиційна інфраструктура забезпечує повний контроль над усіма аспектами роботи IT-систем, включаючи безпеку даних, швидкість обробки інформації та фізичний доступ до обладнання. Такий підхід часто використовується великими корпораціями, урядовими установами або підприємствами з високими вимогами до конфіденційності та відповідності нормативним актам.

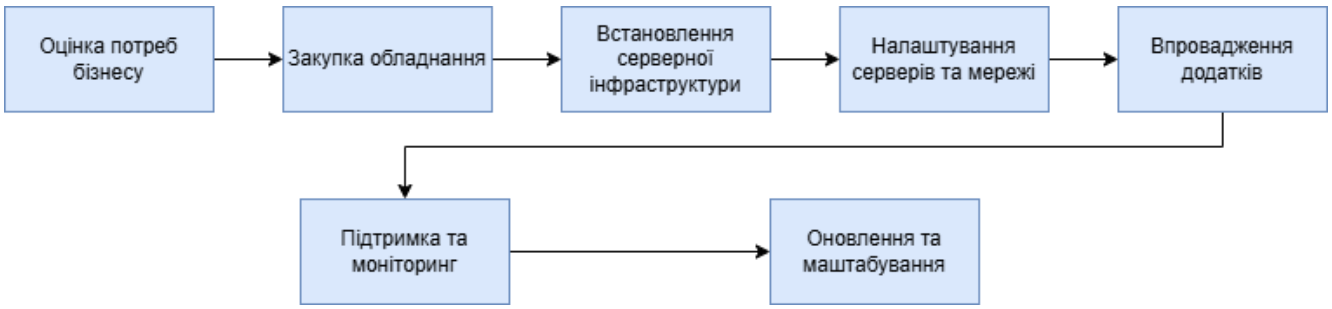


Рисунок 5.1 – Традиційна система впровадження інфраструктури

Традиційна схема впровадження інфраструктури базується на фізичних серверах та їхній локальній підтримці. Цей підхід включає кілька поступових етапів які відображені на рисунку 5.1. Детальний опис кожного з етапів:

1. **Оцінка потреб бізнесу**. На першому етапі компанія визначає основні завдання, які повинна вирішувати інфраструктура, оцінює очікувані навантаження, кількість користувачів та необхідні потужності.
2. **Закупівля обладнання**. Після визначення потреб проводиться вибір та закупівля необхідного серверного обладнання, комунікаційних пристроїв (маршрутизаторів, комутаторів), систем зберігання даних та іншого обладнання. Цей процес може бути витратним як за часом, так і за фінансами.
3. **Встановлення серверної інфраструктури**. Закуплене обладнання встановлюється у виділеному приміщенні, яке повинно відповідати вимогам безпеки та мати відповідні умови для роботи (охолодження, стабільне електроживлення). Це включає монтаж серверів у стійки, підключення до мережі та початкову конфігурацію.
4. **Налаштування серверів та мережі**. На цьому етапі здійснюється програмне налаштування серверів, операційних систем та мережевих пристроїв. Це включає створення мережевої топології, встановлення програмного забезпечення та налаштування баз даних.
5. **Впровадження додатків**. Після налаштування інфраструктури відбувається встановлення бізнес-додатків, які будуть використовуватись на серверах. Це можуть бути CRM-системи, ERP-платформи, системи документообігу та інші програми.
6. **Оновлення та масштабування**. У процесі експлуатації компанія стикається з необхідністю оновлення обладнання або додавання нових ресурсів через зростання навантаження. Це потребує додаткових фінансових витрат та часу на закупівлю нового обладнання.
7. **Підтримка та моніторинг**. Останнім етапом є регулярне обслуговування серверної інфраструктури. Це включає моніторинг роботи серверів, резервне копіювання даних, усунення технічних несправностей та забезпечення безпеки систем.

5.2 Система впровадження хмарної інфраструктури

З роками класична інфраструктура почала все менше відповідати потребам сучасного бізнесу через низку обмежень, які ускладнювали її ефективне використання. До таких обмежень належать високі початкові витрати на обладнання, складність у масштабуванні, обмеженість гнучкості та необхідність у постійному технічному обслуговуванні. Крім того, класична інфраструктура часто вимагала значних витрат часу та ресурсів на модернізацію, що особливо ускладнювало швидку адаптацію до змін ринкових умов або нових викликів.

Сучасні тенденції розвитку бізнесу, які базуються на швидкості, гнучкості та економічній ефективності, змушують компанії все частіше звертатися до хмарних технологій. Хмарна інфраструктура надає можливість створення серверів, які можна легко масштабувати під потреби компанії без необхідності інвестувати в дороге обладнання. Використовуючи моделі, такі як IaaS (інфраструктура як послуга) або PaaS (платформа як послуга), підприємства отримують доступ до обчислювальних потужностей, сховищ даних і програмного забезпечення, які можна використовувати за потребою, оплачуючи лише спожиті ресурси.

Перехід до хмарних серверів не тільки знижує витрати, але й значно підвищує гнучкість і продуктивність. Завдяки хмарним технологіям бізнес може швидко запускати нові проекти, автоматизувати рутинні процеси, покращувати співпрацю між командами та забезпечувати безперебійний доступ до даних і сервісів з будь-якого місця у світі. Таким чином, хмарна інфраструктура стає не просто альтернативою, а необхідною умовою для досягнення конкурентних переваг у сучасному бізнес-середовищі.

