

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи магістра
на тему: «Проект системи зберігання, обробки та
аналізу медичних даних з використанням сервісів
Amazon Web Services та штучного інтелекту»

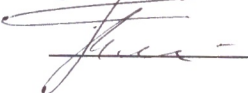
Виконав студент 2-го курсу групи КІм-22-1
спеціальності 123 «Комп'ютерна інженерія»

 Андрусенко О. М.

Керівник доцент кафедри КІ та КБ, к. т. н., доцент

 Бродський Ю. Б.

Рецензент д. т. н., старший науковий співробітник

 Ковбасюк С. В.

Житомир – 2023

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко

13 жовтня 2023 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу магістра

Студента Андрусенка Олега Михайловича

Тема роботи: «Проект системи зберігання, обробки та аналізу медичних даних з використанням сервісів Amazon Web Services та штучного інтелекту»

Затверджена Наказом університету від «13» жовтня 2023 р. № 571/с

Термін здачі студентом закінченої роботи 26 грудня 2023 р.

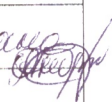
Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Воротніков В.В.	13.10.23 	13.10.23 
2	Воротніков В.В.	13.10.23 	13.10.23 
3	Воротніков В.В.	13.10.23 	13.10.23 

Керівник

(підпис)

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання. Формулювання технічного завдання. Опрацювання літературних джерел.	13.10.2023	Виконано 
2	Проектування структури системи.	14.10-30.10.2023	Виконано 
3	Підбір компонентів для реалізації системи.	01.11-20.11.2023	Виконано 
4	Налаштування системи.	21.11-07.12.2023	Виконано 
5	Тестування розробленої системи.	08.12-24.12.2023	Виконано 
6	Захист.	28.12.2023	

Студент  Андрусенко О.М.

Керівник  Бродський Ю. Б.

АНОТАЦІЯ

Текстова частина викладена на 93 сторінках друкованого тексту. Пояснювальна записка має 2 сторінки додатків. Список використаних джерел містить 30 найменувань та займає 3 сторінки. В роботі наведено 110 рисунків. Загальний обсяг роботи – 95 сторінок.

Мета кваліфікаційної роботи полягає у розробці проекту системи зберігання, обробки та аналізу медичних даних.

У кваліфікаційній роботі досліджено та налаштовано хмарні сервіси, які використовуються для функціонування системи зберігання, обробки та аналізу медичних даних. Розглянуто особливості функціонування таких сервісів та проаналізовано їх переваги та недоліки.

Проведено проектування структурних частин системи, наведено схеми їх роботи та взаємодії між собою. Створено алгоритми роботи системи.

Після розробки структури системи наведено процес налаштування та тестування системи зберігання, обробки та аналізу медичних даних.

Ключові слова: ХМАРНІ ТЕХНОЛОГІЇ, МЕДИЦИНА, ШТУЧНИЙ ІНТЕЛЕКТ, AMAZON WEB SERVICES.

					КІМ.КР.М – 123 – 02 – ПЗ			
		№ докум.	Підпис			Літ.	Арк.	Аркушів
Розроб.	Андрусенко О.М.		22.12.23	Проект системи зберігання, обробки				
Керівник	Бродський Ю. Б.		22.12.23	та аналізу медичних даних з викорис-			4	95
Реценз.	Ковбасюк С.В.		22.12.23	танням сервісів Amazon Web Services	ФІКТ, Гр. КІМ-22-1			
Н. контр.	Бродський Ю. Б.		22.12.23	та штучного інтелекту				
Затверд.	Сфіменко А.А.		22.12.23					

ANNOTATION

The text part is presented on 93 pages of printed text. The explanatory note has 2 pages of appendices. The list of references includes 30 items and occupies 3 pages. The paper contains 110 figures. The total volume of the work is 95 pages.

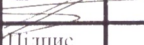
The purpose of the qualification work is to develop a project for a system for storing, processing and analyzing medical data.

The qualification work investigates and configures cloud services that are used to operate a system for storing, processing and analyzing medical data. The features of the functioning of such services are considered and their advantages and disadvantages are analyzed.

The design of the structural parts of the system is carried out, schemes of their work and interaction with each other are presented. The algorithms of the system are created.

After developing the system structure, the process of setting up and testing the system for storing, processing, and analyzing medical data is presented.

Keywords: CLOUD TECHNOLOGIES, MEDICINE, ARTIFICIAL INTELLIGENCE, AMAZON WEB SERVICES.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. В.				
		№ докум.	Підпис			5

ЗМІСТ

Перелік умовних скорочень	8
Вступ.....	9
Розділ 1. Теоретичні аспекти роботи систем зберігання, аналізу та обробки медичних даних з використанням хмарних сервісів та штучного інтелекту.....	10
1.1 Методи та принципи зберігання, обробки та аналізу медичних даних	10
1.2 Основні поняття та принципи використання хмарних технологій в медичних системах	15
1.3 Загальні принципи та поняття штучного інтелекту	20
Висновки з розділу 1	30
Розділ 2. Проектування системи зберігання, обробки та аналізу медичних даних.....	31
2.1 Розробка структури системи на базі хмарних технологій.....	31
2.2 Розробка архітектури підсистем	36
2.3 Алгоритм роботи системи. Взаємодія підсистем у системі	50
Висновки з розділу 2	51
Розділ 3. Реалізація системи зберігання, аналізу та обробки медичних даних з використанням хмарних сервісів та штучного інтелекту	52
3.1. Загальні налаштування хмарної мережі системи	52
3.2 Налаштування серверної групи.....	61
3.3. Налаштування сховища даних	68
3.4. Налаштування сервісів AWS для обробки та аналізу даних.....	73
3.5 Тестування системи.....	76
Висновки з розділу 3	88
Висновки	89
Список використаних джерел.....	90

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			7

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AWS – Amazon Web Services

VPC – Virtual Private Cloud

RDS – Relational Database System

IAM – Identity Access Management

S3 – Simple Storage Service

EC2 – Elastic Cloud Compute

ACL – Access Control List

IP – Internet Protocol

NAT – Network Address Translation

ALB – Application Load Balancer

ASG – Auto Scaling Group

ІІІ – штучний інтелект

МН – машинне навчання

		Андрусенко О.М.			<i>КІМ.КР.Б – 123 – 02 – ПЗ</i>	
		Бродський Ю. Б.				
		№ докум.	Підпис			8

ВСТУП

Актуальність теми. В сучасному світі вдосконалення систем зберігання, обробки та аналізу медичних даних є критично важливим завданням, що відіграє ключову роль у процесі надання медичних послуг та розвитку сфери охорони здоров'я. Нині, за допомогою хмарних сервісів та штучного інтелекту, ми отримуємо унікальну можливість революціонізувати та оптимізувати процеси в області медичної інформатики.

Тема роботи є надзвичайно актуальною в контексті поєднання технологій та охорони здоров'я. Зростання обсягів медичної інформації потребує ефективних систем зберігання та управління даними, а використання AWS (Amazon Web Services) сприяє розробці безпечних та конфіденційних рішень. Штучний інтелект, використовуючи дані, може покращити аналіз, сприяючи виявленню захворювань та розвитку нових методів дослідження. Гнучкість AWS дозволяє ефективно масштабувати систему, важливо у зростаючій області медичної інформатики.

Метою роботи є розробка проекту інфраструктури системи зберігання, обробки та аналізу медичних даних на основі хмарних технологій та засобів штучного інтелекту.

Для досягнення поставленої мети необхідно виконати наступні задачі:

- проаналізувати існуючі методи зберігання, обробки та аналізу медичних даних, основи хмарних технологій та штучного інтелекту;
- створити структуру системи, що базується на хмарних технологіях, описати алгоритм її роботи;
- розробити та налаштувати інфраструктуру системи зберігання, обробки та аналізу медичних даних.

Об'єктом дослідження є процес проектування та створення інфраструктури системи зберігання, обробки та аналізу медичних даних.

Предметом дослідження є використання хмарних технологій та засобів штучного інтелекту в медичних системах.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			9

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ РОБОТИ СИСТЕМ ЗБЕРІГАННЯ, АНАЛІЗУ ТА ОБРОБКИ МЕДИЧНИХ ДАНИХ З ВИКОРИСТАННЯМ ХМАРНИХ СЕРВІСІВ ТА ШТУЧНОГО ІНТЕЛЕКТУ

1.1 Методи та принципи зберігання, обробки та аналізу медичних даних

У сучасному світі системи зберігання, обробки та аналізу медичних даних стають необхідною основою для покращення якості надання медичних послуг, діагностики та наукових досліджень в галузі охорони здоров'я. Збільшення обсягу медичної інформації, швидкість змін в медичних технологіях та розширення можливостей аналізу вимагають ефективних систем, що забезпечують безпечно, швидко та доступне управління цими даними.

Сучасні системи зберігання медичних даних використовують передові технології для забезпечення надійності, конфіденційності та доступності цієї важливої інформації. Існує ряд методів і принципів, що визначають цей процес з максимальною деталізацією:

1. *Стандартизація даних:* використання уніфікованих стандартів для представлення та обробки медичної інформації, що включає в себе використання медичних кодів, термінології та форматів даних, що дозволяє легко обмінюватись та інтегрувати дані між різними системами.

2. *Електронна медична документація:* застосування електронних систем для запису та зберігання медичних даних, які дозволяють пришвидшити доступ до інформації, автоматизовують процеси документування та забезпечують можливість аналізу даних для поліпшення медичних практик.

3. *Системи безпеки та конфіденційності:* впровадження технологічних та організаційних заходів для забезпечення безпеки та конфіденційності медичних даних. Включає захист від несанкціонованого доступу, шифрування даних та визначення рівнів доступу для різних користувачів.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			10

4. *Резервне копіювання та відновлення даних:* впровадження систем для регулярного резервного копіювання медичних даних, що дозволяє відновлювати інформацію в разі втрати або пошкодження та сприяє уникненню втрати важливої медичної історії пацієнтів.

5. *Ідентифікація та аутентифікація:* застосування механізмів ідентифікації та аутентифікації для забезпечення того, що доступ до медичних даних мають лише авторизовані користувачі. Це включає в себе використання паролів, біометричних технологій та інших методів перевірки особи.

6. *Аудит та моніторинг доступу:* систематичний аналіз та запис подій щодо доступу до медичних даних з метою виявлення та реагування на ненормативну або підозрілу діяльність, що допомагає вчасно виявляти можливі порушення безпеки.

7. *Інтероперабельність:* забезпечення можливості взаємодії та обміну даними між різними системами та провайдерами медичних послуг, що сприяє наданню комплексної медичної допомоги та покращенню координації між медичними установами.

8. *Забезпечення якості даних:* розробка та застосування методів для контролю та покращення якості медичних даних, включаючи валідацію, стандартизацію та аудит.

Отже, системи зберігання медичних даних відіграють ключову роль у покращенні якості та доступності медичної інформації. Вони є необхідністю для сучасної медицини, де швидкий та ефективний доступ до даних може врятувати життя. Завдяки цим системам, зменшується ймовірність помилок у лікуванні через забезпечення повноцінного обміну інформацією між різними ланками медичного процесу. Зручний доступ до медичних записів також сприяє більш ефективному веденню медичної документації та допомагає у виявленні ранніх ознак захворювань.

Однак важливо враховувати аспекти конфіденційності та безпеки даних в контексті зберігання медичної інформації. Забезпечення конфіденційності

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			11

пацієнтів та відповідне управління доступом до даних є критичними аспектами, які потрібно узгоджувати при впровадженні та розвитку таких систем.

Аналіз медичних даних не лише полегшує роботу медичного персоналу, але й дозволяє виявляти нові зв'язки та закономірності у сфері охорони здоров'я. Він відкриває можливості для персоналізованої медицини та розвитку інноваційних методів лікування та включає в себе ряд методів та принципів, які допомагають враховувати та аналізувати різноманітні аспекти медичної інформації. Виділяють шість основних груп аналізу:

1. *Статистичний аналіз*: опис та узагальнення медичних даних. Це включає в себе обчислення середніх значень, медіан, дисперсій, та інших статистичних параметрів, які допомагають визначити центр та розкид даних.

2. *Епідеміологічний аналіз*: встановлення тенденцій поширення захворювань, їх вплив на популяцію, та ідентифікація факторів ризику. Дослідження епідеміології допомагає розуміти розподіл захворювань та сприяє розробці ефективних стратегій профілактики.

3. *Клінічний аналіз*: вивчення конкретних медичних показників та симптомів. Це включає аналіз результатів лабораторних досліджень, зображень, а також клінічних спостережень.

4. *Молекулярний аналіз*: вивчення генетичних та молекулярних аспектів захворювань. Дослідження генетичних мутацій та експресії генів дозволяє розкрити особливості хвороб та розробляти персоналізовані методи лікування.

5. *Системний аналіз*: дослідження взаємодії різних систем організму. Це може включати вивчення впливу лікування на різні органи та системи, а також взаємодію різних медичних втручань.

6. *Динамічний аналіз*: вивчення змін в часі. Це включає аналіз хронологічного розвитку захворювань, відслідковування ефективності лікування та прогнозування результатів.

Використання цих методів та принципів аналізу дозволяє отримувати глибше розуміння медичних даних, визначати закономірності, виявляти фактори

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			12

впливу та розробляти науково обґрунтовані підходи до лікування та профілактики захворювань.

Загалом, системи аналізу медичних даних є важливим інструментом, який сприяє покращенню якості медичної допомоги через збільшення доступності та аналізу великих обсягів інформації. Їх впровадження вимагає уважного балансу між технологічними можливостями та етичними аспектами для досягнення максимальної користі для пацієнтів та медичного персоналу.

Обробка медичних даних також є складним та відповідальним процесом, який вимагає дотримання визначених методів та принципів для забезпечення конфіденційності, цілісності та доступності інформації. Основні аспекти цього процесу включають:

- *збір медичних даних* повинен відбуватися тільки за інформованою згодою пацієнта або його законного представника. Медичні дані, які збираються, повинні бути обмежені обсягом, необхідним для досягнення конкретної мети обробки;

- обробка медичних даних повинна *базуватися на законних підставах*, таких як необхідність для виконання медичних послуг або дотримання юридичних обов'язків;

- пацієнти *повинні бути повідомлені* про мету, обсяг та терміни обробки їхніх медичних даних. Забезпечення безпеки медичної інформації передбачає використання заходів шифрування, контролю доступу та інших технічних та організаційних заходів;

- медичні дані повинні оброблятися *лише у тому обсязі, який необхідний для досягнення визначеної мети* та не повинні зберігатися довше, ніж це визначено законом чи правилами медичної етики;

- забезпечення точності та ідентифікації пацієнтів передбачає *використання систем аутентифікації*, щоб уникнути помилок та неправильної інтерпретації медичних даних;

- пацієнти мають право на *доступ до своїх медичних даних*, а також на виправлення та видалення невірної інформації;

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			13

- регулярний моніторинг та аудит діяльності з обробки медичних даних для виявлення порушень та вдосконалення систем безпеки.

Ці принципи та методи складають основу для створення та управління системами обробки медичних даних, які відзначаються не лише високою ефективністю, але й визначеними етичними стандартами. Вони виступають як керівні принципи, спрямовані на забезпечення надійності, прозорості та дбайливого ставлення до конфіденційності в медичних інформаційних системах.

Системи обробки медичних даних визначають новий рівень медичної практики, полегшуючи роботу фахівців, поліпшуючи якість надання медичної допомоги та сприяючи розвитку сучасної медичної науки. Однак важливо постійно розвивати та вдосконалювати ці системи, враховуючи виклики та можливості, що виникають в сучасному медичному середовищі. Загалом, вони відіграють критичну роль у сучасній медицині, сприяючи покращенню діагностики, лікування та управління здоров'ям пацієнтів. Їх внесок полягає в полегшенні доступу до інформації, ефективному її аналізі та подальшому використанні для прийняття інформованих медичних рішень.

Зберігання медичних даних в електронному форматі дозволяє збільшити доступність інформації для медичних фахівців, що пришвидшує реагування на невідкладні ситуації та поліпшення координації догляду. Обробка цих даних за допомогою сучасних алгоритмів та технологій дозволяє виявляти закономірності, тренди та ризикові фактори для розробки індивідуальних підходів до лікування, а аналіз медичних даних сприяє розумінню характеру захворювань, дозволяючи прогнозувати їхній розвиток та визначати оптимальні терапевтичні стратегії. Крім того, ці системи можуть виявляти аномалії та попереджати про можливі ускладнення, що робить їх невід'ємною частиною сучасної медичної практики.

Забезпечення важливого балансу між якісною медичною допомогою та захистом приватності пацієнтів вимагає уважного керування даними та застосування механізмів контролю доступу. Використання сучасних методів шиф-

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			14

рування та аутентифікації гарантує високий рівень захисту від несанкціонованого доступу, забезпечуючи при цьому безперебійний обмін інформацією для ефективного лікування та діагностики.

Не менш важливим є етичний аспект, що передбачає забезпечення пацієнтам контролю над їх медичними даними, включаючи право на доступ, виправлення та видалення інформації. Це відображає загальний підхід до поваги до прав пацієнтів та конфіденційності, які є невід'ємними принципами медичної етики. Інновації та захист приватності взаємодіють для досягнення найвищих стандартів медичного обслуговування, формуючи інтегрований підхід, що сприяє якісному та етичному медичному обслуговуванню пацієнтів.

1.2 Основні поняття та принципи використання хмарних технологій в медичних системах

Хмарні технології представляють інноваційний підхід до забезпечення доступу, обробки та обміну інформацією через Інтернет. Ця концепція базується на ідеї використання віртуальних ресурсів, які доступні за допомогою глобальної мережі, для забезпечення різноманітних послуг та функцій. Хмарні технології визначають новий спосіб взаємодії з інформацією та використання ресурсів, що дозволяє користувачам ефективно працювати з даними та застосунками, не обмежуючи себе географічно чи наявним обладнанням.

Основними аспектами хмарних технологій є гнучкість та масштабованість, що дозволяє користувачам адаптувати обсяг та функціональні можливості в залежності від конкретних потреб. Це сприяє зручності роботи та оптимізації використання ресурсів. Також, хмарні технології дозволяють реалізацію спільного доступу до даних та співпрацю над проектами, сприяючи комунікації та обміну ідеями.

Новаторський характер хмарних технологій полягає в їхній здатності поєднувати ефективність та простоту використання, що створює умови для ро-

		Андрусенко О.М.			<i>КІМ.КР.Б – 123 – 02 – ПЗ</i>	
		Бродський Ю. Б.				
		№ докум.	Підпис			15

звітку нових моделей бізнесу, покращення робочих процесів та взаємодії, а також відкриває можливості для реалізації інновацій та покращення якості послуг у різних галузях.

В сучасному інформаційному світі хмарні технології є важливим елементом для забезпечення доступу до ресурсів та послуг через мережу Інтернет. Основні принципи хмарних технологій включають:

1. *Еластичність*: можливість миттєво масштабувати обчислювальні ресурси в залежності від потреб користувача, що дозволяє ефективно використовувати ресурси та зменшує витрати на обладнання.

2. *Доступність*: надання безперервного доступу до ресурсів. Користувач може отримати до них будь-де та будь-коли, що сприяє зручності та мобільності.

3. *Самообслуговування*: можливість користувачам самостійно налаштовувати та керувати обчислювальними ресурсами через зручний інтерфейс без необхідності значного втручання адміністратора.

4. *Об'єднання ресурсів*: ресурси можуть бути об'єднані для забезпечення оптимального використання, або ж розділені між різними користувачами або завданнями.

5. *Множинність послуг*: хмарні платформи пропонують різні види послуг, такі як IaaS (Інфраструктура як сервіс), PaaS (Платформа як сервіс) та SaaS (Програмне забезпечення як сервіс), що дозволяє користувачам вибирати та використовувати тільки ті ресурси, які їм необхідні.

6. *Безпека та конфіденційність даних*: забезпечення високого рівня безпеки та конфіденційності даних є критичним аспектом хмарних технологій. Вони використовують різноманітні заходи, такі як шифрування та автентифікація, для захисту інформації користувачів.

7. *Оплата за використання*: модель оплати "плати за використання" дозволяє користувачам платити лише за ті ресурси, які вони використовують, що робить вартість обчислювальних послуг більш прозорою, гнучкою та ефективною.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			16

Використання хмарних технологій у медичних системах стає все більш важливим та вагомим елементом сучасної медицини, вносячи істотні покращення в організацію, доступність та зберігання медичної інформації.

Використання хмарних технологій у медичних системах має як переваги, так і недоліки, які важливо враховувати при розгляді їхнього застосування.

До переваг використання хмарних технологій можна віднести:

- **зручний доступ до даних:** забезпечення легкого доступу до медичних даних з будь-якого місця, що полегшує роботу лікарів та інших медичних фахівців;
- **спільний доступ до інформації:** дозволяє різним медичним фахівцям ефективно співпрацювати та обмінюватися даними для покращення якості діагностики та лікування;
- **масштабованість:** хмарні системи легко масштабуються відповідно до потреб користувача, що робить їх гнучкими та здатними адаптуватися до зростання обсягу медичної інформації;
- **зменшення витрат на обладнання:** уникнення значних витрат на придбання та обслуговування великої кількості обладнання;
- **оновлення та безпека:** автоматичне оновлення та підтримка програмного забезпечення серверів, що забезпечує високий рівень безпеки та актуальності.

Проте, окрім переваг використання хмарних технологій у медичних системах можна виділити й наступні недоліки:

- **проблеми з приватністю та конфіденційністю:** зберігання медичних даних в хмарі може викликати побоювання з приводу приватності та безпеки, особливо внаслідок можливості несанкціонованого доступу;
- **залежність від мережі:** робота з хмарними технологіями передбачає постійний доступ до Інтернету. Відсутність з'єднання може призвести до обмеженого доступу до медичної інформації;

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			17

– **високі витрати на передачу даних:** обмін великим обсягом медичних даних через мережу може призвести до високих витрат на передачу даних, що може стати фактором обмеження для деяких медичних установ;

– **небезпека втрати доступу:** якщо хмарний провайдер має проблеми або припиняє надавати послуги, це може призвести до втрати доступу до важливої медичної інформації;

– **обмеження відповідності до регуляторних вимог:** медичні установи повинні дбати про те, щоб використання хмарних технологій відповідало всім необхідним нормам та вимогам у сфері конфіденційності та безпеки даних.

За останні кілька років хмарні платформи стали невід'ємною частиною сучасної інформаційної технології, революціонізуючи спосіб, яким ми зберігаємо, обробляємо та спільно використовуємо дані. Вони надають користувачам необмежені можливості доступу до обчислювальних ресурсів, забезпечуючи при цьому гнучкість, ефективність та безпеку.

Найбільші хмарні платформи, такі як Amazon Web Services (AWS), Microsoft Azure та Google Cloud Platform (GCP), визначають нові стандарти у світі обчислювальної техніки. За допомогою їх потужностей користувачі можуть легко масштабувати свої проекти, використовуючи лише ті ресурси, які їм потрібні, і ефективно взаємодіяти з великим обсягом даних.

Розглянемо платформу Amazon Web Services. AWS є провідним провайдером хмарних обчислень та послуг в ІТ-галузі та є частиною великої корпорації Amazon, що надає широкий спектр хмарних послуг, які охоплюють обчислення, зберігання даних, машинне навчання, інтернет речей, аналітику та багато іншого.

Однією з ключових характеристик є масштабованість. Платформа може легко масштабуватися від невеликих підприємств до великих корпорацій, надаючи їм можливість користуватися ресурсами відповідно до їхніх потреб.

AWS пропонує широкий спектр сервісів, таких як Amazon EC2 (Elastic Cloud Compute) для обчислень в хмарі, Amazon S3 (Simple Storage Service) для

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			18

зберігання даних, Amazon RDS (Relational Database Service) для управління реляційними базами даних та інші. Ця різноманітність послуг дозволяє користувачам вибирати та налаштовувати необхідні інструменти для вирішення конкретних завдань.

Безпека також відіграє важливу роль у сервісах AWS. Провайдер використовує різні методи шифрування та захисту для забезпечення конфіденційності та цілісності даних користувачів. Крім того, платформа дозволяє налаштовувати політики доступу, щоб контролювати, як саме дані обробляються та кому вони доступні.

Наступною ключовою особливістю AWS є його гнучкість. Користувачі можуть вибирати і комбінувати послуги відповідно до своїх потреб, створюючи унікальні конфігурації для своїх проектів.

Використання хмарних технологій в медичних системах є значущим кроком у розвитку сучасної медицини. Вони дозволяють ефективно збирати, обробляти, зберігати та обмінюватися великими обсягами медичної інформації, полегшуючи роботу лікарів та покращуючи якість надання медичних послуг.

Однією з ключових переваг є зручний та безпечний доступ до медичних даних з будь-якого місця, що сприяє оперативному прийняттю рішень та координації догляду. Спільний доступ до інформації підвищує комунікацію між медичними фахівцями та сприяє колективному підходу до лікування.

Однак разом із перевагами приходять і виклики, зокрема щодо приватності та безпеки медичних даних. Забезпечення високого рівня захисту від несанкціонованого доступу важливе для збереження довіри до хмарних медичних систем. Також важливо враховувати залежність від доступу до Інтернету та виклики, пов'язані з обміном великим обсягом даних.

