МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
Кафедра Комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
_____Аліна САВЧЕНКО
«___»_____2022 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ДИПЛОМНА РОБОТА, ПОЯСНЮВАЛЬНА ЗАПИСКА)
ВИПУСКНИЦІ ОСВІТНЬОГО СТУПЕНЯ **"МАГІСТР"**
ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ
"ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: "Безвідмовне середовище на базі Amazon Web Services"

Виконавець: студентка групи УС-212М Гасанова Аліса Мавсумівна

Керівник: к.т.н., доцент Холявкіна Тетяна Володимирівна

Нормоконтролер: Ігор РАЙЧЕВ

Київ 2022

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *Кібербезпеки, комп'ютерної та програмної інженерії*

Кафедра *Комп'ютерних інформаційних технологій*

Галузьзнань, спеціальність, освітньо-професійна програма: 12 «Інформаційні технології», 122 «Комп'ютерні науки», «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
Завідувач випускової кафедри
_____Аліна САВЧЕНКО
«_____»_____ 2022 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студентки

Гасанової Аліси Мавсумівни
(прізвище, ім'я, по батькові)

1. Тема роботи: «Безвідмовне середовище на базі Amazon Web Services» затверджена наказом ректора № 1774/ст від 28.09.2022р.

2. Термін виконання роботи: з 26 вересня 2022 року по 21 листопада 2022 року.

3. Вихідні дані до роботи: мова розробки додатку – Python, операційна система – Linux, хмарна платформа – AWS, технологія – Docker.

4. Зміст пояснювальної записки: вступ, аналітичний огляд і постановка завдання, хмарні платформи, технології створення інфраструктури, практичний приклад створення інфраструктури, висновки.

5. Перелік обов'язкового графічного матеріалу: модель роботи технології Docker, зображення тестів , ілюстровані пояснення.

6. Календарний план-графік

№ п/п	*Завдання*	*Термін виконання*	*Підпис керівника*
1.	Проаналізувати літературу та джерела за темою дипломної роботи.	26.09.22 – 01.10.22р.	
2.	Розроблення та затвердження плану дипломної роботи.	02.10.22 – 04.10.22р.	
3.	Привести консультації з науковим керівником щодо створення першого розділу.	05.10.22 – 06.10.22р.	
4.	Розробка розділу 1.	07.10.22 –14.10.22р.	
5.	Розробка розділу 2.	17.10.22 – 29.10.22р.	
6.	Розробка розділу 3.	31.10.22 – 11.11.22р.	
7.	Висновки та оформлення пояснювальної записки дипломної роботи.	14.11.22 – 16.11.22р.	
8.	Підписання необхідних документів у встановленому порядку.	16.11.22 –17.11.22р.	
9.	Підготовка до захисту та попередній захист дипломного проекту на випусковій кафедрі дипломної роботи.	17.11.22 – 19.11.22р.	

7. Дата видачі завдання 26 вересня 2022р.

Керівник дипломної роботи _____ Тетяна ХОЛЯВКІНА
(підпис керівника) (П.І.Б.)

Завдання прийняла до виконання _____ Аліса ГАСАНОВА
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Безвідмовне середовище на базі Amazon Web Services» складається зі вступу, трьох розділів, висновку, списку бібліографічних посилань і викладена на 83 сторінках, містить 51 рисунок. Список бібліографічних посилань складається з 20 найменувань.

Ключові слова: ДОДАТОК, AMAZON WEB SERVICES, LINUX, DOCKER, PYTHON, БЕЗВІДМОВНІСТЬ, МАСШТАБУВАННЯ.

Актуальність роботи: забезпечення якості та надійності роботи додатку з використанням передових технологій.

Мета роботи: формування, аналіз та розробка середовища на базі Amazon Web Services для безвідмовної роботи додатку.

Об'єкт дослідження: безвідмовність роботи Telegram боту на базі сервісів Amazon Web Services.

Предмет дослідження: комплексні можливості хмарної платформи.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

AWS – Amazon Web Services, хмарна платформа, яка надає користувачам передплатні сервіси.

IaaS – «інфраструктура як послуга».

PaaS – «платформа як послуга».

SaaS – «програмне забезпечення як послуга.

DRaaS – «аварійне відновлення як послуга».

BaaS – «резервне копіювання як послуга».

Telegram – це мультиплатформенна служба обміну повідомленнями.

Amazon EC2 – Amazon Elastic Compute Cloud.

Amazon CloudWatch – це сервіс моніторингу хмарних ресурсів AWS та програм, що працюють на AWS.

VPC – Amazon Virtual Private Cloud це сервіс, який дає можливість запускати ресурси AWS у логічно ізольованій віртуальній мережі, що визначається користувачем.

Docker – це технологія контейнерезації.

Amazon S3 – Amazon Simple Storage Service це служба зберігання об’єктів.

ВСТУП

Сучасні технології, такі як аналітика великих даних, штучний інтелект і навіть розміщення веб- та мобільних додатків, потребують великої обчислювальної потужності. Хмарні обчислення та хмарні платформи пропонують підприємствам альтернативу розбудові внутрішньої інфраструктури. Завдяки хмарним обчисленням будь-хто, хто користується Інтернетом, може насолоджуватися масштабованою обчислювальною потужністю за принципом підключи і грай. Оскільки це позбавляє організації від необхідності інвестувати та підтримувати дорогу інфраструктуру, воно стало дуже популярним рішенням. Є багато компаній, які пропонують хмарні платформи для розробки, керування та розгортання програм.

Amazon Web Services (AWS) – це найпоширеніша у світі хмарна платформа з найширшими можливостями, що надає понад 200 повнофункціональних сервісів для центрів обробки даних по всій планеті. Мільйони клієнтів, у тому числі стартапи, які стали лідерами за швидкістю зростання, найбільші корпорації та передові урядові установи, використовують AWS для зниження витрат, підвищення гнучкості та прискореного впровадження інновацій

РОЗДІЛ 1. ХМАРНІ ПЛАТФОРМИ

Хмара дає користувачам доступ до тих самих файлів і програм практично з будь-якого пристрою, оскільки обчислення та зберігання здійснюються на серверах у центрі обробки даних, а не локально на пристрої користувача. Ось чому користувач може увійти до свого облікового запису Instagram на новому телефоні після того, як його старий телефон зламав, і все ще знайти свій старий обліковий запис на місці з усіма своїми фотографіями, відео та історією розмов. Так само працює з хмарними постачальниками електронної пошти, такими як Gmail або Microsoft Office 365, і з постачальниками хмарного сховища, такими як Dropbox або Google Drive.

Для компаній перехід на хмарні обчислення знімає деякі витрати на IT та накладні витрати: наприклад, їм більше не потрібно оновлювати та підтримувати власні сервери, оскільки це зробить постачальник хмари, який вони використовують.

Це особливо впливає на малі підприємства, які, можливо, не мали змоги дозволити собі власну внутрішню інфраструктуру, але можуть аутсорсувати свої потреби в інфраструктурі за доступною ціною через хмару. Хмара також може полегшити діяльність компаній на міжнародному рівні, оскільки співробітники та клієнти можуть отримати доступ до тих самих файлів і програм з будь-якого місця.

Хмарні обчислення можливі завдяки технології, яка називається віртуалізація. Віртуалізація дозволяє створити імітований, лише цифровий «віртуальний» комп’ютер, який веде себе так, ніби це фізичний комп’ютер із власним обладнанням. Технічний термін для такого комп’ютера – віртуальна

машина. При правильному застосуванні віртуальні машини на одній і тій самій хост-машині відокремлені одна від одної, тому вони взагалі не взаємодіють один з одним, а файли та програми з однієї віртуальної машини не видимі для інших віртуальних машин, навіть якщо вони увімкнені. та сама фізична машина [1].

Віртуальні машини також більш ефективно використовують обладнання, на якому вони розміщені. Запускаючи багато віртуальних машин одночасно, один сервер перетворюється на безліч серверів, а центр обробки даних перетворюється на цілу низку центрів обробки даних, здатних обслуговувати багато організацій. Таким чином, хмарні провайдери можуть запропонувати використання своїх серверів набагато більшій кількості клієнтів одночасно, ніж вони могли б зробити інакше, і вони можуть зробити це за низькою ціною.

Навіть якщо окремі сервери виходять з ладу, хмарні сервери загалом мають бути завжди онлайн та завжди доступні. Хмарні постачальники зазвичай створюють резервні копії своїх служб на кількох машинах і в кількох регіонах.

Користувачі отримують доступ до хмарних служб або через браузер, або через додаток, підключаючись до хмари через Інтернет, тобто через багато взаємопов’язаних мереж, незалежно від того, який пристрій вони використовують.

Віртуальні машини також більш ефективно використовують обладнання, на якому вони розміщені. Запускаючи багато віртуальних машин одночасно, один сервер перетворюється на безліч серверів, а центр обробки даних перетворюється на цілу низку центрів обробки даних, здатних обслуговувати багато організацій. Таким чином, хмарні провайдери можуть запропонувати використання своїх серверів набагато більшій кількості клієнтів одночасно, ніж вони могли б зробити інакше, і вони можуть зробити це за низькою ціною.

Навіть якщо окремі сервери виходять з ладу, хмарні сервери загалом

мають бути завжди онлайн та завжди доступні. Хмарні постачальники зазвичай створюють резервні копії своїх служб на кількох машинах і в кількох регіонах.

Користувачі отримують доступ до хмарних служб або через браузер, або через додаток, підключаючись до хмари через Інтернет, тобто через багато взаємопов'язаних мереж, незалежно від того, який пристрій вони використовують.

1.1. Ведучі провайдери хмарних платформ

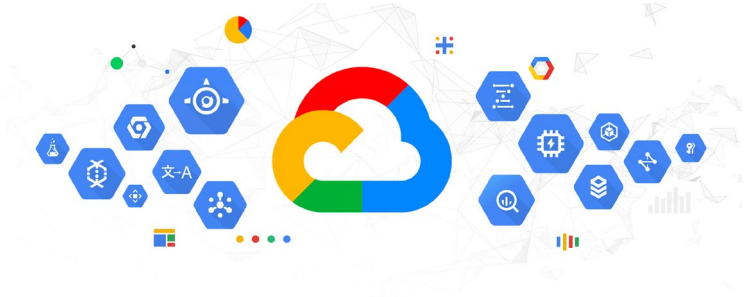


Рис. 1.1. Логотип GCP

Google пропонує свої загальнодоступні хмарні рішення під назвою Google Cloud Platform, або GCP. Він пропонує послуги у всіх основних сферах, включаючи обчислення, мережу, зберігання, машинне навчання (ML) та Інтернет речей (IoT). Він також включає інструменти для управління хмарою, безпеки та розробки. Google Cloud Storage — це високодинамічне рішення для зберігання даних, яке підтримує сховище баз даних як SQL (Cloud SQL), так і NoSQL (Cloud Datastore).

Google Compute Engine (інфраструктура як послуга, або IaaS) надає користувачам екземпляри віртуальних машин для розміщення робочого навантаження. Google App Engine (платформа як послуга, або PaaS) пропонує розробникам програмного забезпечення доступ до хостингу Google на вимогу та комплекту розробки програмного забезпечення (SDK) для розробки додатків, які працюють на платформі додатків. До всіх цих послуг можна

отримати доступ через загальнодоступний Інтернет або через виділені мережі.

Google Cloud Platform або GCP є домом для Kubernetes. Kubernetes був розроблений Google у 2014 році як відкритий код. Google Kubernetes Engine або GKE надає користувачам кероване середовище для розгортання, керування та масштабування контейнерних додатків за допомогою інфраструктури Google.

Рис. 1.2. Логотип AWS

Amazon Web Services (AWS) є дочірньою компанією Amazon (провідної компанії в сфері електронної комерції). Під загальним терміном AWS Amazon надає платформи хмарних обчислень на вимогу, такі як зберігання, аналіз даних тощо. Маючи колосальну частку ринку в 35%, Amazon надає свої послуги приватним особам, компаніям та урядам. Веб-сервіси Amazon дозволяють своїм абонентам насолоджуватися повноцінним віртуальним кластером комп'ютерів у будь-який час відповідно до їхніх вимог. Вся послуга доступна через Інтернет.

Серед усіх постачальників хмарних послуг Amazon вважається найпотужнішим і гнучким рішенням. Віртуальна хмарна платформа AWS має більшість атрибутів реального комп'ютера, включаючи апаратне забезпечення (CPU(s) & GPU(s) для обробки, жорсткий диск/SSD для зберігання та локальна/RAM для пам'яті); операційна система на вибір і попередньо завантажені програми, як-от веб-сервери, бази даних, CRM тощо. Фактично, тепер ви можете отримати різні сертифікати AWS, які пропонує Amazon, щоб

показати свої досягнення та продовжувати навчання, доки ви не освоїте цю хмарну службу.

AWS була першим лідером у сфері публічних хмарних обчислень і стала головним гравцем у сфері штучного інтелекту, баз даних, машинного навчання та безсерверного розгортання.

AWS була першою, хто запропонував інфраструктуру хмарних обчислень як послугу в 2008 році і ніколи не озирався назад. Він запускає нові послуги шаленими темпами та створює власний стек обчислень, який має на меті бути більш ефективним і передавати ці заощадження разом. Цей план навряд чи зміниться, оскільки Адам Селіпскі повернеться, щоб стати генеральним директором AWS, а Енді Джассі перейде в Amazon замість Джеффа Безосар [2].

AWS вийшла далеко за межі хмарних обчислень і сховищ. Якщо процесори на основі Arm стануть нормою для дата-центрів, індустрія може подякувати гравітаційному притягуванню AWS, яка випустила процесор Graviton другого покоління та екземпляри на його основі. У разі успіху гравітон і шар абстракції Nitro можуть стати відмінністю для AWS у хмарних війнах.

Рис. 1.3. Логотип Microsoft Azure

Microsoft Azure, часто згадується просто як Azure — хмарна платформа та інфраструктура корпорації Microsoft, призначена для розробників

застосунків хмарних обчислень і покликана спростити процес створення онлайнових додатків.

Корпорація Microsoft представила платформу Windows Azure 27 жовтня 2008 року.

Windows Azure дозволяє створювати додатки як за допомогою Microsoft .NET Framework і Visual Studio, так і за допомогою інших інструментів. Операційна система працює на серверах Microsoft, доступ до неї можна отримати за протоколами HTTP, Representational State Transfer (REST), WS-* і Atom Publishing Protocol (AtomPub).

Платформа Azure Services Platform включає п'ять основних компонентів. Це сама операційна система Windows Azure, що керує дисковим простором, додатками і мережами, і Microsoft SQL Services для роботи з базами даних. Також в платформу входять Microsoft .NET Services, Live Services, і бізнес-компонент, що включає Microsoft SharePoint Services і Microsoft Dynamics CRM Services.

Платформа виготовлена з групи із трьох технологій, що забезпечують спеціалізований набір можливостей для розробників. Більше того, платформу Windows Azure можна використовувати в додатках, що працюють локально на комп'ютерах користувачів і додатків, які працюють в хмарі.

Платформа Windows Azure складається з таких компонентів:

- Windows Azure — надає середовище виконання для додатків, заснованих на операційних системах, та на Windows Server, а також місця для зберігання даних. Система працює на віртуальних машинах за допомогою, аналогічної технології Hyper-V.
- Обчислення — відповідає за обчислення розміщення додатків.
- Зберігання — відповідає за зберігання даних в хмарі.
- SQL Azure — надає можливість використовувати реляційну базу даних для запуску в хмарі.
- Платформа Windows Azure AppFabric — компонент, який забезпечує додаткову функціональність у вигляді послуг.

1.2. Типи (моделі) хмарних платформ

Виділяють наступні 3 основних типа та 2 додаткові:

1) IaaS – «інфраструктура як послуга». IaaS-провайдери надають замовнику обчислювальну інфраструктуру (сервери, сховища даних, операційні системи та мережеві ресурси) для розгортання та запуску власних програмних рішень. Варіант підійде компаніям, потреба яких у ресурсах не однакова в різні моменти часу - бувають сплески потреб, але вони поступово спадають (або організація швидко зростає, і виникає проблема постійного масштабування інфраструктури). Також IaaS буде оптимальним рішенням, коли компанія недостатньо коштів на створення власної інфраструктури.

2) PaaS – «платформа як послуга». Провайдер хмарних сервісів надає замовнику готове програмне середовище та інструменти для його налаштування. Елементами PaaS є апаратне забезпечення, операційна система, СУБД, проміжне ПЗ, інструменти тестування та розробки. Таку платформу клієнт може налаштувати під свої потреби, зробивши з неї майданчик для тестування або, наприклад, систему для автоматизації системи управління. Такий вид сервісу користується особливою популярністю у розробників програмного забезпечення.

3) SaaS – «програмне забезпечення як послуга». Розробник програмної платформи надає віддалений доступ до неї клієнту. Наприклад, саме за моделлю SaaS корпорація Microsoft забезпечує клієнтам користування MS Office Suite (Office Web Apps) поряд із SharePoint Server, Exchange Server та іншими сервісами та додатками. Також за моделлю SaaS надаються поштовий сервіс Gmail та хмарна версія 1С.

4) DRaaS – «аварійне відновлення як послуга». Варіант для забезпечення катастрофостійких рішень за допомогою хмар провайдера. На майданчик постачальника хмарних рішень реплікуються дані із основного майданчика клієнта. При виході з ладу сервісів клієнта вони протягом декількох хвилин перезапускаються, але вже в хмарі. Такі рішення особливо

цікаві компаніям із великою кількістю бізнес-критичних додатків.

5) BaaS – «резервне копіювання як послуга». Як і слідує з розшифровки абревіатури, йдеться про резервне копіювання даних клієнта в хмару провайдера. Постачальник надає не тільки місце для зберігання інформації, але й інструменти для швидкого та надійного копіювання.
Та декілька моделів розгортання систем хмарних обчислень, на прикладі AWS [3].

1) **Хмара**

Розгортання та робота всіх елементів додатків на основі систем хмарних обчислень повністю здійснюється у хмарі. Програму можна одразу створити у хмарі або перенести з існуючої інфраструктури і надалі користуватися всіма перевагами систем хмарних обчислень. Хмарні програми можна створити на інфраструктурі низького рівня або скористатися високорівневими послугами, що дозволяють не турбуватися про управління, побудову архітектури та масштабування базової інфраструктури.

2) **Гібридна**

Розгортання гібридної архітектури дозволяє пов'язати інфраструктуру та застосування хмарних ресурсів з існуючими ресурсами, що не належать хмарі. Найпоширеніший спосіб розгортання гібридної архітектури – комбінація хмари та існуючої локальної інфраструктури з метою розширення та нарощування інфраструктури організації у хмару шляхом підключення хмарних ресурсів до внутрішньої системи.

3) **Локальна**

Розгортання ресурсів у локальній інфраструктурі з використанням засобів віртуалізації та управління ресурсами також називають приватною хмарою. При локальному розгортанні не можна скористатися цілою низкою переваг систем хмарних обчислень, але іноді до нього вдаються для забезпечення виділених ресурсів. У більшості випадків ця модель розгортання є нічим іншим, як використанням існуючої IT-інфраструктури із застосуванням функцій управління додатками та технологій віртуалізації для більш

продуктивної роботи ресурсів [4].

1.3 Amazon Web Services

Релевантість того чи іншого хмарного провайдера залежить від спектру послуг і звичайно якості цих послуг. Сьогодні на ринку лідером у світі є AWS.

Так як надає понад 200 повнофункціональних сервісів для центрів обробки даних по всій планеті.

Найпопулярніші:

1. **Elastic Compute Cloud (EC2).** Сервіс надає передплатникам віртуальні серверні платформи, системи зберігання даних та балансувальник навантаження. Користувачі можуть вибирати як заздалегідь налаштований сервер з попередньо встановленою операційною системою, так і зібрати його самостійно. Сервіс також дає можливість створювати образи або використовувати власну ОС. Для забезпечення безпеки передплатники можуть розмежовувати доступ до серверів EC2 за IP-адресами.

Вартість послуги оплачується за кожну годину, а деякі опції – на щомісячній основі. Якщо клієнт планує використовувати EC2, рекомендується скористатися функцією резервації. Передплатник оплачує відразу 3 місяці роботи, і підсумкова вартість стає нижчою у півтора рази.

2. **Simple Storage (S3)** надає передплатникам розподілений дисковий простір із максимальним об'ємом 5 Тб. S3 зберігає файли в так званих бакетах, що розташовані на різних майданчиках Amazon.