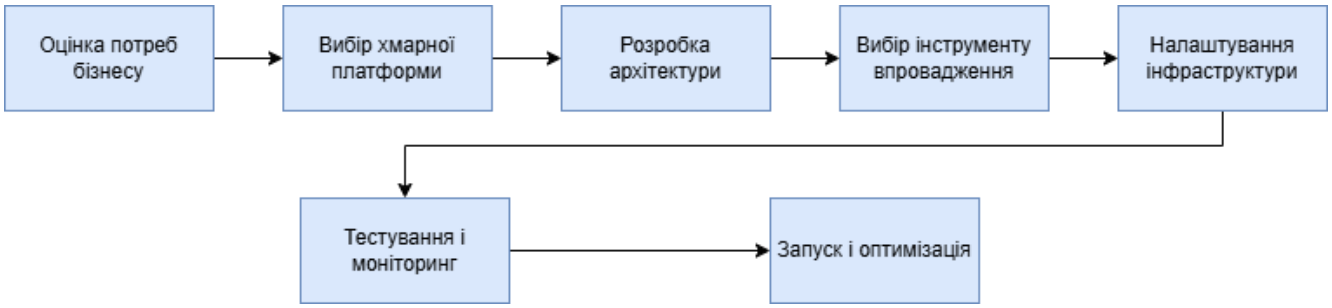


Рисунок 5.2 – Система впровадження хмарної інфраструктури

У схемі на рисунку 5.2 представлені основні етапи впровадження хмарних технологій на підприємстві з використанням двох інструментів: веб-інтерфейсу AWS та Terraform. Схема демонструє інтеграцію цих інструментів для ефективного управління хмарною інфраструктурою, автоматизації процесів та забезпечення масштабованості ресурсів. Розбір всіх етапів впровадження хмари:

1. **Оцінка потреб бізнесу.** На початковому етапі проводиться аналіз вимог компанії. Визначаються основні цілі впровадження хмарних технологій: зменшення витрат, масштабування, підвищення продуктивності або автоматизація. Аналізуються існуючі процеси, обсяги даних, необхідні ресурси та передбачувані навантаження.
2. **Вибір хмарного провайдера та моделі впровадження.** Обирається хмарна платформа (AWS, Azure, GCP тощо) залежно від потреб компанії, а також модель надання хмарних послуг (IaaS, PaaS, SaaS). Визначаються регіони розміщення серверів для оптимізації швидкості доступу та відповідності регуляторним нормам.
3. **Проектування хмарної архітектури.** Розробляється архітектура хмарної інфраструктури, яка включає вибір типів віртуальних машин, сховищ даних, мережевих налаштувань і додаткових сервісів. Передбачаються сценарії масштабування та резервування для забезпечення безперебійної роботи.
4. **Впровадження інфраструктури.** На цьому етапі створюються необхідні ресурси в хмарі. Це може виконуватися через веб-інтерфейс хмарного провайдера (AWS Management Console) або автоматизаційні

інструменти, такі як Terraform. Створюються віртуальні машини, бази даних, мережеві підключення та налаштовуються системи зберігання даних.

5. **Міграція даних та налаштування додатків.** Існуючі дані переносяться на хмарну платформу, а додатки адаптуються до роботи в хмарному середовищі. Водночас проводиться оптимізація додатків для зменшення затримок та підвищення продуктивності.

6. **Налаштування безпеки.** Налаштовуються параметри безпеки, такі як політики доступу, використання шифрування даних, створення ключів SSH, контроль мережевого доступу через Security Groups та аудит. Забезпечується відповідність стандартам безпеки та конфіденційності.

7. **Тестування інфраструктури.** Виконується тестування нової інфраструктури для перевірки її працездатності, продуктивності та безпеки. Виявлені проблеми усуваються, а налаштування оптимізуються.

8. **Моніторинг та підтримка.** Після запуску хмарної інфраструктури впроваджуються інструменти моніторингу для контролю за її станом і продуктивністю. Налаштовуються системи сповіщення про можливі помилки або перевантаження. Проводиться регулярна підтримка, масштабування або оновлення системи.

5.3 Система впровадження гібридної інфраструктури

Гібридна система впровадження — це підхід, який поєднує локальні сервери (*on-premises*) з хмарними платформами, забезпечуючи гнучкість, надійність та оптимізацію витрат на інфраструктуру. Така система дозволяє підприємствам комбінувати можливості обох середовищ, що особливо актуально для організацій з високими вимогами до безпеки даних та масштабування обчислювальних ресурсів.

Основна ідея гібридної системи полягає в тому що дана система дозволяє розподіляти навантаження між локальною інфраструктурою та хмарою, що допомагає бізнесу ефективно керувати ресурсами. Чутливі або критично важливі

дані залишаються на локальних серверах для забезпечення безпеки, контролю та відповідності вимогам регулювання. Менш критичні завдання, які потребують високої продуктивності, гнучкості або масштабування, передаються на хмарну платформу.

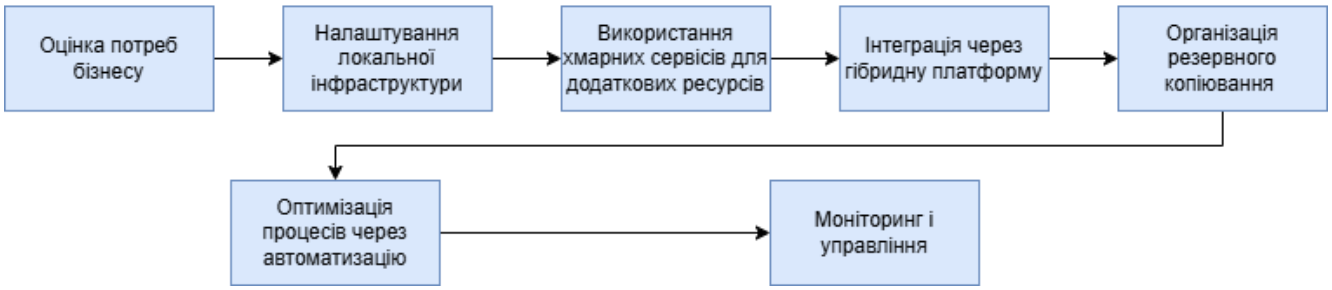


Рисунок 5.3 – Система впровадження хмарної інфраструктури

На рисунку 5.3 представлено схему впровадження гібридної інфраструктури, яка демонструє поетапний процес інтеграції локальних ресурсів з хмарними технологіями. Ця модель побудована на декількох ключових етапах, кожен з яких є невіддільною частиною загальної стратегії реалізації гібридної архітектури.