У цілому, хмарні технології в медичних системах визначають новий стандарт для ефективного управління інформацією та надання високоякісної медичної допомоги. З їхньою допомогою медична галузь може реалізовувати інноваційні рішення, забезпечуючи при цьому етичні та безпечні стандарти дії, що сприяє подальшому розвитку сфери здоров'я.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			19

1.3 Загальні принципи та поняття штучного інтелекту

В загальному, штучний інтелект (ШІ) це галузь науки та технологій, яка зосереджується на розробці систем, здатних виконувати завдання, які зазвичай вимагають інтелекту людини. ШІ спрямований на створення комп'ютерних програм та систем, які можуть аналізувати дані, робити висновки, навчатися від досвіду та приймати рішення у складних ситуаціях.

У своєму еволюційному розвитку штучний інтелект пройшов довгий шлях від простих алгоритмів до глибоких нейронних мереж, що використовуються для розв'язання найскладніших завдань.

Основу ШІ становлять різноманітні підходи та методи, такі як машинне навчання, нейронні мережі, обробка природних мов, комп'ютерне зорове сприйняття, робототехніка та інші. Розглянемо основні поняття штучного інтелекту:

Машинне навчання (Machine Learning, МН) – це підгалузь штучного інтелекту, яка дозволяє комп'ютерним системам навчатися з даних без явного програмування. Машинне навчання використовує алгоритми, які аналізують дані та виявляють закономірності, щоб зробити передбачення або прийняти рішення.

МН є однією з найважливіших галузей штучного інтелекту. Його основна ідея полягає в тому, щоб системи могли самостійно навчатися з даних, виявляти закономірності та робити висновки. Це дозволить моделям адаптуватися до змінних умов та вдосконалювати свою продуктивність з часом. Принципи машинного навчання включають:

1. *Навчання на основі прикладів:* алгоритми машинного навчання використовують набір прикладів (даних), щоб навчити модель відповідним патернам та залежностям у даних.

2. *Функція втрат:* для оцінки якості прогнозів моделі використовується функція втрат, яка визначає відхилення передбачень моделі від дійсних значень.

3. *Параметри моделі:* алгоритми машинного навчання підбирають оптимальні параметри моделі, що мінімізують функцію втрат та забезпечують точність передбачень на нових даних.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			20

4. *Навчання на великих обсягах даних:* моделі машинного навчання стають більш точними при навчанні на більшому обсязі даних, оскільки вони здатні виявляти складні патерни та закономірності.

Нейронні мережі (Neural Networks) – це математичні моделі, інспіровані роботою нервової системи людини. Вони складаються з шарів нейронів, які обробляють та передають інформацію. Глибокі нейронні мережі, або глибокі відповідні нейронні мережі, використовують багато шарів для розпізнавання складних зразків у великих наборах даних.

Нейронні мережі використовують ідею штучного нейрону, щоб моделювати складні функції та відношення у даних. Принципи нейронних мереж включають:

1. *Нейрони та шари:* нейронні мережі складаються зі шарів нейронів, де кожен нейрон обчислює ваговану суму вхідних сигналів та застосовує функцію активації.

2. *Глибокі нейронні мережі:* дані мережі мають багато шарів, що дозволяє їм виявляти абстрактні та високорівневі патерни у даних.

3. *Зворотнє розповсюдження помилки:* процес навчання включає зворотній процес розповсюдження помилки, коли модель коригує ваги на основі відхилення передбачень від дійсних значень.

4. *Функції активації:* такі функції визначають, як нейрон буде реагувати на вхідний сигнал. Вони надають нейронам здатність виражати нелінійні залежності.

Обробка природних мов (Natural Language Processing, NLP) – це галузь штучного інтелекту, що займається розробкою систем, які можуть розуміти, інтерпретувати та взаємодіяти з людською мовою. Технології NLP використовуються для автоматичного перекладу мов, аналізу тексту, генерації тексту та багато іншого.

Обробка природних мов вивчає способи, якими комп'ютери можуть розуміти та взаємодіяти з людською мовою. Принципи обробки природних мов включають:

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			21

1. *Синтаксичний та семантичний аналіз*: моделі NLP аналізують структуру та сенс речень, що допомагає розуміти їх контекст.

2. *Машинний переклад*: алгоритми NLP дозволяють перекладати тексти з однієї мови на іншу, використовуючи розуміння контексту, а також граматики.

3. *Розпізнавання іменованих сутностей*: моделі визначають та класифікують іменовані сутності, такі як імена людей, місця та інші.

4. *Генерація тексту*: моделі можуть генерувати тексти, що мають логічний зміст, зв'язок та структуру.

Комп'ютерний зір(Computer Vision) – це галузь штучного інтелекту, яка дозволяє комп'ютерам "бачити" і розпізнавати візуальні об'єкти на зображеннях та відео. Завдяки алгоритмам комп'ютерного зору системи можуть аналізувати зображення, розпізнавати обличчя, об'єкти, рух та інше. Принципи комп'ютерного зору включають:

1. *Виявлення об'єктів*: алгоритми комп'ютерного зору визначають та виділяють об'єкти на зображеннях, включаючи розпізнавання облич, автомобілів, тварин тощо.

2. *Семантична сегментація*: цей процес розподіляє кожний піксель зображення до конкретного класу, що дозволяє розрізняти об'єкти та їх частини.

3. *Детекція руху*: алгоритми детектують та відстежують рух об'єктів на відеозаписах.

4. *3D реконструкція*: використовуючи зображення з різних кутів, можливо створити тривимірну модель об'єктів.

Штучний інтелект заснований на ряді ключових принципів, що дозволяють системам здійснювати аналіз даних, виконувати розумові завдання та навіть імітувати людський інтелект. Давайте розглянемо детальніше кожен з цих принципів.

1. *Навчання на основі даних*: один з найважливіших принципів штучного інтелекту - це здатність систем навчатися на основі даних. Замість традиційного програмування, де розробники вказують конкретні інструкції для систем, системи ШІ навчаються на прикладах. Алгоритми машинного навчання, такі як

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			22

нейронні мережі, допомагають системам визначати закономірності та патерни у великих обсягах даних. Також даний принцип дозволяє системам автоматично визначати та коригувати свої моделі, враховуючи реальний досвід та зміни в даних, що дозволяє системам пристосовуватися до нових ситуацій, навчатися на помилках та покращувати свою продуктивність з часом.

2. Автоматизація і оптимізація: штучний інтелект революціонізує сфери, які вимагають автоматизації та оптимізації процесів. Системи ШІ можуть виконувати повторювані та рутинні завдання на порядки швидше та точніше, ніж це робили б люди. Наприклад, в медицині ШІ може аналізувати медичні зображення для виявлення хвороб або допомагати при виборі оптимального лікування. Цей принцип дозволяє економити час та ресурси, покращувати ефективність та якість роботи, а також знижувати витрати.

3. Апроксимація функцій: принцип апроксимації функцій є важливим у галузі нейронних мереж та глибокого навчання. Нейронні мережі можуть використовуватися для апроксимації складних математичних функцій, які можуть включати нелінійні та складні залежності між даними. Також даний принцип дозволяє системам виявляти та розпізнавати найскладніші патерни та зв'язки у даних, які важко або навіть неможливо зробити за допомогою традиційних методів.

4. Адаптація до нових даних: штучний інтелект уміє адаптуватися до нових даних та ситуацій, що відіграє важливу роль у динамічних середовищах. Системи можуть навчатися на нових даних, а також враховувати зміни в даних та внесення коригувань у свої моделі. Даний принцип дозволяє системам залишатися актуальними та ефективними у змінних умовах, а також пристосовуватися до непередбачуваних ситуацій.

5. Імітація інтелекту людини: багато підходів у штучному інтелекті спрямовані на імітацію роботи людського мозку та інтелекту. Нейронні мережі, зокрема, намагаються моделювати роботу нейронів у мозку, щоб відтворити процеси сприйняття та прийняття рішень. Цей принцип дозволяє системам розуміти

та адаптуватися до людських потреб, створюючи зручні та ефективні взаємодії між системами та користувачами.

6. *Здатність до розуміння контексту*: розуміння семантики речень, врахування історії взаємодій або зважування різних факторів при прийнятті рішень. Такий принцип дозволяє системам виявляти більш глибокі та точні зв'язки між даними, покращуючи якість розуміння та прийняття рішень.

7. *Етика та безпека*: розвиток штучного інтелекту покладає велику відповідальність на розробників та користувачів в плані етики та безпеки. Питання конфіденційності даних, можливого впливу на суспільство, ризику зловживання та безпеки систем стають дедалі важливішими. Цей принцип вимагає ретельного вивчення та розробки стандартів, норм та політик, які б забезпечили етичне та безпечне застосування ШІ у різних галузях та ситуаціях.

Кожен із розглянутих вище принципів відіграє важливу роль у розвитку та застосуванні штучного інтелекту. Вони допомагають системам адаптуватися до реальних умов, здійснювати аналіз даних, виконувати розумові завдання та взаємодіяти з оточенням. Також дуже важливо збалансувати їх застосування з етичними та соціальними аспектами для досягнення максимально позитивного впливу ШІ на суспільство.

Загалом, штучний інтелект – це революційна галузь, яка перетворює спосіб, яким ми створюємо, обробляємо та використовуємо інформацію. Перехід від традиційного програмування до систем, здатних навчатися та адаптуватися, змінює спосіб, яким ми сприймаємо світ. Принципи ШІ відображають цю еволюцію - від автоматизації до апроксимації, від навчання до адаптації. Однак, зростаючий вплив ШІ також покладає великі виклики, такі як етика та безпека.

Штучний інтелект продовжує шокувати світ і відкривати нові горизонти можливостей для медицини, науки, бізнесу та багатьох інших сфер життя та відіграє ключову роль у вдосконаленні медичної діагностики, лікування та досліджень. Його можливості в аналізі великих обсягів даних та виявленні складних патернів роблять ШІ важливим інструментом для покращення якості медичної практики та розвитку нових методів лікування. Розглянемо детальніше, як

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			24

саме штучний інтелект використовується в медичних дослідженнях та діагностиці.

1. Медичне зображення та діагностика: штучний інтелект відіграє важливу роль у аналізі медичних зображень, таких як рентгенограми, КТ-сканування, МРТ та ультразвукові зображення. Використовуючи нейронні мережі та інші методи машинного навчання, системи можуть автоматично виявляти аномалії, розпізнавати патології та допомагати лікарям при діагностиці.

Приклади застосування:

- Виявлення раку на зображеннях мамографії та позитронно-емісійної томографії (ПЕТ-сканування).
- Виявлення захворювань серця та судин на ехокардіограмах та КТ-зображеннях.
- Визначення ризику захворювань на основі зображень ретини.

2. Прогнозування захворювань: системи штучного інтелекту можуть аналізувати велику кількість даних про пацієнтів та про їх стан, що допомагає прогнозувати ризики та визначати індивідуальні підходи до лікування.

Приклади застосування:

- Прогнозування ризику серцевих захворювань на основі медичних даних та статистики.
- Визначення ризику розвитку діабету на основі генетичних даних та інших факторів.

3. Геноміка та молекулярна діагностика: штучний інтелект допомагає аналізувати геномні дані та інші молекулярні дані для визначення генетичних причин захворювань та розробки індивідуальних підходів до лікування.

Приклади застосування:

- Визначення генетичних мутацій, які можуть призвести до розвитку раку або інших захворювань.
- Розробка персоналізованих методів лікування на основі генетичних характеристик пацієнта.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			25

4. *Аналіз медичних текстів:* штучний інтелект може аналізувати великі обсяги медичних текстів, включаючи медичні записи, статті та дослідження. Це допомагає лікарям швидше знаходити релевантну інформацію та оновлювати свої знання.

Приклади застосування:

- Автоматичний аналіз медичних записів для виявлення патологій та тенденцій.
- Виділення ключової інформації з наукових статей для допомоги при прийнятті рішень.

5. *Розробка нових лікарських препаратів:* штучний інтелект може прискорити процес розробки нових лікарських препаратів шляхом аналізу великих обсягів даних про хімічні сполуки та їх взаємодії.

Приклади застосування:

- Відбір потенційних лікарських сполук для подальших досліджень.
- Прогнозування ефективності та побічних ефектів нових лікарських засобів.

6. *Персоналізована медицина:* завдяки аналізу великих обсягів генетичних, клінічних та молекулярних даних, штучний інтелект допомагає розробляти індивідуальні підходи до лікування. Персоналізована медицина передбачає врахування генетичних та інших особистих характеристик пацієнта при виборі методів лікування, що забезпечує більшу ефективність та знижує ризик побічних ефектів.

7. *Швидка діагностика та виявлення патологій:* штучний інтелект дозволяє автоматично аналізувати великі обсяги медичних даних, що значно сприяє виявленню патологій та хвороб на ранніх стадіях, що допомагає лікарям швидше вживати відповідних заходів та розробляти відповідні ефективні плани лікування.

8. *Робототехніка та асистенти для лікарів:* штучний інтелект може бути використаний для розробки медичних роботів та асистентів, які допомагатимуть

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			26

лікарям у виконанні рутинних завдань, операціях чи навіть надаватимуть пацієнтам допомогу, що зможе збільшити точність та ефективність процедур.

9. Медичні дослідження та розробка нових методів лікування: застосування штучного інтелекту у медичних дослідженнях допомагає аналізувати складні взаємозв'язки між різними медичними та фізичними факторами та розробляти нові методи лікування. Це може сприяти відкриттю нових медичних підходів та вдосконалених лікарських засобів.

10. Статистична аналітика та передбачення пандемій: штучний інтелект може аналізувати великі обсяги даних про захворювання, минулі пандемії та інші фактори здоров'я. Це дозволяє створювати моделі для передбачення поширення захворювань та вжиття відповідних заходів для їх запобігання.

Застосування штучного інтелекту в медичних дослідженнях та діагностиці може вже сьогодні покращити точність діагнозів, збільшити ефективність лікування та сприяти розвитку нових методів медичної практики. Однак це також вимагає ретельного врахування етичних, правових та конфіденційних аспектів для забезпечення безпеки та довіри пацієнтів, а також розкриває великі можливості для покращення охорони здоров'я та медичної практики.

ШІ має потенціал внести значний внесок у сферу медицини, принісши переваги та відкривши нові можливості в аналізі медичних даних та діагностики захворювань. Однак, разом із перевагами, використання штучного інтелекту також супроводжується викликами та обмеженнями. Розглянемо детально ці аспекти.

Серед переваг використання штучного інтелекту в медицині можна виділити наступні:

1. Точність діагностики та прогнозування: здатність виявляти субтельні патології та аномалії на медичних зображеннях. Це дозволяє швидше встановлювати більш точні діагнози. ШІ також може аналізувати великі обсяги даних для прогнозування ризиків та результатів лікування, що сприяє розробці індивідуальних планів лікування.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			27

2. *Персоналізована медицина*: здатність враховувати генетичні, клінічні та інші індивідуальні характеристики пацієнтів для розробки персоналізованих методів лікування, що може покращити ефективність та результати терапії.

3. *Автоматизація рутинних завдань*: виконання рутинних та часом монотонних завдань, такі як аналіз медичних зображень, обробка текстових даних та моніторинг пацієнтів. Це допомагає звільнити лікарів від рутинних обов'язків та сконцентрувати їх увагу на більш складніших завданнях.

4. *Збільшення швидкості діагностики*: аналіз великих обсягів даних в реальному часі, що дозволяє швидко виявляти патології та здійснювати діагностику навіть на ранніх стадіях захворювань.

5. *Вдосконалення обчислювальних можливостей*: швидша обробка великих обсягів медичних даних та виконання складних аналізів, що раніше може було б неможливо або дуже витратно за часом.

Також використання штучного інтелекту в медицині має свої виклики та обмеження. Серед них можна виділити наступні:

1. *Недостатність даних*: для ефективного функціонування штучного інтелекту потрібні великі обсяги якісних та репрезентативних медичних даних. Однак, іноді може бути важко зібрати відповідні дані, особливо у випадку рідкісних захворювань.

2. *Якість даних*: недостовірні або неправильно оброблені дані також можуть призвести до невірних аналітичних результатів та негативних наслідків у практиці.

3. *Етичні та конфіденційні питання*: збір, зберігання та обробка медичних даних за допомогою штучного інтелекту ставить питання про приватність та конфіденційність пацієнтів.

4. *Невідомість*: важко передбачити поведінку систем ШІ в нових, незвичайних ситуаціях.

5. *Витрати та доступність*: впровадження технологій штучного інтелекту може бути дорогим завданням, яке потребує великих витрат на обладнання, навчання фахівців та підтримку систем, що для деяких медичних закладів це може бути важкодоступною можливістю.

Отже, застосування штучного інтелекту в медицині відкриває перед сучасною медичною наукою та практикою широкий спектр можливостей. Новаторські технології машинного та глибокого навчання та обробки даних роблять можливими революційні зміни в процесі діагностики, лікування та досліджень. Однак, як у будь-якому іншому секторі, існують як переваги, так і виклики.

Завдяки застосуванню штучного інтелекту точність діагностики зростає завдяки аналізу медичних зображень та геномних даних. Персоналізована медицина стає реальністю, забезпечуючи індивідуальний підхід до кожного пацієнта. Автоматизація допомагає відсіювати рутинні завдання та звільняє фахівців для складніших завдань. Швидкість діагностики збільшується завдяки можливості аналізувати великі обсяги даних в реальному часі. Обчислювальні можливості дозволяють оптимізувати обробку масивних обсягів даних та сприяють швидкості аналізу.

Проте, використання штучного інтелекту супроводжується викликами. Недостатність якісних даних може ускладнити використання ШІ. Недостовірні дані можуть призвести до невірних висновків. Питання приватності та конфіденційності стають особливо актуальними, оскільки збір, зберігання та обробка медичних даних потребує особливої уваги. Важливо вирішити етичні питання та забезпечити відповідальне використання ШІ у медицині.

Всецільне впровадження штучного інтелекту у медицину вимагає спільних зусиль медичної спільноти, дослідників, правовиків та розробників. Вирішення складних питань етики, безпеки даних та відповідальності допоможе максимізувати переваги штучного інтелекту та забезпечити довіру пацієнтів та медичної галузі загалом. Таким чином, використання штучного інтелекту може зробити медицину більш точною, персоналізованою та доступною для всіх.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			29

Висновки з розділу 1

У першому розділі було розглянуто наступні питання:

- методи та принципи зберігання, аналізу та обробки медичних даних;
- основи хмарних технологій, переваги та недоліки їх використання у медичних системах;
- аналіз хмарної платформи AWS (Amazon Web Services), розглянуто її переваги та недоліки;
- принципи та поняття штучного інтелекту, приклади його використання у медичній практиці.

Аналіз основних концепцій хмарних сервісів у контексті медичних систем, включаючи зручний доступ до даних, масштабованість та ефективність обчислень показав, що взаємодія хмарних сервісів та штучного інтелекту розкриває потенційні можливості для покращення доступу до медичної інформації та оптимізації процесів обробки даних. Розглянуті технології у зв'язку один з одним можуть створити потужну систему для зберігання, обробки, в тому числі й за допомогою штучного інтелекту, та аналізу медичних даних із забезпеченням високої точності, ефективності та продуктивності.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			30

РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ ЗБЕРІГАННЯ, ОБРОБКИ ТА АНАЛІЗУ МЕДИЧНИХ ДАНИХ

2.1 Розробка структури системи на базі хмарних технологій

При розробці структури системи на базі хмарних технологій важливо здійснити обґрунтований огляд відповідних хмарних сервісів. У рамках нашого дослідження ми звернемо увагу на ключові аспекти платформи Amazon Web Services, одного з провідних постачальників хмарних послуг, який пропонує широкий спектр сервісів для забезпечення інфраструктури, обчислень та аналізу даних.

Система буде заснована на базі віртуальної приватної хмарної мережі (VPC) для забезпечення зручного та безпечного розподілу ресурсів. В рамках цієї VPC буде створено підмережі, що дозволять поділити мережу системи на логічні сегменти, що допоможе в управлінні ізоляцією, безпекою та ефективністю ресурсів.

Віртуальна приватна хмарна мережа є ключовим елементом інфраструктури на платформі Amazon Web Services. Вона надає можливість створювати та керувати власними віртуальними мережами в хмарному середовищі та дозволяє ізолювати ресурси та контролювати їхню взаємодію, що забезпечує безпеку та гнучкість у конфігурації хмарної інфраструктури.

Перевагами використання VPC є:

1. Ізоляція ресурсів: можливість створення власної віртуальної мережі в хмарному середовищі, що дозволяє ізолювати ресурси та забезпечити конфіденційність даних.

2. Контроль мережевого трафіку: налаштування правил мережевого екрану та маршрутизації, що надає повний контроль над мережевим трафіком та забезпечує безпеку мережі.

3. Масштабованість: мережа легко масштабується, дозволяючи розширювати ресурси та змінювати конфігурації відповідно до потреб проекту.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			31

4. Інтеграція з іншими сервісами AWS: безліч сервісів та можливостей AWS можна інтегрувати з VPC для підтримки різноманітних сценаріїв використання.

Проте використання VPC має й деякі недоліки:

1. Складність налаштувань: для ефективного використання VPC потрібне глибоке розуміння мережеских аспектів, що може бути складним для користувачів без великого досвіду.

2. Можливі витрати на ресурси: неправильна конфігурація може призвести до зайвих витрат на ресурси, оскільки вартість використання VPC залежить від обсягу та типу ресурсів.

3. Потреба в плануванні мережевої архітектури: правильне планування та конфігурація мережі вимагають часу та уважності, оскільки неправильні рішення можуть вплинути на продуктивність та безпеку.

4. Можливий обмежений обсяг IP-адрес: в залежності від конфігурації, може виникнути обмеження на обсяг доступних IP-адрес в межах мережі, що важливо враховувати при проектуванні.

Для ефективного використання адресного простору віртуальної хмарної мережі та ізоляції ресурсів необхідно використовувати поділ на підмережі. Поділ на підмережі (subnetting) використовується для розділення IP-адресного простору мережі на менші частини, які можна використовувати окремо. Кожна підмережа представляє собою логічний сегмент мережі з власним діапазоном IP-адрес, який може бути налаштований незалежно від інших підмереж. Цей підхід дозволяє ефективно використовувати доступний IP-адресний простір та забезпечує можливість розподілення ресурсів на зони доступності. Зони доступності є фізично відокремленими центрами обробки даних, які забезпечують високу доступність та надійність інфраструктури.

Підмережі можуть мати різні налаштування безпеки, такі як списки керування доступом (ACL) та групи захисту мережі (Security Groups), що дозволить точно налаштовувати доступ до ресурсів в межах конкретних сегментів мережі.

Основними аспектами поділу VPC на підмережі є:

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			32

- **адресний простір:** використання підмножини діапазону IP-адрес, який необхідно встановити при створенні VPC;
- **діапазони підмереж:** поділ основного діапазону на підмережі з власним діапазоном адрес;
- **ізоляція та безпека:** логічна ізоляція різних частин інфраструктури в межах VPC, тобто пов'язування кожної підмережі з конкретною доступною зоною (Availability Zone);
- **розташування ресурсів:** різні ресурси можуть бути розташовані в різних підмережах в межах одного VPC;
- **маршрутизація:** кожна підмережа пов'язана з конкретною таблицею маршрутизації, що визначає, які ресурси доступні з даної підмережі та які підмережі можуть взаємодіяти з ними;

Проте при розгляді поділу VPC на підмережі важливо пам'ятати про безпеку, а також слід доцільно обирати розміри підмереж та враховувати потреби у масштабуванні і високій доступності. Розглянемо деякі загальні принципи безпеки при налаштуванні VPC:

1. Використання приватних підмереж: розміщення сервісів та ресурси, які не повинні бути напряму доступними з Інтернету, у приватних підмережах.

2. Використання публічних підмереж для зовнішнього доступу: розміщення ресурсів, які повинні бути доступні з Інтернету, у публічних підмережах.

3. Маршрутизація та контроль доступу: ефективне налаштування маршрутизації між підмережами та списків контролю доступу (ACL) для керування трафіком між різними ресурсами.

4. Використання Network Access Control Lists (NACLs) та Security Groups: використання мережних списків контролю доступу для налаштування контролю доступу на рівні підмережі, а груп безпеки - для контролю доступу на рівні елементів, оскільки вони працюють на рівні елементів та контролюють вхідний та вихідний трафік.

		Андрусенко О.М.			Кім.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			33

5. Шифрування даних: використання шифрування для захисту даних під час транзиту та зберігання.

6. Моніторинг та журналювання: увімкнення моніторингу та журналювання подій для вчасного виявлення та реагування на потенційні загрози безпеці.

7. Використання AWS Identity and Access Management (IAM): використання IAM для керування доступом до AWS-ресурсів, тобто надання користувачам лише тих привілеїв, які їм необхідні для виконання їх завдань.

8. Захист власних ключів: забезпечення безпечного зберігання та керування ключами шифрування.

9. Резервне копіювання та відновлення: забезпечення резервного копіювання важливих даних та створення механізмів відновлення, щоб відновити інфраструктуру у випадку втрати даних або інших подій.

Важливим елементом системи є Gateway Endpoint. В нашій системі цей елемент буде засобом зв'язку між публічною та приватною підмережами системи. В загальному, він є частиною архітектури системи, яка дозволяє різним частинам системи чи мережі спілкуватися між собою, долаючи потенційні різниці в протоколах чи інші обмеження.

Загальна схема (модель складу) та структурна схема (модель структури) системи зберігання, обробки та аналізу медичних даних зображені відповідно на рисунках 2.1 та 2.2.

