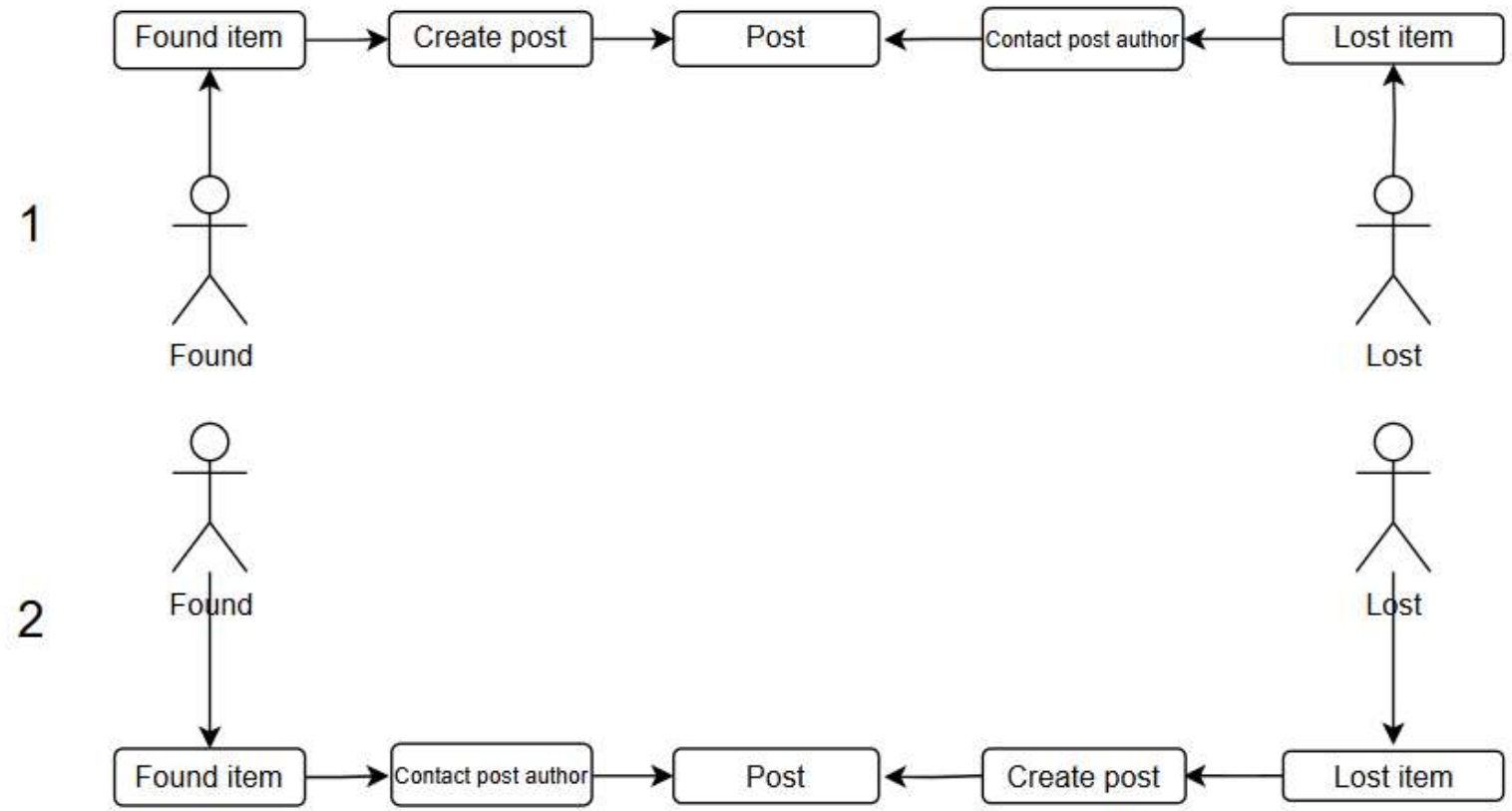
Побудова мережевого застосування з високою доступністю на хмарній платформі

Виконав студент ІПЗ-2: Кириченко Євгеній

Керівник: кандидат технічних наук Черкасов Д.І.

Цифрове бюро знахідок на AWS



Завдання

- Аналіз аналогів
- Огляд компонентів багатошарової архітектури
- Огляд обчислювльних сервісів AWS
- Огляд інструментарію AWS
- Проектування схеми, яка задовольняє вимогам високої доступності та масштабованості
- Реалізація окремих компонентів системи

Аналіз аналогів

verni.com.ua
Всеукраїнське бюро знахідок «Поверни!»

poisk ua

Огляд компонентів веб-архітектур

- DNS
- CDN
- Load Balancer
- API Gateway
- Cache DB
- MQ
- Моніторинг та логування
- Firerwall

Стратегії досягнення високої доступності

- Надлишковість та реплікація
- Балансування навантаження
- Кластер відновення
- Розподілене сховище даних
- Моніторинг стану систем та оповіщення
- Регулярне обслуговування та оновлення системи
- Географічний розподіл

Фундаментальні моделі хмарних обчислень

- IaaS - інфраструктура як сервіс
- CaaS - контейнери як сервіс
- FaaS - функції як сервіс