1. Перший етап – Аналіз потреб бізнесу та класифікація робочих навантажень. На цьому етапі проводиться детальне вивчення вимог підприємства, що включає аналіз поточних бізнес-процесів, оцінку робочих навантажень, визначення критично важливих задач та чутливих даних. Основною метою є розділення робочих навантажень на ті, які краще обробляти локально, та ті, які ефективніше перемістити до хмарної інфраструктури. Чутливі дані чи операції з високим рівнем безпеки зберігаються локально, тоді як менш критичні процеси та змінні навантаження передаються у хмару. Окрім цього, на даному етапі проводиться оцінка витрат, ризиків та вигод, які можуть виникнути при впровадженні гібридної системи.
2. Другий етап – Налаштування локальної інфраструктури. Після аналізу бізнес-потреб починається процес підготовки локальних ресурсів. Це включає налаштування серверів, систем зберігання даних, мережевого

обладнання та забезпечення захисту інформації. Важливою складовою є інтеграція з хмарними платформами за допомогою рішень, як-от VPN або спеціалізованих інструментів (наприклад, AWS Outposts, Azure Stack чи Google Anthos). На цьому етапі забезпечується злагоджена робота локальних ресурсів, що дозволяє підготувати інфраструктуру до подальшої інтеграції з хмарою.

3. Третій етап – Використання хмарних сервісів для додаткових ресурсів. На даному етапі організація починає використовувати хмарні технології для оптимізації роботи та розширення можливостей. Хмарні платформи дозволяють обробляти пікові навантаження, масштабувати ресурси за потребою, а також зберігати значні обсяги даних у сховищах, таких як Amazon S3 чи Azure Blob Storage. Крім того, хмарні обчислювальні ресурси (наприклад, Amazon EC2 або Google Compute Engine) надають додаткові можливості для виконання ресурсомістких завдань, які не потребують локального зберігання.

4. Четвертий етап – Інтеграція через гібридну платформу. Цей етап спрямований на об’єднання локальної інфраструктури з хмарним середовищем. Використовуються спеціалізовані платформи, такі як Azure Arc, Google Anthos чи VMware Cloud, які дозволяють централізовано керувати всією інфраструктурою. Платформи забезпечують синхронізацію даних, спрощують управління ресурсами та оптимізують процеси перенесення даних між локальними та хмарними середовищами.

5. П’ятий етап – Організація резервного копіювання та відновлення даних. На цьому етапі налаштовується система резервного копіювання та відновлення даних для забезпечення їх надійності та безперервності бізнес-процесів. Використовуються як локальні, так і хмарні сховища для зберігання копій даних. У разі збоїв чи втрат даних хмарні сервіси дозволяють швидко відновити роботу завдяки аварійним копіям, що мінімізує простої та фінансові збитки.

6. Шостий етап – Оптимізація процесів через автоматизацію. Цей етап передбачає автоматизацію рутинних завдань для підвищення ефективності управління інфраструктурою. Застосовуються інструменти, такі як Terraform, Ansible чи CloudFormation, які дозволяють автоматично налаштовувати, масштабувати та керувати ресурсами. Це знижує ймовірність людських помилок, прискорює розгортання інфраструктури та забезпечує гнучкість у реагуванні на зміни в робочих навантаженнях.
7. Сьомий етап – Моніторинг і управління. На заключному етапі налаштовуються системи моніторингу для постійного спостереження за продуктивністю інфраструктури. Використовуються інструменти, такі як AWS CloudWatch, Azure Monitor чи Prometheus, для аналізу ефективності роботи ресурсів, контролю витрат та оперативного реагування на неполадки. Моніторинг допомагає визначити вузькі місця, оптимізувати витрати та підвищити загальну надійність системи.

Таким чином, схема впровадження гібридної інфраструктури є комплексною та багаторівневою. Вона поєднує переваги локальної інфраструктури з можливостями хмарних сервісів, дозволяючи підприємствам забезпечити високу продуктивність, масштабованість та надійність у своїй роботі.

5.4 Порівняння систем

Впровадження хмарних технологій та традиційної серверної інфраструктури представляє два різні підходи до управління IT-ресурсами підприємства, кожен з яких має свої переваги та недоліки. Традиційні серверні системи передбачають фізичне розміщення серверів на території підприємства, що вимагає значних капіталовкладень у закупівлю обладнання, його встановлення та обслуговування. Крім того, масштабування таких систем є трудомістким процесом, що потребує додаткових фінансових витрат і часу для встановлення нових серверів або оновлення існуючих.

На відміну від цього, хмарні технології пропонують гнучкість та масштабованість, які значно спрощують адаптацію IT-інфраструктури до змінних потреб бізнесу. Хмарні сервіси дозволяють підприємствам швидко збільшувати або зменшувати обсяг використовуваних ресурсів без необхідності фізичного втручання, що забезпечує економію часу та зниження витрат. Крім того, хмарні платформи, такі як AWS, Azure та Google Cloud, надають широкий спектр сервісів, включаючи обчислювальні потужності, зберігання даних, аналітику та інструменти для машинного навчання, що дозволяє підприємствам ефективно використовувати сучасні технології без необхідності глибоких технічних знань.

$$\text{Економія витрат (\%)} = \frac{\text{Витрати на традиційну інфраструктуру} - \text{Витрати на хмарну інфраструктуру}}{\text{Витрати на традиційну інфраструктуру}} \times 100$$

Рисунок 5.4 – Формула розрахунку економії витрат

Впровадження хмарних технологій значно підвищує ефективність підприємств у порівнянні з традиційними серверними системами. Наприклад, за даними досліджень, компанії, які перейшли на хмарні рішення, спостерігають зниження витрат на IT-інфраструктуру приблизно на 30-50%. Дані значення отримуються за формулою з рисунку 5.3. Це обумовлено відсутністю необхідності інвестувати у фізичне обладнання, а також зменшенням витрат на його обслуговування та оновлення.

Однією з ключових переваг хмарних технологій є їхня здатність забезпечувати високу доступність та надійність завдяки розподіленій архітектурі. Хмарні провайдери використовують резервні копії даних та географічно розподілені дата-центри, що мінімізує ризик втрати даних та забезпечує безперервність бізнес-процесів навіть у разі відмови окремих компонентів системи. Традиційні серверні системи, навпаки, часто піддаються ризикам, пов'язаним з фізичними пошкодженнями обладнання або природними катастрофами, що може призвести до серйозних перебоїв у роботі підприємства.

$$\text{Покращення швидкості масштабування (\%)} = \frac{\text{Час для традиційної інфраструктури} - \text{Час для хмарної інфраструктури}}{\text{Час для традиційної інфраструктури}} \times 100$$

Рисунок 5.5 – Формула розрахунку покращення масштабування

Щодо масштабованості, хмарні платформи дозволяють збільшувати або зменшувати обсяг ресурсів за потребою протягом кількох хвилин, що при розрахунку за формулою з рисунку 5.4 збільшує до 70% швидкість реагування на зміни навантаження порівняно з традиційними серверними системами, де масштабування може займати дні або навіть тижні.