Сервіс веде постійний лог всіх операцій на сховищі та зберігає інформацію у спеціальному бакеті. Це дозволяє клієнтам відразу отримувати докладну інформацію про останні операції. Відмінна риса сервісу - заміна протоколу http на BitTorrent для завантаження файлів.

Передплатники можуть самостійно вибирати безпекові політики, які застосовуються до бакетів. Це дозволяє розмежувати права доступу у

користувачів, зробити хмару повністю приватною тощо. Оплата послуги стягується щомісяця. Вартість залежить від обсягу дискового простору, кількості запитів та вихідного трафіку.

3. **Relational Database Service (RDS)** надає користувачам віртуальну базу даних, розташовану на виділеному сервері. При цьому платформа налаштована та оптимізована під обрану БД. Мінімальний розмір дискового простору під базу даних - 5 Тб.

Доступ налаштовується залежно від побажань користувачів та політик безпеки. Наприклад, замовники можуть дозволити підключення тільки з певних IP-адрес (підмереж) або для тих груп безпеки, які вказані в сервісі EC2.

Передплатникам доступні функції миттєвих знімків (SnapShot), реплікація (асинхронна та синхронна), резервне збереження даних в автоматичному режимі та метрокластер.

Оплата здійснюється аналогічно до сервісу EC2 — щогодини. Якщо оплачувати три і більше місяців заздалегідь, то підсумкова вартість знижується в півтора рази. Додаткові функції оплачуються окремо.

4. **Route 53 (DNS)** дає користувачам можливість піднімати DNS-сервер у хмарі Amazon. Послуга легко інтегрується з іншими AWS.

Оплата стягується за кількість запитів до сервера, але є безкоштовний ліміт.

Amazon Web Services надає користувачам більшість хмарних рішень за доступною ціною. Надійність і стійкість до відмов гарантується на рівні «шості дев'яток» (99,9999%). Клієнт може гнучко налаштувати потрібний список доступних сервісів, вибравши лише необхідні. Кожна послуга має власний персональний ліміт (Free Tier), що суттєво знижує вартість сервісу, що надається [5].

Вигляд головної консолі управління усіма існуючими сервісами в AWS.

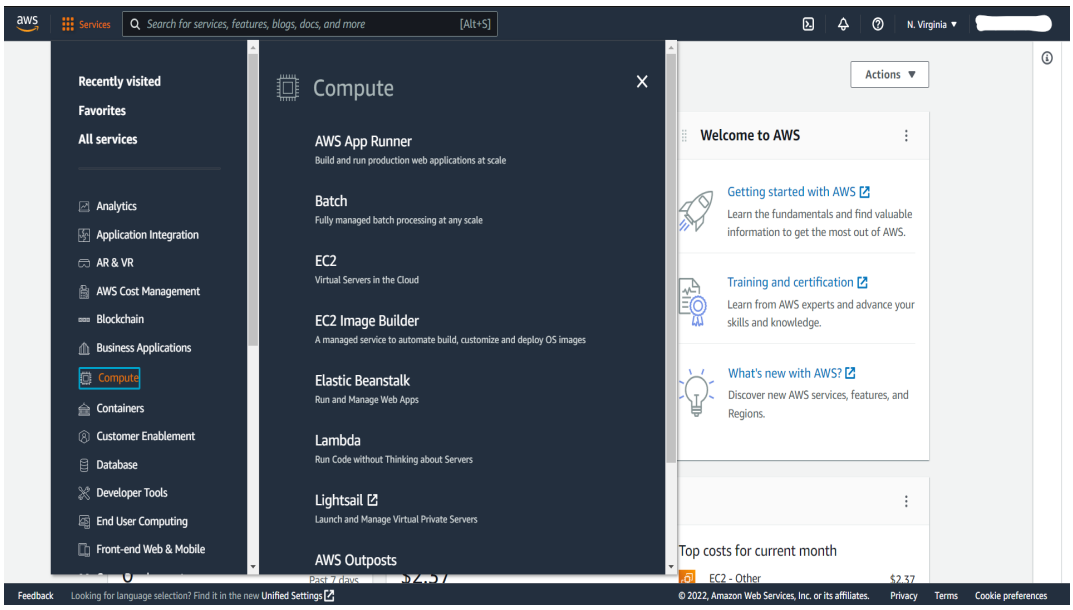


Рис. 1.4. Вигляд AWS консолі

1.4. Переваги та недоліки AWS

Той факт, що AWS має більш ніж 10-річний досвід роботи у хмарних обчисленнях, гарантує безліч переваг платформи. Недарма багато відомих організацій довіряють цьому сервісу свої дані, проекти, аналізи, звіти і навіть всю свою інфраструктуру.

Основні переваги AWS:

- Найширша мережа в дата-центрах по всьому світу: вона пропонує високу надійність та безпеку у своїх послугах та у мережній інфраструктурі, яку він пропонує своїм клієнтам. Його великий досвід також дозволив розширити спектр послуг, які він пропонує порівняно з іншими аналогічними платформами.
- Гарна гнучкість для проектів розробка: незалежно від того, який тип продукту, проекту або рішення вам потрібно інтегрувати у вашу робочу команду або бізнес, ви знайдете в AWS всі сервіси, необхідні для цього. Інструменти легко адаптуються та легко налаштовуються відповідно до потреб кожного середовища.
- Бюджетний та можливість економії: позбавляючись необхідності

вкладати кошти в обслуговування та оновлення обладнання та програмного забезпечення, ви можете зосередитися на основних цілях свого бізнесу або організації. Крім того, AWS стягує плату тільки за використання сервісів і може масштабуватися, наприклад, ціни та функції, які пропонують Amazon EC2.

Основні недоліки AWS:

- Це дуже складно для недосвідчених користувачів: Якщо ви хочете виконувати прості завдання чи самостійно створювати програмне забезпечення і у вас немає досвіду, AWS – не найкращий варіант для вас. Інструменти трохи складні для користувачів-початківців, у цих випадках краще вирішити створити обліковий запис в Google Cloud, його інструменти простіше у використанні.
- Він не адаптований до конкретного середовища: будучи такою великою та різноманітною платформою, дуже легко загубитися у кількості пропонованих послуг та залишити осторонь роботу, проект чи потреби компанії. Якщо ви шукаєте платформу з інструментами для регульованих середовищ AWS не найкращий варіант.
- Складання бюджету та планування витрат утруднені: у деяких випадках плата за користування може бути перевагою, в інших – ні. Коли планування витрат є суворим, це може бути дуже складно визначити загальну суму комісійних за придбані послуги, особливо коли робочий процес доволі напружений.

1.5. Світове використання Amazon Web Services

Amazon Web Services (AWS), запущена в 2006 році, пропонує послуги хмарних обчислень державним організаціям, підприємствам, навчальним закладам і окремим особам за моделлю оплати за використання. Відповідно до звіту CSA (Cloud Security Alliance), AWS займає 41,5% ринку хмарних обчислень — це більше, ніж усі його конкуренти разом узяті — Microsoft Azure (29,4%), Google Cloud (3,0%) і IBM (2,6%). Одним із маловідомих фактів

про AWS є те, що в 2013 році Amazon Web Services уклала угоду з Центральним розвідувальним управлінням (ЦРУ) на суму 600 мільйонів доларів для підтримки своїх потреб у хмарних обчисленнях [6].

Перший учасник ринку хмарної інфраструктури, AWS повідомила про прибуток у 3 мільярди доларів у 2013 році. Дохід компанії підскочив до 35 мільярдів доларів до 2019 року завдяки всесвітньому впровадженню хмарних сервісів, що робить її основним джерелом доходу для материнської компанії. організація Amazon.

Amazon Web Services має понад 1 мільйон активних користувачів. За даними різних консалтингових фірм, клієнти корпоративного масштабу складають близько 10 відсотків користувачів AWS, а решта — це малий і середній бізнес.

AWS пропонує 175 повнофункціональних сервісів через динамічну екосистему, що дозволяє користувачам розгортати робочі навантаження додатків із затримкою в мілісекунди лише за один клік.

Більшість компаній зі списку Fortune 500 і понад 90 відсотків організацій зі списку Fortune 100 використовують APN (AWS Partner Network) для розробки послуг і рішень для клієнтів. Деякі з найбільших світових брендів, як-от Facebook, Netflix, Adobe і BBC, покладаються на нього для підтримки своїх найкращих проектів.

AWS Marketplace пропонує цифровий каталог із понад 7000 продуктами даних і програмним забезпеченням. Marketplace дозволяє постачальникам даних і продавцям керувати відповідними сторонніми продуктами даних і програмним забезпеченням шляхом включення цифрового каталогу у веб-ресурси.

Згідно з новим звітом TSO Logic, ціни на Amazon Web Services продовжують падати. Звіт показує, що AWS знижував ціни шістдесят сім разів з моменту свого запуску в 2006 році.

Багатоканальна та багатоплатформна медіа-група Asianet News Media & Entertainment Pvt., Ltd. змогла скоротити свої операційні витрати на 50

відсотків за допомогою субсекундної моделі вимірювання AWS Lambda.

Amazon Web Services пропонує широкий вибір із п’ятнадцяти спеціально створених баз даних. Оптимізовані служби баз даних Amazon забезпечують чудову продуктивність, доступність і масштабованість, забезпечуючи відмінну підтримку вимогливих робочих навантажень.

Тепер Amazon Web Services охоплює сімдесят сім зон доступності (AZ) у двадцяти чотирьох географічних регіонах по всьому світу. Компанія планує запустити ще вісімнадцять AЗ і ще шість регіонів в Індії, Швейцарії, Іспанії, Японії, Індонезії та Австралії [6].

Інфраструктура Amazon Web Services обслуговує тисячі підприємств у 245 країнах і територіях. Компанія підтримує 5 локальних зон, 12 зон довжин хвиль, 97 місць прямого з’єднання, 12 регіональних кеш-пам’ятей, понад 210 локацій і вдвічі більше регіонів, ніж її найближчий конкурент.

Щоб зрозуміти справжній масштаб AWS, потрібно поглянути на цифри.

AWS має понад 1 мільйон активних користувачів у 190 країнах.

AWS має в 5 разів більше розгорнутої хмарної інфраструктури, ніж їхні наступні 14 конкурентів разом узяті.

Щодня AWS додає стільки інфраструктури, скільки було загалом 7 років тому

Amazon S3 розроблено для забезпечення довговічності 99,999999999% і масштабування понад трильйони об’єктів по всьому світу.

На партнера AWS, Netflix, припадає до однієї третини інтернет-трафіку під час пікового використання.

ВИСНОВОК ДО РОЗДІЛУ 1

Хмарні сервіси — це платформи та програми, які "живуть" і працюють на серверах спеціальних компаній — хмарних провайдерів. Ці програми не потрібно встановлювати на комп’ютер, а доступ до них можна отримати з будь-якої точки світу. Dropbox, Google Drive, iCloud — всі ми давно користуємось цими та схожими продуктами, навіть не усвідомлюючи, що вони розміщені у хмарах.

Світовий ринок хмарних сервісів концентрується навколо трьох IT-гігантів: Google, Amazon та Microsoft, які займають 70% ринку сервісів IaaS. Послуги Amazon та Microsoft найбільше використовують компанії США та Європи. В Китаї ринок практично повністю монополізував місцевий провайдер Alibaba Cloud.

Завдяки хмарним обчисленням дані організації можна аналізувати для пошуку шаблонів і відомостей, робити прогнози, покращувати їх і приймати інші бізнес-рішення. Хмарні служби можуть надати вашій організації більш високу обчислювальну потужність і просунуті засоби для отримання величезної кількості даних, а також можливість швидкого масштабування середовища в міру збільшення їх обсягу.

РОЗДІЛ 2. ТЕХНОЛОГІЇ СТВОРЕННЯ ІНФРАСТРУКТУРИ

2.1. Telegram

Telegram — це інструмент обміну миттєвими повідомленнями, який дозволяє надсилати та отримувати повідомлення від ваших контактів, навіть не повідомляючи свій номер телефону.

Це робиться за допомогою протоколу зв'язку, відомого як MTProto, який дає вам можливість відкривати різні сеанси на кількох пристроях без одночасного підключення.

Telegram — це мультиплатформенна служба обміну повідомленнями. Він вперше вийшов на iOS та Android наприкінці 2013 року, і зараз його відвідують приблизно 550 мільйонів користувачів щомісяця. База користувачів Telegram, як правило, збільшується щоразу, коли скандал із конфіденційністю вражає одного з його більших конкурентів.

Що робить Telegram унікальним, так це його фокус на конфіденційності, шифруванні та API з відкритим кодом. Є незліченна кількість неофіційних клієнтів, які підходять разом з офіційними програмами та веб-інтерфейсом Telegram. Це також дозволяє кільком пристроям використовувати один і той самий обліковий запис (підтверджено за допомогою SMS) і кілька облікових записів на одному пристрої. Основні функції Telegram такі ж, як і в більшості інших програм обміну повідомленнями: можливість надсилати повідомлення іншим користувачам Telegram, створювати групові розмови, телефонувати контактам, здійснювати відеодзвінки та надсилати файли та наклейки. Однак є кілька особливостей, які відрізняють його роботу від інших програм для чату [7].

Перш за все, основною функцією Telegram є конфіденційність, і для забезпечення цього він використовує наскрізне шифрування. Це те, що заважає тим, хто не підтримує двосторонню розмову — будь то компанія, уряд, хакери чи хтось інший.

Однак Telegram використовує це шифрування лише під час дзвінків і у своїй функції «секретних чатів», а не в звичайних чатах. Це лише зашифровані клієнт-сервер.

Причиною цього є розширене використання хмари Telegram. По суті, він зберігає всі повідомлення та фотографії на безпечному сервері. Це означає, що ви можете отримати доступ до них із будь-якого підключеного пристрою, що робить Telegram набагато зручнішим для багатьох платформ.

Ще одна функція безпеки, яка додає зручності використання, — це імена користувачів. Замість того, щоб повідомляти людям свій номер телефону, можете просто дати їм своє ім’я користувача. Це дає кращий контроль над тим, яка інформація є там, і як люди можуть зв’язатися з вами в майбутньому.

Telegram використовують для багатьох речей, деякі з них:

1. Надсилання повідомлень.

Звичайно, будучи платформою обміну миттєвими повідомленнями, вона дозволяє надсилати та отримувати повідомлення вашим контактам і від них. Не потрібно ділитися своїм номером телефону, фактично, можете приховати його, щоб захистити свою конфіденційність.

Telegram дає можливість створити власне ім’я користувача, щоб вас можна було знайти через його пошукову систему. Потрібно лише знати імена користувачів.

Можливо надсилати відео, аудіо, текстові повідомлення, здійснювати необмежену кількість відеодзвінків з багатьма учасниками. Все це на мобільному телефоні або комп'ютері.

2. Зберігання файлів.

Без сумніву, це одна з найважливіших переваг Telegram. Платформа має власну хмару, тому всі повідомлення та файли, які ви маєте в чатах, зберігатимуться, доки їх не видалите. Обсяг, який хочете, його місткість нескінченна, і зможете зберігати відео, аудіо та будь-який аудіовізуальний документ.

Створюйте окремі чати, щоб розділити їх на різні теми. Прикладом можуть бути: сімейні фотографії, робочі речі тощо.

Розмір не може перевищувати 15 Гб, але цього вистачить практично для будь-якого документа.

3. Створення групи.

Створення груп є основною функцією будь-якої програми обміну миттєвими повідомленнями, але супергрупи – це щось зовсім інше.

За допомогою цієї функції можливо створити групу з 20000 членами.

4.Організація ваших чатів у папках.

У Telegram можливо мати мільйони контактів, оскільки після встановлення він синхронізується з даними вашого телефону.

Проблема полягає в тому, що іноді не можливо знайти людину, з якою потрібно поговорити.

Отже, можливе рішення, яке допоможе вам залишатися організованим у хаосі, — це можливість створювати папки для різних тем.

У кожній папці у будуть чати людей на певні теми. Таким чином є можливість організувати платформу та працюватимете ефективніше.

5. Створення каналів.

Канали Telegram — чудовий інструмент, особливо якщо є цифрове підприємство.

Це односторонній чат, у якому лише адміністратор зможе надсилати повідомлення своїм абонентам. Папки можуть бути публічними або приватними, і вони працюють як канал зв’язку зі спільнотою.

Канали є чудовими для вашого бренду, оскільки вони дозволяють вам підтримувати зв’язок із потенційними клієнтами, інформувати їх про останні

новини про ваші продукти чи послуги, а також знати їхню думку про вдосконалення, які необхідно внести.

6. Кілька сеансів.

Telegram має дуже жорсткий рівень безпеки, настільки, що є можливість надіслати запит на двоетапну перевірку, щоб переконатися, що входить у свій обліковий запис один і той самий користувач.

Це також дозволяє входити з інших пристроїв, на яких ви відкрили обліковий запис.

7. Безпека.

Telegram настільки відомий своїм наскрізним шифруванням, що багато фахівців рекомендують його.

2.2. Telegram бот

Боти Telegram — це невеликі програми, які можна вбудовувати в чати Telegram або публічні канали та виконувати певну функцію. Вони можуть пропонувати налаштовані клавіатури, створювати котячі меми на вимогу або навіть приймати платежі та діяти як цифровий магазин.

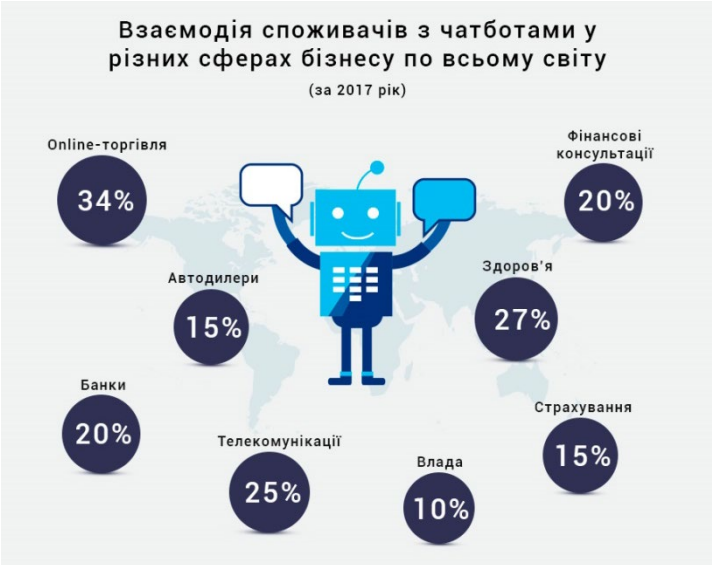


Рис. 2.1. Статистичні дані застосування чат-ботів

Боти популярні в Telegram, тому що вони зручні, і Telegram підтримує їх з 2015 року. По суті, це автоматизовані облікові записи Telegram. Зокрема, боти Telegram не використовують MTProto, протокол шифрування Telegram, який створює структуру, у якій повідомлення користувачів один одному зашифровані та нерозбірливі під час передачі між пристроями відправника та одержувача. Але бот-платформа Telegram покладається на протокол безпеки транспортного рівня, який використовується в веб-шифруванні HTTPS.

Telegram-бот — це особливий тип користувача, який не є людиною, а комп'ютерною програмою, яка може обслуговувати компанії чи бренди за допомогою багатьох функцій, таких як надсилання інформації, нагадувань, відтворення мелодій, замовлення тощо. Бот може опублікувати повідомленн в групі або каналі.

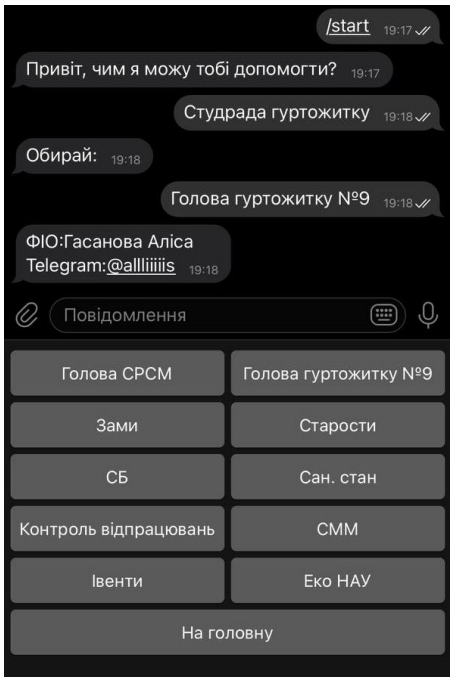


Рис. 2.2. Приклад телеграм бота

Звичайні користувачі можуть стежити за будь-яким телеграм-ботом, якого хочуть отримувати в курсі. З усіх чотирьох типів бот Telegram має багато

функцій, які дійсно корисні для бізнесу. Telegram надає API для створення ботів для соціальних взаємодій, продуктивності, ігор і послуг електронної комерції на платформі.