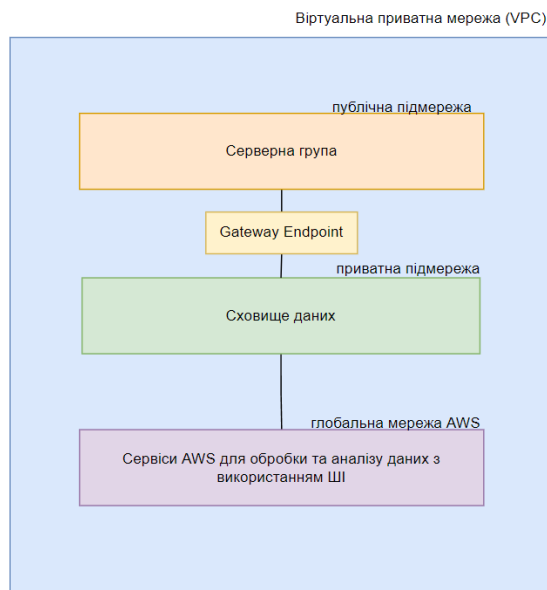


Рисунок 2.1 – Загальна схема системи зберігання, обробки та аналізу медичних даних

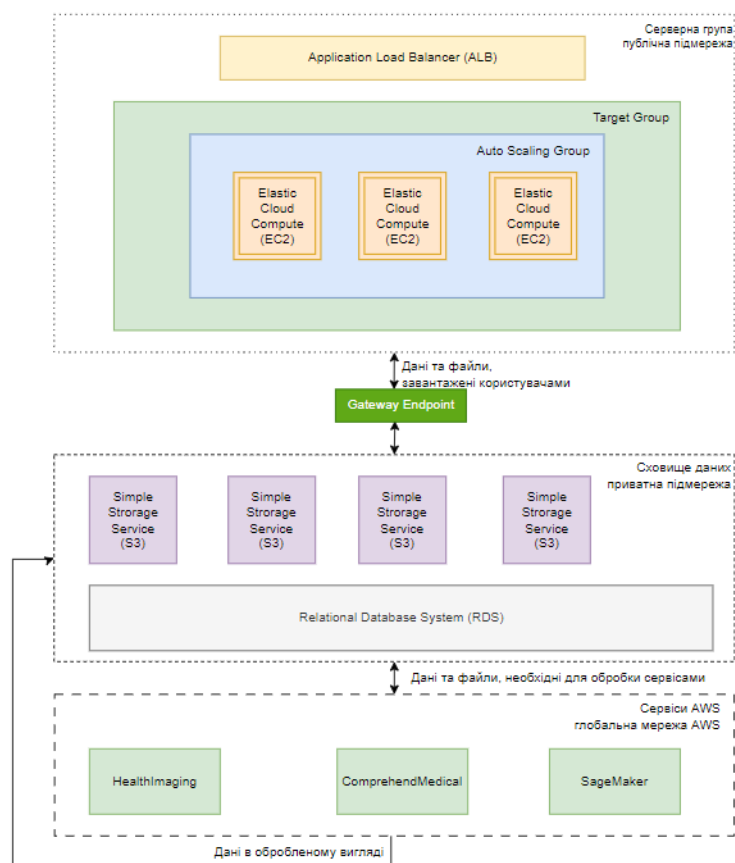


Рисунок 2.2 – Структурна схема системи зберігання, обробки та аналізу медичних даних

Дані потрапляють в систему через серверну групу, яка знаходиться в публічній підмережі хмарної мережі (VPC) і є загальнодоступною з мережі Інтернет. Далі через балансувальник навантаження (ALB) дані рівномірно розподіляються між серверами обробки (EC2), які знаходяться в групі призначення (Target Group), яка в свою чергу знаходиться в автоматично змінюваній групі (Auto Scaling Group), що дозволяє розширювати обчислювальну потужність даної групи в залежності від навантаження. Від групи серверів дані через Gateway Endpoint потрапляють до сховища даних. У сховищі даних медичні дані пацієнтів потрапляють до реляційної бази даних (RDS), а інші дані – у відповідні сховища Simple Storage Service (S3).

Зі сховища передаються до відповідних сервісів AWS для обробки та аналізу. Оскільки дані сервіси знаходяться у глобальній мережі AWS, то вони безпечно можуть бути отримані зі сховища та записані в нього без використання Gateway Endpoint.

2.2 Розробка архітектури підсистем

Структура сховища даних на базі хмарних технологій AWS складається з комплексу сервісів зберігання даних, що взаємодіють між собою задля забезпечення надійності, масштабованості та безпеки інформації. Компонентна схема даної підсистеми наведена на рис. 2.3.

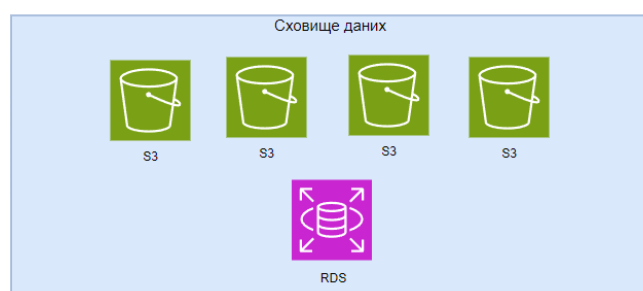


Рисунок 2.3 – Компонентна схема сховища медичних даних

В основі масштабованості підсистеми лежить використання Amazon S3 (Simple Storage Service) – хмарного сервісу для зберігання об'єктів, який надає

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			36

можливість зберігати петабайти даних та автоматично масштабуватися для забезпечення необхідного рівня доступності та продуктивності.

З погляду безпеки, дані, що зберігаються в Amazon S3, можуть бути зашифровані на різних рівнях. Використання серверного шифрування у сполученні з ключами управління доступом AWS IAM (Identity and Access Management) дозволить забезпечити конфіденційність медичних даних.

Для забезпечення надійності та відновлення даних в разі втрати використовуватиметься автоматичне резервне копіювання. Сервіс S3 дозволяє налаштувати регулярні точкові копії даних, що забезпечить можливість відновлення системи до попередніх станів у випадку непередбачених ситуацій, що гарантує надійність та стійкість системи зберігання медичних даних в умовах різних сценаріїв втрати чи пошкодження інформації. Досягнення максимальної надійності та доступності забезпечується реплікацією даних між різними центрами обробки даних.

Велика важливість приділяється безпеці даних в S3. Для забезпечення конфіденційності та цілісності можна використовувати серверне чи клієнтське шифрування. Більше того, за допомогою політик управління доступом (IAM) та унікальних ключів можна обмежити доступ до резервних копій лише авторизованим користувачам.

Відновлення з резервних копій в Amazon S3 є простим та швидким процесом, що дозволяє оперативно відновити важливі дані у випадку аварії чи видалення. Завдяки зручному інтерфейсу та різним опціям відновлення, ви можете точно визначити момент часу, до якого необхідно відновити дані.

Усі ці характеристики роблять Amazon S3 потужним і надійним інструментом для зберігання медичних даних у хмарному середовищі, забезпечуючи високий рівень доступності та безпеки.

Важливим елементом підсистеми зберігання медичних даних є реляційна база даних, у якій будуть зберігатися дані про пацієнтів та користувачів системи. Для даних цілей гарним варіантом є Amazon RDS (Relational Database Service).

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			37

RDS є хмарним сервісом для управління реляційними базами даних, який спрощує розгортання, управління та масштабування баз даних. Він дозволяє легко створювати та керувати реляційними базами даних без необхідності вирішувати складні завдання адміністрування.

Однією з ключових особливостей Amazon RDS є його підтримка різних типів реляційних баз даних, таких як MySQL, PostgreSQL, Oracle, SQL Server та інші, що надає велику гнучкість при виборі бази даних. Також RDS автоматизує багато завдань, пов'язаних з управлінням баз даних, таких як резервне копіювання, відновлення та виправлення помилок в роботі системи. Система автоматично стежить за продуктивністю бази даних, надаючи можливість реагувати на зміни в навантаженні та оптимізувати роботу бази для забезпечення ефективності.

Amazon RDS надає потужний механізм резервного копіювання для забезпечення безпеки та відновлення баз даних. Система автоматично створює регулярні резервні копії баз даних та транзакцій, що дозволяє вам відновлювати дані до конкретних моментів у минулому. Ці резервні копії зберігаються в Amazon S3, забезпечуючи високий рівень надійності та доступності. Зазвичай використовується комбінація повних резервних копій та журнальних файлів транзакцій для максимізації точності відновлення до конкретних моментів часу.

Даний сервіс також надає можливість налаштування рівня безпеки використовуючи інтеграцію з AWS Identity and Access Management (IAM) для керування доступом користувачів, а також використовувати можливості шифрування для захисту конфіденційності даних.

Одним з основних переваг використання Amazon RDS є гнучкість масштабування, оскільки можна легко змінювати розмір бази даних чи автоматично масштабувати її, щоб відповідати зростаючим потребам проекту.

Загалом, Amazon RDS дозволяє ефективно керувати реляційними базами даних в хмарному середовищі, спрощуючи багато рутинних операцій та надаючи надійну та гнучку інфраструктуру для вашого застосунку.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			38

Усі ці елементи підсистеми зберігання медичних даних взаємодіють для створення динамічної та безпечної структури зберігання, яка може масштабуватися відповідно до росту обсягів даних, захищати інформацію від несанкціонованого доступу та забезпечувати надійне резервне копіювання для відновлення в критичних ситуаціях. Структурну схему сховища медичних даних наведено на рис. 2. 4.

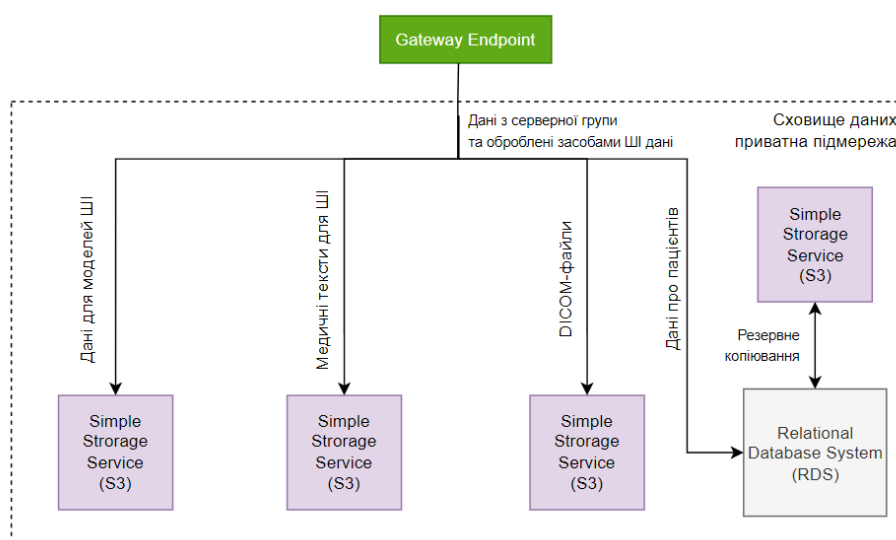


Рисунок 2.4 – Структурна схема підсистеми зберігання медичних даних

Наступною розглянемо серверну групу. Вона використовуватиме концепцію автоматичного масштабування, щоб адаптуватися до змін обсягів медичних даних, які необхідно обробити. Такі сервіси Amazon як EC2 (Elastic Cloud Compute), ASG (Auto Scaling Group) та ALB (Application Load Balancer) надають можливість динамічного збільшення чи зменшення потрібних ресурсів залежно від кількості запитів, а також рівномірно розділяти дані між серверами обробки, що підвищить ефективність та швидкість їх обробки. Компонентна схема даної частини системи наведена на рис. 2.5.

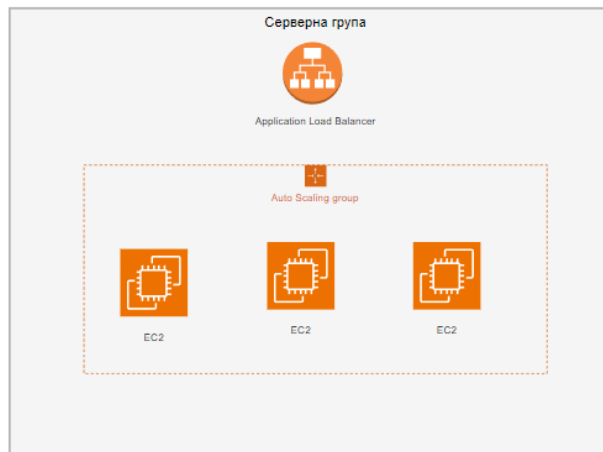


Рисунок 2.5 – Компонентна схема підсистеми обробки медичних даних

Amazon Elastic Compute Cloud (AWS EC2) - це сервіс хмарних обчислень, який надає масштабовані обчислювальні ресурси в хмарному середовищі. Він дозволяє користувачам легко запускати віртуальні сервери, відомі як екземпляри, для виконання різноманітних завдань.

Перевагами даного сервісу є:

1. Масштабованість: можливість миттєвого масштабування ресурсів вгору або вниз в залежності від потреб, що дозволяє користувачам оптимізувати витрати та забезпечити необхідну продуктивність.

2. Гнучкість: різноманітні типи екземплярів для різних видів завдань, включаючи оптимізовані під певні типи робіт.

3. Еластичні образи: користувачі можуть використовувати готові образи або ж створювати власні, що дає можливість швидкого розгортання серверів з необхідними конфігураціями та програмним забезпеченням.

4. Широкий вибір операційних систем: підтримка різноманітних популярних операційних системи, в тому числі й Linux та Windows.

Проте разом з перевагами даний сервіс має ряд недоліків, серед яких:

1. Складність ціноутворення: ціни на екземпляри можуть бути складними і варіюватися в залежності від багатьох факторів, таких як тип екземпляра, регіон та інші параметри.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			40

2. Можливий ризик безпеки: використання публічних хмарних ресурсів може призвести до ризиків забезпечення безпеки даних, хоча AWS надає різноманітні інструменти для захисту інфраструктури.

3. Залежність від Інтернету: для використання EC2 користувачам потрібний доступ до Інтернету, що може бути проблематичним в деяких випадках.

Загалом, AWS EC2 є потужним та гнучким інструментом для обчислювальних завдань в хмарному середовищі, проте користувачам важливо розуміти його особливості та обмеження при використанні.

Наступним елементом підсистеми є Application Load Balancer (AWS ALB), який є сервісом балансування навантаження, спеціально призначеним для розподілу трафіку між різними екземплярами нашої підсистеми, що дозволяє покращити надійність та доступність обчислювальних ресурсів.

Одна з ключових особливостей AWS ALB – це можливість працювати на рівні додатків (7 рівень в моделі OSI). Це означає, що він може розуміти інформацію на рівні HTTP та HTTPS, включаючи обробку маршрутів та правил перенаправлення. AWS ALB також підтримує WebSocket, що дозволить створювати багатофункціональні додатки, які працюватимуть в режимі реального часу та які вимагають постійного з'єднання між клієнтом та сервером.

Одна з інших важливих рис ALB – це автоматична масштабованість. Він може динамічно адаптувати розподіл трафіку в залежності від змін у навантаженні, забезпечуючи ефективне використання ресурсів та високу доступність. Важливим елементом є також можливість використання сертифікатів SSL/TLS безпосередньо на ALB, що робить його ідеальним вибором для забезпечення безпеки та конфіденційності взаємодії підсистем.

Загалом, AWS ALB є потужним інструментом для забезпечення балансування навантаження в хмарних середовищах, забезпечуючи високий рівень доступності та ефективного управління трафіком нашої підсистеми обробки медичних даних.

Важливим елементом підсистеми є Auto Scaling Group – це сервіс управління групами віртуальних серверів у хмарному середовищі AWS. Основна

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			41

ідея ASG полягає в автоматизації масштабування групи віртуальних серверів відповідно до змін навантаження або інших умов.

Загалом, ASG робить процес масштабування віртуальних серверів більш ефективним та гнучким та дозволяє визначити параметри автоматичного масштабування, такі як мінімальна та максимальна кількість екземплярів у групі. На основі цих параметрів ASG автоматично додає чи видаляє екземпляри відповідно до змін потреб у навантаженні. Сервіс також враховує стан керованих екземплярів: якщо виявляється, що один з них несправний, ASG автоматично замінить його новим екземпляром, що забезпечує високий рівень доступності та надійності для системи.

Auto Scaling Groups є потужним інструментом для автоматизації масштабування інфраструктури в хмарному середовищі. Однією з основних переваг є можливість динамічно адаптувати ресурси під навантаження за допомогою налаштувань автоматичного масштабування. Це дозволяє ефективно використовувати ресурси, забезпечуючи необхідну продуктивність в разі підвищення обсягу роботи. Крім того, використання AWS ASG спрощує процес управління ресурсами, оскільки ви можете визначити правила для автоматичного масштабування, а система буде самостійно виконувати зміни відповідно до цих правил. Це дозволяє значно заощаджувати час і знижує ймовірність помилок під час ручного масштабування.

З іншого боку, важливо враховувати, що AWS ASG не є універсальним рішенням і може виявитися складним у деяких сценаріях.

Також важливо враховувати витрати, пов'язані з використанням AWS ASG. Автоматичне масштабування може призвести до збільшення вартості послуг, оскільки додаткові ресурси будуть використовуватися відповідно до потреб. Потрібно ретельно розглядати вартість і вигоди автоматичного масштабування для конкретного проекту.

Отже, хоча AWS ASG може ефективно вирішувати багато завдань з автоматизації масштабування інфраструктури, важливо враховувати його обмеження

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			42

і здатність вирішувати конкретні виклики, які можуть виникнути у вашому проєкті. Структурна схема серверної групи наведена на рис. 2.6.

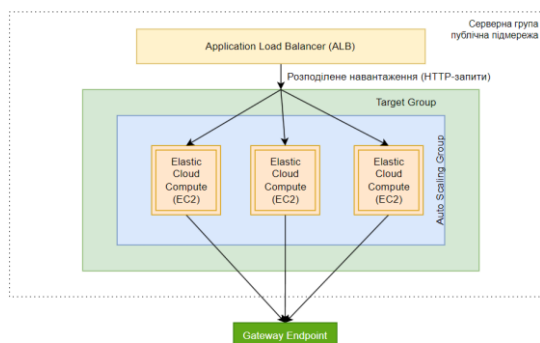


Рисунок 2.6 – Структурна схема підсистеми обробки медичних даних

Структура частини системи з сервісами AWS має бути розроблена з метою забезпечення ефективної обробки та аналізу медичних даних з використанням штучного інтелекту. Компонентна схема даної підсистеми наведена на рис. 2.7.

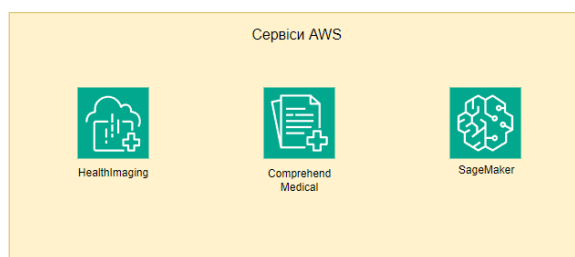


Рисунок 2.7 – Компонентна схема частини системи з сервісами AWS

Розберемо детально її елементи, а саме такі засоби як SageMaker, ComprehendMedical та HealthImaging.

Amazon SageMaker - це хмарний сервіс у сфері машинного навчання, наданий Amazon Web Services (AWS). Він призначений для спрощення та прискорення процесу розробки, навчання і впровадження моделей машинного навчання в хмарному середовищі.

SageMaker надає можливість будувати, тренувати та оптимізувати моделі машинного навчання без необхідності глибокого розуміння інфраструктури або складних алгоритмів. Він дає можливість використовувати готові алгоритми, або створювати власні моделі, враховуючи різноманітні завдання - від класифікації до прогнозування.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			43

Однією з ключових переваг SageMaker є його інтеграція з іншими сервісами AWS, такими як S3, Lambda та інші, що дозволяє легко обмінюватися даними та використовувати інші хмарні сервіси для автоматизації різних етапів роботи з моделями.

Крім того, SageMaker надає можливість створення експериментів, візуалізації результатів навчання та автоматичного масштабування ресурсів відповідно до потреб вашого проекту. Це дозволяє ефективно використовувати обчислювальні ресурси та прискорює розгортання моделей в продуктивне середовище.

Загалом, Amazon SageMaker спрощує процес роботи з машинним навчанням в хмарному середовищі, забезпечуючи високий рівень автоматизації та інтеграції з іншими сервісами AWS.

Одним із основних використаннях SageMaker в медицині є розвиток моделей машинного навчання для прогнозування захворювань, діагностики та планування лікування. За допомогою SageMaker вчені та медичні фахівці можуть використовувати різноманітні дані, такі як медичні зображення, генетичні дані, клінічні записи тощо, для створення моделей, які допомагають в ранньому виявленні хвороб та виборі оптимального методу лікування.

Як і будь-який інший сервіс SageMaker має свої переваги та недоліки. Серед переваг можна виділити:

1. Легкість використання: сервіс надає інтуїтивний інтерфейс та API, що полегшує процес розробки, навчання та розгортання моделей машинного навчання.

2. Масштабованість: платформа може легко масштабуватися для обробки великих обсягів даних та великої кількості обчислювальних ресурсів.

3. Широкий вибір алгоритмів: SageMaker має багатий набір вбудованих алгоритмів машинного навчання та підтримує різні фреймворки, такі як TensorFlow, PyTorch та інші.

4. Автоматизований процес: сервіс надає можливості автоматичного налаштування параметрів моделі та вибору алгоритму для досягнення кращих результатів.

5. Інтеграція з іншими сервісами AWS: легка інтеграція з іншими сервісами Amazon Web Services дозволяє створювати повноцінні рішення на основі хмарних обчислень.

Недоліками сервісу є:

1. Вартість: використання SageMaker може бути вартісним, особливо при великому обсязі обчислень та використанні інших сервісів AWS.

2. Залежність від хмарної інфраструктури: використання сервісу пов'язане з інфраструктурою AWS, що може створювати обмеження в разі необхідності переходу на інші хмарні сервіси або локальні рішення.

3. Обмежена гнучкість: хоча SageMaker має широкий функціонал, може бути ситуації, коли розробнику не вистачає гнучкості для вирішення специфічних завдань.

4. Навчання моделі в реальному часі: деякі користувачі відзначають, що процес навчання моделі в реальному часі може бути більш витратним в порівнянні з іншими методами навчання моделей.

Крім того, цей сервіс може бути використаний для розробки персоналізованих моделей лікування, враховуючи індивідуальні особливості пацієнта, що сприяє розвитку підходів до медицини, орієнтованих на пацієнта.

Таке використання AWS SageMaker в медицині може значно покращити якість надання медичних послуг, зменшити час діагностики та лікування, а також викликати нові можливості для наукових досліджень у галузі медицини та біології.

Amazon Comprehend Medical – це сервіс від AWS, спроектований для обробки та аналізування медичної інформації. Головна мета цього сервісу - автоматизувати та полегшити процеси обробки медичних текстів та документів. Він використовує технології обробки природної мови для витягування корисної інформації з текстів, пов'язаних з медичними записами.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			45

Завдяки Amazon Comprehend Medical можна виконувати розпізнавання та аналіз текстової інформації, пов'язаної зі схильністю до хвороб, медичними умовами, лікарняними виписками та іншими медичними контекстами. Це допомагає організаціям у сфері охорони здоров'я ефективніше управляти та використовувати свої дані, щоб отримати більше цінної інформації з них.

Даний сервіс може розпізнавати медичні терміни, визначати взаємозв'язки між різними поняттями, та витягувати важливі дані, які можуть бути використані для покращення клінічних досліджень, управління хворобами та інших медичних аспектів. Це стає важливим інструментом для медичних установ та організацій, що прагнуть оптимізувати свої процеси та винести більше значущих висновків з обширної медичної інформації.

Ця технологія вирішує завдання витягання суттєвої інформації з лікарських записів, рецептів, клінічних документів та інших медичних текстів, що дозволяє медичним установам та фахівцям отримувати більше значущих даних.

В першу чергу, сервіс допомагає автоматизувати процес аналізу та категоризації медичної інформації. Він розпізнає медичні терміни, визначає зв'язки між хворобами та симптомами, а також ефективно розуміє контекст лікарських умов. Це забезпечує медичним фахівцям і організаціям доступ до швидкого та точного аналізу великої кількості медичної документації.

Крім того, використання Comprehend Medical може покращити клінічні дослідження та допомагати в управлінні опікою пацієнтів. За допомогою цієї технології можна швидше виявляти тенденції в хворобах, забезпечуючи таким чином раннє виявлення та ефективніші методи лікування.

Розглянемо переваги та недоліки даного сервісу.

Перевагами Amazon Comprehend Medical є:

1. Автоматизація обробки медичних текстів: автоматичний аналіз великих обсягів медичної інформації та швидке виділення ключових понять та елементів.

		Андрусенко О.М.			Кім.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			46

2. Ефективність та збереження часу: швидка обробка та отримання інформації із медичних текстів, тобто економія часу, який зазвичай витрачається на ручний аналіз.

3. Масштабованість: масштабування для обробки великих обсягів даних, що робить сервіс ефективним для великих медичних організацій та дослідницьких проектів.

4. Інтеграція з іншими сервісами Amazon: сервіс можна ефективно інтегрувати з іншими сервісами Amazon.