Автоматизація процесів за допомогою інструментів, таких як Terraform, сприяє підвищенню продуктивності праці до 40%, оскільки значно зменшує час, необхідний для розгортання та налаштування інфраструктури. Це також мінімізує ризик людських помилок, що підвищує загальну надійність системи на 25%.

У сфері управління ресурсами хмарні технології забезпечують до 60% кращу видимість та контроль над використанням IT-ресурсів завдяки інтегрованим інструментам моніторингу та аналітики. Це дозволяє підприємствам ефективніше планувати та оптимізувати витрати, а також швидше виявляти та вирішувати проблеми.

$$\text{Покращення безпеки (\%)} = \frac{\text{Кількість інцидентів у традиційній інфраструктурі} - \text{Кількість інцидентів у хмарній інфраструктурі}}{\text{Кількість інцидентів у традиційній інфраструктурі}} \times 100$$

Рисунок 5.6 – Формула розрахунку покращення безпеки

З точки зору безпеки, хмарні технології також пропонують сучасні рішення для захисту даних, включаючи шифрування, багаторівневу автентифікацію та автоматизовані системи моніторингу. Хоча традиційні серверні системи дозволяють підприємствам мати повний контроль над своїми даними, це часто вимагає значних ресурсів для впровадження та підтримки відповідних заходів безпеки.

$$\text{Доступність (\%)} = \frac{\text{Час безперебійної роботи}}{\text{Загальний час}} \times 100$$

Рисунок 5.7 – Формула розрахунку доступності

Також до безпеки можна віднести і доступність до серверів. Хмарні платформи пропонують до 99.99% часу безперебійної роботи завдяки розподіленій архітектурі та резервним копіям даних у різних регіонах. У порівнянні, традиційні серверні системи зазвичай забезпечують до 99.5% доступності, що може призводити до частіших збоїв та втрати даних. Така різниця пов’язана з основним показником для розрахунку з рисунку 5.6, часом безперебійної роботи. Так як при класичній інфраструктурі немає можливості перенести все на інший сервер і доводиться проводити виправлення на проді.

Безпека даних у хмарних середовищах також демонструє покращення до 35% завдяки сучасним механізмам шифрування, багаторівневій автентифікації та автоматизованим системам моніторингу. Традиційні серверні системи, хоча й забезпечують повний контроль над даними, часто потребують значних ресурсів для впровадження подібних заходів безпеки.

Водночас, традиційні серверні системи можуть бути більш вигідними для підприємств з передбачуваними та стабільними навантаженнями, де інвестиції в власну інфраструктуру можуть окупитися протягом тривалого часу. Хмарні технології, хоча й пропонують більшу гнучкість, можуть виявитися менш економічно вигідними для малих підприємств з обмеженим бюджетом, які не потребують великої кількості ресурсів або складних конфігурацій.

Таблиця 5.1 – порівняльна таблиця традиційної та хмарної системи

Параметр	Традиційна інфраструктура	Хмарна інфраструктура	Різниця
Початкові витрати	$100,000 – $500,000 на обладнання	$30,000 – $50,000 на впровадження	Зниження на 50–70%
Щорічні витрати	$20,000 – $50,000 на обслуговування	$10,000 – $20,000	Зниження на 40–60%

Продовження таблиця 5.1 – порівняльна таблиця традиційної та хмарної системи

Параметр	Традиційна інфраструктура	Хмарна інфраструктура	Різниця
Час масштабування	2–4 тижні для додавання обладнання	5–10 хвилин для збільшення хмарних ресурсів	Скорочення часу на 80–90%
Доступність	99.5% часу безперебійної роботи	99.99% завдяки резервуванню	Підвищення доступності на 0.49%
Продуктивність	До 10,000 запитів/сек при пікових навантаженнях	До 100,000 запитів/сек при масштабуванні	Зростання продуктивності в 10 разів
Безпека даних	Ручне налаштування, витрати $5,000–$10,000/рік	Вбудовані механізми безпеки за $2,000–$5,000/рік	Зниження витрат на 50%
Гнучкість впровадження	2–3 місяці для запуску нових послуг	1–2 дні для розгортання нових середовищ	Скорочення часу на 90–95%
Автоматизація	До 10% завдань автоматизовано	До 70% завдань автоматизовано	Зростання автоматизації на 60%

Таким чином, вибір між впровадженням хмарних технологій та традиційною серверною інфраструктурою залежить від конкретних потреб та можливостей підприємства. Як видно з таблиці 5.1 хмарні технології забезпечують більшу гнучкість, масштабованість та доступність сучасних IT-ресурсів, що робить їх ідеальним вибором для динамічно розвиваючихся компаній. Традиційні серверні системи можуть залишатися актуальними для підприємств з стабільними потребами та бажанням мати повний контроль над своєю IT-інфраструктурою.

Також важливо здійснити детальний аналіз та порівняння гібридної системи впровадження інфраструктури із традиційною (класичною) моделлю. Таке порівняння дозволяє виявити переваги та недоліки кожного з підходів, оцінити їхню ефективність, вартість, безпеку, продуктивність та масштабованість. Гібридна система, яка поєднує локальну інфраструктуру з хмарними технологіями, надає можливість комбінувати найкращі аспекти обох підходів. З іншого боку, класична система, що базується виключно на локальних ресурсах, продовжує залишатися актуальною для організацій із жорсткими вимогами до зберігання та обробки даних.

Порівняння цих двох моделей допоможе визначити, який підхід є оптимальним для конкретних бізнес-потреб, враховуючи сучасні тенденції автоматизації, цифровізації та зростаючі вимоги до адаптивності IT-інфраструктури. Аналіз має охоплювати аспекти витрат на впровадження та підтримку, швидкість масштабування, рівень безпеки, стабільності, гнучкості та можливості інтеграції з сучасними рішеннями. Такий підхід дозволить сформувати цілісне уявлення про потенціал обох моделей і зробити обґрунтований вибір для підприємства.