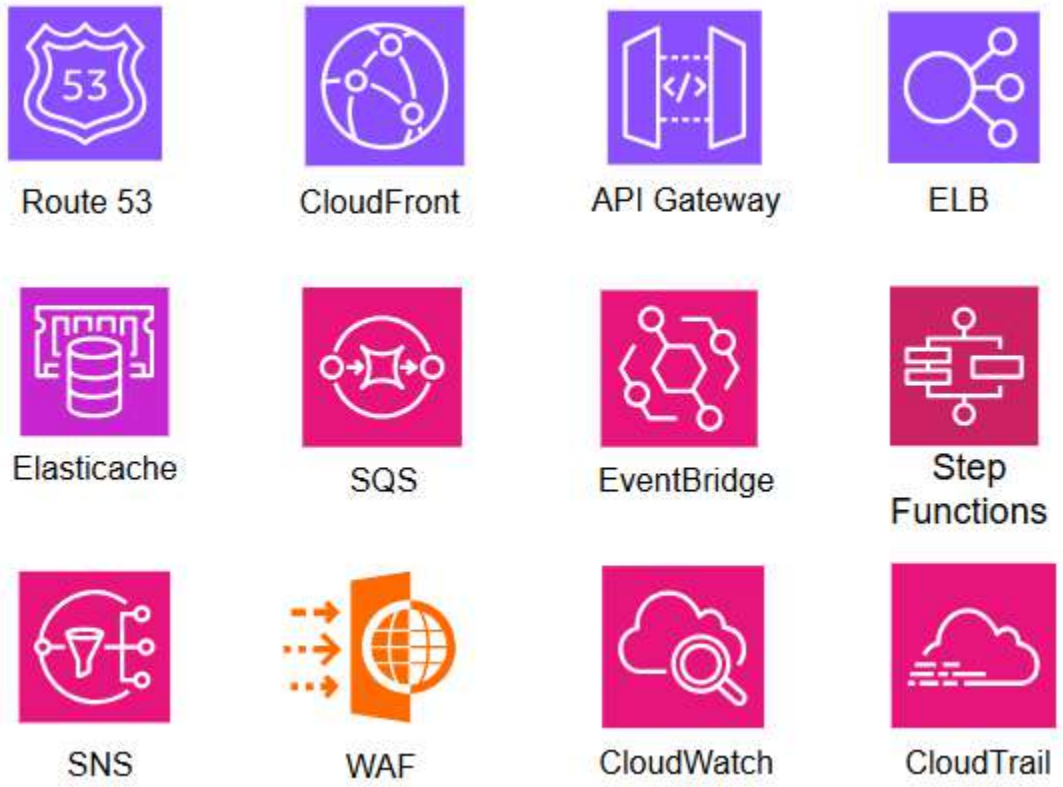
Порівняння CaaS рішень в AWS

Сервіс	Керування серверами	Гнучкість	Складність використання	Ціна
ECS + EC2	Так	Висока	Середня	Низька
EKS + EC2	Так	Дуже висока	Висока	Середня
ECS/EKS + Fargate	Ні	Низька/Трохи вища	Низька/Трохи вища	Вища ніж з EC2

EC2, ECS, Lambda

Характеристика	EC2	ECS (на EC2/Fargate)	Lambda
Модель обчислень	IaaS	CaaS	FaaS
Гнучкість	Найбільша	Досить висока / Досить низька	Найменша
Масштабування	Користувач налаштовує	Користувач налаштовує 2 рівні масштабування / тільки контейнери	Автоматичне без участі користувача
Оплата	За час роботи	За EC2 / виділені ресурси	За кількість викликів, CPU і пам’ять
Сценарії використання	Моноліти, великі сервіси	Мікросервіси, подієві сервіси	Подієві, короткі або рідко виконувані задачі
Інтеграція з AWS	Потребує ручної конфігурації	Краща, ніж в EC2 (на рівні контейнерів)	Ідеальна інтеграція
Тривалість виконання	Необмежена	Необмежена / залежить від задачі	До 15 хв на 1 виконання
Холодний старт	Відсутній	Відсутній	Є, після 15 хв паузи
Ціна	Дешево при постійному навантаженні, дорого при простої	Аналогічно EC2 / дорожче за зручність	Дешево при нечастому навантаженні, дорого при сталому

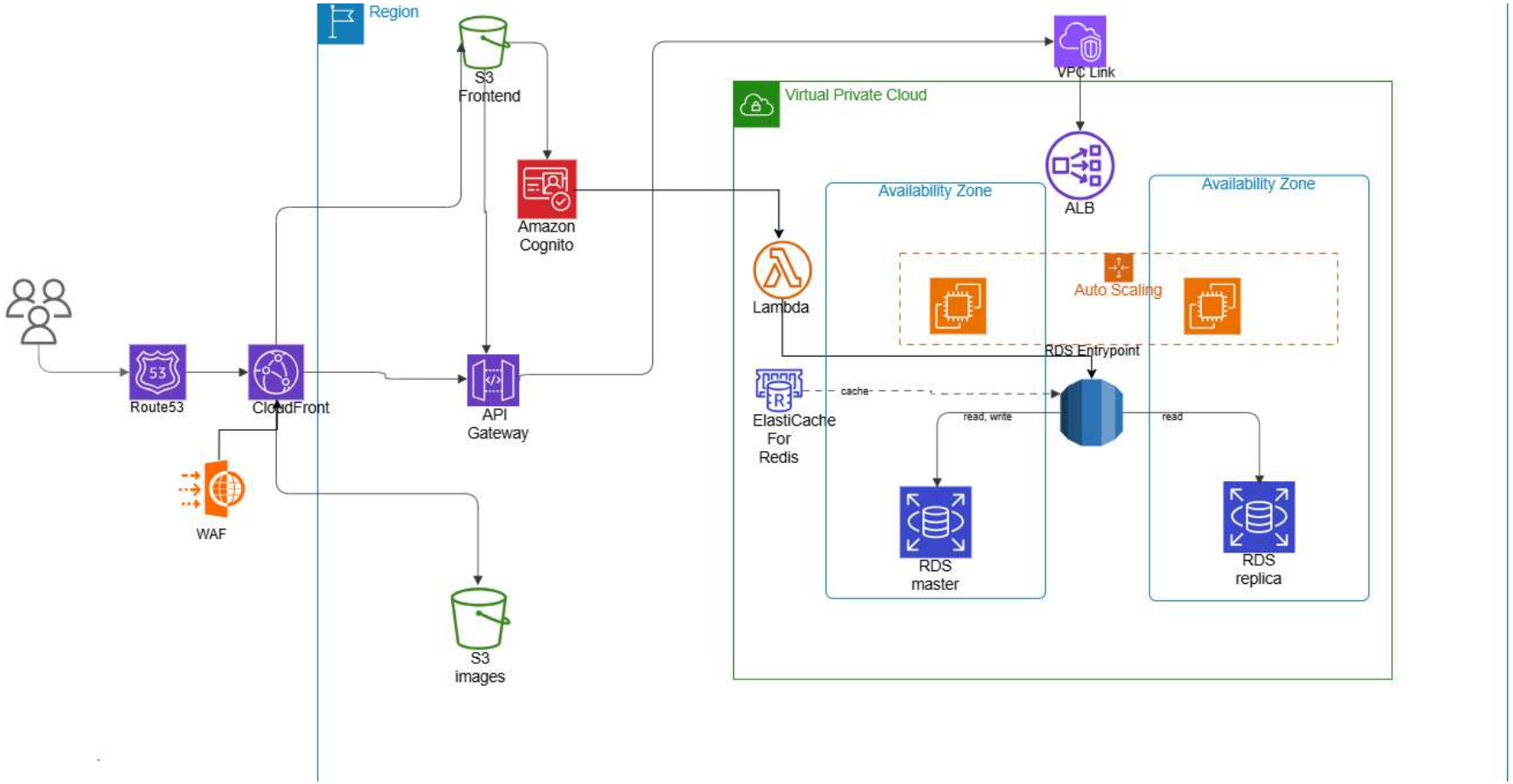
Огляд інструментарію AWS



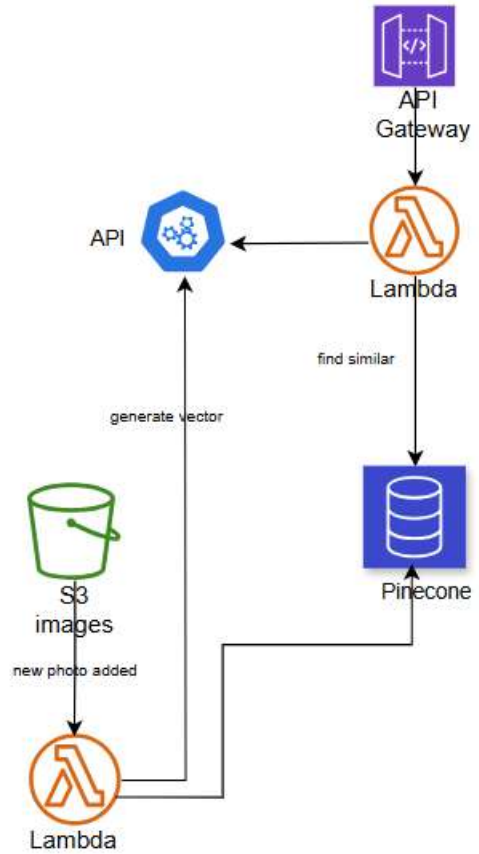
Технічне завдання: додаткові функції системи