Окрім цього, боти Telegram також можуть надавати підтримку клієнтів або збирати потенційних клієнтів, підключаючи їх до CRM, системи продажу квитків або платформи обміну повідомленнями.

Щодня з'являються нові бізнес-випадки використання ботів, боти стають все більш популярними для роботи в середовищі Telegram. За допомогою API телеграм легко створюєте телеграм-бота, просуваєте його, щоб отримати більше користувачів для вашої бізнес-взаємодії.

Telegram безкоштовний незалежно від кількості повідомлень і мети використання (професійне чи особисте використання). Оскільки Telegram є безкоштовним як для користувачів, так і для бізнесу, це дозволяє компаніям отримувати потенційних клієнтів, ефективно використовуючи його. Настільки, що створення телеграм-бота також безкоштовне. Підприємства можуть використовувати цю безкоштовну платформу для створення своєї клієнтської бази, надсилаючи інформаційні бюлетені, надаючи підтримку клієнтам або спілкуючись з ними.

Якщо клієнтська база — це активне використання Telegram, це чудова можливість зацікавити ваших клієнтів, оскільки Telegram дозволяє користувачам надсилати компаніям аудіо, відео та зображення. Підприємства можуть створювати свої маркетингові кампанії, щоб залучити своїх клієнтів різними способами. Крім маркетингу, деяким компаніям потрібно цілодобово підтримувати клієнтів, тож компанії можуть створити чат-бота для Telegram або створити команду підтримки клієнтів, яка майже миттєво відповідає на запити клієнтів. Це також дозволяє нам покращити взаємодію з клієнтами.

Враховуючи глобальні тенденції щодо негативних випадків на кожній платформі соціальних мереж, Telegram вважається досить безпечним головним чином тому, що повідомлення надсилаються через платформу в зашифрованому вигляді. Це вагомий привід використовувати його в бізнес-

цілях, оскільки він захистить усі ваші дані та дані вашого клієнта. Це забезпечить безпечне та краще місце для взаємодії щодо продуктів і послуг.

Месенджер Telegram охоплює всі основні платформи, такі як телефони Android, iOS, Windows, із настільними програмами для Mac, Linux і Windows. Крім того, він також має веб-версію, яка дозволить орієнтуватися на потенційних клієнтів у дуже широкому масштабі. По-друге, не потрібно хвилюватися, що клієнти не зможуть отримати доступ до цього месенджера. За допомогою телеграм-бота є можливість надати своїм клієнтам швидкість отримання інформації, і вони зможуть використовувати її будь-де та будь-коли без проблем. Легкість доступності збільшує можливість пошуку потенційних покупців і підвищує коефіцієнт конверсії [8].

Telegram є одним із перших, хто прив’язав банківський рахунок до чат-месенджерів. Це дозволить користувачам здійснювати фінансові операції з іншими користувачами або компаніями на платформі. Така можливість на цій платформі відкриває більше варіантів використання для бізнесу. Тож телеграм-бот може дозволити клієнту переказувати гроші чи оплачувати послуги, не виходячи з платформи.

2.3. Мова програмування Python

За останні роки Python став однією з найпопулярніших мов програмування у світі. Його використовують у всьому, від машинного навчання до створення веб-сайтів і тестування програмного забезпечення. Його можуть використовувати як розробники, так і не розробники. Python — це мова комп’ютерного програмування, яка часто використовується для створення веб-сайтів і програмного забезпечення, автоматизації завдань і аналізу даних. Python є мовою загального призначення, тобто її можна використовувати для створення різноманітних програм і не спеціалізується на конкретних проблемах. Ця універсальність, а також зручність для початківців зробили її однією з найбільш використовуваних мов програмування сьогодні.

Опитування, проведене галузевою аналітичною компанією RedMonk, показало, що це була друга за популярністю мова програмування серед розробників у 2021 році.

Python зазвичай використовується для розробки веб-сайтів і програмного забезпечення, автоматизації завдань, аналізу та візуалізації даних. Оскільки його відносно легко вивчити, Python був прийнятий багатьма непрограмістами, такими як бухгалтери та вчені, для різноманітних повсякденних завдань, як-от організація фінансів [9].

Python популярний з кількох причин. Ось глибший погляд на те, що робить його таким універсальним і простим у використанні для програмістів.

Він має простий синтаксис, який імітує природну мову, тому його легше читати та розуміти. Це дозволяє швидше створювати проекти та швидше їх покращувати.

Він універсальний. Python можна використовувати для багатьох різних завдань, від веб-розробки до машинного навчання.

Це зручно для початківців, що робить його популярним для програмістів початкового рівня.

Це відкритий код, що означає, що його можна безкоштовно використовувати та поширювати навіть у комерційних цілях.

Архів модулів і бібліотек Python — наборів коду, створених сторонніми користувачами для розширення можливостей Python — величезний і постійно зростає.

Сумісний із різними платформами, включаючи Windows, Mac, Linux, Raspberry Pi та інші

Використовує простий синтаксис, який можна порівняти з англійською мовою, що дозволяє розробникам використовувати менше рядків, ніж інші мови програмування.

Працює на системі інтерпретатора, яка дозволяє негайно виконувати код, швидко відстежуючи прототипування.

Можна обробляти процедурним, об’єктно-орієнтованим або функціональним способом.

Python, мова з динамічною типізацією, є особливо гнучкою, усуває жорсткі правила створення функцій і пропонує більшу гнучкість у вирішенні проблем за допомогою різноманітних методів. Він також дозволяє компілювати та запускати програми аж до проблемної області, оскільки використовує перевірку типу під час виконання, а не під час компіляції.

З іншого боку, Python нелегко підтримувати. Одна команда може мати кілька значень залежно від контексту, оскільки Python є мовою з динамічним типом. І підтримувати програму Python, коли вона зростає в розмірі та складності, може бути дедалі складніше, особливо знаходити та виправляти помилки. Користувачам знадобиться досвід для розробки коду або написання модульних тестів, які спрощують обслуговування.

Швидкість — ще одна слабка сторона Python. Його гнучкість, оскільки він динамічно типізований, вимагає значної кількості посилань, щоб отримати правильне визначення, що сповільнює продуктивність. Це можна пом’якшити, використовуючи альтернативну реалізацію Python (наприклад, PyPy).

2.4. Операційна система Linux

Як і Windows, iOS і Mac OS, Linux є операційною системою. Насправді, одна з найпопулярніших платформ на планеті, Android, працює на базі операційної системи Linux. Операційна система — це програмне забезпечення, яке керує всіма апаратними ресурсами, пов’язаними з вашим настільним комп’ютером або ноутбуком. Простіше кажучи, операційна система керує зв’язком між вашим програмним забезпеченням і апаратним забезпеченням. Без операційної системи (ОС) програмне забезпечення не функціонувало б.

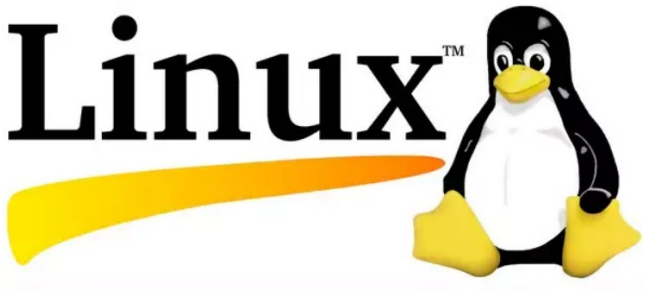


Рис. 2.3. Логотип Linux

Операційна система Linux складається з кількох різних частин:

Завантажувач – програмне забезпечення, яке керує процесом завантаження комп’ютера. Для більшості користувачів це буде просто екран-заставка, який з’являється та згодом зникає для завантаження операційної системи.

Ядро – це єдина частина цілого, яка насправді називається «Linux». Ядро є ядром системи та керує ЦП, пам'яттю та периферійними пристроями. Ядро — найнижчий рівень ОС.

Система ініціалізації – це підсистема, яка завантажує простір користувача та відповідає за керування демонами. Однією з найпоширеніших систем ініціалізації є systemd, яка також є однією з найбільш суперечливих. Це система ініціалізації, яка керує процесом завантаження після того, як початкове завантаження передається із завантажувача (тобто GRUB або GRand Unified Bootloader).

Демони – це фонові служби (друк, звук, планування тощо), які запускаються під час завантаження або після входу на робочий стіл.

Графічний сервер – це підсистема, яка відображає графіку на моніторі. Його зазвичай називають сервером X або просто X.

Середовище робочого столу – це частина, з якою користувачі фактично взаємодіють. Існує багато робочих середовищ на вибір (GNOME, Cinnamon,

Mate, Pantheon, Enlightenment, KDE, Xfce тощо). Кожне середовище робочого столу містить вбудовані програми (такі як файлові менеджери, інструменти налаштування, веб-браузери та ігри).

Програми. Настільні середовища не пропонують повний набір програм. Так само, як Windows і macOS, Linux пропонує тисячі і тисячі високоякісних програм, які можна легко знайти та встановити. Більшість сучасних дистрибутивів Linux (докладніше про це нижче) включають інструменти, подібні до App Store, які централізують і спрощують установку програм. Наприклад, Ubuntu Linux має Центр програмного забезпечення Ubuntu (ребрендинг програмного забезпечення GNOME), який дозволяє швидко шукати серед тисяч програм і встановлювати їх з одного централізованого місця [10].

Linux перетворився на одну з найнадійніших комп'ютерних екосистем на планеті. Поєднайте цю надійність із нульовою вхідною вартістю, і ви отримаєте ідеальне рішення для настільної платформи. Компанії та приватні особи обирають Linux для своїх серверів, оскільки він безпечний, гнучкий, і ви можете отримати чудову підтримку від великої спільноти користувачів, на додаток до таких компаній, як Canonical, SUSE та Red Hat, кожна з яких пропонує комерційну підтримку.

Багато пристроїв, якими ви, ймовірно, володієте, як-от телефони та планшети Android і комп'ютери Chromebook, цифрові пристрої зберігання даних, персональні відеомагнітофони, камери, переносні пристрої тощо, також працюють під управлінням Linux. Під капотом вашого автомобіля працює Linux. Навіть Microsoft Windows містить компоненти Linux як частину підсистеми Windows для Linux (WSL).

Linux можливо встановити на скільки завгодно комп'ютерів, не сплачуючи жодного цента за програмне забезпечення чи ліцензування сервера.

Вартість сервера Linux у порівнянні з Windows Server 2016. Ціна версії Windows Server 2016 Standard становить 882,00 доларів США (придбано безпосередньо в Microsoft). Це не включає ліцензію клієнтського доступу

(CAL) і ліцензії на інше програмне забезпечення, яке вам може знадобитися (наприклад, базу даних, веб-сервер, поштовий сервер тощо).

Linux також поширюється за ліцензією з відкритим кодом. Відкритий код дотримується таких основних принципів:

1.Свобода запускати програму з будь-якою метою.

2.Свобода вивчати, як працює програма, і змінювати її.

3.Свобода розповсюджувати копії.

4.Свобода розповсюдження копій ваших змінених версій іншим.

Ці моменти мають вирішальне значення для розуміння спільноти, яка працює разом над створенням платформи Linux. Без сумніву, Linux — це операційна система, створена «народом, для людей». Ці принципи також є основним фактором, чому багато людей обирають Linux. Йдеться про свободу, свободу використання та свободу вибору.

2.5. Сервіси AWS

Amazon Web Services (AWS) – це найпоширеніша у світі хмарна платформа з найширшими можливостями, що надає понад 200 повнофункціональних сервісів для центрів обробки даних по всій планеті. Тут можна знайти готові та гнучкі рішення (сервіси) для будь-яких потреб користувача, бізнесу чи компанії.

2.5.1. Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud (Amazon EC2) - сервіс Amazon Web Services, що дозволяє користувачеві орендувати віртуальну машину, яка називається інстансом (англ. instance). Для запуску інстансів використовуються попередньо налаштовані образи (англ. Amazon Machine Image - AMI), що скорочує час створення та завантаження нового сервера, основуючись на тому, яка операційна система потрібна користувачу: Amazon Linux, Red Hat EL, Suse ES, Windows 2008, Oracle EL, Windows Server та інші.

Є вже сконфігуровані амазоном образи і також можна самому створити свій індивідуальний чи використати конфігурацію спільноти користувачів AWS [11].

Взаємодіяти з сервісом можна за допомогою веб-інтерфейсу, інтерфейсу командного рядка (CLI), а також програмно за допомогою API.

Доступно створення, запуск, зупинка та видалення інстансів за запитом користувача в міру необхідності, з посекундною оплатою - звідси і термін "Elastic" - еластичний. EC2 також надає і контроль над географічним розташуванням інстансів, що дозволяє мінімізувати затримки та забезпечувати високий рівень доступності.

Є різні типи EC2 інстансів:

- on-demand інстанси - звичайні інстанси з щосекундною оплатою, які надаються користувачеві на вимогу, їх перевага в тому, що їх можна використовувати для короткочасних завдань (тестів та ін.).
- спотові інстанси - хмарні провайдери зазвичай мають запаси обчислювальної потужності, які вони можуть продати іншим користувачам, але без гарантій того, що надання послуги не буде зупинено в будь-який момент, при цьому ціна на послугу плаваюча, і дозволяє заощадити користувачеві до 90% від вартості on-demand інстансів, це дуже вигідно для тих користувачів яким не критично втратити свої данні чи зроблену роботу і при цьому це найдешевший варінт.
- зарезервовані інстанси - використовуються користувачами для резервування обчислювальної потужності від одного до трьох років і дозволяють заощадити, за твердженням Amazon'a, до 72%. Це як довгострокова оренда комп’ютера і обчислювальних потужностей, але із привабливою знижкою. Дуже вигідний варіант для середніх і великих компаній [11].

2.5.2. Elastic Block Storage

Частиною EC2 сервісу є AWS EBS (Elastic Block Storage).

Amazon EBS дозволяє створювати томи сховища та підключати їх до інстансів Amazon EC2. Після підключення томів можна створювати на них файлові системи, запускати бази даних або використовувати іншим чином, що підходить для блокових сховищ. Тома Amazon EBS розташовані в конкретних зонах доступності, всередині яких вони автоматично реплікуються, щоб захистити інформацію у разі збоїв окремих елементів обладнання. Всі типи томів EBS мають однаково надійні можливості створення знімків стану і забезпечують доступність на рівні 99,999%.

Amazon EBS надає сім типів томів: томи Provisioned IOPS SSD (io2 Block Express io2 та io1), універсальні томи SSD (io1 та gp3), томи Throughput Optimized HDD з оптимізованою пропускною здатністю (gp2) та «холодні» томи Cold HDD (st1). Ці типи томів мають різні характеристики продуктивності та різну ціну, завдяки чому ви можете адаптувати продуктивність і вартість сховища до потреб ваших додатків [12]. Середня затримка між інстансами EC2 та EBS становить частки мілісекунд.

2.5.3. Elastic Load Balancing

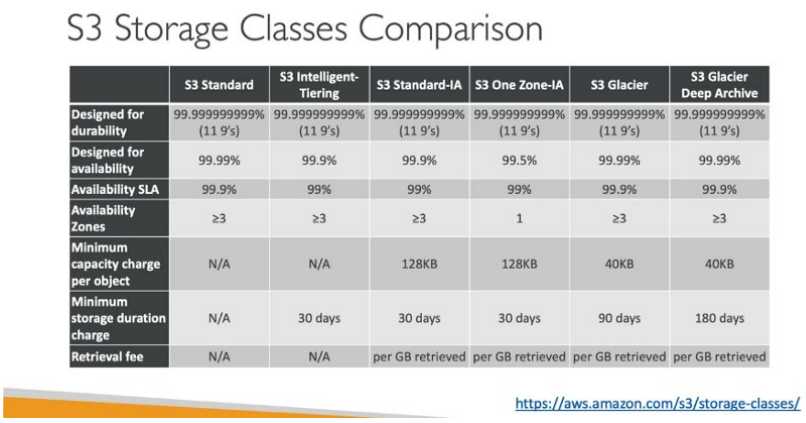
Ще один сервіс із сімейства EC2 – AWS ELB (Elastic Load Balancing) - сервіс, який автоматично розподіляє вхідний трафік додатків між кількома інстансами Amazon EC2. ELB може виявляти інстанси, що впали, і автоматично перенаправляти трафік на працюючі, поки непрацюючі інстанси знову не ввійдуть в експлуатацію, що, безумовно, підвищує надійність всієї системи в цілому. Він дає можливість досягти стійкості до відмови в додатках, ефективно виділяючи обсяг ресурсів, необхідний для розподілу навантаження відповідно до обсягу вхідного трафіку додатків.

Amazon Simple Storage Service (Amazon S3) — це служба зберігання об’єктів, яка пропонує найкращі в галузі масштабованість, доступність даних,

безпеку та продуктивність. Клієнти будь-якого розміру та галузі можуть використовувати Amazon S3 для зберігання та захисту будь-якої кількості даних для різноманітних випадків використання, таких як веб-сайти, мобільні програми, резервне копіювання та відновлення, архівування, корпоративні програми, пристрої IoT та великі дані. аналітика. Amazon S3 надає функції керування, щоб ви могли оптимізувати, упорядковувати та налаштовувати доступ до своїх даних відповідно до конкретних бізнес-вимог, організаційних вимог і вимог відповідності [13].

Сервіс дозволяє вибрати конфігурацію послуги/ціна для абсолютно всіх користувачів.

Нижче зображена таблиця цін і конфігурацій в вигляді S3 класів.



	S3 Standard	S3 Intelligent-Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved

Рис.2.4. Порівння класів S3

1)Amazon S3 Standard.

Він використовується для загальних цілей і забезпечує високу довговічність, доступність і продуктивність зберігання об'єктів для часто використовуваних даних. Standard S3 підходить для широкого спектру випадків використання, включаючи хмарні програми, динамічні веб-сайти, розповсюдження контенту, мобільні та ігрові програми та аналітику великих даних. Критерії доступності - 99,99%.

2)Amazon S3 Intelligent-Tiering.

Перше хмарне сховище автоматично зменшує вартість зберігання користувача. Це забезпечує дуже економічно ефективний доступ на основі частоти, не впливаючи на інші характеристики. Він також справляється з важкими операціями. Amazon S3 Intelligent – Tiering автоматично зменшує вартість гранульованих об’єктів. Плата за отримання в Amazon S3 Intelligent – Tiering не стягується. Критерії доступності - 99,9%.

3)Amazon S3 Standard-Infrequent Access.

Для доступу до даних, які використовуються менш часто, користувачі використовують S3 Standard-IA. За потреби якщо, потрібен швидкий доступ. Ми можемо досягти високої потужності, високої продуктивності та низької затримки за допомогою S3 Standard-IA. Це найкраще для зберігання резервних копій і відновлення даних протягом тривалого часу. Він діє як сховище даних для файлів аварійного відновлення. Критерії доступності - 99,9%.

4)Amazon S3 One Zone-Infrequent Access.

На відміну від інших класів зберігання S3, які зберігають дані щонайменше в трьох зонах доступності, S3 One Zone-IA зберігає дані в одній зоні доступності та коштує на 20% менше, ніж S3 Standard-IA. Це дуже хороший вибір для зберігання вторинних резервних копій локальних даних або даних, які легко відтворити. S3 One Zone-IA забезпечує таку ж високу довговічність, високу пропускну здатність і низьку затримку, як і в S3 Standard. Критерії доступності - 99,5%.

5)Amazon S3 Glacier.

Це клас архівних сховищ, який забезпечує найнижчу вартість зберігання для архівування даних і організований так, щоб забезпечити вам найвищу продуктивність і більшу гнучкість. S3 Glacier забезпечує найшвидший доступ до архівного сховища. Так само, як і в стандарті S3, отримання даних за мілісекунди. Критерії доступності - 99,99%.

6)Amazon S3 Glacier Deep Archive.