Недоліками Amazon Comprehend Medical є:

1. Точність інтерпретації: як і в будь-якій автоматизованій системі обробки тексту, може виникати питання точності інтерпретації медичної інформації, особливо у випадках складних або специфічних сценаріїв.

2. Необхідність у перевірці: отримані результати можуть вимагати додаткової перевірки від медичних фахівців, оскільки неправильна інтерпретація може призвести до невірних висновків.

3. Вартість використання: хоч використання сервісу дозволяє значно економити час, вартість його використання може бути вагомим фактором, особливо для маленьких медичних установ з обмеженим бюджетом.

4. Залежність від якості вхідних даних: точність результатів може значно залежати від якості та форматування вхідних медичних текстів.

Загалом, Amazon Comprehend Medical може бути потужним інструментом для обробки медичної інформації, але його використання вимагає уважного підходу та валідації результатів від фахівців.

Впровадження Amazon Comprehend Medical у медичному секторі допомагає зробити обробку даних більш ефективною, забезпечуючи швидку і точну аналітику, яка є критично важливою для вдосконалення медичної діагностики та лікування.

Останнім елементом даної частини системи є Amazon HealthImaging. AWS HealthImaging – це сервіс, що відповідає вимогам HIPAA, яка дає змогу медич-

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			47

ним закладам та їхнім партнерам зберігати, аналізувати та обмінюватися медичними зображеннями в хмарі в петабайтному масштабі. HealthImaging забезпечує майже миттєвий доступ до даних зображень (наприклад, рентгену, КТ, МРТ, УЗД), завдяки чому додатки для медичної візуалізації, побудовані в хмарі, можуть досягти продуктивності, яка раніше була можлива лише в локальній мережі. За допомогою HealthImaging зменшуються витрати на інфраструктуру, запускаючи свої додатки для медичної візуалізації в масштабі з єдиної авторитетної копії кожного медичного зображення в AWS Cloud.

AWS HealthImaging надає наступні можливості:

Зручні для розробників метадані DICOM: AWS HealthImaging спрощує розробку додатків, повертаючи метадані DICOM у зручному для розробників форматі. Після імпорту ваших даних візуалізації окремі атрибути метаданих доступні за допомогою зручних для людини ключових слів, а не незнайомих шістнадцяткових чисел груп/елементів.

Декодування зображень з прискоренням SIMD: AWS HealthImaging повертає кадри зображень (піксельні дані), закодовані за допомогою High Throughput JPEG 2000 (HTJ2K), вдосконаленого кодека стиснення зображень. HTJ2K використовує переваги технології "одна команда - декілька даних" (SIMD) на сучасних процесорах для забезпечення нового рівня продуктивності. HTJ2K на порядок швидший за JPEG2000 і щонайменше вдвічі швидший за всі інші синтаксиси передачі DICOM.

Верифікація піксельних даних: AWS HealthImaging забезпечує вбудовану верифікацію піксельних даних, перевіряючи стан кодування і декодування без втрат кожного зображення під час імпорту.

Найкраща в галузі продуктивність: AWS HealthImaging встановлює новий стандарт продуктивності завантаження зображень завдяки ефективному кодуванню метаданих, стисненню без втрат і доступу до даних з прогресивною роздільною здатністю. Ефективне кодування метаданих дозволяє програмам перегляду зображень і алгоритмам штучного інтелекту розуміти вміст DICOM-до-

		Андрусенко О.М.			Кім.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			48

слідження без необхідності завантажувати дані зображення. Зображення завантажуються швидше без втрати якості завдяки вдосконаленому стисненню зображень. Прогресивна роздільна здатність дозволяє ще швидше завантажувати ескізи, області інтересу і мобільні пристрої з низькою роздільною здатністю.

Масштабований імпорт DICOM: AWS HealthImaging використовує сучасні хмарні технології для паралельного імпорту декількох досліджень у форматі DICOM. Історичні архіви можна імпортувати швидко, не впливаючи на клінічне навантаження для нових даних.

Перевагами AWS HealthImaging є:

Безпека: забезпечує високий рівень безпеки для медичних зображень, використовуючи технології, такі як шифрування, аутентифікація та авторизація.

Масштабованість: сервіс можна масштабувати відповідно до потреб організації, щоб підтримувати зростання обсягів даних.

Ефективність: допомагає організаціям медичної галузі оптимізувати використання ресурсів та знизити витрати.

Аналіз: пропонує широкий спектр функцій для аналізу медичних зображень, які можуть допомогти лікарям у діагнозі та лікуванні пацієнтів.

Недоліки AWS HealthImaging включають в себе:

Вартість: сервіс може бути дорогим для невеликих організацій.

Складність: AWS HealthImaging може бути складним для використання, особливо для організацій, які не мають досвіду роботи з хмарами.

Загалом, AWS HealthImaging – це потужний інструмент, який може допомогти організаціям медичної галузі у покращенні якості медичної допомоги, зниженні витрат та підвищенні ефективності, однак важливо враховувати його потенційні недоліки.

Структурна схема даної частини системи наведена на рис. 2.8.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			49

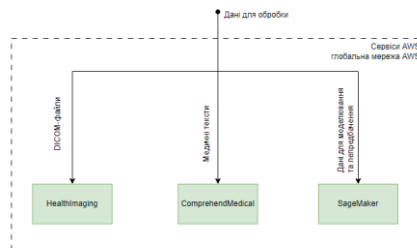


Рисунок 2.8 – Структурна схема частини системи з сервісами AWS

2.3 Алгоритм роботи системи. Взаємодія підсистем у системі

Після вибору складових частин системи необхідно описати алгоритм роботи системи в цілому. Загалом, робота системи складатиметься з таких ключових етапів як: збір, обробка та аналіз даних. Також важливим елементом алгоритму є перевірка роботи системи на наявність помилок. Алгоритм роботи системи наведений на рис. 2.9.



Рисунок 2.9 – Загальний алгоритм роботи системи

Опишемо детально наведений алгоритм. На початку роботи починається збір даних від користувачів, або ж від медичних пристроїв. Після збору дані відправляються до сховища даних та записуються до бази даних та відповідних сховищ об'єктів.

Наступним етапом є обробка та аналіз зібраних даних засобами штучного інтелекту. Це включає в себе отримання даних з медичних текстів, побудову мо-

делей машинного навчання та їх використання для передбачень, а також сортування зображень медичних досліджень. Далі оброблені дані відправляються знову в сховище даних для їх зберігання.

Після кожної такої ітерації відбувається перевірка системи на помилки. Якщо такі наявні, то система завершить свою роботу та відобразить помилку в консолі керування AWS, якщо ж ні – продовжить цикл спочатку.

Перейдемо до опису взаємодії частин системи між собою. Серверна група отримуватиме дані від користувачів та медичних пристроїв та записуватиме їх до бази даних та сховищ об'єктів. Зі сховища даних відповідним сервісами передаватимуться дані для обробки та аналізу. Після закінчення опрацювання дані буде знову передано до сховища даних, а також до серверної групи для ознайомлення користувачами. Модель взаємодії підсистем між собою наведена на рис. 2.10.

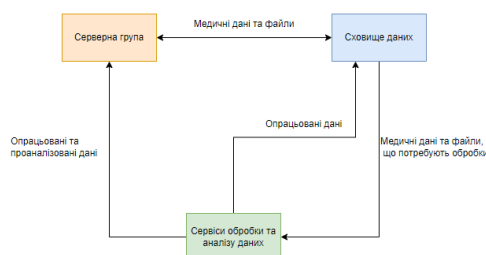


Рисунок 2.10 – Модель взаємодії частин системи між собою

Висновки з розділу 2

В другому розділі було спроектовано систему зберігання, обробки та аналізу медичних даних. Проект було виконано на хмарній платформі Amazon Web Services.

Після проведення аналізу було обрано хмарні сервіси для кожної складової системи, а також наведено їх переваги та недоліки. Кожна частина системи складатиметься з декількох таких сервісів та буде налаштована із дотриманням принципів високої швидкості обробки даних, продуктивності та доступності. Також було описано алгоритм роботи системи та взаємодію між її частинами.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ ЗБЕРІГАННЯ, АНАЛІЗУ ТА ОБРОБКИ МЕДИЧНИХ ДАНИХ З ВИКОРИСТАННЯМ ХМАРНИХ СЕРВІСІВ ТА ШТУЧНОГО ІНТЕЛЕКТУ

3.1. Загальні налаштування хмарної мережі системи

Підібравши необхідні для функціонування системи хмарні сервіси перейдемо до їх створення та налаштування.

Для автоматизації та спрощення процедури розгортання системи буде використано AWS Console, а також такий засіб як Terraform.

Terraform - це інструмент для інфраструктурного коду, розроблений компанією HashiCorp. Він дозволяє автоматизувати процес створення, налаштування та управління інфраструктурою в хмарних середовищах та на локальних серверах. Він використовує декларативний підхід до конфігурації, дозволяючи описувати бажаний стан інфраструктури у вигляді коду. Terraform підтримує різноманітних провайдерів, що дозволяє вам управляти ресурсами в різних хмарних платформах, в тому числі й AWS, на базі якої буде створено нашу систему.

Першим етапом є створення користувача в AWS Identity Access Management (IAM), від імені якого будуть виконуватися скрипти та отримання ключів доступу до нього. Для цього перейдемо на вкладку Users в IAM за допомогою веб-інтерфейсу та натиснемо кнопку «Create user», рисунок 3.1.

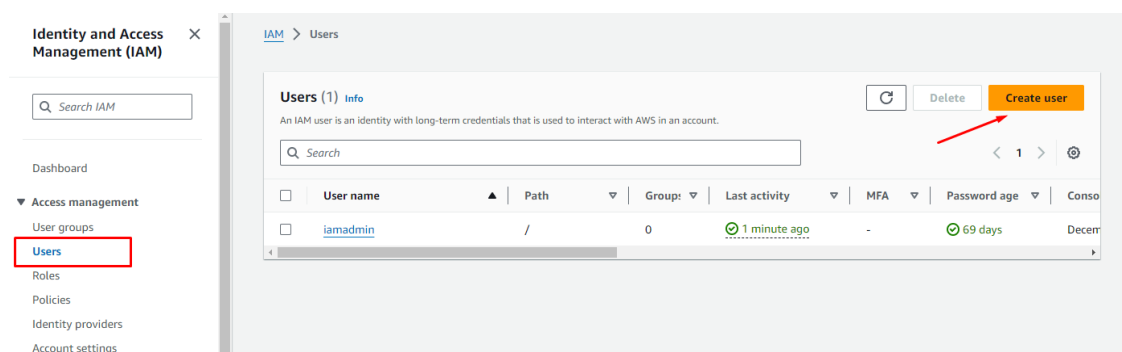


Рисунок 3.1 – Створення нового користувача в IAM

Наступним кроком створюємо ім'я користувача (в даному випадку – terraform), рисунок 3.2.

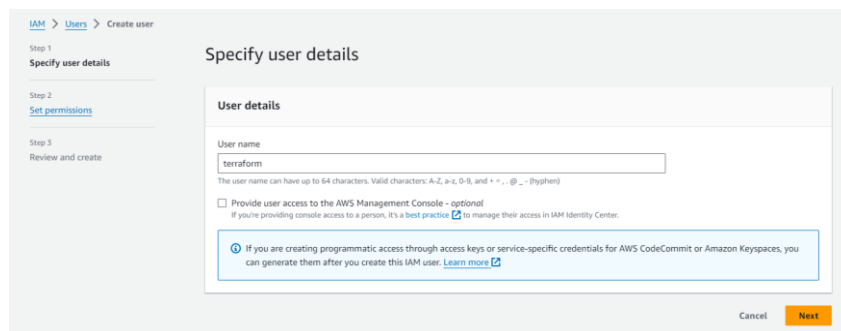


Рисунок 3.2 – Налаштування імені користувача

Після чого надаємо даному користувачу необхідні для розгортання інфраструктури системи за допомогою скриптів дозволи, а саме прикріплюємо до даного користувача існуючу політику безпеки під назвою AdministratorAcces, рис. 3.3.

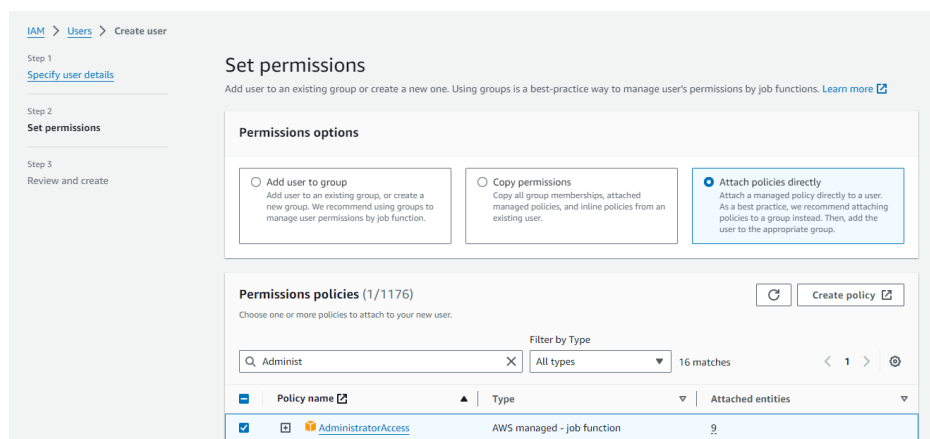


Рисунок 3.3 – Налаштування дозволів користувача

Після створення користувача необхідно згенерувати ключі доступу для нього. Для цього скористаємось кнопкою «Create access key» на сторінці створеного вище користувача. На сторінці створення ключів обираємо мету його використання, а саме пункт інтерфейс командного рядка (CLI), рисунок 3.4.

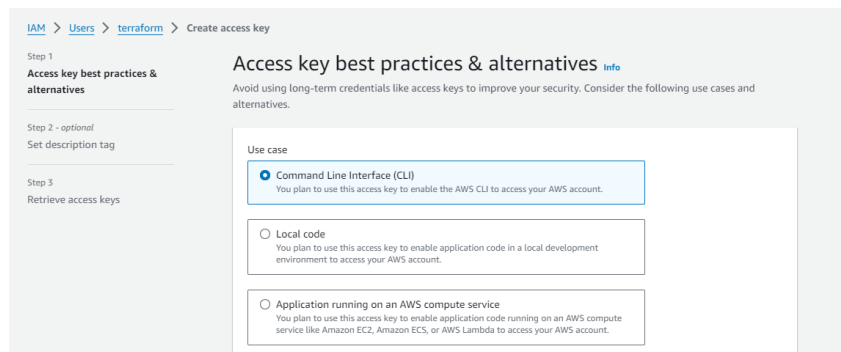


Рисунок 3.4 – Вибір мети використання ключів доступу

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	53
		Бродський Ю. Б.				
		№ докум.	Підпис			

Після успішного створення ключів вони з'являться на екрані, звідки їх можна скопіювати та використовувати, а також завантажити їх у CSV-форматі, рисунок 3.5.

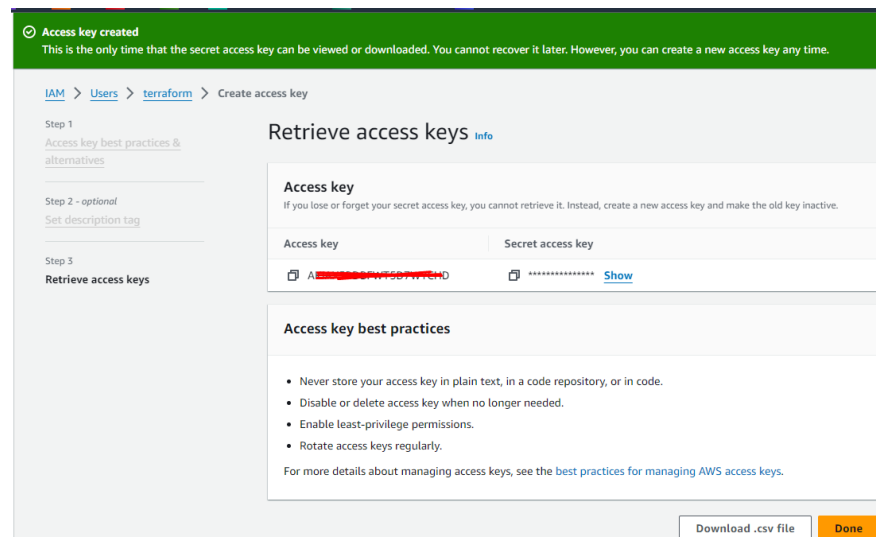


Рисунок 3.5 – Створені ключі доступу

Наступним кроком є створення Terraform-скрипта для розгортання нашої системи. На його початку необхідно створити та налаштувати провайдера хмарної платформи з використанням створених вище ключів доступу та вказати регіон розміщення ресурсів за замовчуванням. Відповідний фрагмент коду зображено на рисунку 3.6.

```
//Provider settings
provider "aws" {
  access_key = "AKIA5BDBFWT5D7WYCHD"
  secret_key = "n6vhpX/H+Z55kLwt1fluT38nFP+07y9zH1FAharH"
  region     = "us-east-1"
}
```

Рисунок 3.6 – Налаштування провайдера хмарної платформи

Далі необхідно створити віртуальну приватну хмарну мережу (VPC) та налаштувати її. При налаштуванні вказуємо адресу мережі 10.1.0.0 з префіксом 18, а також назвемо даний елемент «Main system VPC». Відповідний фрагмент коду зображено на рисунку 3.7.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			54

```
//VPC
resource "aws_vpc" "main-system-vpc" {
  cidr_block = "10.1.0.0/18"
  tags = {
    Name = "Main system VPC"
  }
}
```

Рисунок 3.7 – Налаштування хмарної мережі (VPC)

Перейдемо до поділу створеної мережі на підмережі, яких буде 5: 2 публічних для доступу до системи з мережі Інтернет за допомогою Application Load Balancer та 2 приватних для розміщення серверів. Параметри адресації підмереж наведені у таблиці 3.1. Підмережі в коді пов'язуються зі створеною вище хмарною мережею відбуваються за допомогою вказання її ідентифікатора. Відповідний фрагмент скрипта наведений на рисунках 3.8-3.9.

Таблиця 3.1 – Параметри адресації підмереж системи

Назва підмережі	Адреса підмережі	Зона доступності (AZ)	Публічність
alb-public-subnet-1	10.1.1.0/28	us-east-1a	Так
alb-public-subnet-2	10.1.2.0/28	us-east-1b	Так
servers-private-subnet-1	10.1.3.0/24	us-east-1a	Ні
servers-private-subnet-2	10.1.4.0/24	us-east-1b	Ні

```
//Public subnets for ALB
resource "aws_subnet" "alb-public-subnet-1" {
  vpc_id = aws_vpc.main-system-vpc.id
  cidr_block = "10.1.1.0/28"
  availability_zone = "us-east-1a"
  map_public_ip_on_launch = "true"

  tags = {
    Name = "alb-public-subnet-1"
  }
}
resource "aws_subnet" "alb-public-subnet-2" {
  vpc_id = aws_vpc.main-system-vpc.id
  cidr_block = "10.1.2.0/28"
  availability_zone = "us-east-1b"
  map_public_ip_on_launch = "true"

  tags = {
    Name = "alb-public-subnet-2"
  }
}
```

Рисунок 3.8 – Налаштування публічних підмереж системи

```
//Private subnets fro servers
resource "aws_subnet" "servers-private-subnet-1" {
  vpc_id            = aws_vpc.main-system-vpc.id
  cidr_block        = "10.1.3.0/24"
  availability_zone = "us-east-1a"
  tags = {
    Name = "servers-private-subnet-1"
  }
}

resource "aws_subnet" "servers-private-subnet-2" {
  vpc_id            = aws_vpc.main-system-vpc.id
  availability_zone = "us-east-1b"
  cidr_block        = "10.1.4.0/24"
  tags = {
    Name = "servers-private-subnet-2"
  }
}
```

Рисунок 3.9 – Налаштування приватних підмереж системи

Важливим етапом є налаштування шлюзів системи. В даному випадку їх буде два: Інтернет-шлюз (Internet Gateway) для публічних підмереж та NAT-шлюз (NAT Gateway) для приватних підмереж. Дані елементи забезпечать компонентам системи можливість доступу до глобальної мережі.

Для створення Інтернет-шлюзу необхідно вказати ідентифікатор хмарної мережі, до якої він буде прив'язаний, а також його ім'я. Відповідний фрагмент коду наведений на рисунку 3.10.

```
//Internet gateway
resource "aws_internet_gateway" "system-igw" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "Main internet gateway"
  }
}
```

Рисунок 3.10 – Створення Інтернет-шлюзу (Internet Gateway)

Створення NAT-шлюзу потребує наявності еластичної IP-адреси для його створення. Elastic IP в даному контексті використовується для стабільної ідентифікації NAT Gateway в Amazon Web Services (AWS). Важливість Elastic IP полягає в тому, що вона надає сталу точку доступу до NAT Gateway з інтернету. При створенні такої адреси необхідно вказати зону її дії, в нашому випадку це наша хмарна мережа (VPC), а також ім'я для зручної ідентифікації адреси при

необхідності. Фрагмент коду, який відповідає за створення еластичної адреси для шлюзу наведений на рисунку 3.11.

```
//Elastic IP
resource "aws_eip" "nat-eip" {
  domain = "vpc"
  tags = {
    "Name" = "Elastic IP for NAT gateway"
  }
}
```

Рисунок 3.11 – Створення еластичної IP-адреси (Elastic IP)

Далі створюємо NAT-шлюз, в налаштуваннях якого прив'язуємо до нього створену вище еластичну IP-адресу та додаємо цей шлюз до однієї з публічних підмереж. Відповідний фрагмент коду наведений на рисунку 3.12.

```
//NAT gateway
resource "aws_nat_gateway" "nat-gw" {
  allocation_id = aws_eip.nat-eip.id
  subnet_id     = aws_subnet.alb-public-subnet-1.id

  tags = {
    Name = "NAT gateway"
  }
}
```

Рисунок 3.12 – Створення NAT-шлюзу (NAT Gateway)

Наступним кроком є налаштування маршрутизації для забезпечення зв'язку між компонентами системи, а також доступу за її межі.

Для публічних підмереж налаштування проводиться шляхом редагування основної таблиці маршрутизації хмарної мережі (Default VPC Route Table). До даної таблиці потрібно додати маршрут, який скеровуватиме весь трафік на створений раніше Інтернет-шлюз, а також створити асоціації публічних підмереж системи з цією таблицею. Фрагмент коду, який виконує дані дії наведений на рисунку 3.13.

```
//Edit default RT
resource "aws_default_route_table" "def-rt"{
  default_route_table_id = aws_vpc.main-system-vpc.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.system-igw.id
  }
}

//Route tables associations (public)
resource "aws_route_table_association" "alb-pub-1-rt" {
  subnet_id      = aws_subnet.alb-public-subnet-1.id
  route_table_id = aws_vpc.main-system-vpc.default_route_table_id
}
resource "aws_route_table_association" "alb-pub-2-rt" {
  subnet_id      = aws_subnet.alb-public-subnet-2.id
  route_table_id = aws_vpc.main-system-vpc.default_route_table_id
}
```

Рисунок 3.13 – Редагування таблиці маршрутизації та додавання публічних підмереж до неї

Налаштування маршрутизації приватних підмереж дещо відрізняється. Для цього потрібно створити нову таблицю маршрутизації, до якої додати маршрут, який скеровуватиме весь трафік з приватних підмереж до NAT-шлюзу, а також створити асоціації приватних підмереж зі створеною таблицею. Відповідний фрагмент коду наведений на рисунку 3.14.

```
//Route table (NAT)
resource "aws_route_table" "nat-rt" {
  vpc_id = aws_vpc.main-system-vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_nat_gateway.nat-gw.id
  }

  tags = {
    Name = "NAT route table"
  }
}

//Route tables associations (private)
resource "aws_route_table_association" "servers-priv-1-rt" {
  subnet_id      = aws_subnet.servers-private-subnet-1.id
  route_table_id = aws_route_table.nat-rt.id
}
resource "aws_route_table_association" "servers-priv-2-rt" {
  subnet_id      = aws_subnet.servers-private-subnet-2.id
  route_table_id = aws_route_table.nat-rt.id
}
```

Рисунок 3.14 – Створення таблиці маршрутизації та додавання приватних підмереж до неї

Останнім етапом загального налаштування системи є створення IAM-ролі з відповідними дозволами для доступу серверів до необхідних сервісів. Для

цього оберемо EC2 зі списку сервісів, для яких призначена робхль при її створенні, рисунок 3.15.