- Повідомлення про схожі знахідки
- Пошук схожих знахідок по фото
- Автоматична очистка застарілих даних

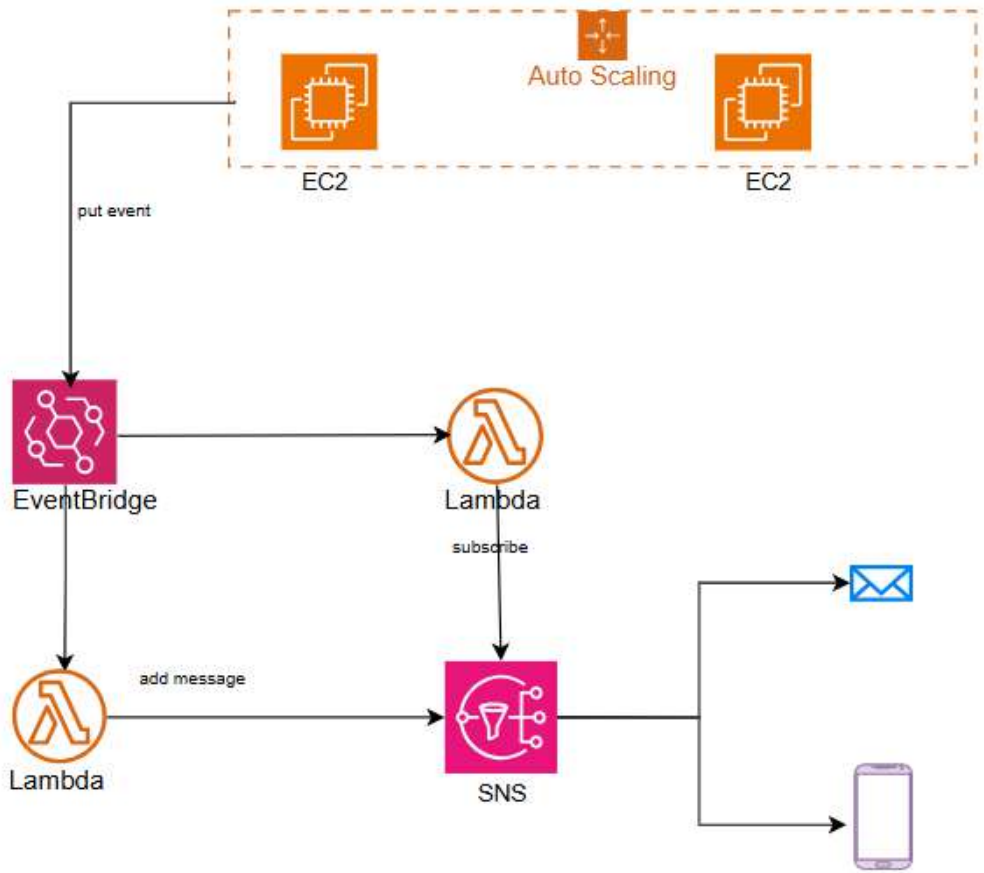
Базова архітектура



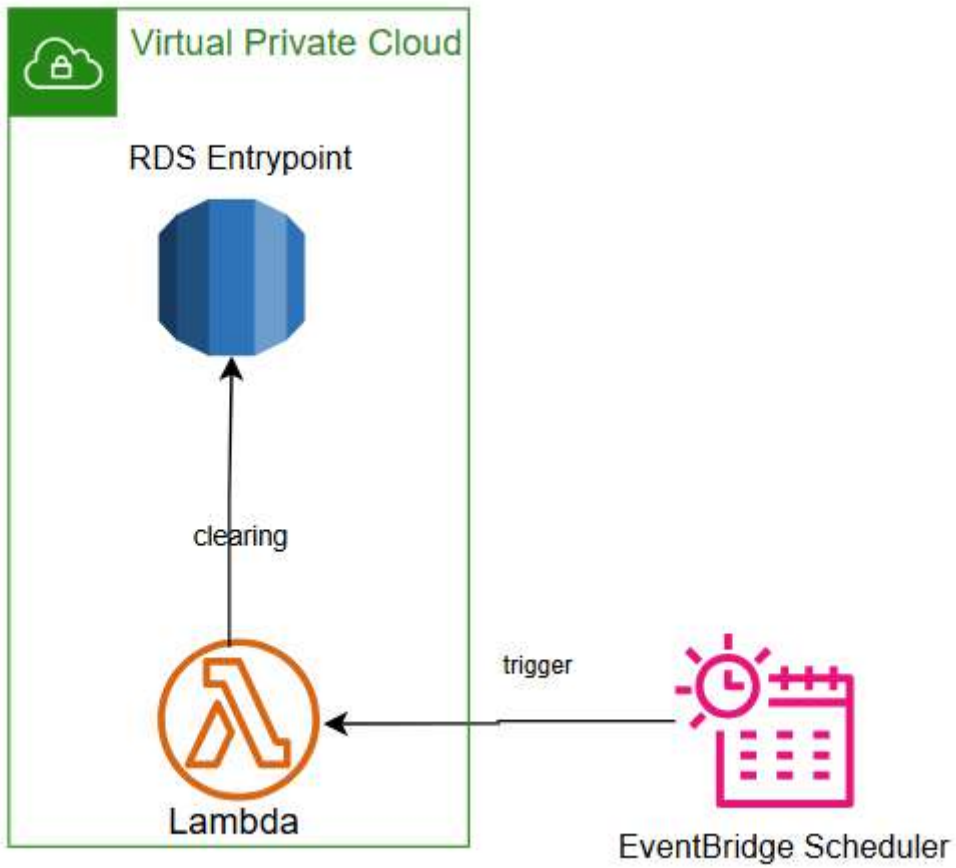
Пошук схожих знахідок по фото



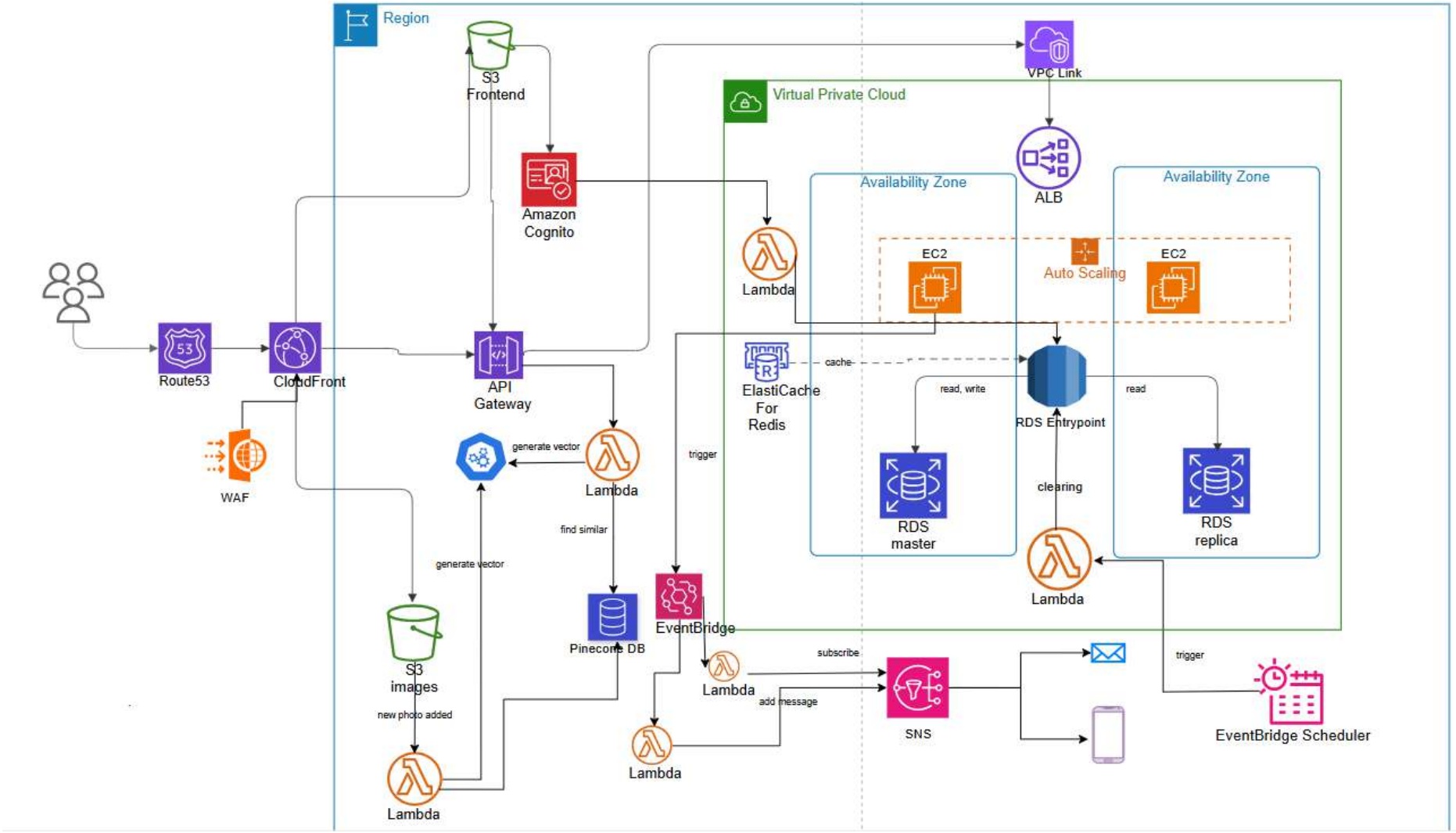
Нотифікація користувачів



Очистка постів



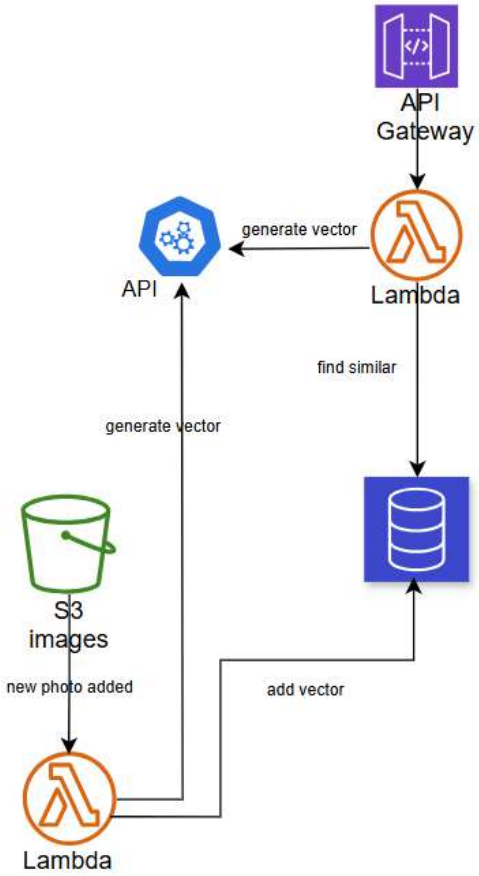