Порівняння традиційної та гібридної інфраструктури впровадження дає змогу оцінити, наскільки кожен підхід відповідає сучасним викликам і потребам бізнесу. Традиційна модель, що базується на локальних серверах, залишається популярною завдяки своєму повному контролю над інфраструктурою. Водночас гібридна модель пропонує новий рівень ефективності завдяки інтеграції локальних і хмарних ресурсів, що робить її більш адаптивною та економічно вигідною.

З точки зору витрат, традиційна інфраструктура вимагає значних початкових інвестицій у придбання серверного обладнання, систем зберігання даних, програмного забезпечення та інфраструктури для центрів обробки даних (ЦОД). До цього додаються витрати на енергоспоживання та оплату праці персоналу для обслуговування. У середньому початкові витрати становлять 60–70% від загального бюджету, а щорічні витрати на підтримку складають ще 20–30%. Гібридна інфраструктура значно знижує ці витрати завдяки використанню хмарних ресурсів для нестабільних навантажень і задач, які не потребують локального розміщення. Початкові витрати в цьому випадку зменшуються на 30–40%, а автоматизація процесів дозволяє оптимізувати витрати на обслуговування.

Масштабованість є ще однією важливою характеристикою. У традиційній інфраструктурі масштабування залежить від наявного обладнання, і збільшення ресурсів потребує значного часу й інвестицій. Зазвичай цей процес займає кілька тижнів або навіть місяців. У гібридній моделі масштабування досягається швидко та гнучко: хмарні ресурси можуть бути активовані за лічені хвилини, що дозволяє підприємствам оперативно реагувати на зміни навантаження.

Надійність і доступність також суттєво відрізняються. Традиційна інфраструктура зазвичай залежить від кількох локальних ЦОД, що робить її вразливою до збоїв або катастрофічних подій. У разі відсутності резервування рівень доступності може досягати лише 99.5%, що призводить до простоїв і втрати даних. Гібридна інфраструктура поєднує локальну надійність із резервуванням у хмарі, забезпечуючи доступність до 99.99% завдяки розподіленій архітектурі та автоматизованим механізмам аварійного відновлення.

Безпека даних у традиційній інфраструктурі залежить від ресурсів компанії для впровадження сучасних механізмів шифрування, моніторингу та захисту. Однак цей процес часто є дорогим і технічно складним. У гібридній моделі локальний контроль над критичними даними поєднується з перевагами сучасних хмарних технологій, таких як багаторівневе шифрування та автоматизоване виявлення загроз.

Гнучкість і автоматизація є слабкими сторонами традиційної інфраструктури. Налаштування нових серверів або послуг займає багато часу й вимагає значних зусиль. Гібридна модель надає більшу гнучкість завдяки можливості використовувати хмарні ресурси під конкретні завдання, а інструменти автоматизації, як-от Terraform чи Ansible, дозволяють швидко налаштовувати та керувати інфраструктурою.

Таблиця 5.2 – порівняльна таблиця традиційної та гібридної систем

Показник	Традиційна інфраструктура	Гібридна інфраструктура
Початкові витрати	100%	Зменшення на 30-40%
Час масштабування	100% (тижні/місяці)	Зменшення на 70-80% (хвилини)
Доступність	99.5%	До 99.99%
Гнучкість	Обмежена	Висока
Резервне копіювання	Дороге та складне	Автоматизоване та ефективне
Безпека	Потребує додаткових ресурсів	Підвищена на 20-30%

Таким чином, гібридна інфраструктура, поєднуючи переваги локальних і хмарних рішень, і як видно з порівняльної таблиці 5.2 є більш адаптивною, економічною та ефективною у порівнянні з традиційною моделлю. Це робить її ідеальним вибором для сучасних підприємств, які прагнуть досягти гнучкості, надійності та конкурентоспроможності.

Таблиця 5.3 – порівняльна таблиця запропонованих систем

Параметр	Традиційна інфраструктура	Хмарна інфраструктура	Гібридна інфраструктура
Початкові витрати	$100,000 – $500,000	$30,000 – $50,000	$50,000 – $100,000
Щорічні витрати	$20,000 – $50,000	$10,000 – $20,000	$15,000 – $25,000
Час масштабування	2–4 тижні	5–10 хвилин	1–3 дні для інтеграції нових сервісів
Доступність	99.5%	99.99%	99.9%
Продуктивність	До 10,000 запитів/сек	До 100,000 запитів/сек	До 50,000 запитів/сек
Безпека даних	Потребує значних інвестицій	Вбудовані рішення за $2,000–$5,000/рік	Поєднання локального контролю та хмарної безпеки
Гнучкість впровадження	2–3 місяці для запуску нових послуг	1–2 дні для розгортання нових середовищ	1–2 тижні для налаштування комплексної системи
Автоматизація	До 10% завдань автоматизовано	До 70% завдань автоматизовано	До 50% завдань автоматизовано
Масштабованість	Обмежена фізичними ресурсами	Безмежна завдяки хмарним обчисленням	Обмежена локально, але гнучка у хмарі
Резервування даних	Потребує додаткових інвестицій	Автоматичне резервування	Поєднання локального резервування та хмарного
Швидкість впровадження	Тривалий (до кількох місяців)	Швидкий (до кількох годин)	Помірний (до кількох днів або тижнів)
Ефективність витрат	Високі початкові та операційні витрати	Зниження витрат на 40–70%	Збалансовані витрати між локальними та хмарними ресурсами

Традиційна інфраструктура залишається актуальною для компаній, які потребують повного контролю над своїми ресурсами, особливо у випадках роботи

з чутливими даними або специфічними регуляторними вимогами. Проте її висока вартість і тривалий час масштабування роблять цей підхід менш ефективним у сучасних умовах швидких змін та зростаючих навантажень. Хмарна інфраструктура, навпаки, пропонує значну економію витрат, швидкість впровадження та легкість масштабування. Вона забезпечує високу доступність і продуктивність, але може бути обмеженою у випадках роботи з критично важливими або конфіденційними даними, що потребують локального зберігання.