Клас зберігання Glacier Deep Archive розроблено для забезпечення тривалого та безпечного тривалого зберігання великих обсягів даних за ціною,

яка є конкурентоспроможною з дуже дешевими послугами зовнішнього архівування на стрічках. Вам більше не потрібно мати справу з дорогими послугами. Доступність дуже ефективна, оскільки вона може відновити дані протягом 12 годин. Цей клас сховища розроблено таким чином, що користувачі можуть легко отримати довговічне та надійніше сховище для величезної кількості даних за дуже меншу ціну [13]. S3 Glacier Deep Archive також має функцію реплікації об’єктів. Критерії доступності - 99,99%.

Для всіх типів сховища критерій збереження даних – 99,999999999%.

2.5.4. Amazon DynamoDB

Amazon DynamoDB – це повністю керований сервіс бази даних NoSQL, що забезпечує прогнозовану продуктивність із ефективною масштабованістю. Amazon DynamoDB дає клієнтам можливість перенести на AWS адміністративне навантаження, пов’язану з розподіленими базами даних і їх масштабуванням, тобто управлінням, виключеним з необхідності введення обладнання в експлуатацію, його налаштувань і конфігурації, реплікації, оновлення ПО і масштабування кластерів [14].

В Amazon DynamoDB використовуються такі моделі даних:

1)Таблиця: представляє собою загальну сукупність елементів даних, подібну загальну кількість рядка в таблиці реляційної бази даних. Кожна таблиця може мати безкінечну кількість елементів даних. Amazon DynamoDB – сервіс з гнучким описом даних, де елементи даних у таблиці не зобов’язані володіти однаковими атрибутами або навіть рівною кількістю атрибутів. Кожна таблиця повинна мати свій первинний ключ.

2)Елемент: елемент складається з первинного або складного ключа та змінної кількості атрибутів. Явно заданих обмежень за кількістю атрибутів, пов'язаних з окремим елементом, не існує, однак агрегований розмір елемента, включаючи всі імена та значення атрибутів, не повинен перевищувати 400 КБ.

3) Атрибут: кожен атрибут, пов'язаний з елементом даних, складається з імені атрибута (наприклад, «колір») та значення або набору значень (наприклад, «червоний» або «червоний, жовтий, зелений») [14]. Окремі атрибути не мають явно заданих обмежень за розміром, але загальне значення елемента (включаючи усі імена та значення атрибуту) не повинно перевищувати 400 КБ.

2.5.5. Amazon CloudFront

Amazon CloudFront – це сервіс глобальної мережі доставки контенту (CDN), що забезпечує швидку та безпечну передачу даних, відео, додатків та API клієнтам.

Сервіс CloudFront інтегрований з AWS: його фізичні місцезнаходження безпосередньо підключені до глобальної інфраструктури AWS, а програмне забезпечення працює з іншими сервісами, включаючи AWS Shield для нейтралізації DDoS-атак, Amazon S3, Elastic Load Balancing або Amazon EC2 як сервери-джерела для додатків, а також Lambda@Edge для запуску спеціального коду у безпосередній близькості до кінцевих користувачів [15].

Для прикладу, Якщо є S3 bucket в Європі, а користувач намагається завантажити файл зі США, то завдяки функціоналу S3 Transfer Acceleration система направить його на найближчий edge location, що дозволить завантажити файл на платформу швидше, а потім він спеціалізованою приватною мережею Amazon з високою доступністю і низьким рівнем затримок потрапить у потрібний регіон.

2.5.6. Amazon CloudWatch

Amazon CloudWatch – це сервіс моніторингу хмарних ресурсів AWS та програм, що працюють на AWS. Можна використовувати Amazon CloudWatch для збирання та відстеження метрик, накопичення та аналізу логів, а також для

створення попереджень. Amazon CloudWatch може вести моніторинг ресурсів AWS (таких як інстанси Amazon EC2, таблиці Amazon DynamoDB, інстанси БД Amazon RDS), метрик додатків і сервісів, а також моніторинг будь-яких журналів додатків. Amazon CloudWatch можна використовувати для отримання зведеної інформації про систему, включаючи інформацію про ресурси, продуктивність додатків і загальну працездатність системи [16]. Ці дані застосовуються для оперативного реагування та забезпечення стабільної роботи програм.

Для початку моніторингу можна використовувати Automatic Dashboards із вбудованими рекомендаціями AWS, вивчити подання метрик та оповіщень на основі акаунту та ресурсів, а також легко провести детальне дослідження, щоб з'ясувати першопричину проблем з продуктивністю.

CloudWatch надає корисну інформацію, яка дозволяє оптимізувати продуктивність додатків, керувати використанням ресурсів та оцінювати працездатність системи загалом. CloudWatch оновлює метрики та дані журналів з інтервалом за одну секунду, забезпечує зберігання даних (метрик) протягом 15 місяців і дає можливість виконувати розрахунки з використанням метрик. Це дозволяє аналізувати історичні дані для оптимізації витрат та отримувати в режимі реального часу аналітику для оптимізації додатків та ресурсів інфраструктури. CloudWatch Container Insights можна використовувати для моніторингу контейнерних програм та мікросервісів, а також усунення несправностей у них та налаштування оповіщень [16]. CloudWatch збирає, зберігає та узагальнює дані про використання процесора, пам'яті, диска та мережі, а також діагностичну інформацію, наприклад про збої при перезапуску контейнера, щоб допомогти інженерам знаходити проблеми та швидко усувати їх.

2.5.7. Amazon Route 53

Amazon Route 53 – це високодоступний хмарний веб-сервіс системи доменних імен (DNS), що масштабується. Розробники та власники веб-сервісів використовують його як виключно надійний та економічний спосіб перенаправлення кінцевих користувачів до інтернет-додатків за рахунок перетворення доменних імен (наприклад, www.example.com) у формат цифрових IP-адрес (наприклад, 192.0.2.1), що використовуються комп'ютерами. При цьому Amazon Route 53 повністю сумісний із протоколом IPv6.

Сервіс Amazon Route 53 надсилає запити користувачів до інфраструктури AWS, наприклад до інстансів Amazon EC2, балансувальників навантаження Elastic Load Balancing або кошиків Amazon S3. Крім того, він може використовуватися для перенаправлення користувачів на інфраструктуру за межами AWS. За допомогою Amazon Route 53 можна налаштувати перевірки працездатності DNS, а потім безперервно відстежувати та керувати відновленням програм через контролер Route 53 Application Recovery Controller.

За допомогою сервісу Amazon Route 53 Traffic Flow можна просто управляти глобальним трафіком, використовуючи різні типи маршрутизації (такі як маршрутизація на базі затримки, DNS з урахуванням географічного положення, географічна близькість та циклічний зважений алгоритм), поєднуючи їх з можливістю перекидання сервісу DNS і створюючи в внаслідок відмовостійкої архітектури з низькою затримкою. Використовуючи нескладний візуальний редактор Amazon Route 53 Traffic Flow, можна просто керувати маршрутизацією кінцевих користувачів до адрес додатків як в рамках одного регіону AWS, так і при розподілі трафіку по всьому світу. Крім того, у сервісі Amazon Route 53 можна зареєструвати доменне ім'я: при покупці доменів (наприклад, example.com) та керуванні ними Amazon Route 53 автоматично налаштує для них параметри DNS [17].

В сервісі наявні сконфігуровані політики перенаправлення:

Проста політика маршрутизації – використовуйте для одного ресурсу, який виконує задану функцію для вашого домену, наприклад, веб-сервера, який обслуговує вміст для веб-сайту example.com. Ви можете використовувати просту маршрутизацію для створення записів у приватній зоні розміщення.

Політика маршрутизації після відмови – використовуйте, якщо потрібно налаштувати активне-пасивне відновлення після відмови. Для створення записів у приватній зоні розміщення можна використовувати маршрутизацію після відмови.

Політика маршрутизації за геолокацією – використовуйте, якщо ви хочете скеровувати трафік на основі розташування ваших користувачів. Ви можете використовувати геолокаційну маршрутизацію для створення записів у приватній зоні розміщення.

Політика маршрутизації геопроксимації – використовуйте, коли ви хочете спрямувати трафік на основі розташування ваших ресурсів і, за бажанням, перемістити трафік від ресурсів в одному місці до ресурсів в іншому.

Політика маршрутизації затримки – використовуйте, якщо у вас є ресурси в кількох регіонах AWS і ви хочете скерувати трафік до регіону, який забезпечує найменшу затримку. Ви можете використовувати маршрутизацію затримки для створення записів у приватній зоні розміщення.

Політика маршрутизації на основі IP – використовуйте, якщо ви хочете маршрутизувати трафік на основі розташування ваших користувачів і мати IP-адреси, з яких надходить трафік.

Політика маршрутизації багатозначних відповідей – використовуйте, якщо потрібно, щоб Route 53 відповідав на запити DNS до восьми справних записів, вибраних випадковим чином. Ви можете використовувати багатозначну маршрутизацію відповіді для створення записів у приватній зоні розміщення.

Політика зваженої маршрутизації – використовуйте для маршрутизації трафіку до кількох ресурсів у вказаних пропорціях [17]. Ви можете використовувати зважену маршрутизацію для створення записів у приватній зоні розміщення.

2.5.8. Amazon Virtual Private Cloud

Amazon Virtual Private Cloud (VPC) – це сервіс, який дає можливість запускати ресурси AWS у логічно ізольованій віртуальній мережі, що визначається користувачем. Це дозволяє повністю контролювати середовище віртуальної мережі, у тому числі вибирати власний діапазон IP-адрес, створювати підмережі, а також налаштовувати таблиці маршрутизації та мережеві шлюзи. Для більшості ресурсів VPC можна використовувати і IPv4, і IPv6, і таким чином отримати безпечний і зручний доступ до ресурсів і додатків.

Як один із основних сервісів AWS, Amazon VPC спрощує індивідуальне налаштування параметрів мережі VPC. Можна створити загальнодоступну мережу для своїх веб-серверів з доступом до Інтернету. Можна також помістити свої серверні системи, такі як бази даних або сервери додатків, у приватну мережу без доступу до Інтернету. Amazon VPC дає можливість використовувати багаторівневу систему безпеки, яка складається з груп безпеки та мережевих списків контролю доступу. Така система дозволяє контролювати доступ до інстансів Amazon Elastic Compute Cloud (Amazon EC2) у кожній підмережі [18].

За допомогою цього сервісу і його можливостей можна зробити дуже захищену мережу для робочого середовища або продукту компанії, а саме:

1)Журнали потоків.

Моніторинг журналів потоків VPC, що доставляються в Amazon Simple Storage Service (Amazon S3) або Amazon CloudWatch, забезпечує операційний контроль мережевих залежностей та моделей трафіку, виявлення аномалій та

запобігання витоку даних та усунення несправностей мережних підключень та проблем конфігурації. Збагачені метадані в журналах потоків допомагають дізнатися більше про ініціатор TCP-підключень, джерело на рівні пакетів і точку призначення трафіку, що проходить за середніми рівнями, таким як шлюз NAT.

2) IP Address Manager (IPAM).

За допомогою IPAM легше планувати, відстежувати та перевіряти IP-адреси для робочих навантажень AWS. IPAM автоматизує IP-адресу для Amazon VPC, дозволяючи більше не використовувати власні програми для планування або програми на базі електронних таблиць. Крім того, сервіс комплексно демонструє використання IP-адрес у кількох акаунтах та VPC, за рахунок чого підвищується можливість відстеження мережі.

3) IP-адресація.

За допомогою IP-адрес ресурси VPC взаємодіють один з одним і з ресурсами, розташованими в Інтернеті. Amazon VPC підтримує як протокол адресації IPv4, так і IPv6. У VPC можна створити підмережі, призначені тільки для IPv4, тільки для IPv6 або з подвійним стеком, а потім запускати інстанси Amazon EC2 в межах цих підмереж. Amazon також пропонує різні способи призначення публічних IP-адрес для інстансів. Ви можете використовувати надані Amazon публічні IPv4-адреси, еластичні IPv4-адреси або IP-адреси із запропонованих Amazon діапазонів CIDR IPv6. Крім того, можна додати до Amazon VPC та призначити інстансам власні адреси IPv4 або IPv6.

4) Вхідна маршрутизація.

Ця можливість дозволяє спрямовувати вхідний та вихідний трафік, що проходить через шлюз Інтернету або шлюз віртуальної приватної мережі, до еластичного мережного інтерфейсу конкретного інстансу Amazon EC2. Virtual private cloud можна налаштувати таким чином, щоб весь трафік прямував до шлюзу або інстансу Amazon EC2, перш ніж потрапить у робочі навантаження.

5) Network Access Analyzer.

За допомогою Network Access Analyzer можна перевірити, чи мережа відповідає в AWS вимогам мережної безпеки та іншим нормативним вимогам. Network Access Analyzer дозволяє вказати необхідні вимоги до мережної безпеки та нормативні вимоги, а також здатний виявити ненавмисний доступ до мережі, яка їм не відповідає. Network Access Analyzer використовується для аналізу мережного доступу до ресурсів, який допомагає визначити, які покращення варто внести до хмарної системи безпеки та продемонструвати відповідність вимогам [18].

6) Групи безпеки.

Групи безпеки діють як брандмауер для пов'язаних інстансів Amazon EC2 і контролюють вхідний та вихідний трафік на рівні інстансу. Під час запуску інстансу можна зв'язати його з однією або декількома групами безпеки. Якщо група не вибрана, інстанс автоматично зв'яжеться з групою, яку VPC пропонує за промовчанням. Кожен інстанс VPC може належати до окремого набору груп.

2.6. Docker

Docker – це програмна платформа для швидкої розробки, тестування та розгортання програм. Docker упаковує програмне забезпечення у стандартизовані блоки, які називаються контейнерами. Кожен контейнер включає все необхідне для застосування: бібліотеки, системні інструменти, код і середовище виконання. Завдяки Docker можна швидко розгортати та масштабувати програми у будь-якому середовищі та зберігати впевненість у тому, що код працюватиме.

Використання Docker на AWS надає розробникам та системним адміністраторам надійний та економічний спосіб складання, доставки та запуску розподілених програм будь-якого масштабу.

В основі роботи Docker лежить стандартизований спосіб виконання коду. Docker – це операційна система контейнерів. Подібно до того, як

віртуальна машина створює віртуальне уявлення апаратного забезпечення сервера (тобто усуває необхідність безпосередньо керувати таким), контейнери створюють віртуальне уявлення серверної операційної системи. Після встановлення на кожен сервер Docker надає доступ до простих команд, необхідних для збирання, запуску або зупинки контейнерів [19].

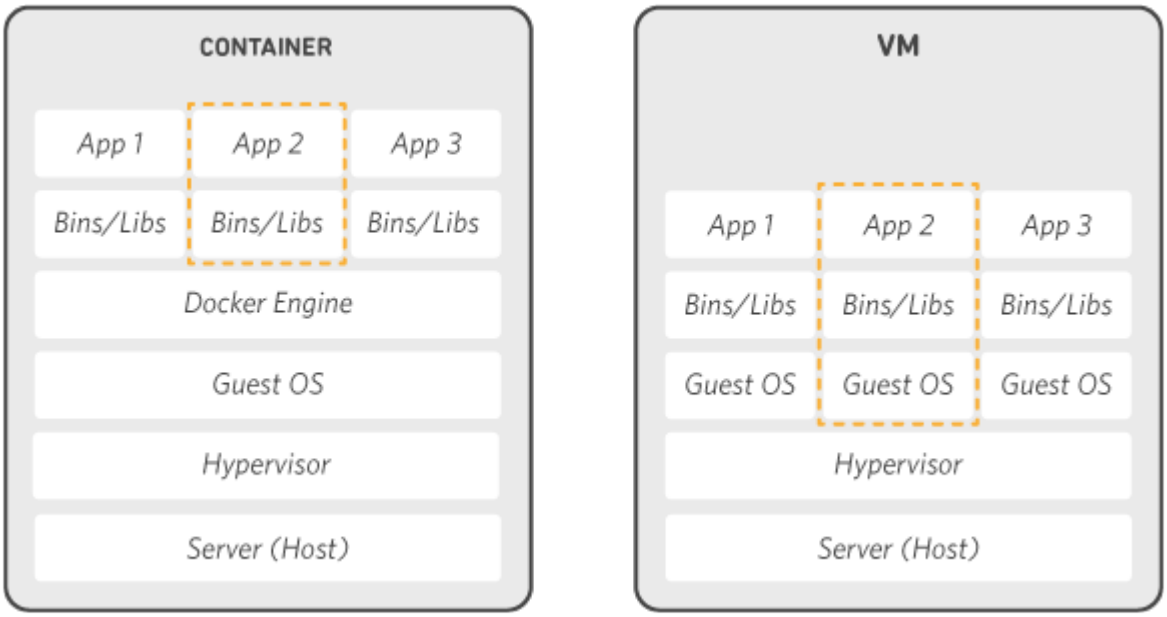


Рис. 2.5. Порівння технології Docker та віртуалізації

Контейнери — це спосіб стандартизації розгорнення програми та відокремлення її від загальної інфраструктури. Примірник програми запускається в ізольованому середовищі, що не впливає на основну операційну систему.

Розробникам не потрібно замислюватися, в якому оточенні буде працювати їхня програма, чи будуть там потрібні налаштування та залежності. Вони просто створюють додаток, упаковують усі залежності та налаштування у певний єдиний образ. Потім цей образ можна запускати на інших системах, не турбуючись, що програма не запуститься.

Контейнеризація схожа на віртуалізацію, але це не одне й те саме. Віртуалізація запускає повноцінний хост на гіпервізорі зі своїм віртуальним обладнанням та операційною системою. При цьому всередині однієї ОС

можна запустити іншу ОС. У разі контейнеризації процес запускається прямо з ядра основної операційної системи та не віртуалізує обладнання. Це означає, що контейнеризоване додаток може працювати тільки в тій самій ОС, що й основна. Контейнери не віртуалізують обладнання, тому споживають менше ресурсів [19].

Docker вирішує проблеми залежностей та робочого оточення.

Контейнери дозволяють запакувати в єдиний образ програму та всі її залежності: бібліотеки, системні утиліти та файли налаштування. Це полегшує перенесення програми на іншу інфраструктуру.

Наприклад, розробники створюють додаток у системі розробки - там все налаштовано, програма працює. Коли воно готове, його потрібно перенести в систему тестування, а потім у продуктивне середовище. Якщо в одній з них немає потрібної залежності, програма не працюватиме. Програмістам доведеться відволіктися від розробки та спільно з командою підтримки розібратися в ситуації.

У контейнерах такої проблеми немає, оскільки вони містять все необхідне для запуску програми. Фахівці займаються розробкою, а не розв'язанням інфраструктурних проблем.

Контейнер — це набір процесів ізольованих від основної операційної системи. Програми працюють лише всередині контейнерів і не мають доступу до основної операційної системи. Це підвищує безпеку додатків: вони не зможуть випадково чи навмисне нашкодити основній системі. Якщо програма в контейнері завершиться з помилкою або зависне, це ніяк не торкнеться основної ОС.

Прискорення та автоматизація розгортання додатків та масштабованість.

Контейнери полегшують розгортання додатків. У класичному підході для встановлення програми потрібно зробити кілька дій: виконати скрипт, змінити налаштування файлів і так далі. У цьому процесі не виключена можливість людської помилки: користувач запустить скрипт двічі, переплутає

послідовність або щось не зрозуміє. Контейнери дозволяють повністю автоматизувати цей процес, оскільки включають всі необхідні залежності і порядок виконання дій [20].

Також контейнери полегшують розгортання на декількох серверах. У класичному підході для того, щоб розгорнути один і той же додаток на декількох машинах, потрібно буде повторювати одні й ті самі дії. Контейнери позбавляють цієї рутинної роботи і дозволяють автоматизувати розгортання.

Контейнери наближають до мікросервісної архітектури.

Контейнери добре вписуються у мікросервісну архітектуру. Це підхід до розробки, у якому додаток розбивається на невеликі компоненти, наскільки можна незалежні. Зазвичай протиставляється монолітної архітектури, де всі частини системи сильно пов'язані одна з одною.

Це дозволяє розробляти нову функціональність швидше, адже у випадку з монолітною архітектурою зміна якоїсь частини може торкнутися всієї іншої системи.

ВИСНОВОК ДО РОЗДІЛУ 2

У другому розділі були розглянуті інструменти, технології та сервіси, що будуть використовуватися для забезпечення безвідмовності додатку.

Для практичного прикладу був вибраний Телеграм, тому що це найпоширеніший месенджер з великими можливостями для розробників. А саме дуже приваблива можливість використання Телеграм ботів.