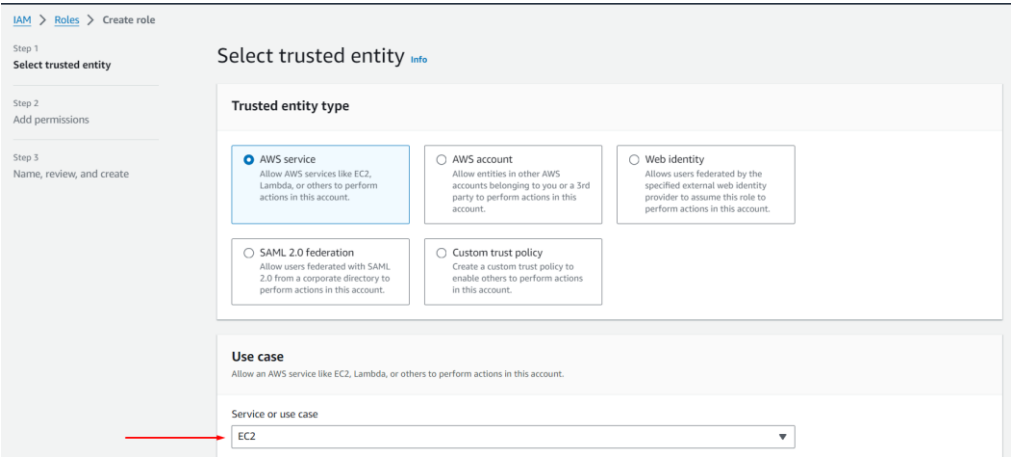


Рисунок 3.15 – Вибір сервісу призначення при створенні ролі

Наступним кроком обираємо потрібні дозволи. В даному випадку це AmazonS3FullAccess для доступу до сховища даних та AmazonSSMManagedInstanceCore для захищеного віддаленого підключення до серверів з метою тестування, рисунок 3.16.

Permissions policy summary		
Policy name	Type	Attached as
AmazonS3FullAccess	AWS managed	Permissions policy
AmazonSSMManagedInstanceCore	AWS managed	Permissions policy

Рисунок 3.16 – Надання необхідних дозволів ролі

Останнім кроком створення ролі є створення її назви та опису, рисунок 3.17.

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.

server-role

Maximum 64 characters. Use alphanumeric and '+', '@', '-' characters.

Description
Add a short explanation for this role.

Allows EC2 instances to connect to S3 and allow SSM.

Maximum 1000 characters. Use alphanumeric and '+', '@', '-' characters.

Рисунок 3.17 – Присвоєння імені та опису ролі

Скрипт загального налаштування мережі готовий, запустимо його та переконаємося у його працездатності. Для цього запустимо створений скрипт та перейдемо до веб-інтерфейсу AWS.

В першу чергу перевіримо створення хмарної мережі та підмереж, рисунок 3.17.

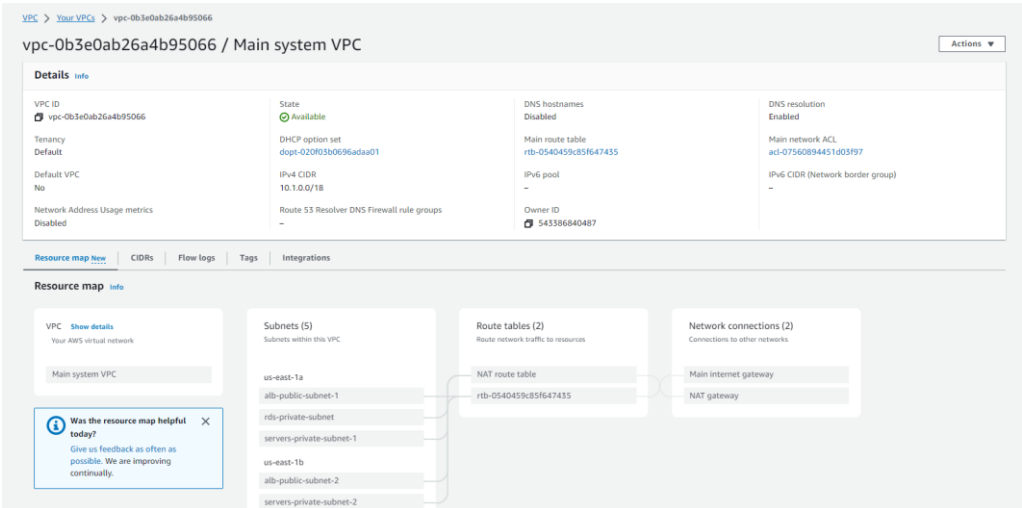


Рисунок 3.17 – Створена хмарна мережа та підмережі в ній

Наступним кроком переглянемо створені шлюзи, рисунок 3.18, та таблиці маршрутизації, рисунки 3.19-3.20.

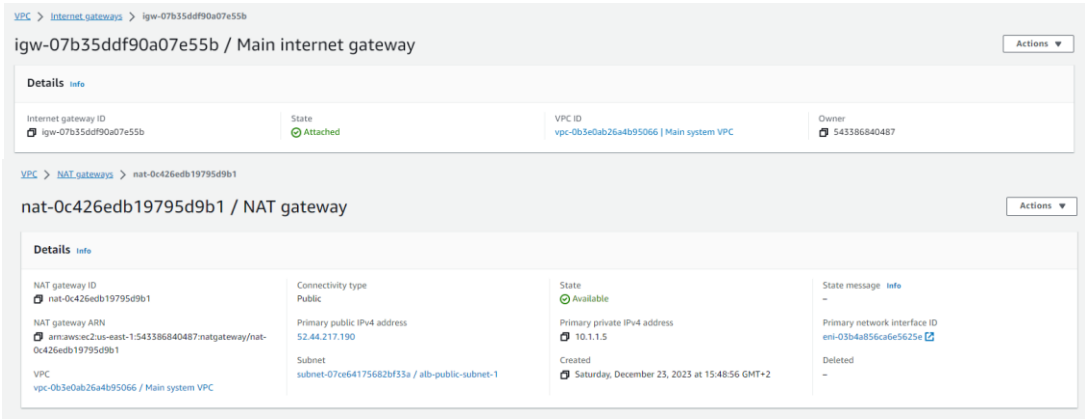


Рисунок 3.18 – Створені шлюзи

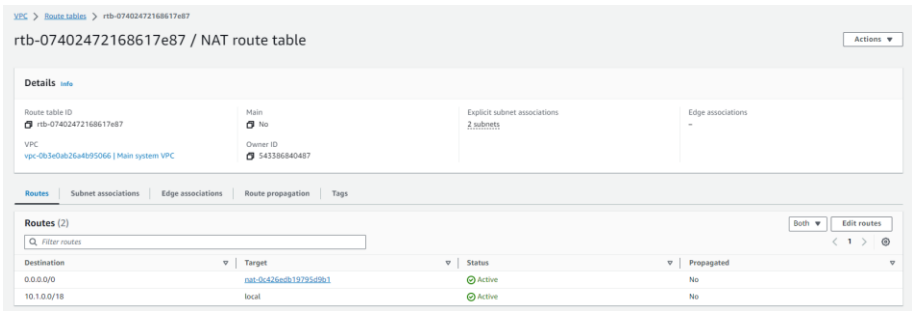


Рисунок 3.19 – Таблиця маршрутизації для приватних підмереж

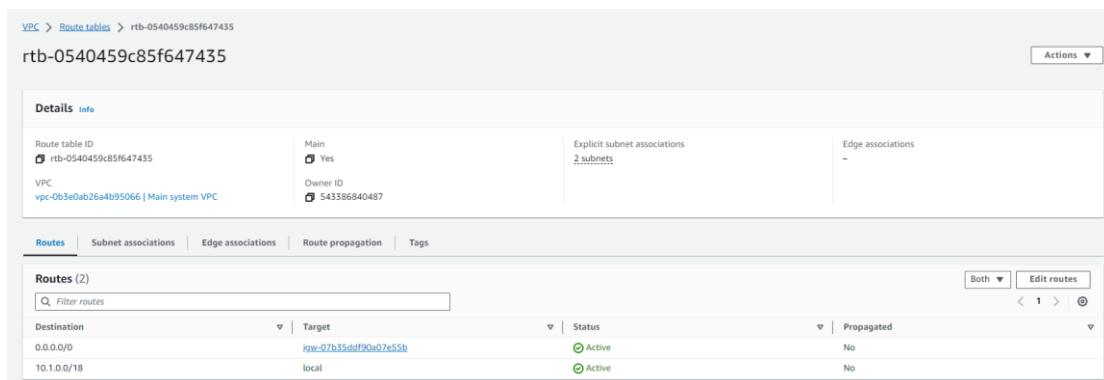


Рисунок 3.20 – Таблиця маршрутизації для публічних підмереж

Всі описані в скрипті елементи успішно створено, отже загальне налаштування хмарної мережі системи завершено.

3.2 Налаштування серверної групи

Перейдемо до налаштування компонентів серверної групи, яке здійснюватиметься аналогічно попереднім за допомогою скриптів.

В першу чергу необхідно налаштувати групи безпеки, які контролюватимуть трафік. Загалом в серверній групі буде створено дві групи безпеки: для Application Load Balancer (ALB) та для серверів.

Почнемо зі створення групи безпеки для ALB. На початку налаштування прив'язуємо дану групу до хмарної мережі системи за допомогою її ідентифікатора. Після цього до створеної групи додаємо правила для вхідного та вихідного трафіку. На вхід дозволитиметься будь-який HTTP-трафік, вихідний трафік буде дозволено лише до серверів. Відповідний фрагмент коду наведено на рисунку 3.21.

```
//Security group for ALB
resource "aws_security_group" "alb-sg" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "ALB security group"
  }
}

resource "aws_security_group_rule" "inbound-alb" {
  type      = "ingress"
  from_port = 80
  to_port   = 80
  protocol  = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
  security_group_id = aws_security_group.alb-sg.id
}

resource "aws_security_group_rule" "outbound-alb" {
  type      = "egress"
  from_port = 80
  to_port   = 80
  protocol  = "tcp"
  security_group_id = aws_security_group.alb-sg.id
  source_security_group_id = aws_security_group.instance-sg.id
}
```

Рисунок 3.21 – Створення групи безпеки для ALB та правил для неї

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			61

Наступною створимо групу безпеки для серверів. Прив'язування до хмарної мережі та додавання правил до даної групи здійснюється аналогічно як в попередній групі безпеки. Проте в даній групі вхідний трафік буде дозволено лише зі створеної вище групи безпеки для ALB, на вихідний трафік обмежень немає. Фрагмент коду зі створенням та налаштуванням цієї групи безпеки наведений на рисунку 3.22.

```
//Security group for instances
resource "aws_security_group" "instance-sg" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "Instance security group"
  }
}

resource "aws_security_group_rule" "inbound-instance" {
  type            = "ingress"
  from_port       = 80
  to_port         = 80
  protocol        = "tcp"
  security_group_id = aws_security_group.instance-sg.id
  source_security_group_id = aws_security_group.alb-sg.id
}

resource "aws_security_group_rule" "outbound-instance" {
  type            = "egress"
  from_port       = 0
  to_port         = 0
  protocol        = "-1"
  cidr_blocks     = ["0.0.0.0/0"]
  security_group_id = aws_security_group.instance-sg.id
}
```

Рисунок 3.22 – Створення групи безпеки для серверів та правил для неї

Після створення необхідних груп безпеки створимо шаблон, за яким створюватимуться сервери. В ньому вказуємо префікс імені з яким створюватимуться сервери, образ операційної системи (в даному випадку – Linux), роль для сервера, яку було створено раніше, тип сервера, відповідну групу безпеки та скрипт, який виконається при першому запуску сервера. Даний скрипт встановлюватиме програмне забезпечення веб-серверів та дозволить при запиті до нього виводити на екран ім'я сервера, до якого було здійснено запит. Фрагмент коду, який відповідає за створення шаблону запуску серверів наведено на рисунку 3.23. Вміст скрипта, який виконуватиметься на серверах при першому запуску наведено на рисунку 3.24.

```
//Launch template for servers
resource "aws_launch_template" "server-template" {
  name_prefix      = "server"
  image_id         = "ami-018ba43095ff50d08"
  iam_instance_profile {name = "server-role"}
  instance_type    = "t2.micro"
  vpc_security_group_ids = ["${aws_security_group.instance-sg.id}"]
  user_data        = filebase64("install.sh")
}
```

Рисунок 3.23 – Створення шаблону запуску серверів

```
$ install.sh
1  #!/bin/bash
2  sudo yum update -y
3  sudo yum install -y httpd
4  sudo touch /var/www/html/index.html
5  sudo echo "<h1 style='text-align: center;'>This machine is: " > /var/www/html/index.html
6  sudo hostname -f >> /var/www/html/index.html
7  sudo echo "</h1>" >> /var/www/html/index.html
8
9  sudo systemctl start httpd
10 sudo systemctl enable httpd
11
```

Рисунок 3.24 – Вміст скрипта

Наступним елементом серверної групи, який ми створимо буде група призначення (Target Group), у якій знаходитимуться сервери. Для її створення вказуємо ім'я, порт, протокол, тип призначення та здійснюємо прив'язку до VPC по ідентифікатору. Відповідний фрагмент коду, який відповідає за створення групи призначення наведений на рисунку 3.25.

```
//Target group for ALB
resource "aws_lb_target_group" "servers-tg" {
  name      = "processing-servers"
  port      = 80
  protocol  = "HTTP"
  target_type = "instance"
  vpc_id    = aws_vpc.main-system-vpc.id
}
```

Рисунок 3.25 – Створення групи призначення (Target Group)

Наступним важливим елементом, який буде створено є Auto Scaling Group (ASG). Для його створення необхідно вказати потрібну, мінімальну та максимальну кількість робочих серверів, ідентифікатори підмереж у яких їх буде створено, а також ідентифікатор створеного вище шаблону серверів. Фрагмент коду, який відповідає за створення ASG наведений на рисунку 3.26.

Також для правильної роботи ALB створені раніше групи призначення та ASG необхідно поєднати за допомогою відповідних ідентифікаторів. Код скрипта, який відповідає за це наведений на рисунку 3.27.

```
# //Auto Scaling Group
resource "aws_autoscaling_group" "servers-autoscaling-group" {
  desired_capacity = 2
  max_size        = 4
  min_size        = 2
  vpc_zone_identifier = [aws_subnet.servers-private-subnet-1.id, aws_subnet.servers-private-subnet-2.id]
  launch_template {
    id      = aws_launch_template.server-template.id
    version = "$Latest"
  }
}
```

Рисунок 3.26 – Створення Auto Scaling Group (ASG)

```
//ASG attachment to target group
resource "aws_autoscaling_attachment" "asg-attach" {
  autoscaling_group_name = aws_autoscaling_group.servers-autoscaling-group.id
  lb_target_group_arn    = aws_lb_target_group.servers-tg.arn
}
```

Рисунок 3.27 – Поєднання групи призначення та ASG

Важливим елементом серверної групи є Application Load Balancer (ALB), елемент який розподілятиме навантаження між серверами. Для його створення необхідно вказати його ім'я, чи є він внутрішнім, його тип, групу безпеки, публічні підмережі, в межах яких він працюватиме, міжзональне балансування та захист від видалення. Відповідний фрагмент коду, який відповідає за створення ALB наведений на рисунку 3.28.

```
//Application Load Balancer
resource "aws_lb" "main-alb" {
  name          = "main-alb"
  internal      = false
  load_balancer_type = "application"
  security_groups = [aws_security_group.alb-sg.id]
  subnets      = [aws_subnet.alb-public-subnet-1.id, aws_subnet.alb-public-subnet-2.id]
  enable_cross_zone_load_balancing = "true"
  enable_deletion_protection = "false"
  tags = {
    Name = "Main load balancer"
  }
}
```

Рисунок 3.28 – Створення Application Load Balancer (ALB)

Також важливо додати групу призначення для відслідковування за допомогою ALB. Додавання в даному випадку відбуватиметься за ARN (Amazon Resource Name). Відповідний програмний код наведений на рисунку 3.29.

```
resource "aws_lb_listener" "main-alb" {
  load_balancer_arn = aws_lb.main-alb.arn
  port              = "80"
  protocol          = "HTTP"
  default_action {
    type             = "forward"
    target_group_arn = aws_lb_target_group.servers-tg.arn
  }
}
```

Рисунок 3.29 – Поєднання групи призначення та Application Load Balancer

Останнім важливим елементом серверної групи, який забезпечить доступ серверів до сховища даних є Gateway Endpoint. Для його створення потрібно вказати ідентифікатор хмарної мережі, ім'я сервісу, до якого здійснюватиметься доступ, тип та відповідну таблицю маршрутизації. Відповідний фрагмент коду наведений на рисунку 3.30.

```
//VPC endpoint
resource "aws_vpc_endpoint" "s3-endpoint" {
  vpc_id          = aws_vpc.main-system-vpc.id
  service_name    = "com.amazonaws.us-east-1.s3"
  vpc_endpoint_type = "Gateway"
  route_table_ids = [aws_route_table.nat-rt.id]
}
```

Рисунок 3.30 – Створення Gateway Endpoint

Додамо описані вище фрагменти до скрипта загального налаштування мережі та запустимо його. Та перейдемо до веб-інтерфейсу AWS, щоб переконатися, що всі описані ресурси було створено.

Переглянемо створені групи безпеки, рисунок 3.31.

	Name	Security group ID	Security group name	VPC ID	Descr
☐	Instance security group	sg-09ea67425c356c301	terraform-202312231725153799000...	vpc-0ed027760316a5a77	Mana
☐	ALB security group	sg-00c5639a2e81f674a	terraform-202312231725153799000...	vpc-0ed027760316a5a77	Mana

Рисунок 3.31 – Створені групи безпеки

Наступним кроком переглянемо шаблони запуску, рисунок 3.32, та список запущених серверів, рисунок 3.33.

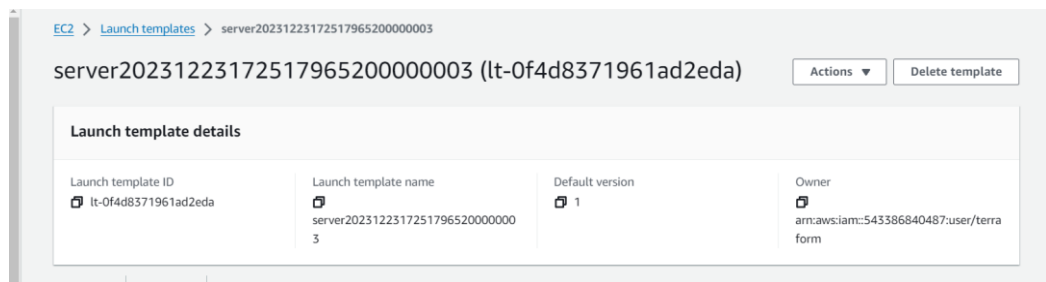


Рисунок 3.32 – Створений шаблон запуску серверів

Instances (2) Info							
Find Instance by attribute or tag (case-sensitive)							
☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☐		i-04df2a08190b7fd79	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a
☐		i-0637067434efd2a04	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b

Рисунок 3.33 – Список запущених серверів

Далі розглянемо групу призначення, рисунок 3.34, та Auto Scaling Group, рисунок 3.35.

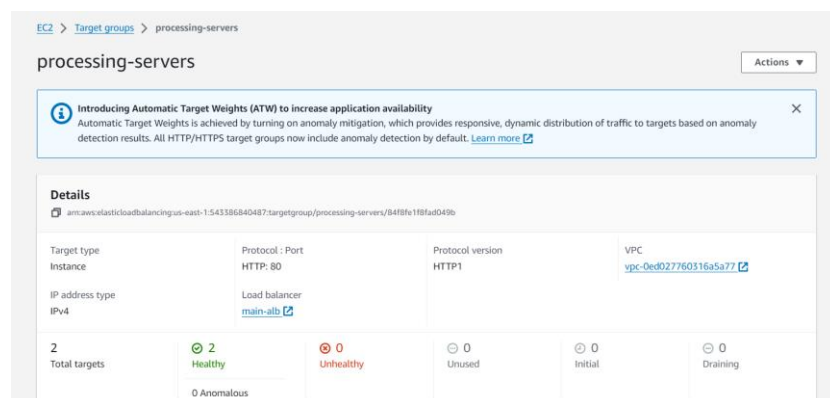


Рисунок 3.34 – Створена група призначення

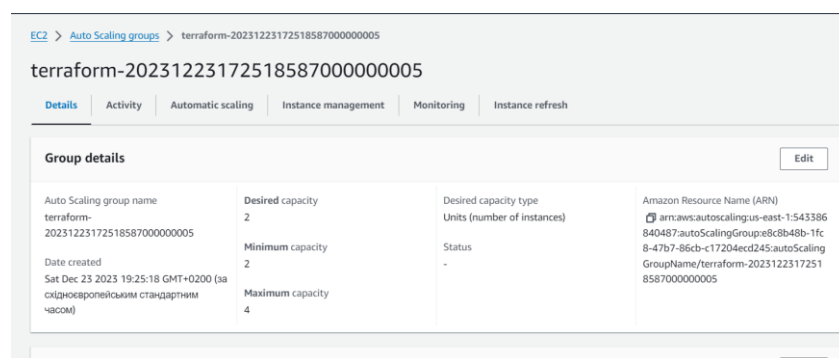


Рисунок 3.35 – Створена Auto Scaling Group

Далі перевіримо Application Load Balancer, а саме його стан та налаштування, рисунок 3.36, та приклад роботи, рисунок 3.37.

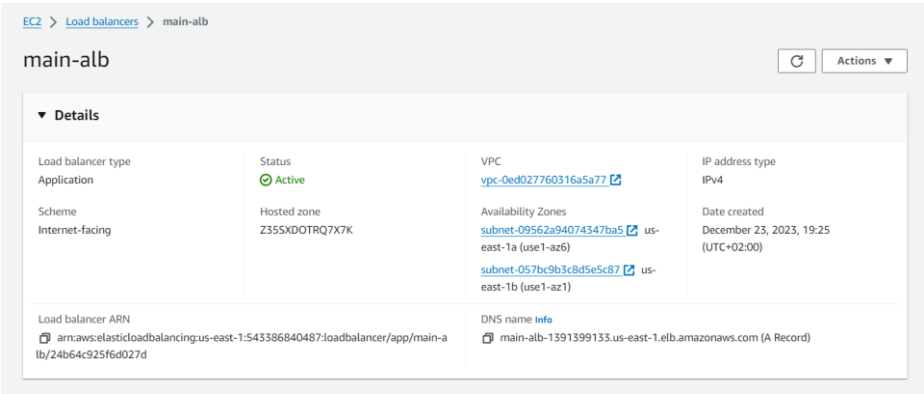


Рисунок 3.36 – Створений Application Load Balancer

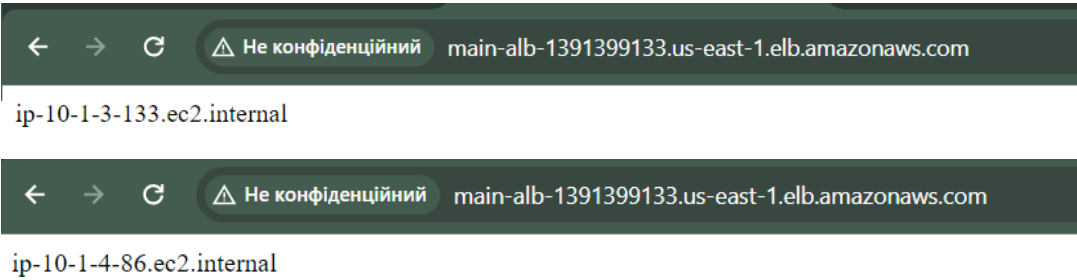


Рисунок 3.37 – Результат роботи Application Load Balancer

Як бачимо при оновленні сторінки адреса сервера, на який було надіслано запит змінюється, отже Application Load Balancer налаштовано вірно та він функціонує правильно.

Останнім елементом перевірки буде Gateway Endpoint, рисунок 3.38.

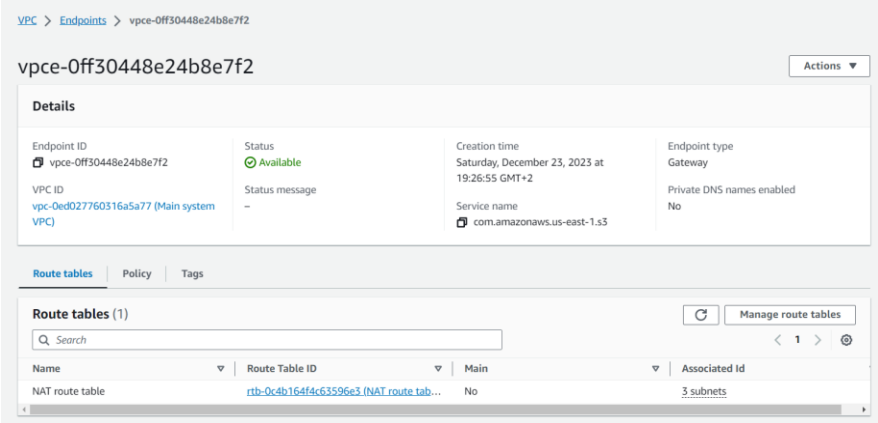


Рисунок 3.38 – Створений Gateway Endpoint

Також перевіримо працездатність даного компонента. Для цього підключимось до одного з серверів та виконаємо команду, яка відображає список всіх S3 сховищ, рисунок 3.39.

```
sh-4.2$ aws s3 ls
2023-10-29 10:21:51 codepipeline-us-east-1-177327223615
2023-10-03 06:05:05 elasticbeanstalk-us-west-1-543386840487
2023-10-29 10:34:41 my-website-prod-3444
2023-10-29 10:33:56 my-website-source-3444
sh-4.2$
```

Рисунок 3.40 – Результат виконання команди `aws s3 ls` на сервері

Як бачимо на екрані з'явився список сховищ, отже Gateway Endpoint налаштовано правильно. Також всі описані в скрипті сервіси створились із вказаними налаштуваннями без помилок, отже налаштування серверної групи завершено.