Гібридна інфраструктура поєднує переваги обох підходів, дозволяючи підприємствам зберігати чутливі дані локально, забезпечуючи їхню безпеку, і водночас використовувати можливості хмар для забезпечення гнучкості, масштабованості та обробки нестабільних або пікових навантажень. Такий підхід є оптимальним рішенням для компаній, які прагнуть балансувати між ефективністю, безпекою та витратами, водночас забезпечуючи стабільну основу для подальшого зростання та розвитку.

ВИСНОВОК

У підсумку можна стверджувати, що хмарні технології стали одним із ключових елементів у сучасному бізнес-середовищі. Вони забезпечують ефективну оптимізацію бізнес-процесів, суттєве зниження витрат на інфраструктуру, підвищення продуктивності підприємств і відкривають нові можливості для адаптації до змін ринкових умов. Завдяки хмарним технологіям компанії отримують доступ до масштабованих систем, здатних задовольняти зростаючі потреби бізнесу, автоматизувати рутинні процеси та підвищувати загальну ефективність роботи.

У межах цієї роботи було детально розглянуто два підходи до впровадження хмарної інфраструктури: через web-інтерфейс постачальника послуг та за допомогою інструмента автоматизації Terraform. Web-інтерфейс є зручним інструментом для невеликих проєктів або первинного ознайомлення з можливостями хмарної платформи. Він не потребує глибоких технічних знань і забезпечує швидкий старт завдяки інтуїтивному користувацькому інтерфейсу. Однак цей підхід менш ефективний при роботі з великою кількістю ресурсів, оскільки потребує значних ручних зусиль для налаштування та оновлення інфраструктури.

Terraform, як інструмент інфраструктури як коду, виявився значно ефективнішим у середніх і великих проєктах, де важливі автоматизація, масштабованість і точність. Цей підхід дозволяє зберігати конфігурації інфраструктури у вигляді коду, що спрощує управління змінами, забезпечує повторюваність процесів і знижує ризики людських помилок. Використання Terraform також надає можливість інтеграції з іншими інструментами управління та моніторингу, що робить його універсальним рішенням для побудови сучасної хмарної інфраструктури.

Крім того, у роботі розглянуто й порівняно традиційну, хмарну та гібридну інфраструктури. Традиційна інфраструктура підходить для організацій, які потребують повного контролю над своїми ресурсами, проте її високі початкові

витрати, повільне масштабування та низька гнучкість значно обмежують її застосування в умовах сучасного бізнесу. Хмарна інфраструктура забезпечує найвищу ефективність у масштабованості, продуктивності та доступності з мінімальними витратами, але має певні обмеження у сфері обробки чутливих даних. Гібридна інфраструктура, своєю чергою, є оптимальним компромісом, дозволяючи поєднувати переваги локальних серверів для критичних завдань із можливостями хмар для масштабування та гнучкості.

Таким чином, впровадження хмарних технологій за допомогою автоматизованих інструментів, таких як Terraform, стає стратегічно важливим для підприємств, які прагнуть ефективно реагувати на виклики сучасного бізнес-середовища. Цей підхід дозволяє компаніям мінімізувати витрати, оптимізувати процеси, підвищити продуктивність і зосередитися на досягненні стратегічних цілей, залишаючи управління інфраструктурою надійним і перевіреним технологіям.

ПЕРЕЛІК ПОСИЛАНЬ

1 ЩО ТАКЕ ХМАРНІ ТЕХНОЛОГІЇ І НАВІЩО ВОНИ ПОТРІБНІ [Електронний ресурс] - https://edin.ua/shho-take-xmarni-texnologi%D1%97-i-navishho-voni-potribni/ [1]

2 Порівняння табличка: AWS vs Azure vs GCP [Електронний ресурс] - https://smartnet.ua/porivnjannja-tablichka-aws-vs-azure-vs-gcp/ [2]

3 Початок роботи для IT-операторів Azure [Електронний ресурс] - https://learn.microsoft.com/ru-ru/azure/guides/operations/azure-operations-guide [3]

4 Початок роботи з AWS [Електронний ресурс] - https://aws.amazon.com/ru/getting-started/ [4]

5 Хмарні обчислення за допомогою AWS [Електронний ресурс] - https://aws.amazon.com/ru/what-is-aws/?nc1=f_cc [5]

6 Getting Started with the AWS Management Console [Електронний ресурс] - https://aws.amazon.com/ru/getting-started/hands-on/getting-started-with-aws-management-console/?pg=gs&sec=gtkaws [6]

7 Регіони та зони доступності [Електронний ресурс] - https://aws.amazon.com/ru/about-aws/global-infrastructure/regions_az/ [7]

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра комп'ютерної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня магістра

НА ТЕМУ: «Система впровадження хмарних технологій для підвищення ефективності підприємства»

Виконав студент групи КСДМ-62
Бородін Назар Володимирович

Керівник дипломної роботи:
Торошенко Ярослав Іванович,
канд.техн.наук, доцент

Київ – 2025

Об'єкт дослідження – Налаштування хмарного сервера за допомогою інструментів web-сторінки та Terraform

Предмет дослідження – методи та технології впровадження хмарних обчислень для підвищення ефективності роботи підприємств, які базуються на використанні сучасних обчислювальних платформ та автоматизованих рішень.

Мета роботи – Налаштувати хмарні сервіси на платформі AWS від імені адміністратора, забезпечивши швидший процес впровадження завдяки використанню інструменту Terraform