Основною мовою для написання додатку вибраний Python, так як це найпрогресивніша і технологічна мова.

Базою для технічної частини було вибрано можливості платформи AWS. Тому, що це комплексна платформа з дуже великим спектром вже готових рішень для багатьох задач (на сьогодні там нараховується більше 200 сервісів).

Були описані вибрана операційна система з обґрунтуванням та використання такої технології як Docker.

РОЗДІЛ 3. ПРАКТИЧНИЙ ПРИКЛАД СТВОРЕННЯ ІНФРАСТУРКТУРИ

Для практичного прикладу буде використовуватись розроблений на Python телеграм бот для якого потрібно створити середовище, яке буде максимально безвідмовне і незалежне від зовнішніх чинників. Під «безвідмовністю» мається на увазі не абсолютно безвідмовне середовище, тому що це не можливо, а максимально наближені умови до безвідмовності. А в разі відмови максимально короткий термін відновлення функцій та працездатності, бажано без впливу на користувачів. Більше того, що умови та характеристики середовища будуть змінюватись залежно від збільшення чи зменшення навантаження, а це в свою чергу регулюється потребами клієнтів та їх кількістю.

Виходячи з вищевказаного, на мою думку, найкраще підійде для цього хмарна платформа Amazon Web Services. Вона пропонує нам безмежні, швидкі, легко масштабовані ресурси, технічну підтримку та багато іншого за прийнятними цінами.

3.1. Створення та налаштування root акаунту в AWS

Перш за все, щоб мати можливість користуватись послугами AWS, нам потрібно створити акаунт на платформі. Це можна зробити легко та швидко за 5 кроків.

Крок 1 – Використати або зареєструвати працюючу пошту, так як потрібно буде підтвердження.

Крок 2 – Придумати комплексний та надійний пароль. Заповнити основі данні про себе, та вибрати робочий профіль, для приватного користування або бізнесу.

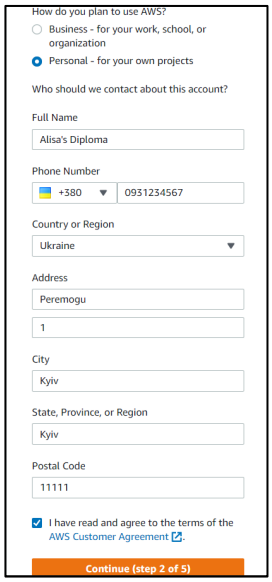


Рис. 3.1. Перший і другий крок створення акаунту

Крок 3 – Додати данні про картку, з якої буде знято 1 долар, для верифікації, що це реальна картка і на ній є гроші, за тиждень гроші повернуться на картку. За політикою AWS це обов’язковий крок.

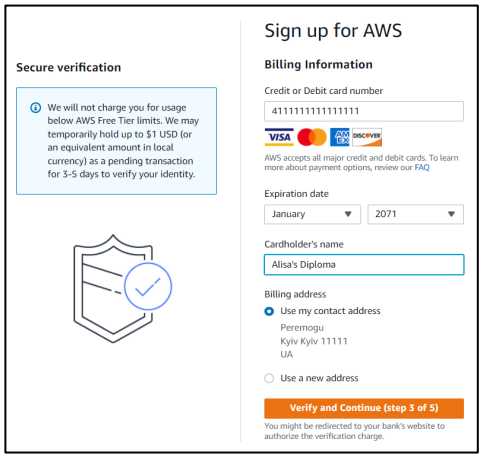


Рис. 3.2. Третій крок створення акаунту

Крок 4 – Це підтвердження контактного мобільного номеру телефону. На нього прийде повідомлення з кодом або телефонний дзвінок. Це також обов'язковий крок.

Крок 5 – Вибір плану підтримки. Цей крок є важливим для бізнесу. Але для нас достатньо вибрати Basic Plan. Він створений спеціально для знайомства та включає в себе деякі ліміти на безкоштовне використання сервісів.

На цьому все, ми створили акаунт AWS який активується за лічені секунди, проте цей процес може тривати до 24 годин (така практика для бізнес акаунтів з найкращим планом технічної підтримки).

Створений нами акаунт – це ROOT акаунт, він має максимальний і повний доступ до всього. Для подальших дій потрібно зробити цей акаунт максильно захищеним від шахраїв і не використовувати його для роботи з сервісами AWS. Один із способів підвищити рівень захисту це налаштувати MFA (Multi-factor authentication) багатофакторну автентифікацію і робити ми це будемо в декілька кроків за допомогою мобільного додатку Google Authenticator.

1) Встановити Google Authenticator на смартфон.
2) Зайти у AWS консоль в пункт «My Security Credentials» -> «Assign MFA Device».

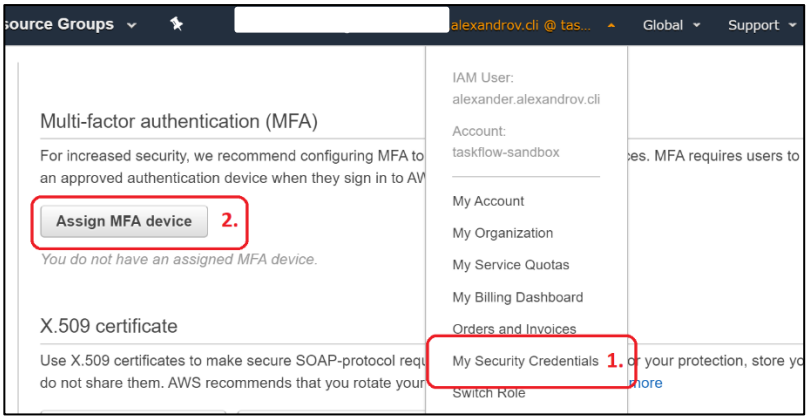


Рис. 3.3. Вікно ввімкнення двухфакторної автентифікації

3) Вибираємо Virtual MFA Device

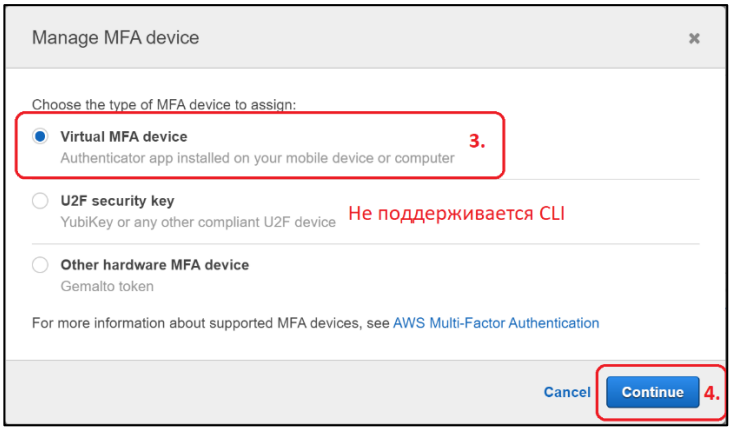


Рис. 3.4. Вікно вибору типу двухфакторної автентифікації

4) Далі слідуємо інструкція і скануємо QR код додатком на телефоні.

З цього моменту під час входу в наш ROOT AWS акаунт крім звичної пошти і паролю потрібно буде вводити одноразовий код перевірки з додатку на телефоні, який змінюється кожних 60 секунд.

3.2. Створення та налаштування робочого акаунту

На цьому етапі нам потрібен акаунт який буде мати обмежені права, але достатні для виконання поставлених задач.

Для цього ми переходимо в IAM (Identity and Access Management) сервіс та вибираємо User Groups. За допомогою груп користувачів набагато легше давати права на ті чи інші ресурси кільком людям. Налаштувавши одну групу і просто додавши акаунти в цю групу, ми дамо їм вже налаштований список дозволів.

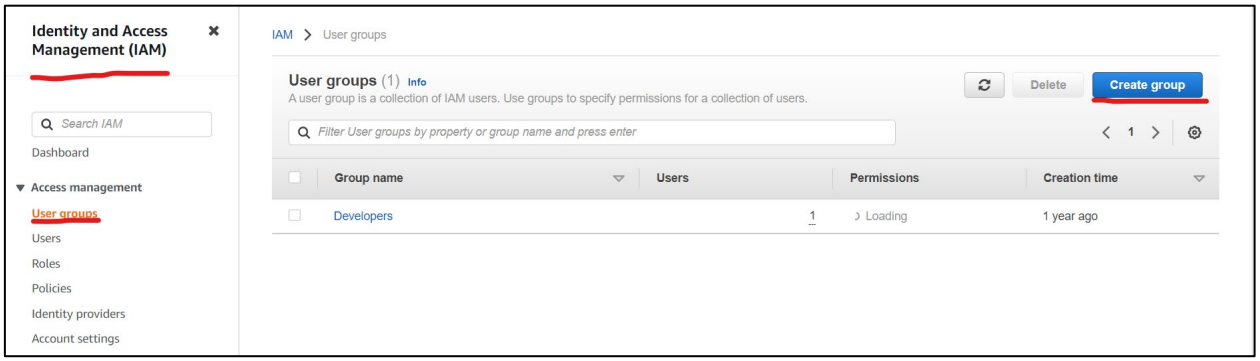


Рис. 3.5. Процес створення групової політики

Тут ми створюємо саму групу і вибираємо потрібні нам права, можна використовувати права які вже налаштовані AWS чи можна зібрати з найменших частинок свою політику прав.

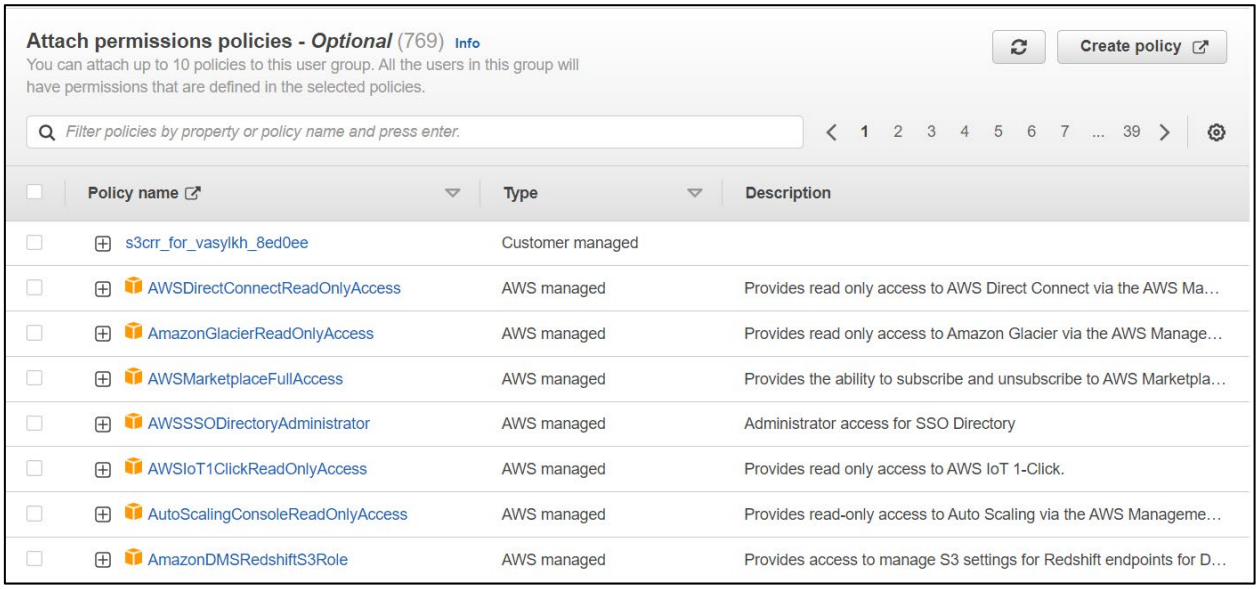


Рис. 3.6. Приклад наявних налаштованих політик

Як ми бачимо AWS має вже 769 налаштованих прав доступу. В нашому випадку для групи користувачів «Developers» ми дозволимо повний

доступ до всієї технічної частини, виключаючи частину фінансів і фінансових карток.

Тепер ми можемо створити робочий акаунт, який буде використовуватися для виконання всіх задач та робіт. Переходимо у вкладку Users і натискаємо Add Users.

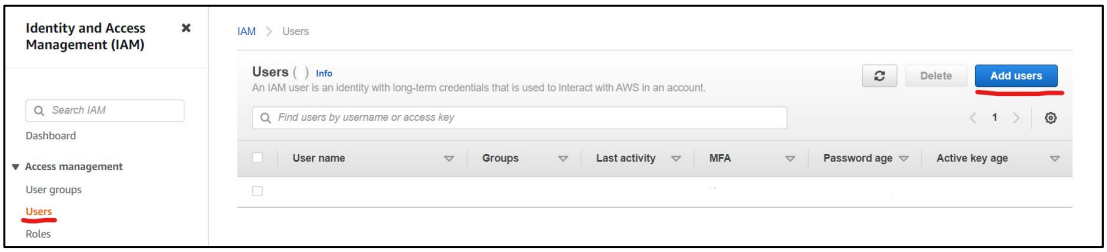


Рис. 3.7. Процес створення користувача

В цьому сервісі є можливість створити кілька акаунтів одразу, що полегшує і пришвидшує процес адміністрування. В нашому випадку буде достатньо одного користувача. Вибираємо 2 типи доступу, це - Access key - Programmatic access для використання в майбутньому AWS CLI з консолі та Password - AWS Management Console access для доступу в головну консоль управління.

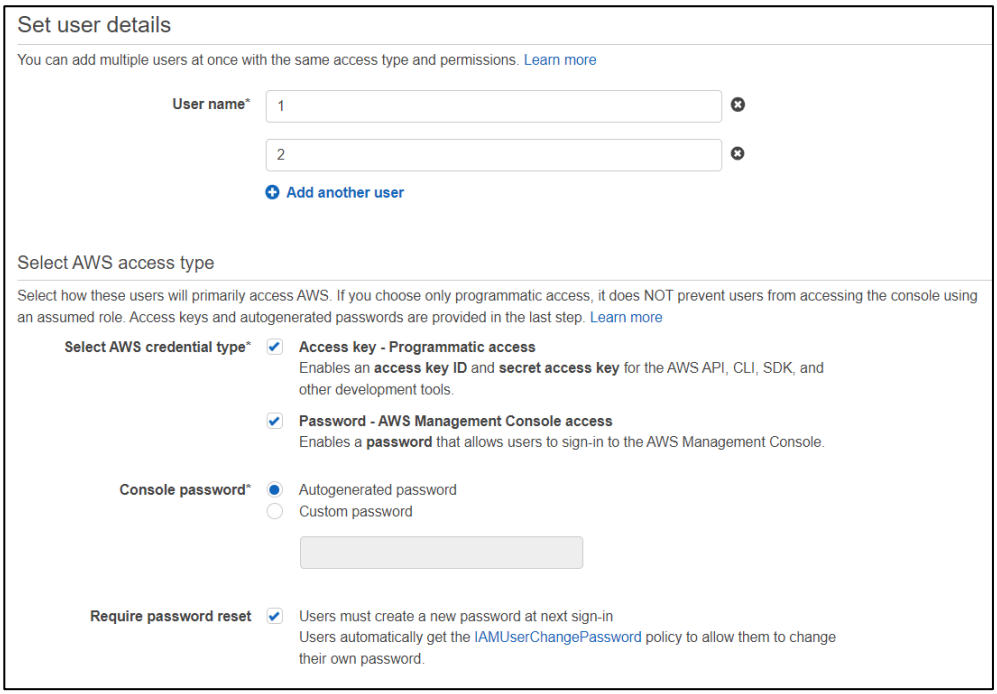


Рис. 3.8. Вікно створення користувача або користувачів

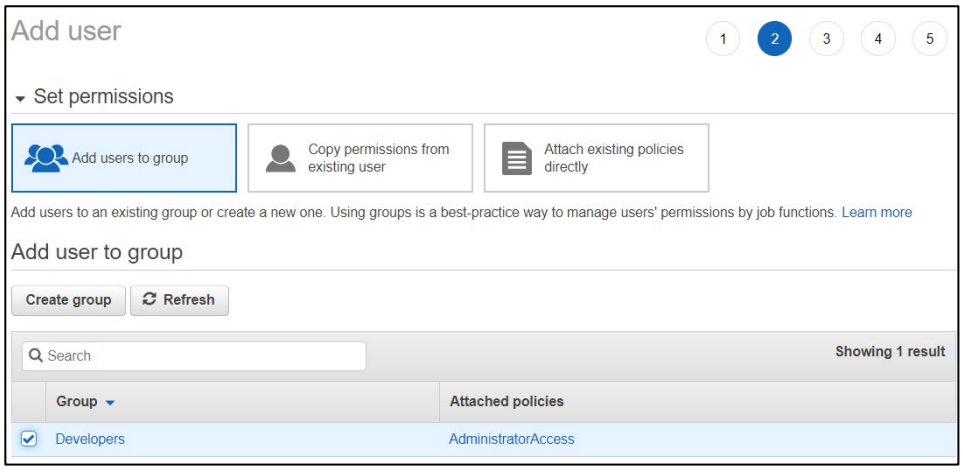


Рис. 3.9. Процес додавання користувача в групу дозволів

Вибираємо до якої групи додати користувача, в наступному вікні ми можемо додати теги до акаунту, якщо потрібно, і потім просто перевіряємо вибрані данні та налаштування і підтверджуємо створення.

Ми бачимо, що акаунт створився і тут є декілька важливих моментів. Тепер ми бачимо Access key ID, Secret access key, Password та Email login

instructions. За допомогою «Email login instructions» ми можемо відправити данні користувача і інструкцію по використанню безпосередньо новому користувачу. Варто всі ці данні десь зберегти, так як більше ми не зможемо їх побачити, у випадку втрати доступу потрібно буде генерувати новий пароль і ключ доступу.

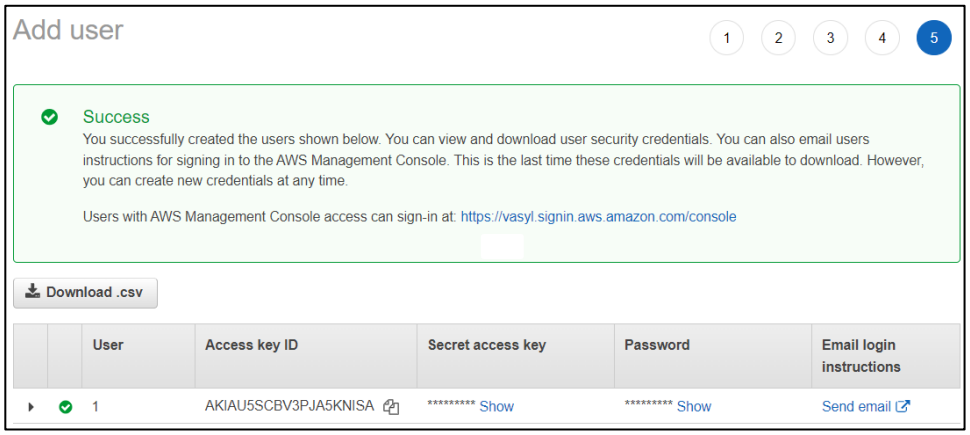


Рис. 3.10. Вікно з даними для входу нового акаунта

3.3. Налаштування VPC, Security Group та Volumes

Тепер нам потрібно зробити базові налаштування нашої приватної мережі по необхідним критеріям. Так як в нас буде всього 1 або 2 інстанси, тому нам достатньо найменшої мережі на 16 приватних IP адрес (/28 маска).

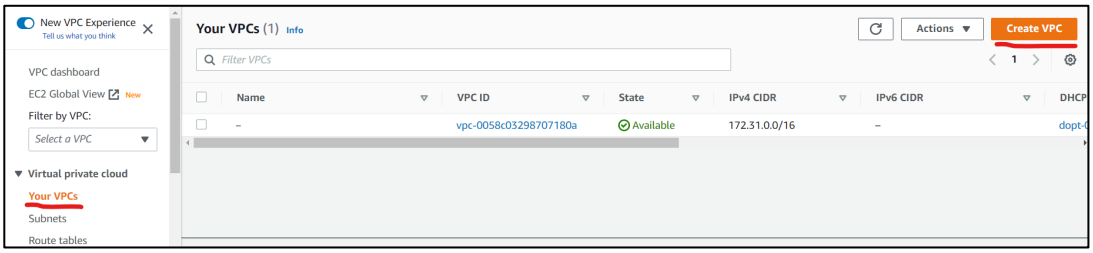


Рис. 3.11. Створення нової VPC

Заходимо в AWS VPC сервіс і створюємо нову приватну мережу. Це ще одна перевага перед фізичними серверами і мережевим обладнанням, в AWS

ми можемо це робити швидко і просто, а у випадку, якщо нам потрібно щось змінити або розширити мережу, то це також робиться моментально і просто.