Повна архітектура



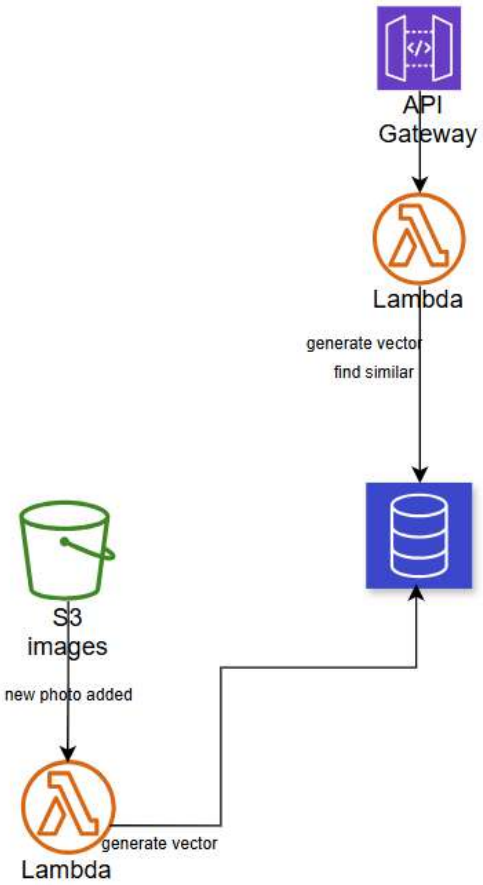
Сценарії роботи системи

Операції	Сервіси
CRUD для постів про знахідки	EC2
Очистка знайдених та застарілих постів	Event Bridge + Lambda
Пошук схожих фотографій	S3 trigger + Lambda + Pinecone
Нотифікація користувачів про знахідку схожої речі	Event Bridge + Lambda + SNS