3.3. Налаштування сховища даних

Перейдемо до налаштування сховища даних. Для цього створимо скрипт, який буде створювати необхідні для зберігання файлів сховища (бакети). Загалом буде створено 4 таких компоненти: для зберігання резервних копій, медичних текстів, зображень та даних для машинного навчання. Всередині деяких бакетів буде створено відповідні папки для зручного сортування даних. Програмний код, який створює необхідні сховища даних наведений на рисунках 3.41-3.42.

```
//S3 buckets
resource "aws_s3_bucket" "backups-bucket" {
  bucket = "backup-bucket-aom"

  tags = {
    Name = "Bucket for backups."
  }
}
resource "aws_s3_bucket" "sagemaker-files" {
  bucket = "sagemaker-bucket-aom"
  tags = {
    Name = "Bucket for SageMaker usage."
  }
}
resource "aws_s3_object" "train-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key    = "train/"
}
```

Рисунок 3.41– Створення сховищ даних

```

resource "aws_s3_object" "test-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key    = "test/"
}
resource "aws_s3_object" "result-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key    = "result/"
}
resource "aws_s3_bucket" "healthimaging-files" {
  bucket = "healthimaging-bucket-aom"
  tags = {
    Name = "Bucket for HealthImaging usage."
  }
}
resource "aws_s3_bucket" "comprehendmed-files" {
  bucket = "comprehendmedical-bucket-aom"
  tags = {
    Name = "Bucket for ComprehendMedical usage."
  }
}
resource "aws_s3_object" "output-cm-folder" {
  bucket = aws_s3_bucket.comprehendmed-files.id
  key    = "output/"
}

```

Рисунок 3.42– Створення сховищ даних

Даний фрагмент коду є останнім файлі розгортання інфраструктури системи. Повний його код розміщено у додатку А.

Наступні налаштування компонентів проводитимемо з використанням веб-інтерфейсу AWS Console. Для зберігання даних пацієнтів створимо екземпляр Relational Database System (RDS) на базі MySQL. Налаштування екземпляру бази даних показано на рисунках 3.43-3.53.

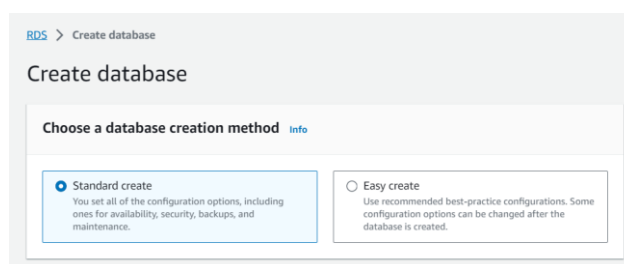


Рисунок 3.43 – Вибір режиму налаштувань

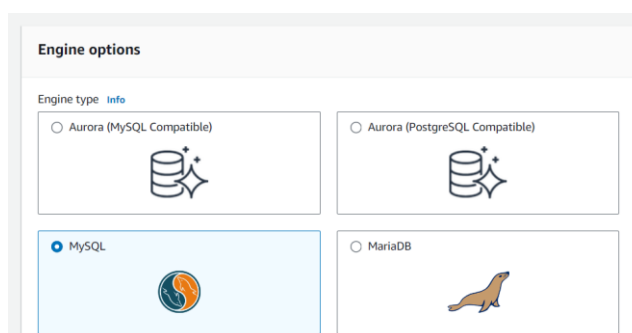


Рисунок 3.44 – Вибір ядра бази даних

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			69

Templates
Choose a sample template to meet your use case.

☐ **Production**
Use defaults for high availability and fast, consistent performance.

☐ **Dev/Test**
This instance is intended for development use outside of a production environment.

☒ **Free tier**
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. [Info](#)

Рисунок 3.45 – Застосування безкоштовного шаблону

Settings

DB instance identifier [Info](#)
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

main-db

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Рисунок 3.46 – Присвоєння ідентифікатора екземплярові бази даних

▼ **Credentials Settings**

Master username [Info](#)
Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

☐ **Manage master credentials in AWS Secrets Manager**
Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.

[Info](#) If you manage the master user credentials in Secrets Manager, some RDS features aren't supported. [Learn more](#)

☐ **Auto generate a password**
Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)
Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), ' (single quote), " (double quote) and @ (at sign).

Confirm master password [Info](#)

Рисунок 3.47 – Створення облікових даних для підключення

Instance configuration
The DB instance configuration options below are limited to those supported by the engine that you selected above.

DB instance class [Info](#)

▼ **Hide filters**

☐ **Show instance classes that support Amazon RDS Optimized Writes** [Info](#)
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

☐ **Include previous generation classes**

☒ **Standard classes (includes m classes)**

☐ **Memory optimized classes (includes r and x classes)**

☐ **Burstable classes (includes t classes)**

db.t3.micro
2 vCPUs 1 GiB RAM Network: 2.085 Mbps

Рисунок 3.48 – Вибір класу екземпляра

		Андрусенко О.М.			Кім.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			70

▼ **Additional configuration**
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name [Info](#)

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group [Info](#)

Option group [Info](#)

Backup

☒ **Enable automated backups**
Creates a point-in-time snapshot of your database

Рисунок 3.52 – Налаштування бази даних за замовчуванням

Backup retention period [Info](#)
The number of days (1-35) for which automatic backups are kept.
 day

Backup window [Info](#)
The daily time range (in UTC) during which RDS takes automated backups.
☒ Choose a window
☐ No preference

Start time: : UTC
Duration: hours

☒ Copy tags to snapshots

Backup replication [Info](#)
☐ Enable replication in another AWS Region
Enabling replication automatically creates backups of your DB instance in the selected Region, for disaster recovery, in addition to the current Region.

Рисунок 3.53 – Налаштування резервного копіювання

Після запуску створеного скрипта та налаштування бази даних перейдемо до веб-інтерфейсу AWS, щоб переконатися, що всі необхідні елементи було створено.

Переглянемо список створених сховищ даних (бакетів), рисунок 3.54.

General purpose buckets (8) Info				
Buckets are containers for data stored in S3. Learn more				
aom 4 matches				
Name	AWS Region	Access	Creation date	
backup-bucket-aom	US East (N. Virginia) us-east-1	Bucket and objects not public	December 24, 2023, 12:45:02 (UTC+02:00)	
comprehendmedical-bucket-aom	US East (N. Virginia) us-east-1	Bucket and objects not public	December 24, 2023, 12:45:02 (UTC+02:00)	
healthimaging-bucket-aom	US East (N. Virginia) us-east-1	Bucket and objects not public	December 24, 2023, 12:45:02 (UTC+02:00)	
sagemaker-bucket-aom	US East (N. Virginia) us-east-1	Bucket and objects not public	December 24, 2023, 12:45:02 (UTC+02:00)	

Рисунок 3.54 – Створені сховища даних

Також перевіримо створену базу даних, рисунок 3.55, та підключимось до неї за допомогою одного з серверів, рисунок 3.56.

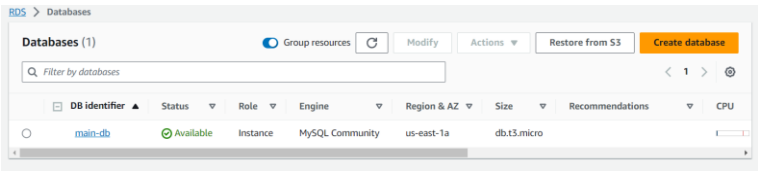


Рисунок 3.55 – Створений екземпляр бази даних

```
[root@ip-10-1-3-149 ~]# mysql --host=main-db.cawidt74do9u.us-east-1.rds.amazonaws.com --user=admin --password=123321123
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Рисунок 3.56 – Підключення до створеної бази даних

Як бачимо всі сховища даних створені, база даних налаштована правильно та має зв’язок з серверами, отже всі налаштування виконані вірною.

3.4. Налаштування сервісів AWS для обробки та аналізу даних

Після налаштування інфраструктури системи перейдемо до налаштування серверів для обробки та аналізу даних. Пепшим сервісо, який ми налаштуємо буде Amazon SageMaker. Для того, щоб почати роботу з ним необхідно створити SageMaker домен, рисунок 3.57, та користувача в ньому, рисунок 3.58.

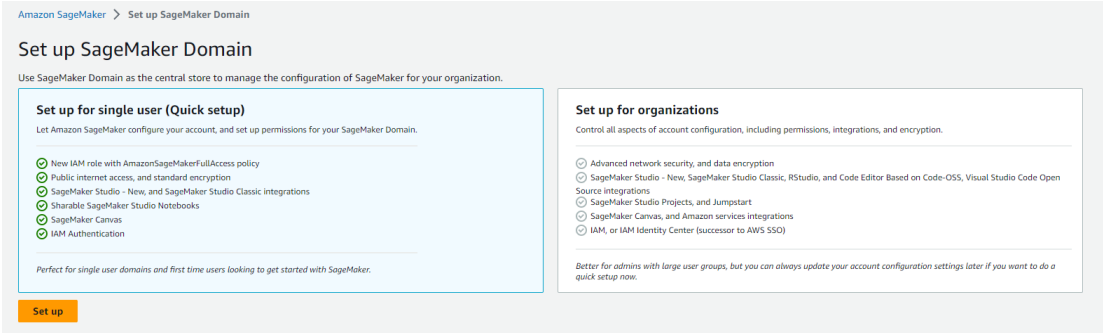


Рисунок 3.57 – Створення домену SageMaker

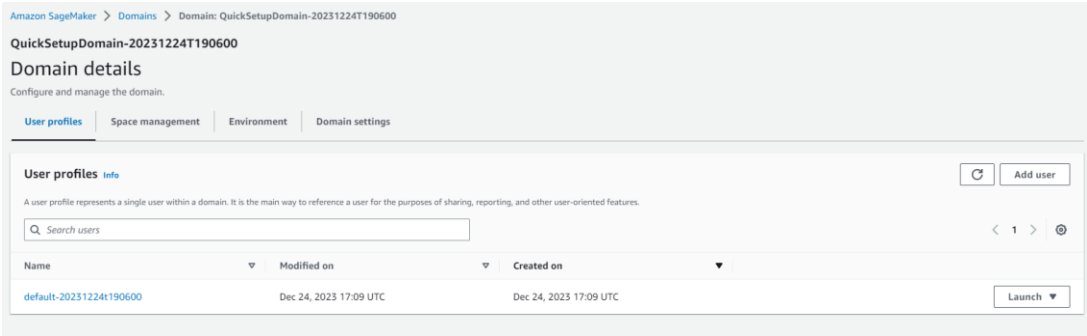


Рисунок 3.58 – Створений користувач домену

Наступним та останнім кроком підготовки сервісу до роботи є запуск SageMaker Studio та елементу Canva в ньому, рисунок 3.59.

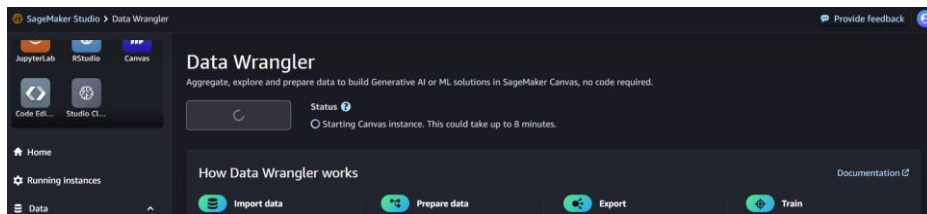


Рисунок 3.59 – Запуск Canva в SageMaker Studio

Після запуску Canva сервіс буде готовий до роботи.

Наступним сервісом, який буде налаштовано стане HealthImaging. Для початку роботи з даним сервісом необхідно створити спеціальне сховище даних (Data Store), рисунок 3.60.

Рисунок 3.60 – Створення сховища даних для HealthImaging

Після створення сховища необхідно імпортувати дані до нього з сервісу S3, який було створено раніше, рисунок 3.61. Також для правильної роботи сервісу потрібно створити для нього відповідну IAM-роль, рисунок 3.62.

Рисунок 3.61 – Імпорт DICOM-даних до сховища

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			74

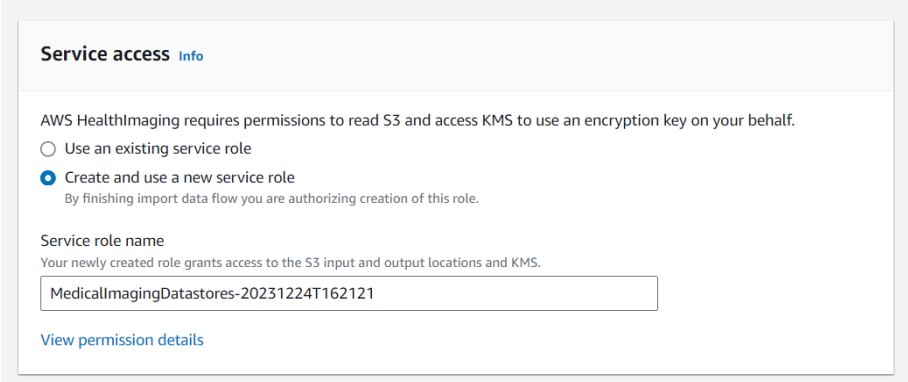


Рисунок 3.62 – Створення ІАМ-ролі для сервісу

Після успішного виконання імпорту на екрані з'явиться відповідне повідомлення, рисунок 3.63, а також імпортовані дані, у згрупованому вигляді, з'являться у сховищі, рисунок 3.64.

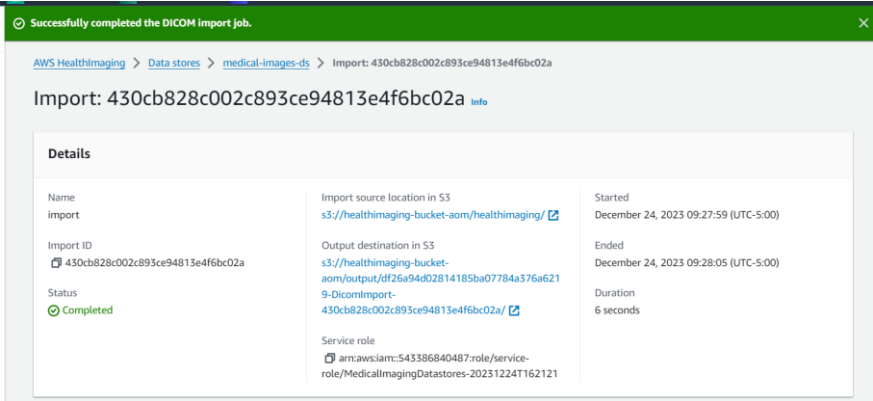


Рисунок 3.63 – Повідомлення про успішне імпортування файлів

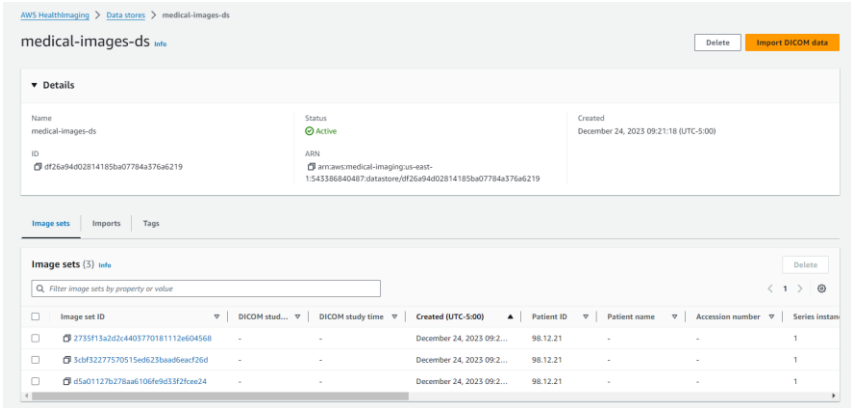


Рисунок 3.64 – Імпортовані файли у сховищі

Останнім необхідним для роботи системи сервісом є ComprehendMedical, який не потребує налаштувань та є готовим до роботи за замовчуванням. На цьому налаштування усіх необхідних сервісів завершено.

3.5 Тестування системи

Після проведення всіх необхідних налаштувань перейдемо до тестування роботи нашої системи. Оскільки створення веб-інтерфейсу не є завданням даної роботи, то її тестування проведемо за допомогою демонстрації основних функцій використовуючи командний рядок, деякі скрипти та AWS Management Console.

Протестуємо можливість доступу серверів до сховища даних. Для початку отримаємо список всіх доступних бакетів за допомогою команди `aws s3 ls`, рисунок 3.65.

```
[ssm-user@ip-10-1-3-44 bin]$ aws s3 ls
2023-12-25 07:56:56 backup-bucket-aom
2023-12-25 07:56:56 comprehendmedical-bucket-aom
2023-12-25 07:56:56 healthimaging-bucket-aom
2023-12-25 07:56:56 sagemaker-bucket-aom
[ssm-user@ip-10-1-3-44 bin]$
```

Рисунок 3.65 – Результат виконання команди `aws s3 ls`

Як бачимо всі потрібні бакети відобразились на екрані, отже зв'язок з ними встановлено правильно. Наступним кроком протестуємо можливість завантаження та отримання файлів зі сховища. Для цього виконаємо команду `aws s3api put-object` з потрібними параметрами, рисунок 3.66.

```
[ssm-user@ip-10-1-3-44 sagemaker-csv]$ aws s3api put-object --bucket sagemaker-bucket-aom --key patients.csv --body patients.csv
{
  "ETag": "\"63ce3714a5785221a0e3876a18e94940\"",
  "ServerSideEncryption": "AES256"
}
[ssm-user@ip-10-1-3-44 sagemaker-csv]$
```

Рисунок 3.66 – Результат завантаження файлу

Переглянемо наш бакет зі допомогою AWS Console та переконаємось, що його справді було завантажено правильно, рисунок 3.67.

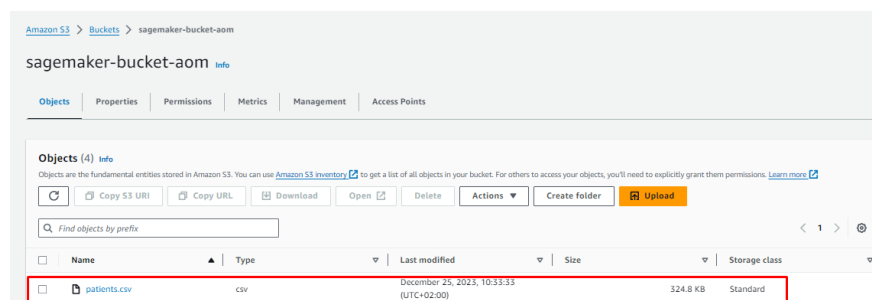


Рисунок 3.67 – Завантажений у бакет файл

Також перевіримо можливість отримання файлів зі сховища. Для цього скористаємось командою `aws s3api get-object` з необхідними параметрами, рисунок 3.68.

```
[s3m-user@ip-10-1-3-44 sagemaker-csv]$ sudo aws s3api get-object --bucket sagemaker-bucket-aom --key patients.csv s3patients.csv
{
  "AcceptRanges": "bytes",
  "ContentType": "binary/octet-stream",
  "LastModified": "Mon, 25 Dec 2023 08:33:33 GMT",
  "ContentLength": 332570,
  "ETag": "\"63ce3714e5785221a0e3876a18e94940\"",
  "ServerSideEncryption": "AES256",
  "Metadata": {}
}
```

Рисунок 3.68 – Результат виконання команди `aws s3api get-object`

```
[s3m-user@ip-10-1-3-44 sagemaker-csv]$ cat s3patients.csv
id,BIRTHDATE,DEATHDATE,SSN,DRIVERS,PASSPORT,PREFIX,FIRST,SUFFIX,MAIDEN,MARITAL,RACE,ETHNICITY,GENDER,BIRTHPLACE,ADDRESS,CITY,STATE,COUNTY,ZIP,LAT,LON,HEALTHCARE_EXPENSES,HEALTHCARE_COVERAGE
1d604da9-9a81-4ba9-80c2-de3375d59b40,1989-05-25,,999-76-6866,999984236,X19277260X,Mr.,José Eduardo181,Gómez206,,M,white,hispanic,M,Marigot Saint Andrew Parish DM,427 Ballistreri Way Unit 19,Chicopee,Massachusetts,Hampden County,01013,42.22835382315942,-72.56295055096882,271227.08,1334.8799999999999
034e9e3b-2def-4559-bb2a-7850888ae060,1983-11-14,,999-73-5361,999962402,X88275464X,Mr.,Milo271,Fell794,,M,white,nonhispanic,M,Danvers Massachusetts US,422 Parrell Path Unit C9,Somerville,Massachusetts,Middlesex County,02149,42.36069732777065,-71.12653095133149,793946.01,3204.49
1039b10-3cd1-4ac3-ac13-ec26728cb592,1992-06-02,,999-27-3989,999972692,X79754411X,Mr.,Dayson808,Fade1536,,M,white,nonhispanic,M,Springfield Massachusetts US,1056 Harris Lane Suite 70,Chicopee,Massachusetts,Hampden County,01020,42.181642300277424,-72.60884222581657,574111.9,2606.3999999999996
8d4c4326-e9de-4f45-9a4c-f8c36bfff89ae,1978-05-27,,999-85-4926,999974448,X40915583X,Mrs.,Mariana775,Rutherford999,,Williamson769,M,white,nonhispanic,F,Yarmouth Massachusetts US,999 Kuhn Forge,Lowell,Massachusetts,Middlesex County,01851,42.63614335069588,-71.3432549217789,935630.3,8756.189999999999
f5dcd418-09fe-4a2f-baa0-3da80bd8c3a,1996-10-18,,999-60-7372,999915787,X86772562X,Mr.,Gregorio366,Auer97,,M,white,nonhispanic,M,Patras Achaea GR,1050 Lindgren Extension Apt 30,Boston,Massachusetts,Suffolk County,02135,42.352493426303177,-71.02861042115062,599763.07,3772.2
72c0b9ce-fa44-490b-eaff-bd0ce7846e55,2017-07-27,,999-68-6630,,Jacintho644,Kris249,,M,white,nonhispanic,M,Beverly Massachusetts US,888 Hickley Ferry Suite 3 Springfield Massachusetts,Chicopee,Massachusetts,Hampden County,01013,42.22835382315942,-72.56295055096882,271227.08,1334.8799999999999
```

Рисунок 3.69 – Вміст завантаженого файлу

Як бачимо файл було отримано успішно та в непошкодженому вигляді, отже функція отримання файлів теж працює вірно.

Наступним кроком перевіримо роботу з базою даних. Для цього встановимо з'єднання з нею, рисунок 3.70, та виконаємо скрипт, який створить структуру медичної бази даних, рисунки 3.71-3.72.