2

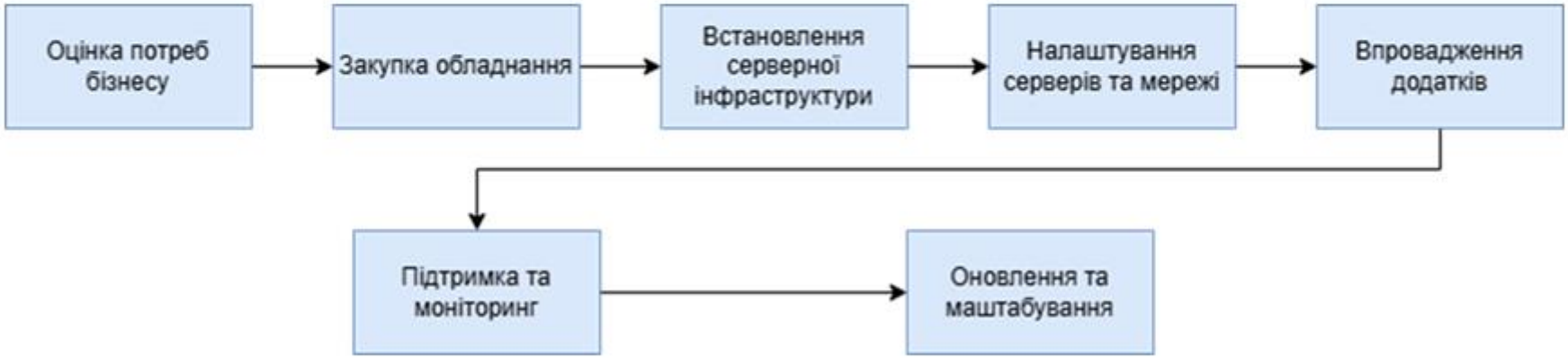
АКТУАЛЬНІСТЬ ТЕМИ РОБОТИ

У роботі розглянуто впровадження хмарних технологій для підвищення ефективності підприємств, зокрема, використання провідних платформ AWS, Microsoft Azure та Google Cloud Platform.

Проаналізовано сучасні тенденції у впровадженні хмарних обчислень, а також обмеження традиційних IT-інфраструктур, які потребують значних ресурсів для їх створення та підтримки. Хмарні технології забезпечують гнучкість, масштабованість і доступність, дозволяючи компаніям адаптуватися до швидкозмінних ринкових умов. Використання автоматизації через інструменти, такі як Terraform, мінімізує людські помилки, спрощує управління ресурсами та оптимізує витрати.

Запропоноване рішення на основі хмарних технологій дозволяє ефективно використовувати обчислювальні ресурси, знижувати витрати на підтримку локальної інфраструктури та покращувати продуктивність бізнес-процесів. Такий підхід є актуальним для підприємств різного масштабу, оскільки відповідає вимогам сучасного бізнесу щодо швидкості, безпеки та економічної ефективності.

Система впровадження класичної інфраструктури



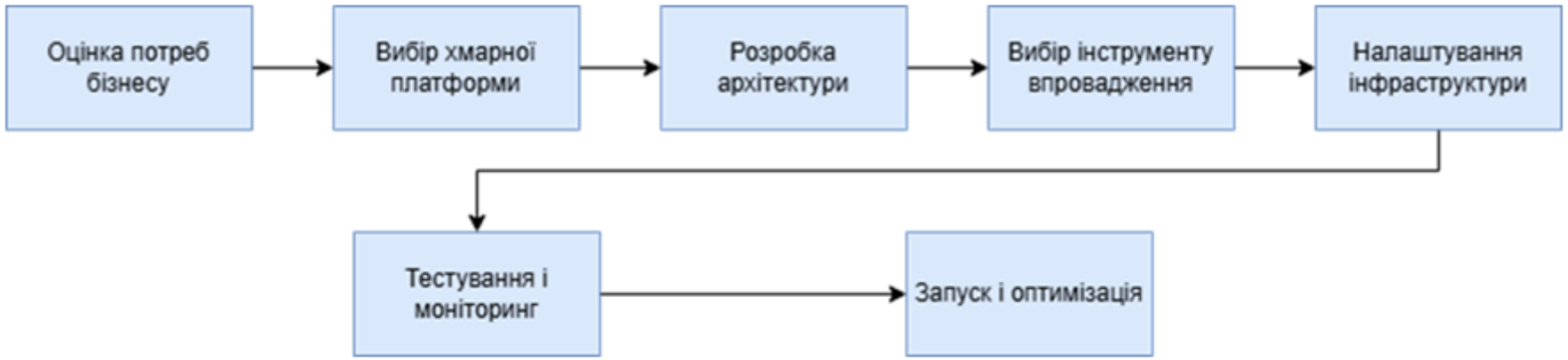
4

Проблеми класичної інфраструктури

- Високі початкові витрати
- Низька масштабованість та гнучкість
- Високі витрати на обслуговування
- Обмеження у доступності та надійності

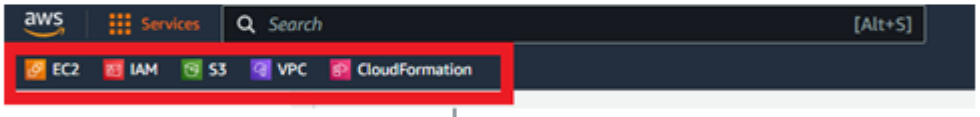
5

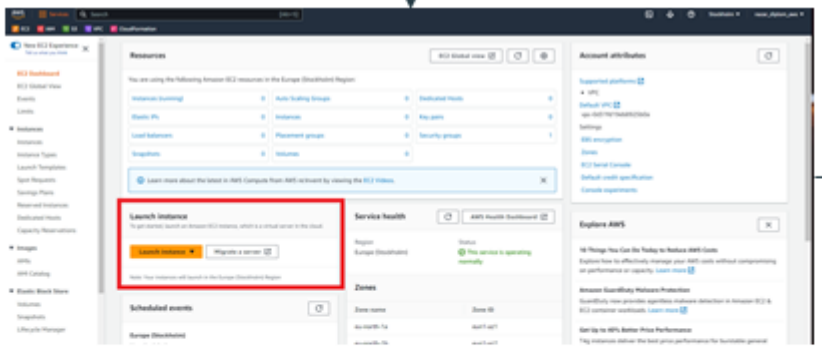
Система впровадження хмарної інфраструктури

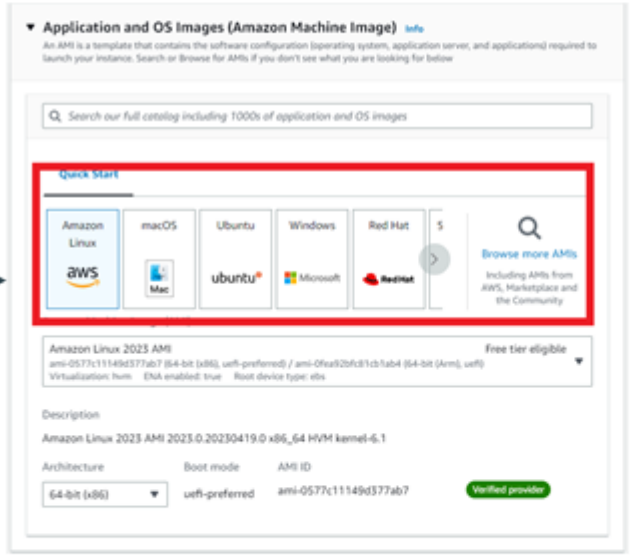


6

AWS. Створення EC2







7

Створення EC2 за допомогою Terraform

```
# Визначення провайдера AWS
provider "aws" {
  region = "us-west-2" # Вкажіть потрібний регіон AWS
}
```