Даємо бажану назву нашій мережі – Alisa’s Diploma. Задаємо бажаний IPv4 CIDR на 16 IP адрес (172.168.1.0/28). IPv6 використовувати не будемо. І додаємо теги до цього VPC для швидкого орієнтування по всім ресурсам AWS з тим чи іншим тегом.

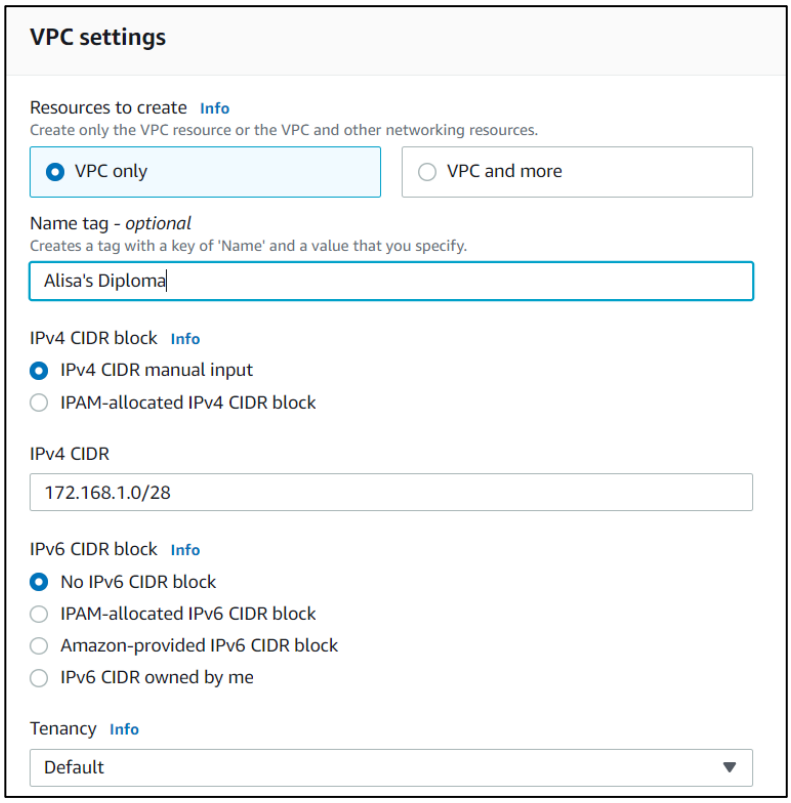


Рис. 3.12. Налаштування VPC

Тепер нам потрібно створити Security Group для того, щоб дозволити якийсь трафік для того чи іншого сервісу, в нашому випадку це інстанс в EC2 сервісі. Отож, в сервісі VPC знаходимо пункт Security Groups, переходимо і створюємо нову.

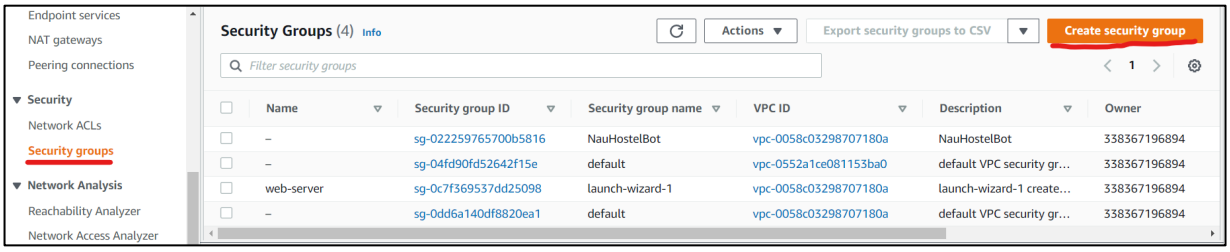


Рис. 3.13. Створення нової Security Group

Заповнюємо поля імені та опису вибираємо нашу новостворену приватну мережу VPC. За замовчуванням всі вхідні з'єднання заборонені, тому нам потрібно відкрити діапазон IP адрес і порти, які ми будемо використовувати. Для доступу на сервер нам потрібен SSH протокол (порт 22) зі всіх IP адрес (тому, що в мене динамічна зовнішня адреса, а не статична), проте є можливість дозволити лише з однієї або декількох статичних IP. Далі HTTPS протокол з номер порту 443 без обмеження IP та NFS з портом 2049, на випадок, якщо нам знадобиться підключити мережеву файлову систему.

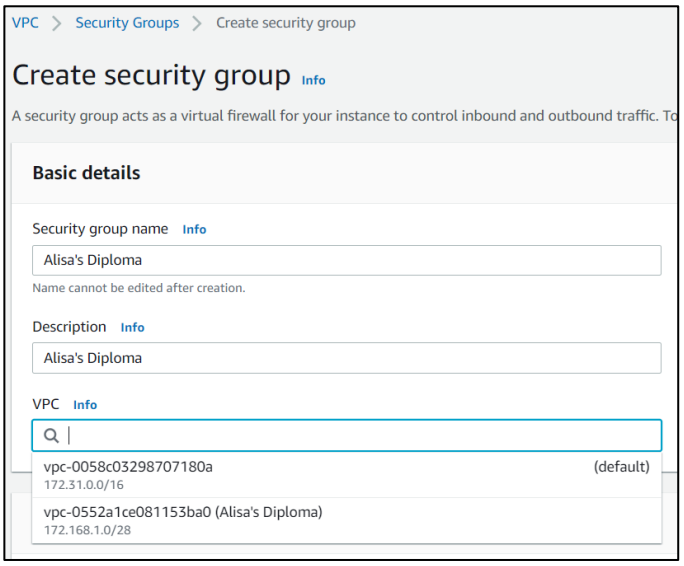


Рис. 3.14. Налаштування Security Group

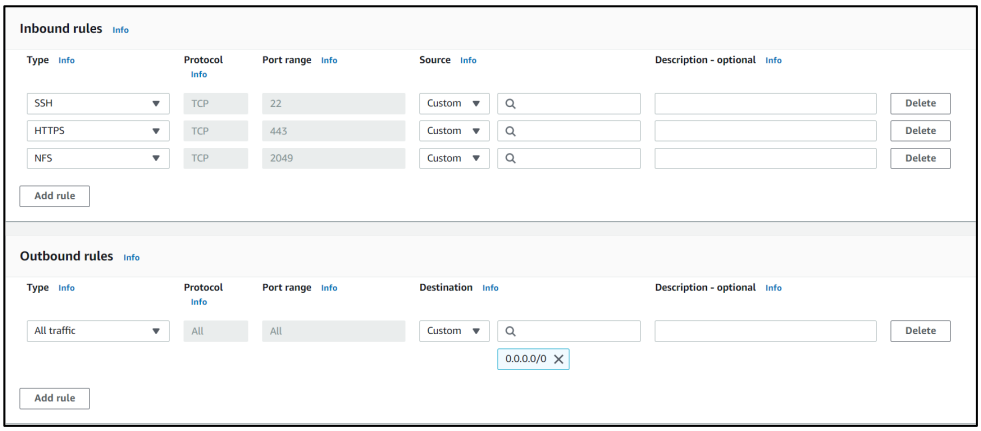


Рис. 3.15. Налаштування вхідних та вихідних мережевих правил

Правило вихідного трафіку можемо залишити як є. Хоча краще перенаправити увесь внутрішній трафік на фаєрвол, наприклад, проте тут немає такої необхідності. І звісно ж додаємо теги, для легшого адміністрування і пошуку всього, що відноситься до одного проекту чи програми.

Далі створюємо диск для нашого інстансу.

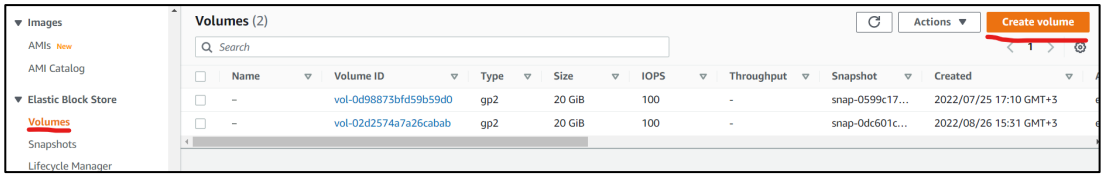


Рис. 3.16. Створення нового дискового простору

Так як ми будемо використовувати ОС на платформі Linux, то це означає, що для коректної роботи самої системи потрібно 8-10 ГБ диску. І додатково ще додаємо 20 ГБ для власних потреб і сервісних програм, що дає нам базових 100 IOPS з автоматичним масштабуванням до 3000 IOPS (для класу сховища gp2/gp3). Для класу gp2/gp3 максимальне значення IOPS = 16000 при розмірі диску 5333 ГБ. Для підвищення швидкодії системи і

програм буде використовувати SSD накопичувач (gp2). Але в нас є можливість вибрати між SSD і жорстким диском (HDD) різних поколінь і продуктивності.

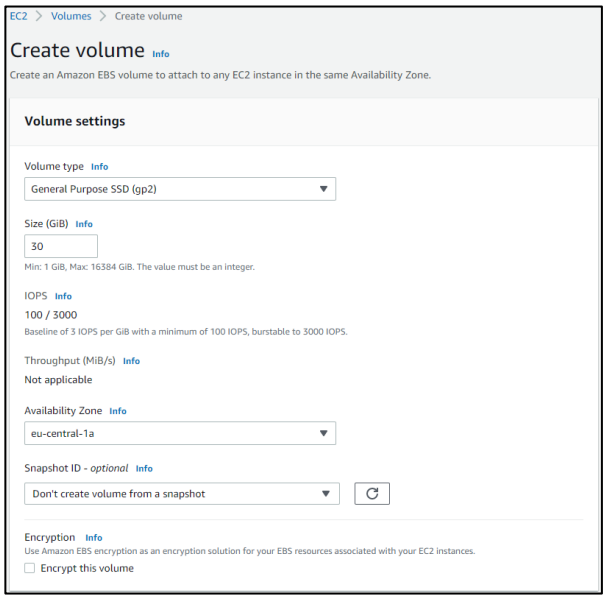


Рис. 3.17. Налаштування дискового простору

3.4 Вибір характеристик та створення інстансу

Переходимо у сервіс EC2 та вибираємо пункт Instances, далі Launch Instances.

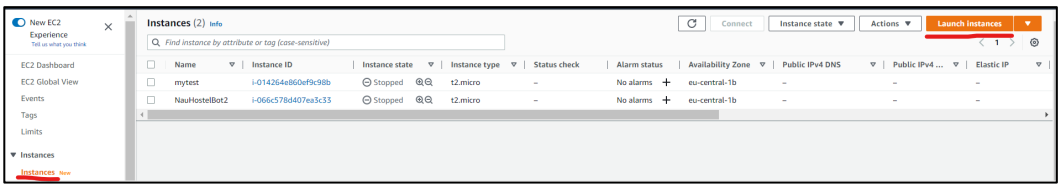


Рис. 3.18. Створення нашого інстансу

Придумуємо зрозуміле ім’я, далі потрібно вибрати ОС (MacOS, різні Linux дистрибутиви чи Windows Server) яка нам потрібна. Ми можемо зробити знімок з нашого вже працюючого інстансу, але в нас такого немає,

тому ми можемо взяти чисті дистрибутиви з AWS або вже зібрані під якісь потреби шаблон спільноти.

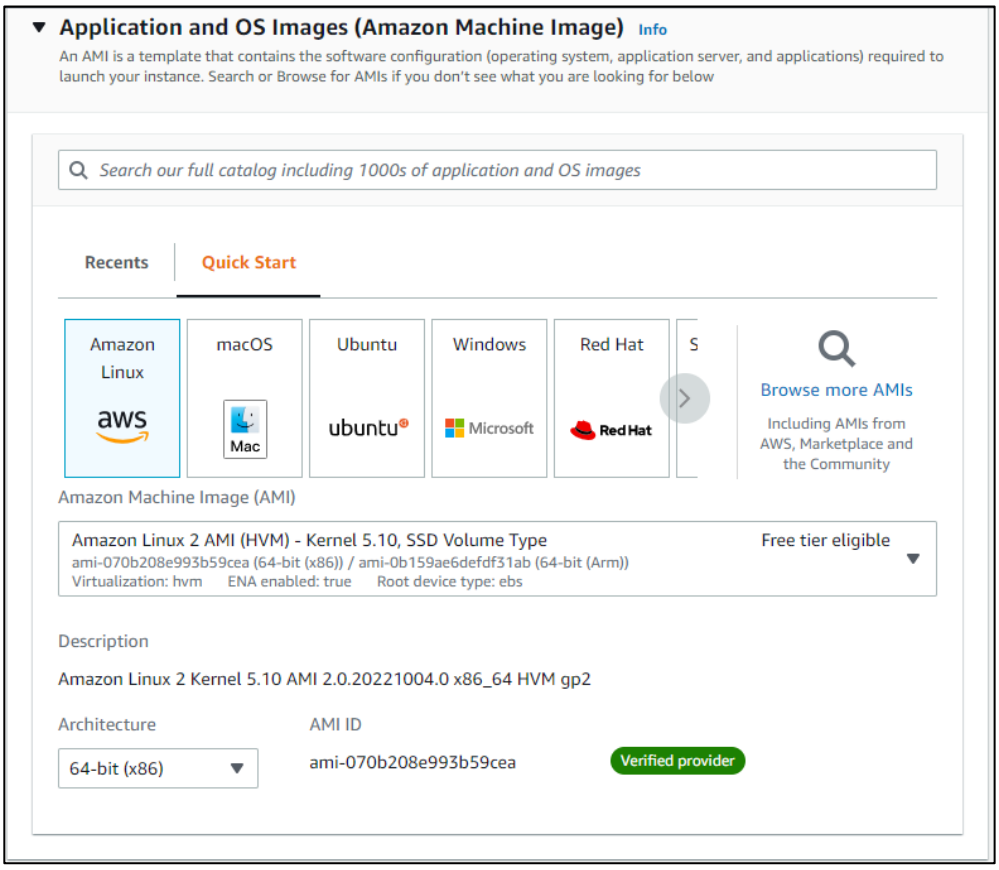


Рис. 3.19. Вибір параметрів для майбутнього інстансу

Як ми бачимо, загалом, нам доступні для вибору близько 6800 різних шаблонів, вони включають в себе Windows, Linux/UNIX та MacOS.

Але для нас підійде Amazon Linux 2-го покоління на ядрі Kernel 5.10 з 64-біиною архітектурою, тому що ОС базовані на Linux працюють швидше і базово потребують менше ресурсів, таких як RAM, CPU та інше і більше того ціна за Linux інстанс в півтора рази менше ніж за Windows.

Далі потрібно вибрати обладнання під наші потреби. Спрогнозувати реальне використання практично неможливо, але в цьому питанні нам і допоможе AWS, так як змінити тип інстансу доволі легко і швидко в залежності від навантаження.

Більше того є тимчасове масштабування, це коли за додаткову плату ви отримуєте більші потужності короткочасно коли потрібно більше ресурсів аніж включено в тип вибраного інстансу, цю функцію можна відключити, для уникнення додаткових витрат.

Підсумовуючи вище сказане, ми можемо вибрати інстанс типу t3.small, який включає в себе 2 віртуальних ядра, 2 ГБ оперативної пам’яті та швидкість інтернет каналу до 5 Гигабіт/с. За розрахунками споживання програми, цього має бути більш ніж достатньо.

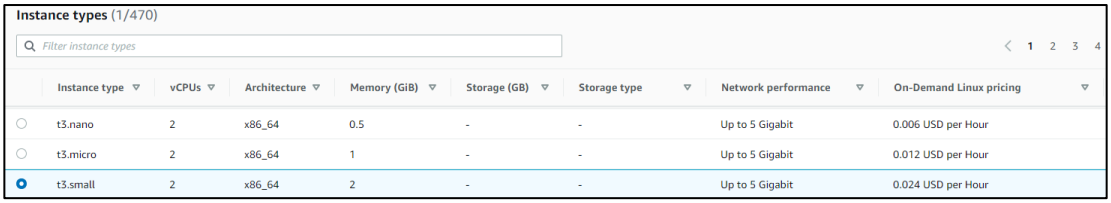


Рис. 3.20. Підбір підходящого обладнання

Далі створюємо нову пару SSH ключів для доступу на сам інстанс або використовуємо вже створену.

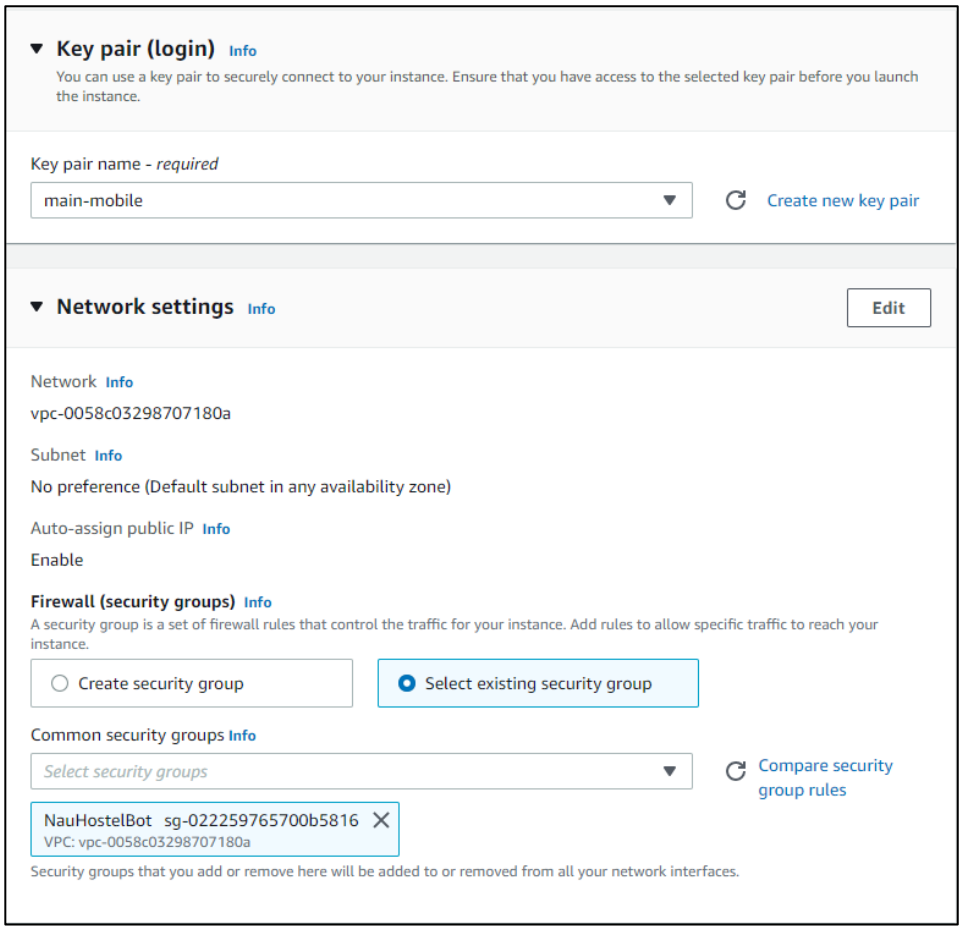


Рис. 3.21. Вибір способу доступу та налаштування мережі

В мережевих налаштуваннях вибираємо нашу приватну VPC та створену Security Group. При створенні інстансу буде присвоєно статичну приватну IP адресу, а при старті вже буде присвоєно публічну IP адресу, до наступного рестарту, тому є сенс зробити зовнішній DNS до Load Balancer і вже з нього по приватній адресі відкрити доступ до нашого хосту з додатком.

Приєднуємо створений Volume до цього інстансу. В додаткових налаштуваннях вибираємо опцію Instance auto-recovery – Default, за замовчуванням AWS, якщо станеться якась проблема з обладнанням і наш інстанс впаде, то він автоматично перезапуститься на новому обладнанні із збереженням усіх даних.

Всього за 15 секунд наш інстанс створився, запустився і став доступний по SSH.