Реалізація пошуку за зображенням



В схемі



При реалізації

Реалізація пошуку за зображенням

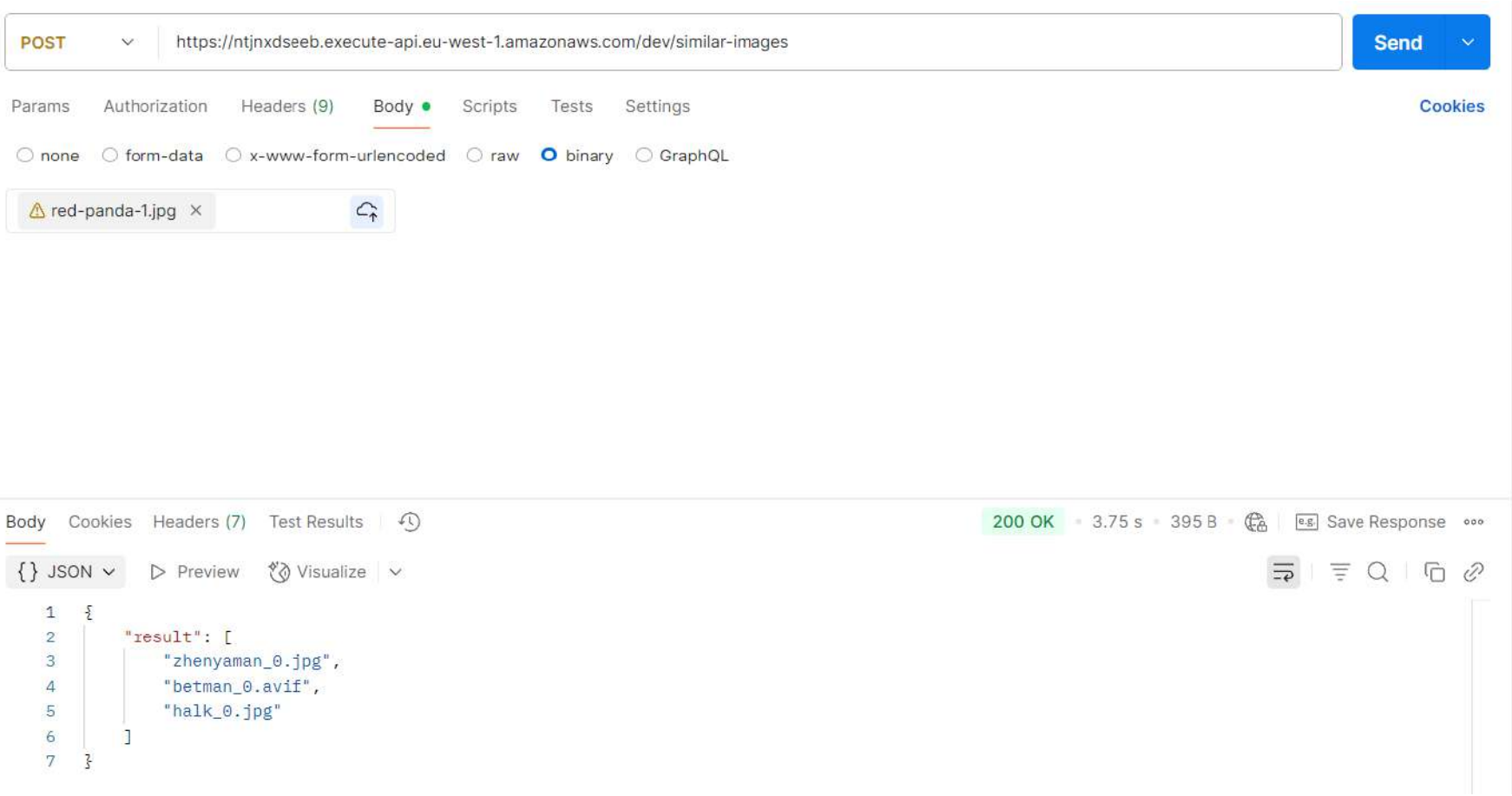
```
await index.upsert([{
    id: key,
    values: vector,
    metadata: metadata
}]);
```

Додавання вектора

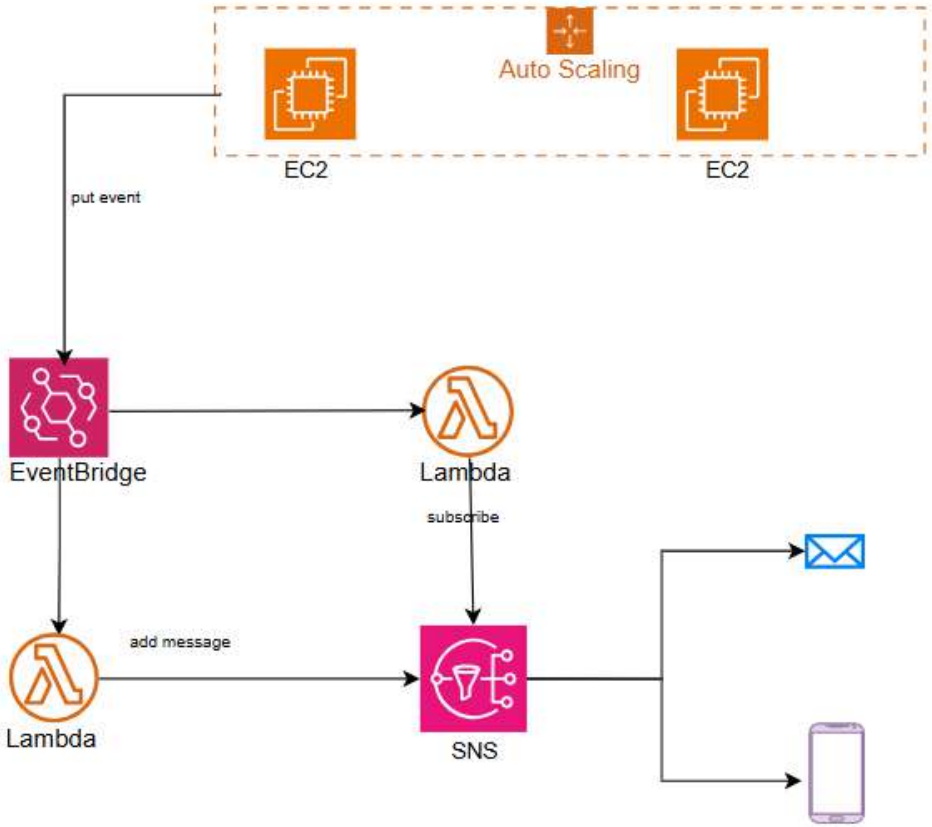
```
const resp = await index.query({
    vector: queryVector,
    topK: 3,
    includeValues: false,
    includeMetadata: true,
})
```