```
[s3m-user@ip-10-1-3-44 sagemaker-csv]$ mysql --host=main-db.cawid74do9u.us-east-1.rds.amazonaws.com --user=admin --password=123321123
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 30
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

Рисунок 3.70 – З'єднання з базою даних

```
[s3m-user@ip-10-1-3-44 sagemaker-csv]$ mysql --host=main-db.cawid74do9u.us-east-1.rds.amazonaws.com --user=admin --password=123321123 patient_data < create_tables.sql
```

Рисунок 3.71 – Виконання скрипта створення структури бази даних

```
MySQL [patient_data]> show tables
-> ;
+-----+
| Tables_in_patient_data |
+-----+
| Doctors                 |
| MedicalProcedures      |
| MedicalRecords          |
| MedicalTests            |
| PatientProcedures      |
| PatientTestResults      |
| Patients                |
+-----+
7 rows in set (0.00 sec)

MySQL [patient_data]>
```

Рисунок 3.72 – Створена структура бази даних

Перевіримо можливість запису даних в базу. Для цього виконаємо команди, зображені на рисунку 3.73 та переглянемо результат додавання, рисунок 3.74.

```
MySQL [patient_data]> INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Address)
-> VALUES
-> (1, 'John', 'Doe', '1980-01-15', 'M', '123 Main St'),
-> (2, 'Jane', 'Smith', '1990-05-20', 'F', '456 Oak Ave'),
-> (3, 'Robert', 'Johnson', '1975-08-10', 'M', '789 Pine Rd');
Query OK, 3 rows affected (0.00 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Рисунок 3.73 – Додавання нових записів у базу даних

```
MySQL [patient_data]> select * from Patients;
+-----+-----+-----+-----+-----+-----+
| PatientID | FirstName | LastName | DateOfBirth | Gender | Address |
+-----+-----+-----+-----+-----+-----+
| 1 | John | Doe | 1980-01-15 | M | 123 Main St |
| 2 | Jane | Smith | 1990-05-20 | F | 456 Oak Ave |
| 3 | Robert | Johnson | 1975-08-10 | M | 789 Pine Rd |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

MySQL [patient_data]> 
```

Рисунок 3.74 – Створені записи

Як ми можемо спостерігати записи було додано, отже сервер успішно працює з базою даних.

Наступним елементом тестування системи буде сервіс HealthImaging. Для його тестування розгорнемо додаток HealthImaging UI за допомогою сервісу Amplify, рисунок 3.75.

Deploy App

App Information

✓ GitHub authorization was successful.

App name
aws-healthimaging-samples

We will fork the main branch from repository aws-healthimaging-samples App in your GitHub account.

You are connecting an Amplify app repository
An Amplify app contains backend and frontend resources.

Select service role
A service role allows Amplify Hosting to deploy backend resources to AWS along with your frontend.[Info](#)

amplifyconsole-backend-role

📘 Create a new service role. In the window that opens, accept the pre-selected defaults on each screen to create a new service role. [Create new role](#)

Рисунок 3.75 – Розгортання додатку для роботи з сервісом HealthImaging

Для роботи з додатком проходимо просту реєстрацію. Після цього функціонал застосунку стане доступним для використання. В ньому можна переглянути

список всіх сховищ, рисунок 3.76, а також знайти певний набір медичних зображень за його ідентифікатором, рисунок 3.77.

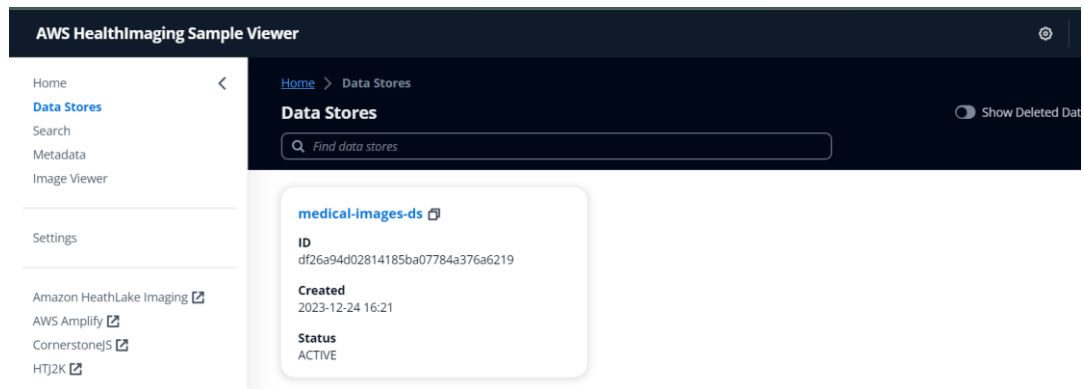


Рисунок 3.76 – Список доступних сховищ наборів зображень

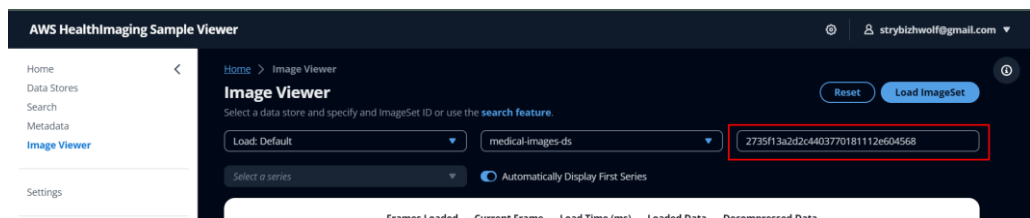


Рисунок 3.77 – Пошук набору зображень за ідентифікатором

Знайдений набір зображень можна переглянути покадрово, а також регулювати яскравість та контрастність певного кадру при перегляді, рисунки 3.78-3.79.

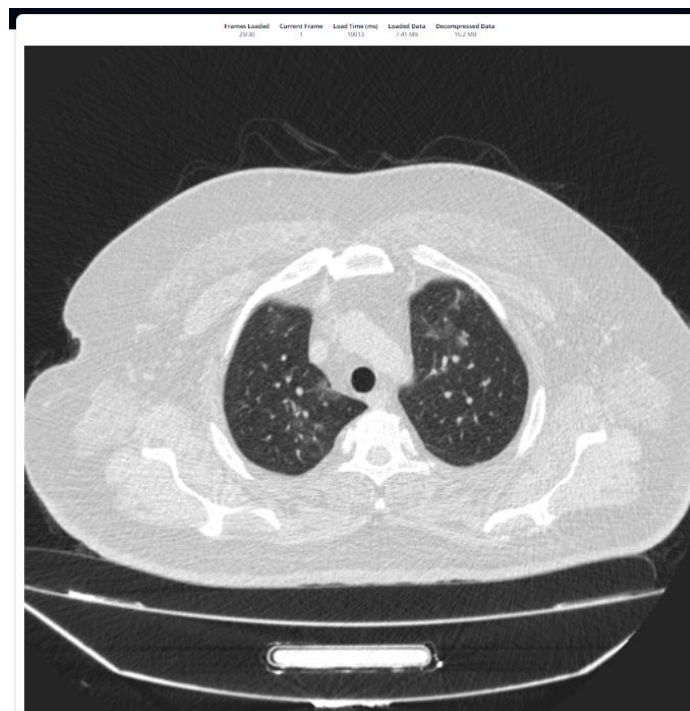


Рисунок 3.78 – Перегляд зображення зі знайденого набору

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			79



Рисунок 3.79 – Перегляд іншого кадру зі зміною його яскравості

Також у даному застосунку можна переглянути метадані бажаного набору, рисунок 3.80.

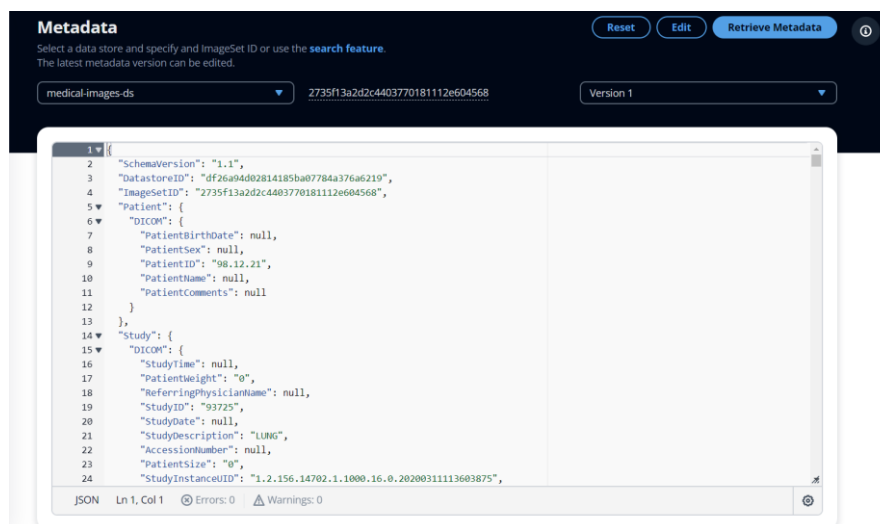


Рисунок 3.80 – Метадані набору зображень

Отже, даний сервіс у парі із розгорнутим застосунком дозволить нам зручно та швидко отримувати набори зображень та обробляти їх, корегуючи у разі потреби відповідні кадри.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			80

Наступним сервісом, який ми протестуємо буде Comprehend Medical. Для тестування його роботи створимо скрипт на мові Python, який отримуватиме текст, відправлятиме його на обробку в даний сервіс, а відповідь у форматі JSON записуватиме у відповідне сховище даних. Код скрипта наведений на рисунку 3.81.

```
import boto3
from datetime import datetime
import json

> text='' ...
client = boto3.client(aws_access_key_id="AKIA5BDBFWT5D7WYCHD",
                     aws_secret_access_key="n6vhpX/H+Z55kLwt1fluT38nFP+07y9zH1FAharH",
                     service_name='comprehendmedical',
                     region_name="us-east-1")
result = client.detect_entities(Text=text)
result = json.dumps(result)

filename=str(f'{datetime.now().timestamp()}.json')
print("File name: ", filename, ". Upload started.")
with open(filename, 'a') as fs:
    fs.write(result)

s3client = boto3.client(aws_access_key_id="AKIA5BDBFWT5D7WYCHD",
                       aws_secret_access_key="n6vhpX/H+Z55kLwt1fluT38nFP+07y9zH1FAharH",
                       service_name='s3',
                       region_name="us-east-1")

s3client.upload_file(filename, 'comprehendmedical-bucket-aom', filename)

print("Upload done.")
```

Рисунок 3.81 – Код скрипта для роботи з Comprehend Medical

Після виконання скрипта результат обробки тексту у форматі JSON з'явиться у відповідному сховищі даних, рисунок 3.82.

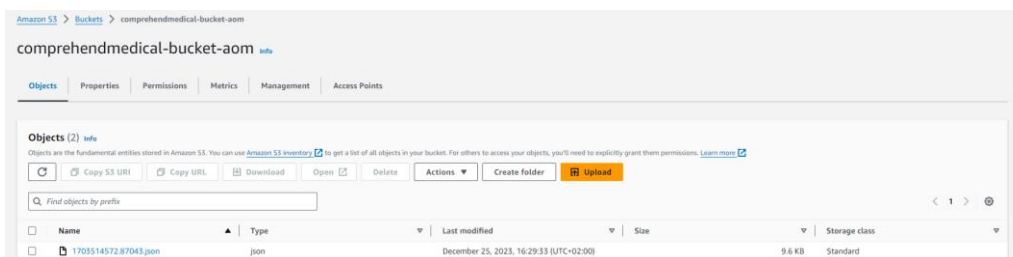


Рисунок 3.82 – Завантажений файл з результатами обробки

Даний файл містить повний результат аналізу введеного тексту і, оскільки формат файлу JSON, то отримати інформацію з нього буде доволі просто.

Останнім елементом тестування стане сервіс SageMaker, за допомогою якого будуються моделі машинного навчання для передбачень.

Розглянемо роботу даного інструменту на прикладі обробки зображень КТ легенів. Для цього відповідні зображення завантажимо до сховища даних та

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			81

створимо потік даних (Data Flow) у SageMaker, рисунок 3.83, та імпортуємо завантажені дані, рисунок 3.84.

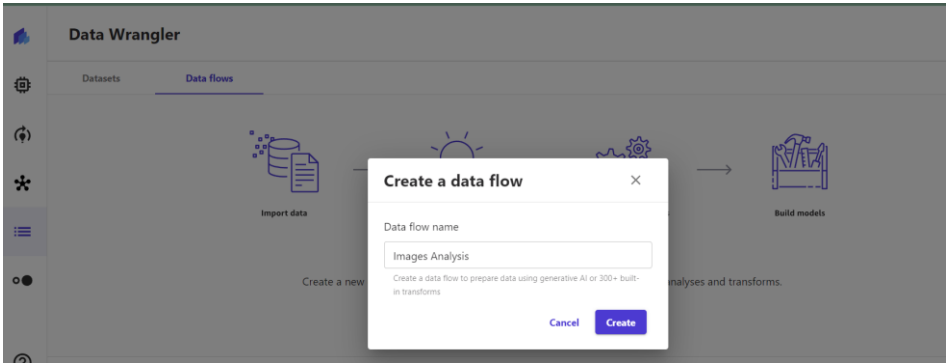


Рисунок 3.83 – Створення потоку даних

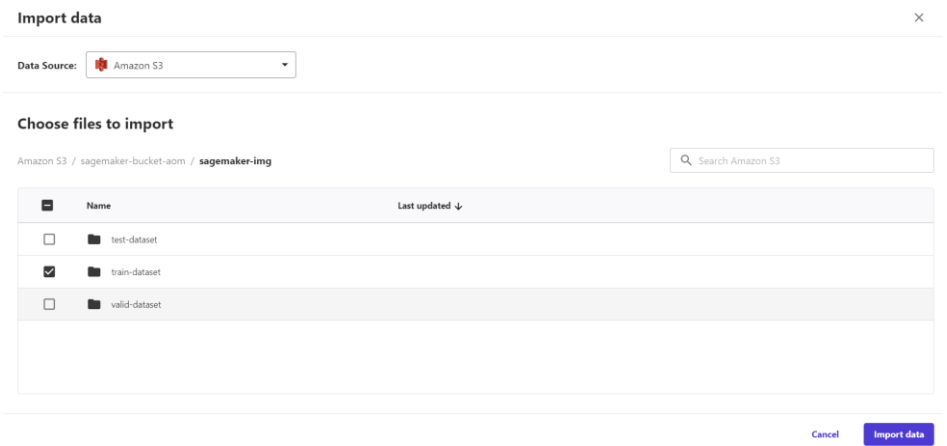


Рисунок 3.84 – Імпорт завантажених даних

Після створення потоку даних необхідно створити модель для його обробки. Процес створення моделі наведений на рисунок 3.85-3.86.

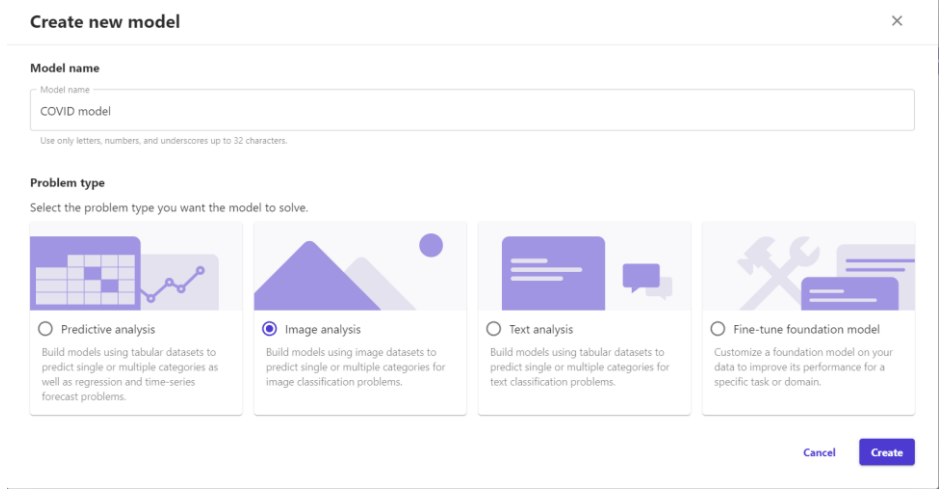


Рисунок 3.85 – Створення моделі

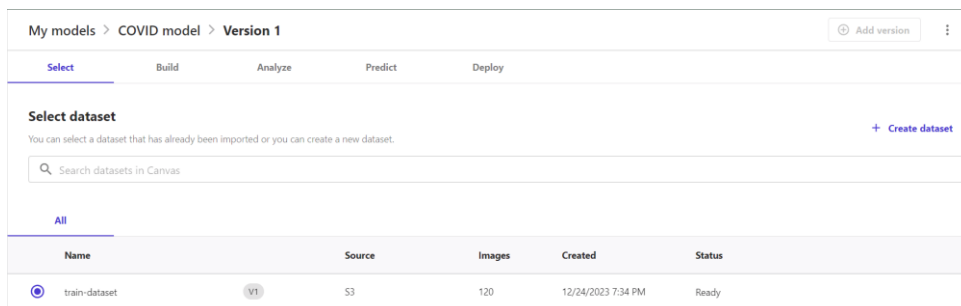


Рисунок 3.86 – Вибір створеного потоку даних

Після завантаження усі зображення надходять без міток (Unlabeled), рисунок 3.87. Для створення моделі створюємо та розподіляємо між зображеннями відповідні мітки, рисунок 3.88.

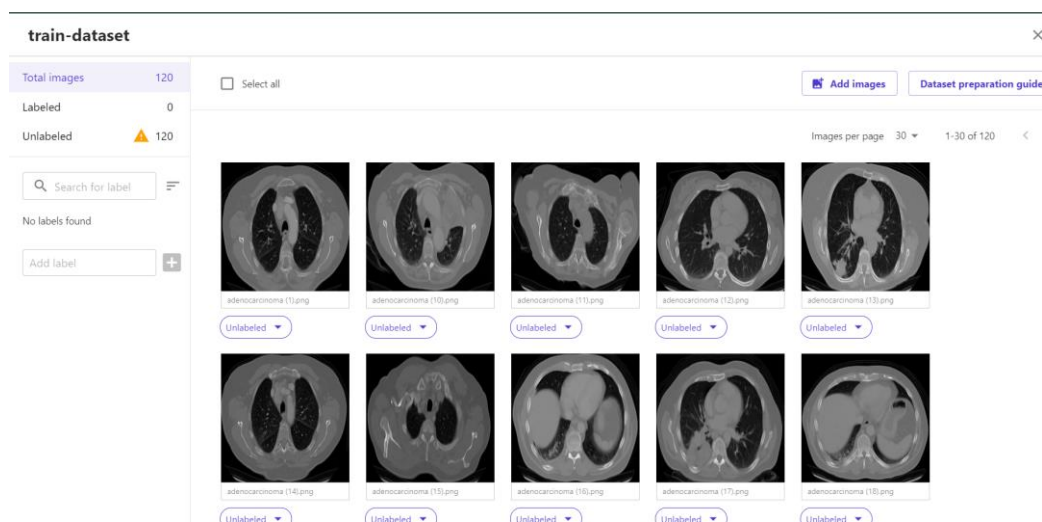


Рисунок 3.87 – Зображення без міток

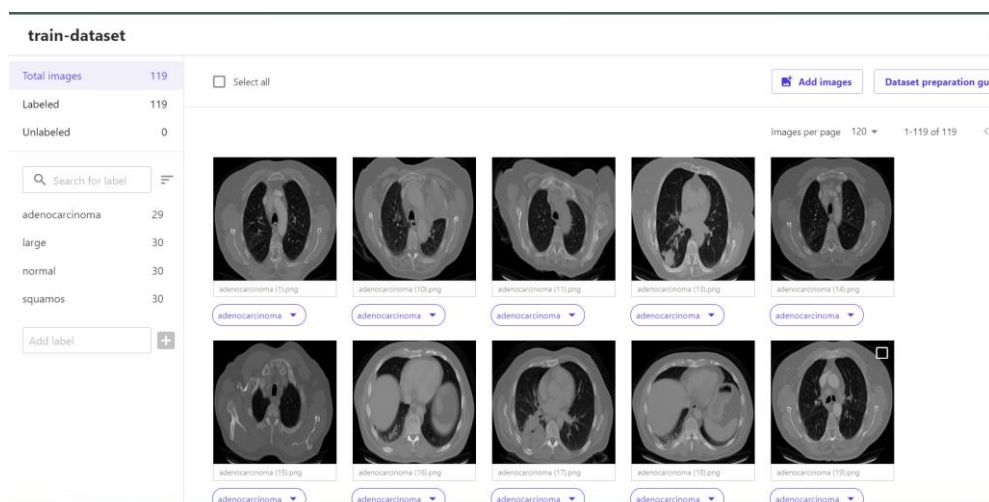


Рисунок 3.88 – Зображення з відповідними мітками

Після того, як усі зображення мають свої мітки можна розпочати побудову моделі, рисунок 3.89.

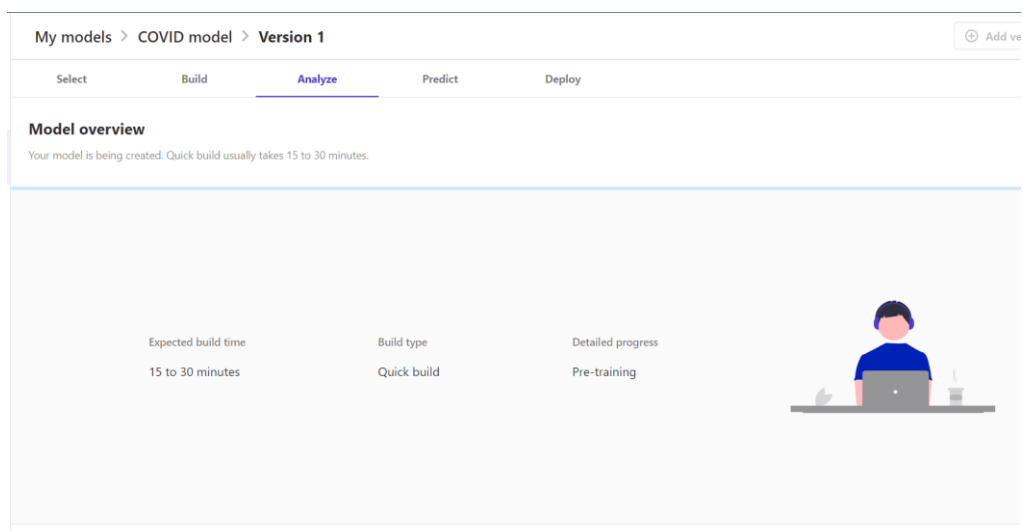


Рисунок 3.89 – Побудова моделі

Після завершення створення моделі на екрані з'явиться результат правильного передбачення зображень цією моделлю. В даному випадку дане значення становить 29,2%, рисунок 3.90.

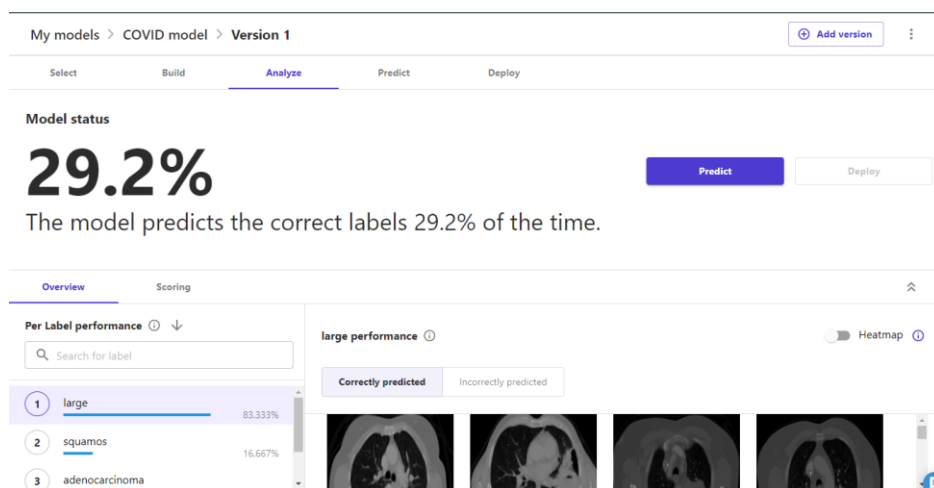


Рисунок 3.90 – Результат передбачень моделі

Імпортуємо зображення із завчасно відомим нам результатом та виконуємо передбачення за допомогою створеної моделі, рисунки 3.91-3.93.

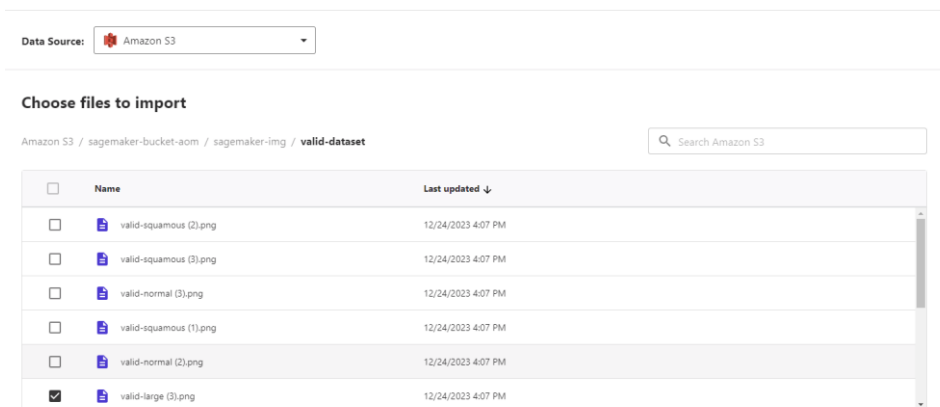


Рисунок 3.91 – Імпорт зображення

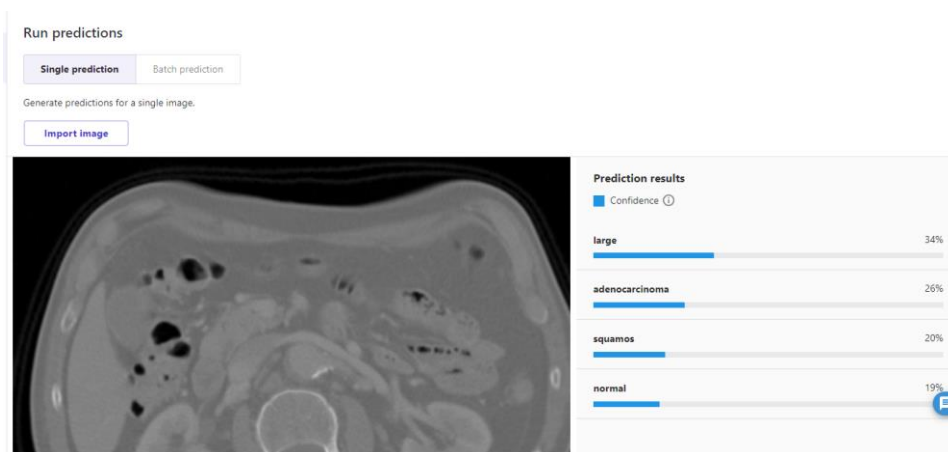


Рисунок 3.92 – Результат передбачення

Як бачимо модель, хоч і не з 100% впевненістю, проте визначила правильну мітку даного зображення. Для збільшення точності визначення необхідно для побудови моделі використовувати більше тестових даних, а також виконати декілька навчальних циклів.

Розглянемо побудову моделі для табличних даних. Для цього створимо нову модель, рисунок 3.93, та оберемо тестовий датасет для неї, рисунок 3.94.

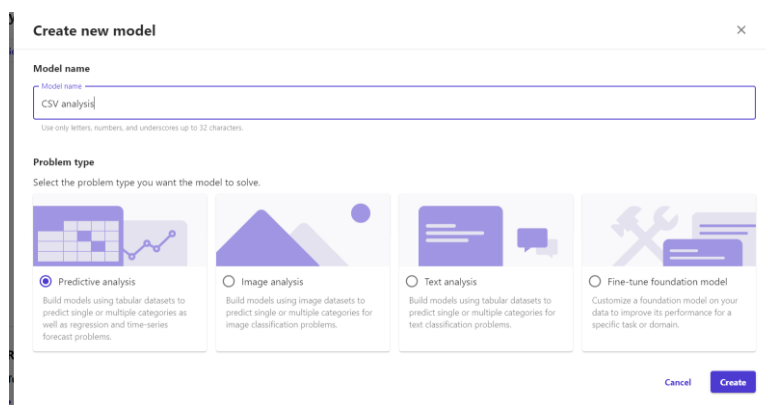


Рисунок 3.93 – Створення нової модель

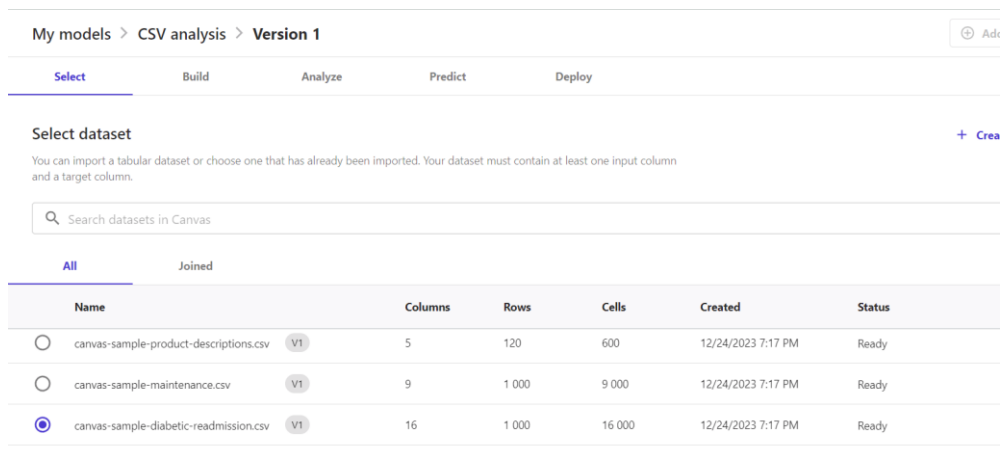


Рисунок 3.94 – Вибір тестового датасету для моделі

Після вибору тестового датасету оберемо колонку призначення (time_in_hospital), рисунок 3.95, та розпочнемо побудову моделі, рисунок 3.96.

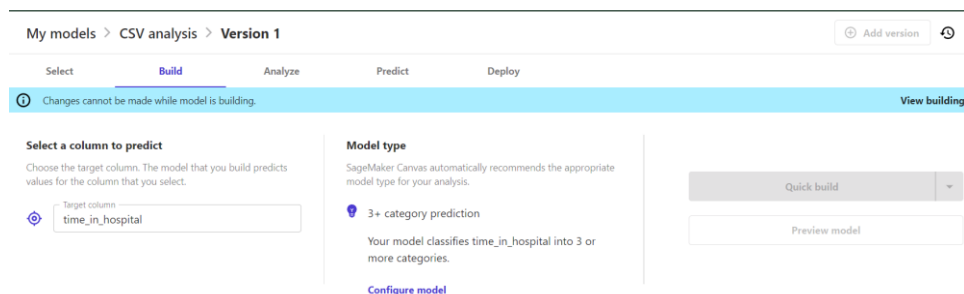


Рисунок 3.95 – Вибір колонки призначення для передбачень

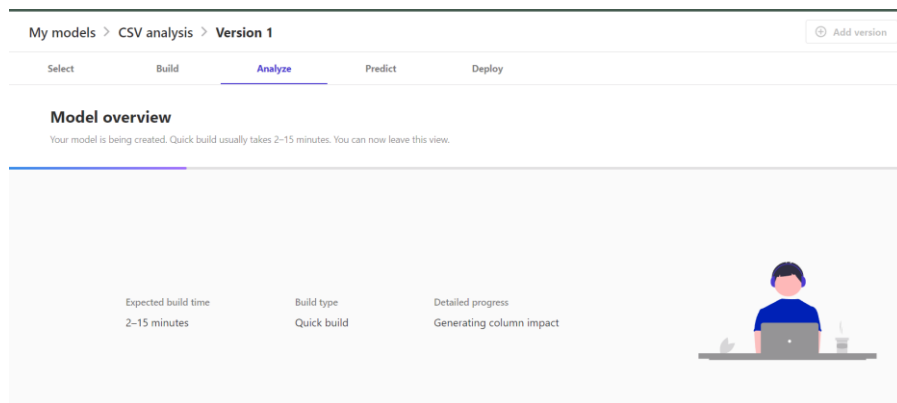


Рисунок 3.96 – Процес побудови моделі

Аналогічно з моделлю обробки зображень дана модель після побудови має невелику точність передбачення (21,845%). Також на сторінці передбачення можна прослідкувати вплив різних факторів на точність передбачень, рисунок 3.97. Також можна детально розглянути всі метрики машинного навчання на вкладці розширених метрик, рисунок 3.98.

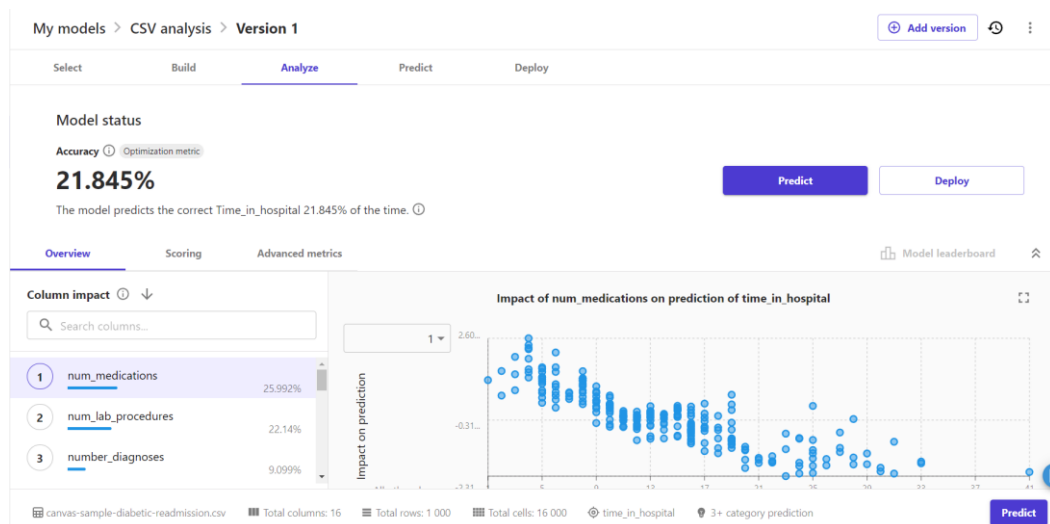


Рисунок 3.97 – Точність передбачення та графік впливу різних факторів на нього

My models > CSV analysis > Version 1				
Accuracy 21.845% The model predicts the correct Time_in_hospital 21.845% of the time. Predict Deploy				
Overview Scoring Advanced metrics Model leaderboard				
Column impact 1 num_medications 25.992% 2 num_lab_procedures 22.14% 3 number_diagnoses 9.099% Impact of num_medications on prediction of time_in_hospital				
canvas-sample-diabetic-readmission.csv Total columns: 16 Total rows: 1 000 Total cells: 16 000 time_in_hospital 3+ category prediction Predict				

Рисунок 3.98 – Метрики моделі

На основі побудованої моделі зробимо передбачення для нового датасету, рисунок 3.99.

Select dataset for predictions

To make predictions on a dataset, select it or import it. The dataset that you select must have the same number of feature columns as the training dataset.

Search datasets in Canvas

Name	Columns	Rows	Cells	Created	Status
☒ canvas-sample-diabetic-readmission.csv V1	16	1 000	16 000	12/25/2023 8:20 PM	Ready

Рисунок 3.99 – Вибір датасету для передбачення

Результатом є таблиця, у якій відображено передбачені значення та його точність, рисунок 3.100.

batchInfer-CSV analysis-canvas-sample-diabetic-readmission.csv-1703529574

Prediction (time_in_hospital)	Probability	readmitted	race	gender	age	num_lab_pro...	num_proced...
3	33.8%	no	caucasian	female	5	41	0
3	66.8%	>30	caucasian	female	15	59	0
2	60.3%	no	african_american	female	25	11	5
3	37.2%	no	caucasian	male	35	44	1
1	38.3%	no	caucasian	male	45	51	0
3	49.3%	>30	caucasian	male	55	31	6
4	36.3%	no	caucasian	male	65	70	1
5	30.5%	>30	caucasian	male	75	73	0

Send to Amazon QuickSight Download

Рисунок 3.100 – Передбачені значення

Покращити точність даної моделі можна додавши для тренування нові унікальні значення та провівши декілька тренувань моделі.

Висновки з розділу 3

В розділі 3 було налаштовано компоненти спроектованої системи. Відповідні налаштування були пояснені та наведені відповідні фрагменти програмного коду, а також продемонстровано основні етапи налаштування системи за допомогою веб-інтерфейсу AWS Management Console.

Систему було реалізовано на базі приватної хмарної мережі, створено необхідні для її роботи елементи, такі як балансувальник навантаження, сервери, сховища даних, реляційна база даних та сервіси обробки та аналізу даних (SageMaker, HealthImaging та ComprehendMedical). Продемонстровано основні функції системи та приклади обробки та аналізу медичних даних засобами штучного інтелекту.

ВИСНОВКИ

У даній роботі було проведено дослідження основних аспектів та принципів хмарних технологій та штучного інтелекту в контексті медичних систем. За допомогою цього дослідження можна зробити наступні висновки.

По-перше, використання хмарних технологій значно полегшує інфраструктурне управління, дозволяючи медичним установам ефективно використовувати ресурси та зосередитися на наданні високоякісної медичної допомоги. Гнучкість та масштабованість хмарних сервісів забезпечують надійну основу для зберігання та обробки великих обсягів медичних даних.

По-друге, впровадження штучного інтелекту в аналіз медичних даних дозволяє здійснювати швидкий та точний аналіз, що може покращити діагностику та розробку індивідуалізованих планів лікування. Методи машинного навчання дозволяють виявляти складні патерни та залежності, що не завжди можуть бути виявлені традиційними методами.

Загалом, використання хмарних технологій та штучного інтелекту в медичних системах демонструє значний потенціал для вдосконалення медичної практики та розширення можливостей діагностики та лікування.

Також в даній роботі було спроектовано та реалізовано на базі хмарного провайдера AWS систему зберігання, обробки та аналізу медичних даних. Вибір компонентів для реалізації було обґрунтовано, наведено їх переваги та недоліки та продемонстровано особливості налаштування.

Розроблена у ході виконання кваліфікаційної роботи система може бути використана у невеликих медичних установах з подальшою можливістю розширення як обчислювальних потужностей, так і функціоналу.

		Андрусенко О.М.			КІМ.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			89

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Райкомар, А., Орен, Е., Чен, К., Дай, А. М., Хайяж, Н., Хардт, М. та ін. (2018). Масштабоване та точне глибоке навчання з використанням електронних медичних записів. NPJ Digital Medicine.
2. Естева, А., Купрел, Б., Новоа, Р. А., Ко, Дж., Светтер, С. М., Блау, Г. М. та ін. (2017). Класифікація раку шкіри на рівні дерматолога за допомогою глибоких нейронних мереж.
3. Бім, А. Л., і Кохейн, І. С. (2018). Великі дані та машинне навчання в охороні здоров'я. JAMA.
4. Раві, Д., Вонг, К., Делігіанні, Ф., Бертелот, М., Андреу-Перес, Дж., Ло, Б., і Янг, Г. З. (2016). Глибоке навчання для інформатики в охороні здоров'я. IEEE Journal of Biomedical and Health Informatics.
5. Тополь, Е. Дж. (2019). Високоєфективна медицина: злиття штучного та штучного інтелекту. Nature Medicine.
6. Шардрен, Г., Ченг, П. М., Воронцов, Е., Дроздзал, М., Туркотт, С., Пал, К. Дж. та ін. (2017). Глибоке навчання: вступ для радіологів. Radiographics.
7. Міотто, Р., Ванг, Ф., Ванг, С., Джіанг, С. І Дадлі, Дж. Т. (2017). Машинне навчання в охороні здоров'я: огляд, можливості та виклики. Briefings in Bioinformatics.
8. Сарія, С., і Голденберг, А. (2015). Субтипуювання: що це таке і яка його роль у точній медицині. IEEE Intelligent Systems.
9. Обермайер, З., і Емануель, Е. Дж. (2016). Прогнозування майбутнього - великі дані, машинне навчання і клінічна медицина. New England Journal of Medicine.
10. Чен, Дж. Х., Аш, С. М., і Мельцер, Д. О. (2017). Модель машинного навчання для прогнозування ризику 30-денної реадмісії у пацієнтів з серцевою недостатністю: ретроспективний аналіз даних електронних медичних записів. BMC Medical Informatics and Decision Making.
11. Ву, Г., Чжу, К. та Фу, Ц. В. (2019). Зрозумілість ШІ для аналізу медичних даних. Computerized Medical Imaging and Graphics.

		Андрусенко О.М.			<i>Кім.КР.Б – 123 – 02 – ПЗ</i>	
		Бродський Ю. Б.				
		№ докум.	Підпис			90

12. Шен, Д., і Бу, Г. (2017). Машинне навчання в аналізі медичних зображень. IEEE Signal Processing Magazine.
13. Тополь, Е. (2019). Глибока медицина: як штучний інтелект може зробити охорону здоров'я знову людською. Basic Books.
14. Ванг, Д., Хосла, А., Гарджія, Р., Іршад, Х., і Бек, А. Х. (2016). Глибоке навчання для виявлення метастатичного раку грудей.
15. Голдштейн, Б. А., Навар, А. М., Картер, Р. Е., і Блекер, С. (2017). Застосування машинного навчання для прогнозування серцево-судинних захворювань і загальної смертності у пацієнтів з діабетом. Diabetes Care.
16. Райкомар, А., Дін, Ж., і Кохейн, І. (2019). Машинне навчання в медицині. New England Journal of Medicine.
17. Чой, Е., Бахадорі, М. Т., Шютц, А., Стюарт, В. Ф., і Сун, Дж. (2017). Doctor AI: прогнозування клінічних подій за допомогою рекурентних нейронних мереж. Journal of Biomedical Informatics.
18. Дарсі, А. М., Луї, А. К., Робертс, Л. В., і Різзо, А. С. (2016). Машинне навчання та професія лікаря. JAMA.
19. Арбабшірані, М. Р., Форнвалт, Б. К., Монгеллуццо, Г. Дж., Сувер, Дж. Д., Гейз, Б. Д., Патель, А. А. та ін. (2018). Розширене навчання на дії в дії: ідентифікація інтранасального кровотечі на комп'ютерних томографічних знімках голови з інтеграцією клінічних робочих процесів.
20. Бім, А. Л., і Кохейн, І. С. (2018). Переклад штучного інтелекту в клінічну практику.
21. Обермайер, З., Пауерс, Б., Вогелі, С., і Муллайнатан, С. (2019). Розкладання расового упередження в алгоритмі, що використовується для управління здоров'ям популяцій. Science.
22. Шікел, Б., Тіге, П. Дж., Біхорак, А., і Рашиді, П. (2018). Глибоке ЕГП: огляд останніх досягнень в глибокому навчанні для аналізу електронних медичних записів (ЕМЗ). IEEE Journal of Biomedical and Health Informatics.

23. Естева, А., Робікве, А., Рамсундар, Б., Кулешов, В., Де Прісто, М., Чоу, К. та ін. (2019). Керівництво по глибокому навчанню в охороні здоров'я. Nature Medicine.

24. Янг, П., Джіанг, Й., Лонг, І., Лі, Д., і Чен, Ш. Г. (2019). Трансформація клінічних записів у структуровані медичні дані: підхід глибокого навчання. Computers in Biology and Medicine.

25. Джонсон, А. Е., Поллард, Т. Дж., Шен, Л., Леман, Л. Х., Фенг, М., Гхассемі, М., Селі, Л. А. (2016). MIMIC-III, вільна доступна база даних інтенсивної терапії. Scientific Data.

26. Чо, С., і Кім, К. (2018). Огляд дослідження глибокого навчання для медичних зображень. Journal of the Korean Society of Radiology.

27. Раджпуркар, П., Ірвін, Дж., Бол, Р. Л., Жу, К., Янг, Б., Мета, Х., Лунгрен, М. П. (2018). Глибоке навчання для діагностики рентгенівського діагнозу: ретроспективний порівняльний аналіз алгоритму CheXNeXt з практикуючими радіологами.

28. Каруана, Р., Лу, І., Герке, Дж., Гомс, Дж., Хсу, Дж., Нгуєн, Дж., Кіонг, Х. (2015). Розумні моделі для охорони здоров'я: прогнозування ризику пневмонії та госпіталізації в 30 днів. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

29. Гульшан, В., Пенг, Л., Корам, М., Стампе, М. К., Ву, Дж., Нараяна-свами, А., Вебстер, Дж. Р. (2016). Розробка та перевірка алгоритму глибокого навчання для виявлення діабетичної ретинопатії на рентгенівських фотографіях сітківки. JAMA.

30. Їе, Дж. (2018). Глибоке навчання в аналізі медичних зображень. IEEE Signal Processing Magazine.

ДОДАТКИ

		Андрусенко О.М.			Кім.КР.Б – 123 – 02 – ПЗ	
		Бродський Ю. Б.				
		№ докум.	Підпис			93

Додаток А – Код розгортання інфраструктури системи