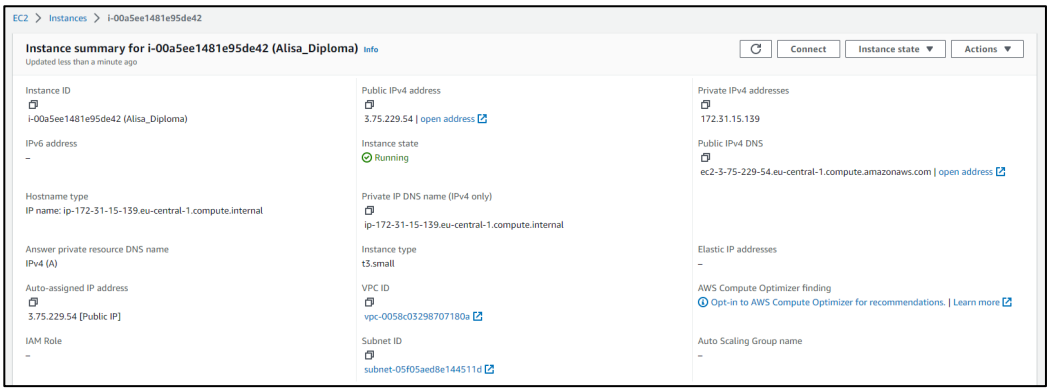


Рис. 3.22. Вікно детальної інформації створеного інстансу

3.5. Налаштування Linux інстансу та перенесення в Docker

Тепер у нас є працюючий і досяжний інстанс. Зв'язок для роботи з самим інстансом організовуємо через інтерактивний, зручний та багатофункціональний термінал MobaXterm.

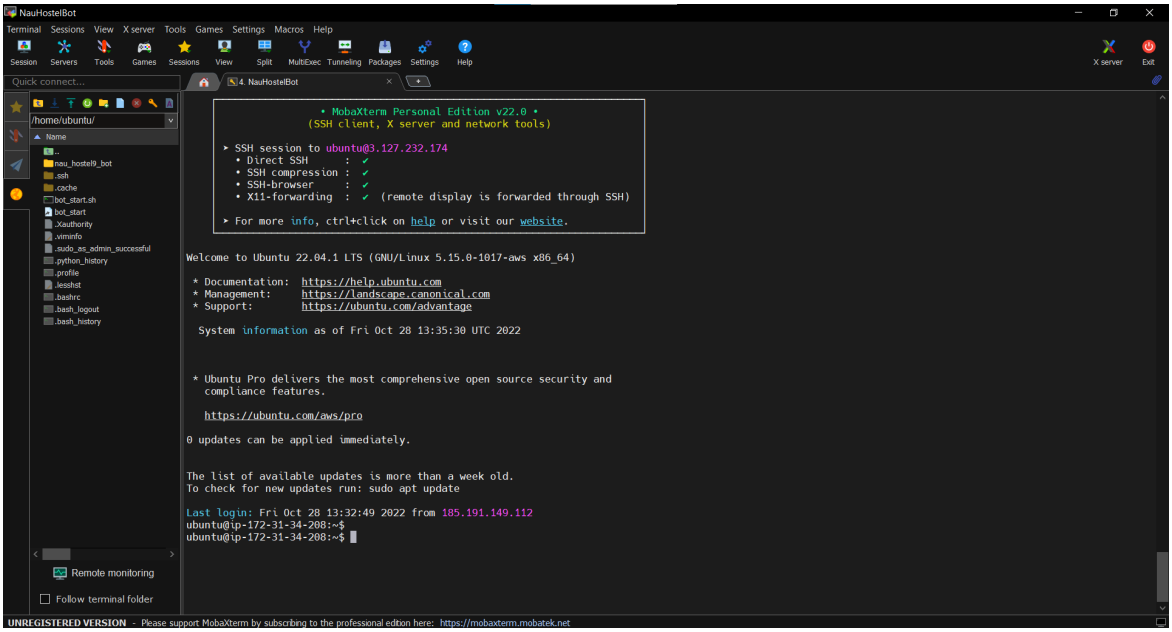


Рис. 3.23. Вікно терміналу MobaXterm підключеного до створеного інстансу

Як ми бачимо створення інстансу пройшло успішно, за допомогою створеної пари SSH ключів і стандартного логіну Ubuntu ми підключилися. Тепер ми можемо завантажити сюди наш додаток та запустити і все має працювати, але перед цим було б добре встановити всі останні оновлення для Linux серверу, встановити Python 3.9 зі всіма допоміжними пакетами та декілька сервісних програм для моніторингу, такі як atop, top та htop.

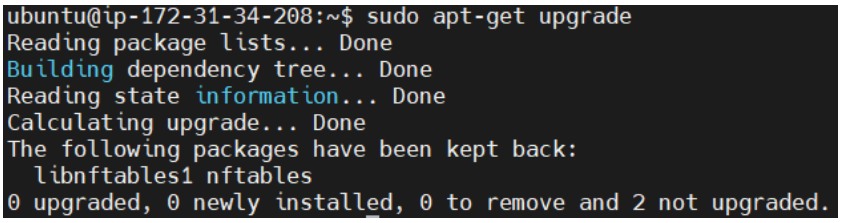


Рис. 3.24. Статус оновлення ОС

Можем побачити, що сервер оновлень не потребує і це можливо вирішить майбутні проблеми. Python 3.9 та atop встановлено.

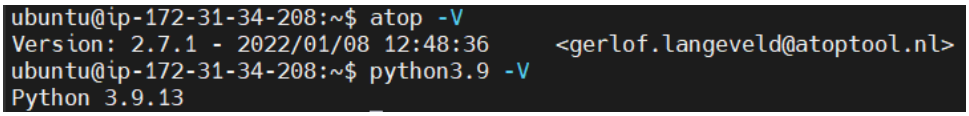


Рис. 3.25. Встановлення необхідних програм

Далі для перевірки самого додатку та його працездатності ми створили нове Python середовище, запустили бота і перевірили в телеграм чи все працює правильно і немає ніяких проблем із сервером чи інтернет з’єднанням.

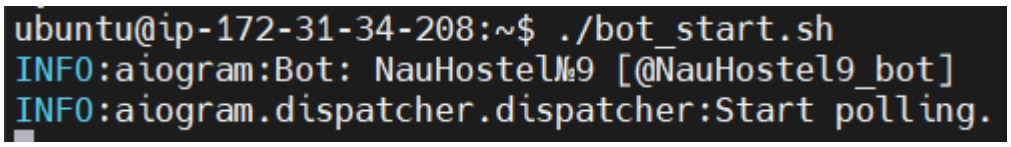


Рис. 3.26. Перевірка працездатності додатку

Зі скріншоту вище можемо зрозуміти, що додаток запрацював без помилок. Перевірка в самому Telegram дала позитивний результат.

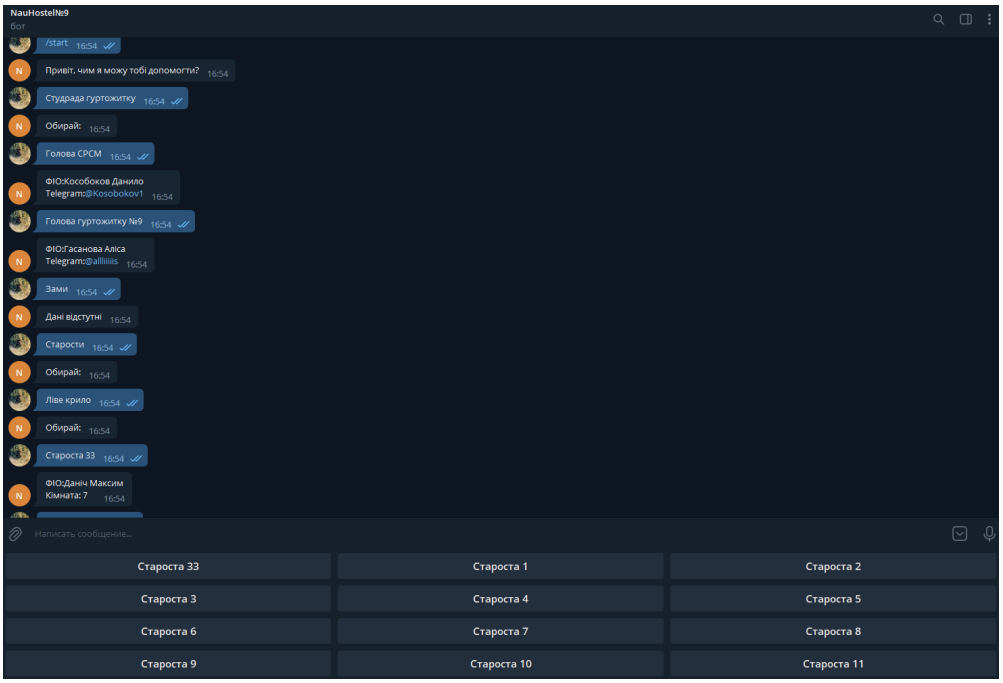


Рис. 3.27. Перевірка працездатності додатку в Telegram

Так як наша програма працює, сумісна з встановленим середовищем та характеристик створеного інстансу їй достатньо, тому тепер ми можемо перенести програму в Docker контейнер, для того, щоб в неї не було інших залежностей і нічого їй не заважало.

Отож встановлюємо сам Docker engine. Всі оновлення для ОС ми перевірили та встановили раніше. За допомогою команди:

```
[ec2-user@ip-172-31-44-181 ~]$ sudo yum install docker
```

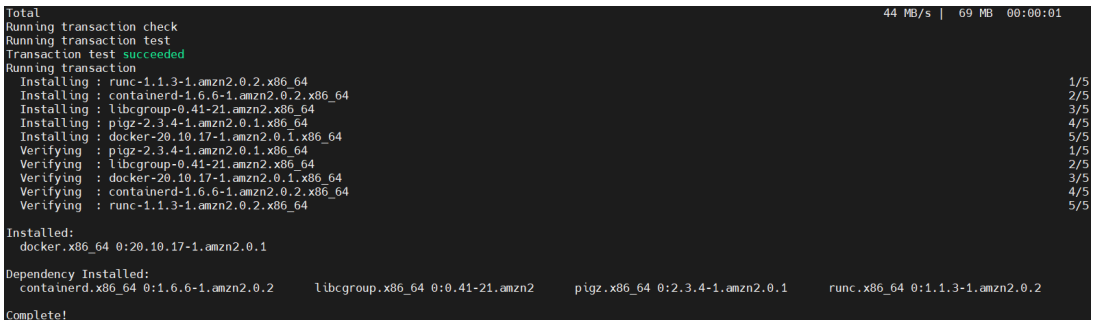


Рис. 3.28. Успішний процес встановлення Docker

[ec2-user@ip-172-31-44-181 ~]$ docker --version

Docker version 20.10.17, build 100c701

Простою командою в нас встановлено Docker Engine. Але потрібно дозволити нашому юзеру працювати і управляти Докером, для цього виконуємо наступні команди:

[ec2-user@ip-172-31-44-181 ~]$ sudo usermod -a -G docker ec2-user

[ec2-user@ip-172-31-44-181 ~]$ id ec2-user

uid=1000(ec2-user) gid=1000(ec2-user) groups=1000(ec2-user),4(adm),10(wheel),190(systemd-journal),992(docker)

[ec2-user@ip-172-31-44-181 ~]$ newgrp docker

Тепер у нас є права для використання докеру.

Додатково ми локально вже завантажили наш додаток на інстанс, але поки що наша служба не працює (вимкнута) і не запускається автоматично при завантажені інстансу. Вмикаємо автозапуск при старті системи і активуємо сервіс:

[ec2-user@ip-172-31-44-181 ~]$ sudo systemctl enable docker.service

Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.

[ec2-user@ip-172-31-44-181 ~]$ sudo systemctl start docker.service

[ec2-user@ip-172-31-44-181 ~]$ sudo systemctl status docker.service

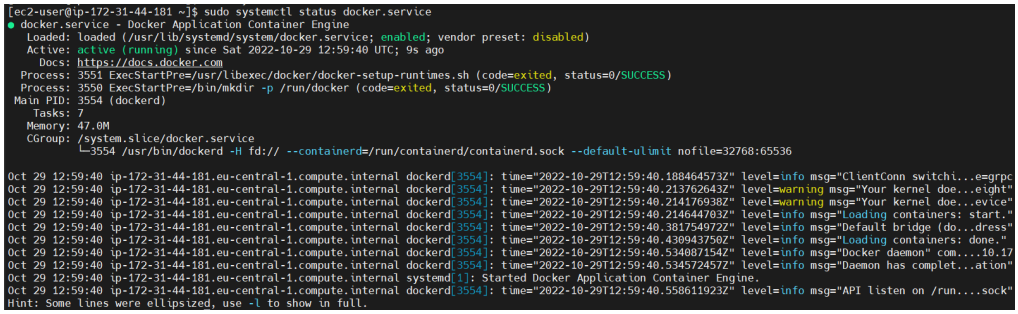


Рис. 3.29. Підтвердження встановлення і роботи Docker

Тепер потрібно створити і зібрати докер образ для майбутнього ізольованого контейнеру з нашим додатком.

Спочатку створюємо робочу папку і файл з інструкціями для докера - Dockerfile:

[ec2-user@ip-172-31-44-181 ~]$ mkdir dockerprog

[ec2-user@ip-172-31-44-181 ~]$ cd dockerprog/

[ec2-user@ip-172-31-44-181 dockerprog]$ vim Dockerfile

Тепер задаємо інструкції які потрібно буде виконати при створенні образу (пишемо в файлі Dockerfile):

```
# встановлення базового образа (host OS)
FROM python:3.9
# встановлення робочої папки в контейнері
WORKDIR /code
# створення віртуального середовища для Python додатку
ENV VIRTUAL_ENV "/venv"
RUN python -m venv $VIRTUAL_ENV
ENV PATH "$VIRTUAL_ENV/bin:$PATH"
# копіювання файлів програми і залежності в робочу папку
```

```
COPY nauhostel9/ ./
# встановлення залежностей
RUN pip install -r requirements.txt
# команда, яка виконується при запускі контейнера
CMD [ "python3.9", "./app.py" ]
```

Тепер створюємо сам образ з програмою:

```
[ec2-user@ip-172-31-38-145 dockerprog]$ docker build -t pythonprog .
Sending build context to Docker daemon   40.8MB
Step 1/8 : FROM python:3.9
 ---> ab0d2f900193
Step 2/8 : WORKDIR /code
 ---> Running in f0db93f390f1
Removing intermediate container f0db93f390f1
 ---> 9f0887a6925e
Step 3/8 : ENV VIRTUAL_ENV "/venv"
 ---> Running in 7d7c9e37db91
Removing intermediate container 7d7c9e37db91
 ---> d9b2dd6e8bc6
Step 4/8 : RUN python -m venv $VIRTUAL_ENV
 ---> Running in b01fb40d2f28
Removing intermediate container b01fb40d2f28
 ---> d32444f26a15
Step 5/8 : ENV PATH "$VIRTUAL_ENV/bin:$PATH"
 ---> Running in 8f2e086d07e8
Removing intermediate container 8f2e086d07e8
 ---> f76ec545ca6d
Step 6/8 : COPY nauhostel9/ ./
 ---> 4d2d5c82870f
Step 7/8 : RUN pip install -r requirements.txt
```

---> Running in 2af9eeced6db

Step 8/8 : CMD ["python3.9", "./app.py"]

---> Running in ed5ac2afc280

Removing intermediate container ed5ac2afc280

---> 7a70e3e89da9

Successfully built 7a70e3e89da9

Successfully tagged pythonprog:latest

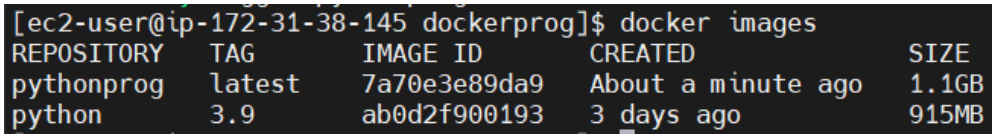


Рис. 3.30. Створений Docker образ

Тепер ми маємо готовий образ ізольованого середовища з додатком, залишилося створити сам працюючий контейнер і задати йому команду автозапуску при старті системи і у випадку, коли контейнер «впав» через помилку в роботі додатку.

[ec2-user@ip-172-31-38-145 dockerprog]$ docker create --name NauHostelBot --restart=always pythonprog

549545fb70a43d1b19dd94dc130d0aacbd6b29de8ea5294195a91001c73d5bf1

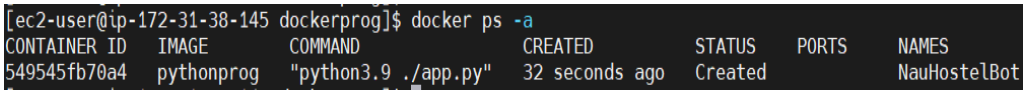


Рис. 3.31. Готовий контейнер з програмою

Контейнер створено, запускаємо і перевіряємо працездатність програми в контейнері. Перевіряємо функцію автостарту контейнера при перезавантаженні інстансу та у видку, якщо контейнер сам зламався і «впав».

```
[ec2-user@ip-172-31-38-145 dockerprog]$ docker start NauHostelBot
NauHostelBot
[ec2-user@ip-172-31-38-145 dockerprog]$ docker ps -a
CONTAINER ID   IMAGE        COMMAND               CREATED         STATUS          PORTS     NAMES
549545fb70a4   pythonprog   "python3.9 ./app.py"   3 minutes ago   Up 11 seconds             NauHostelBot
```

Рис. 3.32.Успішний запуск програми в контейнері

Ми бачимо, що контейнер сам моментально перезавантажився після того як ми примусово зламали його і він впав, така ж сама реакція буде у випадку рестарту інстансу.

```
[ec2-user@ip-172-31-38-145 dockerprog]$ docker ps -a
CONTAINER ID   IMAGE        COMMAND               CREATED         STATUS              PORTS     NAMES
549545fb70a4   pythonprog   "python3.9 ./app.py"   4 minutes ago   Up About a minute             NauHostelBot
[ec2-user@ip-172-31-38-145 dockerprog]$ sudo kill -9 6179
[ec2-user@ip-172-31-38-145 dockerprog]$ docker ps -a
CONTAINER ID   IMAGE        COMMAND               CREATED         STATUS         PORTS     NAMES
549545fb70a4   pythonprog   "python3.9 ./app.py"   4 minutes ago   Up 1 second              NauHostelBot
```

Рис. 3.33. Перевірка безвідмовної роботи програми в контейнері

3.6. Налаштування інструментів для аналізу та удосконалення

Кожна система або програма завжди розвивається чи змінюється, як і середовище потребує змін, для стабільної, безвідмовної праці, в залежності від навантаження і потреб людини, розробника чи бізнесу, а один із головних інструментів для аналізу і покращення – це лог файли.

Тому нам потрібно зробити так, щоб була можливість переглядати, зберігати та аналізувати лог файли програми, системи, моніторингових додатків.

Для цього ми використаємо утиліту Logrotate, AWS S3 та S3CMD.

Перш за все завантажимо, встановимо і настроємо доступ S3CMD до AWS S3 корзини.

Встановлюємо пакет для коректної роботи додатку, завантажуємо саму програму:

[root@ip-172-31-38-145 ~]# sudo yum install python-setuptools

[root@ip-172-31-38-145 ~]# cd /tmp/

[root@ip-172-31-38-145 tmp]# curl -LO https://github.com/s3tools/s3cmd/releases/download/v2.3.0/s3cmd-2.3.0.tar.gz

[root@ip-172-31-38-145 tmp]# tar xf s3cmd-*.tar.gz

[root@ip-172-31-38-145 tmp]# cd s3cmd-*

[root@ip-172-31-38-145 tmp]# sudo python setup.py install

[root@ip-172-31-38-145 tmp]# s3cmd --version

s3cmd version 2.3.0

Програму встановлено, тепер потрібно ввести свої ключі для програмного доступу AWS S3 CLI (Secret Key та інше.) та сконфігурувати місце завантаження.

sudo s3cmd --configure --config=/root/logrotate-s3cmd.config

Після вводу всіх даних ми отримуємо результат, який означає, що все пройшло успішно.