Пошук схожих векторів

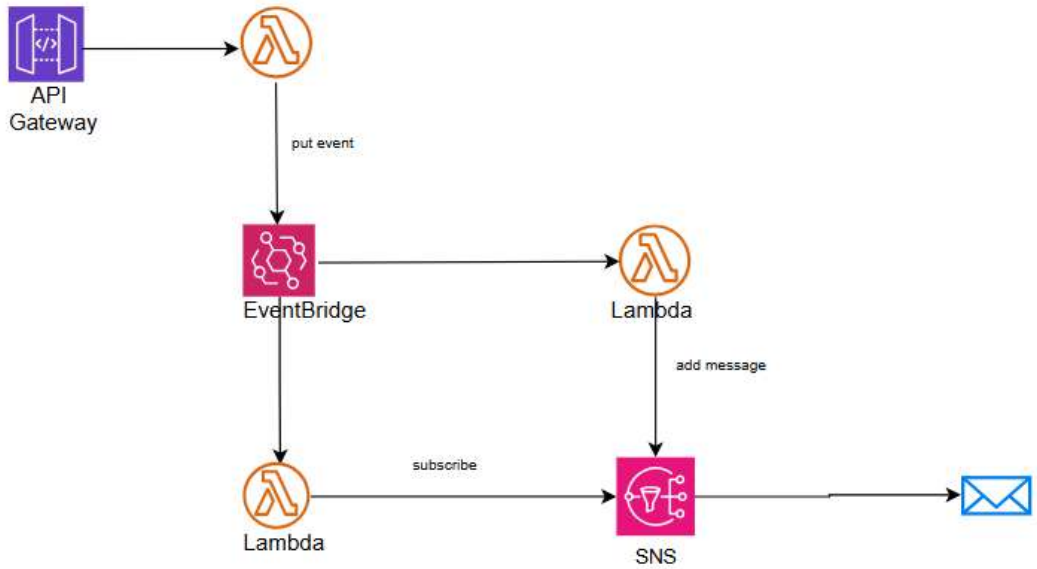
Демонстрація



Реалізація нотифікації користувачів



В схемі



При реалізації

Реалізація нотифікації користувачів

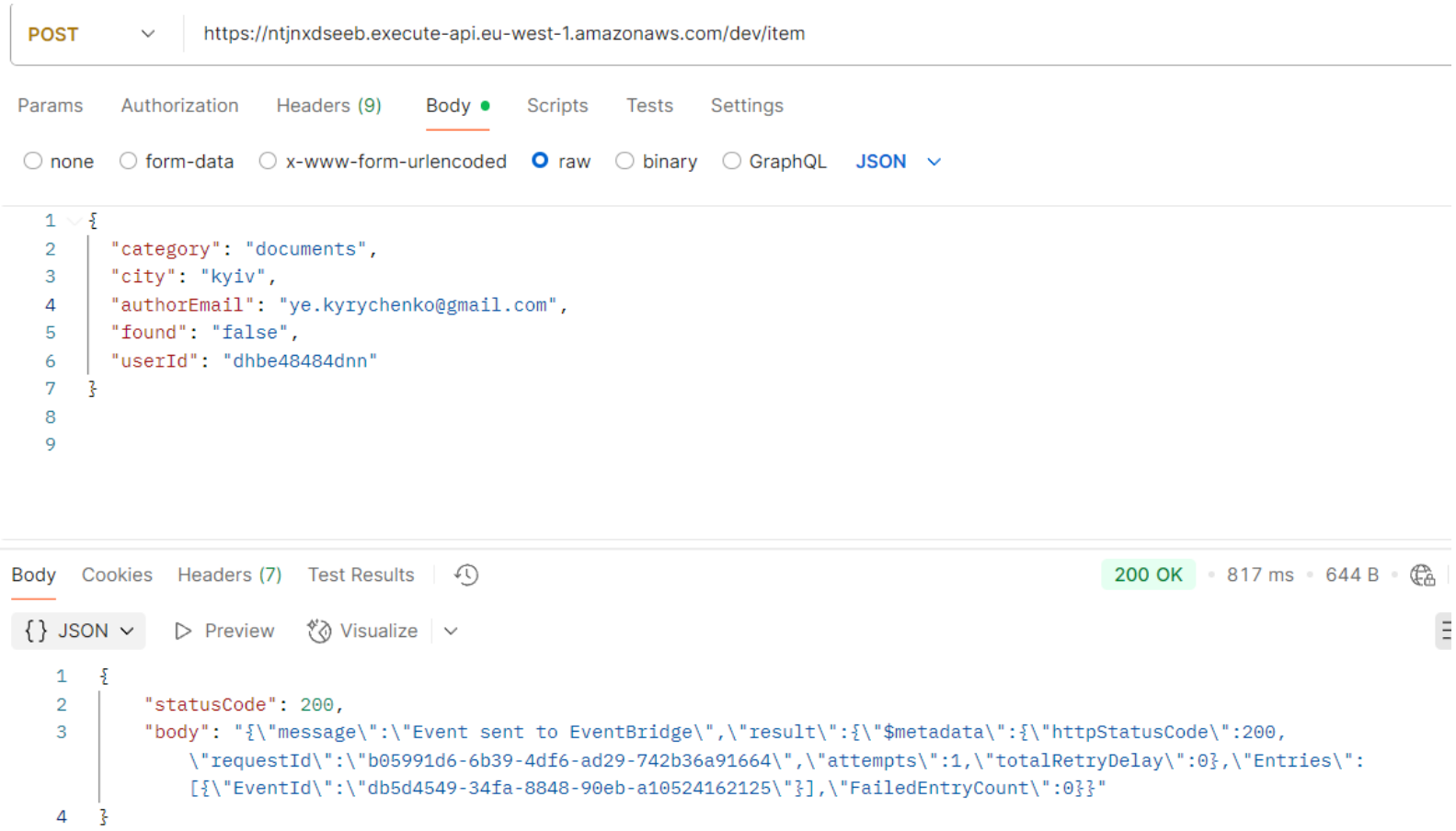
```
const body = event.detail;
const { category, city, authorEmail } = body;

const topicName = `lost-${category}-${city}`;
const topicArn = `arn:aws:sns:${process.env.AWS_REGION}:
${process.env.AWS_ACCOUNT_ID}:${topicName}`;

const message = `Someone may have found your lost item in ${city}.
Contact: ${authorEmail}`;

console.log("Publishing to SNS", { topicArn, message });
await sns.send(new PublishCommand({
  TopicArn: topicArn,
  Message: message,
  Subject: "Possible Match Found"
}));
```

Публікація в топік

```
const { category, city, authorEmail } = event.detail;

const topicName = `lost-${category}-${city}`.toLowerCase();

const { TopicArn } = await sns.send(
  new CreateTopicCommand({ Name: topicName })
);

await sns.send(
  new SubscribeCommand({
    TopicArn,
    Protocol: "email",
    Endpoint: authorEmail
  })
);
```