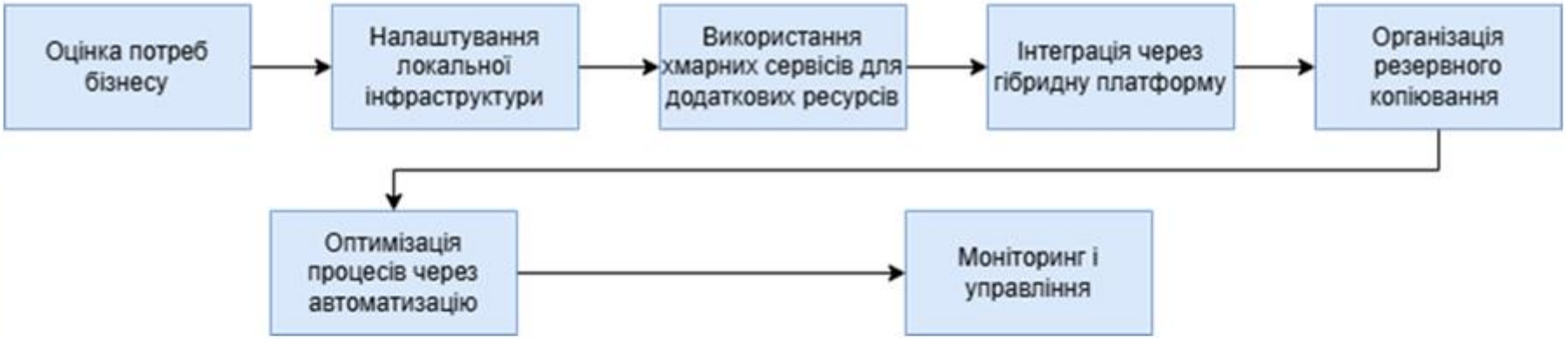
```
resource "aws_instance" "example" {
  ami           = "ami-0c55b159cbfafe1f0" # Ідентифікатор AMI
  instance_type = "t2.micro"              # Тип інстанса
}
```

```
resource "aws_security_group" "example" {
  name_prefix = "example-security-group"

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Доступ через SSH
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # HTTP-доступ
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"] # Дозвіл на вихідний трафік
  }
}
```

```
# Визначення провайдера AWS
provider "aws" {
  region = "us-west-2" # Вкажіть потрібний регіон AWS
}

# Ресурс для створення EC2 Instance
resource "aws_instance" "example" {
  ami           = "ami-0c02fb55956c7d316" # Вкажіть AMI ID, актуальний для вашого регіону
  instance_type = "t2.micro"              # Тип EC2 Instance

  # Підключення ключа SSH
  key_name = "my-key-pair"                # Вкажіть ім'я вашого ключа SSH

  # Теги для ідентифікації
  tags = {
    Name = "TerraformInstance"            # Ім'я EC2 Instance
  }
}

# Ресурс для створення Security Group
resource "aws_security_group" "allow_ssh" {
  name        = "allow_ssh"
  description = "Security group to allow SSH access"

  # Правило для дозволу SSH
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]           # Відкриває доступ для всіх IP-адрес (змініть для більшої безпеки)
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

# Додавання Security Group до EC2 Instance
resource "aws_instance" "example_with_sg" {
  ami           = "ami-0c02fb55956c7d316" # Вкажіть AMI ID
  instance_type = "t2.micro"              # Тип EC2 Instance
  key_name      = "my-key-pair"           # Ім'я ключа SSH

  # Підключення Security Group
  vpc_security_group_ids = [aws_security_group.allow_ssh.id]

  tags = {
    Name = "TerraformInstanceWithSG"      # Ім'я EC2 Instance
  }
}
```

8

Система впровадження гібридної інфраструктури



9

Порівняння систем впровадження інфраструктури

Параметр	Традиційна інфраструктура	Хмарна інфраструктура	Гібридна інфраструктура
Початкові витрати	$100,000 – $500,000	$30,000 – $50,000	$50,000 – $100,000
Щорічні витрати	$20,000 – $50,000	$10,000 – $20,000	$15,000 – $25,000
Час масштабування	2–4 тижні	5–10 хвилин	1–3 дні для інтеграції нових сервісів
Продуктивність	До 10,000 запитів/сек	До 100,000 запитів/сек	До 50,000 запитів/сек
Гнучкість впровадження	2–3 місяці для запуску нових послуг	1–2 дні для розгортання нових середовищ	1–2 тижні для налаштування комплексної системи
Автоматизація	До 10% завдань автоматизовано	До 70% завдань автоматизовано	До 50% завдань автоматизовано
Масштабованість	Обмежена фізичними ресурсами	Безмежна завдяки хмарним обчисленням	Обмежена локально, але гнучка у хмарі
Резервування даних	Потребує додаткових інвестицій	Автоматичне резервування	Поєднання локального резервування та хмарного
Ефективність витрат	Високі початкові та операційні витрати	Зниження витрат на 40–70%	Збалансовані витрати між локальними та хмарними ресурсами

10

Висновки

Хмарні технології стали невід'ємною частиною сучасного бізнесу, забезпечуючи оптимізацію процесів, зниження витрат і гнучкість. У роботі порівняно традиційну, хмарну та гібридну інфраструктури. Традиційна модель, хоч і забезпечує повний контроль, є затратною та обмеженою в масштабуванні. Хмарна інфраструктура пропонує максимальну гнучкість і продуктивність із меншими витратами, але менш ефективна для чутливих даних. Гібридна модель поєднує переваги обох підходів, забезпечуючи баланс між безпекою, гнучкістю та доступністю. Такий підхід стає оптимальним вибором для багатьох підприємств, що прагнуть ефективно адаптуватися до сучасних викликів.