```
//Provider settings
provider "aws" {
  access_key = "*****"
  secret_key = "*****"
  region     = "us-east-1"
}

//VPC
resource "aws_vpc" "main-system-vpc" {
  cidr_block = "10.1.0.0/18"
  tags = {
    Name = "Main system VPC"
  }
}

//Public subnets for ALB
resource "aws_subnet" "alb-public-subnet-1" {
  vpc_id            = aws_vpc.main-system-vpc.id
  cidr_block        = "10.1.1.0/28"
  availability_zone = "us-east-1a"
  map_public_ip_on_launch = "true"
  tags = {
    Name = "alb-public-subnet-1"
  }
}

resource "aws_subnet" "alb-public-subnet-2" {
  vpc_id            = aws_vpc.main-system-vpc.id
  cidr_block        = "10.1.2.0/28"
  availability_zone = "us-east-1b"
  map_public_ip_on_launch = "true"
  tags = {
    Name = "alb-public-subnet-2"
  }
}

//Private subnets fro servers
resource "aws_subnet" "servers-private-subnet-1" {
  vpc_id            = aws_vpc.main-system-vpc.id
  cidr_block        = "10.1.3.0/24"
  availability_zone = "us-east-1a"
  tags = {
    Name = "servers-private-subnet-1"
  }
}

resource "aws_subnet" "servers-private-subnet-2" {
  vpc_id            = aws_vpc.main-system-vpc.id
  availability_zone = "us-east-1b"
  cidr_block        = "10.1.4.0/24"
  tags = {
    Name = "servers-private-subnet-2"
  }
}

//Internet gateway
resource "aws_internet_gateway" "system-igw" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "Main internet gateway"
  }
}

//Elastic IP
resource "aws_eip" "nat-eip" {
  domain = "vpc"
  tags = {
    Name = "Elastic IP for NAT gateway"
  }
}

//NAT gateway
resource "aws_nat_gateway" "nat-gw" {
  allocation_id = aws_eip.nat-eip.id
  subnet_id     = aws_subnet.alb-public-subnet-1.id
  tags = {
    Name = "NAT gateway"
  }
}

//Edit default RT
resource "aws_default_route_table" "def-rt" {
  default_route_table_id = aws_vpc.main-system-vpc.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.system-igw.id
  }
}

//Route tables associations (public)
resource "aws_route_table_association" "alb-pub-1-rt" {
  subnet_id      = aws_subnet.alb-public-subnet-1.id
  route_table_id = aws_vpc.main-system-vpc.default_route_table_id
}

resource "aws_route_table_association" "alb-pub-2-rt" {
  subnet_id      = aws_subnet.alb-public-subnet-2.id
  route_table_id = aws_vpc.main-system-vpc.default_route_table_id
}

//Route table (NAT)
resource "aws_route_table" "nat-rt" {
  vpc_id = aws_vpc.main-system-vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_nat_gateway.nat-gw.id
  }
  tags = {
    Name = "NAT route table"
  }
}

//Route tables associations (private)
resource "aws_route_table_association" "servers-priv-1-rt" {
  subnet_id      = aws_subnet.servers-private-subnet-1.id
  route_table_id = aws_route_table.nat-rt.id
}

resource "aws_route_table_association" "servers-priv-2-rt" {
  subnet_id      = aws_subnet.servers-private-subnet-2.id
  route_table_id = aws_route_table.nat-rt.id
}

//Security group for ALB
resource "aws_security_group" "alb-sg" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "ALB security group"
  }
}

resource "aws_security_group_rule" "inbound-alb" {
  type            = "ingress"
  from_port       = 80
  to_port         = 80
  protocol        = "tcp"
  cidr_blocks     = ["0.0.0.0/0"]
  security_group_id = aws_security_group.alb-sg.id
}

resource "aws_security_group_rule" "outbound-alb" {
  type            = "egress"
  from_port       = 80
  to_port         = 80
  protocol        = "tcp"
  security_group_id = aws_security_group.alb-sg.id
  source_security_group_id =
aws_security_group.instance-sg.id
}
```

```

//Security group for instances
resource "aws_security_group" "instance-sg" {
  vpc_id = aws_vpc.main-system-vpc.id
  tags = {
    Name = "Instance security group"
  }
}
resource "aws_security_group_rule" "inbound-
instance" {
  type = "ingress"
  from_port = 80
  to_port = 80
  protocol = "tcp"
  security_group_id = aws_security_group.instance-
sg.id
  source_security_group_id =
aws_security_group.alb-sg.id
}
resource "aws_security_group_rule" "outbound-
instance" {
  type = "egress"
  from_port = 0
  to_port = 0
  protocol = "-1"
  cidr_blocks = ["0.0.0.0/0"]
  security_group_id = aws_security_group.instance-
sg.id
}
//Launch template for servers
resource "aws_launch_template" "server-template" {
  name_prefix = "server"
  image_id = "ami-018ba43095ff50d08"
  iam_instance_profile {name = "server-role"}
  instance_type = "t2.micro"
  vpc_security_group_ids =
["${aws_security_group.instance-sg.id}"]
  user_data =
filebase64("install.sh")
}
//Target group for ALB
resource "aws_lb_target_group" "servers-tg" {
  name = "processing-servers"
  port = 80
  protocol = "HTTP"
  target_type = "instance"
  vpc_id = aws_vpc.main-system-vpc.id
}
# //Auto Scaling Group
resource "aws_autoscaling_group" "servers-
autoscaling-group" {
  desired_capacity = 2
  max_size = 4
  min_size = 2
  vpc_zone_identifier = [aws_subnet.servers-
private-subnet-1.id, aws_subnet.servers-private-
subnet-2.id]
  launch_template {
    id = aws_launch_template.server-
template.id
    version = "$Latest"
  }
}
//ASG attachment to target group
resource "aws_autoscaling_attachment" "asg-attach"
{
  autoscaling_group_name =
aws_autoscaling_group.servers-autoscaling-group.id
  lb_target_group_arn =
aws_lb_target_group.servers-tg.arn
}
//Application Load Balancer
resource "aws_lb" "main-alb" {
  name = "main-alb"
  internal = false
  load_balancer_type = "application"

  security_groups = [aws_security_group.alb-
sg.id]
  subnets = [aws_subnet.alb-public-
subnet-1.id, aws_subnet.alb-public-subnet-2.id]
  enable_cross_zone_load_balancing = "true"
  enable_deletion_protection = "false"
  tags = {
    Name = "Main load balancer"
  }
}
resource "aws_lb_listener" "main-alb" {
  load_balancer_arn = aws_lb.main-alb.arn
  port = "80"
  protocol = "HTTP"
  default_action {
    type = "forward"
    target_group_arn =
aws_lb_target_group.servers-tg.arn
  }
}
//VPC endpoint
resource "aws_vpc_endpoint" "s3-endpoint" {
  vpc_id = aws_vpc.main-system-vpc.id
  service_name = "com.amazonaws.us-east-1.s3"
  vpc_endpoint_type = "Gateway"
  route_table_ids = [aws_route_table.nat-rt.id]
}
//S3 buckets
resource "aws_s3_bucket" "backups-bucket" {
  bucket = "backup-bucket-aom"
  tags = {
    Name = "Bucket for backups."
  }
}
resource "aws_s3_bucket" "sagemaker-files" {
  bucket = "sagemaker-bucket-aom"
  tags = {
    Name = "Bucket for SageMaker usage."
  }
}
resource "aws_s3_object" "train-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key = "train/"
}
resource "aws_s3_object" "test-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key = "test/"
}
resource "aws_s3_object" "result-folder" {
  bucket = aws_s3_bucket.sagemaker-files.id
  key = "result/"
}
resource "aws_s3_bucket" "healthimaging-files" {
  bucket = "healthimaging-bucket-aom"
  tags = {
    Name = "Bucket for HealthImaging
usage."
  }
}
resource "aws_s3_bucket" "comprehendmed-files" {
  bucket = "comprehendmedical-bucket-aom"
  tags = {
    Name = "Bucket for ComprehendMedical
usage."
  }
}
resource "aws_s3_object" "output-cm-folder" {
  bucket = aws_s3_bucket.comprehendmed-files.id
  key = "output/"
}

```