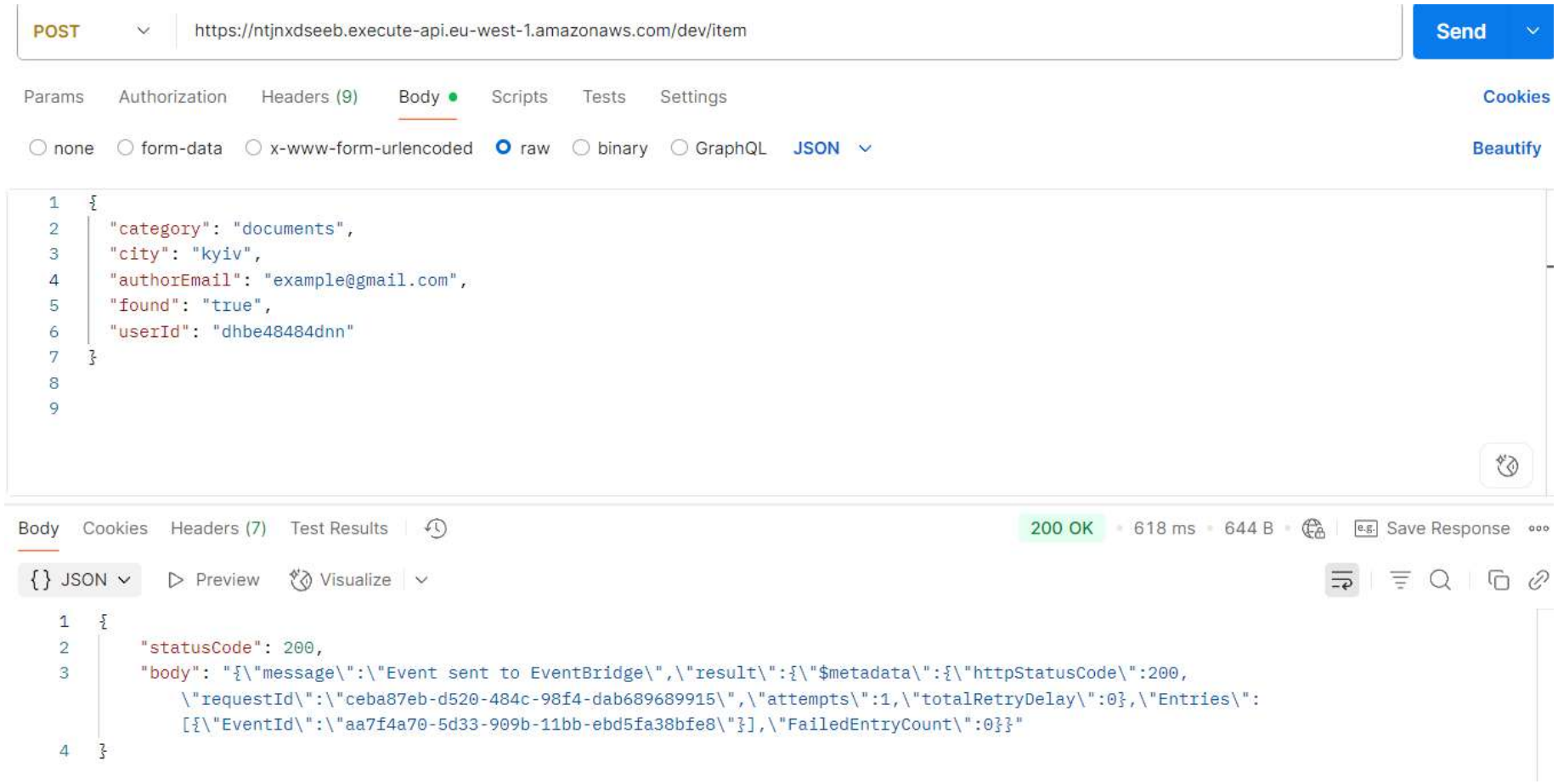
Генерація топіка та підписка на нього

Демонстрація



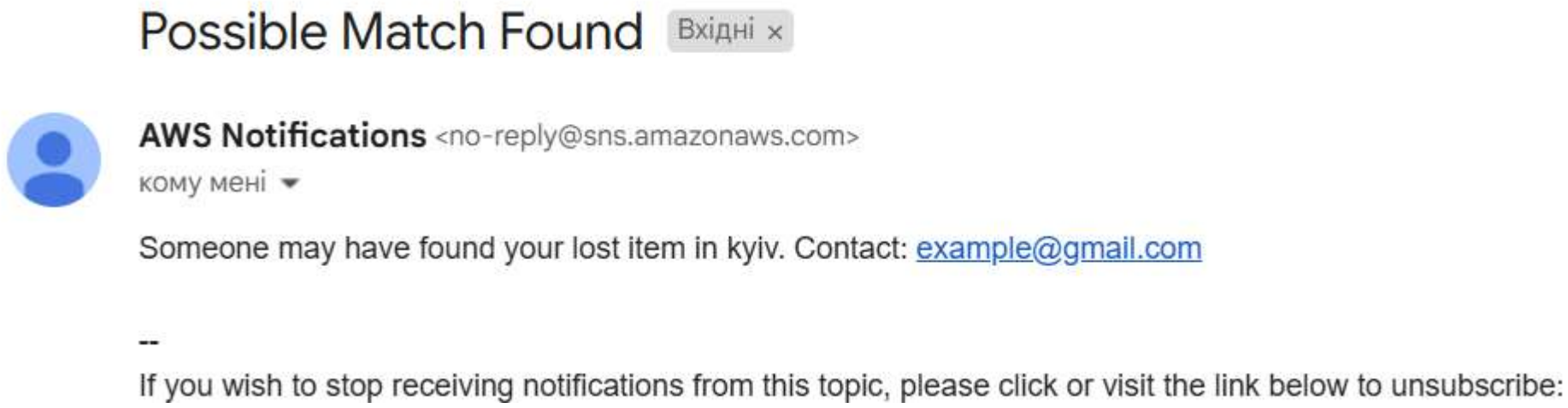
Запит на створення поста про втрату

Демонстрація



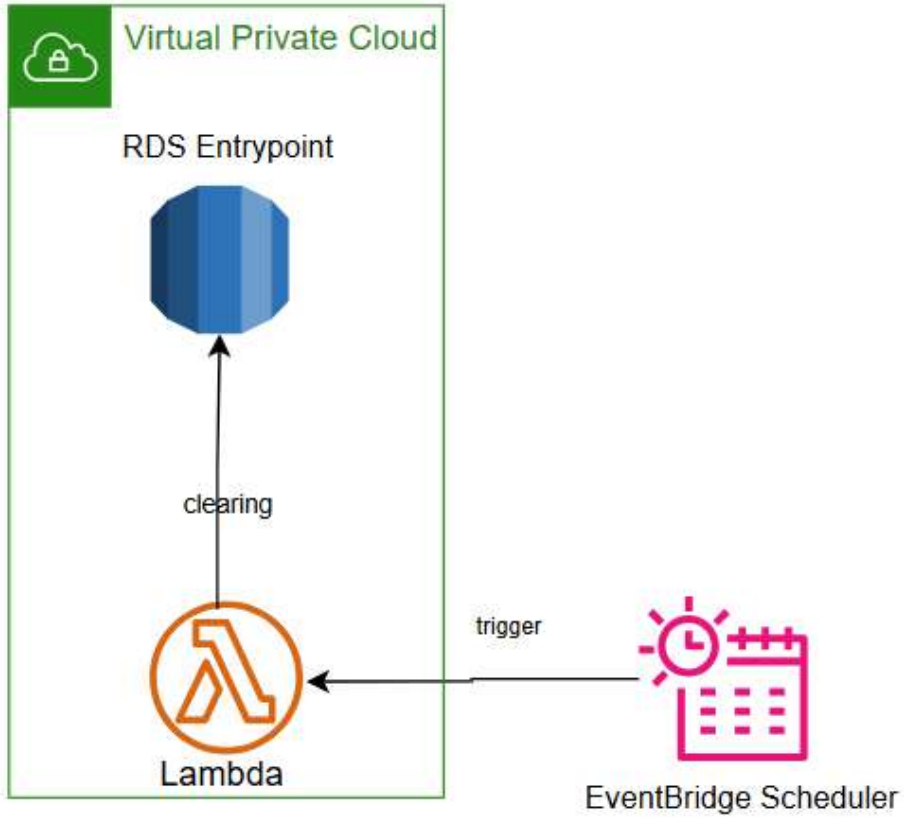
Запит на створення поста про знахідку

Демонстрація

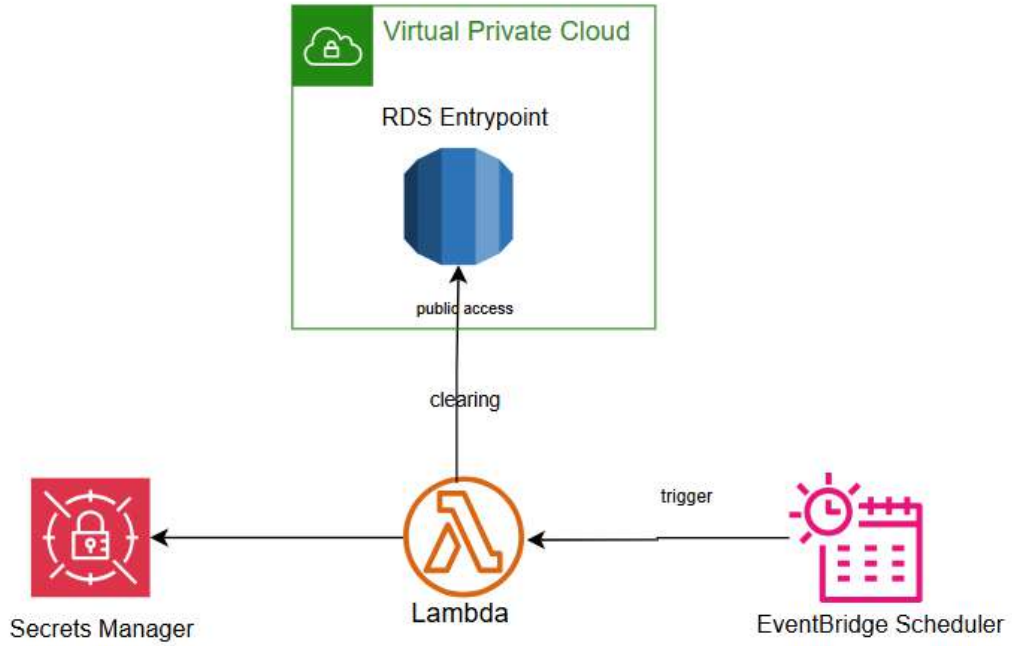


Нотифікація про появу схожої знахідки

Очистка бази даних за розкладом



В схемі



При реалізації

Очистка бази даних за розкладом

```
await appClient.query(`
  CREATE TABLE IF NOT EXISTS posts (
    id SERIAL PRIMARY KEY,
    title TEXT NOT NULL,
    created_at TIMESTAMP NOT NULL DEFAULT NOW(),
    close BOOLEAN NOT NULL DEFAULT false
  );
`);
```

Створення таблиці

```
const posts = await appClient.query(`
  DELETE FROM posts
  WHERE created_at < NOW() - INTERVAL '4 years'
    OR close = true;
`);
```

Очищення таблиці

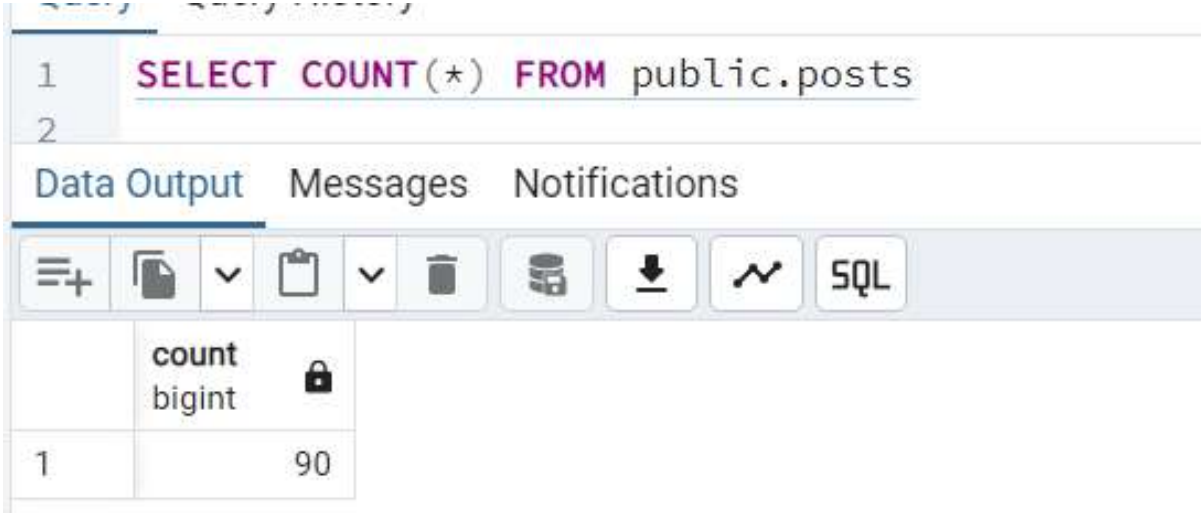
```
for (let i = 1; i <= 100; i++) {
  let title = `Post #${i}`;
  let createdAtClause;
  let closeFlag = false;

  if (i <= 5) {
    // старіші за 4 роки
    createdAtClause = `NOW() - INTERVAL '4 years'`;
  } else if (i <= 10) {
    // закриті
    createdAtClause = `NOW()`;
    closeFlag = true;
  } else {
    createdAtClause = `NOW()`;
  }

  await appClient.query(
    `
      INSERT INTO posts (title, created_at, close)
      VALUES ($1, ${createdAtClause}, $2)
    `,
    [title, closeFlag]
  );
```

Заповнення таблиці

Демонстрація

```
2025-06-02T13:40:12.862Z        5d683da9-b483-4d56-9ed3-67c70cec5807    INFO    Result {
  command: 'DELETE',
  rowCount: 10,
```

Запис в логах лямбди про видалення десяти рядків



Кількість постів після очистки

Висновки

- Проаналізовано аналоги
- Здійснено огляд компонентів веб-архітектур
- Здійснено огляд обчислювальних сервісів AWS
- Здійснено огляд AWS інструментарію
- Спроектовано високодоступну масштабовану архітектуру
- Описано взаємодію компонентів та сценарії роботи системи
- Реалізовано пошук по фото, нотифікацію, очистку бази даних

Дякую за увагу!