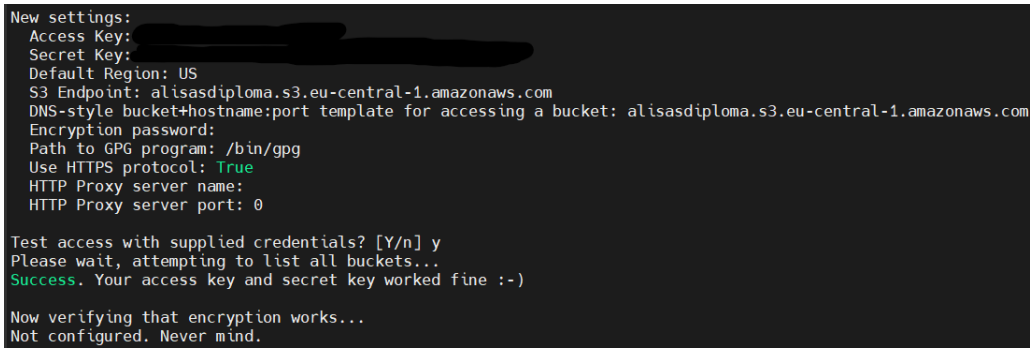


Рис. 3.34. Налаштування та перевірка S3CMD

Часто Logrotate вже встановлена і використовується як системна утиліта, тому нам потрібно тільки додати свою конфігурацію або виправити вже наявну.

Нам системні логи потрібно, тому ми почнемо з них. Йдемо в директорію /etc/logrotate.d/ та змінюємо файл syslog.

```
/var/log/cron
/var/log/maillog
/var/log/messages
/var/log/secure
/var/log/spooler
{
rotate 10
daily
compress
    missingok
    sharedscripts
    postrotate
        /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null || true
    endscript
dateext
dateformat -%Y-%m-%d-%s
lastaction
HOSTNAME=`hostname`
/bin/s3cmd sync --config=/root/logrotate-s3cmd.config /var/log/*.gz "s3://alisasdiploma/logs/$HOSTNAME/"
endscript
}
```

Рис. 3.35. Налаштування конфігурації ротації лог файлів

Кожного дня логи будуть архівуватись для економії дискового місця на інстансі та в S3 корзині, буде дописуватись час та дата архівування, далі завантаження в наш створений S3. Створена політика ротації логів між класами зберігання в S3 для економії коштів.

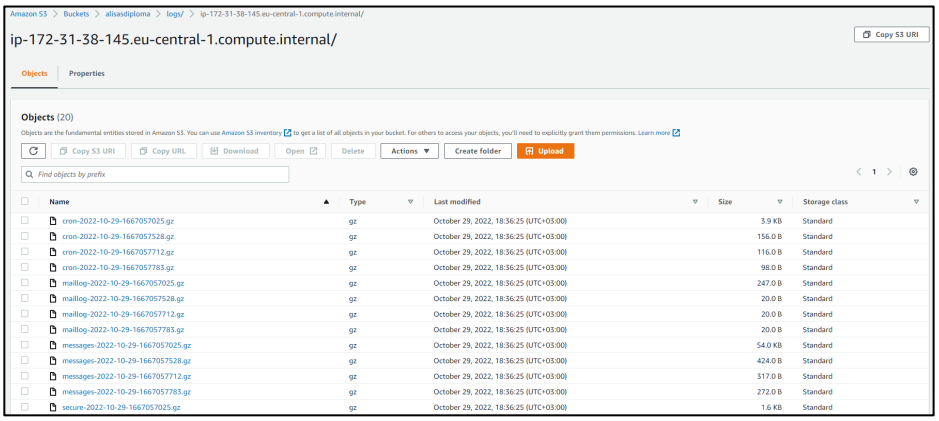


Рис. 3.36. Приклад ротованих лог файлів

Таку операцію можна зробити для будь-яких логів, як для системних, так і для логів власних додатків та програм.

3.7. Налаштування моніторингу для технічних показників інстансу та фінансів

Для того, щоб розуміти чи все працює правильно і стабільно, чи достатньо ресурсів для роботи додатку нам потрібно налаштувати моніторинг системних ресурсів, зберігати данні використання ресурсів та алертинг для оперативного усунення проблем або превентивного виконання дій для попередження можливих проблем. У цій справі нам допоможе AWS CloudWatch.

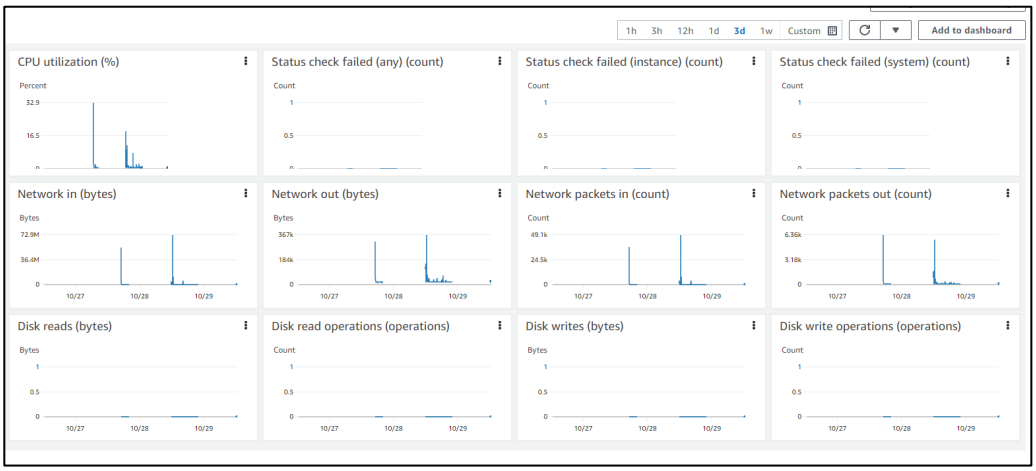


Рис. 3.37. Спрощене вікно моніторингу інстансу

В самій інформації кожного інстансу ми можемо бачити спрощене вікно від сервісу CloudWatch, де зібрані стандартні метрики для аналізу, можна вибрати різний історичний період, налаштувати свої метрики на головну сторінку та багато іншого, але весь спектр можливостей доступний в CloudWatch.

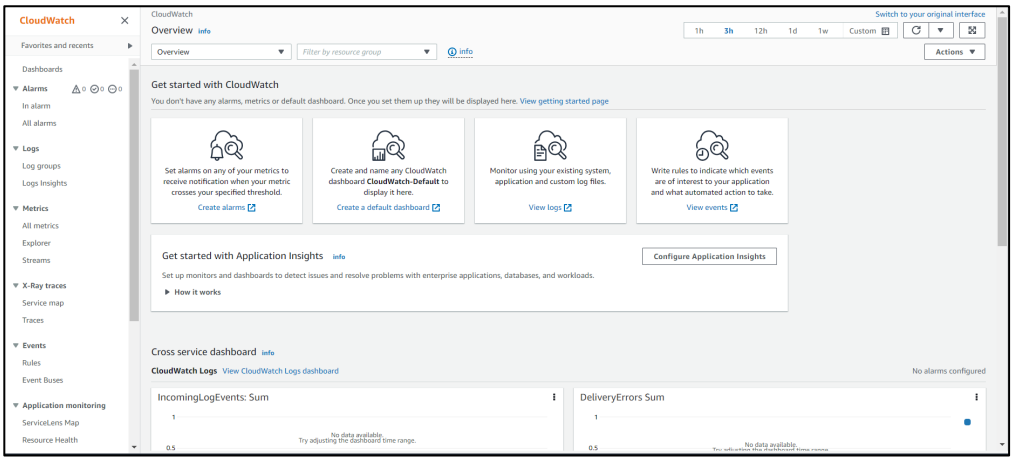


Рис. 3.38. Головна сторінка CloudWatch

Головна функція для нас, це моніторинг використаних ресурсів серверу, збереження цих даних та наступний аналіз. Для цього ми використаємо основні, такі як використання віртуальних процесорних ядер, вхідний та вихідний трафік, використання диску і статус чеки доступності серверу на рівні операційної системи та обладнання. На основі цих метрик створимо сповіщення з відповідними рівняти та нотифікацією по пошті, але можна налаштувати сповіщення повідомлення на телефон або в якийсь месенджер, як Телеграм.

За замовчуванням та безкоштовно метрики знімаються раз на 5 хвилин, можна увімкнути функцію «детального моніторингу» для окремих інстансів за додаткові кошти, в такому випадку метрики будуть зніматись щохвилинно.

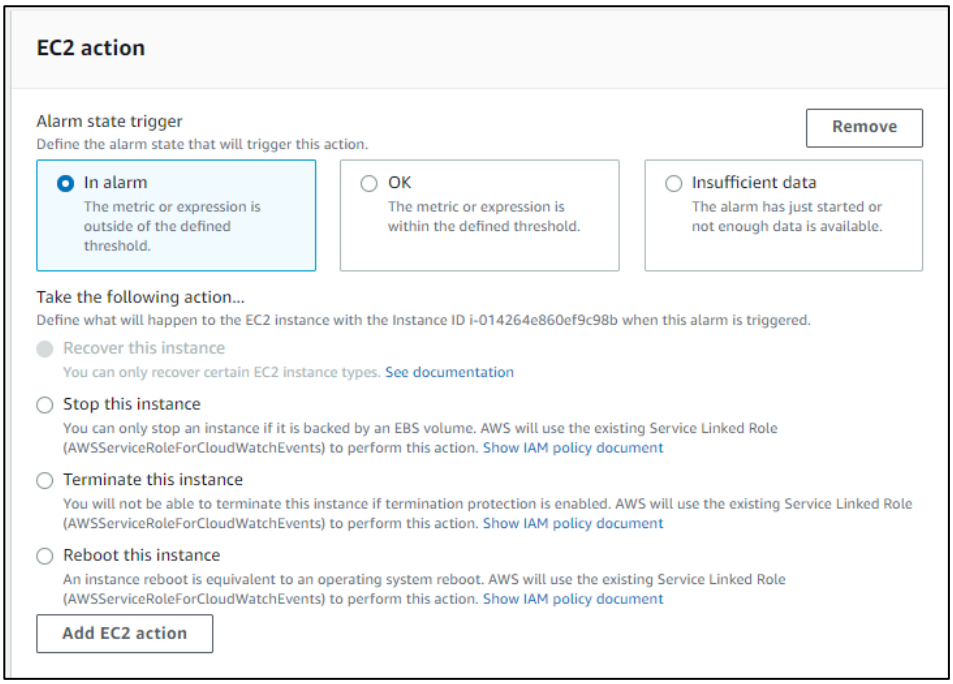


Рис. 3.39. Можливі автоматичні дії при появі проблеми

Налаштовані сповіщення паралельно можуть виконувати якісь дії при отримані даних про проблему, наприклад перезавантажити інстанс, зупинити його, це актуально якщо інстанс використовується для тестів і може бути вимкнутий будь-коли.

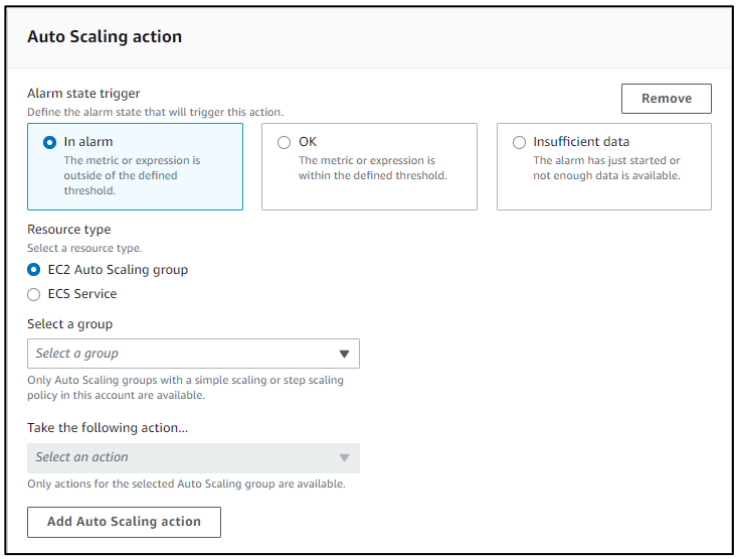


Рис. 3.40. Можливі автоматичні дії при появі проблеми

Або інстанс може бути автоматично розширений короткочасно або довгостроково і це все автоматизовано, а значить додає «безвідмовності».

Тут ми вже налаштувати оповіщення, яке відправить нам лист на пошту, якщо сумарне навантаження віртуальних ядер буде більше ніж 75% за будь-який період перевірки. Аналогічні оповіщення налаштовані на всі важливі метрики.

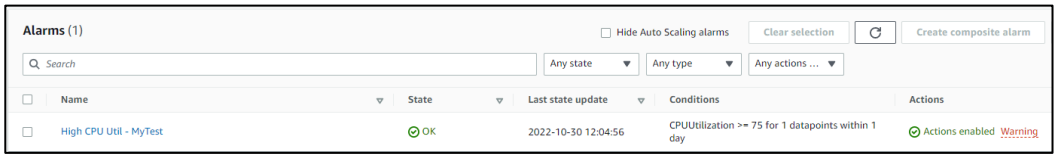


Рис. 3.41. Створене технічне оповіщення з правилом

Так як найчастіше це використовує бізнес і за все потрібно буде заплатити, тому важливо налаштувати такі ж фінансові сповіщення.

За попередніми розрахунками щомісячно нам потрібно буде платити до 10 американських доларів, тому ми і створили моніторинг та оповіщення, якщо наш місячний рахунок стане більше ніж 10 доларів.

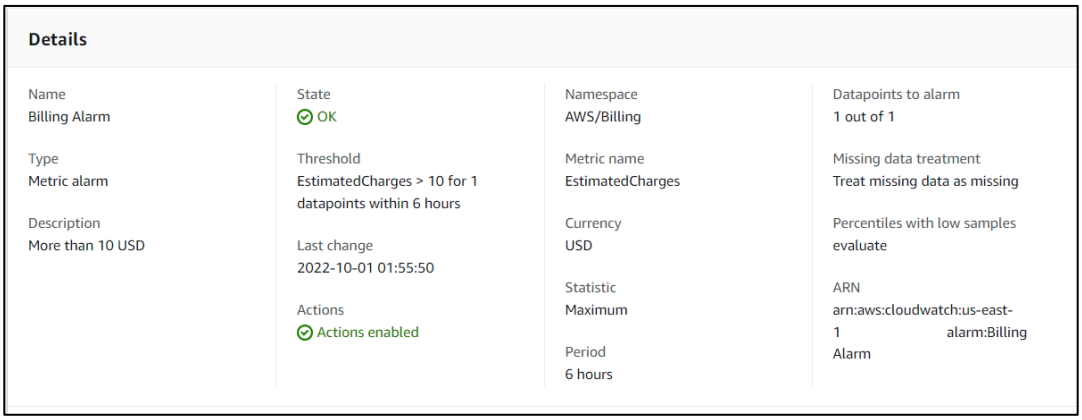


Рис. 3.42. Створене фінансове оповіщення з правилом

ВИСНОВОК ДО РОЗДІЛУ 3

У третьому розділі практично було зроблено умови максимально прийнятні та наближені до «безвідмовних» базуючись на технічних вимогах нашого додатку. Досягнуто це за рахунок багатьох доступних інструментів та факторів хмарної платформи AWS. Сервіс EC2 дозволяє нам швидко і просто створювати, змінювати, масштабувати потрібні нам об’єми ресурсів та дає безвідмовність на рівні обладнання, тому що обладнання для інстансів адмініструє та обслуговує сам AWS, а у випадку будь-яких проблем моментально відновлює працездатність із збереженням всіх даних за рахунок автоматичних безкоштовних бекапів, які не потрібно адмініструвати користувачу.

На рівні операційної системи безвідмовність досягається за рахунок деяких системних інструментів Linux та технології Docker Engine. Так як наша програма працює в ізольованому середовищі з виділеними для неї ресурсами і механізмом моментального відновлення процесу. Для інших програм можуть потребуватись інші умови або задачі і відповідно будуть використовуватись інші механізми для організації максимально безвідмовного середовища.

Створені інструменти збору, збереження та аналізу всіх даних операційної системи, програми та обладнання для подальшої оптимізації, розробки нових функцій та підвищення стабільності. Можуть бути введені нові технології на базі дослідження та аналізу роботи всіх складових середовища.

Комплексні можливості хмарної платформи AWS можуть задовільнити 90% потреб сучасної IT інфраструктури за прийнятними цінами.

ВИСНОВКИ

У першому розділі цього дипломного проєкту було розглянуто загальне поняття хмарних платформ, проаналізовано основні пропозиції на ринку. Після аналізу переваг та недоліків 3-х лідируючих хмарних провайдерів було вибрано використовувати AWS. Описані головні переваги перед конкурентами, виражена статистика реального використання AWS в світі.

У другому розділі були вибрані та детально описані сучасні, інноваційні технології та сервіси, які ми будемо використовувати на практиці. Висвітлені найважливіші сторони і причини використання деяких основних сервісів з платформи AWS. Базуючись на технічних вимогах досліджуваної програми були вибрані потрібні нам сервіси та технології і ці рішення обґрунтовані. Дали визначення «безвідмовної інфраструктури» - це інфраструктура яка максимально наближена до безвідмовного стану та має дуже маленький час самовідновлення.

У практичній частині детально було висвітлено вибір сервісів, технологій, додатків для задоволення вимог технічної бази нашого додатку. Було обґрунтовано та доведено поняття «безвідмовності», як стану максимально наближеного до безвідмовного. Було обґрунтовано вибір базової ОС – Linux, використання технології Docker та багато інших допоміжних програм і сервісів. Показано покроковий процес створення та налаштування інфраструктури на базі AWS, відповідно до вимог додатку. Висвітлена фінансова сторона використання AWS з прикладом.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ

1. Що таке хмарні технології і навіщо вони потрібні. [Електронний ресурс] – Режим доступу:https://edin.ua/shho-take-xmarni-texnologi%D1%97-i-navishho-voni-potribni/ (дата звернення 26.08.2022)

2. Amazon Web Services. [Електронний ресурс] – Режим доступу:https://itglobal.com/ru-ru/company/glossary/amazon-web-services/ (дата звернення 29.08.2022)

3. Типы облачных вычислений. [Електронний ресурс] – Режим доступу:https://aws.amazon.com/ru/types-of-cloud-computing/?nc1=h_ls (дата звернення 30.08.2022)

4. Введение в облачные вычисления. [Електронний ресурс] – Режим доступу::https://habr.com/ru/post/585064/ (дата звернення 04.09.2022)

5. Andreas Wittig, Michael Wittig, Amazon Web Services in Action. Kyiv, 2015. 115p.

6. Top AWS Stats You Should Know About in 2021. [Електронний ресурс] – Режим доступу:https://www.simplilearn.com/aws-stats-article (дата звернення 12.09.2022)

7. What is Telegram? [Електронний ресурс] – Режим доступу:https://www.businessinsider.com/guides/tech/what-is-telegram (дата звернення 20.09.2022)

8. What is a Telegram bot? [Електронний ресурс] – Режим доступу:https://www.opc-router.com/what-is-a-telegram-bot (дата звернення 22.09.2022)

9. Python. [Електронний ресурс] – Режим доступу:https://uk.wikipedia.org/wiki/Python (дата звернення 25.09.2022)

10. Linux operating system. [Електронний ресурс] – Режим доступу::https://www.techtarget.com/searchdatacenter/definition/Linux-operating-system (дата звернення 28.09.2022)

11. AWS EC2. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html (дата звернення 05.10.2022)

12. AWS EBS. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html (дата звернення 7.10.2022)

13. AWS ELB. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/what-is-load-balancing.html (дата звернення 10.10.2022)

14. AWS DynamoDB. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html (дата звернення 15.10.2022)

15. AWS CloudFront. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html (дата звернення 17.10.2022)

16. AWS CloudWatch. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch_architecture.html (дата звернення 20.10.2022)

17. Route 53. [Електронний ресурс] – Режим доступу:https://aws.amazon.com/route53/features/ (дата звернення 01.11.2022)

18. VPC. [Електронний ресурс] – Режим доступу:https://docs.aws.amazon.com/vpc/latest/userguide/how-it-works.html (дата звернення 03.11.2022)

19. Docker. [Електронний ресурс] – Режим доступу:https://uk.wikipedia.org/wiki/Docker (дата звернення 05.11.2022)

20. Docker infrastructure. [Електронний ресурс] – Режим доступу:https://docs.docker.com/get-started/overview/ (дата звернення 06.11.2022)